

BỘ XÂY DỰNG  
TRƯỜNG ĐẠI HỌC KIẾN TRÚC HÀ NỘI

---

GIÁO TRÌNH  
**AN TOÀN VÀ BẢO MẬT HỆ THỐNG THÔNG TIN**

HÀ NỘI, 2021

## Mục lục

<b>CHƯƠNG 1. TỔNG QUAN VỀ AN TOÀN BẢO MẬT HTTT.....</b>	<b>6</b>
1.1 . Ba thành phần cơ bản.....	6
1.1.1. Tính bí mật.....	7
1.1.2. Tính toàn vẹn .....	8
1.1.3. Tính khả dụng .....	9
1.2. Các mối đe dọa.....	10
1.2.1. Các nguy cơ từ nhân tố con người.....	11
1.2.2 Các nguy cơ từ nhân tố kỹ thuật .....	13
1.2.3 Các nguy cơ khác .....	15
1.3. Chính sách và cơ chế .....	16
1.3.1. Chính sách bảo mật .....	16
1.3.2. Cơ chế bảo mật .....	17
1.4. Vấn đề con người.....	19
<b>CHƯƠNG 2. ĐIỀU KHIỂN TRUY CẬP .....</b>	<b>20</b>
2.1. Mục tiêu của điều khiển truy cập.....	20
2.2. Các kiểu xác thực.....	20
2.2.1. Xác thực dựa trên những gì người sử dụng biết .....	21
2.2.2. Xác thực dựa trên những gì người sử dụng có.....	26
2.2.3. Xác thực dựa trên những gì người sử dụng sở hữu bẩm sinh .....	37
2.3. Quản trị mật khẩu.....	49
2.4. Chính sách điều khiển truy cập.....	50
2.4.1. Thẩm quyền/cấp phép .....	50
2.4.2 Thực hiện .....	50
2.4.3 Hoạt động.....	51
2.5. Phương pháp điều khiển truy cập .....	51
2.5.1 Mô hình điều khiển truy xuất bắt buộc (Mandatory Access Control _ MAC):.....	51
2.5.2 Mô hình điều khiển truy xuất tự do (Discretionary Access Control _ DAC) .....	52
Mô hình điều khiển truy xuất tự do .....	52
2.5.3 Mô hình điều khiển truy xuất theo chức năng (Role Based Access Control _ RBAC).....	52
2.6 Các kiểu tấn công.....	53

<b>CHƯƠNG 3: CHÍNH SÁCH AN NINH .....</b>	<b>53</b>
3.1. Các khái niệm .....	53
3.2. Các kiểu chính sách an ninh (Tìm hiểu thêm) .....	54
3.3. Chính sách bảo mật (Tìm hiểu thêm).....	54
3.4. Chính sách toàn vẹn (Tìm hiểu thêm).....	54
<b>CHƯƠNG 4: MÃ HÓA CĂN BẢN.....</b>	<b>55</b>
4.1. Tổng quan .....	55
4.2. Mã hóa khóa bí mật.....	56
4.2.1 Các hệ mật thay thế đơn biểu.....	56
<b>4.2.2 Các phép thay thế đơn giản khác .....</b>	<b>62</b>
A. CÁC HỆ MẬT THAY THẾ ĐA BIỂU .....	63
1. Hệ mật Vigenere.....	63
2. Hệ mật Hill.....	64
3. Hệ mật Playfair .....	69
B. CÁC HỆ MẬT THAY THẾ KHÔNG TUẦN HOÀN .....	71
1. Hệ mật khóa tự sinh.....	71
2. Hệ mật Vernam .....	72
C. CÁC HỆ MẬT HOÁN VỊ .....	73
D. CÁC HỆ MẬT TÍCH .....	74
E. CHUẨN MÃ DỮ LIỆU (DES).....	77
1. Mở đầu .....	77
2. Mô tả DES.....	77
3. Một ví dụ về DES .....	85
4. Một số ý kiến thảo luận về DES .....	88
5. Các tính chất và sức mạnh của DES: .....	90
6. Các chế độ hoạt động của DES .....	91
7. Một số biến thể của DES.....	94
8. Thám mã vi sai và thám mã tuyến tính.....	97
F. CHUẨN MÃ DỮ LIỆU TIỀN TIẾN (AES) .....	104
1. Cơ sở toán học của AES .....	105
2. Thuật toán AES .....	105
<b>4.2.3 ƯU NHƯỢC ĐIỂM CỦA CÁC HỆ MẬT KHÓA BÍ MẬT</b> .....	<b>108</b>
4.3. Mã hóa khóa công khai .....	109
<b>4.3.1 GIỚI THIỆU VỀ MẬT MÃ KHÓA CÔNG KHAI</b> .....	<b>109</b>

<b>4.3.2 BÀI TOÁN PHÂN TÍCH THÙA SỐ VÀ CÁC HỆ MẬT CÓ LIÊN QUAN.....</b>	111
1. Bài toán phân tích thừa số .....	111
2.    Hệ mật RSA .....	113
3.    Hệ mật Rabin.....	116
<b>4.3.3 BÀI TOÁN LOGARIT RỜI RẠC VÀ CÁC HỆ MẬT CÓ LIÊN QUAN</b>	117
2. Hệ mật Elgamal.....	118
<b>4.3.4 BÀI TOÁN XẾP BALO VÀ HỆ MẬT MERKLE – HELLMAN .....</b>	119
4.3.4.1    Định nghĩa dây siêu tăng .....	119
4.3.4.2    Bài toán xếp balô.....	119
4.3.4.3    Giải bài toán xếp balô trong trường hợp dây siêu tăng	
119	
4.3.4.4    Thuật toán mã công khai Merkle – Hellman .....	120
<b>4.3.5 BÀI TOÁN MÃ SỬA SAI VÀ HỆ MẬT MC ELICE .....</b>	121
4.3.5.1    Định nghĩa 1.....	121
4.3.5.2    Định lý 2 .....	122
<b>4.3.6 HỆ MẬT TRÊN ĐƯỜNG CONG ELLIPTIC .....</b>	125
4.3.6.1    Các đường cong Elliptic .....	125
4.3.6.2    Các đường cong Elliptic trên trường Galois .....	125
4.3.6.3    Các phép toán cộng và nhân trên các nhóm E....	127
4.3.6.4    Mật mã trên đường cong Elliptic. ....	129
4.3.6.5    Độ an toàn của hệ mật trên đường cong Elliptic. 130	
<b>4.3.7 ƯU NHƯỢC ĐIỂM CỦA HỆ MẬT KHÓA CÔNG KHAI.....</b>	130
<b>CHƯƠNG 5: QUẢN LÝ KHÓA .....</b>	131
5.1. Khóa phiên và khóa trao đổi .....	131
5.2. Trao đổi khóa .....	132
5.2.1 Quản lý và phân phối khoá bí mật. ....	132
5.2.2 Những kỹ thuật phân phối khoá công khai. ....	134
5.3. Chữ ký số .....	145
5.3.1. <i>Khái niệm.</i> .....	145
5.3.2. <i>Chữ ký số trực tiếp.</i> .....	146
5.3.3. <i>Chữ ký số phân xử.</i> .....	147
<b>CHƯƠNG 6: QUẢN TRỊ VÀ KIỂM SOÁT .....</b>	150
6.1. Các nguyên tắc an ninh: .....	150
a.      Nguyên tắc tính hệ thống .....	150

b.	Nguyên tắc tổng thể .....	150
c.	Nguyên tắc bảo vệ liên tục .....	151
d.	Nguyên tắc đầy đủ hợp lý .....	151
e.	Nguyên tắc mềm dẻo hệ thống.....	151
f.	Nguyên tắc công khai của thuật toán và cơ chế bảo vệ ..	152
g.	Nguyên tắc đơn giản trong sử dụng .....	152
6.2.	Đánh giá rủi ro .....	152
<b>CHƯƠNG 7: PHẦN MỀM MÃ ĐỘC.....</b>		<b>154</b>
7.1	Theo hình thức lây nhiễm .....	154
7.1.1	Trap Door.....	155
7.1.2	Logic Bombs.....	155
7.1.3	Trojan Horses.....	156
7.1.4	Virus.....	157
7.1.5	Worm.....	159
7.1.6	Zombie.....	160
7.2	Phân loại của NIST160 .....	160
7.2.1	Virus.....	160
7.2.2	Worm.....	161
7.2.3	Trojan Horse .....	161
7.2.4	Malicious Mobile Code .....	161
7.2.5	Tracking Cookie .....	161
7.2.6	Phần mềm gián điệp.....	162
7.2.7	Phần mềm quảng cáo .....	162
7.2.8	Attacker Tool .....	162
7.2.9	Phishing .....	164
7.2.10	Virus Hoax.....	164

# CHƯƠNG 1. TỔNG QUAN VỀ AN TOÀN BẢO MẬT HTTT

## 1.1. Ba thành phần cơ bản

*Hệ thống thông tin là một hệ thống gồm con người, dữ liệu và những hoạt động xử lý dữ liệu và thông tin trong một tổ chức.*

Theo quan điểm Công nghệ thông tin (CNTT), hệ thống thông tin (Information Systems) là một hệ thống sử dụng CNTT để thu thập, truyền, lưu trữ, xử lý và biểu diễn thông tin trong quá trình hoạt động.

Thông tin là tập hợp các dữ liệu (các tin tức) về thế giới bao quanh chúng ta (các sự kiện, các cá nhân, các hiện tượng, các quá trình, các nhân tố và các mối liên hệ giữa chúng), được thể hiện trong dạng thức phù hợp cho việc truyền đi bởi những người này và tiếp nhận bởi những người kia, được sử dụng với mục đích thu nhận kiến thức (các tri thức) cũng như đưa ra những quyết định. Thông tin tồn tại khách quan, có thể được tạo ra, truyền đi, lưu trữ, chọn lọc. Thông tin cũng có thể bị sai lệch, méo mó do nhiều nguyên nhân khác nhau: bị xuyên tạc, cắt xén... Những yếu tố gây sự sai lệch thông tin gọi là các yếu tố nhiễu.

Dữ liệu là các sự kiện không có cấu trúc, không có ý nghĩa rõ ràng, cho đến khi chúng được tổ chức theo một tiến trình tính toán nào đó. Như vậy, thông tin là dữ liệu đã được xử lý xong, mang ý nghĩa rõ ràng.

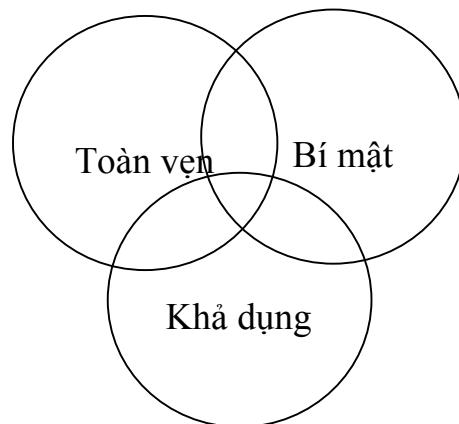
Đối với các hệ thống thông tin thì thông tin là những dữ liệu được nhập vào hệ thống và đã được xử lý sao cho những dữ liệu được xuất ra chứa đựng những thông tin thực sự có ý nghĩa đối với người sử dụng.

Ví dụ ta có dữ liệu sau: Nguyễn Văn Nam, 845102, 14/10/2002, HTTT, 9.

Với dữ liệu trên có thông tin như sau: Học viên Nguyễn Văn Nam có mã sinh viên là 845102 vào ngày 14/10/2002 thi môn HTTT với điểm số là 9.

*Một hệ thống thông tin bảo mật (Secure Information System) là một hệ thống mà thông tin được xử lý trên nó phải đảm bảo được 3 đặc trưng:*

- *Tính bí mật của thông tin (Confidentiality).*
- *Tính toàn vẹn của thông tin (Integrity).*
- *Tính khả dụng của thông tin (Availability).*



**Ba đặc trưng của hệ thống thông tin bảo mật**

Ba đặc trưng này được liên kết lại và xem như là mô hình tiêu chuẩn của các hệ thống thông tin bảo mật, hay nói cách khác, đây là 3 thành phần cốt yếu của một hệ thống thông tin Bảo mật. Mô hình này được sử dụng rộng rãi trong nhiều ngữ cảnh và nhiều tài liệu khác nhau, và được gọi tắt là mô hình CIA.

### **1.1.1. Tính bí mật**

*Tính bí mật (Confidentiality) của thông tin là tính giới hạn về đối tượng được quyền truy xuất đến thông tin.*

Đối tượng truy xuất có thể là con người, là máy tính hoặc phần mềm, kể cả phần mềm phá hoại như virus, worm, spyware,...

Tuỳ theo tính chất của thông tin mà mức độ bí mật của chúng có khác nhau. Ví dụ: các thông tin về chính trị và quân sự luôn được xem là các thông tin nhạy cảm nhất đối với các quốc gia và được xử lý ở mức bảo mật cao nhất. Các thông tin khác như thông tin về hoạt động và chiến lược kinh doanh của doanh nghiệp, thông tin cá nhân, đặc biệt của những người nổi tiếng, thông tin cấu hình hệ thống của các mạng cung cấp dịch vụ, v.v... đều có nhu cầu được giữ bí mật ở từng mức độ.

Để đảm bảo tính bí mật của thông tin, ngoài các cơ chế và phương tiện vật lý như nhà xưởng, thiết bị lưu trữ, dịch vụ bảo vệ,... thì kỹ thuật mật mã hoá (Cryptography) được xem là công cụ bảo mật thông tin hữu hiệu nhất trong môi trường máy tính. Ngoài ra, kỹ thuật kiểm soát truy cập (Access Control) cũng được thiết lập để bảo đảm chỉ có những đối tượng được cho phép mới có thể truy xuất thông tin.

Sự bí mật của thông tin phải được xem xét dưới dạng 2 yếu tố tách rời: sự tồn tại của thông tin và nội dung của thông tin đó.

Đôi khi, tiết lộ sự tồn tại của thông tin có ý nghĩa cao hơn tiết lộ nội dung của nó. Ví dụ: chiến lược kinh doanh bí mật mang tính sống còn của một công ty đã bị tiết lộ cho một công ty đối thủ khác. Việc nhận thức được rằng có điều đó tồn tại sẽ quan trọng hơn nhiều so với việc biết cụ thể về nội dung thông tin, chẳng hạn như ai đã tiết lộ, tiết lộ cho đối thủ nào và tiết lộ những thông tin gì,...

Cũng vì lý do này, trong một số hệ thống xác thực người dùng (user authentication) ví dụ như đăng nhập vào hệ điều hành Netware hay đăng nhập vào hộp thư điện tử hoặc các dịch vụ khác trên mạng, khi người sử dụng cung cấp một tên người dùng (user-name) sai, thay vì thông báo rằng user-name này không tồn tại, thì một số hệ thống sẽ thông báo rằng mật khẩu (password) sai, một số hệ thống khác chỉ thông báo chung chung là “Invalid user name/password” (người dùng hoặc mật khẩu không hợp lệ). Dụng ý đăng sau câu thông báo không rõ ràng này là việc từ chối xác nhận việc tồn tại hay không tồn tại một user-name như thế trong hệ thống. Điều này làm tăng sự khó khăn cho những người muốn đăng nhập vào hệ thống một cách bất hợp pháp bằng cách thử ngẫu nhiên.

### **1.1.2. Tính toàn vẹn**

*Đặc trưng này đảm bảo sự tồn tại nguyên vẹn của thông tin, loại trừ mọi sự thay đổi thông tin có chủ đích hoặc hư hỏng, mất mát thông tin do sự cố thiết bị hoặc phần mềm.*

Tính toàn vẹn đề cập đến khả năng ngăn các dữ liệu không bị thay đổi một cách trái phép hoặc thay đổi không như ý muốn. Điều này có nghĩa là sự thay đổi trái phép hoặc việc xóa dữ liệu (hay các phần dữ liệu) hoặc nó có thể có nghĩa là có sự ủy quyền nhưng không mong muốn làm thay đổi hay xóa dữ liệu của chúng ta. Để duy trì tính toàn vẹn chúng ta không chỉ cần có các phương tiện ngăn chặn những thay đổi dữ liệu một cách trái phép mà còn cần có khả năng để khôi phục các thay đổi đã được thay đổi có thẩm quyền.

Tính toàn vẹn đặc biệt quan trọng khi chúng ta đang nói rằng dữ liệu cung cấp nền tảng cho các quyết định khác.

Ví dụ nếu kẻ tấn công đã làm thay đổi các dữ liệu có chứa các kết quả của các xét nghiệm y tế, chúng ta có thể thấy việc điều trị sai phương pháp có khả năng dẫn đến cái chết của bệnh nhân.

Tính toàn vẹn được xét trên 2 khía cạnh:

- Tính nguyên vẹn của nội dung thông tin.
- Tính xác thực của nguồn gốc của thông tin.

Nói một cách khác, tính toàn vẹn của thông tin phải được đánh giá trên hai mặt: toàn vẹn về nội dung và toàn vẹn về nguồn gốc.

Ví dụ: một ngân hàng nhận được lệnh thanh toán của một người tự xưng là chủ tài khoản với đầy đủ những thông tin cần thiết. Nội dung thông tin được bảo toàn vì ngân hàng đã nhận được một cách chính xác yêu cầu của khách hàng (đúng như người xưng là chủ tài khoản gửi đi). Tuy nhiên, nếu lệnh thanh toán này không phải cho chính chủ tài khoản đưa ra mà do một người nào khác nhò biết được thông tin bí mật về tài khoản đã mạo danh chủ tài khoản để đưa ra, ta nói nguồn gốc của thông tin đã không được bảo toàn.

Một ví dụ khác, một tờ báo đưa tin về một sự kiện vừa xảy ra tại một cơ quan quan trọng của chính phủ, có ghi chú rằng nguồn tin từ người phát ngôn của cơ quan đó. Tuy nhiên, nếu tin đó thật sự không phải do người phát ngôn công bố mà được lấy từ một kênh thông tin khác, không xét đến việc nội dung thông tin có đúng hay không, ta nói rằng nguồn gốc thông tin đã không được bảo toàn.

Sự toàn vẹn về nguồn gốc thông tin trong một số ngữ cảnh có ý nghĩa tương đương với sự đảm bảo tính không thể chối cãi (non-repudiation) của hệ thống thông tin.

Các cơ chế đảm bảo sự toàn vẹn của thông tin được chia thành 2 loại: các cơ chế ngăn chặn (Prevention mechanisms) và các cơ chế phát hiện (Detection mechanisms).

Cơ chế ngăn chặn có chức năng ngăn cản các hành vi trái phép làm thay đổi nội dung và nguồn gốc của thông tin. Các hành vi này bao gồm 2 nhóm: hành vi cố gắng thay

đổi thông tin khi không được phép truy xuất đến thông tin và hành vi thay đổi thông tin theo cách khác với cách đã được cho phép.

Ví dụ: một người ngoài công ty có gắng truy xuất đến cơ sở dữ liệu kế toán của một công ty và thay đổi dữ liệu trong đó. Đây là hành vi thuộc nhóm thứ nhất. Trường hợp một nhân viên kế toán được trao quyền quản lý cơ sở dữ liệu kế toán của công ty, và đã dùng quyền truy xuất của mình để thay đổi thông tin nhằm biến thủ ngân quỹ, đây là hành vi thuộc nhóm thứ hai.

Nhóm các cơ chế phát hiện chỉ thực hiện chức năng giám sát và thông báo khi có các thay đổi diễn ra trên thông tin bằng cách phân tích các sự kiện diễn ra trên hệ thống mà không thực hiện chức năng ngăn chặn các hành vi truy xuất trái phép đến thông tin.

Nếu như tính bí mật của thông tin chỉ quan tâm đến việc thông tin có bị tiết lộ hay không, thì tính toàn vẹn của thông tin vừa quan tâm tới tính chính xác của thông tin và cả mức độ tin cậy của thông tin. Các yếu tố như nguồn gốc thông tin, cách thức bảo vệ thông tin trong quá khứ cũng như trong hiện tại đều là những yếu tố quyết định độ tin cậy của thông tin và do đó ảnh hưởng đến tính toàn vẹn của thông tin. Nói chung, việc đánh giá tính toàn vẹn của một hệ thống thông tin là một công việc phức tạp.

### **1.1.3. Tính khả dụng**

*Tính khả dụng của thông tin là tính sẵn sàng của thông tin cho các nhu cầu truy xuất hợp lệ.*

Khả năng đáp ứng của thông tin là điều rất quan trọng, điều này thể hiện tính sẵn sàng phục vụ của các dịch vụ. Khả năng đáp ứng của hệ thống chịu ảnh hưởng bởi khá nhiều thành phần: có thể là phần cứng, phần mềm hay hệ thống sao lưu. Khả năng đáp ứng của hệ thống cần được tính đến dựa trên số người truy cập và mức độ quan trọng của dữ liệu.

Mất tính khả dụng là sự phá vỡ ở bất kỳ vị trí nào trong hệ thống của chúng ta mà làm cản trở việc truy cập của chúng ta vào dữ liệu của mình. Các vấn đề như thế có thể dẫn đến sự tổn thất về điện năng, các vấn đề về hệ điều hành hoặc ứng dụng, các tấn công mạng, sự tổn thương của hệ thống hoặc các vấn đề khác.

Ví dụ: các thông tin về quản lý nhân sự của một công ty được lưu trên máy tính, được bảo vệ một cách chắc chắn bằng nhiều cơ chế đảm bảo thông tin không bị tiết lộ hay thay đổi. Tuy nhiên, khi người quản lý cần những thông tin này thì lại không truy xuất được vì lỗi hệ thống. Khi đó, thông tin hoàn toàn không sử dụng được và ta nói tính khả dụng của thông tin không được đảm bảo.

Tính khả dụng là một yêu cầu rất quan trọng của hệ thống, bởi vì một hệ thống tồn tại nhưng không sẵn sàng cho sử dụng thì cũng giống như không tồn tại một hệ thống thông tin nào. Một hệ thống khả dụng là một hệ thống làm việc trôi chảy và hiệu quả, có khả năng phục hồi nhanh chóng nếu có sự cố xảy ra.

Trong thực tế, tính khả dụng được xem là nền tảng của một hệ thống bảo mật, bởi vì khi hệ thống không sẵn sàng thì việc đảm bảo 2 đặc trưng còn lại (bí mật và toàn vẹn) sẽ trở nên vô nghĩa.

Hiện nay, các hình thức tấn công từ chối dịch vụ DoS (Denial of Service) và DDoS (Distributed Denial of Service) được đánh giá là các nguy cơ lớn nhất đối với sự an toàn của các hệ thống thông tin, gây ra những thiệt hại lớn và đặc biệt là chưa có giải pháp ngăn chặn hữu hiệu. Các hình thức tấn công này đều nhắm vào tính khả dụng của hệ thống.

Một số hướng nghiên cứu đang đưa ra các mô hình mới cho việc mô tả các hệ thống an toàn. Theo đó, mô hình CIA không mô tả được đầy đủ các yêu cầu an toàn của hệ thống mà cần phải định nghĩa lại một mô hình khác với các đặc tính của thông tin cần được đảm bảo như:

- Tính khả dụng (Availability)
- Tính tiện ích (Utility)
- Tính toàn vẹn (Integrity)
- Tính xác thực (Authenticity)
- Tính bảo mật (Confidentiality)
- Tính sở hữu (Possession)

## 1.2. Các mối đe dọa

Các mối đe dọa bảo mật (*security threat*) hay còn gọi là nguy cơ là những sự kiện có ảnh hưởng đến an toàn của hệ thống thông tin.

Ví dụ: tấn công từ chối dịch vụ (DoS và DDoS) là một mối đe dọa đối với hệ thống các máy chủ cung cấp dịch vụ trên mạng.

Khi nói đến mối đe dọa, nghĩa là sự kiện đó chưa xảy ra, nhưng có khả năng xảy ra và có khả năng gây hại cho hệ thống. Có những sự kiện có khả năng gây hại, nhưng không có khả năng xảy ra đối với hệ thống thì không được xem là mối đe dọa.

Ví dụ: tấn công của sâu Nimda (năm 2001) có khả năng gây tê liệt toàn bộ hệ thống mạng nội bộ. Tuy nhiên, sâu Nimda chỉ khai thác được lỗi bảo mật của phần mềm IIS (Internet Information Service) trên Windows (NT và 2000) và do đó chỉ có khả năng xảy ra trên mạng có máy cài đặt hệ điều hành Windows. Nếu một mạng máy tính chỉ gồm toàn các máy cài hệ điều hành Unix hoặc Linux thì sâu Nimda hoàn toàn không có khả năng tồn tại, và do vậy, sâu Nimda không phải là một mối đe dọa trong trường hợp này.

Các mối đe dọa được chia làm 4 loại:

- Xem thông tin một cách bất hợp pháp: Các hành vi thuộc nhóm này có thể đơn giản như việc nghe lén một cuộc đàm thoại, mở một tập tin trên máy của người khác, hoặc phức tạp hơn như xen vào một kết nối mạng (wire –tapping) để ăn cắp dữ liệu, hoặc cài các chương trình ghi bàn phím (key-logger) để ghi lại những thông tin quan trọng được nhập từ bàn phím.

- Chính sửa thông tin một cách bất hợp pháp: Các hành vi thuộc nhóm này bao gồm những hành vi tương tự như nhóm ở trên nhưng mang tính chủ động, tức là có thay đổi thông tin gốc. Hành vi này không chỉ gây ra sai dữ liệu mà còn có thể làm thay đổi các chính sách an toàn của hệ thống hoặc ngăn chặn hoạt động bình thường của hệ thống (trường hợp thông tin bị thay đổi là thông tin điều khiển hệ thống).

- Tù chối dịch vụ bao gồm các hành vi có mục đích ngăn chặn hoạt động bình thường của hệ thống bằng cách làm chậm hoặc gián đoạn dịch vụ của hệ thống. Tấn công từ chối dịch vụ hoặc virus là những nguy cơ thuộc nhóm này

- Chiếm quyền điều khiển: Chiếm quyền điều khiển hệ thống gây ra nhiều mức độ thiệt hại khác nhau, từ việc lấy cắp và thay đổi dữ liệu trên hệ thống, đến việc thay đổi các chính sách bảo mật và vô hiệu hóa các cơ chế bảo mật đã được thiết lập. Ví dụ điển hình cho nhóm nguy cơ này là các phương thức tấn công nhằm chiếm quyền root trên các máy tính chạy Unix hoặc Linux bằng cách khai thác các lỗ phần mềm hoặc lỗ cấu hình hệ thống. Tấn công tràn bộ đệm (buffer overflow) là cách thường dùng nhất để chiếm quyền root trên các hệ thống Linux vốn được xây dựng trên nền tảng của ngôn ngữ lập trình C.

Cần chú ý phân biệt giữa thuật ngữ mối đe dọa và rủi ro. Mỗi đe dọa là những hành vi, những sự kiện hoặc đối tượng có khả năng gây hại cho hệ thống. Rủi ro là những thiệt hại có khả năng xảy ra đối với hệ thống.

Ví dụ: Tấn công từ chối dịch vụ là một mối đe dọa (threat). Đây là một sự kiện có khả năng xảy ra đối với bất kỳ hệ thống cung cấp dịch vụ nào. Thiệt hại do tấn công này gây ra là hệ thống bị gián đoạn hoạt động, đây mới là rủi ro (risk). Tuy nhiên, không phải bất kỳ tấn công từ chối dịch vụ nào xảy ra cũng đều làm cho hệ thống ngưng hoạt động, và hơn nữa, tấn công từ chối dịch vụ không phải là nguồn gốc duy nhất gây ra gián đoạn hệ thống; những nguy cơ khác như lỗi hệ thống (do vận hành sai), lỗi phần mềm (do lập trình), lỗi phần cứng (hư hỏng thiết bị, mất điện,...) cũng đều có khả năng dẫn đến gián đoạn hệ thống.

Một ví dụ khác, xét trường hợp lưu trữ tập tin trên một máy tính chạy hệ điều hành Windows 98 đã nói ở trên. Nguy cơ đối với hệ thống là các hành vi sửa hoặc xoá tập tin trên máy người khác. Những người hay sử dụng máy tính của người khác cũng được xem là nguy cơ đối với hệ thống. Rủi ro đối với hệ thống trong trường hợp này là việc tập tin bị mất hoặc bị sửa.

Theo thống kê thấy nguy cơ cơ đến đến từ con người 80%, nguy cơ tấn công từ mạng, hệ thống máy tính 8%, nguy cơ liên quan hệ thống vật lý 8% còn lại đến từ các nguy cơ khác.

### 1.2.1. Các nguy cơ từ nhân tố con người

#### a. Nguy cơ từ lỗi của người lập trình

Nguy cơ này được xếp vào hàng nguy hiểm nhất. Khi lập trình, các cảnh báo và lỗi do trình biên dịch đưa ra thường bị bỏ qua và nó có thể dẫn đến những sự việc không

đáng có, ví dụ như tràn bộ đệm, tràn heap. Khi người dùng vô tình (hay cố ý) sử dụng các đầu vào không hợp lý thì chương trình sẽ xử lý sai, hoặc dẫn đến việc bị khai thác, đổ vỡ (crash). Kỹ thuật lập trình đóng vai trò rất quan trọng trong mọi ứng dụng. Và lập trình viên phải luôn luôn cập nhật thông tin, các lỗi bị khai thác, cách phòng chống, sử dụng phương thức lập trình an toàn.

- Một cách tốt nhất để phòng tránh là sử dụng chính sách “least privilege” (đặc quyền tối thiểu). Người dùng sẽ chỉ được xử lý, truy cập đến một số vùng thông tin nhất định.

### b. Gian lận và trộm cắp thông tin

Trong nội bộ của mỗi cơ quan tổ chức đã xảy ra các trường hợp nhân viên tại cơ quan tổ chức đó bị các thế lực bên ngoài mua chuộc để đánh cắp thông tin quan trọng trong nội bộ, đặc biệt là những cơ quan quân sự, cơ quan nhà nước... Việc trộm cắp và gian lận có thể được thực hiện dưới nhiều hình thức như lấy cắp bản in hay lấy cắp thông tin số, cung cấp thông tin nội bộ cho bên ngoài.

Việc phát hiện kẻ gian lận và trộm cắp thông tin khó phát hiện từ những người trong nội bộ của cơ quan, tổ chức.

- Cách tốt nhất để phòng tránh nguy cơ này là: phải có những chính sách bảo mật được thiết kế tốt. Những chính sách có thể giúp người quản lý bảo mật thông tin thu thập thông tin, từ đó điều tra và đưa ra những kết luận chính xác, nhanh chóng. Khi đã có một chính sách tốt, người quản trị có thể sử dụng các kỹ thuật điều tra sói (forensics) để truy vết các hành động tấn công. Ví dụ như hình thức lấy cắp thông tin số, nếu một nhân viên truy cập vào khu vực đặt tài liệu bí mật của công ty, hệ thống sẽ ghi lại được thời gian, IP, tài liệu bị lấy, sử dụng phần mềm gì để truy cập, phần mềm bị cài đặt trái phép... từ đó người quản trị sẽ chứng minh được ai đã làm việc này.

- Cần đào tạo nâng cao ý thức của những người trong nội bộ cơ quan tổ chức phải biết được giới hạn cho phép.

- Cần sao lưu thông tin dự phòng tránh trường hợp bị lộ lọt, tấn công bởi những kẻ xấu.

### c. Nguy cơ từ những kẻ tấn công

Có một số nhóm người dùng Internet sẽ tấn công hệ thống thông tin. Ba nhóm chính là hacker, cracker và phreak. Trong khi thuật ngữ phổ biến để gọi tất cả ba nhóm này là "hacker" nhưng có một số sự khác biệt giữa các nhóm.

Một hacker là một người dùng thâm nhập vào một hệ thống chỉ để xem xét xung quanh và xem những gì có thể. Quy tắc của các hacker là sau khi họ đã thâm nhập vào hệ thống, họ sẽ thông báo cho quản trị hệ thống để cho người quản trị biết rằng hệ thống có một lỗ hổng. Người ta thường nói rằng một hacker chỉ muốn an toàn được cải thiện trên tất cả các hệ thống Internet.

Nhóm tiếp theo, cracker, là nhóm thực sự nguy hiểm. Một cracker không có quy tắc nào để xâm nhập vào một hệ thống. Cracker sẽ gây thiệt hại hoặc phá hủy dữ liệu nếu

họ có thể xâm nhập vào một hệ thống. Mục tiêu của cracker là gây ra càng nhiều thiệt hại cho tất cả các hệ thống trên Internet càng tốt.

Nhóm cuối cùng - Phreak, cố gắng đột nhập vào hệ thống điện thoại của một tổ chức. Các Phreak sau đó có thể sử dụng truy cập điện thoại miễn phí để che giấu số điện thoại thực sự mà họ đang dùng để gọi, và cũng có thể làm cho tổ chức phải thanh toán hóa đơn với một số tiền lớn cho chi phí điện thoại đường dài.

Các cách một hacker tấn công một hệ thống có thể khác nhau rất nhiều. Mỗi kẻ tấn công có các thủ thuật riêng của mình có thể được sử dụng để xâm nhập vào một hệ thống. Phương pháp cơ bản của hacker có năm thành phần chính: trinh sát/thăm dò, quét, đoạt quyền truy cập, duy trì quyền truy cập, và xóa dấu vết. Điều này có vẻ kỳ lạ để các hacker suy nghĩ về một phương pháp luận, nhưng như với bất cứ cái gì khác, đó chỉ là vấn đề thời gian. Vì vậy, để tối đa hóa thời gian, hầu hết các hacker đi theo một phương pháp tương tự.

### **1.2.2 Các nguy cơ từ nhân tố kỹ thuật**

#### **a. Tấn công mã độc**

Trong khi những người dùng ác ý hay hacker có thể tấn công hệ thống, các chương trình được phát hành bởi cùng một nhóm người này thường sẽ thành công hơn trong việc tiếp cận các thành phần bảo vệ của tổ chức. Mã độc hại được định nghĩa là bất kỳ đoạn mã trong một phần mềm hay kịch bản (script) được thiết kế với mục đích làm cho một hệ thống thực hiện bất kỳ hành động nào theo ý của người sử dụng (mã độc hại) với quyền của chủ sở hữu hệ thống.

Tấn công mã độc là hình thức phổ biến nhất hiện nay. Mã độc bao gồm spyware (phần mềm gián điệp), ransomware (mã độc tống tiền), virus và worm, trojan horse, logic bomb... (phần mềm độc hại có khả năng lây lan nhanh). Thông thường, tin tức sẽ tấn công người dùng thông qua các lỗ hổng bảo mật, cũng có thể là dụ dỗ người dùng bấm vào một đường liên kết hoặc email (phishing) để phần mềm độc hại tự động cài đặt vào máy tính. Một trong những cách nhanh nhất để đưa mã độc vào mạng được bảo vệ của một tổ chức mục tiêu là gửi các mã độc hại thông qua thư điện tử.



### **Minh họa quá trình thực thi một mã độc**

Cài mã độc vào máy tính có thể qua nhiều con đường: lỗ hổng phần mềm (điển hình như adobe Flash, rất nhiều lỗ hổng 0-days được phát hiện, hay Java Runtime Environment thời gian gần đây cũng liên tục đưa ra bản vá bảo mật); hệ thống đã bị hacker điều khiển; sử dụng phần mềm crack, không có giấy phép sử dụng;

Một khi được cài đặt thành công, mã độc có thể gây ra một số vấn đề như:

- Ngăn cản người dùng truy cập vào một tệp tin hoặc folder quan trọng (ransomware)

- Cài đặt thêm những phần mềm độc hại khác
- Lén lút theo dõi người dùng và đánh cắp dữ liệu (spyware)
- Làm hư hại phần mềm, phần cứng, làm gián đoạn hệ thống.

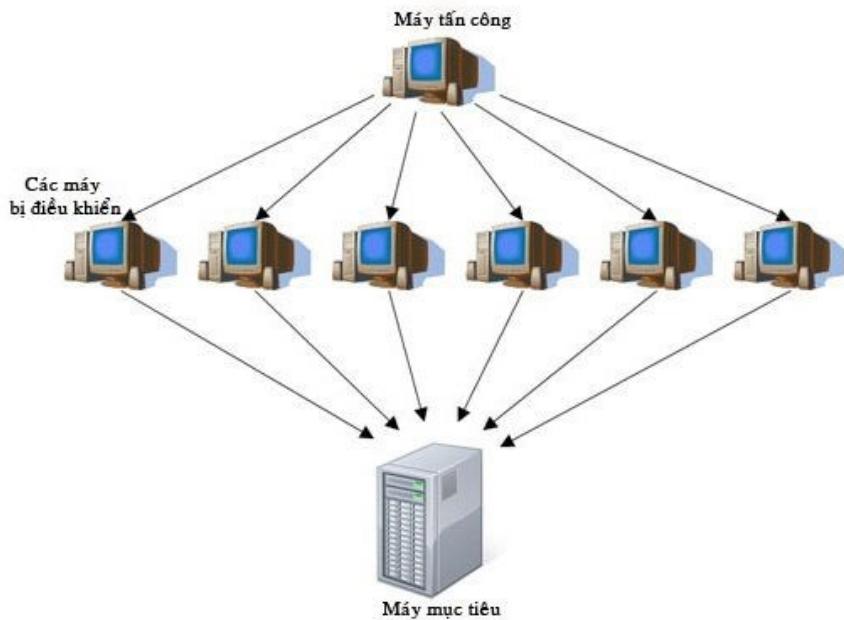
Facebook cũng từng bị một nhóm hacker tấn công do máy tính của một số nhân viên bị cài mã độc, hay một số vụ việc điển hình nhất ở việt nam là vụ việc hàng loạt website của VCCorp bị tấn công, vụ tấn công vào hãng hàng không Việt Nam Airline, vụ tấn công vào ngân hàng TP Bank...

Cách tốt nhất để tránh nguy cơ này là luôn cập nhật phần mềm xử lý dữ liệu, hệ điều hành và phần mềm an ninh mạng, diệt virus.

#### **b. Tấn công từ chối dịch vụ**

Nếu một hacker không thể cướp quyền truy cập vào một hệ thống, kẻ tấn công sẽ tìm cách tấn công từ chối dịch vụ làm hệ thống không thể phục vụ người dùng được trong một khoảng thời gian, bằng cách truy cập đến hệ thống liên tục, số lượng lớn, có tổ chức. Có 2 kiểu tấn công từ chối dịch vụ:

DoS (Denial of Service) là hình thức tấn công mà tin tặc “đánh sập tạm thời” một hệ thống, máy chủ, hoặc mạng nội bộ. Để thực hiện được điều này, chúng thường tạo ra một lượng traffic/request khổng lồ ở cùng một thời điểm, khiến cho hệ thống bị quá tải, từ đó người dùng không thể truy cập vào dịch vụ trong khoảng thời gian mà cuộc tấn công DoS diễn ra. Tấn công này có thể xảy ra với cả ứng dụng trực tuyến và ứng dụng offline. Với ứng dụng trực tuyến, hacker sử dụng các công cụ tấn công (tấn công Syn floods, Fin floods, Smurfs, Fraggles) trên một máy tính để tấn công vào hệ thống, khiến nó không thể xử lý được yêu cầu, hoặc làm nghẽn băng thông khiến người dùng khác khó mà truy cập được. Với ứng dụng offline, hacker tạo ra những dữ liệu cực lớn, hoặc các dữ liệu xấu (làm cho quá trình xử lý của ứng dụng bị ngưng trệ, treo).



### Tấn công từ chối dịch vụ

DDoS (Distributed Denial of Service): tin tặc sử dụng một mạng lưới các máy tính (botnet) để tấn công nạn nhân. Các máy tính botnet được điều khiển bởi một (một vài) server của hacker (gọi là server điều khiển). Điều nguy hiểm là chính các máy tính thuộc mạng lưới botnet cũng không biết bản thân đang bị lợi dụng để làm công cụ tấn công. Loại tấn công này khó phát hiện ra hơn cho các hệ thống phát hiện tự động, giúp hacker ẩn mình tốt hơn.

Để chống lại nguy cơ này, hệ thống cần có nhiều máy chủ phục vụ, máy chủ phân tải, cơ chế phát hiện tấn công DoS hiệu quả.

#### 1.2.3 Các nguy cơ khác

##### a. Nguy cơ vật lý

Nguy cơ mất an toàn thông tin về khía cạnh vật lý là nguy cơ do mất điện, nhiệt độ, độ ẩm không đảm bảo, hỏa hoạn, thiên tai, thiết bị phần cứng bị hư hỏng, các phần tử phá hoại như nhân viên xấu bên trong và kẻ trộm bên ngoài.

Các mối nguy hiểm vật lý gồm:

**Hỏa hoạn và cháy nổ:** Hỏa hoạn và cháy nổ sẽ gây hư hỏng toàn bộ hệ thống và dữ liệu. Biện pháp phòng chống hỏa hoạn và cháy nổ như sau: Có hệ thống phát hiện hỏa hoạn; Luôn có sẵn thiết bị chữa cháy và phải kiểm tra thiết bị này thường xuyên; Cấm hút thuốc trong các khu vực chứa máy móc, thiết bị; Các vật liệu dễ cháy cần chứa ở phòng riêng.

**Nhiệt độ và độ ẩm:** Nhiệt độ và độ ẩm không đúng sẽ làm giảm tuổi thọ của thiết bị. Biện pháp bảo vệ như sau: hệ thống điều hòa được dùng để điều khiển nhiệt độ và độ ẩm; Theo ASHRAE (American Society of Heating, Refrigerating and Air Conditioning Engineers) các thiết bị điện tử hoạt động tốt ở nhiệt độ 20–25 °C và độ ẩm 40– 60%.

**Đe dọa do các thảm họa thiên tai:** Các thảm họa thiên tai như ngập lụt, bão, sấm chớp, động đất sẽ gây hư hỏng toàn bộ hệ thống và dữ liệu, khó lường trước được hậu quả của nó. Biện pháp phòng tránh là thường xuyên theo dõi, cập nhật các thông tin về dự báo thời tiết để có phương án đề phòng trước.

**Hóa chất:** Có thể gây biến dạng thiết bị và mất dữ liệu. Biện pháp phòng chống là không đặt gần những hóa chất độc.

**Mất điện:** Biện pháp phòng chống trường hợp mất điện đột ngột là đảm bảo nguồn điện 24/24: UPS và máy phát điện; Tránh tăng giảm điện áp đột ngột.

**Thiết bị hỏng:** Biện pháp phòng chống thiết bị hỏng là sử dụng và bảo trì đúng kỹ thuật và thường xuyên back-up dữ liệu.

c. Cơ sở hạ tầng mạng

Cơ sở hạ tầng không đồng bộ, không đảm bảo yêu cầu thông tin được truyền trong hệ thống an toàn và thông suốt.

d. Thông tin

Dữ liệu chưa được mô hình hóa và chuẩn hóa theo tiêu chuẩn về mặt tổ chức và mặt kỹ thuật. Yếu tố pháp lý chưa được trú trọng trong truyền đưa các dữ liệu trên mạng, nghĩa là các dữ liệu được truyền đi trên mạng phải đảm bảo tính hợp pháp về mặt tổ chức và mặt kỹ thuật.

e. Công nghệ

Chưa chuẩn hóa cho các loại công nghệ, mô hình kiến trúc tham chiếu nhằm đảm bảo cho tính tương hợp, tính sử dụng lại được, tính mở, an ninh, mở rộng theo phạm vi, tính riêng tư vào trong HTTT.

### **1.3. Chính sách và cơ chế**

Hai khái niệm quan trọng thường được đề cập khi xây dựng một hệ thống bảo mật: Chính sách bảo mật (Security policy), cơ chế bảo mật (Security mechanism).

#### **1.3.1. Chính sách bảo mật**

*Chính sách bảo mật là hệ thống các quy định nhằm đảm bảo sự an toàn của hệ thống.*

Việc bảo mật hệ thống cần có một chính sách bảo mật rõ ràng. Cần có những chính sách bảo mật riêng cho những yêu cầu bảo mật khác nhau.

Xây dựng và lựa chọn các chính sách bảo mật cho hệ thống phải dựa theo các chính sách bảo mật do các tổ chức uy tín về bảo mật định ra (compliance) như: NIST, SP800, ISO17799, HIPAA.

Chính sách bảo mật phải cân bằng giữa 3 yếu tố khả dụng, bảo mật, hiệu suất. Chính sách bảo mật có thể được biểu diễn bằng ngôn ngữ tự nhiên hoặc ngôn ngữ toán học.

Ví dụ: trong một hệ thống, để bảo đảm an toàn cho một tài nguyên (resource) cụ thể, chính sách an toàn quy định rằng chỉ có người dùng nào thuộc nhóm quản trị hệ thống (Administrators) mới có quyền truy xuất, còn những người dùng khác thì không. Đây là cách biểu diễn bằng ngôn ngữ tự nhiên.

Có thể biểu diễn quy định này bằng ngôn ngữ toán học như sau:

Gọi:  $U$  là tập hợp các người dùng trong hệ thống.

$A$  là tập hợp các người dùng thuộc nhóm quản trị.

$O$  là tập hợp các đối tượng (tài nguyên) trong hệ thống.

Thao tác  $\text{Access}(u, o)$  cho giá trị TRUE nếu người dùng  $u$  có quyền truy xuất đến đối tượng  $o$ , ngược lại, cho giá trị FALSE.

Quy định p trong chính sách an toàn được phát biểu như sau:

$$\forall u \in U, \forall o \in O: \text{Access}(u, o) = \text{TRUE} \Leftrightarrow u \in A$$

Ma trận cũng thường được dùng để biểu diễn một chính sách bảo mật.

Ví dụ: một hệ thống với các tập người dùng  $U = \{u_1, u_2, u_3, u_4\}$  và tập đối tượng  $O = \{o_1, o_2, o_3, o_4\}$ . Các thao tác mà một người dùng  $u$  có thể thực hiện được trên một đối tượng  $o$  bao gồm đọc (r), ghi (w) và thực thi (x). Quy định về khả năng truy xuất của từng người dùng đến từng đối tượng trong hệ thống được biểu diễn bằng ma trận như Bảng 1.1.

Quan sát ma trận, ta biết rằng người dùng  $u_3$  được quyền đọc trên tất cả các đối tượng từ  $o_1$  đến  $o_4$ , trong khi đó người dùng  $u_4$  thì không có quyền truy xuất đến bất kỳ đối tượng nào.

**Bảng 0.1. Ma trận về khả năng truy xuất của từng người dùng**

	$u_1$	$u_2$	$u_3$	$u_4$
$o_1$	x	x	r	
$o_2$	x	r	r	
$o_3$	w		r	
$o_4$	w		r	

Trong thực tế, việc đề ra chính sách bảo mật cho một hệ thống thông tin phải đảm bảo được sự cân bằng giữa lợi ích của việc bảo đảm an toàn hệ thống và chi phí thiết kế, cài đặt và vận hành các cơ chế bảo vệ chính sách đó.

### 1.3.2. Cơ chế bảo mật

Cơ chế bảo mật là hệ thống các phương pháp, công cụ, thủ tục,... dùng để thực thi các quy định của chính sách bảo mật.

Xác định cơ chế bảo mật phù hợp để hiện thực các chính sách bảo mật và đạt được các mục tiêu bảo mật đề ra.

Cơ chế bảo mật thông thường là các biện pháp kỹ thuật.

Ví dụ: xây dựng bức tường lửa (firewall), xác thực người dùng, dùng cơ chế bảo vệ tập tin của hệ thống quản lý tập tin NTFS để phân quyền truy xuất đối với từng tập tin/thư mục trên đĩa cứng, dùng kỹ thuật mã hoá để che giấu thông tin,...

Tuy nhiên, đôi khi cơ chế chỉ là những thủ tục (procedure) mà khi thực hiện nó thì chính sách được bảo toàn.

Ví dụ: phòng thực hành máy tính của trường đại học quy định: sinh viên không được sao chép bài tập của sinh viên khác đã được lưu trên máy chủ. Đây là một quy định của chính sách bảo mật. Để thực hiện quy định này, các cơ chế được áp dụng bao gồm: tạo thư mục riêng trên máy chủ cho từng sinh viên, phân quyền truy xuất cho từng sinh viên đến các thư mục này và yêu cầu sinh viên phải lưu bài tập trong thư mục riêng, mỗi khi rời khỏi máy tính phải thực hiện thao tác logout khỏi hệ thống.

Trong cơ chế này, các biện pháp như tạo thư mục riêng, gán quyền truy xuất,... là các biện pháp kỹ thuật. Biện pháp yêu cầu sinh viên thoát khỏi hệ thống (logout) khi rời khỏi máy là một biện pháp thủ tục. Nếu sinh viên ra về mà không thoát ra khỏi hệ thống, một sinh viên khác có thể sử dụng phiên làm việc đang mở của sinh viên này để sao chép bài tập. Khi đó, rõ ràng chính sách bảo mật đã bị vi phạm.

Thông thường, việc xây dựng một hệ thống bảo mật phải dựa trên 2 giả thiết sau đây:

- Chính sách bảo mật phân chia một cách rõ ràng các trạng thái của hệ thống thành 2 nhóm: an toàn và không an toàn.

- Cơ chế bảo mật có khả năng ngăn chặn hệ thống tiến vào các trạng thái không an toàn.

Chỉ cần một trong hai giả thiết này không đảm bảo thì hệ thống sẽ không an toàn. Từng cơ chế riêng lẻ được thiết kế để bảo vệ một hoặc một số các quy định trong chính sách. Tập hợp tất cả các cơ chế triển khai trên hệ thống phải đảm bảo thực thi tất cả các quy định trong chính sách.

Hai nguy cơ có thể xảy ra khi thiết kế hệ thống bảo mật do không đảm bảo 2 giả thiết ở trên:

- Chính sách không liệt kê được tất cả các trạng thái không an toàn của hệ thống, hay nói cách khác, chính sách không mô tả được một hệ thống bảo mật thật sự.

- Cơ chế không thực hiện được tất cả các quy định trong chính sách, có thể do giới hạn về kỹ thuật, ràng buộc về chi phí, ...

Dựa trên những nhận thức này, có thể đánh giá mức độ an toàn của một cơ chế như sau:

Gọi P là tập hợp tất cả các trạng thái của hệ thống, Q là tập hợp các trạng thái an toàn theo định nghĩa của chính sách bảo mật, giả sử cơ chế đang áp dụng có khả năng giới hạn các trạng thái của hệ thống trong tập R. Ta có các định nghĩa như sau:

- Nếu  $R \subseteq Q$ : cơ chế được đánh giá là an toàn (secure mechanism).
- Nếu  $R = Q$ : cơ chế được đánh giá là chính xác (precise mechanism).
- Nếu tồn tại trạng thái  $r \in R$  sao cho  $r \notin Q$ : cơ chế được đánh giá là lỏng lẻo (broad mechanism).

Có 4 cơ chế bảo mật phổ biến:

- Điều khiển truy cập (Access control)
- Điều khiển suy luận (Inference control)
- Điều khiển dòng thông tin (Flow control)
- Mã hóa (Encryption)

#### 1.4. Vấn đề con người

Con người luôn là trung tâm của tất cả các hệ thống bảo mật, bởi vì tất cả các cơ chế, các kỹ thuật được áp dụng để bảo đảm an toàn hệ thống đều có thể dễ dàng bị vô hiệu hóa bởi con người trong chính hệ thống đó.

Ví dụ: hệ thống xác thực người sử dụng yêu cầu mỗi người trong hệ thống khi muốn thao tác trên hệ thống đều phải cung cấp tên người dùng và mật khẩu. Tuy nhiên, nếu người được cấp mật khẩu không bảo quản kỹ thông tin này, hoặc thậm chí đem tiết lộ cho người khác biết, thì khả năng xảy ra các vi phạm đối với chính sách an toàn là rất cao vì hệ thống xác thực đã bị vô hiệu hóa.

Những người có chủ ý muốn phá vỡ chính sách bảo mật của hệ thống được gọi chung là những người xâm nhập (intruder hoặc attacker) và theo cách nghĩ thông thường thì đây phải là những người bên ngoài hệ thống. Tuy nhiên, thực tế đã chứng minh được rằng chính những người bên trong hệ thống, những người có điều kiện tiếp cận với hệ thống lại là những người có khả năng tấn công hệ thống cao nhất. Đó có thể là một nhân viên đang bất mãn và muốn phá hoại, hoặc chỉ là một người thích khám phá và chứng tỏ mình. Các tấn công gây ra bởi các đối tượng này thường khó phát hiện và gây thiệt hại nhiều hơn các tấn công từ bên ngoài.

Những người không được huấn luyện về an toàn hệ thống cũng là nơi tiềm ẩn các nguy cơ do những hành vi vô ý của họ như thao tác sai, bỏ qua các khâu kiểm tra an toàn, không tuân thủ chính sách bảo mật thông tin như lưu tập tin bên ngoài thư mục an toàn, ghi mật khẩu lên bàn làm việc, ...

## CHƯƠNG 2. ĐIỀU KHIỂN TRUY CẬP

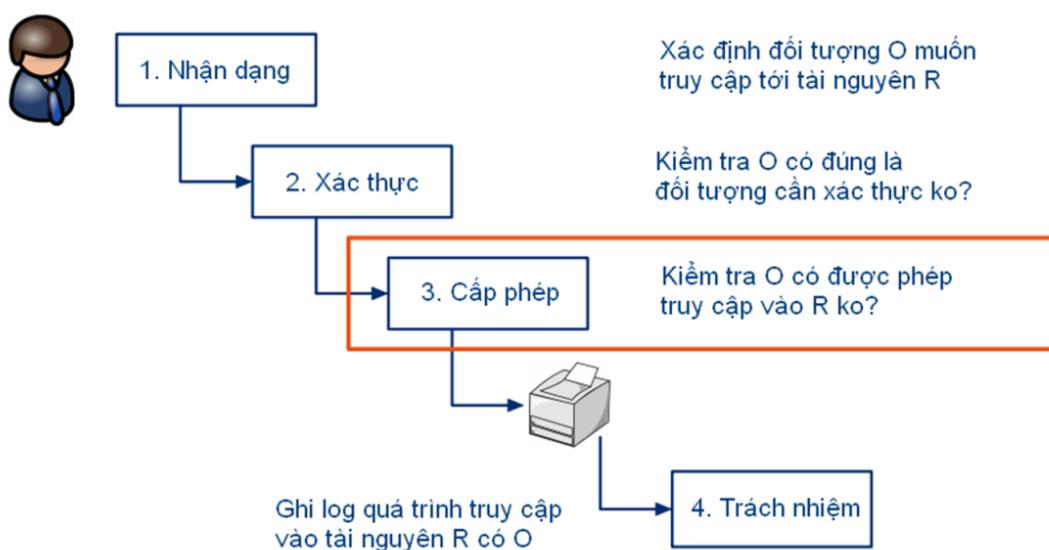
### 2.1. Mục tiêu của điều khiển truy cập

Điều khiển truy nhập (Access control) được định nghĩa là một quy trình được thực hiện bởi một thiết bị phần cứng hay một module phần mềm, có tác dụng chấp thuận hay từ chối một sự truy nhập cụ thể đến một tài nguyên cụ thể.

Điều khiển truy nhập được thực hiện tại nhiều vị trí khác nhau của hệ thống, chẳng hạn như tại thiết bị truy nhập mạng (như remote access server-RAS hoặc wireless access point -WAP), tại hệ thống quản lý tập tin của một hệ điều hành ví dụ NTFS trên Windows hoặc trên các hệ thống Active Directory Service trong Netware 4.x hay Windows 2000 server,...

Có ba khái niệm cơ bản trong mọi ngữ cảnh điều khiển truy cập, bao gồm:

- Chính sách (policy): Là các luật do bộ phận quản trị tài nguyên đề ra.
  - Chủ thể (subject): Có thể là người sử dụng, mạng, các tiến trình hay các ứng dụng yêu cầu được truy cập vào tài nguyên.
  - Đối tượng (object): Là các tài nguyên mà chủ thể được phép truy cập.
- Ba bước để thực hiện điều khiển truy cập:
- Nhận dạng (Identification): Xử lý nhận dạng một chủ thể khi truy cập vào hệ thống.
  - Xác thực (Authentication): Chứng thực nhận dạng chủ thể đó.
  - Trao quyền (Authorization): Gán quyền được phép hoặc không được phép truy cập vào đối tượng.



### Tiến trình điều khiển truy cập

### 2.2. Các kiểu xác thực

Xác thực (Authentication) là một hành động nhằm thiết lập hoặc chứng thực một cái gì đó (hoặc một người nào đó) đáng tin cậy, có nghĩa là, những lời khai báo do người đó đưa ra hoặc về vật đó là sự thật. Xác thực một đối tượng còn có nghĩa là công nhận nguồn gốc của đối tượng, trong khi, xác thực một người thường bao gồm việc thẩm tra

*nhận dạng của họ. Việc xác thực thường phụ thuộc vào một hoặc nhiều nhân tố xác thực (authentication factors) để minh chứng cụ thể.*

- Trong an ninh máy tính (*computer security*), những chủ thể trực tiếp (*subject*) tham gia vào môi trường xác thực là các chương trình phần mềm, chính xác là các tiến trình, nhưng chúng hoạt động thay mặt cho (*dưới sự điều khiển*) của các thực thể bên ngoài (*external entity*), thông thường là những người sử dụng (*user*). Vì vậy về mặt kỹ thuật, cơ chế xác thực chính là cơ chế gắn kết (*binding*) của một danh tính (*của thực thể bên ngoài*) với một chủ thể bên trong hoạt động thay mặt (*subject*).

- Trong một mạng lưới tín nhiệm, việc xác thực là một cách để đảm bảo rằng người dùng chính là người mà họ nói họ là, và người dùng hiện đang thi hành những chức năng trong một hệ thống, trên thực tế, chính là người đã được ủy quyền để làm những việc đó.

- Xác thực có thể phân loại thành:

+ Xác thực thực thể (*Entity Authentication*): *Xác thực 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. Xác thực thực thể có thể một phía hoặc nhiều phía. Trong xác thực một phía, chỉ một phía thực hiện sự xác thực tự bản thân mình khi liên lạc, nhưng trái lại, trong xác thực thực thể đồng thời, cả hai bên phải xác thực lẫn nhau.

+ Xác thực dữ liệu (*Data Authentication*): *Xác thực dữ liệu 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, đảm bảo tính toàn vẹn dữ liệu.*

### **2.2.1. Xác thực dựa trên những gì người sử dụng biết**

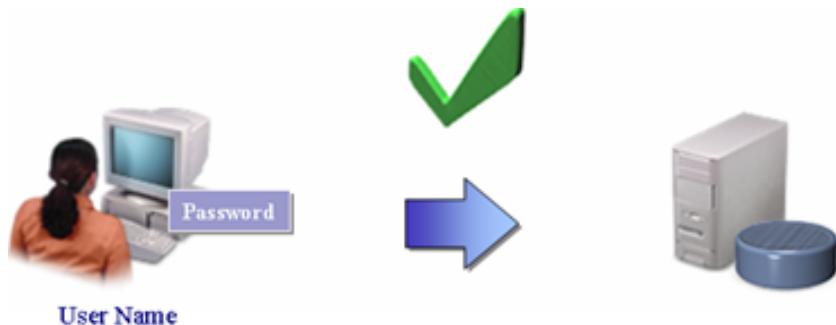
Xác thực dựa trên những gì người sử dụng biết (*tri thức*) là phương thức sử dụng rộng rãi nhất hiện nay trên thế giới. Đây là phương thức đã rất quen thuộc với đa số người sử dụng. Những yếu tố người sử dụng biết, chẳng hạn như mật khẩu (*password*) bằng dãy các ký tự và hình ảnh (*nhung sử dụng mật khẩu là các ký tự thì đang được phổ biến rộng rãi hơn*), mật ngữ (*pass phrase*) hoặc mã số định danh cá nhân (*personal identification number - PIN*).

#### **2.2.1.1. Xác thực Username/Password**

Xác thực dựa trên User name và Password là cách xác thực cơ bản nhất nhưng cũng phổ biến nhất.

### Xác thực dựa trên User name và Password của Yahoo mail

Với phương thức này, khi người dùng đăng nhập vào hệ thống, hệ thống yêu cầu người dùng nhập các thông tin username và password để xác thực, các thông tin nhập vào được đối chiếu với dữ liệu lưu trữ trên cơ sở dữ liệu hệ thống, nếu trùng khớp username và password, thì người sử dụng (User) được xác thực còn nếu không thì người sử dụng bị từ chối hoặc cấm truy cập.



### Xác thực bằng Username/Password

Các hệ xác thực dựa trên mật khẩu thường được phân biệt bởi phương thức lưu trữ mật khẩu trong hệ thống. Ở các hệ thống thế hệ trước, mật khẩu thường được lưu dưới dạng bản rõ (*plaintext*) trên hệ thống. Phương thức này có tính bảo mật không cao vì thông tin xác nhận người dùng được lưu trữ trong tình trạng “plain text” (kí tự văn bản chuẩn), tức là không được mã hóa và bất kỳ ai có thẩm quyền truy cập hệ thống (*quản trị viên hoặc hacker khi chiếm được quyền kiểm soát*) đều có thể lấy cắp được mật khẩu và thông tin của người dùng. Để khắc phục nhược điểm này, các hệ thống ngày nay thường lưu mật khẩu dưới dạng mã băm sử dụng hàm băm 1 chiều (Ví dụ là MD5, SHA-256) của mật khẩu người dùng. Theo đó, khi người dùng nhập mật khẩu đăng nhập, mật khẩu đó sẽ được băm thành mã băm, sau đó hệ thống sẽ so sánh mã băm đó với mã băm mật khẩu được lưu trong hệ thống. Biện pháp này đảm bảo rằng mật khẩu nguyên gốc của người dùng không bị lộ, ngay cả khi người quản trị hay hacker xem được bản chứa mật khẩu được lưu trong hệ thống.

Mô hình xác thực dựa trên mật khẩu hiện nay đã bộc lộ một số yếu điểm về an ninh. Kẻ tấn công có nhiều phương pháp chiếm quyền kiểm soát của người sử dụng bằng cách đánh cắp mật khẩu:

- Phương pháp đơn giản nhất là kẻ tấn công sẽ sử dụng phương pháp nghe lén đường truyền, nếu mật khẩu không được mã hóa trên đường truyền, kẻ tấn công có thể dễ dàng lấy được mật khẩu của người sử dụng.

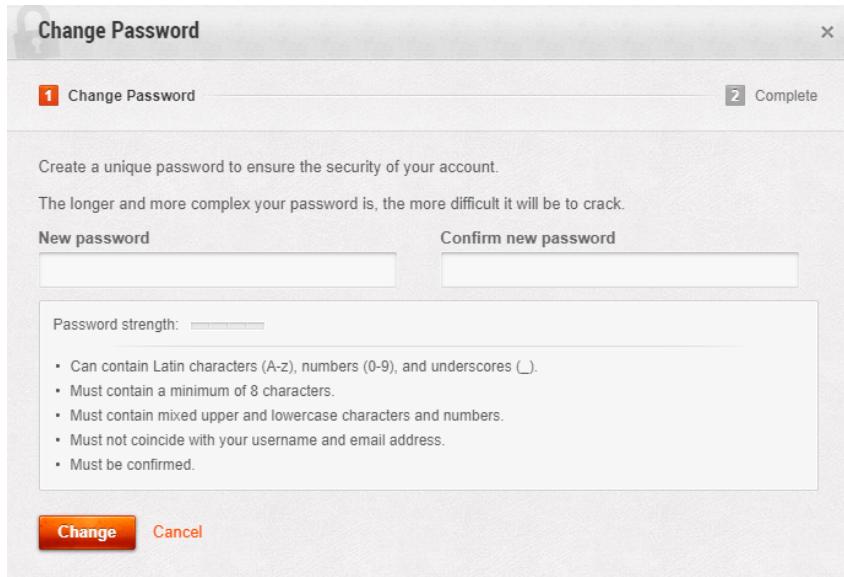
- Nếu mật khẩu đã được mã hóa trên đường truyền, kẻ tấn công có phương pháp khác là sử dụng tấn công bằng từ điển (Dictionary attack), thông qua việc đoán mật khẩu dựa vào một số thông tin như các dạng/kết cấu mật khẩu hay được sử dụng (dạng “123456”, “abc123”,...) và các thông tin cá nhân liên quan có thể có được như tên, tuổi, ngày sinh, số điện thoại, tên người thân cận ... của đối tượng mà kẻ tấn công nhắm tới.

- Sử dụng phương pháp tấn công ngày sinh (Birthday attack) [3] để tìm ra điểm xung đột của thuật toán băm.

Ngoài ra, con người chỉ nhớ được một số lượng bản ghi hữu hạn, họ phải nhớ mật khẩu của nhiều tài khoản khác nhau do đó họ chọn cách sử dụng lại mật khẩu cho nhiều tài khoản khác nhau. Vì vậy nên những kẻ tấn công dễ dàng truy cập tới tài khoản khác khi đã lấy được một tài khoản.

Các kỹ thuật để cải tiến nâng cao độ an toàn trong hệ thống xác thực dựa trên mật khẩu, đó là:

+ Tạo mật khẩu khó đoán. Lý tưởng là sinh mật khẩu ngẫu nhiên. Tuy nhiên mật khẩu ngẫu nhiên quá là khó nhớ nên thường không được dùng. Vì vậy việc chọn đặt mật khẩu của người dùng thông thường có các xu hướng sau: chọn mật khẩu dựa vào các thông tin cá nhân (tên tài khoản, tên người dùng, tên người thân, địa điểm, mã số thẻ các loại, số điện thoại, ngày sinh...), đặt mật khẩu sao cho phát âm được, đọc được (các loại, các ngôn ngữ khác nhau). Tuy nhiên tất cả các xu hướng trên sẽ tạo khả năng cho kẻ tấn công từ điển thành công tăng lên nhiều vì từ điển các mật khẩu có thể chọn theo các xu hướng trên là thu hẹp hơn không gian đầy đủ rất nhiều. Vì vậy các hệ thống có tính bảo mật cao cần phô biến kỹ cho người dùng tầm quan trọng của việc biết chọn mật khẩu tốt, khó đoán. Vấn đề để người dùng hoàn toàn tự quyết chọn mật khẩu cũng dễ đưa đến mật khẩu tồi. Vì vậy trong một số hệ thống người ta đề xuất sử dụng cơ chế “proactive password checking” – kiểm tra mật khẩu chủ động, tức là mật khẩu đã chọn của người sử dụng sẽ được hệ thống kiểm tra đánh giá trước, nếu thấy chưa đủ tốt (theo các thuật toán đánh giá dựa vào một số tiêu chí đã được khảo sát nghiên cứu kỹ), sẽ yêu cầu người sử dụng phải đặt lại mật khẩu khác. Quá trình đó có thể lặp đi lặp lại cho đến bao giờ chương trình đánh giá mật khẩu này chấp nhận mật khẩu mới của người dùng.



## Hệ thống xác thực sử dụng cơ chế kiểm tra mật khẩu chủ động

+ Cơ chế làm chậm tấn công từ điển: Cơ chế này thường được gọi là “thêm muối”, tức hệ thống “trộn thêm” một chuỗi bit ngẫu nhiên vào chuỗi mật khẩu cung cấp của người dùng khi đăng nhập, trước khi tiến hành thủ tục băm và chuyển cho các thao tác kiểm tra tiếp theo. Không gian mật khẩu coi như được nở ra theo hàm mũ nhờ vào việc trộn chuỗi bit ngẫu nhiên (*hay gọi là các bit muối – salt bit*). Trong thực tế chuỗi bit này có thể coi là một tham số khóa của hệ thống và được hệ thống lưu trữ theo tên người dùng. Vì kẻ tấn công hoàn toàn không thể đoán được chuỗi bit này (*ngẫu nhiên*), nên bắt buộc phải thử tất cả các khả năng của nó, dù chỉ là thử một mật khẩu đoán thử nào đó. Vì vậy quá trình tấn công sẽ bị làm chậm  $2^k$  lần, với  $k$  là độ dài chuỗi bit muối.

Ví dụ 2.1: Hệ mật khẩu Vanilla Unix sử dụng cơ chế salt. Hàm băm của nó chính là một biến thể của thuật toán sinh mã DES với 25 vòng lặp, tác động lên thông điệp 0; Tức là giá trị băm của giá trị X sẽ là DESX(0). Bảng biến đổi E trong thuật toán DES cải biến này sẽ có 12 bit tùy chọn, tức là có thể có 4096 version khác nhau. Với việc sử dụng 12 salt bit, rõ ràng kẻ tấn công sẽ phải tốn thời gian thử 1 mật khẩu lâu hơn đến 4096 lần.

+ Cơ chế thu ngắn số lần thử mật khẩu: Có thể tăng thời gian trễ giữa 2 lần thử không thành công theo một hàm tăng nhanh, ví dụ hàm mũ (*Exponential Backoff*) hoặc có thể đặt ngưỡng cho phép gõ sai mật khẩu và bắt dừng khá lâu khi bị vượt ngưỡng, thậm chí tháo bỏ quyền đăng nhập hoặc có thể giảm lỏng (Jailing), tức là đưa vào một môi trường mô phỏng thử nghiệm để nghiên cứu hành vi của kẻ tấn công.

+ Qui định chu kỳ người sử dụng phải thay đổi mật khẩu. Một mật khẩu cũ đến hạn (*quá tuổi sử dụng*) sẽ phải bị thay thế. Người sử dụng sẽ có thời gian để lựa chọn mật khẩu mới (*qua việc nhắc, đếm dần từng ngày, trước khi tiến hành bắt đổi mật khẩu*). Ngược lại, khi đã thay đổi mật khẩu mới, người dùng sẽ bị cấm thay đổi mật khẩu trong một thời gian đủ lâu để có thể đảm bảo sử dụng mật khẩu mới thực sự (*và ghi nhớ được nó*). Điều này cần có để bắt buộc người dùng phải thực sự tôn trọng luật thay đổi mật khẩu đã quá hạn, không thể cố tình đổi phó với qui định để quay về dùng lại mật khẩu cũ một cách dễ dàng.

+ Sử dụng mật khẩu một lần (*One time password*). Nguyên lý của hệ mật khẩu một lần là mỗi mật khẩu được chỉ có hiệu lực 1 lần, trong một khoảng thời gian nhất định. Mật khẩu một lần đảm bảo việc cho dù kẻ tấn công có lấy được mật khẩu trên đường truyền, thì kẻ đó cũng không sử dụng lại mật khẩu đó. Mật khẩu một lần được hệ thống sinh ra theo một thuật toán ngẫu nhiên, hoặc giả ngẫu nhiên, có kích thước cố định và được thông báo lại cho người sử dụng bằng một kênh truyền thông nào đó như qua tin nhắn di động, qua internet, hoặc qua các thiết bị chuyên dụng. , ví dụ như thiết bị Secure ID của hãng bảo mật RSA, hay OTP token của hãng VeriSign.

#### 2.2.1.2. Xác thực bằng mã PIN

Mã PIN (*Personal Identification Number*) là mã số định danh cá nhân là mã số được dùng để xác nhận người dùng. Mã PIN được nhà cung cấp cấp cho người sử dụng. Mã PIN thường được dùng để xác thực khi dùng thẻ tín dụng ngân hàng, ATM, sim...

Mô hình xác thực dựa trên mã định danh cá nhân thường sử dụng các thiết bị vật lý để lưu trữ mã định danh cá nhân như thẻ SmartCard, hay USB Token. Mỗi người sử dụng được cấp một mã cá nhân (PIN) có chiều dài từ 4-10 chữ số, mã cá nhân được bảo vệ trong thiết bị vật lý bằng mật khẩu tự đặt của người sử dụng. Khi đăng nhập vào hệ thống, ở bước xác thực, người dùng phải cầm thiết bị vào máy tính để nhập mã PIN, thông thường khi đó thiết bị sẽ yêu cầu người sử dụng nhập mật khẩu để có thể kích hoạt mã PIN được lưu trong thiết bị. Để đảm bảo mã PIN của người sử dụng được an toàn ngay cả trong trường hợp người sử dụng bị mất, thiết bị token sẽ khóa hoặc tự hủy nếu kẻ tấn công nhập sai mật khẩu sử dụng trong một số lần liên tiếp (Thường là 3 lần hoặc 5 lần).

Đây là phương pháp xác thực cơ bản, đơn giản nhất. Dễ sử dụng và người quản trị dễ dàng trong việc quản lý. Chi phí thấp cho thời gian và kinh phí trong quá trình thiết lập. Tuy nhiên nếu số PIN để kích cỡ quá lớn sẽ làm cho người dùng khó nhớ và việc xác thực bị lỗi. Tương tự vậy, nếu để kích cỡ PIN quá ngắn sẽ dẫn đến số PIN đó dễ bị đoán được gây mất an toàn.

#### 2.2.1.3. Xác thực bằng Passphrase

Passphrase là một chuỗi từ hoặc văn bản khác được sử dụng để kiểm soát quyền truy cập vào hệ thống, chương trình hoặc dữ liệu của máy tính. Passphrase được sử dụng tương tự như mật khẩu, nhưng dài hơn và phức tạp hơn để tăng cường tính bảo mật. Passphrase thường được sử dụng để kiểm soát việc truy cập và hoạt động của các chương trình và hệ thống mật mã, đặc biệt là các công cụ mã hóa mật khẩu.

Mật khẩu thường là từ sáu đến mười ký tự. Các mật khẩu như vậy có thể có thể được sử dụng trong nhiều trường hợp khác nhau: truy cập vào hệ thống máy tính, kích hoạt thẻ thông minh hoặc làm mã PIN cho thẻ ATM. Tuy nhiên, mật khẩu thường không an toàn để sử dụng làm khóa cho các hệ thống an ninh độc lập (ví dụ như các hệ thống mã hoá), dễ bị tin tặc tấn công khai thác mật khẩu. Passphrase về mặt lý thuyết mạnh hơn và do đó là sự lựa chọn tốt hơn trong các trường hợp so với mật khẩu thông thường.

- Độ dài của Passphrase thường từ 20 đến 30 ký tự trở lên, ngăn chặn hiệu quả nhiều hình thức tấn công khai thác mật khẩu.

- Nếu được chọn tốt, Passphrase sẽ không được tìm thấy trong bất kỳ cụm từ nào hoặc từ điển trích dẫn nào, do đó các cuộc tấn công từ điển như vậy sẽ gần như không thể.

- Passphrase có thể được cấu trúc để có thể dễ dàng hơn nhớ so với mật khẩu mà không bị ghi lại, giảm nguy cơ bị tin tặc tấn công ở các hình thức cao hơn.

- Tuy nhiên, Passphrase cần được bảo mật tốt bởi người sử dụng, tránh không bị đánh cắp và tiết lộ. [5]

### **2.2.2. Xác thực dựa trên những gì người sử dụng có**

Xác thực dựa theo những gì người sử dụng có (sở hữu) sẽ dựa vào những gì mà người dùng có, chủ yếu là các đối tượng vật lý (chẳng hạn như thẻ). Tồn tại của phương thức này là không chứng minh được quyền sở hữu vì thẻ/token/điện thoại... dễ dàng bị đánh cắp hoặc được nhân đôi bởi các phương tiện gian lận tinh vi.

#### **2.2.2.1. Thẻ thông minh**

Xác thực theo thẻ là công nghệ dùng để xác thực người dùng bằng cách cấp cho mỗi một người dùng một thẻ riêng dùng để đăng nhập vào một hệ thống, mạng hay máy chủ. Hiện nay được sử dụng khá phổ biến trên thế giới cũng như ở Việt Nam.

Ví dụ như : Key Card, Bank Card, Smart Card, ATM Card... Mật khẩu mà người dùng phải nhớ đó là số PIN. Ở nước ta xác thực người dùng theo thẻ chỉ xuất hiện phổ biến ở các ngân hàng, bằng việc sử dụng các thẻ ATM.



**Thiết bị đọc thẻ của hệ thống xác thực người dùng**

Thẻ thông minh (Smart Card) có kích thước giống như một chiếc thẻ ATM, nhưng được gắn bên trong một con chíp điện tử, có khả năng xử lý như một máy tính thu nhỏ.

Các thiết bị giao tiếp với thẻ (để cấp năng lượng điện và trao đổi dữ liệu với thẻ) có nhiều dạng nhưng thường là đầu đọc thẻ (Card Reader).



### Thẻ thông minh

Chip điện tử được gắn trong thẻ có khả năng lưu trữ, xử lý và thực hiện các nhiệm vụ được yêu cầu giống như máy tính bởi các thành phần chính bên trong con chip điện tử này gồm : Bộ nhớ truy cập ngẫu nhiên (RAM), bộ nhớ chỉ được quyền đọc (ROM), bộ nhớ lưu trữ dữ liệu (EEPROM) và hệ điều hành của thẻ Smard Card. Mặt ngoài của chip được thể hiện trong hình 2.6



### Sơ đồ sắp xếp các điểm tiếp xúc.

VCC : Đầu vào cung cấp nguồn

RST : Tín hiệu Reset cung cấp từ máy đọc hoặc dùng tổ hợp với mạch điều khiển Reset bên trong (tùy theo loại thẻ).

CLK : Tín hiệu xung đồng hồ

GND : Đất

VPP : Đầu vào điện áp lập trình

I/O : Dữ liệu ra hay vào của chip nằm bên trong thẻ.

C4; C8 : Hai điểm tiếp xúc còn lại sẽ được xác định dựa vào chuẩn ứng dụng thích hợp.

Thông thường thẻ nhớ Smart Card có kích thước theo chuẩn ID-1 của ISO/IEC 7810 quy định là 85,60 x 53,98 mm. Một kích thước khác cũng khá thông dụng là ID-000 tức cỡ 25x15mm. Cả hai kích thước này đều có bề dày là 0,76mm.

Điểm quan trọng nhất của thẻ nhớ Smart Card là khả năng bảo mật cao, bởi các thành phần vật lý của con chíp đều ở dạng siêu nhỏ và chúng đều có khả năng chống lại các tấn công vật lý.

Vì không thuận tủy như là một thiết bị nhớ, toàn bộ hoạt động bên trong Smart Card được điều hành bởi bộ vi xử lý bằng chương trình đã được ghi cứng trong ROM khi chế tạo Card. Và như vậy, CPU bảo vệ tài nguyên của Smart Card là những dữ liệu phía trong, mà chỉ có một đường duy nhất giao tiếp với bên ngoài là cổng truyền thông giữa Card và Reader. Như vậy nếu không phải chính CPU của Smart Card, thì không ai có thể làm thay đổi cũng như biết được tài nguyên, chính là dữ liệu phía bên trong Card. Thông tin trong bộ nhớ chỉ đọc ROM được ghi trong quá trình chế tạo chíp, kỹ thuật này làm tăng sự bảo mật của chíp, đó chính là chương trình được mã hóa vĩnh cửu cho Smart Card. Để thẩm tra nội dung của ROM là không thể, kể cả đối với những thiết bị phát hiện rất đắt tiền đi chăng nữa. Không giống như BIOS của PC có thể xem được hay được công bố một phần ở những tài liệu chuyên ngành, thông tin trong ROM của Smart Card có thể coi như bản quyền của chính hãng mà vì lợi nhuận và sự sống còn của hệ thống nên không thể tiết lộ cho ai. Và vì thế nên ta không thể biết CPU sẽ làm gì và như thế nào với những dữ liệu bên trong Card.

Loại bộ nhớ thứ hai trong Smart Card là bộ nhớ bất biến NVM với kích cỡ từ 1 Kb đến 16 Kb EEPROM, có khả năng xóa và ghi lại khoảng 100.000 lần với thời gian lưu trữ thông tin đến 10 năm, đủ để an tâm khi ta cần lưu trữ dữ liệu trên Smart Card. Với kích thước nhỏ bé và bền vững về cơ học, chịu sự quản lý duy nhất bởi CPU của Card, nên độ tin cậy và an toàn của dữ liệu trong bộ nhớ loại này là rất cao. Ngoài hệ điều hành tối thiểu được chất trong ROM, một số phần mềm chuyên dụng hay các chức năng mật mã đặc biệt nạp vào vùng bộ nhớ bất biến này sau khi đã hoàn thành chế tạo Card. Các file trên Smart Card cũng khá phức tạp và đa dạng, tất cả việc tổ chức, bảo mật, quản lý và cấp phát các file trên Smart Card được quy định bởi hệ điều hành và được phân quyền nghiêm ngặt.

Kiến trúc hệ thống phần mềm của Smart Card là một sự tích hợp để đi vào các hệ thống lớn, chia thành từng phần bảo mật để sao cho từng phần riêng lẻ không thể mang lại thông tin gì cho những ai muốn khám phá, phá hoại sự bí mật của toàn hệ thống. Với những hệ thống phức tạp, phần mềm chính của hệ thống không chỉ nằm trên Card mà còn được phân bổ một cách thích hợp trên toàn hệ thống, quyết định chức năng nào nạp vào Smart Card, chức năng nào trên thiết bị đọc Reader, chức năng nào trên trong hệ thống máy tính quản lý mạng cũng là bí mật của các hãng nhằm mục đích tăng cường tính bảo mật của toàn hệ thống. Đây chính là việc phân định phần mềm chủ và phần mềm Card thành các loại phần mềm khác nhau, mỗi phần được phân cấp quyền hạn nhất định, rõ ràng và hoàn toàn độc lập. Phần mềm chủ của Reader ra lệnh bằng các thông điệp quy chuẩn bởi các giao thức truyền thông và lắng nghe sự hồi âm từ phần mềm Card của Smart Card, Smart

Card nhận đúng lệnh và trả lời. Phần mềm chủ không có khả năng khám phá hay thay đổi nội dung của phần mềm Card và ngược lại. Như vậy toàn bộ hệ thống được đóng kín, ta có thể biết lớp phần mềm ứng dụng trên PC, mặc dù cũng rất khó khăn, nhưng khó có thể khám phá phần mềm chủ trên Reader và lại càng không thể biết những gì sẽ xảy ra nội tại trong phần mềm Card. Nhất là khi phần mềm Card được viết bằng ngôn ngữ máy bậc thấp cho chíp thì khả năng bảo mật dữ liệu càng được tăng cường.

Các giao thức liên kết cũng như các giao thức mức ứng dụng cho thấy các quy định phức tạp để có thể thiết lập kênh thông tin đáng tin cậy được truyền đi giữa các thành phần trong hệ thống, và một loạt các chức năng mật đã được định rõ để hạn chế tối đa sự truy nhập tới các phần mềm ứng dụng trên Card hoặc hạn chế truy cập tới hệ thống file và các dữ liệu mật trên Card.

Ngoài ra để nâng cao độ bảo mật của chíp Smart Card người ta thiết kế toàn bộ hệ thống trên 1 chíp duy nhất nhằm tránh việc rò rỉ thông tin giữa các yếu tố, các phần tử khác nhau của chíp trong quá trình trao đổi dữ liệu giữa các yếu tố.

Sự giao tiếp của chíp với bên ngoài thông qua một khói điều khiển giao tiếp duy nhất là công vào ra. Công này có nhiệm vụ canh gác toàn bộ hoạt động của bộ vi xử lý trong chíp thông qua các giao thức viễn thông bậc cao. Bộ vi xử lý lọc toàn bộ các tin tức được đi qua hoặc từ các phần tử khác nhau của chíp, qua giao thức này có thể yêu cầu xác thực, nhận dạng trong chương trình Reader Side đang giao tiếp với Smart Card.

Chức năng an toàn trong xử lý Credit Card còn thể hiện ở chỗ có thấy được hoặc không thấy được của Card định danh để bảo vệ quyền của người sử dụng nhằm tránh bị giả mạo. Tức là khi người sử dụng trái phép khi đưa Card vào các thiết bị đọc Card, các thiết bị sẽ không nhận ra sự có mặt của Card trong thiết bị đọc, đồng nghĩa với khả năng không đọc được dữ liệu từ Card.

Người ta có thể thiết kế để bảo mật dữ liệu cho Smart Card theo nhiều cách. Sử dụng việc bảo mật cứng bằng các khóa cứng được điều khiển bằng bộ vi xử lý tránh việc truy cập bộ nhớ bất hợp pháp, các khóa cứng thường được thiết kế sao cho sự truy cập bộ nhớ kể cả bộ vi xử lý Card cũng đều phải thông qua khóa cứng đó hoặc có thể sử dụng phần mềm kiểm tra quyền truy cập Smart Card bằng việc kiểm tra mật khẩu truy cập hoặc mã hóa dữ liệu được ghi trên Smart Card bằng thuật toán xác định... Như vậy những người không có thẩm quyền sẽ không thể truy cập bộ nhớ của Smart Card để lấy cắp thông tin dữ liệu và chương trình được lưu trên Card.

Chúng ta có thể sử dụng biện pháp xáo trộn BUS và các ô nhớ nhằm tránh không cho những người không có thẩm quyền thâm nhập bộ nhớ hoặc lấy cắp dữ liệu. Phương pháp này người ta tiến hành ngẫu nhiên hóa tần số truy nhập vào các ô nhớ bằng chương trình trong bộ vi xử lý, tức là cá đường địa chỉ của các ô nhớ khác nhau không xuất phát trong một chuỗi tuyến tính, sự biến đổi từ ô nhớ này sang ô nhớ khác phải thông qua một vài thuật toán phức tạp nào đó. Người không có thẩm quyền sẽ không thể biết được bất kỳ một thông tin nào về nơi chứa dữ liệu và cách sử dụng dữ liệu nếu chỉ đơn giản bằng việc quan sát tần số truy cập tần số ô nhớ riêng nào đó. Ngoài việc làm ngẫu nhiên các

tần số truy cập bộ nhớ để tránh việc làm giả mạo quyền người sử dụng ta có thể mã hóa chuỗi bít và kết hợp sử dụng khóa bí mật vào việc mã hóa. Khi đó chỉ có thiết bị đọc của ta mới có thể nhận ra Card và kết nối xử lý thông tin trên Card. Phương pháp này tiến hành như sau :

+ Đầu tiên khi ta đưa Card vào thiết bị ghép nối thiết bị sẽ gửi một xung tín hiệu gồm một chuỗi bít và một khóa đến Smart Card. Card tiến hành mã hóa chuỗi bít và khóa bằng khóa bí mật của Smart Card đồng thời gửi trả lại thiết bị ghép nối. Thiết bị ghép nối có khóa giải mã, giải mã và so sánh với khóa và chuỗi bít đã gửi nhằm xác thực Card, đây là phương pháp xác thực bên ngoài.

Như vậy có rất nhiều phương pháp nhằm đảm bảo thông tin trên Card tránh việc giả mạo người sử dụng và lấy cắp thông tin trên Card.

Ưu điểm:

+ Xác thực người dùng bằng thẻ là giải pháp mang tính kinh tế cho các tổ chức.

+ Người dùng muốn đăng nhập vào tài khoản của mình thì phải dùng thẻ, và việc làm giả thẻ khó khăn nhất là đối với thẻ chip. Vì vậy những kẻ muốn tấn công vào tài khoản của người khác, phải có thẻ mới có thể tấn công được.

Nhược điểm:

+ Mật khẩu người dùng là số pin chỉ người dùng biết mới có thể thay đổi. Nhưng do là người dùng thường đặt mật khẩu đơn giản và dễ đoán hoặc do đặt phức tạp lên thường ghi ra đâu đấy lên dễ bị lộ.

+ Bởi vì hệ thống sử dụng các thiết bị vật lí để đọc thẻ, cả thẻ và các thiết bị vật lí này có giá rất đắt, hơn nữa các thiết bị này cần phải sử dụng nhiều trên một phạm vi rộng nên chi phí cho hệ thống xác thực người dùng bằng thẻ là tốn kém. Vì vậy mà hệ thống xác thực người dùng bằng thẻ bây giờ vẫn chưa sử dụng rộng rãi ngoài khu vực ngân hàng.

+ Xác thực theo thẻ, là công nghệ để xác thực người dùng muốn đăng nhập vào một hệ thống, mạng hay máy chủ, được sử dụng khá phổ biến hiện nay trên thế giới cũng như ở Việt Nam. Ví dụ như : key card, bank card, smart card, ATM card... Mật khẩu mà người dùng phải nhớ đó là số PIN. Xác thực người dùng bằng thẻ là giải pháp mang tính kinh tế cho các hệ thống, tổ chức. Mỗi một người dùng sẽ có thẻ riêng, với lần đầu tiên dùng thẻ hệ thống sẽ sinh ngẫu nhiên cho người dùng số PIN. Trên mỗi thẻ sẽ lưu một seed duy nhất (key) và key này cũng được lưu trên cơ sở dữ liệu của server. Key này sẽ được mã hóa bằng những thuật toán mã hóa như: DES, 3DES, AES 128-bit hoặc 192-bit và 256-bit. Khi người dùng đăng nhập vào hệ thống phải đưa thẻ vào thiết bị nhập thẻ sau đó nhập số PIN nếu đúng thì người dùng sẽ đăng nhập thành công, nếu sai người dùng sẽ phải thực hiện lại quá trình trên. Thông thường sau mười lần liên tục người dùng đăng nhập không thành công hệ thống sẽ khoá tài khoản của người dùng. Mật khẩu của người dùng là số PIN, người dùng có thể thay đổi mật khẩu bất cứ lúc nào với độ dài mật khẩu do người dùng lựa chọn. Mật khẩu có thể là các kí tự, số. . . Người dùng tránh đặt những mật khẩu quá dễ nhớ.

+ Hệ thống xác thực sử dụng công nghệ này yêu cầu phải có các thiết bị vật lý dùng để đọc thẻ và phải được kết nối với hệ thống máy tính để quản lý, đặt ở nhiều nơi thuận tiện cho người dùng sử dụng. Dưới đây là một số hình ảnh về thẻ mà người dùng sử dụng:

#### 2.2.2.2. Token và mật khẩu OTP

Token là những phương tiện vật lý chứa bộ tạo mật khẩu dùng một lần OTP.

Mật khẩu một lần (OTP – One Time Password) là một mật khẩu có giá trị chỉ cho một phiên đăng nhập hoặc giao dịch, trên một hệ thống máy tính hoặc thiết bị kỹ thuật số khác. OTP thường được tạo ra dựa trên các thông tin đã chia sẻ trước giữa hai bên xác thực, hoặc các sự kiện diễn ra đồng thời ở cả hai bên.

##### a. Sinh và phân phối OTP:

Các thuật toán sinh OTP điển hình thường sử dụng pseudorandomness hoặc randomness, làm cho việc dự đoán OTP của kẻ tấn công trở lên khó khăn.Thêm vào đó với các tính năng băm (hash functions) sẽ rất khó để sử dụng các giá trị để đảo ngược (reverse) và do đó sẽ càng khó khăn hơn nữa cho kẻ tấn công muốn có được những dữ liệu đã được băm. Điều này là cần thiết vì nếu không nó sẽ được dễ dàng để dự đoán các giá trị của OTP sinh tiếp theo bằng cách quan sát mã trước đây. Thuật toán OTP khác nhau rất nhiều chi tiết của họ. Các phương pháp khác nhau để sinh OTP:

\* Dựa vào đồng bộ hóa thời gian giữa các máy chủ xác thực và máy trạm cung cấp mật khẩu (OTP chỉ có hiệu lực trong một thời gian ngắn)

- Một OTP sinh bởi đồng bộ thời gian thường liên quan đến phần cứng được gọi là Token (*ví dụ, mỗi người sử dụng được nhận một Token cá nhân sẽ sinh ra một mật khẩu một lần (one-time password)*). Trên các hệ thống OTP, thời gian là một phần quan trọng của các thuật toán mật mã, từ khi mật khẩu mới được sinh ra dựa trên thời gian hiện thời chứ không phải là mật khẩu được thêm mới vào, mật khẩu trước đó hay một khóa bí mật. Thiết bị Token này có thể gọi là một thiết bị độc quyền, hay như một điện thoại di động tương tự chạy phần mềm đó là độc quyền, duy nhất chứ không phải là phần mềm miễn phí, mã nguồn mở. Một ví dụ về OTP tiêu chuẩn đồng bộ thời gian là thuật toán sinh mật khẩu một lần dựa vào thời gian Time-based One-time Password Algorithm (TOTP).

- Loại đồng bộ thời gian tạo ra mã số khó đoán (*mật mã hay khóa*) dựa vào đồng hồ trong và mã số này được xác thực với điều kiện đồng hồ trong của thiết bị OTP đồng bộ với máy chủ xác thực. Do sự xê dịch của đồng hồ, việc đồng bộ tuyệt đối chính xác là không thể nên máy chủ xác thực phải chấp nhận các khóa có sự sai lệch đôi chút. Điều quan trọng đó là thu hẹp hết mức “khung cửa” này để giảm thiểu khả năng bị tấn công. Đa phần các nhà cung cấp thiết bị OTP áp dụng phương thức cộng đồng thời gian xê dịch để điều chỉnh với mỗi xác thực thành công. Thiết bị OTP đồng bộ thời gian có thể phải cân chỉnh lại nếu không được sử dụng một thời gian dài

\* Sử dụng một thuật toán toán học để tạo ra một mật khẩu mới dựa trên mật khẩu trước đây (*OTP là một chuỗi và phải được sử dụng theo một thứ tự xác định trước*).

Một ví dụ cho thuật toán này là thuật toán của Leslie Lamport – thuật toán sử dụng các hàm một chiều f. Hệ thống OTP làm việc dựa trên một giá trị mầm khởi tạo s để sinh mật khẩu lần đầu tiên. Mật khẩu OTP được sử dụng cho phiên làm việc đầu tiên sẽ được tính như sau:

$$\text{OTP}_1 = f^N(s)$$

Trong đó  $f^N(s) = f(f^{N-1}(s))$  – với N là số lần áp dụng hàm f lên giá trị s.

Khi đó ta sẽ có lần lượt các mật khẩu OTP cho các phiên như ở bảng dưới đây

Phiên 1	Phiên 2	Phiên 3	Phiên 4
$\text{OTP}_1 = f_N(s)$	$\text{OTP}_2 = f_{N-1}(s)$	$\text{OTP}_3 = f_{N-2}(s)$	$\text{OTP}_4 = f_{N-3}(s)$

Nếu một kẻ tấn công nào đó có thể bắt được giá trị OTP của một phiên làm việc nào đó thì hắn có thể dùng để đăng nhập vào hệ thống khi mà giá trị OTP này còn hợp lệ, tuy nhiên khi OTP không còn hợp lệ nữa thì hắn không thể thực hiện việc đăng nhập nữa và để lấy được giá trị OTP cho lần đăng nhập kế tiếp thì hắn sẽ phải đổi mặt với việc tính hàm f ngược. Do hàm f đã được chọn là hàm một chiều nên việc này là cực kì khó để thực hiện. Nếu hàm f là một hàm băm mật mã thì việc tính toán là cực kì khó khăn thậm chí là không thể.

\* Sử dụng một thuật toán toán học để tạo ra một mật khẩu mới không dựa trên mật khẩu trước, các mật khẩu mới được sinh dựa trên một yêu cầu (ví dụ, một số ngẫu nhiên được lựa chọn để xác thực máy chủ hay trong giao dịch nào đó) và một bộ đếm.

### b. Phương pháp phân phối OTP

#### \* Tin nhắn văn bản

Một công nghệ phổ biến nhất thường được sử dụng cho việc cung cấp các chương trình OTP là tin nhắn văn bản. Bởi vì gửi tin nhắn văn bản là một kênh truyền thông phổ biến, được sử dụng trực tiếp có sẵn trong hầu hết các điện thoại di động, với bất kỳ điện thoại di động hoặc điện thoại cố định, tin nhắn văn bản có một tiềm năng rất lớn để đạt được tất cả người tiêu dùng với chi phí thấp để thực hiện. Tuy nhiên, cần phải lưu ý chi phí của tin nhắn văn bản cho mỗi OTP và khả năng mã hóa sử dụng một thuật toán A5 / x không được mạnh. Một số báo cáo cho thấy một số nhóm Hacker thông báo có thể giải mã thành công trong vòng vài phút hoặc vài giây, hoặc OTP qua SMS có thể không được mã hóa bởi một số nhà cung cấp dịch vụ. Ngoài mối đe dọa từ tin tặc, các nhà điều hành điện thoại di động trở thành mặt xích quan trọng việc đảm bảo an toàn. Trong trường hợp chuyển vùng (roaming) giữa các nhà điều hành điện thoại di động có thể bị tấn công man-in-the-middle

Trong năm 2011, Google đã bắt đầu cung cấp OTP đến điện thoại di động và điện thoại cố định cho tất cả các tài khoản Google. Người dùng có thể nhận được OTP hoặc như là một tin nhắn văn bản hoặc thông qua một cuộc gọi tự động bằng cách sử dụng chuyển đổi text-to-speech.

#### \* Điện thoại di động

Một điện thoại di động sẽ giúp chi phí thấp vì một số lượng lớn khách hàng đã sở hữu một chiếc điện thoại di động cho các mục đích khác nhau tạo ra các chương trình OTP. Khả năng tính toán và lưu trữ cần thiết cho chương trình OTP là thường không đáng kể so với các thiết bị điện thoại và điện thoại thông minh hiện đại thường sử dụng. Điện thoại di động hỗ trợ bất kỳ con số của Token với chỉ một cài đặt ứng dụng, cho phép người dùng khả năng để xác thực cho nhiều tài nguyên từ một thiết bị. Giải pháp này cũng cung cấp các ứng dụng với mô hình cụ thể tới điện thoại di động của người dùng.



### Xác thực MOTP

#### \* Token độc quyền

Hình thức phân phối này chủ yếu dành cho phương pháp sinh OTP dựa trên thời gian. Khách hàng sẽ được cung cấp một thiết bị etoken – thiết bị này cũng được cài đặt thuật toán sinh OTP như ở trên server và đã được đồng bộ hóa thời gian với server. Nó có thể trông giống như một máy tính nhỏ hay móc chìa khóa, với một màn hình LCD nhỏ, thường xuyên thay đổi các con số. Bên trong Token có một chiếc đồng hồ đã được đồng bộ thời gian với đồng hồ trên một máy chủ xác thực cá nhân.



### Xác thực TOTP

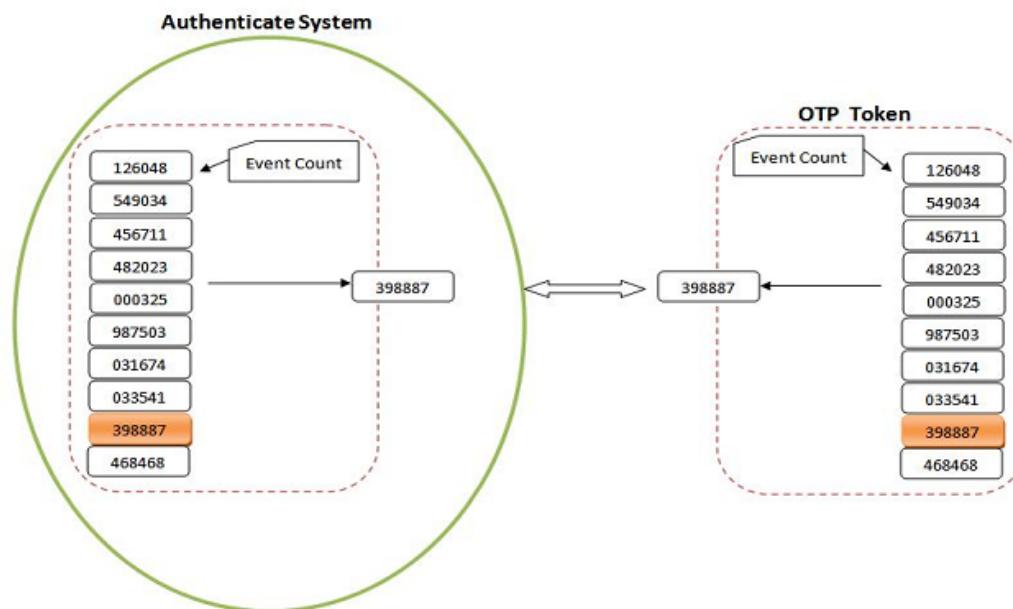
#### c. Một số tiêu chuẩn

Nhiều công nghệ OTP được cấp bằng sáng chế. Điều này làm cho tiêu chuẩn hóa trong khu vực này khó khăn hơn, vì mỗi công ty sẽ cố gắng để thúc đẩy công nghệ riêng của mình. Tiêu chuẩn làm, tuy nhiên, tồn tại - ví dụ, RFC 1760 ( S / KEY ), RFC 2289 (OTP), RFC 4226 ( HOTP ) và RFC 6238 (TOTP).

Có hai tiêu chuẩn chính để sinh ra mật khẩu một lần đó là: HOTP (RFC 4226) và TOTP (6238), cả hai tiêu chuẩn này đều tuân theo chuẩn xác thực mở OATH (Initiative For Open Authentication).

#### \* Tiêu chuẩn RFC 4226 ( HOTP )

Tiêu chuẩn này dựa vào thuật toán Hmac-based One-Time Password (HOTP) dựa vào yếu tố chính: Thứ nhất là khóa chia sẻ (shared secret) và 1 bộ đếm (moving factor). Một phần của thuật toán là hàm băm HmacSHA1 (nói chính xác là băm đoạn tin nhắn dựa trên mã xác thực) của bộ đếm sẽ sinh ra khóa chia sẻ.



#### Sơ đồ đồng bộ hóa HOTP

Thuật toán HOTP là thuật toán sinh OTP dựa trên sự kiện, nghĩa là bất cứ khi nào một OTP mới được sinh, bộ đếm (moving factor) sẽ tăng, kể từ đó các mật khẩu sinh ra sau đó sẽ hoàn toàn riêng biệt.

- K là khóa chia sẻ
- C là bộ đếm

$$\text{HMAC}(K,C) = \text{SHA1}(K \oplus 0x5c5c\dots \parallel \text{SHA1}(K \oplus 0x3636\dots \parallel C))$$

là một phép tính HMAC (Hash-based message authentication code) với thuật toán băm SHA-1

- Truncate là 1 hàm lựa chọn 4 bytes từ kết quả của HMAC ở trên
- Sau đó thuật toán HOTP(K,C) sẽ có được là:

$$\text{HOTP}(K,C) = \text{Truncate}(\text{HMAC}(K,C)) \& 0x7FFFFFFF$$

với Mask: 0x7FFFFFFF là kết quả của MSB (most significant bit) đến 0

- Để cho HOTP có giá trị riêng trước khi nhập vào hệ thống, kết quả trên cần được biết đỗi trong một giá trị HOTP tùy thuộc vào số thực thi có 6-8 chữ số:

$$\text{HOTP-Value} = \text{HOTP}(K, C) \bmod 10d$$

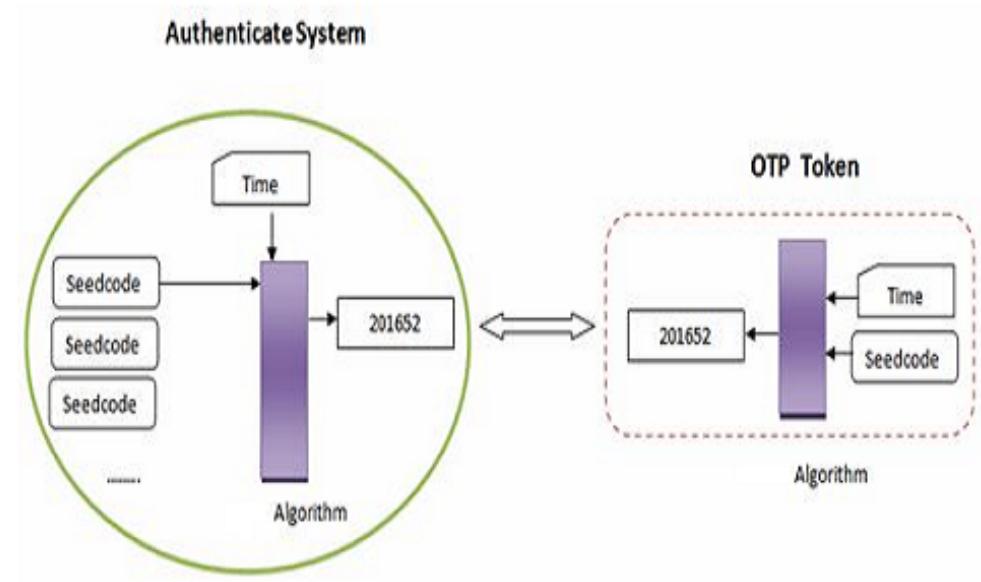
Với d là chiều dài của số muốn tìm

### Phân phối thẻ Tokens

Có cả thiết bị Token bằng phần cứng và phần mềm được hỗ trợ với nhiều hãng khác nhau. Các tokens phần cứng phổ biến được dùng với chuẩn OATH HOTP với nhiều các ưu điểm. Các phần mềm token gần đây cũng sẵn có nhiều trên các nền tảng thiết bị điện thoại di động thông minh như: Android, iPhone, Blackberry, Windows Mobile...

#### \* Tiêu chuẩn RFC 6238 ( TOTP )

Tiêu chuẩn này dựa vào thuật toán Time-based One-Time Password (TOTP), nó có phương thức hoạt động tương tự HOTP và cũng có hai nhân tố chính là khóa chia sẻ (shared secret) và một bộ đếm (moving factor). Tuy nhiên bộ đếm ở đây hoạt động có một chút khác biệt.



### Sơ đồ đồng bộ hóa TOTP

Trong thuật toán TOTP, bộ đếm thay đổi dựa theo thời gian trôi qua từng giai đoạn nghĩa là nó sẽ tự thay đổi bộ mã OTP tự động. Hàm băm HmacSHA1 cũng được tính toán tương tự như HOTP

- TOTP cơ bản dựa vào HOTP với timestamp thay đổi khi bộ đếm tăng.
- Timestamp hiện tại được lần lượt tích hợp vào thời gian của bộ đếm Time-counter (TC) bởi một điểm bắt đầu (T0) và đếm đến 1 đơn vị của Time step (TS)

Ví dụ:

$$TC = (\text{unixtime}(\text{now}) - \text{unixtime}(T0)) / TS$$

$$\text{TOTP} = \text{HOTP}(\text{SecretKey}, TC), \text{ với HOTP là thuật toán được tính ở dưới}$$

$$\text{TOTP-Value} = \text{TOTP} \bmod 10d$$

Với d là chiều dài của số muôn tám (OTP)

- Tính HOTP:

Với K là khóa chia sẻ

C là bộ đếm

$$\text{HMAC}(K,C) = \text{SHA1}(K \oplus 0x5c5c\dots \parallel \text{SHA1}(K \oplus 0x3636\dots \parallel C))$$

là một phép tính HMAC (Hash-based message authentication code) với thuật toán băm SHA-1

- Truncate là một hàm lựa chọn bốn bytes từ kết quả của HMAC ở trên
- Sau đó thuật toán HOTP(K,C) sẽ có được là:

$$\text{HOTP}(K,C) = \text{Truncate}(\text{HMAC}(K,C)) \& 0x7FFFFFFF$$

với Mask: 0x7FFFFFFF là kết quả của MSB (most significant bit) đến 0

- Dựa vào tiêu chuẩn RFC 6238, việc thực thi TOTP gồm các bước sau:
  - + Sinh một khóa K, với một đoạn mã tùy ý và chia sẻ nó an toàn đến các client
  - + Cho phép chạy từ T0 với một khoảng TI, và sử dụng để tính toán giá trị bộ đếm C (Mặc định trong Unix khởi đầu là T0 và khoảng 30 giây là TI)
  - + Cho phép chạy phương pháp mã băm (Mặc định là SHA-1)
  - + Cho phép tính độ dài của Token, N (Mặc định là 6)

#### d. So sánh giữa HOTP và TOTP

Sự khác nhau chính giữa HOTP và TOTP là mật khẩu của HOTP có giá trị trong thời gian vô thời hạn, tuy vào nhu cầu của người sử dụng, trong khi đó mật khẩu của TOTP chỉ có giá trị trong khoảng thời gian rất ngắn trước khi tự thay đổi. Bởi vì vậy, theo quan điểm chung trong giải pháp xác thực bằng mật khẩu một lần thì TOTP được coi là an toàn hơn HOTP.

#### e. Ưu điểm nhược điểm

##### \* Ưu điểm

- An toàn: Giải quyết tốt các vấn đề giả mạo, đánh cắp, Key Logger. Đối với hai yếu tố xác thực, thiết bị này có thể được kết hợp với một mã PIN hoặc mật khẩu.

- Dễ dàng: Việc nhận dạng và xác thực được thực hiện trong vài giây tránh được nguy cơ bị lỗi khi gõ các mã OTP dài qua các mã từ một thiết bị chứng thực vào một máy tính (Ví dụ OTP Token sử dụng màn hình hiển thị). Nó hoạt động với tài nguyên và đăng nhập được trên tất cả các nền tảng máy tính và trình duyệt không cần phần mềm cài đặt Client. Nhanh chóng và tích hợp dễ dàng vào bất kỳ ứng dụng web nào (Windows, Linux, Mac, Internet Explorer, Firefox...).

- Linh hoạt: Dễ dàng di chuyển giữa các máy tính. Dễ mang theo với bạn, trong ví của bạn.

- Mã nguồn mở: Sẵn sàng tích hợp với nhiều ứng dụng mã nguồn mở.

\* Nhược điểm

- Bởi vì hệ thống sử dụng phần cứng lên chi phí đắt hơn các hình thức xác thực khác.
- Quá trình thiết kế, cài đặt và quản trị phức tạp và khó khăn hơn các phương pháp khác.

### **2.2.3. Xác thực dựa trên những gì người sử dụng sở hữu bẩm sinh**

Những nhân tố con người sở hữu bẩm sinh như dấu vân tay hoặc mẫu dạng võng mạc mắt, chuỗi ADN, mẫu dạng giọng nói, tín hiệu sinh điện đặc thù do cơ thể sống tạo ra. Những nhân tố này được gọi là định danh sinh trắc học (biometric identifier).

Xác thực dựa trên các đặc điểm sinh trắc học (hay nói gọn hơn là “xác thực sinh trắc học”) hoạt động dựa trên việc phân tích cả các đặc điểm cơ thể lẫn các đặc điểm hành vi, nhận thức để có thể định danh một người nào đó. Các đặc điểm cơ thể được xác định dựa trên việc đo một phần nào đó của cơ thể, chẳng hạn khuôn mặt, vân tay, móng mông mắt... Các đặc điểm hành vi được dựa trên những dữ liệu dẫn xuất từ các hành động của con người, chẳng hạn như giọng nói, hình dáng... Bảng 2.1 dưới đây liệt kê các đặc điểm cơ thể và hành vi có thể được sử dụng trong các hệ thống xác thực sinh trắc học:

#### **Đặc điểm cơ thể và hành vi có thể được sử dụng trong các hệ thống xác thực sinh trắc**

<b>Đặc điểm thân thể</b>	<b>Đặc điểm hành vi</b>
Chữ ký sinh học	Theo dõi chuyển động của mắt
Trường điện sinh học	Nhận dạng tay nắm động
Dấu cẩn	Dáng đi
Nhip tin	Chữ viết tay
Bề mặt giác mạc	Lực gõ phím
Hình học nha khoa	Lực di chuột
DNA	
Tai	
Hình học khuôn mặt (2D/3D)	
Biểu đồ nhiệt khuôn mặt	
Hình học ngón tay	
Vân tay	
Hình học bàn tay	
Biểu đồ nhiệt độ tay	
Mô hình tĩnh mạch tay	
Nếp gấp đốt ngón tay	
Môi	
Móng tay	
Mùi	
Sự phản xạ của sóng âm trong đầu	
Mô hình võng mạc	
Trở kháng da	
Mẫu da	
Quang phổ da	
Nụ cười	
Giọng nói	
Móng mông	

Theo đánh giá của Jain và một số tác giả khác, xác thực sinh trắc học thường sử dụng các đặc điểm chính như khuôn mặt, móng mắt, vân tay, mạch máu ngón tay, giọng nói... Độ tin cậy, độ an toàn cũng như chi phí cho mỗi phương pháp xác thực dựa trên các đặc điểm đó được thể hiện trong bảng 2.2 dưới đây:

### So sánh một số đặc điểm sinh trắc

Đặc điểm	Độ an toàn	Tính tiện lợi	Chi phí
Vân tay	Tốt	Trung bình	Trung bình
Khuôn mặt	Trung bình	Tốt	Thấp
Móng mắt	Tốt	Tốt	Cao
Mô hình tĩnh mạch	Tốt	Trung bình	Trung bình
Tiếng nói	Trung bình	Tốt	Thấp
Chữ ký	Không tốt	Trung bình	Trung bình
Hình học bàn tay	Trung bình	Trung bình	Trung bình

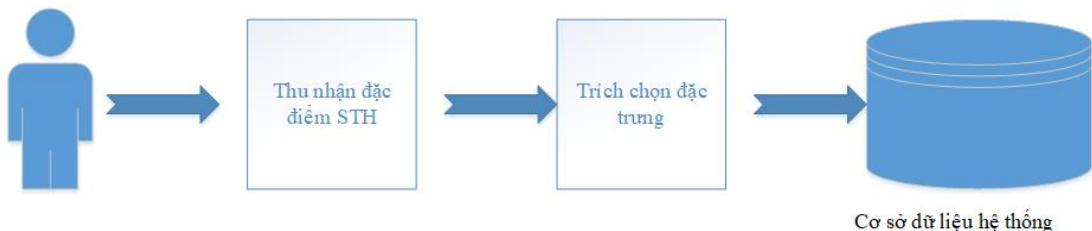
### Mô hình chung của hệ thống xác thực sinh trắc

Các hệ thống xác thực sinh trắc học nói chung đều hoạt động dựa trên các thành phần cho phép nhận biết mẫu (pattern recognition) đặc tả các đặc điểm sinh trắc học của người. Tùy thuộc vào cách thức để nhận biết một cá thể mà người ta chia các hệ thống đó thành hai loại: hệ thống kiểm tra (verification) và hệ thống nhận diện (identification).

- Hệ thống kiểm tra một cá thể hoạt động dựa trên quá trình so sánh những đặc điểm sinh trắc đo được bởi các thành phần nhận dạng của hệ thống, sau đó so sánh theo kiểu một – một với các đặc điểm của người đó đã được lưu trữ trong hệ thống. Kết quả so sánh sẽ cho phép hệ thống xác nhận hoặc phủ nhận người muốn khai báo danh tính (tức cho phép trả lời câu hỏi: liệu A có phải là người đã khai báo là A hay không?)

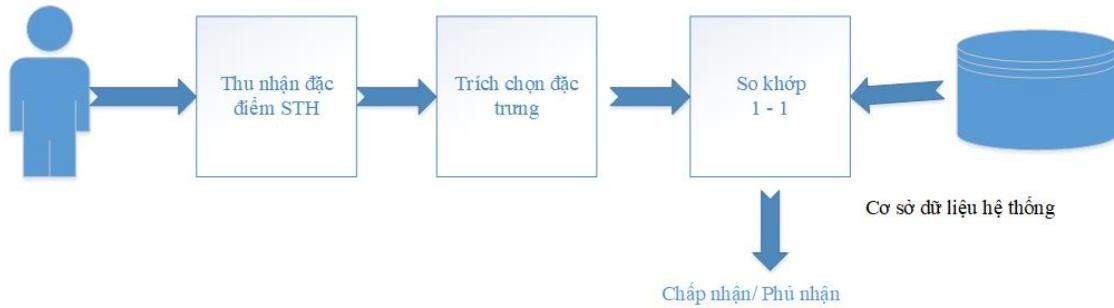
- Với hệ thống nhận diện, hoạt động của nó dựa trên việc so sánh các đặc điểm sinh trắc học của người cần nhận diện với mẫu của toàn bộ người đã lưu trong hệ thống. Ở đây, kiểu so sánh này là một – nhiều, và kết quả so sánh sẽ xác định được định danh của người đó (tức cho phép trả lời câu hỏi tôi là ai?) mà không cần người đó phải khai báo thông tin.

Như vậy, cả hai loại hệ thống xác thực đều cần phải có một cơ sở dữ liệu chứa thông tin các đặc điểm sinh trắc học của một tập người dùng. Việc xây dựng cơ sở dữ liệu này có thể được mô hình hóa bằng sơ đồ dưới đây:

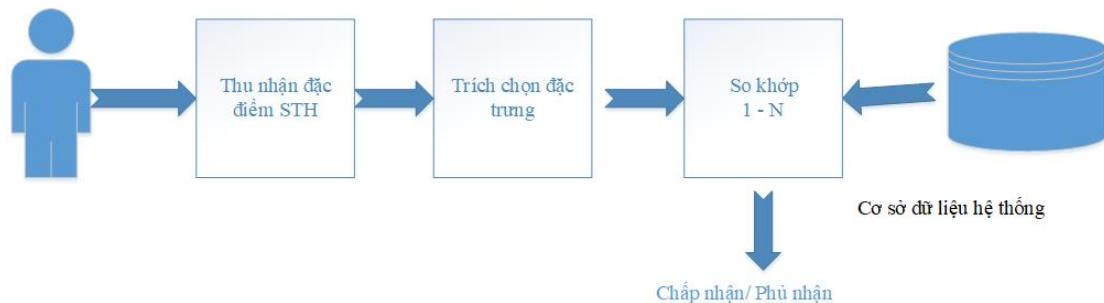


### Sơ đồ thu thập dữ liệu sinh trắc học

Quá trình kiểm tra hay nhận diện được thể hiện tổng quát trong hai hình dưới đây:



## Mô hình hệ thống kiểm tra



## Mô hình hệ thống nhận diện

Trong thực tế, người ta chỉ quan tâm đến việc xác định danh tính của một cá thể nào đó và không quan tâm đến hệ thống đó thuộc kiểu kiểm tra hay nhận diện. Chính vì vậy, trong nội dung tài liệu này, khái niệm xác thực sinh trắc học được hiểu là bao hàm cả hai loại kiểm tra và nhận diện. Trong các mục sau, tài liệu trình bày một số phương pháp gần đây nhất được ứng dụng trong xác thực sinh trắc học. Các phương pháp này cũng được liệt kê dựa trên đặc điểm cơ thể (*sinh trắc học tiêu chuẩn*) và các đặc điểm hành vi (*sinh trắc học hành vi*).

### 2.2.3.1. Sinh trắc học tiêu chuẩn

#### 1. Xác thực thông qua nhận dạng vân tay

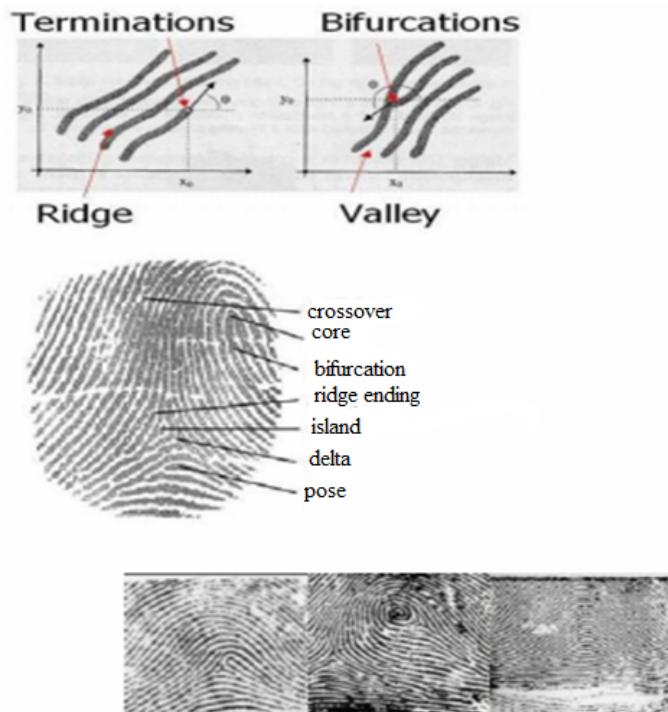
Nhận dạng vân tay trong xác thực sinh trắc học là quá trình so sánh vân tay của một cá thể với một hay một tập vân tay mẫu đã được lưu trữ trong hệ thống.

Vân tay là một trong những đặc điểm khá đặc biệt của con người bởi vì tính đa dạng của nó, mỗi người sở hữu một dấu vân tay khác nhau, rất ít trường hợp những người có dấu vân tay trùng nhau.

##### a. Cấu tạo, đặc điểm của vân tay:

Dấu vân tay của mỗi cá nhân là độc nhất. Xác suất hai cá nhân - thậm chí ngay cả anh em (hoặc chị em) sinh đôi cùng trứng - có cùng một bộ dấu vân tay là  $1/64$  tỉ. Ngay cả các ngón trên cùng bàn tay cũng có vân khác nhau. Dấu vân tay của mỗi người là không đổi trong suốt cuộc đời. Người ta có thể làm phẫu thuật thay da ngón tay, nhưng chỉ sau một thời gian dấu vân tay lại được hồi phục như ban đầu.

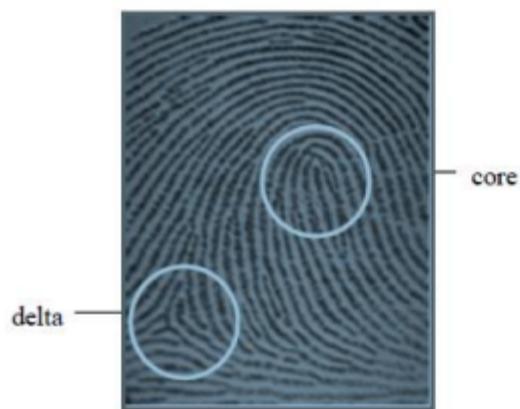
Vân tay là những đường có dạng dòng chảy có trên ngón tay người. Nó là một tham số sinh học bất biến theo tuổi tác đặc trưng cho mỗi cá thể. Trên vân tay có các đường gợn và các luống.



## Cấu tạo vân tay

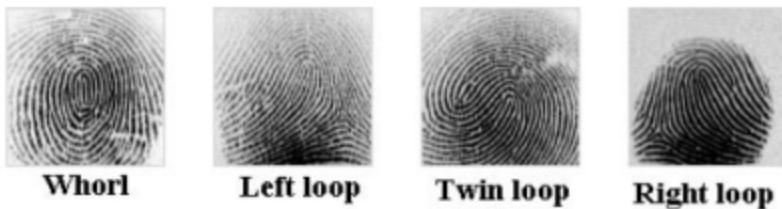
Trên các ảnh vân tay có các điểm đặc trưng (là những điểm đặc biệt mà vị trí của nó không trùng lặp trên các vân tay khác nhau) được phân thành hai loại: singularity và minutiae:

- *Singularity*: Trên vân tay có những vùng có cấu trúc khác thường so với những vùng bình thường khác (thường có cấu trúc song song), những vùng như vậy gọi là singularity. Có hai loại singularity là core và delta.



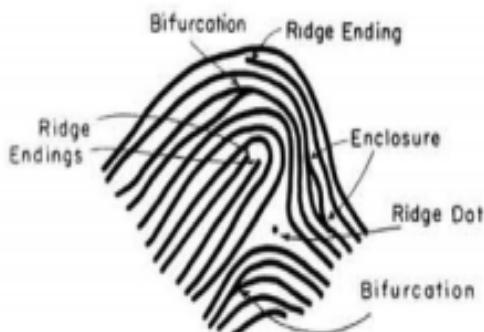
## Các điểm singularity

Core thường có một số dạng như sau:



### Các dạng core

- Minutiae: Khi dò theo từng đường vân ta sẽ thấy có những điểm đường vân kết thúc (Ridge Ending) hoặc rẽ nhánh (Bifurcation), những điểm này được gọi chung là minutiae.



### Các điểm minutiae

b. Phương pháp nhận dạng vân tay:

Có ba phương pháp chính áp dụng cho việc so sánh này là:

- So sánh tương quan – Correlation-based matching: với phương pháp này, ảnh của hai vân tay cần so sánh sẽ được xếp chồng để từ đó có thể tính được tương quan của hai ảnh đó.

- So sánh dựa trên các chi tiết đặc biệt – Minutiae-based matching: với phương pháp này, vân tay của một người sẽ được phân tích để từ đó xác định được những điểm đặc trưng nhất. Từ đó, việc so sánh hai vân tay quy về việc so sánh giữa các điểm đặc trưng đó với nhau.

- So sánh dựa trên hình dáng vân – ridge feature-based matching: với những ảnh vân tay chất lượng xấu, việc xác định các điểm đặc trưng trở nên khó khăn hơn. Lúc đó, việc phân tích các hình dáng, đặc trưng vân sẽ trở nên thích đáng hơn cho việc so sánh hai mẫu vân tay.

c. Xây dựng hệ thống nhận dạng vân tay:

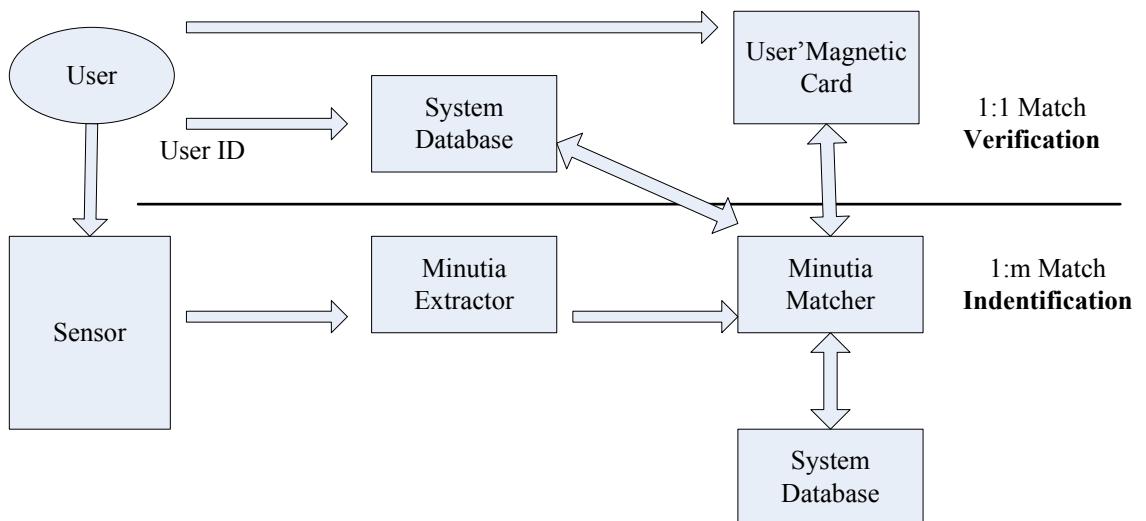
#### \* Quá trình nhận dạng vân tay

- Xác nhận dấu vân tay (*fingerprint verification*): Một người sẽ cung cấp dấu vân tay cùng với chứng minh thư, hoặc là các đặc điểm cá nhân của người đó, ví dụ như Họ tên, ngày sinh, quê quán... (*trong chứng minh thư*) hoặc là username, tên tài khoản, các quyền hạn của người đó, ... (*trong bảo mật*). Bước này nhằm tạo ra một cơ sở dữ liệu tương ứng dấu vân tay và các đặc điểm liên quan. Nguyên lý cơ bản của hệ thống này là sử dụng các diot phát sáng để truyền các tia gần hồng ngoại NIR (Near Infrared) tới ngón tay và chúng sẽ được hấp thụ lại bởi hồng cầu trong máu. Vùng các tia bị hấp thụ trở thành vùng tối trong hình ảnh và được chụp lại bởi camera CCD (Charge Coupled Device). Sau đó, hình ảnh được xử lý và tạo ra mẫu vân tay. Mẫu vân tay được chuyển đổi thành tín hiệu số và là dữ liệu để nhận dạng người sử dụng.

+ Công nghệ truyền ánh sáng của Hitachi cho phép ghi lại rõ nét sơ đồ vân chỉ trong vòng chưa đến 2 giây nhờ độ tương phản cao và khả năng tương thích với mọi loại da tay, kể cả da khô, da dầu hay có vết bẩn, vết nhăn hoặc bị khiếm khuyết do tạo hoá trên bề mặt của các ngón tay. Lượng dữ liệu nhỏ đó là căn cứ cho việc nhận dạng và tạo nên một hệ thống nhỏ gọn, an toàn, thân thiện và nhanh nhất trên thế giới. Hệ thống này có thể lưu trữ từ 6.000-8.000 ngón tay trong một máy và mỗi người có thể được nhận dạng bởi 1 trong 5 ngón tay khác nhau đã đăng ký trước đó. Ưu điểm vượt trội của hệ thống này là chỉ tương tác với cơ thể sống nên việc bắt chước, giả mạo hoặc ăn cắp dữ liệu là điều hoàn toàn bất khả thi. Công nghệ sinh trắc học nhận dạng bằng vân tay FVB (Finger Vein Biometrics) ra đời hồi đầu năm 2006, đã nhanh chóng thành công tại thị trường Nhật Bản, Singapor, Trung Quốc...

- Nhận diện dấu vân tay (*finger identification*): Dấu vân tay sẽ được đưa vào để đối chiếu với database chứa các vân tay để truy ra các đặc điểm muốn truy xuất. Việc đối sánh ảnh vân tay cần nhận dạng chỉ cần được tiến hành trên các vân tay (có trong cơ sở dữ liệu) thuộc loại đã được xác định nhờ quá trình phân loại. Đây là giai đoạn quyết định xem hai ảnh vân tay có hoàn toàn giống nhau hay không và đưa ra kết quả nhận dạng, tức là ảnh vân tay cần nhận dạng tương ứng với vân tay của cá thể nào đã được lưu trữ trong cơ sở dữ liệu.

#### \* Sơ đồ nhận dạng dấu vân tay



## Sơ đồ nhận dạng vân tay

Theo phân tích ở trên thì hệ thống này gồm 2 phần:

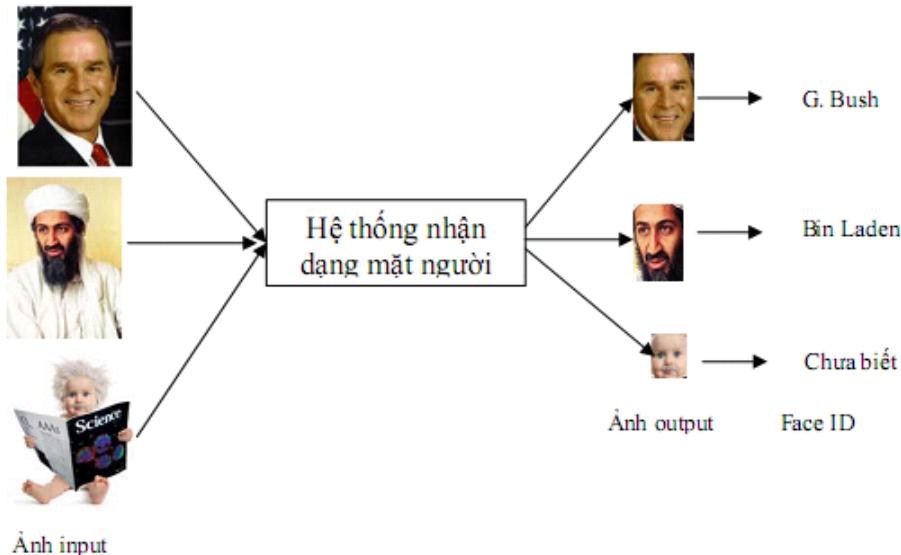
- + Verification: Ban đầu một người dùng trong hệ thống cung cấp thông tin dấu vân tay của mình và lưu trữ vào cơ sở dữ liệu.
- + Identification: Khi sử dụng hệ thống này, dấu vân tay được thu thập từ một sensor và đem đi xử lý. Quá trình xử lý có thể tùy chọn một trong 2 phương pháp trên.

### 2. Xác thực thông qua nhận dạng mặt người

Khuôn mặt là một trong những đặc trưng sinh trắc học thường dùng nhất trong quá trình nhận dạng giữa con người với nhau. Tuy nhiên, quá trình nhận dạng mặt của các hệ thống sinh trắc lại gặp phải nhiều vấn đề rất phức tạp như việc cải trang khuôn mặt, các hiệu ứng do tuổi tác gây ra, do việc thể hiện các cảm xúc trên khuôn mặt, do việc thu nhận ảnh liên quan đến góc độ thu nhận, ánh sáng...

#### a. Đặc điểm hệ thống nhận dạng khuôn mặt:

Hệ thống nhận dạng mặt người là một hệ thống nhận vào là một ảnh hoặc một đoạn video (một chuỗi các ảnh). Qua xử lý tính toán hệ thống xác định được vị trí mặt người trong ảnh (nếu có) và xác định là người nào trong số những người hệ thống đã được biết (qua quá trình học) hoặc là người lạ.



### Ví dụ về hệ thống nhận dạng mặt người

#### b. Phương pháp nhận dạng khuôn mặt:

Quá trình nhận dạng khuôn mặt trên cả ảnh 2D và 3D liên quan chủ yếu đến những vấn đề sau:

- Phát hiện mặt (Detection/Segmentation): nhằm tìm ra được chính xác vùng ảnh chứa khuôn mặt của người cần nhận dạng. Có nhiều phương pháp khác nhau đã được đề xuất cho việc phát hiện khuôn mặt trong một ảnh, cũng như trong một chuỗi ảnh kết tiếp nhau, chẳng hạn như sử dụng mạng neural [4], dựa theo kỹ thuật SVM (Support Vector Machines) [5], theo mô hình Gaussians trộn [6]...

- Trích chọn đặc trưng: để từ đó thu được những đặc trưng quan trọng nhất có thể biểu diễn được khuôn mặt của một người. Có nhiều cách tiếp cận khác nhau cho việc trích chọn đặc trưng, sử dụng dấu hiệu điểm (point signature) và bộ lọc Gabor để xác lập nên các đặc trưng mặt [4], khai thác thông tin màu sắc ở ảnh 2D để trích vùng màu biên và từ đó xác định được mắt và miệng [4], đánh giá tư thế và đỉnh mũi kết hợp với trích chọn dữ liệu liên quan đến mắt và miệng trong ảnh 3D [4].

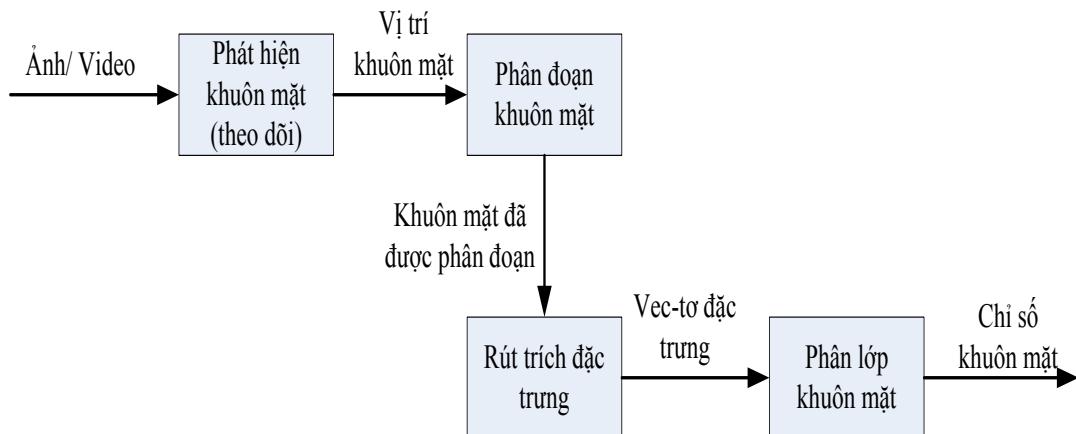
- Kiểm tra, so sánh: Sau khi đã trích chọn được những đặc trưng tiêu biểu của khuôn mặt, việc nhận dạng mặt sẽ dựa chủ yếu vào việc so sánh những đặc trưng đó với những đặc trưng của mẫu đã được lưu trong hệ thống.

Kết quả của quá trình nhận dạng mặt thường thấp hơn so với nhận dạng vân tay. Lý do chính của điều đó liên quan đến những vấn đề phức tạp đã được đề cập ở trên. Chính vì thế các hệ thống xác thực sinh trắc học không sử dụng duy nhất nhân tố khuôn mặt mà kết hợp với các nhân tố khác, chẳng hạn như vân tay, móng mỉa... nhằm nâng cao độ chính xác của quá trình xác thực.

#### c. Kiến trúc của một hệ thống nhận dạng mặt người

Một hệ thống nhận dạng mặt người thông thường bao gồm bốn bước xử lý sau: phát hiện khuôn mặt (face detection), phân đoạn khuôn mặt (face alignment hay

segmentation), rút trích đặc trưng (feature extraction), và phân lớp khuôn mặt (face classification).

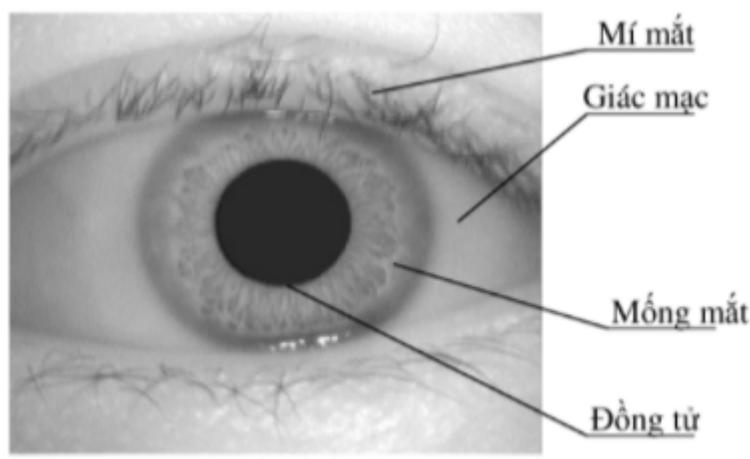


### Các bước chính trong một hệ thống nhận dạng mặt người

Phát hiện khuôn mặt dò tìm và định vị những vị trí khuôn mặt xuất hiện trong ảnh hoặc trên các frame video. Phân đoạn khuôn mặt sẽ xác định vị trí mắt mũi, miệng, và các thành phần khác của khuôn mặt và chuyển kết quả này cho bước rút trích đặc trưng. Từ những thông tin về các thành phần trên khuôn mặt, chúng ta có thể dễ dàng tính được véc-tơ đặc trưng trong bước rút trích đặc trưng. Những véc-tơ đặc trưng này sẽ là dữ liệu đầu vào cho một mô hình đã được huấn luyện trước để phân loại khuôn mặt. Bên cạnh những bước chính nêu trên, chúng ta còn có thể áp dụng thêm một số bước khác như tiền xử lý, hậu xử lý nhằm làm tăng độ chính xác cho hệ thống. Do một số thông số như: tư thế khuôn mặt, độ sáng, điều kiện ánh sáng..., nên phát hiện khuôn mặt được đánh giá là bước khó khăn và quan trọng nhất so với các bước còn lại của hệ thống

#### 3. Xác thực thông qua nhận dạng mống mắt

Mống mắt là màng tròn mỏng, nằm giữa giác mạc và thủy tinh thể của mắt người. Một ảnh nhìn chính diện của mống mắt được chỉ ra trong hình 1.22 dưới đây. Mống mắt bị đục thủng gần tâm của nó bởi lỗ tròn gọi là đồng tử.



#### Ảnh mắt của con người nhìn trực diện từ phía trước

Mặc dù màu sắc và cấu trúc của mống mắt và cấu trúc của mống mắt gắn với vấn đề di truyền học, nhưng những đặc trưng chính của mỗi mống mắt là không giống nhau.

Mắt phát triển trong suốt thời kỳ trước khi trưởng thành thông qua một quá trình định hình chặt chẽ và sự tạo nếp của các màng mô. Sự hình thành móng mắt bắt đầu vào tháng thứ 3 của thai kỳ và việc tạo ra cấu trúc kiểu của nó khá dày dì vào tháng thứ 8. Nhưng kiểu dáng duy nhất trên bề mặt móng mắt được tạo thành trong suốt một năm đầu tiên, và sự phát triển của các sắc tố chất nền xảy ra khoảng vài năm đầu sau khi sinh. Sự hình thành các kiểu dáng đơn nhất của móng mắt là ngẫu nhiên không liên quan tới bất kỳ nhân tố gien nào. Chỉ những đặc tính mà phụ thuộc vào gien là sắc tố của móng mắt mới xác định màu sắc của nó. Nhờ biểu sinh tự nhiên của các kiểu móng mắt, hai mắt của một cá nhân hoàn toàn độc lập về kiểu móng mắt và ngay cả các cặp sinh đôi giống hệt nhau cũng có các kiểu móng mắt khác nhau. Chính vì đặc điểm mỗi móng mắt là duy nhất và các cấu trúc khác biệt nêu trên nên ảnh móng mắt có thể được sử dụng cho mục đích nhận dạng/xác thực người dùng. Với phương pháp này, móng mắt của người dùng sẽ được sao chụp lại bằng một monochrome-camera đặc biệt [4]. Ảnh móng mắt của người đó sẽ được phân tích, trích chọn những đặc trưng tốt nhất cho quá trình so sánh kiểm tra..

Phương pháp được sử dụng gần đây nhất cho quá trình trích chọn đặc trưng được thực hiện như sau: từ ảnh móng mắt, bộ lọc 2D Gabor sẽ được sử dụng để phân đoạn thành nhiều vector được gọi là các phasors [4]. Các phasors chứa cả tần số, hướng và vị trí của từng vùng của móng mắt gọi là các mã móng mắt – IrisCodes [4]. Từ đó, quá trình nhận dạng quy về việc so sánh hai IrisCodes thông qua nguyên lý khoảng cách Hamming [4] để kiểm tra tính độc lập thống kê giữa chúng. Phép kiểm tra đó thất bại sẽ cho phép khảng định hai IrisCodes đó là giống nhau và từ đó cho phép xác thực người dùng.

Quá trình xác thực dựa trên nhận dạng móng mắt cho kết quả với độ chính xác cao hơn so với các phương pháp khác, do khả năng không thể làm giả, mô phỏng hay sửa đổi những đặc trưng móng mắt của một người. Đây là một trong những nhân tố được sử dụng trong việc xây dựng mô hình hộ chiếu điện tử mà nhiều quốc gia đang quan tâm, nghiên cứu, triển khai.

#### 4. Xác thực thông qua nhận dạng giọng nói

Giọng nói cũng là một trong những đặc điểm sinh trắc học thường được sử dụng trong các hệ thống xác thực người dùng. Đối với các ứng dụng trên điện thoại, nhận dạng giọng nói là phương thức đơn giản và tiện lợi nhất cho phép định danh người sử dụng. Tuy nhiên, chỉ với một mình giọng nói thì độ chính xác của quá trình xác thực thường thấp bởi nhiều lý do như chất lượng âm thanh, do khả năng bắt chước giọng nói của một số người, rồi các hiệu ứng do cảm xúc, do sức khỏe...

Cũng giống như việc nhận dạng các đặc điểm khác, nhận dạng giọng nói cũng bao gồm hai quá trình: trích chọn đặc trưng và sau đó so sánh với mẫu đã được lưu trữ trong hệ thống. Có hai cách tiếp cận chính đối với nhận dạng giọng nói: nhận dạng phụ thuộc text và nhận dạng độc lập text. Với cách tiếp cận đầu, việc định danh một người sẽ thông qua việc yêu cầu người đó phát âm một câu đã được định trước. Khi đó, quá trình trích chọn đặc trưng và so sánh sẽ đơn giản hơn. Ngược lại, đối với nhận dạng độc lập text, hệ

thông chỉ lưu những đặc trưng của người dùng và không cần người dùng phải phát âm một đoạn text bắt buộc trước.

Có nhiều phương pháp khác nhau cho việc trích chọn đặc trưng, nhưng ưu trung các phương pháp đề xuất đều làm việc ở mức ngữ âm (acoustic level) thông qua việc phân tích phổ tín hiệu tiếng nói thành chuỗi vector đặc trưng. Các phương pháp trích chọn đặc trưng được sử dụng gần đây nhất là dựa trên mô hình GMM (Gaussian Mixture Models) kết hợp với mô hình âm tiết theo HMM (Hidden Markov Model) [4], tăng cường với chuẩn hóa phổ [4], GMM kết hợp với biến đổi sóng [4]...

a. Các phương pháp nhận dạng được áp dụng phổ biến:

- So sánh mẫu bằng phương pháp lập trình động (Dynamic Program)

Khi so sánh mẫu tín hiệu người ta thường phải so sánh với tất cả các mẫu, điều này sẽ làm tốn rất nhiều thời gian tính toán. Để giảm thời gian tính toán và qua đó tăng tốc độ xử lý nhận dạng, người ta có thể sử dụng phương pháp lập trình động. Ở phương pháp nhận dạng mẫu này, các từ cần nhận dạng sẽ được so sánh với các mẫu được lưu trữ trong hệ thống và thực hiện việc so sánh hai mẫu tín hiệu này để tìm ra mẫu có sai số là nhỏ nhất. Bởi vì tín hiệu âm thanh được tạo ra tại các thời điểm khác nhau không bao giờ là giống nhau hoàn toàn. Nó luôn có sự sai khác do một số yếu tố về trọng âm, ngữ điệu, tốc độ.... Vì vậy, cần phải thực hiện so sánh hai mẫu theo các thuật toán biến dạng nhằm giảm thiểu sai số. Thuật toán DTW (Dynamic Time Warping) có thể coi là thuật toán hiệu quả nhất cho việc ứng dụng so sánh hai mẫu tín hiệu có chiều dài khác nhau và cho sai số nhỏ nhất. Thuật toán này sử dụng phương pháp đệ quy, ví dụ: các chương trình con (Procedure) được tự động gọi ra nhưng với các thông số (parameter) khác nhau và tìm các sai số so với các tín hiệu mẫu. Mẫu nào có sai số so với tín hiệu cần so sánh nhỏ nhất thì mẫu đó chính là mẫu cần tìm.

- Nhận dạng từ sử dụng mạng Nơ-ron

Công nghệ nhận dạng tiếng nói chủ yếu sử dụng phương pháp nhận dạng mẫu và mạng Nơ-ron là một trong những công cụ nhận dạng mẫu có hiệu quả, do vậy nhiều hệ thống đã ứng dụng mạng Nơ-ron vào việc nhận dạng tiếng nói. Mạng Nơ-ron cấu trúc Perceptron nhiều lớp được sử dụng nhiều trong các hệ thống nhận dạng. Perceptron là loại đơn giản nhất của các mạng liên kết tiến (là mạng không có liên kết giữa các khối xử lý trong cùng một lớp và không có các liên kết giữa các khối xử lý ở lớp ra quay ngược về lớp vào) sử dụng thuật toán học có giám sát. Một mạng Perceptron bao gồm nhiều đơn vị xử lý được sắp xếp thành các lớp. Mạng này được huấn luyện theo quy tắc Delta hoặc các biến thể của nó. Các khối xử lý được sắp xếp thành các lớp bao gồm 1 lớp vào một khối xử lý ở một lớp ẩn và 1 lớp ra. Các liên kết có trọng số khác nhau kết nối mỗi một khối xử lý ở một lớp nào đó tới tất cả các khối xử lý ở lớp lân cận. Mạng Nơ-ron loại này được huấn luyện bằng cách nhập một vec-tơ mẫu ở lớp đầu vào và tính toán các đầu ra. Sau đó, đầu ra được so sánh với các mẫu đầu ra mong muốn. Sai số giữa đầu ra thực tế với đầu ra mong muốn được tính và phản hồi qua mạng tới mỗi phần tử. Trọng số đầu vào của mỗi phần tử được điều chỉnh để tối thiểu hóa sai số. Quá trình này được lặp lại

đến khi đầu ra thực tế lệch với đầu ra mong muốn trong phạm vi sai số xác định trước. Có rất nhiều cặp mẫu đầu vào, đầu ra được đưa qua mạng và quá trình nêu trên được lặp lại cho mỗi cặp đầu vào, đầu ra. Việc nhận dạng chính là nhập mẫu tiếng nói chưa biết ở nút đầu vào của mạng đã được huấn luyện và tính toán giá trị của các nút đầu ra để xác định mẫu tiếng nói đó

### b. Những khó khăn trong quá trình nhận dạng tiếng nói

Khó khăn lớn nhất trong quá trình nhận dạng tiếng nói là cùng một từ nhưng không bao giờ có thể được phát âm hoàn toàn giống nhau ngay cả với cùng một người nói. Ngoài ra các biến âm thanh cũng còn bị phụ thuộc vào trạng thái vật lý và tâm lý người nói cũng như do ảnh hưởng của ngữ cảnh, chất lượng của Microphone và môi trường cũng là các tác nhân ảnh hưởng đến giọng nói... Nhiều của môi trường xung quanh cũng làm cho tần số của từ được phát âm thay đổi rất nhiều và làm cho hệ thống rất khó nhận dạng và thậm chí còn không thể làm việc được. Ngoài các ảnh hưởng liên quan đến âm thanh thì vẻ mặt, điệu bộ của người nói chuyện cũng được truyền tải rất nhiều thông tin mà hệ thống nhận dạng không có khả năng chuyển đổi và đây cũng là hạn chế rất lớn của các hệ thống.

Cũng như phương pháp xác thực dựa trên nhận dạng mặt, nhận dạng người nói cho kết quả không cao [6] trong việc định danh một người. Vì thế nhận dạng người nói thường được kết hợp với nhận dạng các đặc điểm sinh trắc học khác nhằm nâng cao độ chính xác của hệ thống xác thực sinh trắc.

#### 2.2.3.2. Sinh trắc học hành vi

Sinh trắc học hành vi tương tự như sinh trắc học thông thường, nhưng thay vì nhận dạng dấu vân tay, hệ thống sẽ theo dõi chuyển động chuột và hành vi điển hình trên tài khoản, tốc độ vuốt trên màn hình ứng dụng và những thói quen sử dụng thiết bị của bạn để xác thực danh tính... Tất cả đều được xem là sinh trắc học hành vi (*Ví dụ, khi đăng nhập vào trang web, hệ thống hiển thị cảnh báo “Bạn đang đăng nhập từ thiết bị không thường sử dụng”, tự động nhận ra bạn dùng thiết bị đăng nhập mới. Ngoài ra, còn có cảnh báo bảo mật dựa trên vị trí người dùng, nếu bạn đăng nhập tài khoản ở một khu vực khác nơi sinh sống, hệ thống sẽ gửi cảnh báo và thực hiện các biện pháp xác thực khác để bảo mật dữ liệu và tài khoản của bạn tốt hơn*)

Thực tế những dịch vụ và các trang web đã theo dõi hành vi người dùng trong nhiều năm để phát triển biện pháp bảo mật tiện lợi và an toàn hơn so với phương thức thông thường. Thậm chí ngay cả khi người dùng không sử dụng thiết bị vẫn được ghi nhận là một hành vi (*Ví dụ, nếu tài khoản ngân hàng của bạn bị hack, các giao dịch gian lận được thực hiện trong lúc bạn đang ngủ, khi đó tất cả thiết bị bạn thường sử dụng đều nằm yên và không hoạt động. Hệ thống sẽ nhận dạng và xem xét hành vi bất thường và có thể tự động đưa ra cảnh báo về hoạt động đáng ngờ*).

Một phương pháp nhận dạng khá đơn giản là sử dụng các thông tin thu được từ các thao tác gõ bàn phím của người dùng (Keystroke Dynamics – KD).

Về bản chất KD là một dạng đặc trưng sinh trắc học cho phép mô tả thao tác người dùng khi gõ bàn phím máy tính, nhấn phím trên điện thoại di động (kể cả bàn phím cảm ứng ảo trên các dòng điện thoại thông minh) [10]. Ở đây, cần lưu ý rằng, với đa số các trang web hiện nay đều có khả năng phân biệt người dùng trên điện thoại di động hay máy tính cá nhân để đưa ra giao diện tương tác phù hợp, do vậy, việc khai thác đặc trưng sinh trắc học cũng có thể tiếp cận lợi thế này để biết trước thông tin thu được là từ bàn phím máy tính hay thiết bị di động.

Việc sử dụng KD trong đảm bảo an toàn thông tin có ưu điểm nổi bật là không cần sử dụng thêm các thiết bị phần cứng phụ trợ ngoại trừ bàn phím (Keyboard, Keypad). Việc sử dụng KD sẽ làm mạnh hơn sự xác thực thông tin người dùng, ngay cả trong trường hợp các thông tin đăng nhập (tên đăng nhập, mật khẩu) bị lộ lọt.

### 2.3. Quản trị mật khẩu

Thông thường mật khẩu là một chuỗi ký tự có độ dài xác định; ký tự mật khẩu phải được chọn từ một bộ (bảng) ký tự qui định trước. Không gian mật khẩu là tập tất cả các mật khẩu có thể xây dựng đợt từ qui ước mật khẩu. Ví dụ, có một hệ thống yêu cầu mật khẩu phải là một chuỗi 8 chữ số (từ là ký tự „0“-„9“); như vậy không gian mật khẩu là tập tất cả các chuỗi 8 chữ số (“00000000” đến “99999999”), và như vậy không gian này có 108 mật khẩu.

Để đảm bảo an toàn, người ta không lưu trữ mật khẩu ở dạng bản rõ tại máy chủ. Tại sao vậy? Vì sự có mặt một tệp mật khẩu lưu tại máy chủ sẽ rất nguy hiểm: chỉ cần một sơ suất nhỏ là tệp này có thể bị truy nhập bởi những người không được phép (hoàn cảnh ví dụ: admin/superuser quên logout khi đi ra ngoài chốc lát để cho có kẻ lén vào thao tác nhanh ăn cắp thông tin quan trọng), và toàn bộ mật khẩu của mọi NSD sẽ bị lộ. Thậm chí nếu như tệp mật khẩu này được bảo vệ (tức là mật mã bằng khóa mật) thì cũng không đảm bảo an toàn cao vì khóa mật mã vẫn phải lưu ở đâu đó thuật tiện sử dụng liên tục, tức là cũng có thể bị lộ với kẻ tấn công cao tay (vẫn trong hoàn cảnh ví dụ khi kẻ địch lén vào nói trên).

Vì vậy, các hệ điều hành luôn xây dựng A (tập mật khẩu) và C (tập thông tin đổi chiều lưu trữ phía hệ thống) là khác nhau. Đương nhiên, các hàm  $f: A \rightarrow C$  được sử dụng để biến đổi một giá trị  $a \in A$  về  $c = f(a) \in C$  để đổi chiều. Giải pháp thường dùng là sử dụng các hàm băm vì ngay cả khi giá trị  $c = f(a) \in C$  có bị lộ vì lý do nào đó, thì kẻ tấn công cũng không lấy được mật khẩu  $a$ . Hơn nữa kích thước các tập A và C cũng có thể khác nhau. Một phần thông tin của một giá trị  $c \in C$  có thể được dùng để xác định hàm băm  $f: A \rightarrow C$  đợt lọc dùng cho cặp  $(a, c)$  này. Chẳng hạn như trong một phiên bản truyền thống của cơ chế mật khẩu trong hệ điều hành Unix, có một tập 4096 hàm băm được sử dụng; mỗi giá trị  $c \in C$  là một chuỗi 13 ký tự, trong đó 11 ký tự là chuỗi băm từ  $a \in A$ , còn 2 ký tự được dùng để xác định 1 trong số 4096 hàm băm được dùng.

Ví dụ : Mô tả chi tiết hơn hệ thống mật khẩu Unix. Mỗi mật khẩu của Unix có thể có tối đa 8 ký tự ASCII, loại trừ mã 0, tức là còn 127 giá trị tất cả. Như vậy A có xấp xỉ  $6.9 \times 1016$  mật khẩu. Tuy nhiên, tập C bao gồm các chuỗi có đúng 13 ký tự, nhưng lấy từ

bảng chữ có kích thước 64. Nhờ vậy C có khoảng  $3.0 \times 1023$  thành viên. Nhiều hệ thống phiên bản UNIX lưu trữ tập C này trong tệp /etc/passwd mà tất cả các user đều đọc được. Tuy nhiên, một số phiên bản khác lại lưu trong các file dấu mà chỉ truy nhập được bởi superuser. Các hàm băm ~~F~~F được xây dựng như là các phiên bản của thuật toán mã hóa DES với sự thay đổi tùy chọn của một biến đổi hoán vị. Các thủ tục xác thực của UNIX bao gồm login, su và một số chương trình khác cho phép xác thực mật khẩu NSD trong quá trình thực hiện. Hệ thống sử dụng một số thành phần cầu tạo trong C mà NSD có thể không biết tới. Các thủ tục chọn mật khẩu là passwd hay nispasswd cho phép thay đổi các thông tin mật khẩu gắn với NSD

## 2.4. Chính sách điều khiển truy cập

### 2.4.1. Thẩm quyền/cấp phép

Ta sẽ cần phải đảm bảo rằng chính sách và các tài liệu hỗ trợ được ủy quyền phù hợp. Trong một số tổ chức, chữ ký của giám đốc có thể đủ. Trong các công ty khác, ta có thể tìm kiếm thẩm quyền từ một loạt các giám đốc điều hành cao cấp thuộc các lĩnh vực sau đây:

- Lĩnh vực tài chính
- Những người đứng đầu vận hành
- Lĩnh vực nhân sự
- Lĩnh vực pháp luật

### 2.4.2 Thực hiện

Sẽ rất khó để thực hiện toàn bộ chính sách thiết lập trên một tổ chức tại một thời điểm nhất định. Cách tiếp cận từng giai đoạn thường có hiệu quả nhất, lựa chọn một khu vực có ranh giới dễ dàng xác định và một trong đó hỗ trợ một loạt các dịch vụ khác, ví dụ như trung tâm hoạt động CNTT.

#### Lý do cần các trung tâm hoạt động CNTT

- Điều này là bình thường một nhà cung cấp dịch vụ nội bộ quan trọng, trong đó đảm bảo cao là cần thiết.
  - Nhiều biện pháp an toàn được phân phối thông qua công nghệ.
  - Cung cấp CNTT thường đã được quản lý tốt.
  - Phạm vi không phải bao gồm tất cả người dùng.

Giai đoạn tiếp theo có thể tập trung vào các lĩnh vực kinh doanh cốt lõi và người dùng cuối CNTT, như các hoạt động CNTT sau đó sẽ phù hợp với chính sách.

Một mục tiêu thứ hai có thể là nhân viên hoặc chức năng nguồn nhân lực. Lý do lựa chọn này nhìn chung tương tự như đối với lựa chọn các chức năng CNTT ban đầu.

Các khu vực mục tiêu tiếp theo sẽ được hưởng lợi từ công việc này, và nó sẽ được dễ dàng hơn để thực hiện chính sách như "nhà cung cấp dịch vụ" trung tâm sẽ được bao phủ bởi chính sách đã có.

### **2.4.3 Hoạt động**

Hoạt động là một phần của vấn đề rộng hơn về quản lý an ninh thông tin đến sự cần thiết phải giám sát việc tuân theo (và hiệu quả) các chính sách. Ta sẽ thấy rằng một số phát biểu chính sách không được tôn trọng (ví dụ như mọi người có thể chia sẻ mật khẩu). Điều này có thể bởi vì chúng sẽ làm ảnh hưởng đến hoạt động bình thường, hoặc được coi như là quá mức tốn kém.

Trong hầu hết các trường hợp chìa khóa để bảo mật hiệu quả là thiết lập chính sách có thể đạt được và để đảm bảo tuân thủ. Ta cũng sẽ cần phải có các quy trình phù hợp để đối phó với các trường hợp không thể tránh khỏi, nhưng rất ít, trường hợp các ngoại lệ chính sách hợp lệ. Tuy nhiên, chúng ta cần chuẩn bị để thay đổi các phát biểu của mình dựa trên các thông tin phản hồi. Mọi người sẽ thực hiện dễ dàng hơn nếu họ đã được tham gia xây dựng chính sách và cảm thấy họ là các bên liên quan.

Trong thời gian này, các thiết lập chính sách nên trở thành chuẩn mực cho bất kỳ quy trình kiểm toán được sử dụng.

## **2.5. Phương pháp điều khiển truy cập**

### **2.5.1 Mô hình điều khiển truy xuất bắt buộc (Mandatory Access Control\_MAC):**

Mô hình điều khiển truy xuất bắt buộc là mô hình điều khiển truy xuất được áp dụng bắt buộc đối với toàn hệ thống. Trong môi trường máy tính, cơ chế điều khiển truy xuất bắt buộc được tích hợp sẵn trong hệ điều hành, và có tác dụng đối với tất cả các tài nguyên và đối tượng trong hệ thống, người sử dụng không thể thay đổi được.

Ví dụ: trong hệ thống an toàn nhiều cấp (multilevel security), mỗi đối tượng (subject) hoặc tài nguyên (object) được gán một mức bảo mật xác định. Trong hệ thống này, các đối tượng có mức bảo mật thấp không được đọc thông tin từ các tài nguyên có mức bảo mật cao, ngược lại các đối tượng ở mức bảo mật cao thì không được ghi thông tin vào các tài nguyên có mức bảo mật thấp. Mô hình này đặc biệt hữu dụng trong các hệ thống bảo vệ bí mật quân sự (mô hình BellLaPadula, 1973).

Những đặc điểm phân biệt của mô hình điều khiển truy xuất bắt buộc:

- Được thiết lập cố định ở mức hệ thống, người sử dụng (bao gồm cả người tạo ra tài nguyên) không thay đổi được.

- Người dùng và tài nguyên trong hệ thống được chia thành nhiều mức bảo mật khác nhau, phản ánh mức độ quan trọng của tài nguyên và người dùng.

- Khi mô hình điều khiển bắt buộc đã được thiết lập, nó có tác dụng đối với tất cả người dùng và tài nguyên trên hệ thống.

### ***2.5.2 Mô hình điều khiển truy xuất tự do (Discretionary Access Control\_DAC)***

Mô hình điều khiển truy xuất tự do là mô hình điều khiển truy xuất trong đó việc xác lập quyền truy xuất đối với từng tài nguyên cụ thể do người chủ sở hữu của tài nguyên đó quyết định. Đây là mô hình được sử dụng phổ biến nhất, xuất hiện trong hầu hết các hệ điều hành máy tính.

Ví dụ: trong hệ thống quản lý tập tin NTFS trên Windows XP, chủ sở hữu của một thư mục có toàn quyền truy xuất đối với thư mục, có quyền cho phép hoặc không cho phép người dùng khác truy xuất đến thư mục, có thể cho phép người dùng khác thay đổi các xác lập về quyền truy xuất đối với thư mục.

Đặc điểm phân biệt của mô hình điều khiển truy xuất tự do:

- Không được áp dụng mặc định trên hệ thống
- Người chủ sở hữu của tài nguyên (owner), thường là người tạo ra tài nguyên đó hoặc người được gán quyền sở hữu, có toàn quyền điều khiển việc truy xuất đến tài nguyên.
  - Quyền điều khiển truy xuất trên một tài nguyên có thể được chuyển từ đối tượng (user) này sang đối tượng (user) khác.

### ***2.5.3 Mô hình điều khiển truy xuất theo chức năng (Role Based Access Control\_RBAC)***

Mô hình điều khiển truy xuất theo chức năng là mô hình điều khiển truy xuất dựa trên vai trò của từng người dùng trong hệ thống (user' roles).

Ví dụ: một người quản lý tài chính cho công ty (financial manager) thì có quyền truy xuất đến tất cả các dữ liệu liên quan đến tài chính của công ty, được thực hiện các thao tác sửa, xóa, cập nhật trên cơ sở dữ liệu. Trong khi đó, một nhân viên kế toán bình thường thì chỉ được truy xuất đến một bộ phận nào đó của cơ sở dữ liệu tài chính và chỉ được thực hiện các thao tác có giới hạn đối với cơ sở dữ liệu.

Vấn đề quan trọng trong mô hình điều khiển truy xuất theo chức năng là định nghĩa các quyền truy xuất cho từng nhóm đối tượng tùy theo chức năng của các đối tượng đó. Việc này được định nghĩa ở mức hệ thống và áp dụng chung cho tất cả các đối tượng.

Cơ chế quản lý theo nhóm (account group) của Windows NT chính là sự mô phỏng của mô hình RBAC. Trong cơ chế này, người sử dụng được gán làm thành viên của một hoặc nhiều nhóm trong hệ thống, việc phân quyền truy xuất đến các tài nguyên được thực

hiện đối với các nhóm chứ không phải đối với từng người dùng, khi đó các người dùng thành viên trong nhóm sẽ nhận được quyền truy xuất tương đương một cách mặc định. Việc thay đổi quyền truy xuất đối với từng người dùng riêng biệt được thực hiện bằng cách chuyển người dùng đó sang nhóm khác có quyền truy xuất thích hợp.

Đặc điểm phân biệt của mô hình điều khiển truy xuất theo chức năng:

- Quyền truy xuất được cấp dựa trên công việc của người dùng trong hệ thống (user's role)

- Linh động hơn mô hình điều khiển truy xuất bắt buộc, người quản trị hệ thống có thể cấu hình lại quyền truy xuất cho từng nhóm chức năng hoặc thay đổi thành viên trong các nhóm.

- Thực hiện đơn giản hơn mô hình điều khiển truy xuất tự do, không cần phải gán quyền truy xuất trực tiếp cho từng người dùng.

## 2.6 Các kiểu tấn công

Phân loại:

- 1) Tấn công thăm dò.
- 2) Tấn công sử dụng mã độc.
- 3) Tấn công xâm nhập mạng.
- 4) Tấn công từ chối dịch vụ.

\* Hoặc:

- 1) Tấn công chủ động.
- 2) Tấn công thụ động.

# CHƯƠNG 3: CHÍNH SÁCH AN NINH

## 3.1. Các khái niệm

Chính sách bảo mật là hệ thống các quy định nhằm đảm bảo sự an toàn của hệ thống. Việc bảo mật hệ thống cần có một chính sách bảo mật rõ ràng. Cần có những chính sách bảo mật riêng cho những yêu cầu bảo mật khác nhau.

Xây dựng và lựa chọn các chính sách bảo mật cho hệ thống phải dựa theo các chính sách bảo mật do các tổ chức uy tín về bảo mật định ra (compliance) như: NIST, SP800, ISO17799, HIPAA.

Chính sách bảo mật phải cân bằng giữa 3 yếu tố khả dụng, bảo mật, hiệu suất. Chính sách bảo mật có thể được biểu diễn bằng ngôn ngữ tự nhiên hoặc ngôn ngữ toán học.

Ví dụ: trong một hệ thống, để bảo đảm an toàn cho một tài nguyên (resource) cụ thể, chính sách an toàn quy định rằng chỉ có người dùng nào thuộc nhóm quản trị hệ thống (Administrators) mới có quyền truy xuất, còn những người dùng khác thì không. Đây là cách biểu diễn bằng ngôn ngữ tự nhiên.

Có thể biểu diễn quy định này bằng ngôn ngữ toán học như sau:

Gọi:  $U$  là tập hợp các người dùng trong hệ thống.

$A$  là tập hợp các người dùng thuộc nhóm quản trị.

$O$  là tập hợp các đối tượng (tài nguyên) trong hệ thống.

Thao tác  $Access(u, o)$  cho giá trị TRUE nếu người dùng  $u$  có quyền truy xuất đến đối tượng  $o$ , ngược lại, cho giá trị FALSE.

Quy định  $p$  trong chính sách an toàn được phát biểu như sau:

$$\forall u \in U, \forall o \in O: Access(u, o) = \text{TRUE} \Leftrightarrow u \in A$$

Ma trận cũng thường được dùng để biểu diễn một chính sách bảo mật.

Ví dụ: một hệ thống với các tập người dùng  $U = \{u_1, u_2, u_3, u_4\}$  và tập đối tượng  $O = \{o_1, o_2, o_3, o_4\}$ . Các thao tác mà một người dùng  $u$  có thể thực hiện được trên một đối tượng  $o$  bao gồm đọc (r), ghi (w) và thực thi (x). Quy định về khả năng truy xuất của từng người dùng đến từng đối tượng trong hệ thống được biểu diễn bằng ma trận như Bảng 1.1.

Quan sát ma trận, ta biết rằng người dùng  $u_3$  được quyền đọc trên tất cả các đối tượng từ  $o_1$  đến  $o_4$ , trong khi đó người dùng  $u_4$  thì không có quyền truy xuất đến bất kỳ đối tượng nào.

**Bảng 0.1 Ma trận về khả năng truy xuất của từng người dùng**

	$u_1$	$u_2$	$u_3$	$u_4$
$o_1$	x	x	r	
$o_2$	x	r	r	
$o_3$	w		r	
$o_4$	w		r	

Trong thực tế, việc đề ra chính sách bảo mật cho một hệ thống thông tin phải đảm bảo được sự cân bằng giữa lợi ích của việc bảo đảm an toàn hệ thống và chi phí thiết kế, cài đặt và vận hành các cơ chế bảo vệ chính sách đó.

### 3.2. Các kiểu chính sách an ninh (Tìm hiểu thêm)

### 3.3. Chính sách bảo mật (Tìm hiểu thêm)

### 3.4. Chính sách toàn vẹn (Tìm hiểu thêm)

## CHƯƠNG 4: MÃ HÓA CĂN BẢN

### 4.1. Tổng quan

Mật mã (Encryption) là một kỹ thuật cơ sở quan trọng trong bảo mật thông tin. Nguyên tắc của mật mã là biến đổi thông tin gốc thành dạng thông tin bí mật mà chỉ có những thực thể tham gia xử lý thông tin một cách hợp lệ mới hiểu được.

Một thực thể hợp lệ có thể là một người, một máy tính hay một phần mềm nào đó được phép nhận thông tin. Để có thể giải mã được thông tin mật, thực thể đó cần phải biết cách giải mã (tức là biết được thuật toán giải mã) và các thông tin cộng thêm (khóa).

Quá trình chuyển thông tin gốc thành thông tin mật theo một thuật toán nào đó được gọi là quá trình mã hóa (encryption). Quá trình biến đổi thông tin mật về dạng thông tin gốc ban đầu gọi là quá trình giải mã (decryption). Đây là hai quá trình không thể tách rời của một kỹ thuật mật mã bởi vì mật mã (giấu thông tin) chỉ có ý nghĩa khi ta có thể giải mã (phục hồi lại) được thông tin đó. Do vậy, khi chỉ dùng thuật ngữ mật mã thì nó có nghĩa bao hàm cả mã hóa và giải mã.

Kỹ thuật mã hóa được chia thành hai loại: mã hóa dùng khoá đối xứng (symmetric key encryption) và mã hóa dùng khoá bất đối xứng (asymmetric key encryption) như sẽ trình bày trong các phần tiếp theo

Hình 2.1 mô tả nguyên tắc chung của một hệ thống mật mã quy ước. Các thành phần trong một hệ thống mật mã điển hình bao gồm:

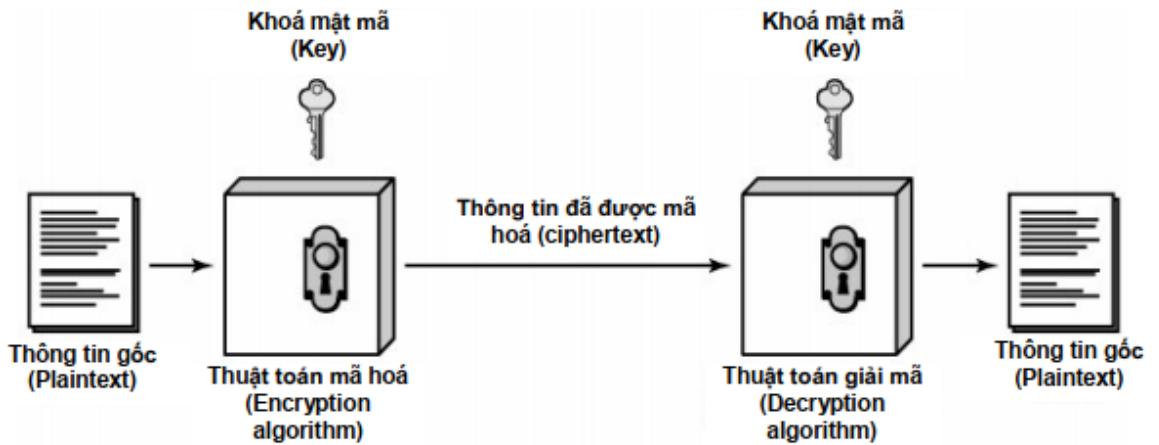
-Plaintext: là thông tin gốc cần truyền đi giữa các hệ thống thông tin

-Encryption algorithm: thuật toán mã hóa, đây là cách thức tạo ra thông tin mật từ thông tin gốc.

-Key: khóa mật mã, gọi tắt là khóa. Đây là thông tin cộng thêm mà thuật toán mã hóa sử dụng để trộn với thông tin gốc tạo thành thông tin mật.

-Ciphertext: thông tin đã mã hóa (thông tin mật). Đây là kết quả của thuật toán mã hóa

-Decryption algorithm: Thuật toán giải mã. Đầu vào của thuật toán này là thông tin đã mã hóa (ciphertext) cùng với khóa mật mã. Đầu ra của thuật toán là thông tin gốc (plaintext) ban đầu.



### Cấu trúc một hệ thống mật mã quy ước

Một hệ thống mã hóa bất kỳ được đặc trưng bởi 3 tiêu chí sau đây:

-Phương pháp mã (operation): có hai phương pháp mật mã bao gồm thay thế (substitution) và chuyển vị (transposition). Trong phương pháp mã thay thế, các đơn vị thông tin (bit, ký tự, byte hoặc khối) trong thông tin gốc được thay thế bằng các đơn vị thông tin khác theo một quan hệ nào đó. Trong phương pháp mã chuyển vị, các đơn vị thông tin trong thông tin gốc được đổi chỗ cho nhau để tạo thành thông tin mã hóa. Các hệ thống mã hóa hiện đại thường kết hợp cả hai phương pháp thay thế và chuyển vị.

-Số khóa sử dụng (number of keys): nếu phía mã hóa (phía gửi) và phía giải mã (phía nhận) sử dụng chung một khóa, ta có hệ thống mã dùng khoá đối xứng (symmetric key) - gọi tắt là mã đối xứng hay còn có các tên gọi khác như mã một khóa (single-key), mã khóa bí mật (secret key) hoặc mã quy ước (conventional cryptosystem). Nếu phía mã hóa và phía giải mã dùng 2 khóa khác nhau, hệ thống này được gọi là mã bất đối xứng (asymmetric key), mã hai khóa (two key) hoặc mã khóa công khai (public key).

- Cách xử lý thông tin gốc (mode of cipher): thông tin gốc có thể được xử lý liên tục theo từng phần tử , khi đó ta có hệ thống mã dòng (stream cipher). Ngược lại, nếu thông tin gốc được xử lý theo từng khối, ta có hệ thống mã khối (block cipher). Các hệ thống mã dòng thường phức tạp và không được phổ biến công khai, do đó chỉ được dùng trong một số ứng dụng nhất định (ví dụ trong thông tin di động GSM).

## 4.2. Mã hóa khóa bí mật

### 4.2.1 Các hệ mật thay thế đơn biểu

#### 4.2.1.1 Mã dịch vòng (MDV)

Giả sử  $P=C=K = Z_{26}$  với,  $0 \leq k \leq 25$ , ta định nghĩa:

$$e_k(x) = x + k \bmod 26$$

$$d_k(y) = y - k \bmod 26$$

$$(x, y \in Z_{26})$$

Mã dịch vòng

Ta sử dụng MDV (với modulo 26) để mã hoá một văn bản tiếng Anh thông thường bằng cách thiết lập sự tương ứng giữa các ký tự và các thặng dư theo mod 26 như sau:

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

### Ví dụ1:

Giả sử khoá cho MDV là  $k = 5$  và bản rõ là meetmeatsunset.

Trước tiên, ta biến đổi bản rõ thành dãy các số nguyên theo bảng trên:

12.4.4.19.12.4.0.19.18.20.13.18.4.19

Sau đó ta cộng 5 vào mỗi giá trị ở trên và rút gọn tổng theo mod 26, ta được dãy số sau:

17.9.9.24.17.9.5.24.23.25.18.23.9.24

Cuối cùng, ta lại biến đổi dãy số nguyên trên thành các ký tự tương ứng, ta có bản mã sau:

RJJYRJFYXZSXJY

Để giải mã cho bản mã này, trước tiên ta biến bản mã thành dãy số nguyên rồi trừ mỗi giá trị cho 5 (rút gọn theo modulo 26), và cuối cùng là lại biến đổi lại dãy số nhận được này thành các ký tự.

### Nhận xét:

Khi  $k = 3$ , hệ mật này thường được gọi là mã Caesar đã từng được Hoàng đế Caesar sử dụng.

MDV (theo mod 26) là không an toàn vì nó có thể bị thám theo phương pháp tìm khoá vét cạn (thám mã có thể dễ dàng thử mọi khoá  $d_k$  có thể cho tới khi tìm được bản rõ có nghĩa). Trung bình có thể tìm được bản rõ đúng sau khi thử khoảng  $(26/2)=13$  quyết giải mã.

Từ ví dụ trên ta thấy rằng, điều kiện cần để một hệ mật an toàn là phép tìm khoá vét cạn phải không thể thực hiện được. Tuy nhiên, một khóa gian lớn vẫn chưa đủ để đảm bảo độ mật.

#### 4.2.1.2 Mã Affine

Trong mã Affine, ta giới hạn chỉ xét các hàm mã có dạng:

$$e(x) = ax + b \pmod{26}$$

$a, b \in Z_{26}$ . Các hàm này được gọi là các hàm Affine (chú ý rằng khi  $a = 1$ , ta có MDV).

Để việc giải mã có thể thực hiện được, yêu cầu cần thiết là hàm Affine phải là đơn ánh. Nói cách khác, với bất kỳ  $y \in Z_{26}$ , ta muốn có đồng nhất thức sau:

$$ax + b \equiv y \pmod{26}$$

phải có nghiệm  $x$  duy nhất. Đồng dư thức này tương đương với:

$$ax \equiv y - b \pmod{26}$$

Vì  $y$  thay đổi trên  $Z_{26}$  nên  $y - b$  cũng thay đổi trên  $Z_{26}$ . Bởi vậy, ta chỉ cần nghiên cứu phương trình đồng dư:

$$ax \equiv y \pmod{26} \quad (y \in Z_{26})$$

Ta biết rằng, phương trình này có một nghiệm duy nhất đối với mỗi  $y$  khi và chỉ khi  $\text{UCLN}(a, 26) = 1$  (ở đây hàm UCLN là ước chung lớn nhất của các biến của nó).

Trước tiên ta giả sử rằng,  $\text{UCLN}(a, 26) = d > 1$ . Khi đó, đồng dư thức  $ax \equiv 0 \pmod{26}$  sẽ có ít nhất hai nghiệm phân biệt trong  $Z_{26}$  là  $x = 0$  và  $x = 26/d$ .

Trong trường hợp này,  $e(x) = ax + b \pmod{26}$  không phải là một hàm đơn ánh và bởi vậy nó không thể là hàm mã hoá hợp lệ.

**Ví dụ 2:** Do  $\text{UCLN}(4, 26) = 2$  nên  $4x + 7$  không là hàm mã hoá hợp lệ:  $x$  và  $x + 13$  sẽ mã hoá thành cùng một giá trị đối với bất kỳ  $x \in Z_{26}$ .

Ta giả thiết  $\text{UCLN}(a, 26) = 1$ . Giả sử với  $x_1$  và  $x_2$  nào đó thoả mãn:

$$ax_1 \equiv ax_2 \pmod{26}$$

Khi đó:

$$a(x_1 - x_2) \equiv 0 \pmod{26}$$

bởi vậy

$$26 \mid a(x_1 - x_2)$$

Bây giờ ta sẽ sử dụng một tính chất của phép chia sau: Nếu  $\text{USLN}(a, b) = 1$  và  $a \mid bc$  thì  $a \mid c$ . Vì  $26 \mid a(x_1 - x_2)$  và  $\text{UCLN}(a, 26) = 1$  nên ta có:

$$26 \mid (x_1 - x_2)$$

tức là

$$x_1 \equiv x_2 \pmod{26}$$

Tới đây ta đã chứng tỏ rằng, nếu  $\text{UCLN}(a, 26) = 1$  thì một đồng dư thức dạng  $ax \equiv y \pmod{26}$  chỉ có (nhiều nhất) một nghiệm trong  $Z_{26}$ . Do đó, nếu ta cho x thay đổi trên  $Z_{26}$  thì  $ax \pmod{26}$  sẽ nhận được 26 giá trị khác nhau theo modulo 26 và đồng dư thức  $ax \equiv y \pmod{26}$  chỉ có một nghiệm y duy nhất.

Không có gì đặc biệt đối với số 26 trong khẳng định này. Bởi vậy, bằng cách tương tự, ta có thể chứng minh được kết quả sau:

### **Định lý 2.1:**

*Đồng dư thức  $ax \equiv b \pmod{m}$  chỉ có một nghiệm duy nhất  $x \in Z_m$  với mọi  $b \in Z_m$  khi và chỉ khi  $\text{UCLN}(a, m) = 1$ .*

Vì  $26 = 2 \times 13$  nên các giá trị  $a \in Z_{26}$  thoả mãn  $\text{UCLN}(a, 26) = 1$  là  $a = 1, 3, 5, 7, 9, 11, 15, 17, 19, 21, 23$  và  $25$ . Tham số b có thể là một phần tử bất kỳ trong  $Z_{26}$ . Như vậy, mã Affine có  $12 \times 26 = 312$  khoá có thể (dĩ nhiên, con số này là quá nhỏ để bảo đảm an toàn).

Bây giờ, ta sẽ xét bài toán chung với modulo m. Ta cần một định nghĩa khác trong lý thuyết số.

### **Định nghĩa 2.2:**

*Giả sử  $a \geq 1$  và  $m \geq 2$  là các số nguyên.  $\text{UCLN}(a, m) = 1$  thì ta nói rằng  $a$  và  $m$  là nguyên tố cùng nhau. Số các số nguyên trong  $Z_m$  nguyên tố cùng nhau với  $m$  thường được ký hiệu là  $\phi(m)$  (hàm này được gọi là hàm phi-Euler).*

Một kết quả quan trọng trong lý thuyết số cho ta giá trị của  $\phi(m)$  theo các thừa số trong phép phân tích theo luỹ thừa các số nguyên tố của m. (Một số nguyên  $p > 1$  là số

nguyên tố nếu nó không có ước dương nào khác ngoài 1 và p). Mọi số nguyên  $m > 1$  có thể phân tích được thành tích của các luỹ thừa các số nguyên tố theo cách duy nhất. Ví dụ  $60 = 2^3 \times 3 \times 5$  và  $98 = 2 \times 7^2$ .

Ta sẽ ghi lại công thức cho  $\phi(m)$  trong định lí sau:

**Định lý 2.2:**

$$\text{Giả sử } m = \prod_{i=1}^n p_i^{e_i} \quad (2.1)$$

Trong đó các số nguyên tố  $p_i$  khác nhau và  $e_i > 0, 1 \leq i \leq n$ . Khi đó :

$$\phi(m) = \prod_{i=1}^n (p_i^{e_i} - p_i^{e_i - 1}) \quad (2.2)$$

Định lý này cho thấy rằng, số khoá trong mã Affine trên  $Z_m$  bằng  $m\phi(m)$ , trong đó  $\phi(m)$  được cho theo công thức trên. (Số các phép chọn của b là m và số các phép chọn của a là  $\phi(m)$  với hàm mã hoá là  $e(x) = ax + b$ ).

Ví dụ, khi  $m = 60$ ,  $\phi(60) = 2 \times 2 \times 4 = 16$  và số các khoá trong mã Affine là 960.

Bây giờ, ta sẽ xét xem các phép toán giải mã trong mật mã Affine với modulo  $m = 26$ . Giả sử  $\text{UCLN}(a, 26) = 1$ . Để giải mã cần giải phương trình đồng dư  $y \equiv ax + b \pmod{26}$  theo x. Từ thảo luận trên thấy rằng, phương trình này có một nghiệm duy nhất trong  $Z_{26}$ . Tuy nhiên, ta vẫn chưa biết một phương pháp hữu hiệu để tìm nghiệm. Điều cần thiết ở đây là có một thuật toán hữu hiệu để làm việc đó. Rất may là một số kết quả tiếp sau về số học modulo sẽ cung cấp một thuật toán giải mã hữu hiệu cần tìm.

**Định nghĩa 2.3:**

*Giả sử  $a \in Z_m$ . Phân tử nghịch đảo (theo phép nhân) của a là phân tử  $a^{-1} \in Z_m$  sao cho  $a \cdot a^{-1} = a^{-1} \cdot a = 1 \pmod{m}$ .*

Bằng các lý luận tương tự như trên, có thể chứng tỏ rằng a có nghịch đảo theo modulo m khi và chỉ khi  $\text{UCLN}(a, m) = 1$ , và nếu nghịch đảo này tồn tại thì nó phải

là duy nhất. Ta cũng thấy rằng, nếu  $b = a^{-1}$  thì  $a = b^{-1}$ . Nếu  $p$  là số nguyên tố thì mọi phần tử khác không của  $Z_p$  đều có nghịch đảo. Một vành trong đó mọi phần tử khác 0 đều có nghịch đảo được gọi là một trường.

Trong [3] có một thuật toán hữu hiệu để tính các nghịch đảo của  $Z_m$  với  $m$  tùy ý. Tuy nhiên, trong  $Z_{26}$ , chỉ bằng phương pháp thử và sai cũng có thể tìm được các nghịch đảo của các phần tử nguyên tố cùng nhau với 26:  $1^{-1} = 1$ ,  $3^{-1} = 9$ ,  $5^{-1} = 21$ ,  $7^{-1} = 15$ ,  $11^{-1} = 19$ ,  $17^{-1} = 23$ ,  $25^{-1} = 25$ . (Có thể dễ dàng kiểm chứng lại điều này, ví dụ:  $7 \times 5 = 105 \equiv 1 \pmod{26}$ , bởi vậy  $7^{-1} = 15$ ).

Xét phương trình đồng dư  $y \equiv ax + b \pmod{26}$ . Phương trình này tương đương với

$$ax \equiv y - b \pmod{26}$$

vì  $\text{UCLN}(a, 26) = 1$  nên  $a$  có nghịch đảo theo modulo 26. Nhân cả hai vế của đồng dư thức với  $a^{-1}$ , ta có:

$$a^{-1}(ax) \equiv a^{-1}(y - b) \pmod{26}$$

Áp dụng tính kết hợp của phép nhân modulo:

$$a^{-1}(ax) \equiv (a^{-1} \cdot a)x \equiv 1 \cdot x \equiv x \pmod{26}$$

Kết quả là  $x \equiv a^{-1}(y - b) \pmod{26}$ . Đây là một công thức tường minh cho  $x$ . Như vậy hàm giải mã là:

$$d(y) = a^{-1}(y - b) \pmod{26}$$

Hình 2 .3 cho mô tả đầy đủ về mã Affine.

Cho  $P = C = Z_{26}$  và giả sử:  $K = \{(a, b) \in Z_{26} \times Z_{26} : \text{UCLN}(a, 26) = 1\}$

Với  $k = (a, b) \in K$  ta định nghĩa:

$$e_k(x) = ax + b \pmod{26}$$

Hình 0-1. Mã Affine

Sau đây là một ví dụ nhỏ.

**Ví dụ 3:**

Giả sử  $k = (7, 3)$ . Như đã nêu ở trên,  $7^{-1} \bmod 26 = 15$ . Hàm mã hoá là:

$$e_k(x) = 7x + 3$$

Và hàm giải mã tương ứng là:

$$d_k(x) = 15(y - 3) = 15y - 19$$

Ở đây, tất cả các phép toán đều thực hiện trên  $Z_{26}$ . Ta sẽ kiểm tra liệu  $d_k(e_k(x)) = x$  với mọi  $x \in Z_{26}$  không?. Dùng các tính toán trên  $Z_{26}$ , ta có:

$$\begin{aligned} d_k(e_k(x)) &= d_k(7x + 3) \\ &= 15(7x + 3) - 19 \\ &= x + 45 - 19 \\ &= x \end{aligned}$$

Để minh họa, ta hãy mã hoá bản rõ "hot". Trước tiên, biến đổi các chữ h, o, t thành các thặng dư theo modulo 26. Ta được các số tương ứng là 7, 14 và 19. Bây giờ sẽ mã hoá:

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

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

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

Bởi vậy, ba ký hiệu của bản mã là 0, 23 và 6, tương ứng với xâu ký tự AXG. Việc giải mã sẽ do bạn đọc thực hiện như một bài tập.

#### 4.2.2 Các phép thay thế đơn giản khác

##### 4.2.2.1 Mã thay thế (MTT)

Cho  $P = C = Z_{26}$ . K chứa mọi hoán vị có thể có của 26 kí tự từ 0 đến 25. Với mỗi phép hoán vị  $\pi \in K$ , ta định nghĩa:

$$e_\pi(x) = \pi(x)$$

$$\text{và } d_\pi(y) = \pi^{-1}(y)$$

Hình 0-2. Mã thay thế

Sau đây là một ví dụ về phép hoán vị ngẫu nhiên  $\pi$  tạo nên một hàm mã hoá (tương tự như trên, các ký tự của bản rõ được viết bằng chữ thường, còn các ký tự của bản mã được viết bằng chữ in hoa).

Kí tự bản rõ	a	b	c	d	e	f	g	h	i	j	k	l	m
--------------	---	---	---	---	---	---	---	---	---	---	---	---	---

Kí tự bản mã	X	N	Y	A	H	P	O	G	Z	Q	W	B	T
Kí tự bản rõ	n	o	p	q	r	s	t	u	v	w	x	y	z
Kí tự bản mã	S	F	L	R	C	V	M	U	E	K	J	D	I

Như vậy,  $e_\pi(a) = X, e_\pi(b) = N, \dots$

Hàm giải mã là phép hoán vị ngược. Điều này được thực hiện bằng cách viết hàng thứ hai lên trước rồi sắp xếp theo thứ tự chữ cái. Ta có:

Kí tự bản mã	A	B	C	D	E	F	G	H	I	J	K	L	M
Kí tự bản rõ	d	l	r	y	v	o	h	e	z	x	w	p	t
Kí tự bản mã	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
Kí tự bản rõ	b	g	f	j	q	n	m	u	s	k	a	c	i

#### Ví dụ 4:

Với phép thay thế trên, từ bản rõ:

meetmeatsunset

ta thu được bản mã sau:

THHMTHXMVUSVHM

Sử dụng phép hoán vị ngược, ta dễ dàng tìm lại được bản rõ ban đầu.

Mỗi khoá của mã thay thế là một phép hoán vị của 26 ký tự. Số các hoán vị này là  $26! > 4.10^{26}$ . Đây là một số rất lớn nên khó có thể tìm được khoá bằng phép tìm khoá vét cạn. Tuy nhiên, bằng phương pháp thông kê, ta có thể dễ dàng thám được các bản mã loại này.

### A. CÁC HỆ MẬT THAY THẾ ĐA BIỂU

#### 1. Hệ mật Vigenere

Trong hai hệ MDV và MTT ở trên, một khi khoá đã được chọn thì mỗi ký tự sẽ được ánh xạ vào một ký tự duy nhất. Vì vậy, các hệ trên còn được gọi là các hệ thay thế đơn biếu. Sau đây ta sẽ trình bày một hệ thay thế đa biếu được gọi là hệ mật Vigenere.

Sử dụng phép tương ứng  $A \leftrightarrow 0, B \leftrightarrow 1, \dots, Z \leftrightarrow 25$  mô tả ở trên, ta có thể gắn cho mỗi khoá k một chuỗi ký tự có độ dài m, được gọi là từ khoá. Mật mã Vigenère sẽ mã hoá đồng thời m ký tự: mỗi phần tử của bản rõ tương đương với m ký tự.

#### Ví dụ:

Giả sử m = 6 và từ khoá là CIPHER. Từ khoá này tương ứng với dãy số k = (2, 8, 15, 7, 4, 17). Giả sử bản rõ là:

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

Bản rõ	12	4	4	19	12	4	0	19	18	20	13	18	4	19
Khóa	2	8	15	7	4	17	2	8	15	7	4	17	2	8
Bản mã	14	12	19	0	16	21	2	1	7	1	17	9	6	1

Như vậy, dãy ký tự tương ứng với xâu bản mã sẽ là:

OMTAQVCBHBRJGB

Ta có thể mô tả mật mã Vigenere như sau:

Cho  $m$  là một số nguyên dương cố định nào đó. Ta định nghĩa  $P = C = K = (Z_{26})^n$

Với khóa  $k = (k_1, k_2, \dots, k_m)$  ta xác định:

$$e_k(x_1, x_2, \dots, x_m) = (x_1 + k_1, x_2 + k_2, \dots, x_m + k_m)$$

và

$$d_k(y_1, y_2, \dots, y_m) = (y_1 - k_1, y_2 - k_2, \dots, y_m - k_m)$$

**Chú ý:** Để giải mã, ta có thể dùng cùng từ khoá nhưng thay cho cộng, ta trừ nó theo modulo 26.

Ta thấy rằng, số các từ khoá có thể với độ dài  $m$  trong mật mã Vigenere là  $26^m$ . Bởi vậy, thậm chí với  $m$  khá nhỏ, phương pháp tìm kiếm vét cạn cũng yêu cầu thời gian khá lớn. Ví dụ, với  $m = 6$  thì không gian khoá cũng có kích thước lớn hơn  $3 \cdot 10^8$  khoá.

## 2. Hệ mật Hill

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

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

$$\begin{aligned} y_1 &= 11x_1 + 3x_2 \\ y_2 &= 8x_1 + 7x_2 \end{aligned}$$

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

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

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

$$(y_1, \dots, y_m) = (x_1, \dots, x_m) \begin{pmatrix} k_{1,1} & k_{1,2} & \dots & k_{1,m} \\ k_{2,1} & k_{2,2} & \dots & k_{2,m} \\ \vdots & \vdots & & \vdots \\ k_{m,1} & k_{m,2} & \dots & k_{m,m} \end{pmatrix}$$

Nói cách khác,  $y = xk$ .

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

Sau đây là một số định nghĩa về những khái niệm cần thiết lấy từ đại số tuyến tính. Nếu  $A = (a_{i,j})$  là một ma trận cấp  $l \times m$  và  $B = (b_{l,k})$  là một ma trận cấp  $m \times n$  thì tích ma trận  $AB = (c_{l,k})$  được định nghĩa theo công thức :

$$c_{i,k} = \sum_{j=1}^m a_{i,j} b_{j,k}$$

với  $1 \leq i \leq l$  và  $1 \leq k \leq n$ . Tức là các phần tử ở hàng i và cột thứ k của AB được tạo ra bằng cách lấy hàng thứ i của A và cột thứ k của B, sau đó nhân tương ứng các phần tử với nhau và cộng lại. Cần để ý rằng AB là một ma trận cấp  $l \times n$ .

Theo định nghĩa này, phép nhân ma trận là kết hợp (tức  $(AB)C = A(BC)$ ) nhưng nói chung là không giao hoán (không phải lúc nào  $AB = BA$ , thậm chí đối với ma trận vuông A và B).

Ma trận đơn vị  $m \times m$  (ký hiệu là  $I_m$ ) là ma trận cấp  $m \times m$  có các số 1 nằm ở đường chéo chính, và các số 0 ở vị trí còn lại. Như vậy, ma trận đơn vị  $2 \times 2$  là:

$$I_2 = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix}$$

$I_m$  được gọi là ma trận đơn vị vì  $AI_m = A$  với mọi ma trận cấp  $1 \times m$  và  $I_m B = B$  với mọi ma trận cấp  $m \times n$ . Ma trận nghịch đảo của ma trận  $A$  cấp  $m \times m$  (nếu tồn tại) là ma trận  $A^{-1}$  sao cho  $AA^{-1} = A^{-1}A = I_m$ . Không phải mọi ma trận đều có nghịch đảo, nhưng nếu tồn tại thì nó duy nhất.

Với các định nghĩa trên, có thể dễ dàng xây dựng công thức giải mã đẽ nêu: Vì  $y = xk$ , ta có thể nhân cả hai vế của đẳng thức với  $k^{-1}$  và nhận được:

$$yk^{-1} = (xk)k^{-1} = x(kk^{-1}) = xI_m = x$$

(Chú ý: sử dụng tính chất kết hợp)

Có thể thấy rằng, ma trận mã hoá ở trên có nghịch đảo trong  $Z_{26}$ :

$$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

vì

$$\begin{aligned} & \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix} = \begin{pmatrix} 11 \times 7 + 8 \times 23 & 11 \times 18 + 8 \times 11 \\ 3 \times 7 + 7 \times 23 & 3 \times 18 + 7 \times 11 \end{pmatrix} \\ & = \begin{pmatrix} 261 & 286 \\ 182 & 131 \end{pmatrix} = \begin{pmatrix} 1 & 0 \\ 0 & 1 \end{pmatrix} \end{aligned}$$

(Hãy nhớ rằng mọi phép toán số học đều được thực hiện theo modulo 26).

Sau đây là một ví dụ minh họa cho việc mã hoá và giải mã trong hệ mật mã Hill.

**Ví dụ :**

$$\text{Giả sử khoá } k = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

Từ các tính toán trên, ta có:

$$k^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

Giả sử cần mã hoá bản rõ "July". Ta có hai phần tử của bản rõ để mã hoá:  $(9, 20)$  (ứng với Ju) và  $(11, 24)$  (ứng với ly). Ta tính như sau:

$$(9 \ 20) \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = (99 + 60 \ 72 + 140) = (3 \ 4)$$

$$(11 \ 24) \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = (121 + 72 \ 88 + 168) = (11 \ 22)$$

Bởi vậy, bản mã của July là DELW. Để giải mã, Bob sẽ tính

$$(3 \ 4)k^{-1} = (9 \ 20) \text{ và } (11 \ 22)k^{-1} = (11 \ 24)$$

Như vậy, Bob đã nhận được bản đúng.

Cho tới lúc này, ta đã chỉ ra rằng có thể thực hiện phép giải mã nếu k có một nghịch đảo. Trên thực tế, để phép giải mã là có thể thực hiện được, điều kiện cần là k phải có nghịch đảo. (Điều này dễ dàng rút ra từ đại số tuyến tính sơ cấp, tuy nhiên sẽ không chứng minh ở đây). Bởi vậy, ta chỉ quan tâm tới các ma trận k khả nghịch.

Tính khả nghịch của một ma trận vuông phụ thuộc vào giá trị định thức của nó. Để tránh sự tổng quát hoá không cần thiết, ta chỉ giới hạn trong trường hợp  $2 \times 2$ .

### **Định nghĩa 2.3:**

*Định thức của ma trận A =  $(a_{i,j})$  cấp  $2 \times 2$  là giá trị*

$$\det A = a_{1,1}a_{2,2} - a_{1,2}a_{2,1}$$

*Nhận xét:* Định thức của một ma trận vuông cấp m x m có thể được tính theo các phép toán hàng sơ cấp (hãy xem một giáo trình bất kỳ về đại số tuyến tính).

Hai tính chất quan trọng của định thức là  $\det I_m = 1$  và quy tắc nhân  $\det(AB) = \det A \times \det B$ .

Một ma trận thực k là có nghịch đảo khi và chỉ khi định thức của nó khác 0. Tuy nhiên, điều quan trọng cần nhớ là ta đang làm việc trên  $Z_{26}$ . Kết quả tương ứng là ma trận k có nghịch đảo theo modulo 26 khi và chỉ khi.  $\text{UCLN}(\det k, 26) = 1$

Sau đây sẽ chứng minh ngắn gọn kết quả này.

Trước tiên, giả sử rằng  $\text{UCLN}(\det k, 26) = 1$ . Khi đó  $\det k$  có nghịch đảo trong  $Z_{26}$ . Với  $1 \leq i \leq m$ ,  $1 \leq j \leq m$ , định nghĩa  $k_{ij}$  là ma trận thu được từ k bằng cách loại bỏ hàng thứ i và cột thứ j. Và định nghĩa ma trận  $k^*$  có phần tử  $(i,j)$  của nó nhận giá trị  $(-1)^{i+j} \det k_{ji}$  ( $k^*$  được gọi là ma trận bù đại số của k). Khi đó, có thể chứng tỏ rằng:

$$k^{-1} = (\det k)^{-1} k^*$$

Bởi vậy  $k$  là khả nghịch.

Ngược lại,  $k$  có nghịch đảo  $k^{-1}$ . Theo quy tắc nhân của định thức:

$$1 = \det I = \det(k k^{-1}) = \det k \det k^{-1}$$

Bởi vậy  $\det k$  có nghịch đảo trong  $Z_{26}$ .

*Nhận xét:* Công thức đối với  $k^{-1}$  ở trên không phải là một công thức tính toán có hiệu quả trừ các trường hợp  $m$  nhỏ (chẳng hạn  $m = 2, 3$ ). Với  $m$  lớn, phương pháp thích hợp để tính các ma trận nghịch đảo phải dựa vào các phép toán hàng sơ cấp.

Trong trường hợp  $2 \times 2$ , ta có công thức sau:

**Định lý 2.3:**

Giả sử  $A = (a_{ij})$  là một ma trận cấp  $2 \times 2$  trên  $Z_{26}$  sao cho

$\det A = a_{1,1}a_{2,2} - a_{1,2}a_{2,1}$  có nghịch đảo. Khi đó:

$$A^{-1} = (\det A)^{-1} \begin{pmatrix} a_{2,2} & -a_{1,2} \\ -a_{2,1} & a_{1,1} \end{pmatrix}$$

Trở lại ví dụ đã xét ở trên. Trước hết ta có:

$$\begin{aligned} \det \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} &= 11 \times 7 - 8 \times 3 \bmod 2 \\ &= 77 - 24 \bmod 26 = 53 \bmod 26 \\ &= 1 \end{aligned}$$

Vì  $1^{-1} \bmod 26 = 1$  nên ma trận nghịch đảo là:

$$\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}^{-1} = \begin{pmatrix} 7 & 18 \\ 23 & 11 \end{pmatrix}$$

Đây chính là ma trận đã có ở trên.

Bây giờ ta sẽ mô tả chính xác mật mã Hill trên  $Z_{26}$  (hình 2.6).

Cho  $m$  là một số nguyên dương cố định. Cho  $P = C = (Z_{26})^m$  và cho:

$K = \{\text{các ma trận khả nghịch cấp } m \times m \text{ trên } Z_{26}\}$

Với một khóa  $k \in K$  ta xác định:

$$e_k(x) = xk$$

## Hình 0-4. Mật mã Hill

### 3. Hệ mật Playfair

Mật mã Playfair hay hình vuông Playfair là một kĩ thuật mã hoá đối xứng thủ công và là thuật toán thay thế chữ ghép đầu tiên. Hệ mật này được Charles Wheatstone phát minh ra năm 1854 nhưng lại được gọi theo tên của Lord Playfair người đã phổ biến để sử dụng làm mật mã.

Kĩ thuật này mã hoá từng cặp kí tự (bộ ghép) thay cho các kí tự đơn trong mã hoá thay thế đơn và cách sử dụng phức tạp hơn mã hoá Vigenere. Tần công Playfair khó vì việc phân tích tần suất vẫn thường sử dụng cho mật mã thay thế đơn không dùng được tuy nhiên phân tích tần suất các bộ chữ ghép thì vẫn có thể nhưng khó hơn nhiều và nói chung phải cần số lượng bản mã rất lớn.

#### Cách dùng Playfair

Playfair dùng một bảng  $5 \times 5$  chứa từ hoặc cụm từ khóa. Ta cần phải nhớ từ khóa và 4 quy tắc thực hiện.

Tạo bảng khóa chứa 25 chữ cái khác nhau (bỏ chữ “Q” trong bảng chữ cái hoặc phiên bản khác thì coi chữ “I” và chữ “J” thuộc cùng một ô trong bảng khóa), trước tiên lấp đầy bảng bởi các chữ cái của từ khóa (bỏ các chữ cái lặp lại) rồi lấp các chữ cái còn lại của bảng chữ cái vào các chỗ trống của bảng khóa theo thứ tự. Khóa có thể được viết ở các hàng đầu của bảng, từ trái qua phải.

Mã hóa thông điệp, tách thông điệp thành các nhóm gồm 2 kí tự rồi ánh xạ chúng vào bảng khóa.

#### Bốn quy tắc:

1) Xen chữ “X” vào giữa hai chữ cái giống nhau (hoặc chèn vào sau nếu chỉ còn lại một chữ cái cuối cùng) của bản rõ rồi mã hóa cặp mới và tiếp tục thực hiện.

*Chú ý:* Phiên bản khác thì chèn chữ “Q” thay cho chữ “X”.

2) Nếu hai chữ cái của một cặp xuất hiện trên cùng một hàng của khóa thì thay thế chúng bằng các chữ cái ở ngay bên phải tương ứng (nếu chữ cái nằm ở tận cùng bên phải của hàng thì thay bằng chữ cái đầu tiên của hàng đó)

3) Nếu hai chữ cái của cặp xuất hiện trên cùng một cột của khóa thì thay thế chúng bởi các chữ ở ngay dưới tương ứng (nếu chữ cái nằm ở tận cùng bên dưới của cột thì thay thế bởi chữ cái đầu tiên của cột đó)

4) Nếu hai chữ cái của cặp không cùng hàng và cột thì thay thế chúng bởi các chữ cái trên cùng hàng tương ứng và ở các góc của hình chữ nhật mà hai chữ cái của cặp này tạo nên trên khóa.

### Ví dụ

Dùng khóa “playfair example” để mã hóa bản rõ “Hide the gold in the tree stump”.

Khóa sẽ được bố trí như sau:

P L A Y F

I R E X M

B C D G H

J K N O S

T U V W Z

Ở đây các kí tự lặp lại sẽ bị bỏ đi, sau đó thêm các chữ cái trong bảng chữ cái mà chưa xuất hiện trong khóa sau khi đã bỏ các chữ lặp lại vào để lấp đầy ô trống của bảng khóa  $5 \times 5$ .

Bản rõ được tách như sau:

H I D E T H E G O L D I N T H E T R E E S T U M P

Nhưng có cặp EE thì ta phải xen chữ X vào giữa, kết quả được là:

H I D E T H E G O L D I N T H E T R E X E S T U M P

Mã hóa:

Chiếu từng cặp của bản rõ sau khi đã tách vào bảng khóa theo các quy tắc mã hóa ta được:

H I → BM (Khác hàng, khác cột)

D E → N D (Cùng cột)

T H → Z B (Khác hàng, khác cột)

E G → X D (Khác hàng, khác cột)

O L → K Y (Khác hàng, khác cột)

D I → B E (Khác hàng, khác cột)

N T → J V (Khác hàng, khác cột)

H E → D M (Khác hàng, khác cột)

T R → U I (Khác hàng, khác cột)

E X → X M (Cùng hàng)

E S → M N (Khác hàng, khác cột)

T U → U V (Cùng hàng)

M P → I F (Khác hàng, khác cột)

Vậy bản mã là: "BMNDZBXDKYBEJVDMUIXMMNUVIF"

*Giải mã:* Vẫn dùng khóa giống như mã hóa còn 4 quy tắc thì thay thế theo chiều ngược lại.

### *Thám mã Playfair*

Việc lấy được khóa là tương đối dễ nếu biết cả bản rõ và bản mã. Nếu chỉ biết bản mã thì phương pháp giải mã theo vét cạn bao gồm việc tìm kiếm qua không gian khóa và phân tích tần suất của các chữ ghép trong ngôn ngữ giả định của thông điệp gốc.

Giải mã Playfair dựa vào sự liên quan của các chữ cái nên việc xác định các xâu bản rõ ứng cử viên dễ dàng hơn. Đặc biệt là chữ ghép Playfair và ngược của chữ ghép đó (ví dụ AB và BA) sẽ giải mã thành cùng một kiểu mẫu chữ cái trong bản rõ (Ví dụ RE và ER). Trong tiếng Anh, có rất nhiều từ chứa những bộ chữ ghép ngược này như RECEIVER và DEPARTED. Việc xác định những bộ chữ ghép ngược mà gần nhau trong bản mã và ghép kiểu mẫu thành một danh sách các từ rõ đã biết chứa kiểu mẫu là đơn giản từ đó đưa ra được các xâu bản rõ có thể rồi sẽ xác định khóa.

Một cách tiếp cận khác là phương pháp Shotgun hill climbing. Bắt đầu với hình vuông các chữ cái ngẫu nhiên sau đó thực hiện những sự thay đổi nhỏ (ví dụ như chuyển các chữ cái, các hàng hay phản xạ toàn bộ hình vuông) để thấy được nếu bản rõ ứng cử viên giống bản rõ chuẩn hơn so với trước khi thay đổi (có thể bằng cách so sánh các nhóm ba thành lược đồ tần suất đã biết). Nếu hình vuông mới có sự cải tiến thì nó sẽ được chấp nhận và sau sẽ được biến đổi thêm để tìm ra một ứng cử viên tốt hơn. Cuối cùng là dù chọn phương pháp phân loại nào thì cũng tìm ra bản rõ hoặc một văn bản rất gần bản rõ với khả năng đúng là lớn nhất. Máy tính có thể chấp nhận thuật toán này để phá các mật mã Playfair với số lượng văn bản tương đối nhỏ.

Phương pháp Playfair thường được áp dụng trong việc giải các trò chơi đố các ô chữ.

## B. CÁC HỆ MẬT THAY THẾ KHÔNG TUẦN HOÀN

### 1. Hệ mật khóa tự sinh

$$\text{Cho } P = C = K = L = Z_{26}$$

$$\text{Cho } z_1 = k \text{ và } z_i = x_{i-1} \quad (i \geq 2)$$

Với  $0 \leq z \leq 25$  ta xác định

$$e_z(x) = x + z \bmod 26$$

$$d_z(y) = y - z \bmod 26$$

### Hình 0-5. Mật mã khóa tự sinh

Lý do sử dụng thuật ngữ "khoá tự sinh" là ở chỗ bản rõ được dùng làm khoá (ngoài "khoá khởi thuỷ" ban đầu k).

Sau đây là một ví dụ minh họa.

#### Ví dụ :

Giả sử khoá là  $k = 8$  và bản rõ là *rendezvous*. Trước tiên, ta biến đổi bản rõ thành dãy các số nguyên:

17 4 13 3 4 25 21 14 20 18

Dòng khoá như sau:

8 17 4 13 3 4 25 21 14 20

Bây giờ ta cộng các phần tử tương ứng rồi rút gọn theo modulo 26:

25 21 17 16 7 3 20 9 8 12

Bản mã ở dạng ký tự là: ZVRQHDUJIM .

Bây giờ ta xem Bob giải mã bản mã này như thế nào. Trước tiên, Bob biến đổi xâu kí tự thành dãy số:

25 21 17 16 7 3 20 9 8 12

Sau đó anh ta tính:

$$x_1 = d_8(25) = 25 - 8 \bmod 26 = 17$$

$$\text{và } x_2 = d_{17}(21) = 21 - 17 \bmod 26 = 4$$

và cứ tiếp tục như vậy.

Mỗi khi Bob nhận được một ký tự của bản rõ, cô ta sẽ dùng nó làm phần tử tiếp theo của dòng khoá. Dĩ nhiên là mã dùng khoá tự sinh là không an toàn do chỉ có 26 khoá.

## 2. Hệ mật Vernam

Giả sử  $n \geq 1$  là số nguyên và  $P = C = K = (Z_2)^n$ . Với  $K$  thuộc  $(Z_2)^n$ , ta xác định  $e_K(x)$  là tổng véc tơ theo modulo 2 của  $K$  và  $x$  (hay tương đương với phép hoặc loại trừ của hai dãy bit tương ứng). Như vậy, nếu  $x = (x_1, \dots, x_n)$  và  $K = (K_1, \dots, K_n)$  thì:

$$e_K(x) = (x_1 + K_1, \dots, x_n + K_n) \bmod 2.$$

Phép mã hóa là đồng nhất với phép giải mã. Nếu  $y = (y_1, \dots, y_n)$  thì:

$$d_K(y) = (y_1 + K_1, \dots, y_n + K_n) \bmod 2.$$

### Hình 0-6. Hệ mật OTP

Sử dụng định lý 1.3, dễ dàng thấy rằng OTP có độ mật hoàn thiện. Hệ thống này rất hấp dẫn do dễ thực hiện mã và giải mã.

Vernam đó đăng ký phát minh của mình với hy vọng rằng nó sẽ có ứng dụng thương mại rộng rãi. Đáng tiếc là có những nhược điểm quan trọng đối với các hệ mật an toàn không điều kiện, chẳng hạn như OTP. Điều kiện  $|K| \geq |P|$  có nghĩa là lượng khóa (cần được thông báo một cách bí mật) cũng lớn như bản rõ. Ví dụ, trong trường hợp hệ OTP, ta cần  $n$  bit khoá để mã hóa  $n$  bit của bản rõ. Vấn đề này sẽ không quan trọng nếu có thể dùng cùng một khoá để mã hóa các bản tin khác nhau; tuy nhiên, độ an toàn của các hệ mật an toàn không điều kiện lại phụ thuộc vào một thực tế là mỗi khoá chỉ được dùng cho một lần mã. Ví dụ OTP không thể đứng vững trước tấn công chỉ với bản rõ đã biết vì ta có thể tính được  $K$  bằng phép hoặc loại trừ xâu bít bất kỳ  $x$  và  $e_K(x)$ . Bởi vậy, cần phải tạo một khoá mới và thông báo nó trên một kênh bảo mật đối với mỗi bản tin trước khi gửi đi. Điều này tạo ra khó khăn cho vấn đề quản lý khoá và gây hạn chế cho việc sử dụng rộng rãi OTP. Tuy nhiên OTP vẫn được áp dụng trong lĩnh vực quân sự và ngoại giao, ở những lĩnh vực này độ an toàn không điều kiện có tầm quan trọng rất lớn.

## C. CÁC HỆ MẬT HOÁN VỊ

Khác với MTT, ý tưởng của mã hoán vị (MHV) là giữ các ký tự của bản rõ không thay đổi nhưng sẽ thay đổi vị trí của chúng bằng cách sắp xếp lại các ký tự này. Ở đây không có một phép toán đại số nào cần thực hiện khi mã hoá và giải mã.

**Ví dụ :**

Giả sử  $m = 6$  và khoá là phép hoán vị sau:

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

Khi đó, phép hoán vị ngược sẽ là:

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

Giả sử ta có bản rõ: asecondclasscarriageontheretrain  
Trước tiên, ta nhóm bản rõ thành các nhóm 6 ký tự:

a sec on|dclass|carria|geonth|etrain

Sau đó, mỗi nhóm 6 chữ cái lại được sắp xếp lại theo phép hoán vị  $\pi$ , ta có:

EOANCS|LSDSAC|RICARA|OTGHNE|RIENAT|

Cuối cùng, ta có bản mã sau:

EOANCSLSDSACRICARAOTGHNERIENAT

Khi sử dụng phép hoán vị ngược  $\pi^{-1}$  trên dãy bản mã (sau khi đã nhóm lại theo các nhóm 6 ký tự), ta sẽ nhận lại được bản rõ ban đầu.

Từ ví dụ trên, ta có thể định nghĩa MHV như sau:

Cho  $m$  là một số nguyên dương xác định nào đó.

Cho  $P = C = (Z_{26})^m$  và cho  $\pi$  là tất cả các hoán vị có thể có của  $\{1, 2, \dots, m\}$

Đối với một khóa  $\pi$  (tức là một phép hoán vị nào đó), ta xác định:

$$e_\pi = (x_1, \dots, x_m) = (x_{\pi(1)}, \dots, x_{\pi(m)})$$

$$\text{Và } d_\pi = (x_1, \dots, x_m) = (y_{\pi^{-1}(1)}, \dots, y_{\pi^{-1}(m)})$$

Trong đó  $\pi^{-1}$  là phép hoán vị ngược của  $\pi$

Hình 0-7. Mã hoán vị

#### D. CÁC HỆ MẬT TÍCH

Một phát minh khác do Shannon đưa ra trong bài báo của mình năm 1949 là ý tưởng kết hợp các hệ mật bằng cách tạo tích của chúng. Ý tưởng này có tầm quan trọng to lớn trong việc thiết kế các hệ mật hiện nay (chẳng hạn, chuẩn mã dữ liệu - DES ).

Để đơn giản, trong phần này chỉ hạn chế xét các hệ mật trong đó  $C = P$ : các hệ mật loại này được gọi là tự đồng cấu. Giả sử  $S_1 = (P, P, K_1, E_1, D_1)$  và  $S_2 = (P, P, K_2, E_2, D_2)$  là hai hệ mật tự đồng cấu có cùng các không gian bản mã và rõ. Khi đó, tích của  $S_1$  và  $S_2$  (kí hiệu là  $S_1 \times S_2$ ) được xác định là hệ mật sau:

$$(P, P, K_1 \times K_2, E, D)$$

Khoá của hệ mật tích có dạng  $k = (k_1, k_2)$  trong đó  $k_1 \in K_1$  và  $k_2 \in K_2$ . Các quy tắc mã và giải mã của hệ mật tích được xác định như sau: Với mỗi  $k = (k_1, k_2)$ , ta có một quy tắc mã  $e_k$  xác định theo công thức:

$$e_{(k_1, k_2)}(x) = e_{k_2}(e_{k_1}(x))$$

và quy tắc giải mã:

$$d_{(k_1, k_2)}(y) = d_{k_1}(d_{k_2}(y))$$

Nghĩa là, trước tiên ta mã hóa x bằng  $e_{(k_1, k_2)}(x) = e_{k_1}(e_{k_2}(x))$  rồi mã  $(e_{k_1}(x))$  qua bằng  $e_{k_2}$ . Quá trình giải mã tương tự nhưng thực hiện theo thứ tự ngược lại:  $d_{k_1}(d_{k_2}(e_{k_1}(x))) = d_{k_1}(e_{k_1}(x)) = x$

Ta biết rằng, các hệ mật đều có các phân bố xác suất ứng với các không gian khoá của chúng. Bởi vậy, cần phải xác định phân bố xác suất cho không gian khoá K của hệ mật tích. Hiển nhiên ta có thể viết:

$$p_K(k_1, k_2) = p_{K_1}(k_1) \times p_{K_2}(k_2)$$

Nói một cách khác, ta chọn  $k_1$  có phân bố  $p_{K_1}$  rồi chọn một cách độc lập  $k_2$  có phân bố  $p_{K_2}(k_2)$ .

Sau đây là một ví dụ đơn giản để minh họa khái niệm hệ mật tích. Giả sử định nghĩa hệ mật mã nhân như trong hình 2.10 sau.

Giả sử  $P = C = Z_{26}$  và giả sử:

$$K = \{ a \in Z_{26} : \text{UCLN}(a, 26) = 1 \}$$

Với  $a \in K$ , ta xác định:  $e_a(x) = ax \bmod 26$

#### Hình. Mã nhân

Cho M là một hệ mã nhân (với các khoá được chọn đồng xác suất) và S là MDV (với các khoá chọn đồng xác suất). Khi đó dễ dàng thấy rằng  $M \times S$  chính là hệ mã Affine (cùng với các khoá được chọn đồng xác suất). Tuy nhiên, việc chứng tỏ  $S \times M$  cũng là hệ mã Affine khó hơn một chút (cũng với các khoá đồng xác suất).

Ta sẽ chứng minh các khẳng định này. Một khoá dịch vòng là phần tử  $k \in Z_{26}$  và quy tắc giải mã tương ứng là  $e_k(x) = x + k \bmod 26$ . Còn khoá trong hệ mã nhân là phần tử  $a \in Z_{26}$  sao cho  $\text{UCLN}(a, 26) = 1$ . Quy tắc mã tương ứng là  $e_a(x) = a \bmod 26$ . Bởi vậy, một khoá trong mã tích  $M \times S$  có dạng  $(a, k)$ , trong đó

$$e_{(a, k)}(x) = ax + k \bmod 26$$

Đây chính là định nghĩa về khoá trong hệ mã Affine. Hơn nữa, xác suất của một khoá trong hệ mã Affine là:  $1/312 = (1/12) \times (1/26)$ . Đó là tích của xác suất tương ứng của các khoá a và k. Bởi vậy  $M \times S$  là hệ mã Affine.

Bây giờ ta sẽ xét  $S \times M$ . Một khoá này trong hệ mã này có dạng  $(k, a)$ , trong đó :

$$e_{(k, a)}(x) = a(x + k) \bmod 26$$

Như vậy, khoá  $(k, a)$  của mã tích  $S \times M$  đồng nhất với khoá  $(a, ak)$  của hệ mã Affine. Vẫn đề còn lại là phải chứng tỏ rằng mỗi khoá của mã Affine xuất hiện với cùng

xác suất  $1/312$  như trong mã tích  $S \times M$ . Nhận thấy rằng  $ak = k_1$  khi và chỉ khi  $k = a^{-1}k_1$ , (hãy nhớ lại rằng  $\text{UCLN}(a, 26) = 1$ , bởi vậy  $a$  có phần tử nghịch đảo). Nói cách khác, khoá  $(a, k_1)$  của hệ mã Affine tương đương với khoá  $(a^{-1}k_1, a)$  của mã tích  $S \times M$ . Bởi vậy, ta có một song ánh giữa hai không gian khoá. Vì mỗi khoá là đồng xác suất nên có thể thấy rằng  $S \times M$  thực sự là mã Affine.

Ta chứng minh rằng  $M \times S = S \times M$ . Bởi vậy, hai hệ mật là giao hoán. Tuy nhiên, không phải mọi cặp hệ mật đều giao hoán; có thể tìm ta được các cặp phản ví dụ. Một khác ta thấy rằng phép tích luôn kết hợp:

$$(S_1 \times S_2) \times S_3 = S_1 \times (S_2 \times S_3)$$

Nếu lấy tích của một hệ mật tự đồng cấu với chính nó thì ta thu được hệ mật  $S \times S$  (kí hiệu là  $S^2$ ). Nếu lấy tích  $n$  lần thì hệ mật kết quả là  $S^n$ . Ta gọi  $S^n$  là hệ mật lặp.

Một hệ mật  $S$  được gọi là luỹ đẳng nếu  $S^2 = S$ . Có nhiều hệ mật đã nghiên cứu trong chương này là hệ mật luỹ đẳng. Chẳng hạn các hệ MDV, MTT, Affine, Hill, Vigenère và hoán vị đều là luỹ đẳng. Hiển nhiên là nếu hệ mật  $S$  là luỹ đẳng thì không nên sử dụng hệ mật tích  $S^2$  vì nó yêu cầu lượng khoá lớn hơn mà không có độ bảo mật cao hơn.

Nếu một hệ mật không phải là luỹ đẳng thì có thể làm tăng độ mật bằng cách lặp nhiều lần. Ý tưởng này đã được dùng trong chuẩn mã dữ liệu (DES). Trong DES dùng 16 phép lặp, tất nhiên hệ mật ban đầu phải là hệ mật không luỹ đẳng. Một phương pháp có thể xây dựng các hệ mật không luỹ đẳng đơn giản là lấy tích của hai hệ mật đơn giản khác nhau.

### **Nhật xét:**

Có thể dễ dàng chứng tỏ rằng, nếu cả hai hệ mật  $S_1$  và  $S_2$  là luỹ đẳng và giao hoán thì  $S_1$  và  $S_2$  cũng là luỹ đẳng. Điều này rút ra từ các phép toán đại số sau:

$$\begin{aligned} (S_1 \times S_2) \times (S_1 \times S_2) &= S_1 \times (S_2 \times S_1) \times S_2 \\ &= S_1 \times (S_1 \times S_2) \times S_2 \\ &= (S_1 \times S_1) \times (S_2 \times S_2) \\ &= S_1 \times S_2 \end{aligned}$$

(Chú ý: Dùng tính chất kết hợp trong chứng minh trên).

Bởi vậy, nếu cả  $S_1$  và  $S_2$  đều là luỹ đẳng và ta muốn  $S_1 \times S_2$  là không luỹ đẳng thì điều kiện cần là  $S_1$  và  $S_2$  không giao hoán.

Rất may mắn là nhiều hệ mật đơn giản thoả mãn điều kiện trên. Kỹ thuật thường được sử dụng trong thực tế là lấy tích các hệ mã kiểu thay thế và các hệ mã kiểu hoán vị.

## E. CHUẨN MÃ DỮ LIỆU (DES)

### 1. Mở đầu

Ngày 15.5.1973. Uỷ ban tiêu chuẩn quốc gia Mỹ đã công bố một khuyến nghị cho các hệ mật trong Hồ sơ quản lý liên bang. Điều này cuối cùng đã dẫn đến sự phát triển của Chuẩn mã dữ liệu (DES) và nó đã trở thành một hệ mật được sử dụng rộng rãi nhất trên thế giới. DES được IBM phát triển và được xem như một cải biến của hệ mật LUCIPHER. DES được công bố lần đầu tiên trong Hồ sơ Liên bang vào ngày 17.3.1975. Sau nhiều cuộc tranh luận công khai, DES đã được chấp nhận chọn làm chuẩn cho các ứng dụng không được coi là mật vào 5.1.1977. Kể từ đó cứ 5 năm một lần, DES lại được Uỷ ban Tiêu chuẩn Quốc gia xem xét lại. Lần đổi mới gần đây nhất của DES là vào tháng 1.1994 và sau là 1998. Tới tháng 10.2000 DES đã không còn là chuẩn mã dữ liệu nữa.

### 2. Mô tả DES

Mô tả đầy đủ của DES được nêu trong Công bố số 46 về các chuẩn xử lý thông tin Liên bang (Mỹ) vào 15.1.1977. DES mã hoá một xâu bit x của bản rõ độ dài 64 bằng một khoá 56 bit. Bản mã nhận được cũng là một xâu bit có độ dài 64. Trước hết ta mô tả ở mức cao về hệ thống.

Thuật toán tiến hành theo 3 giai đoạn:

1. Với bản rõ cho trước x, một xâu bit  $X_0$  sẽ được xây dựng bằng cách hoán vị các bit của x theo phép hoán vị cố định ban đầu IP. Ta viết:

$$X_0 = IP(x) = L_0 R_0, \text{ trong đó } L_0 \text{ gồm 32 bit đầu và } R_0 \text{ là 32 bit cuối.}$$

2. Sau đó tính toán 16 lần lặp theo một hàm xác định. Ta sẽ tính  $L_i R_i$ ,  $1 \leq i \leq 16$  theo quy tắc sau:

$$\begin{aligned} L_i &= R_{i-1} \\ R_i &= L_{i-1} \oplus f(R_{i-1}, k_i) \end{aligned}$$

trong đó  $\oplus$  kí hiệu phép hoặc loại trừ của hai xâu bit (cộng theo modulo 2).  $f$  là một hàm mà ta sẽ mô tả ở sau, còn  $k_1, k_2, \dots, k_{16}$  là các xâu bit độ dài 48 được tính như hàm của khoá k. (trên thực tế mỗi  $k_i$  là một phép chọn hoán vị bit trong k).

$k_1, k_2, \dots, k_{16}$  sẽ tạo thành bảng khoá. Một vòng của phép mã hoá được mô tả trên hình 2.11



Hình 0-8. Một vòng của DES

3. Áp dụng phép hoán vị ngược  $IP^{-1}$  cho xâu bit  $R_{16}L_{16}$ , ta thu được bản mã y. Tức là  $y = IP^{-1}(R_{16}L_{16})$ . Hãy chú ý thứ tự đã đảo của  $L_{16}$  và  $R_{16}$ .

Hàm f có hai biến vào: biến thứ nhất A là xâu bit độ dài 32, biến thứ hai J là một xâu bit độ dài 48. Đầu ra của f là một xâu bit độ dài 32. Các bước sau được thực hiện:

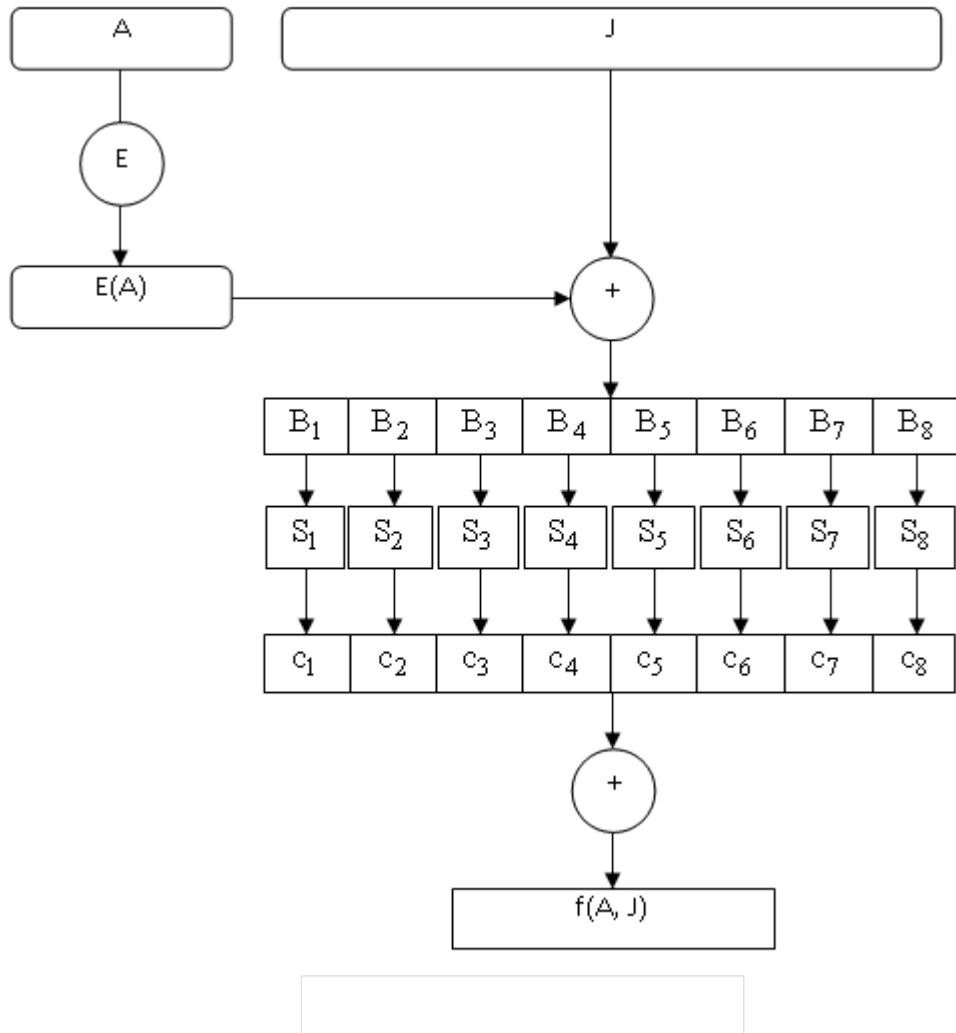
1. Biến thứ nhất A được mở rộng thành một xâu bit độ dài 48 theo một hàm mở rộng cố định E.  $E(A)$  gồm 32 bit của A (được hoán vị theo cách cố định) với 16 bit xuất hiện hai lần.

2. Tính  $E(A) \oplus J$  và viết kết quả thành một chuỗi 8 xâu 6 bit là

$$B_1B_2B_3B_4B_5B_6B_7B_8.$$

3. Bước tiếp theo dùng 8 bảng  $S_1, S_2, \dots, S_8$  (được gọi là các hộp S). Với mỗi  $S_i$  là một bảng  $4 \times 16$  cố định có các hàng là các số nguyên từ 0 đến 15. Với xâu bit có độ dài 6 (kí hiệu  $B_i = b_1b_2b_3b_4b_5b_6$ ), ta tính  $S_j(B_j)$  như sau: hai bit  $b_1b_6$  xác định biểu diễn nhị phân của hàng r của  $S_j$  ( $0 \leq r \leq 3$ ) và bốn bit  $(b_2b_3b_4b_5)$  xác định biểu diễn nhị phân của cột c của  $S_j$  ( $0 \leq c \leq 15$ ). Khi đó,  $S_j(B_j)$  sẽ xác định phần tử  $S_j(r, c)$ ; phần tử này viết dưới dạng nhị phân là một xâu bit có độ dài 4. (Bởi vậy, mỗi  $S_j$  có thể được coi là một hàm mã mà đầu vào là một xâu bit có độ dài 2 và một xâu bit có độ dài 4, còn đầu ra là một xâu bit có độ dài 4). Bằng cách tương tự tính các  $C_j = S_j(B_j)$ ,  $1 \leq j \leq 8$ .

4. Xâu bit  $C = C_1C_2 \dots C_8$  có độ dài 32 được hoán vị theo phép hoán vị cố định P. Xâu kết quả là  $P(C)$  được xác định là  $f(A, J)$ .



Hình 0-9. Hàm f của DES

Hàm f được mô tả trong hình 2.12. Chủ yếu nó gồm một phép thay (sử dụng hộp S), tiếp sau đó là phép hoán vị P. 16 phép lặp của f sẽ tạo nên một hệ mật tích.

Trong phần còn lại của mục này, ta sẽ mô tả hàm cụ thể được dùng trong DES. Phép hoán vị ban đầu IP như sau:

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

Bảng này có nghĩa là bit thứ 58 của x là bit đầu tiên của  $IP(x)$ ; bit thứ 50 của x là bit thứ hai của  $IP(x)$ , .v.v . . .

Phép hoán vi ngược  $IP^{-1}$  là:

IP <sup>-1</sup>								
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	

Hàm mở rộng E được xác định theo bảng sau:

Bảng chọn E bit						
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	

Tám hopped S là:

15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
$S_3$															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
$S_4$															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14
$S_5$															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
$S_6$															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	15	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
$S_7$															
4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
$S_8$															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Và phép hoán vị P có dạng:

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

Cuối cùng, ta cần mô tả việc tính toán bảng khoá từ khoá k. Trên thực tế, k là một xâu bit độ dài 64, trong đó 56 bit là khoá và 8 bit để kiểm tra tính chẵn lẻ nhằm phát hiện sai. Các bit ở các vị trí 8, 16, . . . , 64 được xác định sao cho mỗi byte chứa một số lẻ các số "1". Bởi vậy, một sai sót đơn lẻ có thể phát hiện được trong mỗi nhóm 8 bit. Các bit kiểm tra bị bỏ qua trong quá trình tính bảng khoá.

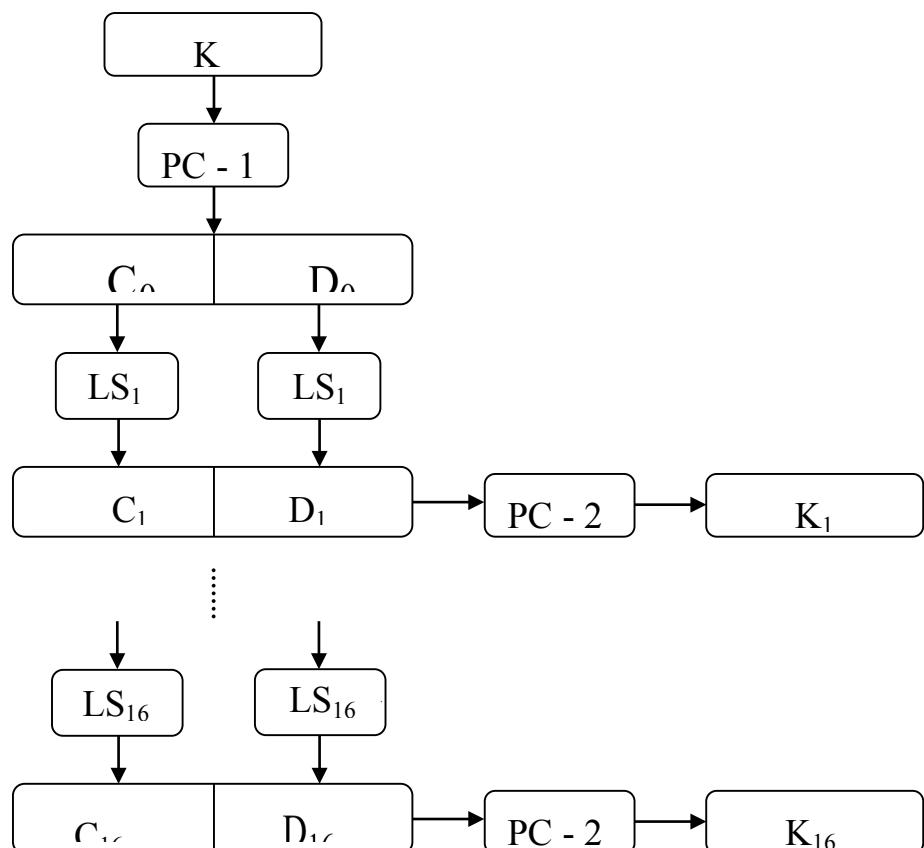
1. Với một khoá k 64 bit cho trước, ta loại bỏ các bit kiểm tra tính chẵn lẻ và hoán vị các bit còn lại của k theo phép hoán vị cố định PC-1. Ta viết:

$$PC - 1(k) = C_0 D_0$$

2. Với i thay đổi từ 1 đến 16:

$$\begin{aligned} C_i &= LS_i(C_{i-1}) \\ D_i &= LS_i(D_{i-1}) \end{aligned}$$

Việc tính bảng khoá được mô tả trên hình 2.13



Hình 0-10. Tính bảng khóa DES

Các hoán vị PC-1 và PC-2 được dùng trong bảng khoá là:

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	

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

Bây giờ ta sẽ  
kết quả. Như đã  
đưa ra bảng khoá  
nói ở trên, mỗi  
vòng sử dụng một khoá 48 bit gồm 48 bit nằm trong K. Các phần tử trong các bảng  
dưới đây biểu thị các bit trong K trong các vòng khoá khác nhau.

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

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

Vòng 3															
51	27	10	36	25	58	9	33	43	50	60	18				

44	11	2	1	49	34	35	42	41	3	59	17
61	4	15	30	13	47	23	6	12	29	62	5
37	28	14	39	54	63	21	53	20	38	31	7

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

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

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

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

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

Vòng 16															
18	59	42	3	57	25	41	36	10	17	27	50				
11	43	34	33	52	1	2	9	44	35	26	49				
30	5	47	62	45	12	55	38	13	61	31	37				
6	29	46	4	23	28	53	22	21	7	63	39				

Phép giải mã được thực hiện nhờ dùng cùng thuật toán như phép mã nếu đầu vào là y nhưng dùng bảng khoá theo thứ tự ngược lại K<sub>16</sub>,...,K<sub>1</sub>. Đầu ra của thuật toán sẽ là bản rõ x.

### 3. Một ví dụ về DES

Sau đây là một ví dụ về phép mã DES. Giả sử ta mã bản rõ (ở dạng mã hexa- hệ đếm 16):

0 1 2 3 4 5 6 7 8 9 A B C D E F

## Bằng cách dùng khoá

1 2 3 4 5 7 7 9 9 B B C D F F 1

Khoá ở dạng nhị phân (không chứa các bit kiểm tra) là:

00010010011010010101101111001001101101111011011111111000

Sử dụng IP, ta thu được  $L_0$  và  $R_0$  (ở dạng nhị phân) như sau:

$$L_0 = 1100110000000000110011001111111$$

$$L_1 = R_0 = 11110000101010101111000010101010$$

Sau đó thực hiện 16 vòng của phép mã như sau:

$$E(R_0) = 0111101000010101010101110100001010101010101$$

$$K_1 = 0001101100000101110111111100011100001110010$$

$$E(R_0) \oplus K_1 = 0110000100010111011101000011001100100100111$$

$$\text{S-box outputs } 01011100100000101011010110010111$$

$$f(R_0, K_1) = 001000110100101010100110111011$$

$$L_2 = R_1 = 11101111010010100110010101000100$$

$$E(R_1) = 0111010111010100101010000110000101010100001001$$

$$K_2 = 0111100110101110110110011101110100111100101$$

$$E(R_1) \oplus K_2 = 000011000100010010001101110101101100111101100$$

$$\text{S-box outputs } 11111000110100000011101010101110$$

$$f(R_1, K_2) = 0011110010101011000011110100011$$

$$L_3 = R_2 = 11001100000000010111011100001001$$

$$E(R_2) = 111001011000000000000010101110101110100001010011$$

$$K_3 = 010101011111001000101001000010110011110011001$$

$$E(R_2) \oplus K_3 = 1011000001111100100010001111000001001111001010$$

$$\text{S-box outputs } 00100111000100001110000101101111$$

$$f(R_2, K_3) = 01001101000101100110111010110000$$

$$L_4 = R_3 = 101000100101110000010111110100$$

$$E(R_3) = 010100000100001011110000000010101111111010100$$

$$K_4 = 0111001010101101110101101101100110100011101$$

$$E(R_3) \oplus K_4 = 00100010111011100101110110111001001010110100$$

$$\text{S-box outputs } 00100001111011011001111100111010$$

$$f(R_3, K_4) = 10111011001000110111011101001100$$

$$L_5 = R_4 = 0111011100100010000000001000101$$

$$E(R_4) = 10111010111010010000100000000000000001000001010$$

$$K_5 = 0111110011101100000011111010110101001110101000$$

$$E(R_4) \oplus K_5 = 1100011000001010000011111010110101000110100010$$

$$\text{S-box outputs } 01010000110010000011000111101011$$

$$f(R_4, K_5) = 00101000000100111010110111000011$$

$$L_6 = R_5 = 1000101001001111010011000110111$$

$$E(R_5) = 1100010101000010010111110100001100000110101111$$

$$K_6 = 0110001110100101001111001010000011101100101111$$

$$E(R_5) \oplus K_6 = 101001101110011101100001100000001011101010000000$$

S-box outputs 01000001111100110100110000111101

$$f(R_5, K_6) = 10011110010001011100110100101100$$

$$L_7 = R_6 = 11101001011001111100110101101001$$

$$E(R_6) = 111101010010101100001111110010110101101010011$$

$$K_7 = 11101100100001001011011111101100001100010111100$$

$$E(R_6) \oplus K_7 = 00011001101011110111000000100111011001111101111$$

S-box outputs 00010000011101010100000010101101

$$f(R_6, K_7) = 10001100000001010001110000100111$$

$$L_8 = R_7 = 00000110010010101011101000010000$$

$$E(R_7) = 000000001100001001010101011110100000010100000$$

$$K_8 = 11110111100010100011101011000001001110111111011$$

$$E(R_7) \oplus K_8 = 111101110100100001101111100111100111101101011011$$

S-box outputs 01101100000110000111110010101110

$$f(R_7, K_8) = 0011110000001101000011011111001$$

$$L_9 = R_8 = 11010101011010010100101110010000$$

$$E(R_8) = 011010101010110101001010100101011110010100001$$

$$K_9 = 11100000110110111110101111011011110011110000001$$

$$E(R_8) \oplus K_9 = 100010100111000010111001010010001001101100100000$$

S-box outputs 0001000100001100010101110111011

$$f(R_8, K_9) = 0010001000110110011111000110101$$

$$L_{10} = R_9 = 00100100011111001100011001111010$$

$$E(R_9) = 000100001000001111111001011000001100001111110100$$

$$K_{10} = 10110001111100110100011101110100100011001001111$$

$$E(R_9) \oplus K_{10} = 101000010111000010111110110110101000010110111011$$

S-box outputs 11011010000001000101001001110101

$$f(R_9, K_{10}) = 0110001010111001001110000100010$$

$$L_{11} = R_{10} = 10110111110101011101011110110010$$

$$E(R_{10}) = 0101101011111101010101111010101111110110100101$$

$$K_{11} = 00100001010111111010011110111101101001110000110$$

$$E(R_{10}) \oplus K_{11} = 011110111010000101111000001101000010111000100011$$

S-box outputs 01110011000001011101000100000001

$$f(R_{10}, K_{11}) = 11100001000001001111101000000010$$

$L_{12} = R_{11} = 1100010101110000011110001111000$
$E(R_{11}) = 01100000101010111100000001111100000111110001$
$K_{12} = 0111010101110001111010110010100011001111101001$
$E(R_{11}) \oplus K_{12} = 00010101110110100000010110001011110010000011000$
S-box outputs 01110011000001011101000100000001
$f(R_{11}, K_{12}) = 1100001001101000110011111101010$
$L_{13} = R_{12} = 01110101101111010001100001011000$

$E(R_{12}) = 001110101011101111101010001110000001011110000$
$K_{13} = 1001011110001011101000111110101011101001000001$
$E(R_{12}) \oplus K_{13} = 10101101011100000101011011101011011100010110001$
Sbox outputs 10011010110100011000101101001111
$f(R_{12}, K_{13}) = 11011101101110110010100100100010$
$L_{14} = R_{13} = 00011000110000110001010101011010$

$E(R_{13}) = 0000111100010110000001101000101010101011110100$
$K_{13} = 01011110100001101101111100101110011100111010$
$E(R_{13}) \oplus K_{14} = 010100000101010110110001011110000100110111001110$
S-box outputs 01100100011110011001101011110001
$f(R_{13}, K_{14}) = 10110111001100011000111001010101$
$L_{15} = R_{14} = 11000010100011001001011000001101$

$E(R_{14}) = 1110000001010100010110010100101100000001011011$
$K_{15} = 1011111100100011000110100111101001111100001010$
$E(R_{14}) \oplus K_{15} = 010111111000101110101000110111111111101010001$
S-box outputs 10110010111010001000110100111100
$f(R_{14}, K_{15}) = 01011011100000010010011101101110$
$R_{15} = 01000011010000100011001000110100$

$E(R_{15}) = 00100000011010100000100000110100100000110101000$
$K_{16} = 11001011001111011000101100001110000101111110101$
$E(R_{15}) \oplus K_{16} = 11101011010101110001111000101000101011001011101$
S-box outputs 10100111100000110010010000101001
$f(R_{15}, K_{16}) = 1100100011000000010011110011000$
$R_{16} = 00001010010011001101100110010101$

Cuối cùng, áp dụng IP<sup>-1</sup> vào L<sub>16</sub>, R<sub>16</sub> ta nhận được bản mã hexa là:

8 5 E 8 1 3 5 4 0 F 0 A B 4 0 5

#### 4. Một số ý kiến thảo luận về DES

Khi DES được đề xuất như một chuẩn mật mã, đã có rất nhiều ý kiến phê phán. Một lý do phản đối DES có liên quan đến các hộp S. Mọi tính toán liên quan đến DES ngoại trừ các hộp S đều tuyến tính, tức 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 hộp S - chứa đựng thành phần phi tuyến của hệ mật là yếu tố quan trọng nhất đối với độ mật của hệ thống (Ta đã thấy là các hệ mật tuyến tính - chẳng hạn như Hill - có thể dễ dàng bị mã thám khi bị tấn công bằng bản rõ đã biết). Tuy nhiên, tiêu chuẩn xây dựng các hộp S không được biết đầy đủ. Một số người đã gợi ý là các hộp S phải chứa các "cửa sập" được dấu kín, cho phép Cục An ninh Quốc gia Mỹ (NSA) giải mã được các thông báo nhưng vẫn giữ được mức độ an toàn của DES. Dĩ nhiên ta không thể bác bỏ được khẳng định này, tuy nhiên không có một chứng cứ nào được đưa ra để chứng tỏ rằng trong thực tế có các cửa sập như vậy.

Năm 1976 NSA đã khẳng định rằng, các tính chất sau của hộp S là tiêu chuẩn thiết kế:

- Mỗi hàng trong mỗi hộp S là một hoán vị của các số nguyên  $0, 1, \dots, 15$ .
- Không một hộp S nào là một hàm Affine hoặc tuyến tính các đầu vào của nó.
- Việc thay đổi một bit vào của S phải tạo nên sự thay đổi ít nhất là hai bit ra.
- Đôi với hộp S bất kì và với đầu vào  $x$  bất kì  $S(x)$  và  $S(x \oplus 001100)$  phải khác nhau tối thiểu là hai bit (trong đó  $x$  là xâu bit độ dài 6).

Hai tính chất khác nhau sau đây của các hộp S có thể coi là được rút ra từ tiêu chuẩn thiết kế của NSA.

- Với hộp S bất kì, đầu vào  $x$  bất kì và với  $e, f \in \{0, 1\}$ :  $S(x) \neq S(x \oplus 11ef00)$ .
- Với hộp S bất kì, nếu cố định một bit vào và xem xét giá trị của một bit đầu ra cố định thì các mẫu vào để bit ra này bằng 0 sẽ xấp xỉ bằng số mẫu ra để bit đó bằng 1. (Chú ý rằng, nếu cố định giá trị bit vào thứ nhất hoặc bit vào thứ 6 thì có 16 mẫu vào làm cho một bit ra cụ thể bằng 0 và có 16 mẫu vào làm cho bit này bằng 1. Với các bit vào từ bit thứ hai đến bit thứ 5 thì điều này không còn đúng nữa. Tuy nhiên, phân bố kết quả vẫn gần với phân bố đều. Chính xác hơn, với một hộp S bất kì, nếu ta cố định giá trị của một bit vào bất kì thì số mẫu vào làm cho một bit ra cố định nào đó có giá trị 0 (hoặc 1) luôn nằm trong khoảng từ 13 đến 19).

Người ta không biết rõ là liệu có còn một chuẩn thiết kế nào đầy đủ hơn được dùng trong việc xây dựng hộp S hay không.

Sự phản đối xác đáng nhất về DES chính là kích thước của không gian khoá:  $2^{56}$  là quá nhỏ để đảm bảo an toàn thực sự. Nhiều thiết bị chuyên dụng đã được đề xuất nhằm phục vụ cho việc tấn công với bản rõ đã biết. Phép tấn công này chủ yếu thực hiện tìm khoá theo phương pháp vét cạn. Tức với bản rõ  $x$  64 bit và bản mã y tương ứng, mỗi khoá

đều có thể được kiểm tra cho tới khi tìm được một khoá k thoả mãn  $e_k(x) = y$ . (Cần chú ý là có thể có nhiều hơn một khoá k như vậy).

Ngay từ năm 1977, Diffie và Hellman đã gợi ý rằng có thể xây dựng một chip VLSI (mạch tích hợp mật độ lớn) có khả năng kiểm tra được  $10^6$  khoá/giây. Một máy có thể tìm toàn bộ không gian khoá cỡ  $10^6$  trong khoảng 1 ngày. Họ ước tính chi phí để tạo một máy như vậy khoảng  $2.10^7$ \$.

Trong cuộc hội thảo tại hội nghị CRYPTO'93, Michael Wiener đã đưa ra một thiết kế rất cụ thể về máy tìm khoá. Máy này xây dựng trên một chip tìm khoá, có khả năng thực hiện đồng thời 16 phép mã và tốc độ tới  $5 \times 10^7$  khoá/giây. Với công nghệ hiện nay, chi phí chế tạo khoảng 10,5\$/chip. Giá của một khung máy chứa 5760 chip vào khoảng 100.000\$ và như vậy nó có khả năng tìm ra một khoá của DES trong khoảng 1,5 ngày. Một thiết bị dùng 10 khung máy như vậy có giá chừng  $10^6$  \$ sẽ giảm thời gian tìm kiếm khoá trung bình xuống còn 3,5 giờ.

Mặc dù việc mô tả DES khá dài dòng song người ta có thể thực hiện DES rất hữu hiệu bằng cả phần cứng lẫn phần mềm. Các phép toán duy nhất cần được thực hiện là phép hoặc loại trừ các xâu bit. Hàm mở rộng E, các hộp S, các hoán vị IP và P và việc tính toán các giá trị  $K_1, \dots, K_{16}$  đều có thể thực hiện được cùng lúc bằng tra bảng (trong phần mềm) hoặc bằng cách nối cứng chúng thành một mạch.

Các ứng dụng phần cứng hiện thời có thể đạt được tốc độ mã hoá cực nhanh. Công ty Digital Equipment đã thông báo tại hội nghị CRYPTO'92 rằng họ đã chế tạo một chip có 50 ngàn tranzistor có thể mã hoá với tốc độ 1 Gbit/s bằng cách dùng nhịp có tốc độ 250MHz. Giá của chip này vào khoảng 300\$. Tới năm 1991 đã có 45 ứng dụng phần cứng và chương trình cơ sở của DES được Uỷ ban tiêu Chuẩn quốc gia Mỹ (NBS) chấp thuận.

Một ứng dụng quan trọng của DES là trong giao dịch ngân hàng Mỹ - (ABA) DES được dùng để mã hoá các số định danh cá nhân (PIN) và việc chuyển tài khoản bằng máy chủ quỹ tự động (ATM). DES cũng được Hệ thống chi trả giữa các nhà băng của Ngân hàng hối đoái (CHIPS) dùng để xác thực các giao dịch vào khoảng trên  $1,5 \times 10^{12}$  USA/tuần. DES còn được sử dụng rộng rãi trong các tổ chức chính phủ. Chẳng hạn như Bộ năng lượng, Bộ Tư pháp và Hệ thống dự trữ liên bang.

### **5. Các tính chất và sức mạnh của DES:**

DES có một số tính chất dễ nhận thấy và đồng thời chúng ta cũng sẽ sơ bộ đánh giá độ an toàn của DES thông qua các tấn công mạnh nhất hiện nay.

- *Tính chất bù:*

Kí hiệu phép mã hóa DES là  $E$ , và  $x^*$  là phần bù của  $x$ . Khi đó ta có: nếu  $y = E_k(x)$  thì  $y^* = E_k(x^*)$

- *Các khóa yếu và khóa nửa yếu:*

Định nghĩa: Một khóa yếu của DES là khóa  $K$  sao cho  $E_K(E_K(x)) = x$  với mọi  $x$ . Một cặp khóa nửa yếu của DES là cặp  $(K_1, K_2)$  sao cho  $E_{K_1}(E_{K_2}(x)) = x$  với mọi  $x$ .

DES có 4 khóa yếu và 6 cặp khóa nửa yếu

- *Các điểm bất động*

Với mỗi khóa yếu của DES sẽ có tương ứng  $2^{32}$  điểm bất động, tức là  $x$  thỏa mãn  $E_K(x) = x$ .

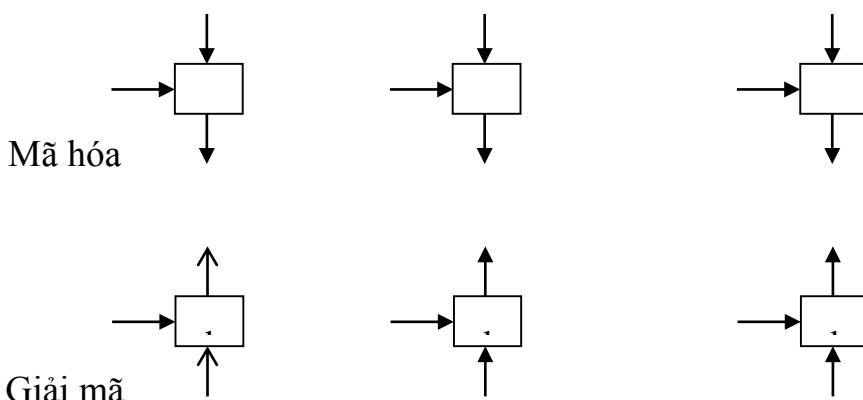
Có 4 trong 12 khóa nửa yếu của DES mỗi cái sẽ có  $2^{32}$  điểm phản bất động, tức là  $x$  sao cho  $E_K(x) = x^*$ .

- *DES không phải là một nhóm dưới phép hợp hàm*

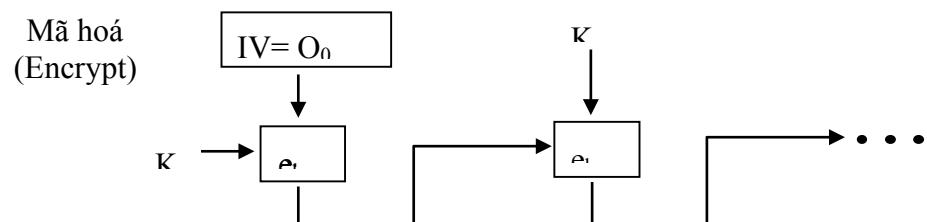
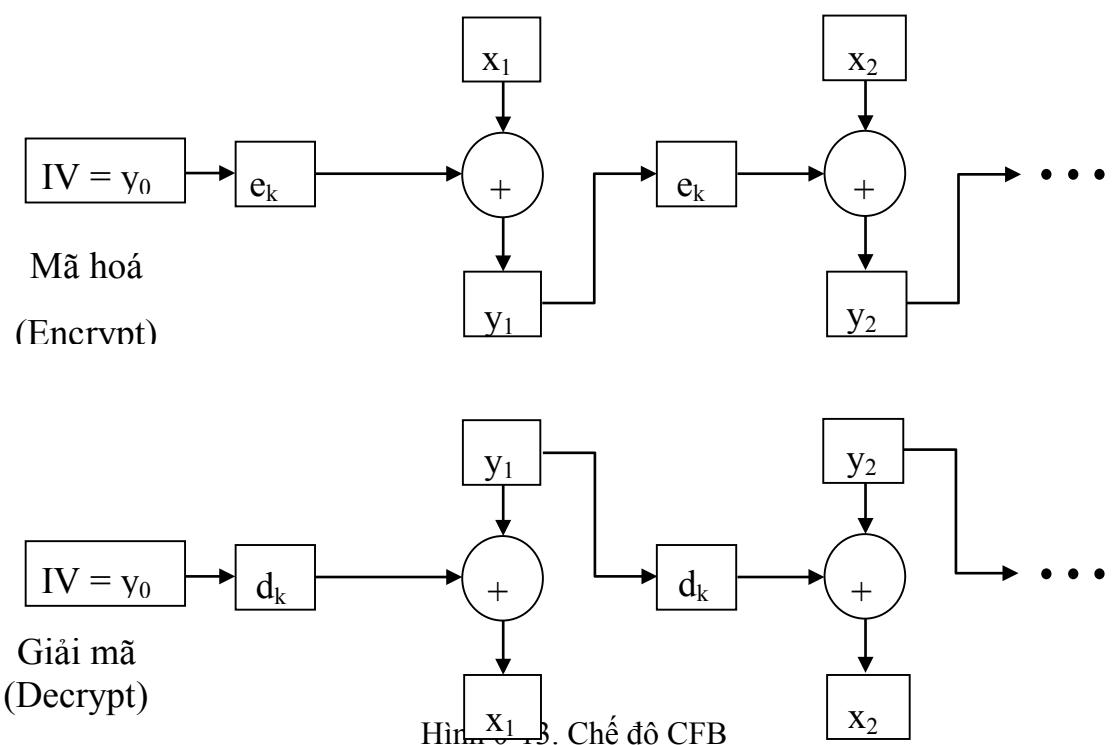
## 6. Các chế độ hoạt động của DES

Có 4 chế độ làm việc đã được phát triển cho DES: Chế độ quyển mã điện tử (ECB), chế độ phản hồi mã (CFB), chế độ liên kết khối mã (CBC) và chế độ phản hồi đầu ra (OFB). Chế độ ECB tương ứng với cách dùng thông thường của mã khối: với một dãy các khối bǎn rõ cho trước  $X_1, X_2, \dots$  (mỗi khối có 64 bit), mỗi  $X_i$  sẽ được mã hoá bằng cùng một khoá  $k$  để tạo thành một chuỗi các khối bǎn mã  $y_1, y_2, \dots$  theo quy tắc  $y_i = e_k(y_{i-1} \oplus X_i)$ ,  $i \geq 1$ . Việc sử dụng chế độ CBC được mô tả trên hình 2.15.

Trong các chế độ OFB và CFB dòng khoá được tạo ra sẽ được cộng mod 2 với bǎn rõ. OFB thực sự là một hệ mã dòng đồng bộ: dòng khoá được tạo bởi việc mã lặp vector khởi tạo 64 bit (vector IV). Ta xác định  $z_0 = IV$  và rồi tính dòng khoá  $Z_1, Z_2, \dots$  theo quy tắc  $z_i = e_k(z_{i-1})$ ,  $i \geq 1$ . Dãy bǎn rõ  $X_1, X_2, \dots$  sau đó sẽ được mã hoá bằng cách tính  $y_i = X_i \oplus z_i$ ,  $i \geq 1$ .



Hình 0-12. Chế độ CBC



C.

Hình 0-14. Chế độ OFB

Trong chế độ CFB, ta bắt đầu với  $y_0 = IV$  (là một vector khởi tạo 64 bit) và tạo phần tử  $z_i$  của dòng khoá bằng cách mã hoá khối bản mã trước đó. Tức  $z_i = e_k(y_{i-1})$ ,  $i \geq 1$ . Cũng như trong chế độ OFB:  $y_i = x_i \oplus z_i$ ,  $i \geq 1$ . Việc sử dụng CFB được mô tả trên hình 2.16 (chú ý rằng hàm mã DES  $e_k$  được dùng cho cả phép mã và phép giải mã ở các chế độ CFB và OFB).

Cũng còn một số biến tấu của OFB và CFB được gọi là các chế độ phản hồi k bit ( $1 < k < 64$ ). Ở đây, ta đã mô tả các chế độ phản hồi 64 bit. Các chế độ phản hồi 1 bit và 8 bit thường được dùng trong thực tế cho phép mã hoá đồng thời 1 bit (hoặc byte) số liệu.

Bốn chế độ công tác có những ưu, nhược điểm khác nhau. Ở chế độ ECB và OFB, sự thay đổi của một khối bản rõ  $x_i$  64 bit sẽ làm thay đổi khối bản mã  $y_i$  tương ứng, nhưng các khối bản mã khác không bị ảnh hưởng. Trong một số tình huống, đây là một tính chất đáng mong muốn. Ví dụ, chế độ OFB thường được dùng để mã khi truyền vệ tinh.

Mặt khác ở các chế độ CBC và CFB, nếu một khối bản rõ  $x_i$  bị thay đổi thì  $y_i$  và tất cả các khối bản mã tiếp theo sẽ bị ảnh hưởng. Như vậy các chế độ CBC và CFB có thể được sử dụng rất hiệu quả cho mục đích xác thực. Đặc biệt hơn, các chế độ này có thể được dùng để tạo mã xác thực bản tin ( MAC - message authentication code). MAC

được gắn thêm vào các khối bản rõ để thuyết phục Bob tin rằng, dãy bản rõ đó thực sự là của Alice mà không bị Oscar giả mạo. Như vậy MAC đảm bảo tính toàn vẹn (hay tính xác thực) của một bản tin (nhưng tất nhiên là MAC không đảm bảo độ mật).

Ta sẽ mô tả cách sử dụng chế độ CBC để tạo ra một MAC. Ta bắt đầu bằng vector khởi tạo IV chứa toàn số 0. Sau đó dùng chế độ CBC để tạo các khối bản mã  $y_1, \dots, y_n$  theo khoá K. Cuối cùng ta xác định MAC là  $y_n$ . Alice sẽ phát đi dãy các khối bản rõ  $x_1, \dots, x_n$  cùng với MAC. Khi Bob thu được  $x_1 \dots x_n$  anh ta sẽ khôi phục lại  $y_1, \dots, y_n$  bằng khoá K bí mật và xác minh xem liệu  $y_n$  có giống với MAC mà mình đã thu được hay không?

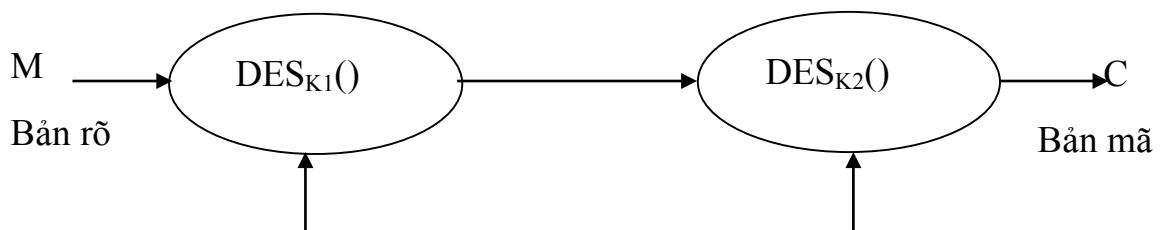
Nhận thấy Oscar không thể tạo ra một MAC hợp lệ do anh ta không biết khoá K mà Alice và Bob đang dùng. Hơn nữa Oscar thu chẵn được dãy khối bản rõ  $x_1, \dots, x_n$  và thay đổi ít nhiều nội dung thì chắc chắn là Oscar không thể thay đổi MAC để được Bob chấp nhận.

Thông thường ta muốn kết hợp cả tính xác thực lẫn độ bảo mật. Điều đó có thể thực hiện như sau: Trước tiên Alice dùng khoá  $K_1$  để tạo MAC cho  $x_1, \dots, x_n$ . Sau đó Alice xác định  $x_{n+1}$  là MAC rồi mã hoá dãy  $x_1, \dots, x_{n+1}$  bằng khoá thứ hai  $K_2$  để tạo ra bản mã  $y_1, \dots, y_{n+1}$ . Khi Bob thu được  $y_1, \dots, y_{n+1}$ , trước tiên Bob sẽ giải mã (bằng  $K_2$ ) và kiểm tra xem  $x_{n+1}$  có phải là MAC đối với dãy  $x_1, \dots, x_n$  dùng  $K_1$  hay không.

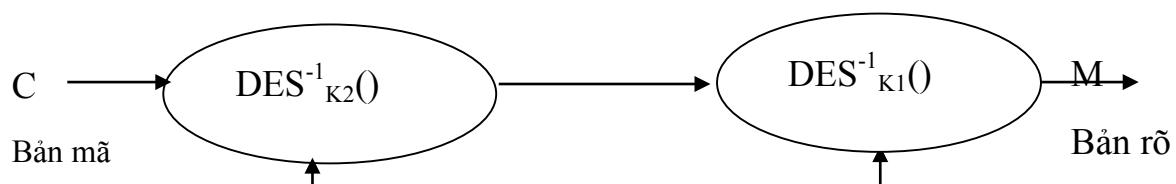
Ngược lại, Alice có thể dùng  $K_1$  để mã hoá  $x_1, \dots, x_n$  và tạo ra được  $y_1, \dots, y_n$ , sau đó dùng  $K_2$  để tạo MAC  $y_{n+1}$  đối với dãy  $y_1, \dots, y_n$ . Bob sẽ dùng  $K_2$  để xác minh MAC và dùng  $K_1$  để giải mã  $y_1, \dots, y_n$ .

## 7. Một số biến thể của DES

- DES bội hai (Double DES)



a. Mã hóa DES bội hai



Hình 0-15. Des bội hai

Mã hóa:  $C = DES_{K_2} [DES_{K_1}(M)]$

Giải mã:  $M = DES_{K_1}^{-1} [DES_{K_2}^{-1}(C)]$

Mặc dù có  $2^{56}$  sự lựa chọn cho khóa  $K_1$  và  $2^{56}$  sự lựa chọn đối với khóa  $K_2$ .

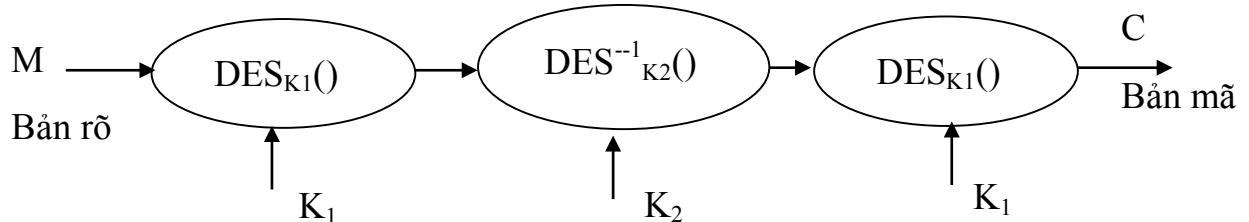
Điều này dẫn tới có  $2^{112}$  sự lựa chọn cho cặp khóa  $(K_1, K_2)$  nhưng sức mạnh của DES bội hai không lớn tới mức như vậy.

- *DES bội ba (Triple DES – TDES)*

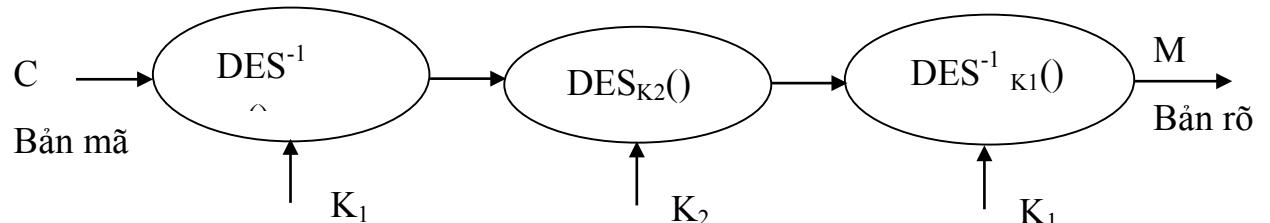
DES bội hai có thể bị tấn công bằng cách thám mã từ hai phía theo đề xuất của Diffie – Hellman. Để khắc phục yếu điểm này người ta đã xây dựng TDES với hai khóa  $K_1$  và  $K_2$  như sau:

Mã hóa:  $C = DES_{K_1} \{DES_{K_2}^{-1} [DES_{K_1}(M)]\}$

Giải mã:  $M = DES_{K_1}^{-1} \{DES_{K_2} [DES_{K_1}^{-1}(C)]\}$



a. Mã hóa TDES với hai khóa



b. Giải mã TDES với hai khóa

Hình 0-16. Mã hóa và giải mã TDES với hai khóa

Với TDES việc tìm kiếm vét cạn yêu cầu khoảng  $2^{112} = 5,1923 \cdot 10^{33}$  phép tính TDES, bởi vậy trên thực tế khó có thể thám mã thành công.

*DES với các khóa con độc lập*

Có thể sử dụng DES với 16 khóa con độc lập để tăng độ mật. Nếu 16 véctơ 48 bit được dùng cho các vòng mã hóa của DES thì người ta phải tạo một khóa k có độ dài 768 bit. Cách tấn công tìm kiếm vét cạn yêu cầu tìm kiếm trong không gian khóa có kích thước  $2^{768}$ . Cách tấn công từ hai phía có thể giảm không gian tìm kiếm xuống  $2^{384}$ , giá trị này vẫn còn rất lớn trong thực tế. Tuy nhiên bằng cách sử dụng thám mã vi sai hệ mật này có thể bị phá với  $2^{61}$  bản rõ được chọn.

### *DES tổng quát (Generalize DES – GDES)*

Vào năm 1981 Johanmuller – Bilch đã đưa ra GDES nhằm tăng tốc độ mã hóa. Thuật toán GDES được mô tả trên hình 2.18

Thay cho việc sử dụng các khối thông báo 64 bit trong DES, GDES chia thông báo thành q khối 32 bit. Giả sử m là thông báo được dùng để mã hóa bằng GDES.

Trong đó  $M_i = m_{i1}, m_{i2}, \dots, m_{i32}$

Ở vòng lặp đầu tiên GDES sẽ mã hóa khối con 32 bit cuối cùng:

$$B_0^{(q)} = M_q = m_{q1}, m_{q2}, \dots, m_{q32}$$

bằng 1 khóa con 48 bit  $K_1$

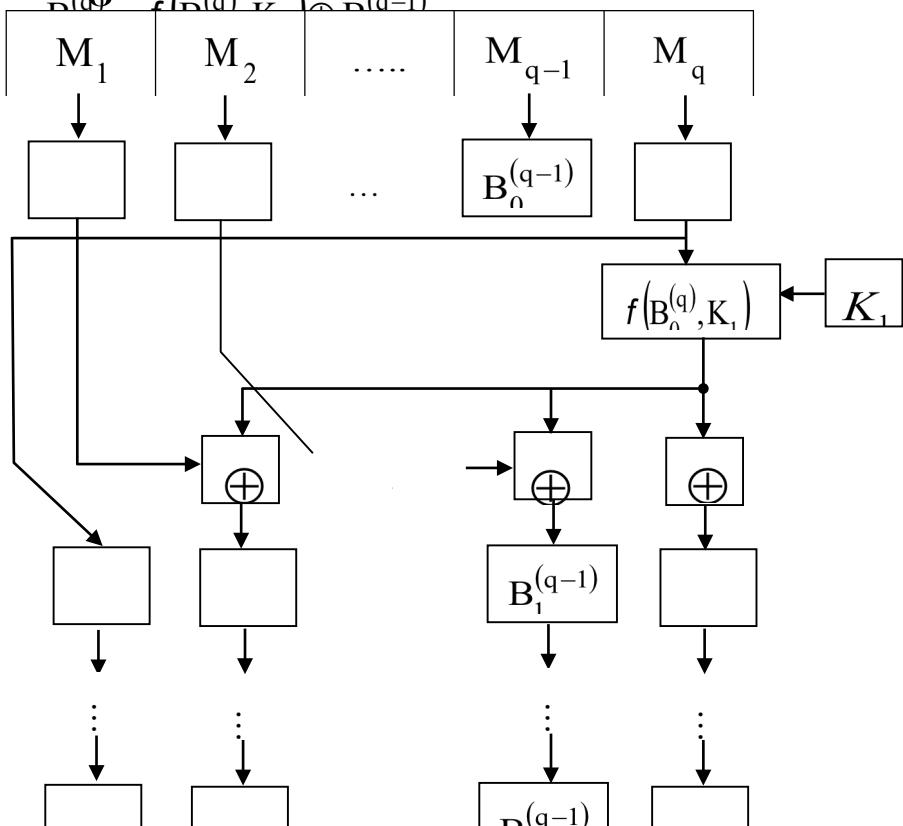
$$f(B_0^{(q)}, K_1) = \{S[K_1 \oplus E(B_0^{(q)})]\}$$

Trong đó  $S[K_1 \oplus E(B_0^{(q)})]$  biểu thị phép thay thế trên véctơ 48 bit  $K_1 \oplus E(B_0^{(q)})$ .

Vectơ 32 bit kết quả  $f(B_0^{(q)}, K_1) \oplus B_0^{(q)}$  sau đó được cộng mod 2 theo từng bít với các nội dung của  $(q-1)$  thanh ghi  $B_1^{(q)}$  bit còn lại.

⋮

Các nội dung trước đó của thanh ghi  $B_1^{(q)}$  sẽ được lưu vào thanh ghi tận cùng bên trái  $B_1^{(1)} = B_1^{(q)} f(B_0^{(q)}, K_1) \oplus B_0^{(q)}$



Hình 0-17. Thuật toán mã hóa GDES

## 8. Thám mã vi sai và thám mã tuyến tính

Phương pháp thám mã truyền thống đối với các mật mã khối (chẳng hạn DES) với bản rõ đã biết là tìm kiếm, vét cạn trên toàn bộ không gian khóa. Tuy nhiên phương pháp tấn công tổng lực này không thể áp dụng được với DES bội đôi và DES bội ba. Các phương pháp tấn công tinh tế hơn đã được đề xuất trong những năm gần đây nhằm làm giảm độ phức tạp tính toán cho thám mã. Sau đây là 2 phương pháp quan trọng nhất.

*Thám mã vi sai (thám mã dựa trên sự khác biệt)*

Thám mã vi sai được đề xuất từ 1990 để thám các mật mã khối như PES, LUCIFER ... Thám mã vi sai xoay quanh việc phân tích phân bố của sự khác biệt (cộng mod 2 theo từng bít) giữa hai bản rõ  $X_1$  và  $X_2$  và hai bản mã  $Y_1$  và  $Y_2$ . Ở đây các bản rõ  $X_1$  và  $X_2$  là các nội dung 32 bít của thanh ghi dịch phải trước phép hoán vị mở rộng  $E(X)$  trong 1 vòng DES. Hai bản mã  $Y_1$  và  $Y_2$  đầu ra 32 bít từ phép hoán vị  $P\circ$  sau các hộp thay thế. Giả sử  $\Delta X$  là hiệu của hai bản rõ đã biết  $X_1$  và  $X_2$  :

$$\Delta X = X_1 \oplus X_2$$

Ở đây  $X_1 \oplus X_2$  biểu thị phép cộng mod 2 theo từng bít của hai véctơ bản rõ. Trong cách tấn công bản rõ có lựa chọn, hai bản rõ  $X_1$  và  $X_2$  được chọn sao cho có  $\Delta X$  mong muốn. Vì  $\Delta X = X_1 \oplus X_2$  và  $A = E(X)$  đơn giản là một phép hoán vị mở rộng của các bít của bản rõ  $A = E(X)$  nên ta cũng biết được  $\Delta A$ .

$$\Delta A = E(X_1) + E(X_2)$$

Ở mỗi vòng của DES, khóa gọn 48 bít  $K_i$  được cộng vào véctơ A 48 bít ở đầu ra của hộp hoán vị mở rộng:

$$B_i = A_i \oplus K_i$$

Vì  $K_i$  là chưa biết nên  $B_1$  và  $B_2$  cũng chưa biết. Tuy nhiên ta lại biết được hiệu của chúng:

$$\Delta B = B_1 \oplus B_2$$

$$\frac{\Delta B}{\Delta A} = \frac{(A_1 + K_i) \oplus (A_2 + K_i)}{\Delta A}$$

Bởi vậy bằng cách chọn  $\Delta X = E(\Delta X)$  và  $X_2$  (tương ứng là  $\Delta X$ ) ta có thể tìm được các đầu vào của 8 hộp thay thế ngay cả khi không biết khóa con.

Từ các bản mã đã biết  $Y_1$  và  $Y_2$  thu được từ việc mã hóa các bản rõ  $X_1$  và  $X_2$  ta cũng xác định được hiệu  $\Delta Y$  của chúng:

$$\Delta Y = Y_1 \oplus Y_2$$

Cả hai véctơ  $Y_1$  và  $Y_2$  đều là các hoán vị của các đầu ra 32 bít  $C_1$  và  $C_2$  của các hộp thay thế.

$$Y_1 = P(C_1)$$

Ta có thể biểu thị các đầu ra  $C_1$  và  $C_2$  của các hộp thay thế như các hàm của  $Y_1$  và  $Y_2$ :

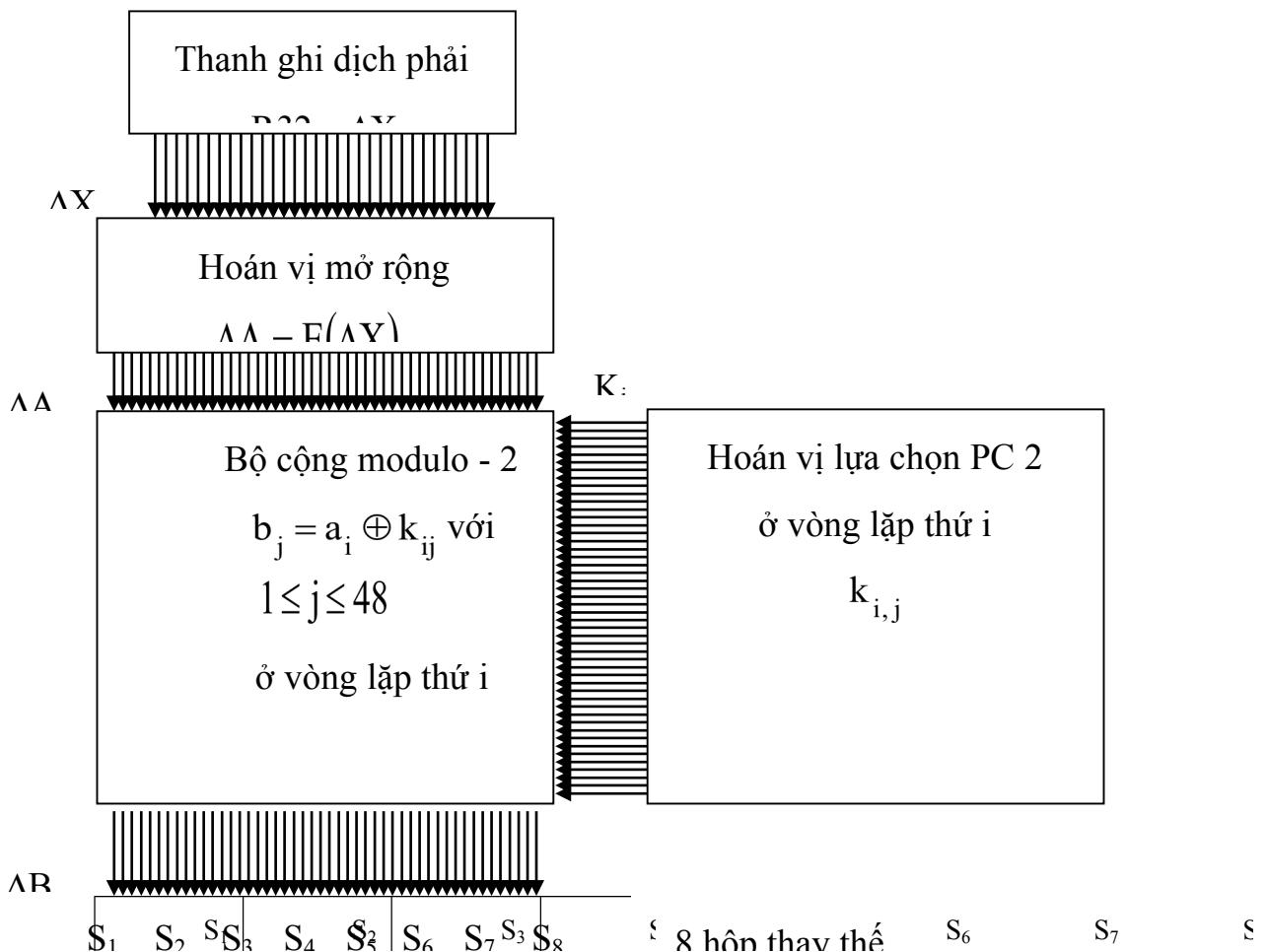
$$C_1 = P^{-1}(Y_1)$$

$$C_2 = P^{-1}(Y_2)$$

Như vậy sự khác biệt ở đầu ra của các hộp thay thế  $\Delta C$  là:

$$\Delta C = (P^{-1}(Y_1)) \oplus (P^{-1}(Y_2))$$

Thám mã vi sai sẽ so sánh phân bố của  $\Delta X$  đối với các cặp bản rõ  $X_1$  và  $X_2$  với phân bố của  $\Delta Y$  đối với các cặp bản mã  $Y_1$  và  $Y_2$  tương ứng.



Trong cách tấn công với cặp rõ – mã được chọn, bản rõ được chọn sao cho tạo được  $\Delta X$  mong muốn. Có một thực tế là các sai khác của bản rõ  $\Delta X$  và các sai khác của bản mã  $\Delta Y$  là không như nhau. Một số sai khác trong các cặp bản rõ có xác suất gây nên sự khác biệt trong các cặp bản mã lớn hơn.

Với mỗi bộ 8 hộp thay thế của DES ta có thể tạo nên một bảng cho mối quan hệ giữa  $\Delta X$  và  $\Delta Y$  (Xem bảng 2.1)

$p_{ij}$  trong bảng biểu thị số các trường hợp mà  $\Delta X_i$  tạo nên  $\Delta Y_j$

	$\Delta Y_1$	...	$\Delta Y_j$	...
$\Delta X_1$	$p_{11}$	...	$p_{1j}$	...
:	:		:	...
$\Delta X_i$	$p_{i1}$	...	$p_{ij}$	...
:	:	...	:	...

Biham và Shamin đã trình diễn một thám mã DES16 vòng dùng  $2^{47}$  cặp rõ – mã được chọn hoặc  $2^{55}$  cặp rõ – mã đã biết với  $2^{37}$  phép toán DES. Điều này chứng tỏ rằng các thám mã này với DES cũng chưa được hiệu quả.

### *Thám mã tuyến tính (TMTT)*

Ý tưởng cơ bản của phương pháp này là cố gắng biểu thị (xấp xỉ) một vòng của DES bằng một phép biến đổi tuyến tính. Hình 2.17 biểu thị cách mà thám mã tuyến tính có thể dùng trên một vòng của DES.

Trong cách tấn công với bản rõ đã biết ta biết được bản rõ M và bản mã C tương ứng. Vì đầu ra IP(M) sau phép hoán vị ban đầu đã biết nên ta cũng biết được nội dung của các thanh ghi dịch trái và phải.

Giả sử  $X = x_1, x_2, \dots, x_{32}$  là nội dung của thanh ghi dịch phải. 32 bít này sẽ qua một phép hoán vị mở rộng  $A = E(X)$ : véctơ 48 bít kết quả  $A = a_1, a_2, \dots, a_{48}$  sẽ được cộng mod 2 theo từng bít với khóa con 48 bít  $K_i = k_{i1}, k_{i2}, \dots, k_{i48}$  ở vòng lặp thứ i lấy ra từ phép biến đổi hoán vị lựa chọn PC 2.

Véctơ 48 bít  $B = b_1, b_2, \dots, b_{48}$  sẽ được đưa qua 8 hộp thay thế  $\{S_k\}_{k=1, \dots, 48}$ . Ở đó mỗi véctơ vào 6 bít  $(b_1, b_2, b_3, b_4, b_5, b_6)$  sẽ được thay thế bằng một véctơ ra 4 bít  $(c_1, c_2, c_3, c_4)$ . Véctơ 32 bít  $C = c_1, c_2, \dots, c_{32}$  lại được biến đổi qua một phép hoán vị P và véctơ 32 bít.

$Y = y_1, y_2, \dots, y_{32}$  sẽ được cộng với nội dung của thanh ghi dịch trái. Thanh ghi dịch phải được cập nhật bằng véctơ 32 bít kết quả này.

$$Y = P(C)$$

Từ hình 2.17 ta thấy  $P^{-1}(Y)$  nếu biết đầu vào X (bản rõ sau phép hoán vị ban đầu IP) thì đầu ra của phép hoán vị mở rộng  $A = E(X)$  cũng đã biết. Tuy nhiên vì khóa con  $K_i = k_{ij}$  với  $j = 1, 2, \dots, 48$  ở vòng lặp thứ i (ta có thể bắt đầu với  $i = 1$ ) là chưa biết nên ta không thể xác định được tổng ở đầu ra của các bộ cộng modulo 2:  $b_j = a_i \oplus k_{ij}$  với  $1 \leq j \leq 48$ . Các bít ở đầu ra các bộ cộng  $(\{b_j\}_{j=1, \dots, 48})$  là các bít vào của 8 hộp thay thế  $S_k$ .

Bây giờ ta quay trở lại nội dung của thanh ghi dịch trái L và nội dung trước đó của thanh ghi dịch phải X' (trên thực tế: Thanh ghi dịch tạm thời TEMP 32 từ vòng lặp trước của DES), ta có thể xác định được véctơ 32 bít Y. Vì Y là kết quả của phép hoán vị chuẩn P của đầu ra từ các hộp thay thế:

$$C = P^{-1}(Y)$$

Véctơ 32 bít  $C = c_1, c_2, \dots, c_{32}$  ở đầu ra của các hộp thay thế cũng được xác định.

Các hộp thay thế  $\{S_k\}_{k=1, \dots, 48}$  phải ngẫu nhiên và không chêch. Với một đầu vào 6 bít bất kỳ  $b_1, b_2, b_3, b_4, b_5$  và  $b_6$ , các bít ra phải có phân bố chuẩn đều. Bây giờ bằng cách tạo của bảng của tất cả 64 véctơ vào của mỗi hộp thay thế, mỗi bít vào  $b_i = 0$  ở một nửa số lần và  $b_i = 1$  ở một nửa số lần khác. Nói một cách khác, ta có thể nói rằng mỗi một bít vào (trong 6 bít) bằng 0 với xác suất  $p = \frac{1}{2}$  và mỗi một bít ra (trong 4 bít) bằng 0 với xác suất  $p = \frac{1}{2}$ .

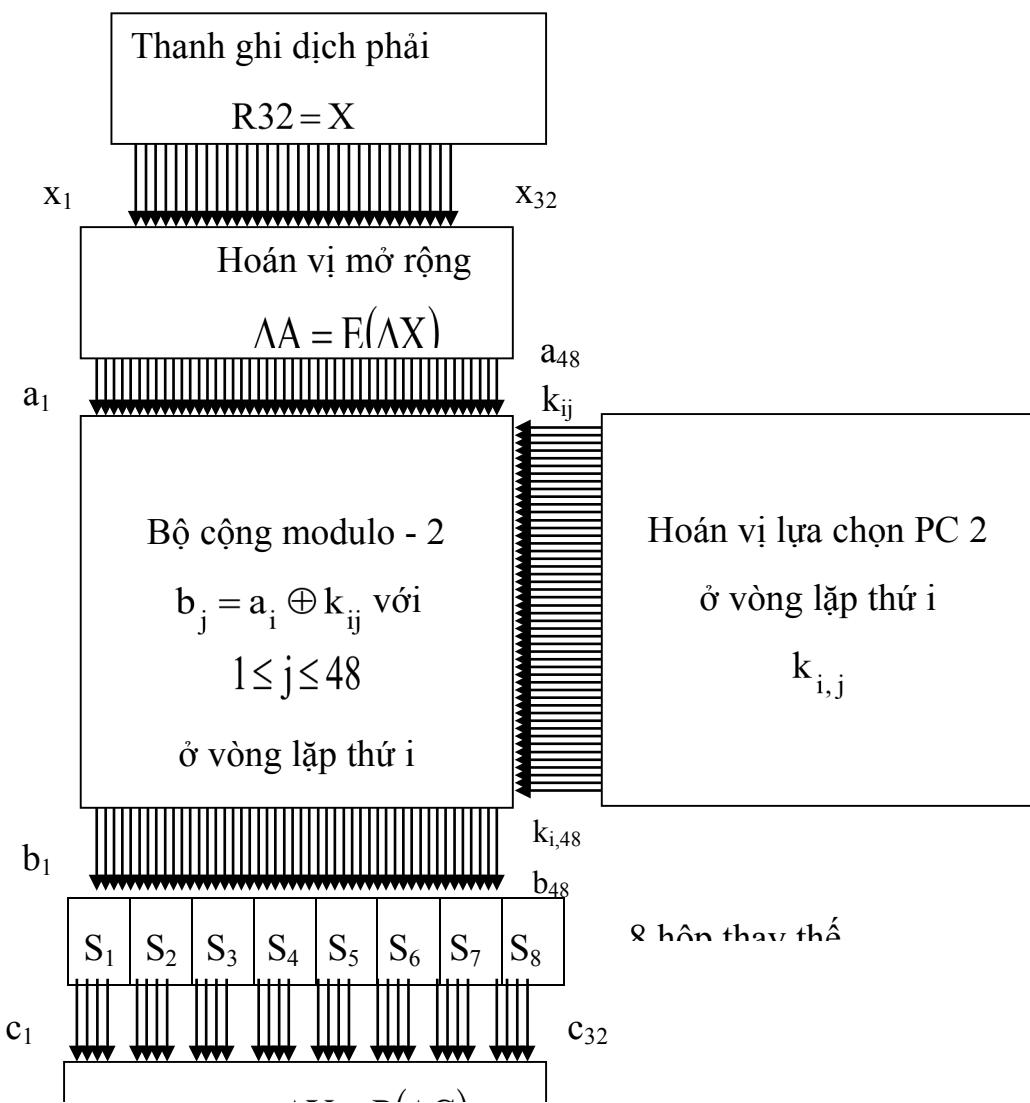
Tuy nhiên ta có thể suy ra đầu vào của một hộp thay thế nếu có thể khai thác được mối quan hệ giữa các đầu vào và các đầu ra của nó. Chẳng hạn nếu ta quan sát 4 bít  $c_1, c_2, c_3$  và  $c_4$  ở đầu ra của một hộp thay thế  $S_k$  và cộng chúng với nhau theo modulo 2 thì đối với 64 véctơ vào khác nhau  $b_1, \dots, b_6$ , kết quả sẽ là  $c_1 \oplus c_2 \oplus c_3 \oplus c_4 = 0$  với một nửa số trường hợp (32 trường hợp) và  $c_1 \oplus c_2 \oplus c_3 \oplus c_4 = 1$  với một nửa số trường hợp còn lại. (Mỗi một giá trị trong 16 véctơ ra sẽ xuất hiện 4 lần trong bảng thay thế).

Ta có thể thấy rằng quan hệ vào – ra của các hộp thay thế không hoàn toàn không chêch. Chẳng hạn, hộp thay thế  $S_5$  là chêch nhất trong các hộp thay thế và ta có thể khai thác nó để suy ra khóa. Bảng 2.2 chỉ ra quan hệ giữa 6 bit vào  $b_{25}, b_{26}, b_{27}, b_{28}, b_{29}$  và  $b_{30}$  và 4 bit ra  $c_{17}, c_{18}, c_{19}$  và  $c_{20}$  trong hộp thay thế  $S_5$ . Từ bảng 2.2 ta có thể thấy rằng ngay cả khi bit vào  $b_{26} = 0$  trong một nửa số trường hợp (tức là với xác suất  $p = \frac{1}{2}$ ) và tổng  $c_1 \oplus c_2 \oplus c_3 \oplus c_4 = 0$

Với xác suất  $p = \frac{1}{2}$  thì phương trình sau:

$$b_{26}^2 = c_1 \oplus c_2 \oplus c_3 \oplus c_4$$

chỉ đúng có 12 lần trong số 62 lần (đúng với xác suất  $p = \frac{12}{64} = \frac{3}{16}$ ) 12 trường hợp này được chỉ ra ở cột kiểm tra trong bảng 2.2.



Hình 0-19. Thám mã tuyến tính của một vòng DES

6 bít vào						Ra	4 bít ra				Kiểm tra
$b_1$	$b_2$	$b_3$	$b_4$	$b_5$	$b_6$		$c_1$	$c_2$	$c_3$	$c_4$	
$b_{25}$	$b_{26}$	$b_{27}$	$b_{28}$	$b_{29}$	$b_{30}$	$c_{17}$	$c_{18}$	$c_{19}$	$c_{20}$		
0	0	0	0	0	0	2	0	0	1	0	
0	0	0	0	0	1	14	1	1	1	0	
0	0	0	0	1	0	12	1	1	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	0	0	0	1	1	11	1	0	1	1	
0	0	0	1	0	0	4	0	1	0	0	
0	0	0	1	0	1	2	0	0	1	0	
0	0	0	1	1	0	1	0	0	0	1	
0	0	0	1	1	1	12	1	1	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	0	1	0	0	0	7	0	1	1	1	
0	0	1	0	0	1	4	0	1	0	0	
0	0	1	0	1	0	10	1	0	1	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	0	1	0	1	1	7	0	1	1	1	
0	0	1	1	0	0	11	1	0	1	1	
0	0	1	1	0	1	13	1	1	0	1	
0	0	1	1	1	0	6	0	1	1	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	0	1	1	1	1	1	0	0	0	1	
0	1	0	0	0	0	8	1	0	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	1	0	0	0	1	5	0	1	0	1	
0	1	0	0	1	0	5	0	1	0	1	
0	1	0	0	1	1	0	0	0	0	0	
0	1	0	1	0	0	3	0	0	1	1	
0	1	0	1	0	1	15	1	1	1	1	
0	1	0	1	1	0	15	1	1	1	1	
0	1	0	1	1	1	10	1	0	1	0	

0	1	1	0	0	0	13	1	1	0	1	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	1	1	0	0	1	3	0	0	1	1	
0	1	1	0	1	0	0	0	0	0	0	
0	1	1	0	1	1	9	1	0	0	1	
0	1	1	1	0	0	14	1	1	1	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	1	1	1	0	1	8	1	0	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
0	1	1	1	1	0	9	1	0	0	1	
0	1	1	1	1	1	6	0	1	1	0	

Hình 0-20. Quan hệ vào ra trong hộp thay thế S<sub>5</sub> (bắt đầu)

6 bít vào						Ra	4 bít ra				Kiểm tra
$b_1$	$b_2$	$b_3$	$b_4$	$b_5$	$b_6$		$c_1$	$c_2$	$c_3$	$c_4$	
$b_{25}$	$b_{26}$	$b_{27}$	$b_{28}$	$b_{29}$	$b_{30}$	$c_{17}$	$c_{18}$	$c_{19}$	$c_{20}$		
1	0	0	0	0	0	4	0	1	0	0	
1	0	0	0	0	1	11	1	0	1	1	
1	0	0	0	1	0	2	0	0	1	0	
1	0	0	0	1	1	8	1	0	0	0	
1	0	0	1	0	0	1	0	0	0	1	
1	0	0	1	0	1	12	1	1	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
1	0	0	1	1	0	11	1	0	1	1	
1	0	0	1	1	1	7	0	1	1	1	
1	0	1	0	0	0	10	1	0	1	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
1	0	1	0	0	1	1	0	0	0	1	
1	0	1	0	1	0	13	1	1	0	1	
1	0	1	0	1	1	14	1	1	1	0	
1	0	1	1	0	0	7	0	1	1	1	
1	0	1	1	0	1	2	0	0	1	0	
1	0	1	1	1	0	8	1	0	0	0	
1	0	1	1	1	1	13	1	1	0	1	
1	1	0	0	0	0	15	1	1	1	1	
1	1	0	0	0	1	6	0	1	1	0	
1	1	0	0	1	0	9	1	0	0	1	
1	1	0	0	1	1	15	1	1	1	1	
1	1	0	1	0	0	12	1	1	0	0	
1	1	0	1	0	1	0	0	0	0	0	
1	1	0	1	1	0	5	0	1	0	1	
1	1	0	1	1	1	9	1	0	0	1	
1	1	1	0	0	0	6	0	1	1	0	
1	1	1	0	0	1	10	1	0	1	0	
1	1	1	1	0	0	3	0	0	1	1	

1	1	1	0	1	1	4	0	1	0	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
1	1	1	1	0	0	0	0	0	0	0	
1	1	1	1	0	1	5	0	1	0	1	
1	1	1	1	1	0	14	1	1	1	0	$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$
1	1	1	1	1	1	3	0	0	1	1	

Hình 0-21. Quan hệ vào ra trong hộp thay thế  $S_5$  (kết thúc)

Ta có thể thấy rằng xác suất để có  $b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$  là  $\frac{3}{16}$  sẽ được dùng để trợ giúp cho việc phá DES. Khi đó với xác suất  $p = \frac{3}{16}$

$$b_{26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$$

$$a_{26} \oplus k_{i26} = c_1 \oplus c_2 \oplus c_3 \oplus c_4$$

Nhưng vì  $A = E(X)$  nên  $a_{26} = x_{17}$ .

Tương tự việc biết ánh xạ của hàm hoán vị chuẩn  $Y = P(C)$  giúp ta có thể thay  $c_{17}, c_{18}, c_{19}$  và  $c_{20}$  bằng các giá trị của bản mã đã biết  $y_3, y_8, y_{14}$  và  $y_{25}$ .

$$\text{Bởi vậy, } k_{i26} = a_{26} \oplus c_{17} \oplus c_{18} \oplus c_{19} \oplus c_{20}$$

$$k_{i26} = a_{26} \oplus c_{17} \oplus c_{18} \oplus c_{19} \oplus c_{20}$$

Vì cặp bản rõ  $X$  và bản mã  $Y$  ở một vòng đã biết nên điều này cung cấp chứng cứ để coi bít  $k_{i26}$  là phần bù của  $x_{17} \oplus y_3 \oplus y_8 \oplus y_{14} \oplus y_{25}$ .

Phân tích một vòng này sẽ được tổng quát hóa lên cho 16 vòng của DES. Điều này có thể thực hiện được vì các nội dung của thanh ghi phải ở vòng lặp thứ hai là một hàm của các kết quả ở vòng lặp thứ nhất.

Thám mã tuyến tính đối với DES vẫn còn khó khả thi vì nó cần tới  $2^{47}$  cặp rõ - mã đã biết để tìm một bít khóa riêng lẻ. Bít khóa thứ hai có thể tìm được.

Người ta đã chỉ ra rằng sử dụng phép xấp xỉ tuyến tính cho DES 14 vòng và đánh giá (phán đoán) 6 bít khóa con  $k_{i25}, k_{i26}, k_{i27}, k_{i28}, k_{i29}$  và  $k_{i30}$  theo 6 bít vào của hộp thay thế  $S_5$  cho các vòng 2 và 14, điều này tương đương với việc thực hiện  $2^{12}$  phép phân tích tuyến tính song song và sẽ tạo ra 26 bít khóa. Điều này sẽ làm giảm không gian khóa cần tìm kiếm từ  $2^{56}$  (khi tìm kiếm vét cạn) xuống còn  $2^{30} = 1.073.741.824$ .

## F. CHUẨN MÃ DỮ LIỆU TIỀN TIẾN (AES)

Vào 1997, Viện tiêu chuẩn và công nghệ quốc gia (NIST) Của Mỹ đã phát động cuộc thi nhằm xây dựng một chuẩn mã dữ liệu mới thay thế cho chuẩn mã dữ liệu cũ DES đã được đưa ra năm 1974. Qua quá trình tuyển chọn vào tháng 10 năm 2000, NIST đã công bố chuẩn mã dữ liệu mới được lựa chọn là thuật toán Rijndael. Đây là một mật mã khối đối xứng với ba kích thước khóa có thể lựa chọn (128 bít, 192 bít và 256 bít). Sau đây ta sẽ mô tả thuật toán AES này.

## 1. Cơ sở toán học của AES

Trong AES các phép toán cộng và nhân được thực hiện trên các byte trong trường hữu hạn  $GF(2^8)$ .

### Phép cộng:

Phép cộng giữa hai phần tử (các byte) trong trường hữu hạn được thực hiện bằng cách cộng theo modulo 2 các bit tương ứng trong biểu diễn của các byte này. Phép cộng các byte A và B với:

$$A = (a_1 \ a_2 \ a_3 \ a_4 \ a_5 \ a_6 \ a_7 \ a_8)$$

$$\text{là } C = A + B \text{ với } C = (c_1 \ c_2 \ c_3 \ c_4 \ c_5 \ c_6 \ c_7 \ c_8)$$

trong đó  $c_i = a_i + b_i \text{ mod } 2$  với  $i = 1, 8$

Các phần tử của trường hữu hạn còn có thể được biểu diễn dưới dạng đa thức. Ví dụ tổng của  $A = 73_H$  và  $B = 4E_H$  (viết dưới dạng cơ số 16 - hexa) là:

$$73_H + 4E_H = 3D_H$$

Viết dưới dạng nhị phân:

$$01110011 + 01001110 = 00111101$$

Viết dưới dạng đa thức:

$$(x^6 + x^5 + x^4 + x + 1) + (x^6 + x^3 + x^2 + x) = (x^5 + x^4 + x^3 + x^2 + 1)$$

### Phép nhân:

Phép nhân được thực hiện trên  $GF(2^8)$  bằng cách nhân hai đa thức rút gọn theo modulo của một đa thức bất khả quy  $m(x)$ .

Trong AES đa thức bất khả quy này là  $m(x) = x^8 + x^4 + x^3 + x + 1$

Ví dụ:  $A = C3_H$ ,  $B = 85_H$  tương ứng với:

$$a(x) = x^7 + x^6 + x + 1 \text{ và } b(x) = x^7 + x^2 + 1$$

$$\text{Khi đó } C = a(x) \cdot b(x) \text{ mod } (x^8 + x^4 + x^3 + x + 1)$$

$$\text{Hay } C = x^7 E_H x^5 = 10101100 + x$$

## 2. Thuật toán AES

AES mã hóa một khối bản rõ M 128 bit thành một khối bản mã C 128 bit bằng cách dùng một khóa mã K có độ dài 128 bit (hoặc 192 hoặc 256 bit) tương ứng với AES – 128 (hoặc AES – 192 hoặc AES – 256). Thuật toán thực hiện trên các byte và kích thước khối đối với đầu vào đầu ra và khóa được biểu thị bằng các từ 32 bit (4 byte).

AES sẽ thực hiện một số vòng mã hóa  $N_r$  phụ thuộc vào độ dài khóa được sử dụng (Xem bảng 2.1)

Thuật toán AES	Độ dài đầu vào/đầu ra	Độ dài khóa $N_k$	Số vòng $N_r$
AES – 128	4 từ	4 từ	10 vòng
AES – 192	4 từ	6 từ	12 vòng
AES – 256	4 từ	8 từ	14 vòng

Hình 0-22. Số các vòng mã hóa của AES

Mã hóa AES:

Mỗi vòng gồm 4 phép biến đổi mật mã theo byte

- Thay thế byte
- Dịch các hàng của mảng trạng thái (State Array)
- Trộn dữ liệu trong một cột của State Array
- Cộng khóa vòng vào State Array

Phép thay thế byte: SubBytes( )

Phép biến đổi AES đầu tiên là một phép thay thế byte phi tuyến gọi là phép biến đổi SubBytes( ), nó hoạt động độc lập trên mỗi byte. Trước tiên nó sẽ tính nghịch đảo của phép nhân trong  $GF(2^8)$ , sau đó sử dụng một phép biến đổi affine trên nghịch đảo này.

$$\begin{bmatrix} b'_0 \\ b'_1 \\ b'_2 \\ b'_3 \\ b'_4 \\ b'_5 \\ b'_6 \\ b'_7 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 & 0 & 1 & 1 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 1 & 1 & 1 & 1 & 0 & 0 & 0 & 1 \\ 1 & 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 0 & 1 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \\ b_4 \\ b_5 \\ b_6 \\ b_7 \end{bmatrix} + \begin{bmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \\ 1 \\ 1 \\ 0 \end{bmatrix}$$

trong đó  $b'_i$  biểu thị bít thứ  $i$  của byte  $b$

Dịch các hàng của State Array; Phép biến đổi ShiftRows()

Phép biến đổi tiếp theo của AES là dịch các hàng của State Array. Lượng dịch Shift( $r, N_b$ ) phụ thuộc vào số hàng  $r$ . Các khối đầu vào (bản rõ) vào các khối đầu ra (bản mã) là các khối 128 bit gồm  $N_b = 4$  từ 32 bit

Phép biến đổi ShiftRows( ) được biểu thị như sau:

$$s'_{r,c} = s_r(c + \text{shift}(r, N_b)) \bmod N_b$$

trong đó  $0 \leq c \leq N_b$

Hàng đầu tiên sẽ không dịch, tức là  $\text{shift}(0, N_b = 4) = 0$

Với các hàng còn lại lượng dịch sẽ tùy theo số hàng

$$\text{shift}(0,4) = 0$$

$$\text{shift}(1,4) = 1$$

$$\text{shift}(2,4) = 2$$

$$\text{shift}(3,4) = 3$$

### Trộn dữ liệu trong một cột State Array: Phép biến đổi Mixcolumns()

Phép biến đổi Mixcolumns( ) được dùng để trộn dữ liệu trong một cột của ma trận trạng thái. Các cột được xem như các đa thức trong  $\text{GF}(2^8)$ . Đầu ra của Mixcolumns( ) là  $s'(x)$  được tạo bằng cách nhân cột với  $s(x)$  với đa thức  $a(x)$  và rút gọn theo  $\text{mod}(X^4 + 1)$

$$s'(x) = a(x).s(x)\text{mod}(X^4 + 1)$$

$$\text{trong đó: } a(x) = 03_H x^3 + 01_H x + 02_H$$

Ở dạng ma trận phép biến đổi này có thể viết như sau:

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02_H & 03_H & 01_H & 01_H \\ 01_H & 02_H & 03_H & 01_H \\ 01_H & 01_H & 02_H & 03_H \\ 03_H & 01_H & 01_H & 02_H \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix}$$

Ở đây  $0 \leq c < N_b$

### Mở rộng khóa AES: KeyExpansion()

Thuật toán AES sẽ tạo từ khóa mã 128 bít (hoặc 192 hoặc 256 bít) một tập khởi tạo  $N_b$  từ 32 bít và  $N_b$  từ 32 bít cho mỗi vòng bao gồm  $N_b(N_r + 1)$  từ 32 bít.

Chương trình giải mã KeyExpansion( ) chứa các SubWord( ) và RotWord( ).

Hàm SubWord( ) là một phép thay thế (hộp S) một từ vào 4 byte bằng một từ ra 4 byte.

Hàm RotWord( ) thực hiện phép hoán vị vòng các byte trong một từ 4 byte (32 bít)  $W_i$ :

$$\text{RotWord}(a_0, a_1, a_2, a_3) = (a_1, a_2, a_3, a_0)$$

$$\text{KeyExpansion} \left( \text{bytekey}[4 * N_k], \text{wordw}[N_b * (N_r + 1)], N_k \right)$$

Begin

$$i = 0$$

```

while ( $i < N_k$ )
     $w[i] = \text{word}[\text{key}[4*i], \text{key}[4*i+1], \text{key}[4*i+2], \text{key}[4*i+3]]$ 
     $i = i + 1$ 
end while

 $i \approx N_k$ 

while ( $i < N_b * (N_r + 1)$ )
    word temp =  $w[i - 1]$ 
    if ( $i \bmod N_k = 0$ )
        temp = SubWord(RotWord(temp)) xor Rconw[i/N_k]
    else if ( $N_k = 8$  and  $i \bmod N_k = 4$ )
        temp = SubWord(temp)
    end if
     $w[i] \approx w[i - N_k] = \text{xor } temp$ 
     $i = i + 1$ 
end while

end

```

(nguồn trích dẫn: Đặc tả thô AES: <http://csrc.nist.gov/encryption/aes/>)

Chương trình giải mã của AES

Cipher ( $\text{bytein}[4*N_b], \text{byteout}[4*N_b], \text{word } w[N_b * (N_r + 1)]$ )

Begin byte state [ $4, N_b$ ] state = in AddRoundKey(state,w)

for round = 1 step 1 to  $N_r - 1$

SubBytes (state), ShifRows (state),

Mixcolumns(state), AddRoundKey(state,w+round\*N\_b)

end for

SubBytes (state), ShifRows (state)

**AddRoundKey(state,w+N\_r\*N\_b)**

out = state

end

#### 4.2.3 ƯU NHƯỢC ĐIỂM CỦA CÁC HỆ MẬT KHÓA BÍ MẬT

Ưu điểm của các hệ mật khóa bí mật:

- Mô hình khá đơn giản
- Dễ dàng tạo ra thuật toán mã hóa đối xứng cho cá nhân
- Dễ cài đặt và hoạt động hiệu quả

- Hoạt động nhanh và hiệu quả do tốc độ mã hóa và giải mã cao

Nhược điểm:

- Dùng chung khóa nên nhiều nguy cơ mất an toàn
- Khóa dùng chung rất dễ bị hóa giải (“bẻ khóa”) do phải truyền trên kênh truyền tin đến bên nhận
- Việc gửi thông tin cùng khóa cho số lượng lớn là khó khăn.

### 4.3. Mã hóa khóa công khai

#### 4.3.1 GIỚI THIỆU VỀ MẬT MÃ KHÓA CÔNG KHAI

Trong mô hình mật mã cổ điển trước đây mà hiện nay đang được nghiên cứu Alice (người gửi) và Bob (người nhận) chọn một cách bí mật khoá K. Sau đó dùng K để tạo luật mã hóa  $e_k$  và luật giải mã  $d_k$ . Trong hệ mật này  $d_k$  hoặc giống  $e_k$  hoặc dễ dàng nhận được từ nó (ví dụ trong hệ DES quá trình giải mã hoàn toàn tương tự như quá trình mã nhưng thủ tục khoá ngược lại). Các hệ mật thuộc loại này được gọi là hệ khoá bí mật, nếu để lộ  $e_k$  thì làm cho hệ thống mất an toàn.

Nhược điểm của hệ mật này là nó yêu cầu phải có thông tin trước về khoá K giữa Alice và Bob qua một kênh an toàn trước khi gửi một bản mã bất kỳ. Trên thực tế điều này rất khó đảm bảo. Chẳng hạn khi Alice và Bob ở cách xa nhau và họ chỉ có thể liên lạc với nhau bằng thư tín điện tử (E.mail). Trong tình huống đó Alice và Bob không thể tạo một kênh bảo mật với giá phải chăng.

Ý tưởng xây dựng một hệ mật khoá công khai (hay dùng chung) là tìm một hệ mật không có khả năng tính toán để xác định  $d_k$  khi biết  $e_k$ . Nếu thực hiện được như vậy thì quy tắc mã  $e_k$  có thể được công khai bằng cách công bố nó trong một danh bạ (bởi vậy nên có thuật ngữ *hệ mật khoá công khai*). Ưu điểm của hệ mật khoá công khai là ở chỗ Alice (hoặc bất kỳ ai) có thể gửi một bản tin đã mã cho Bob (mà không cần thông tin trước về khoá mật) bằng cách dùng mật mã công khai  $e_k$ . Người nhận A sẽ là người duy nhất có thể giải được bản mã này bằng sử dụng luật giải bí mật  $d_k$  của mình.

Có thể hình dung hệ mật này tương tự như sau. Alice đặt một vật vào một hộp kim loại và rồi khoá nó lại bằng một khoá số do Bob để lại. Chỉ có Bob là người duy nhất có thể mở được hộp vì chỉ có anh ta mới biết tổ hợp mã của khoá số của mình.

Ý tưởng về một hệ mật khoá công khai được Diffie và Hellman đưa ra vào năm 1976. Còn việc hiện thực hoá nó thì do Rivesrt, Shamir và Adleman đưa ra lần đầu tiên vào năm 1977, họ đã tạo nên hệ mật nổi tiếng RSA (sẽ được nghiên cứu trong chương này). Kể từ đó đã công bố một số hệ, độ mật của chúng dựa trên các bài tính toán khác nhau. Trong đó, quan trọng nhất là các hệ mật khoá công khai sau:

- *Hệ mật RSA:*

Độ bảo mật của hệ RSA dựa trên độ khó của việc phân tích ra thừa số nguyên lớn. Hệ này sẽ được mô tả trong phần 4.2.

- *Hệ mật xếp ba lô Merkle - Hellman:*

Hệ này và các hệ liên quan dựa trên tính khó giải của bài toán tổng các tập con (bài toán này là bài toán NP đầy đủ - là một lớp khá lớn các bài toán không có giải thuật được biết trong thời gian đa thức). Tuy nhiên tất cả các hệ mật xếp ba lô khác nhau đều đã bị chứng tỏ là không an toàn (ngoại trừ hệ mật Chor-Rivest).

- *Hệ mật McEliece:*

Hệ này dựa trên lý thuyết mã đại số và vẫn còn được coi là an toàn. Hệ mật McEliece dựa trên bài toán giải mã cho các mã tuyến tính (cũng là một bài toán NP đầy đủ). Hệ mật McEliece được trình bày ở phần 4.6.

- *Hệ mật ElGamal:*

Hệ mật ElGamal dựa trên tính khó giải của bài toán logarithm rời rạc trên các trường hữu hạn

- *Hệ mật Chor-Rivest:*

Hệ mật Chor-Rivest cũng được xem như một hệ mật xếp ba lô. Tuy nhiên nó vẫn được coi là an toàn

- *Hệ mật trên các đường cong Elliptic:*

Các hệ mật này là biến tướng của các hệ mật khác (chẳng hạn như hệ mật ElGamal), chúng làm việc trên các đường cong Elliptic chứ không phải là trên các trường hữu hạn. Hệ mật này đảm bảo độ mật với số khoá nhỏ hơn các hệ mật khoá công khai khác.

Một chú ý quan trọng là một hệ mật khoá công khai không bao giờ có thể đảm bảo được độ mật tuyệt đối (an toàn vô điều kiện). Sở dĩ như vậy vì đối phương khi nghiên cứu một bản mã, y có thể mã lần lượt các bản tin rõ bằng luật mã hoá công khai  $e_k$  cho tới khi anh ta tìm được bản rõ duy nhất x đảm bảo  $y = e_k(x)$ . Bản rõ này chính là kết quả giải mã của y. Bởi vậy, ta chỉ nghiên cứu độ mật về mặt tính toán của các hệ mật này.

Một khái niệm có ích khi nghiên cứu hệ mật khoá công khai là khái niệm về hàm cửa sổ một chiều. Ta sẽ định nghĩa khái niệm này một cách không hình thức.

Hàm mã khoá công khai  $e_k$  của Bob phải là một hàm dễ tính toán. Song việc tìm hàm ngược (hàm giải mã) rất khó khăn (đối với bất kỳ ai không phải là Bob). Đặc tính khó tính toán hàm ngược thường được gọi là đặc tính một chiều. Bởi vậy điều kiện cần thiết là  $e_k$  phải là hàm một chiều.

Các hàm một chiều đóng vai trò quan trọng trong mật mã học, chúng rất quan trọng trong các hệ mật khoá công khai và trong nhiều lĩnh vực khác. Đáng tiếc là mặc dù có rất nhiều hàm được coi là hàm một chiều nhưng cho đến nay vẫn không tồn tại một hàm nào có thể chứng minh được là hàm một chiều.

Sau đây là một ví dụ về một hàm được coi là hàm một chiều. Giả sử n là tích của hai số nguyên tố lớn p và q, giả sử b là một số nguyên dương. Khi đó ta xác định ánh xạ  $f : Z_n \rightarrow Z_n$  là  $f(x) = x^b \text{ mod } n$  (với b và n đã được chọn thích hợp thì đây chính là hàm mã RSA, sau này ta sẽ nói nhiều hơn về nó).

Để xây dựng một hệ mật khoá công khai thì việc tìm được một hàm một chiều vẫn chưa đủ. Ta không muốn  $e_k$  là hàm một chiều đối với Bob vì anh ta phải có khả năng giải mã các bản tin nhận được một cách hiệu quả. Điều cần thiết là Bob phải có một cửa sập chứa thông tin bí mật cho phép dễ dàng tìm hàm của  $e_k$ . Như vậy Bob có thể giải mã một cách hữu hiệu vì anh ta có một hiểu biết tuyệt mật nào đó về K. Bởi vậy một hàm được gọi là cửa sập một chiều nếu nó là một hàm một chiều và nó trở nên dễ tính ngược nếu biết một cửa sập nhất định.

#### 4.3.2 BÀI TOÁN PHÂN TÍCH THÙA SỐ VÀ CÁC HỆ MẬT CÓ LIÊN QUAN

##### 1. Bài toán phân tích thừa số

Bài toán phân tích một số nguyên  $> 1$  thành thừa số nguyên tố cũng được xem là một bài toán khó thường được sử dụng trong lý thuyết mật mã. Biết một số  $n$  là hợp số thì việc phân tích  $n$  thành thừa số mới là có nghĩa, do đó thường khi để giải bài toán phân tích  $n$  thành thừa số, ta thử trước  $n$  có là hợp số hay không; và bài toán phân tích  $n$  thành thừa số có thể dẫn về bài toán *tìm một ước số* của  $n$ , vì khi biết một ước số  $d$  của  $n$  thì tiến trình phân tích  $n$  được tiếp tục thực hiện bằng cách phân tích  $d$  và  $n/d$ .

*Bài toán phân tích thành thừa số*, hay bài toán tìm ước số của một số nguyên cho trước, đã được nghiên cứu nhiều, nhưng cũng chưa có một thuật toán hiệu quả nào để giải nó trong trường hợp tổng quát mà người ta có xu hướng giải bài toán này theo những trường hợp đặc biệt của số cần phải phân tích, chẳng hạn khi  $n$  có một ước số nguyên tố  $p$  với  $p - 1$  là B-min với một cận  $B > 0$  nào đó, hoặc khi  $n$  là số Blum, tức là số có dạng tích của hai số nguyên tố lớn nào đó ( $n = p \cdot q$ ).

Ta xét trường hợp thứ nhất với  $(p - 1)$ -thuật toán Pollard như sau: Một số nguyên  $n$  được gọi là B-min nếu tất cả các ước số nguyên tố của nó đều  $\leq B$ . Ý chính chúa trong  $(p - 1)$ -thuật toán Pollard như sau: Giả sử  $n$  là B-min. Kí hiệu  $Q$  là bội chung bé nhất của tất cả các lũy thừa của các số nguyên tố  $\leq B$  mà bản thân chúng  $\leq n$ . Nếu  $q^l \leq n$  thì  $l \ln q \leq \ln n$ , tức  $l \leq \left\lfloor \frac{\ln n}{\ln q} \right\rfloor$  ( $\lfloor x \rfloor$  là số nguyên bé nhất lớn hơn  $x$ )

Ta có:

$$Q = \prod q^{\lfloor \ln n / \ln q \rfloor}$$

Trong đó tích lấy theo tất cả các số nguyên  $q \leq B$  khác nhau  $q \leq B$ . Nếu  $p$  là một thừa số nguyên tố của  $n$  sao cho  $p - 1$  là B-min thì  $p-1|Q$  và do đó với mọi  $a$  bất kì thỏa mãn  $\gcd(a, p) = 1$ , theo định lý Fermat ta có  $a^Q \equiv 1 \pmod{p}$ . Vì vậy, nếu lấy  $d = \gcd(a^Q - 1, n)$  thì  $p|d$ . Nếu  $d = n$  thì coi như thuật toán không cho ta điều mong muốn, tuy nhiên điều đó chắc không xảy ra nếu  $n$  có ít nhất hai thừa số nguyên tố khác nhau. Từ những lập luận đó ta có:

***(p - 1)-thuật toán Pollard phân tích thành thừa số:***

VÀO: một hợp số  $n$  không phải lũy thừa của một số nguyên tố

RA: một thừa số không tầm thường của  $n$ .

1. Chọn một cận cho độ min B

2. Chọn ngẫu nhiên một số nguyên  $a$ ,  $2 \leq a \leq n - 1$ , và tính  $d = \gcd(a, n)$ . Nếu  $d \geq 2$  thì cho ra kết quả (d)

3. Với mỗi số nguyên tố  $q \leq B$  thực hiện:

$$3.1. \text{Tính } l = \left\lfloor \frac{\ln n}{\ln q} \right\rfloor$$

$$3.2. \text{Tính } a \leftarrow a^{q^l} \bmod n$$

4. Tính  $d = \gcd(a - 1, n)$

5. Nếu  $1 < d < n$  thì cho ra kết quả (d). Nếu ngược lại thì thuật toán coi như không có kết quả.

*Ví dụ:* Dùng thuật toán cho số  $n = 19048567$ . Ta chọn  $B = 19$ , và  $a = 3$  và tính được  $\gcd(3, n) = 1$ . Chuyển sang thực hiện bước 3 ta được bảng sau đây (mỗi hàng ứng với một giá trị của  $q$ ):

$q$	1	$a$
2	24	2293244
3	15	13555889
5	10	16937223
7	8	15214586
11	6	9685355
13	6	13271154
17	5	11406961
19	5	554506

Bảng 0-1. Kết quả tính bước 3 của thuật toán Pollard

Sau đó ta tính  $d = \gcd(554506 - 1, 19048567) = 5281$ . Vậy ta được một thừa số  $p = 5281$ , và do đó một thừa số nữa là  $q = n/p = 3607$ . Cả hai thừa số đó đều là số nguyên tố.

Chú ý rằng ở đây  $p - 1 = 2^5 \cdot 3 \cdot 5 \cdot 11$ , có tất cả các ước số nguyên tố  $\leq 19$ , do đó chắc chắn thuật toán sẽ kết thúc có kết quả. Thuật toán sẽ kết thúc không có kết quả khi độ mịn B được chọn quá bé để không một thừa số nguyên tố p nào của n mà  $p - 1$  chỉ chứa các ước số nguyên tố  $\leq B$ . Như vậy, có thể xem  $(p-1)$ -thuật toán Pollard phân tích n thành thừa số nguyên tố là có hiệu quả đối với những số nguyên n là B-mịn, người ta tính được thời gian cần để thực hiện thuật toán đó là cỡ  $O(B \ln n / \ln B)$  phép nhân theo môđuyn.

Bây giờ ta xét trường hợp các số nguyên Blum, tức là các số có dạng  $n = p \cdot q$ , tích của hai số nguyên tố lớn. Trước hết ta chú ý rằng nếu ta biết hai số nguyên khác nhau  $x, y$  sao cho  $x^2 \equiv y^2 \pmod{n}$  thì ta dễ tìm được một thừa số của n. Thực vậy, từ  $x^2 \equiv y^2 \pmod{n}$  ta có  $x^2 - y^2 = (x - y)(x + y)$  chia hết cho n, do n không là ước số của  $x + y$  hoặc  $x - y$  nên  $\gcd(x - y, n)$  phải là một ước số của n, tức bằng p hoặc q.

Ta biết nếu  $n = p \cdot q$  là số Blum thì phương trình đồng dư

$$x^2 \equiv a^2 \pmod{n}$$

có 4 nghiệm, hai nghiệm tầm thường là  $x = a$  và  $x = -a$ . Hai nghiệm không tầm thường khác là  $\pm b$ , chúng là nghiệm của hai hệ phương trình đồng dư bậc nhất sau đây:

$$\begin{cases} x \equiv a \pmod{p} \\ x \equiv -a \pmod{q} \end{cases} \quad \begin{cases} x \equiv -a \pmod{p} \\ x \equiv a \pmod{q} \end{cases}$$

Bằng lập luận như trên ta thấy rằng nếu  $n$  là số Blum,  $a$  là một số nguyên tố với  $n$  và ta biết một nghiệm không tầm thường của phương trình  $x^2 \equiv a^2 \pmod{n}$ , tức biết một  $x \neq \pm a$  sao cho  $x^2 \equiv a^2 \pmod{n}$  thì  $\gcd(x-a, n)$  sẽ là một ước số của  $n$ . Những điều trên đây là căn cứ cho một số phương pháp tìm ước số nguyên tố của một số nguyên dạng Blum; ý chung của các phương pháp đó là dẫn về việc tìm một nghiệm không tầm thường của một phương trình dạng  $x^2 \equiv a^2 \pmod{n}$ , chẳng hạn như phương trình  $x^2 \equiv 1 \pmod{n}$ .

Một trường hợp khá lý thú trong lý thuyết mật mã là khi ta biết hai số  $a, b$  là nghịch đảo của nhau theo mod  $\phi(n)$  (nhưng không biết  $\phi(n)$ ) và tìm một phân tích thành thừa số của  $n$ . Bài toán được đặt ra cụ thể là: Biết  $n$  có dạng Blum, biết  $a$  và  $b$  sao cho  $ab \equiv 1 \pmod{\phi(n)}$ . Hãy tìm một ước số nguyên tố của  $n$ , hay tìm một nghiệm không tầm thường của phương trình  $x^2 \equiv 1 \pmod{n}$ . Ta giả thiết  $ab - 1 = 2^s \cdot r$  với  $r$  là số lẻ. Ta phát triển một thuật toán xác suất kiểu Las Vegas như sau: Ta chọn một số ngẫu nhiên  $v$  ( $1 \leq v \leq n-1$ ). Nếu may mắn  $v$  là bội số của  $p$  hay  $q$ , thì ta được ngay một ước số của  $n$  là  $\gcd(v, n)$ . Nếu  $v$  nguyên tố với  $n$ , thì ta tính các bình phương liên tiếp kể từ  $v^r$ , được  $v^r, v^{2r}, v^{4r}, \dots$  cho đến khi được  $v^{2^t \cdot r} \equiv 1 \pmod{n}$  với một  $t$  nào đó. Số  $t$  như vậy bao giờ cũng đạt được vì có  $2^s \cdot r \equiv 0 \pmod{\phi(n)}$  nên có  $v^{2^t \cdot r} \equiv 1 \pmod{n}$ . Như vậy, ta đã tìm được một số  $x = v^{2^{t-1} \cdot r}$  sao cho  $x^2 \equiv 1 \pmod{n}$ . Tất nhiên có  $x \neq 1 \pmod{n}$ . Nếu cũng có  $x \neq -1 \pmod{n}$  thì  $x$  là nghiệm không tầm thường của  $x^2 \equiv 1 \pmod{n}$ , từ đó ta có thể tìm ước số của  $n$ . Nếu không thì thuật toán coi như thất bại, cho ta kết quả *không đúng*. Người ta có thể ước lượng xác suất cho kết quả *không đúng* với một lần thử với một số  $v$  là  $< 1/2$ , do đó nếu ta thiết kế thuật toán với  $m$  số ngẫu nhiên  $v_1, \dots, v_m$ , thì sẽ có thể đạt được xác suất cho kết quả không đúng là  $< 1/2^m$ !

## 2. Hệ mật RSA

### 1. Thuật toán 1: Tạo khóa.

Tóm lược: Mỗi đầu cần tạo một khoá công khai và một khoá riêng tương ứng theo các bước sau:

- Tạo 2 số nguyên tố lớn ngẫu nhiên và khác nhau  $p$  và  $q$ .  $p$  và  $q$  có độ lớn xấp xỉ nhau.
- Tính  $n = p \cdot q$  và  $\Phi(n) = (p-1)(q-1)$ .
- Chọn một số nguyên ngẫu nhiên  $e$ ,  $1 < e < \Phi$ , sao cho  $(e, \Phi) = 1$ .
- Tính một số nguyên  $d$  duy nhất,  $1 < d < \Phi$  thoả mãn  $ed \equiv 1 \pmod{\Phi}$ .
- Khoá công khai là cặp số  $(n, e)$ . Khoá riêng bí mật là  $d$ .

### 2. Thuật toán 2: Mã hóa công khai RSA

Tóm lược: B mã hóa một thông báo  $m$  để gửi cho A bản mã cần giải.

**Mã hóa:** B phải thực hiện:

- Thu nhận khoá công khai  $(n, e)$  của A.

2. Biểu diễn bản tin dưới dạng một số nguyên  $m$  trong khoảng  $[0, n-1]$
3. Tính  $c = m^e \text{ mod } n$ .
4. Gửi bản mã  $c$  cho A.

**Giải mã:** Khôi phục bản rõ  $m$  từ  $c$ . A phải thực hiện phép tính sau bằng cách dùng khoá riêng  $m = c^d \text{ mod } n$

*Chứng minh hoạt động giải mã:*

Vì  $ed \equiv 1 \pmod{\Phi}$  nên luôn tồn tại một số nguyên  $k$  sao cho  $ed = 1 + k\Phi$ . Bây giờ nếu  $(m, p) = 1$  theo định lý Fermat ta có:  $m^{p-1} \equiv 1 \pmod{p}$ . Luỹ thừa cả hai vế của đồng dư thức trên với số mũ  $k(q-1)$  và rồi nhân cả hai vế với  $m$  ta có:

$$m^{1+k(q-1)(p-1)} \equiv m \pmod{p}$$

Mặt khác nếu  $\text{UCLN}(m, p) = p$  thì đồng dư thức cuối cùng ở trên vẫn đúng vì mỗi vế đều đồng dư với  $0 \pmod{p}$ . Bởi vậy, trong mọi trường hợp ta đều có:

$$m^{ed} \equiv m \pmod{p}$$

Bằng lập luận tương tự ta lại có:  $m^{ed} \equiv m \pmod{q}$

Cuối cùng vì  $p$  và  $q$  là các số nguyên tố khác nhau nên  $m^{ed} \equiv m \pmod{n}$  và bởi vậy  $c^d \equiv (m^e)^d \equiv m \pmod{n}$ .

*Ví dụ*

### Tạo khoá

A chọn các số nguyên tố  $p = 2357$ ,  $q = 2551$  và tính  $n = p \cdot q = 6012707$  và  $\Phi = (p-1)(q-1) = 6007800$ . A chọn  $e = 3674911$  và dùng thuật toán Euclidean mở rộng để tìm được  $d = 422191$  thoả mãn  $ed \equiv 1 \pmod{\Phi}$ . Khoá công khai của A là cặp số ( $n = 6012707$ ,  $e = 3674911$ ), khoá bí mật của A là  $d = 422191$ .

### Mã hoá

Để mã hoá thông báo  $m = 5234673$ , B sử dụng thuật toán lấy luỹ thừa theo modulo để tính.

$$c = m^e \text{ mod } n = 5234673^{3674911} \text{ mod } 6012707 = 3650502$$

rồi gửi  $c$  cho A.

### Giải mã

Để giải mã bản mã  $c$ , A tính:

$$c^d \text{ mod } n = 3650502^{422191} \text{ mod } 6012707 = 5234673$$

### Chú ý (Số mũ vạn năng).

Số  $\lambda = \text{BCNN}(p-1, q-1)$  đôi khi được gọi là số mũ vạn năng của  $n$ ,  $\lambda$  có thể được dùng thay cho  $\Phi = (p-1)(q-1)$  khi tạo khoá RSA. Cần chú ý rằng  $\lambda$  là ước thực sự của

$\Phi$ . Sử dụng  $\lambda$  có thể thu được số mũ giải mã d nhỏ hơn (làm cho giải mã nhanh hơn). Tuy nhiên, nếu  $p$  và  $q$  được chọn ngẫu nhiên thì  $\text{UCLN}(p-1, q-1)$  sẽ khá nhỏ và bởi vậy  $\Phi$  và  $\lambda$  sẽ là các số có kích thước xấp xỉ.

### Vấn đề điểm bắt động trong RSA

Giả sử rằng cặp khóa công khai là  $(e, n) = (17, 35)$ .

Giả sử thông báo có giá trị bằng 8.

Ta có  $8^{17} \equiv 8 \pmod{35}$ .

Như vậy mã hóa của thông báo vẫn là thông báo ban đầu. Nói một cách khác với khóa mã là 17 thì thông tin không được che dấu. Rõ ràng là phải tránh được tình trạng này định lý sau cho ta tính được số bản tin không thể che dấu được với một lựa chọn cho trước của  $(e, n)$ .

### Định lý:

Nếu các thông báo được mã bằng hệ mật RSA với cặp khóa công khai  $(e, n)$  với  $n = p \cdot q$  thì số các thông báo không thể che dấu được bằng:

$$N = (1 + \text{UCLN}(e-1, p-1))(1 + \text{UCLN}(e-1, q-1))$$

### Chứng minh:

Một thông báo là không thể che dấu được nếu  $M^e \equiv M \pmod{n}$

Ta có:  $M^e \equiv M \pmod{p}$  và  $M^e \equiv M \pmod{q}$ .

Ta có thể viết lại các phương trình trên như sau:

$$M^{e-1} \equiv 1 \pmod{p} \text{ hoặc } M^{e-1} \equiv 0 \pmod{p}$$

$$M^{e-1} \equiv 1 \pmod{q} \text{ hoặc } M^{e-1} \equiv 0 \pmod{q}$$

Chú ý rằng phương trình đồng dư  $M^{e-1} \equiv 0 \pmod{p}$  chỉ có một nghiệm tương tự với  $q$  ta có được kết quả của định lý

### Ví dụ: $n = 35$

Giả sử  $e = 3$  ta có  $(1 + \text{UCLN}(2, 4))(1 + \text{UCLN}(2, 6)) = 9$

Các thông báo không thể che dấu được là 9 thông báo sau:

$$\{0, 1, 6, 14, 15, 20, 21, 29, 34\}$$

Giả sử  $e = 17$ . ta có  $(1 + \text{UCLN}(6, 4))(1 + \text{UCLN}(16, 6)) = 15$

Các thông báo không thể che dấu được là 15 thông báo sau:

$$\{0, 1, 6, 7, 8, 13, 14, 15, 20, 21, 22, 27, 28, 29, 34\}$$

Giả sử  $p = 2p' + 1$  và  $q = 2q' + 1$  trong đó  $p'$  và  $q'$  là các số nguyên tố. Khi đó:

$$\text{UCLN}(e-1, 2p') = 1; 2 \text{ hoặc } p'$$

Nếu  $\text{UCLN}(e-1, 2p')$  không phải là  $p'$  và  $\text{UCLN}(e-1, 2q')$  không phải là  $q'$  thì số thông báo không thể che dấu chỉ nhiều nhất là 9.

Nếu  $\text{UCLN}(e-1, 2p') = p'$  thì số các thông báo không thể che dấu tối thiểu là  $2(p'+1)$ . Tuy nhiên xác suất để xảy ra điều này là rất nhỏ ( $b \gg ngl/p'$ )

### 3. Hệ mật Rabin

#### a. Thuật toán 1: Tạo khóa

Tóm lược: Mỗi đầu tạo một khoá công khai và một khoá bí mật tương ứng theo các bước sau:

- (1) Tạo 2 số nguyên tố lớn, ngẫu nhiên và phân biệt  $p$  và  $q$  có kích thước xấp xỉ nhau.
- (2) Tính  $n = p \cdot q$ .
- (3) Khoá công khai là  $n$ , khoá bí mật là các cặp số  $(p, q)$ .

#### b. Thuật toán 2: Mã hóa công khai Rabin

**Mã hóa:** B phải thực hiện các bước sau:

- (1) Nhận khoá công khai của A:  $n$ .
- (2) Biểu thị bản tin dưới dạng một số nguyên  $m$  nằm trong dải  $[0, n - 1]$ .
- (3) Tính  $c = m^2 \bmod n$ .
- (4) Gửi bản mã  $c$  cho A.

**Giải mã:** Để khôi phục bản rõ  $m$  từ  $c$ , A phải thực hiện các bước sau: Tìm 4 căn bậc hai của  $c \bmod n$  là  $m_1, m_2, m_3$  hoặc  $m_4$ .

- (1) Thông báo cho người gửi là một trong 4 giá trị  $m_1, m_2, m_3$  hoặc  $m_4$ . Bằng một cách nào đó A sẽ quyết định  $m$  là giá trị nào.

Ví dụ

#### Tạo khóa.

A chọn các số nguyên tố  $p = 277$  và  $q = 331$ . A tính  $n = p \cdot q = 91687$ . Khoá công khai của A là 91687. Khoá bí mật của A là cặp số  $(p = 277, q = 331)$ .

#### Mã hóa

Giả sử rằng 6 bit cuối cùng của bản tin gốc được lặp lại trước khi thực hiện mã hóa. Việc thêm vào độ thừa này nhằm giúp cho bên giải mã nhận biết được bản mã đúng.

Để mã hóa bản tin 10 bit  $\bar{m} = 1001111001$ , B sẽ lặp lại 6 bit cuối cùng của  $\bar{m}$  để có được bản tin 16 bit sau:  $m = 1001111001111001$ , biểu diễn thập phân tương ứng là  $m = 40596$ .

Sau đó B tính  $c = m^2 \bmod n = 40596^2 \bmod 91687 = 62111$  rồi gửi  $c$  cho A

#### Giải mã

Để giải mã bản mã  $c$ , A tính bốn giá trị căn bậc 2 của  $c \bmod n$ :

$$m_1 = 69654, \quad m_2 = 22033, \quad m_3 = 40596, \quad m_4 = 51118$$

Biểu diễn nhị phân tương ứng của các số trên là:

$$\begin{aligned} m_1 &= 10001000000010110, & m_2 &= 101011000010001 \\ m_3 &= 1001111001111001, & m_4 &= 1100011110101110 \end{aligned}$$

Vì chỉ có  $m_3$  mới có độ thừa cần thiết nên A sẽ giải mã  $c$  bằng  $m_3$  và khôi phục lại bản tin gốc là  $\bar{m} = 1001111001$ .

### **Đánh giá hiệu quả**

Thuật toán mã hoá Rabin là một thuật toán cực nhanh vì nó chỉ cần thực hiện một phép bình phương modulo đơn giản. Trong khi đó, chẳng hạn với thuật toán RSA có  $e = 3$  phải cần tới một phép nhân modulo và một phép bình phương modulo. Thuật toán giải mã Rabin có chậm hơn thuật toán mã hoá, tuy nhiên về mặt tốc độ nó cung tương đương với thuật toán giải mã RSA.

### **4.3.3 BÀI TOÁN LOGARIT RỜI RẠC VÀ CÁC HỆ MẬT CÓ LIÊN QUAN**

#### **1. Bài toán logarit rời rạc**

Giả sử cho  $Z_n^*$  là một trường hữu hạn với  $p$  là một nguyên tố lớn.

Cho  $g$  là phần tử sinh của nhóm nhân  $Z_p^*$  tức là với một phần tử  $a \neq 0$  bất kỳ thuộc  $Z_n$  ta có thể tìm được một số nguyên tố  $x$  duy nhất thỏa mãn:

$$a = g^x$$

Ta có thể viết:  $\log_g a = x$ .

Bài toán logarit rời rạc chính là bài toán tìm  $x$ .

**Ví dụ:** Xét  $Z_{19}$ , phần tử sinh  $g = 2$ . Ta có bảng sau:

x	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$\log_2 x$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

Từ bảng trên ta có:  $2^{13} \equiv 3 \pmod{19}$ .

Nhìn chung đây là một bài toán rất khó khi  $p$  đủ lớn (chẳng hạn  $p \approx 10^{200}$ ). Khi đó ngay cả với các máy tính cực mạnh ta cũng phải chịu bó tay. Tuy nhiên, trên thực tế bài toán này chỉ thực sự khó khi  $p - 1$  không phải là tích của các số nguyên tố nhỏ. Nói chung bài toán logarit rời rạc trên trường hữu hạn  $GF(p)$  có độ phức tạp lớn hơn so với trên  $GF(2^m)$ .

#### **a. Thuật toán bước đi lớn bước đi nhỏ (Baby step giant step):**

VÀO: số nguyên  $n$ , phần tử sinh  $\alpha$  của  $Z_n^*$ , và phần tử  $\beta \in Z_n^*$

RA:  $\log_\alpha \beta$  trên  $Z_n^*$   
1. Tính  $m = \lceil \sqrt{\text{ord}(\alpha)} \rceil$

2. Lập bảng  $(j, \alpha^j \pmod{n})$  với  $j = \overline{0 \rightarrow m-1}$

3. Tính  $\beta \cdot (\alpha^{-m})^i \pmod{n}$  với  $i = \overline{0 \rightarrow m-1}$

4. Tra bảng  $(j, \alpha^j \pmod{n})$  cho tới khi thỏa mãn  $\beta \cdot (\alpha^{-m})^i = \alpha^j$

5. Khi đó  $\log_\alpha \beta = m \cdot i + j$

#### **b. Thuật toán trao đổi khóa Diffie – Hellman**

Các bên A và B mỗi bên gửi cho nhau một thông báo trên kênh mở và sẽ đạt được khóa chia sẻ K giữa hai bên A và B

1. Thiết lập một lần: Một số nguyên tố phù hợp  $p$  và phần tử sinh  $\alpha \in Z_p^*$  với  $2 \leq \alpha \leq p - 2$  được lựa chọn và công bố.

2. Các thông báo của giao thức:

$$A \rightarrow B: \alpha^x \text{ mod } p \quad (1)$$

$$B \rightarrow A: \alpha^y \text{ mod } p \quad (2)$$

3. Các hành động của giao thức: thực hiện các bước sau đây mỗi khi khóa chia sẻ được yêu cầu:

(a) A chọn số ngẫu nhiên số nguyên bí mật  $x$ ,  $1 \leq x \leq p - 2$  và gửi thông báo (1) cho B.

(b) B chọn số ngẫu nhiên số nguyên bí mật  $y$ ,  $1 \leq y \leq p - 2$  và gửi thông báo (2) cho A.

(c) B nhận được  $\alpha^x$  và tính khóa chia sẻ là  $K = (\alpha^x)^y \text{ mod } p$

(d) A nhận được  $\alpha^y$  và tính khóa chia sẻ là  $K = (\alpha^y)^x \text{ mod } p$

## 2. Hệ mật Elgamal

### Thuật toán tạo khóa

Tóm lược: Mỗi đầu liên lạc tạo một khoá công khai và một khoá bí mật tương ứng :

(1) Tạo 1 số nguyên tố  $p$  lớn và một phần tử sinh  $\alpha$  của nhóm nhân  $Z_p^*$  của các số nguyên mod  $p$ .

(2) Chọn một số nguyên ngẫu nhiên  $a$ ,  $1 \leq a \leq p - 2$  và tính  $\alpha^a \text{ mod } p$ .

(3) Khoá công khai là bộ 3 số  $(p, \alpha, \alpha^a)$ , khoá bí mật là  $a$ .

### Thuật toán mã hóa công khai Elgamal

Tóm lược: B mã hóa một thông tin báo  $m$  để gửi cho A bản mã cần gửi.

**Mã hóa:** B phải thực hiện các bước sau:

(1) Nhận khoá công khai  $(p, \alpha, \alpha^a)$  của A.

(2) Biểu thị bản tin dưới dạng một số nguyên  $m$  trong dải  $\{0, 1, \dots, p - 1\}$ .

(3) Chọn số nguyên ngẫu nhiên  $k$ ,  $1 \leq k \leq p - 2$

(4) Tính  $\gamma = \alpha^k \text{ mod } p$  và  $\delta = m(\alpha^a)^k \text{ mod } p$ .

(5) Gửi bản mã  $c = (\gamma, \delta)$  cho A.

**Giải mã:** Để khôi phục bản rõ  $m$  từ  $c$ , A phải thực hiện các bước sau:

(1) Sử dụng khoá riêng  $a$  để tính  $\gamma^{p-1-a} \text{ mod } p$

$$\text{(Chú ý } \gamma^{p-1-a} = \gamma^{-a} = \gamma^{-ak})$$

(2) Khôi phục bản rõ bằng cách tính  $(\gamma^{-a})\delta \text{ mod } p$ .

### Chứng minh hoạt động giải mã:

Thuật toán trên cho phép A thu được bản rõ vì:

$$\gamma^{-a}\delta \equiv \alpha^{-ak} \cdot m \alpha^{ak} \equiv m \text{ mod } p$$

Ví dụ:

### **Tạo khoá.**

A chọn  $p = 2357$  và một phần tử sinh  $\alpha = 2$  của  $Z_{2357}^*$ . A chọn khoá bí mật  $a = 1751$  và tính  $\alpha^a \bmod p = 2^{1751} \bmod 2357 = 1185$ . Khoá công khai của A là  $(p=2357, \alpha=2, \alpha^a=1185)$

### **Mã hoá**

Để mã hoá bản tin  $m = 2035$ , B sẽ chọn một số nguyên ngẫu nhiên  $k = 1520$  và tính:

$$\gamma = 2^{1520} \bmod 2357 = 1430$$

$$\text{và } \delta = 2035 \cdot 1185^{1520} \bmod 2357 = 697$$

Sau đó B gửi  $c = (1430, 697)$  cho A

### **Giải mã**

Để giải mã A phải tính:

$$\gamma^{p-1-a} = 1430^{605} \bmod 2357 = 872$$

Sau đó khôi phục bản rõ m bằng cách tính:  $m = 872 \cdot 697 \bmod 2357 = 2035$ .

## **4.3.4 BÀI TOÁN XẾP BALO VÀ HỆ MẬT MERKLE – HELLMAN**

### **4.3.4.1 Định nghĩa dãy siêu tăng**

*Định nghĩa:* Dãy các số nguyên dương  $(a_1, a_2, \dots, a_n)$  được gọi là dãy siêu tăng nếu  $a_i > \sum a_j$  với  $\forall i, 2 \leq i \leq n$

### **4.3.4.2 Bài toán xếp balô**

Cho một đồng các gói có các trọng lượng khác nhau, liệu có thể xếp một số gói này vào ba lô để ba lô có một trọng lượng cho trước hay không. Về mặt hình thức ta có thể phát biểu bài toán trên như sau:

Cho tập các giá trị  $M_1, M_2, \dots, M_n$  và một tổng S. Hãy tính các giá trị  $b_i$  để:

$$S = b_1 M_1 + b_2 M_2 + \dots + b_n M_n$$

với  $b_i \in \{0, 1\}$

$b_i = 1$ : Có nghĩa là gói  $M_i$  được xếp vào ba lô.

$b_i = 0$ : Có nghĩa là gói  $M_i$  không được xếp vào ba lô.

### **4.3.4.3 Giải bài toán xếp balô trong trường hợp dãy siêu tăng**

Trong trường hợp  $M = \{M_1, M_2, \dots, M_n\}$  là một dãy siêu tăng thì việc tìm  $b = (b_1, b_2, \dots, b_n)$  tương đương như bài toán tìm biểu diễn nhị phân của một số S. Biểu diễn này sẽ tìm được sau tối đa là n bước.

*Thuật toán giải:*

VÀO: Dãy siêu tăng  $M = \{M_1, M_2, \dots, M_n\}$  và một số nguyên S là tổng của một tập con trong M

RA :  $(b_1, b_2, \dots, b_n)$  trong đó  $b_i \in \{0, 1\}$  sao cho:  $\sum b_i M_i = S$

(1)  $i \leftarrow n$

(2) Chừng nào  $i \geq 1$  hãy thực hiện

a. Nếu  $S \geq M_i$  thì:  $x_i \leftarrow 1$  và  $S \leftarrow S - M_i$  ngược lại:  $x_i \leftarrow 0$

b.  $i \leftarrow i - 1$

(3) Return (b)

Nếu  $M$  không phải là dãy siêu tăng thì lời giải của bài toán là một trong  $2^n$  phương án có thể. Đây là một bài toán khó giải nếu  $n$  lớn.

#### 4.3.4.4 Thuật toán mã công khai Merkle – Hellman

*Tóm lược:* B mã hoá bản tin  $m$  để gửi cho A bản mã cần phải giải mã.

**Mã hoá:** B phải thực hiện các bước sau:

- (1) Nhận khoá công khai của A:  $(a_1, a_2, \dots, a_n)$
- (2) Biểu thị bản tin  $m$  như một chuỗi nhị phân có độ dài  $n$   $m = m_1, m_2, \dots, m_n$
- (3) Tính số nguyên  $c = m_1 a_1 + m_2 a_2 + \dots + m_n a_n$
- (4) Gửi bản mã  $c$  cho A.

**Giải mã:** Để khôi phục bản rõ  $m$  từ  $c$ , A phải thực hiện các bước sau:

- (1) Tính  $d = W^{-1}c \bmod M$
- (2) Sử dụng thuật giải xếp ba lô trong trường hợp dãy siêu tăng để tìm các số nguyên  $r_1, r_2, \dots, r_n$ ,  $r_i \in \{0, 1\}$  sao cho:
$$d = r_1 M_1 + r_2 M_2 + \dots + r_n M_n$$
- (3) Các bit của bản rõ là  $m_i = r_{\pi(i)}$ ,  $i = 1, 2, \dots, n$

*Chứng minh:* Thuật toán trên cho phép A thu được bản rõ vì:

$$d \equiv W^{-1}c \equiv W^{-1} \sum m_i a_i \equiv \sum m_i M_{\pi(i)} \bmod M$$

Vì  $0 \leq d < M$ ,  $d = \sum m_i M_{\pi(i)} \bmod M$ , bởi vậy nghiệm của bài toán xếp ba lô ở bước (b) sẽ cho ta các bit của bản rõ sau khi sử dụng phép hoán vị  $\pi$

**Ví dụ:**

#### Tạo khoá.

Cho  $n = 6$ . A chọn dãy siêu tăng sau:  $(12, 17, 33, 74, 157, 316)$ ,  $M = 737$ ,  $W = 635$  thoả mãn  $(W, M) = 1$ .

Phép hoán vị  $\pi$  của  $\{1, 2, 3, 4, 5, 6\}$  được xác định như sau:

$$\pi(1) = 3, \pi(2) = 6, \pi(3) = 1, \pi(4) = 2, \pi(5) = 5, \pi(6) = 4$$

Khoá công khai của A là tập  $(319, 196, 250, 477, 200, 559)$

Khoá bí mật của A là  $(\pi, M, W(12, 17, 33, 74, 157, 316))$

#### Mã hoá

Để mã hoá bản tin  $m = 101101$ , B tính:

$$c = 319 + 250 + 477 + 559 = 1605$$

và gửi  $c$  cho A.

#### Giải mã

Để giải mã A phải tính:  $(W^{-1} = -224 = 513)$

$$d = W^{-1}c \bmod M = 136$$

và giải bài toán xếp ba lô trong trường hợp dãy siêu tăng sau:

$$136 = 12r_1 + 17r_2 + 33r_3 + 74r_4 + 157r_5 + 316r_6$$

và nhận được  $136 = 12 + 17 + 33 + 74$

Bởi vậy  $r_1 = r_2 = r_3 = r_4 = 1$   $r_5 = r_6 = 0$

Sử dụng phép hoán vị  $\pi$  sẽ tìm được các bit của bản rõ như sau:

$$\begin{aligned} m_1 = r_3 &= 1, \quad m_2 = r_6 = 0, \quad m_3 = r_1 = 1, \quad m_4 = r_2 = 1, \quad m_5 = r_5 = 0 \\ m_6 &= r_4 = 1 \end{aligned}$$

Vậy bản rõ  $m = 101101$ .

### 4.3.5 BÀI TOÁN MÃ SỬA SAI VÀ HỆ MẬT MC ELICE

Hệ mật McEliece sử dụng nguyên lý tương tự như hệ mật Merkle-Hellman. Phép giải mã là một trường hợp đặc biệt của bài toán NP đầy đủ nhưng nó được nguy trang giống như trường hợp chung của bài toán. Trong hệ thống này bài toán NP được áp dụng ở đây là bài toán giải mã cho một mã sửa sai (nhị phân) tuyến tính nói chung. Tuy nhiên, đối với nhiều lớp mã đặc biệt đều tồn tại các thuật toán giải mã với thời gian đa thức. Một trong những lớp mã này là mã Goppa, chúng được dùng làm cơ sở cho hệ mật McEliece.

#### 4.3.5.1 Định nghĩa 1.

Giả sử  $k, n$  là các số nguyên dương,  $k \leq n$ . Mã  $C[n, k]$  là một không gian  $k$  chiều của  $(Z_2)^n$  (không gian vectơ của tất cả các vectơ nhị phân  $n$  chiều).

Ma trận sinh của mã  $C[n, k]$  là ma trận nhị phân  $k \times n$ , các hàng của ma trận này tạo nên cơ sở của  $C$ .

Giả sử  $x, y \in (Z_2)^n$ , trong đó  $x = (x_1, \dots, x_n)$  và  $y = (y_1, \dots, y_n)$ . Ta xác định khoảng cách Hamming:  $d(x, y) = |\{i : 1 \leq i \leq n, x_i \neq y_i\}|$  tức là số các toạ độ mà ở đó  $x$  và  $y$  khác nhau.

Khoảng cách mã  $C$  được định nghĩa như sau:

$$d(C) = \min\{d(x, y) : x, y \in C, x \neq y\}$$

Mã  $[n, k]$  có khoảng cách  $d$  được ký hiệu là mã  $[n, k, d]$ .

Mã sửa sai được dùng để sửa các sai ngẫu nhiên xảy ra khi truyền số liệu (nhị phân) qua kênh có nhiễu. Điều đó được thực hiện như sau: Giả sử  $G$  là một ma trận sinh đối với mã  $[n, k, d]$ ,  $x$  là vectơ nhị phân  $k$  chiều cần truyền đi. Người gửi Alice sẽ mã hoá  $x$  thành một vectơ  $n$  chiều  $y = xG$  rồi truyền  $y$  qua kênh.

Giả sử Bob nhận được vectơ  $n$  chiều  $r$  không giống  $y$ , Bob sẽ giải mã  $r$  bằng chiến thuật giải mã "người láng giềng gần nhất". Theo chiến thuật này, Bob sẽ tìm thấy từ  $y'$  có khoảng cách tới  $r$  nhỏ nhất. Sau đó anh ta giải mã  $r$  thành  $y'$ , rồi xác định vectơ  $k$  chiều  $x'$  sao cho  $y' = x'G$ . Bob hy vọng  $y' = y$  và bởi vậy  $x' = x$  (tức là Bob tin rằng các sai số trên đường truyền đã được sửa).

Dễ dàng thấy rằng, nếu sai số trên đường truyền nhiều nhất là  $(d-1)/2$  thì trên thực tế chiến thuật này sẽ sửa được tất cả các sai.

Ta xét trên thực tế, thuật toán giải mã này được thực hiện như thế nào? Vì  $|C| = 2^k$  nên Bob so sánh  $r$  với mỗi từ mã anh ta phải kiểm tra  $2^k$  vectơ là một số lớn theo hàm

mũ so với k. Nói cách khác, thuật toán này không phải là thuật toán chạy trong thời gian đa thức.

Một biện pháp khác (tạo cơ sở cho nhiều thuật toán giải mã thực tế) dựa trên khái niệm về syndrom. Ma trận kiểm tra tính chẵn lẻ của mã  $C[n, k, d]$  (có ma trận sinh  $G$ ) là một ma trận nhị phân  $(n - k) \times n$  chiều (ký hiệu là  $H$ ). Các hàng của  $H$  sẽ tạo cơ sở cho các phần bù trực giao của  $C$  (ký hiệu là  $C^\perp$ ) và được gọi là mã đối ngẫu với  $C$ . Nói cách khác, các hàng của  $H$  là những vectơ độc lập tuyến tính, còn  $G H^\perp$  là một ma trận không cấp  $k \times (n - k)$ .

Cho vectơ  $r \in (\mathbb{Z}_2)^n$ , ta xác định syndrom của  $r$  là  $Hr^\perp$ . Syndrom  $Hr^\perp$  là một vectơ cột có  $(n - k)$  thành phần.

#### 4.3.5.2 Định lý 2

Giả sử  $C$  là một mã  $[n, k]$  có ma trận sinh  $G$  và ma trận kiểm tra tính chẵn lẻ  $H$ . Khi đó  $x \in (\mathbb{Z}_2)^n$  là một từ mã khi và chỉ khi  $Hx^T = [00\dots0]^T$ .

Hơn nữa nếu  $x \in C, e \in (\mathbb{Z}_2)^n$  và  $r = x + e$  thì  $Hx^T = He^T$ .

Ta coi  $e$  là vectơ sai xuất hiện trong quá trình truyền từ mã  $x$ . Khi đó  $r$  biểu diễn vectơ thu được. Định lý trên phát biểu rằng syndrom chỉ phụ thuộc vào các sai số mà không phụ thuộc vào từ mã cụ thể nào được truyền đi.

Điều này gợi ý tới một cách giải mã gọi là *giải mã theo syndrom*. Trước tiên tính  $s = Hr^\perp$  nếu  $s$  là một vectơ không, thì ta giải mã  $r$  thành  $r$ . Nếu không thì ta sẽ lần lượt tạo tất cả các vectơ sai có trọng số 1. Với mỗi vectơ này, ta tính  $He^T$ . Nếu có một vectơ  $e$  nào đó thoả mãn  $He^T = s$  thì ta giải mã  $r$  thành  $r - e$ . Ngược lại, lại tiếp tục tạo các vectơ sai có trọng số  $2, 3, \dots, [(d-1)/2]$ .

Theo thuật toán này, có thể giải mã cho một vectơ nhận được trong nhiều nhất bước.

$$1 + \binom{n}{1} + \dots + \binom{n}{\lceil (d-1)/2 \rceil}$$

Phương pháp này làm việc trên một mã tuyến tính bất kỳ. Đối với một số loại mã đặc biệt, thủ tục giải mã có thể nhanh chóng hơn. Tuy nhiên, trên thực tế, cách giải quyết này cho chiến thuật giải mã "người láng giềng gần nhất" vẫn là một bài toán NP đầy đủ. Như vậy, vẫn chưa có một thuật toán giải trong thời gian đa thức đã biết nào cho bài toán giải mã theo "người láng giềng gần nhất" tổng quát. (Khi số các sai số không bị giới hạn bởi  $[(d-1)/2]$ ).

Cũng giống như bài toán tổng tập con, có thể chỉ ra một trường hợp đặc biệt "dễ", sau đó ngụy trang sao cho nó giống với bài toán chung "khó". Để đưa ra lý thuyết sẽ rất dài dòng, bởi vậy ta sẽ chỉ tóm lược các kết quả ở đây. Một trường hợp khá dễ được McEliece đề nghị là dùng một mã trong lớp các mã Goppa. Trên thực tế, các mã này có một thuật toán giải mã hữu hiệu. Hơn nữa các, các mã này rất dễ tạo và có một số lượng lớn các mã Goppa tương đương có cùng tham số.

Các tham số của mã Goppa có dạng  $n = 2^m, d = 2t + 1$  và  $k = n - mt$ . Để áp dụng trong thực tế cho một hệ mật khoá công khai, McEliece đề nghị chọn  $m = 10$  và  $t = 50$ .

Điều này ứng với mã Goppa [1024, 524, 101]. Mỗi bản rõ là một véctơ nhị phân cấp 524 và mỗi bản mã là một véctơ nhị phân cấp 1024. Khoá công khai là một ma trận nhị phân cấp 524x1024 . Hình 3.3 sẽ mô tả hệ mật McEliece.

Cho  $G$  là một ma trận sinh của một mã Goppa  $C[n, k, d]$ , trong đó  $n = 2^m$ ,  $d = 2t + 1$  và  $k = n - mt$ . Cho  $S$  là một ma trận khả nghịch cấp  $k \times k$  trên  $Z_2$ . Giả sử  $P$  là một ma trận hoán vị cấp  $n \times n$ , ta đặt  $G' = SGP$ . Cho  $P = (Z_2)^2$ ,  $C = (Z_2)^n$  và ký hiệu:  $K = \{(G, S, P, G')\}$

Trong đó  $G, S, P$  được xây dựng như mô tả ở trên và được giữ kín, còn  $G'$  được công khai. Với  $K = (G, S, P, G')$ , ta định nghĩa:  $e_k(x, e) = xG' + e$ . Ở đây,  $e \in (Z_2)^n$  là một véctơ ngẫu nhiên có trọng số  $t$ .

Bob giải mã bản mã  $y \in (Z_2)^n$  theo các bước sau:

1. Tính  $y_1 = yP^{-1}$ .
2. Giải mã (Decode)  $y_1$ , Bob tìm được  $y_1 = x_1 + e_1$ ,  $x_1 \in C$ .
3. Tính  $x_0 \in (Z_2)^k$  sao cho  $x_0 G = x_1$ .

Hình 0-23. Hệ mật Mc Elice

Ví dụ: Ma trận:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

là ma trận sinh của mã Hamming [7, 4, 3]. Giả sử Bob chọn ma trận  $S$  và ma trận  $P$  như sau:

$$S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix} \quad \text{và} \quad P = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

Khi đó ma trận sinh công khai là:

$$G' = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Bây giờ giả sử Alice mã hoá bản rõ  $x = (1, 1, 0, 1)$  bằng cách dùng một vectơ sai ngẫu nhiên trọng số 1 có dạng:  $e = (0, 0, 0, 0, 1, 0, 0)$

Bản mã tính được là:

$$\begin{aligned} y &= x G' + e \\ &= (1, 1, 0, 1) \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} + (0, 0, 0, 0, 1, 0, 0) \\ &= (0, 1, 1, 0, 0, 1, 0) + (0, 0, 0, 0, 1, 0, 0) \\ &= (0, 1, 1, 0, 1, 1, 0) \end{aligned}$$

Khi Bob nhận được bản mã  $y$ , trước hết anh ta tính

$$y_1 = y P^{-1} = (0, 1, 1, 0, 1, 1, 0) \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} = (1, 0, 0, 0, 1, 1, 1)$$

Tiếp theo Bob giải mã  $y_1$  để nhận được  $x_1 = (1, 0, 0, 0, 1, 1, 0)$  (Cần để ý là  $e_1 \neq e$  do phép nhân với  $P^{-1}$ )

Sau đó anh ta lập  $x_0 = (1, 0, 0, 0)$  (bốn thành phần đầu tiên của  $x_1$ ).

Cuối cùng Bob tính:  $x = S^{-1} x_0 = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} (1, 0, 0, 0) = (1, 1, 0, 1)$

Đây chính là bản rõ mã Alice đã mã.

### 4.3.6 HỆ MẶT TRÊN ĐƯỜNG CONG ELLIPTIC

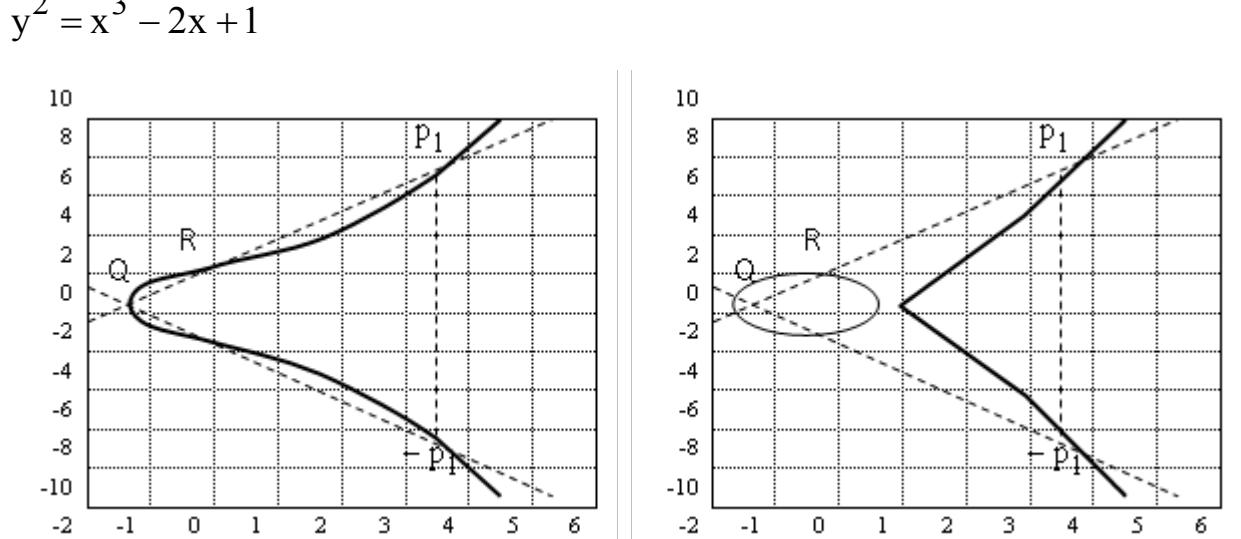
#### 4.3.6.1 Các đường cong Elliptic

Một đường cong Elliptic là một phương trình bậc 3 có dạng sau:

$$y^2 + axy + by = x^3 + cx^2 + dx + e$$

Trong đó a, b, c, d, e là các số thực.

Trên các đường cong E ta xác định một phép cộng đặc biệt với một điểm O được gọi là điểm vô cực. Nếu trên đường thẳng cắt đường cong E ở ba điểm thì tổng của chúng bằng điểm vô cực O (điểm O này có vai trò như phần tử đơn vị trong phép cộng này). Hình 3.1 sau mô tả các đường cong  $y^2 = x^3 + 2x + 5$  và  $y^2 = x^3 - 2x + 1$



Hình 0-24. Các đường cong  $y^2 = x^3 + 2x + 5$  và  $y^2 = x^3 - 2x + 1$

#### 4.3.6.2 Các đường cong Elliptic trên trường Galois

Một nhóm E trên trường Galois  $E_p(a, b)$  nhận được bằng cách tính  $x^3 + ax + b \bmod p$  với  $0 \leq x < p$ . Các hằng số a, b là các số nguyên không âm và nhỏ hơn số nguyên tố p và thỏa mãn điều kiện:  $4a^3 + 27b^2 \bmod p \neq 0$ . Với mỗi giá trị x ta cần xác định xem nó có là một thặng dư bậc hai hay không? Nếu x là thặng dư bậc hai

thì có 2 giá trị trong nhóm Elliptic. Nếu  $x$  không là thặng dư bậc 2 thì điểm này không nằm trong nhóm  $E_n(a, b)$ .

Ví dụ: (cấu trúc của một nhóm E)

Giả sử  $p = 23$ ,  $a = 1$  và  $b = 1$

Trước tiên ta kiểm tra lại:

$$= 4 + 27, \text{od}23 = 31 \bmod 23$$

Xét mỗi giá trị có thể  $x \in Z_{23}$ , tính  $x^3 + x + 1 \bmod 23$  và thử giải phương trình đối với  $y$ . Với giá trị  $x$  cho trước ta có thể kiểm tra xem liệu  $z = x^3 + x + 1 \bmod 23$  có phải là một thặng dư bình phương hay không bằng cách áp dụng tiêu chuẩn Euler (*Tiêu chuẩn Euler*: Giả sử  $p$  là số nguyên tố, khi đó  $x$  là một thặng dư bậc hai theo modulo  $p$  khi và chỉ khi:  $x^{(p-1)/2} \equiv 1 \bmod p$ ).

Ta đã có một công thức tường minh để tính các căn bậc hai của các thặng dư bình phương theo modulo  $p$  với các số nguyên tố  $p \equiv 3 \bmod 4$ . Áp dụng công thức này ta có các căn bậc hai của một thặng dư bình phương  $z$  là:

$$z^{(23+1)/4} = z^6 \bmod 23$$

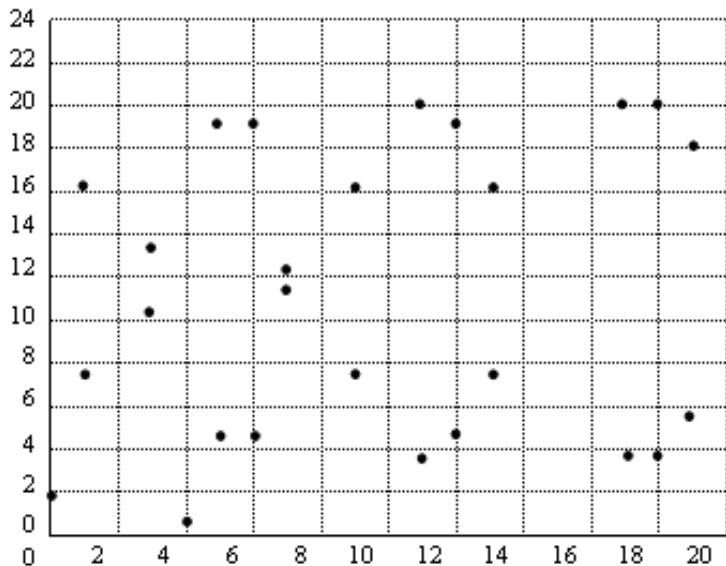
Kết quả của phép tính này được nêu trong bảng 3.2 dưới đây:

x	$x^3+x+1 \bmod 23$	Có trong $Q_{23}$	y
0	1	Có	1, 22
1	3	Có	7, 16
2	11	Không	
3	8	Có	10, 13
4	0	Không	
5	16	Có	4, 19
6	16	Có	4, 19
7	6	Có	11, 12
8	15	Không	
9	3	Có	7, 16
10	22	Không	
11	9	Có	3, 20
12	16	Có	4, 19
13	3	Có	7, 16
14	22	Không	
15	10	Không	
16	19	Không	
17	9	Có	3, 20
18	9	Có	3, 20
19	2	Có	5, 18
20	17	Không	
21	14	Không	
22	22	Không	

Bảng 0-2. Giá trị y tương ứng với x trên  $Z_{23}$

Nhóm Elliptic  $E_p(a, b) = E_{23}(1, 1)$  sẽ gồm các điểm sau:

$$E_{23}(1, 1) = \left\{ \begin{array}{ccccccc} (0, 1) & (0, 22) & (1, 7) & (1, 16) & (3, 10) & (3, 13) & (4, 0) \\ (5, 4) & (5, 19) & (6, 4) & (6, 19) & (7, 11) & (7, 12) & (9, 7) \\ (9, 16) & (11, 3) & (11, 20) & (12, 4) & (12, 19) & (13, 7) & (13, 16) \\ (17, 3) & (17, 20) & (18, 3) & (18, 20) & (19, 5) & (19, 18) & \end{array} \right\}$$



Hình 0-25. Nhóm  $E_{23}(1, 1)$

#### 4.3.6.3 Các phép toán cộng và nhân trên các nhóm E.

Giả sử  $P = (x_1, y_1)$ ,  $Q = (x_2, y_2)$  là các điểm trong nhóm  $E_p(a, b)$ ,  $O$  là điểm vô cực. Các quy tắc đối với phép cộng trên nhóm con  $E_n(a, b)$  như sau:

$$(1) P + O = O + P = P.$$

$$(2) \text{ Nếu } x_2 = x_1 \text{ và } y_2 = -y_1 \text{ tức là } P = (x_1, y_1) \text{ và } Q = (x_2, y_2) = (x_1, -y_1) = -P \text{ thì } P + Q = 0.$$

$$(3) \text{ Nếu } Q \neq -P \text{ thì tổng } P + Q = (x_3, y_3) \text{ được cho bởi:}$$

Trong đó:

$$\lambda \triangleq \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} & x_2 \neq x_1 \\ \frac{3x_1^2 + a}{2} & x_2 = x_1 \end{cases}$$

Ví dụ: Phép nhân trên nhóm  $E_n(a, b)$ .

Phép nhân trên nhóm  $E_p(a, b)$  thực hiện tương tự như phép lũy thừa modulo trong RSA.

Giả sử  $P = (3, 10) \in E_{23}(1, 1)$ , khi đó  $2P = (x_3, y_3)$  bằng:

$$2P = P + P = (x_1, y_1) + (x_1, y_1)$$

Vì  $P = Q$  và  $x_2 = x_1$  nên các giá trị  $\alpha$ ,  $x_3$  và  $y_3$  là:

$$x_3 = \lambda^2 - x_1 - x_2 \bmod p = 6^2 - 3 - 3 \bmod 23 = 30 \bmod 23 = 7$$

Bởi vậy  $2P = (x_2, y_2) = (7, 12)$ .

Phép nhân  $kP$  nhận được bằng cách thực hiện lặp  $k$  lần phép cộng.

$k$	$\lambda = \frac{y_2 - y_1}{x_2 - x_1}$ (nếu $x_2 \neq Q$ ) $\lambda = \frac{3x_1^2 + a}{2y_1}$ (nếu $P \neq Q$ )	$x_3$ $\lambda^2 - x_1 - x_2 \bmod 23$	$y_3$ $\lambda(x_1 - x_3) - y_1 \bmod 23$	$kP$ $(x_3, y_3)$
1				(3, 10)
2	6	7	12	(7, 12)
3	12	19	5	(19, 5)
4	4	17	3	(17, 3)
5	11	9	19	(9, 16)
6	1	12	4	(12, 4)
7	7	11	3	(11, 3)
8	2	13	16	(13, 16)
9	19	0	1	(0, 1)
10	3	6	4	(6, 4)
11	21	18	20	(18, 20)
12	16	5	4	(5, 4)
13	20	1	7	(1, 7)
14	13	4	0	(4, 0)
15	13	1	16	(1, 16)
16	20	5	19	(5, 19)
17	16	18	3	(18, 3)
18	21	6	19	(6, 19)
19	3	0	22	(0, 22)
20	19	13	7	(13, 7)
21	2	11	20	(11, 20)
22	7	12	19	(12, 19)
23	1	9	7	(9, 7)
24	11	17	20	(17, 20)
25	4	19	18	(19, 18)
26	12	7	11	(7, 11)
27	6	3	13	(3, 13)

Bảng 0-3. Bảng tính  $kP$

#### 4.3.6.4 Mật mã trên đường cong Elliptic.

Trong hệ mật này bản rõ  $M$  được mã hóa thành một điểm  $P_M$  trong tập hữu hạn các điểm của nhóm  $E_n(a, b)$ .

Trước hết ta phải chọn một điểm sinh  $G \in E_p(a, b)$  sao cho giá trị nhỏ nhất của  $n$  đảm bảo  $nG = 0$  phải là một số nguyên tố rất lớn. Nhóm  $E_p(a, b)$  và điểm sinh  $G$  được đưa ra công khai.

Mỗi người dùng chọn một khóa riêng  $n_A < n$  và tính khóa công khai  $P_A$  như sau:  $P_A = n_A G$ .

Để gửi thông báo  $P_M$  cho bên B, A chọn một số nguyên ngẫu nhiên  $k$  và tính cặp bản mã  $P_C$  bằng cách dùng khóa công khai  $P_R$  của B:

$$P_C = |(kG), (P_M + kP_R)|$$

Sau khi thu cặp điểm  $P_C$ , B sẽ nhận điểm đầu tiên  $(kG)$  với khóa riêng  $n_B$  của mình rồi cộng kết quả với điểm thứ hai trong cặp điểm  $P_C$  (Điểm  $(P_M + kP_R)$ );

$$(P_M + kP_R) - n_B(kG) = (P_M + kn_B G) - n_B(kG) = P_M$$

Đây chính là điểm tương ứng với bản rõ  $M$ . Chỉ có B mới có khóa riêng  $n_B$  và mới có thể tách  $n_B(kG)$  khỏi điểm thứ hai của  $P_C$  để thu thông tin về bản rõ  $P_M$ .

Ví dụ:

Xét đường cong E sau:  $y^2 = x^3 + ax + b \bmod p$   
 $y = x - x + 188 \bmod 751$

Nhóm E được tạo từ đường cong E ở trên là:

$$E_n(a, b) = E_{751}(-1, 188)$$

Cho điểm sinh  $G = (0, 376)$ . Khi đó phép nhân  $kG$  của  $G$  là ( $1 \leq k \leq 751$ ).

Nếu A muốn gửi cho B bản rõ  $m$  (được mã thành điểm bản rõ  $P_M$ )  $P_M = (443, 253) \in E_{751}(-1, 188)$  thì A phải dùng khóa công khai của B để mã hóa nó.

Giả sử khóa bí mật của B là  $n_B = 85$ , khi đó khóa công khai của B là:  
 $P_R = n_B G = 85(0, 376)$

$$P_R = (671, 558)$$

A chọn số ngẫu nhiên  $k = 113$  và dùng  $P_R$  để mã hóa  $P_M$  thành cặp điểm bản mã:

$$\begin{aligned} P_C &= |(kG); (P_M + kP_R)| \\ P_C &= [113.(0,376), (443,253) + (47,416)] \end{aligned}$$

Dựa vào PHƯƠNG PHÁP TỔNG QUAN VỀ AN TOÀN BẢO MẬT HỆ THỐNG, B sẽ dùng khóa riêng  $n_B = 85$  để tính  $P_M$  như sau: HTTT

$$\begin{aligned} (\text{PHƯƠNG PHÁP TỔNG QUAN VỀ AN TOÀN BẢO MẬT HỆ THỐNG}) \\ &= (217, 606) - [(47, 416)] \quad 1: \end{aligned}$$

$$\begin{aligned} \text{TỔNG QUAN VỀ AN TOÀN BẢO MẬT HỆ THỐNG} &= (217, 606) - (47, 416) \quad (\text{vì } P = (x_1, -y_1)) \\ &= (217, 606) + (47, 335) \quad (\text{vì } -416 \equiv 335 \pmod{751}) \\ &= (443, 253) \end{aligned}$$

Sau đó B ánh xạ điểm - điểm bản rõ  $P_M$  trở lại thông báo gốc M.

#### **4.3.6.5 Độ an toàn của hệ mật trên đường cong Elliptic.**

Sức mạnh ECC nằm ở sự khó khăn đối với thám mã khi phải xác định số ngẫu nhiên bí mật k từ kP và P. Phương pháp nhanh nhất để giải bài toán này là phương pháp phân tích S - Pollard. Để phá ECC độ phức tạp tính toán khi dùng phương pháp S – Pollard là  $3.8 \cdot 10^{10}$  MIPS - năm với kích thước khóa 150 bít (đây là số năm cần thiết với một hệ thống tính toán có tốc độ hàng triệu lệnh/giây). Để so sánh với phương pháp nhanh nhất phá RSA (là phương pháp sàng trường số để phân tích hợp số n thành tích của 2 số nguyên tố p và q) ta thấy rằng với n có kích thước 768 bít độ phức tạp tính toán là:  $2 \cdot 10^8$  MIPS - năm , với n có kích thước 1024 bít, độ phức tạp tính toán là  $3 \cdot 10^{11}$  năm .

Nếu độ dài khóa của RSA tăng lên tới 2048 bít thì cần  $3 \cdot 10^{20}$  MIPS - năm, trong khi đó với ECC chỉ cần độ dài khóa là 234 bít đã phải yêu cầu tới  $1,6 \cdot 10^{28}$  MIPS - năm.

#### **4.3.7 ƯU NHƯỢC ĐIỂM CỦA HỆ MẬT KHÓA CÔNG KHAI**

Vấn đề còn tồn đọng của hệ mật mã khoá đối xứng được giải quyết nhờ hệ mật mã khoá công khai. Chính ưu điểm này đã thu hút nhiều trí tuệ vào việc đề xuất, đánh giá các hệ mật mã công khai. Nhưng do bản thân các hệ mật mã khoá công khai đều dựa vào các giả thiết liên quan đến các bài toán khó nên đa số các hệ mật mã này đều có tốc độ mã dịch không nhanh lắm. Chính nhược điểm này làm cho các hệ mật mã khoá công khai khó được dùng một cách độc lập.

Một vấn đề nữa sinh khi sử dụng các hệ mật mã khoá công khai là việc xác thực mà trong mô hình hệ mật mã đối xứng không đặt ra. Do các khoá mã công khai được công bố một cách công khai trên mạng cho nên việc đảm bảo rằng “khoá được công bố có đúng là của đối tượng cần liên lạc hay không?” là một kẽ hở có thể bị lợi dụng. Vấn đề xác thực này được giải quyết cũng chính bằng các hệ mật mã khoá công khai. Nhiều thủ tục xác thực đã được nghiên cứu và sử dụng như Kerberos, X.509... Một ưu điểm nữa của các hệ mật mã khoá công khai là các ứng dụng của nó trong lĩnh vực chữ ký số, cùng với các kết quả về hàm băm, thủ tục ký để bảo đảm tính toàn vẹn của một văn bản được giải quyết.

## CHƯƠNG 5: QUẢN LÝ KHÓA

### 5.1. Khóa phiên và khóa trao đổi

Giả sử A và B là hai bên của một quan hệ liên lạc mật. Giả sử A có thể sử dụng một khóa kT để chuyển tin bí mật cho B. Một cách tổng quát, khóa kT này có thể là một khóa đối xứng chia sẻ chung giữa A và B nhưng cũng có thể là khóa công khai của B. Nếu A là một người dùng amateur (chưa chuyên nghiệp, “non tay”), có thể A sẽ nghĩ rằng dùng khóa kT này là đủ để mã hóa mọi thông tin muốn chuyển cho B. Thực tế làm như vậy là chưa an toàn. Trên thực tế, để đảm bảo an toàn, người dùng chuyên nghiệp A và B sẽ thường xuyên thay đổi khóa mã mật trong quá trình liên lạc. Mỗi phiên liên lạc lại sử dụng một khóa riêng, và vì thế sẽ gọi là khóa phiên. Hết phiên liên lạc, khóa phiên cũ sẽ hủy, vào phiên mới lại tạo khóa phiên mới. Việc tạo ra khóa phiên mới đương nhiên là dễ dàng, nhờ sử dụng khóa kT ban đầu. Ví dụ, để A có thể gửi văn bản m đến B với một khóa phiên tạo riêng cho phiên liên lạc này, có thể kết hợp cả hai việc (tạo khóa phiên và gửi tin mật) trong một bước như sau:  $A \rightarrow B: \{m\} ks || \{ks\} kT$

Như vậy khi nhận được, B sẽ lần lượt giải mã phần thứ hai để nhận được khóa phiên ks, rồi dùng nó để giải mã phần thứ nhất để thu được văn bản m. Việc tạo khóa phiên có căn cứ chính là để tránh khả năng kẻ tấn công có thể tiếp xúc được với quá nhiều văn bản mật được mã hóa bởi cùng một khóa mật. Điều đó xảy ra sẽ tạo cơ hội cho kẻ địch có ít nhiều khả năng tấn công khi hệ mật mã (đối xứng) không thật sự mạnh. Ngoài ra kẻ địch cũng có thể đoán biết nội dung thông tin bên trong ít nhiều thông qua thống kê, bởi vì nếu thông tin bản rõ lặp đi lặp lại (như trong một số trường hợp đặc biệt) thì bản mã cũng sẽ lặp đi lặp lại, sẽ bị để ý và dò đoán được.

Ví dụ 5.1: Nếu Alice và Bob chỉ gửi lặp đi lặp lại các thông điệp “BUY” hoặc “SELL”. Eve có thể tính trước {“BUY”} ZB và {“SELL”} ZB. Dựa vào các bản mã nghe trộm, Eve có thể dễ dàng đoán đợt lượng nội dung các thông điệp đơn giản như thế này.

## 5.2. Trao đổi khóa

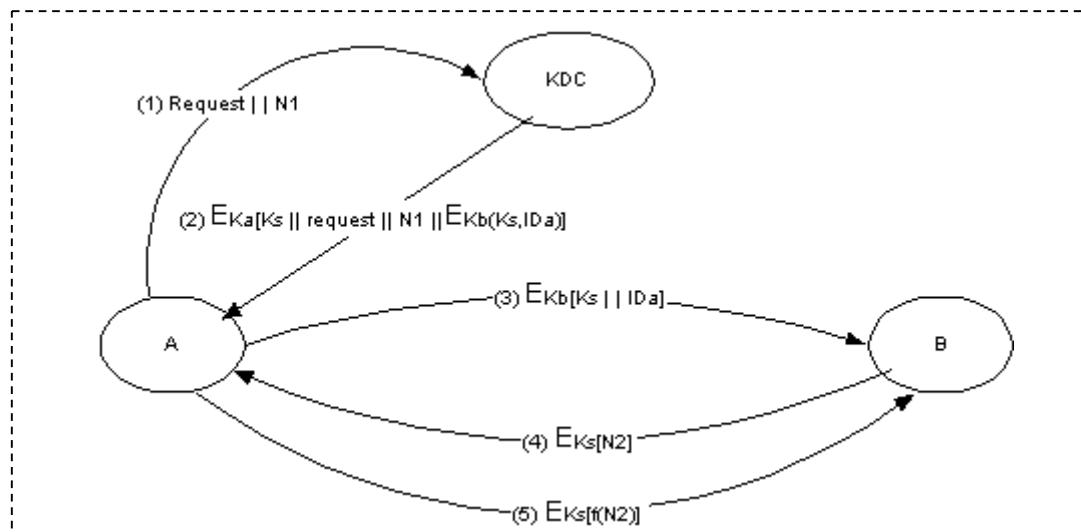
### 5.2.1 Quản lý và phân phối khóa bí mật.

Sức mạnh của hệ thống mật mã nằm ở khâu kỹ thuật phân phối khóa. Đó là quá trình phân phát khóa cho hai bên muốn trao đổi số liệu không cho phép người khác thu được khóa này. Trong mật mã khóa bí mật, hai bên truyền thông phải trao đổi khóa với nhau từ trước theo một kênh an toàn. Một số phương pháp phân phối khóa bí mật như phân phối khóa bằng thủ công, phân phối khóa bí mật có trung tâm phân phối khóa và phân phối khóa phân quyền.

#### a. Sơ đồ phân phối khóa phiên có trung tâm phân phối khóa.

Hình 2.5 là sơ đồ phân phối khóa bí mật trong đó mỗi người dùng được chia sẻ một khóa chủ duy nhất từ trước với trung tâm phân phối khóa (KDC).

Giả sử A muốn thiết lập một kết nối an toàn với B và đòi hỏi một khóa bí mật để mã hóa dữ liệu. A có khóa bí mật  $K_a$ , chỉ có A và KDC biết. Tương tự, B có khóa bí mật  $K_b$ , chỉ có B và KDC biết.



**Hình 5.1: Phân phối khóa bí mật có trung tâm phân phối khóa**

Các bước phân phối khóa bí mật  $K_s$  cho A và B như sau:

1. A gửi cho KDC một thông báo yêu cầu cung cấp một khóa bí mật cho một kết nối an toàn đến B. Thông báo bao gồm định danh của A, B và định danh duy nhất cho phiên giao dịch, chúng ta gọi là nonce. Giá trị nonce này có thể là tem thời gian, bộ đếm hoặc một số ngẫu nhiên, chúng khác nhau trong các đòi hỏi của A.

2. KDC gửi lại A một thông báo được mã hóa bởi khóa  $K_a$ . Như vậy chỉ có A là người có thể đọc được nội dung thông báo và A biết rằng nó xuất phát từ KDC. Thông báo bao gồm hai tham số dành cho A:

- Khoa bí mật  $K_s$  được dùng cho phiên liên lạc.

- Bản thông báo yêu cầu ban đầu và nonce để A biết  $K_s$  tương ứng với bản yêu cầu nào. Và thông báo còn có hai tham số dành cho B được mã hóa bởi khóa chủ  $K_b$  chỉ có KDC và B biết.:

- Khoa bí mật  $K_s$  được dùng cho phiên liên lạc.

- Định danh IDa của A.

3. A lưu giữ khoá bí mật Ks để dùng cho phiên liên lạc sắp tới với B và chuyển tới B giá trị  $E_{Kb}[Ks \parallel IDa]$  trong thông báo do KDC gửi tới. Bởi vì thông tin này được mã bởi Kb nên nó không thể bị lộ. Nay B biết được khoá phiên Ks, biết đối tượng cần giao dịch là A và biết đây là thông tin đi từ KDC (Bởi vì nó được mã dùng  $E_{Kb}$ ). Tại thời điểm này, khoá phiên đã được giao an toàn tới A và B và chúng có thể bắt đầu phiên liên lạc được bảo mật bằng kỹ thuật mật mã. Tuy nhiên sơ đồ còn bao gồm hai pha sau :

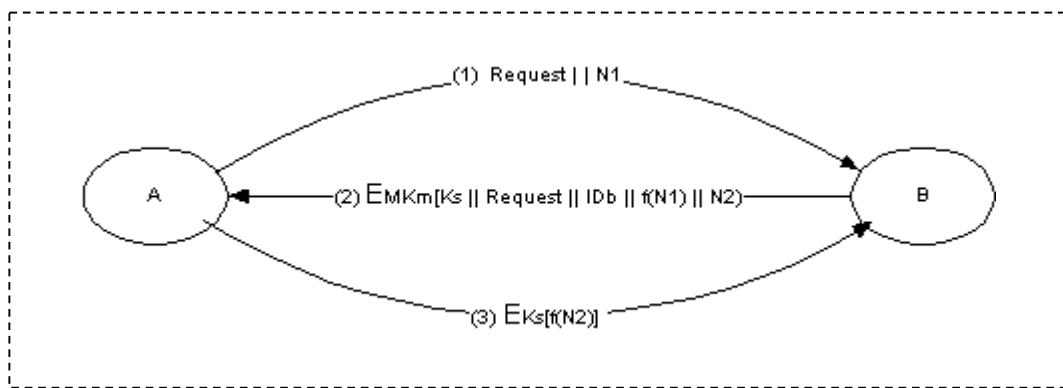
4. B gửi cho A một giá trịNonce N2 được mã bằng khoá phiên mới Ks.

5. A gửi lại B giá trị  $f(N2)$  được mã bằng khoá phiên Ks, với  $f(N2)$  là một hàm của N2.

Các bước này đảm bảo cho B rằng thông báo nguyên bản nó nhận ở bước 3 không bị dùng lại. Chú ý rằng việc phân phối khoá chỉ bao gồm pha 1 đến pha 3 nhưng các pha 4 và 5 có chức năng xác thực.

### b. Phân phối khoá phân quyền.

Việc dùng một trung tâm phân phối khoá đòi hỏi rằng KDC được tin cậy và được bảo vệ để không bị phá hoại. Đòi hỏi này có thể được loại bỏ nếu việc phân phối khoá là phân quyền đầy đủ. Mặc dù việc phân quyền không dùng cho các mạng lớn, nó thuận lợi đối với các mạng vừa và nhỏ. Hình 5.2 là sơ đồ phân phối khoá phân quyền.



**Hình 5.2: Phân phối khoá phân quyền**

Một sơ đồ phân quyền đòi hỏi mỗi hệ thống đầu cuối có thể truyền thông an toàn với tất cả các hệ thống đầu cuối nhằm mục đích phân phối khoá phiên. Với N đầu mối, cần  $[N(N-1)/2]$  khoá chủ.

Với hai đầu mối A và B, một khoá phiên được phân phối như sau:

1. A gửi cho B một thông báo đòi hỏi về khoá phiên, kèm theo một giá trị nonce N1.
2. B gửi lại A một thông báo được mã dùng khoá chủ chia sẻ. Thông báo bao gồm khoá phiên được chọn bởi B, định danh của B, giá trị  $f(N1)$  và N2.
3. A gửi trả lại B giá trị  $f(N2)$  được mã bằng khoá phiên mới.

Như vậy mặc dù mỗi đầu mối phải chứa N-1 khoá chủ, nhưng nhiều khoá phiên có thể được sinh ra. Vì thông báo dùng khoá chủ là ngắn, việc phân tích mã là khó khăn, trong khi đó khoá phiên được dùng trong thời gian hạn chế.

### c. Quản lý khoá phiên.

Khoá phiên càng được trao đổi thường xuyên thì chúng càng trở nên an toàn hơn vì khi đó kẻ địch có ít bản mã hơn đối với cùng một khoá phiên bất kỳ.

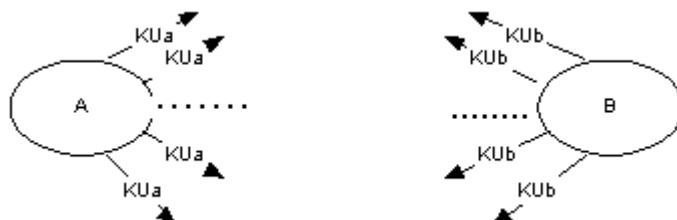
Phân phối khoá phiên làm trễ giai đoạn khởi đầu của trao đổi số liệu và làm tăng thêm lưu lượng mạng do đó cần cân đối để xác định thời gian tồn tại hợp lý của mỗi khoá phiên. Trong giao thức định hướng kết nối thì dùng cùng một khoá phiên trong suốt thời gian kết nối còn mở. Dùng khoá phiên mới cho phiên kết nối mới. Nếu một kết nối logic quá lâu thì phải thay đổi khoá phiên theo chu kỳ.

Đối với giao thức phi kết nối tại đó thời gian bắt đầu và kết thúc không rõ ràng tốt nhất là dùng một khoá phiên cho một khoảng thời gian cố định hay cho một số nhất định các giao dịch.

### 5.2.2 Những kỹ thuật phân phối khoá công khai.

Như chúng ta đã biết, trong mật mã khoá công khai, người dùng có thể công khai khoá mã để cho tất cả những người dùng khác mã hoá thông báo gửi cho họ. Tuy nhiên việc công bố khoá công khai cũng đòi hỏi những điều kiện nhất định. trong phần này chúng ta tìm hiểu các kỹ thuật phân phối khoá công khai.

#### a. Thông báo công khai khoá công khai



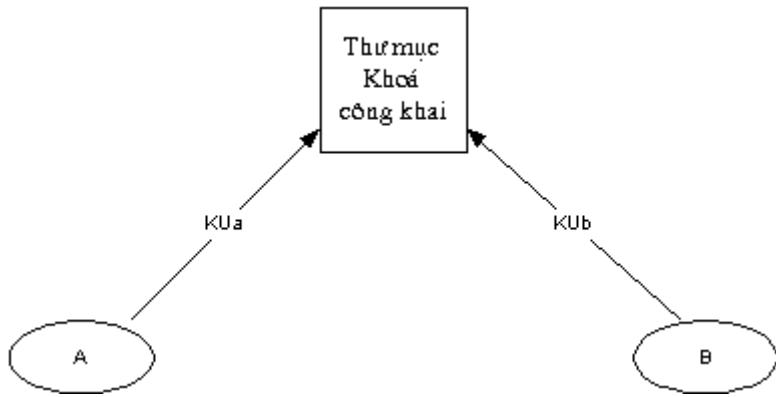
Hình 5.3: Thông báo công khai khoá công khai

Mỗi người dùng của hệ mật khoá công khai có thể gửi hoặc phát khoá công khai của anh ta cho mọi người dùng khác trên mạng.

Hạn chế cơ bản của kỹ thuật này là bất kỳ ai cũng có thể giả mạo thông báo công khai này. Bất kỳ ai cũng đóng giả một người dùng A khác và gửi khoá công khai đi với tư cách là A. Mọi thông tin gửi đến A đều bị nghe trộm thậm chí cả khoá xác thực cũng bị giả mạo.

#### b. Thư mục chung khoá công khai

Duy trì những thư mục động công khai trên mạng. Việc duy trì và phân phối thư mục động giao cho một thực thể hoặc tổ chức được tin cậy được gọi là người thẩm quyền (Authority).



**Hình 5.4: Thư mục chung khoá công khai**

Cơ chế hoạt động của kỹ thuật thư mục chung khoá công khai như sau:

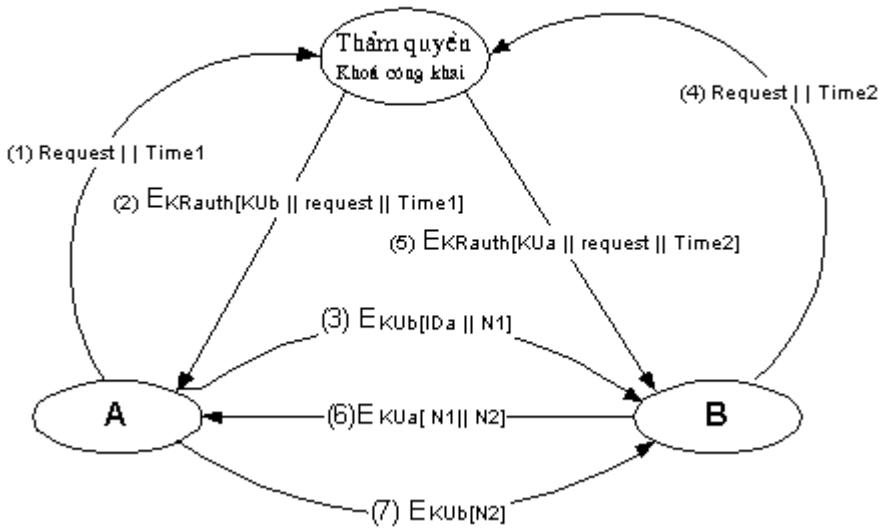
1. Người thẩm quyền duy trì một thư mục với một đầu vào (tên, khoá công khai) cho mỗi thành viên.
2. Mỗi thành viên đăng ký một khoá công khai với người thẩm quyền. Việc đăng ký có thể là đưa trực tiếp hoặc qua kênh truyền thông xác thực an toàn.
3. Một thành viên có thể thay thế khoá công khai hiện tại đã được dùng cho một số lớn dữ liệu hoặc do khoá bí mật tương ứng bị tổn thương.
4. Định kỳ, người thẩm quyền công bố toàn bộ thư mục hoặc cập nhật thư mục.
5. Các thành viên có thể truy nhập thư mục qua phương tiện điện tử. Để thực hiện điều này cần có một kênh truyền thông xác thực an toàn từ người thẩm quyền tới các thành viên.

Sơ đồ này an toàn hơn các thông báo khoá công khai trước kia nhưng vẫn bị tổn thương. Nếu kẻ địch đạt được hoặc tính được khoá bí mật của người chủ thư mục anh ta có thể trao đổi các khoá công khai và đóng giả làm bất kỳ thành viên nào để nghe trộm các thông báo gửi tới thành viên bất kỳ nào. Kẻ địch có thể thay đổi các bản ghi được giữ bởi người có thẩm quyền.

### c. Thẩm quyền khoá công khai

Kỹ thuật này có độ an toàn cao hơn kỹ thuật thư mục chung khoá công khai. Nhà thẩm quyền trung tâm duy trì thư mục động các khoá công khai của tất cả các thành viên. Mỗi thành viên đều biết một cách tin cậy khoá công khai của người có thẩm quyền nhưng chỉ có người có thẩm quyền biết khoá bí mật.

Khoá công khai được phân phối an toàn cho A và B nhờ xác thực dùng khoá công khai của người có thẩm quyền.



**Hình 5.5: Sơ đồ thẩm quyền khoá công khai**

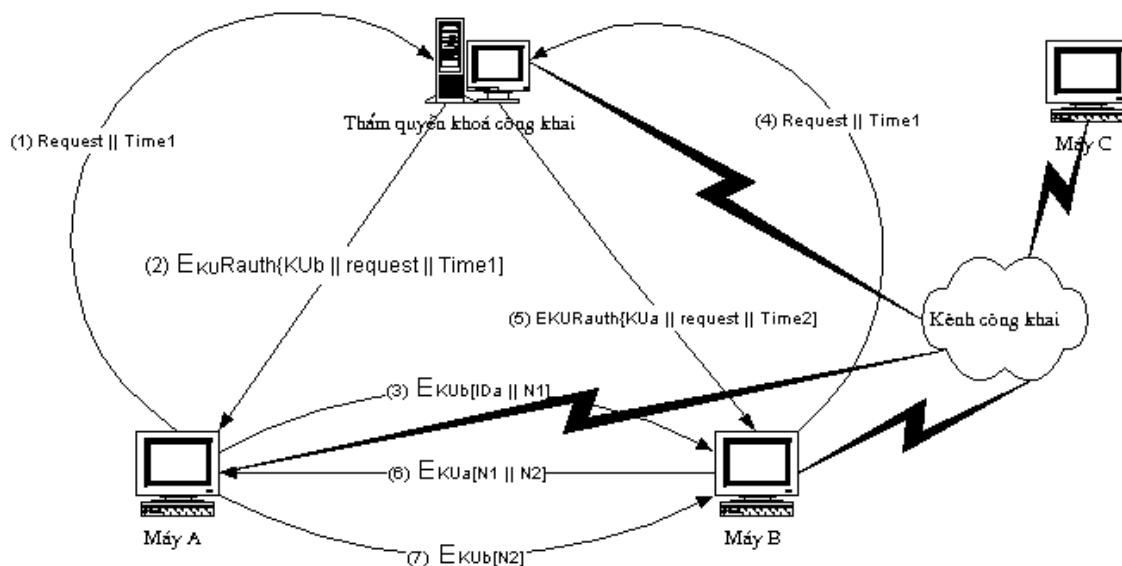
Cơ chế hoạt động như sau:

1. A gửi một thông báo có tem thời gian tới người thẩm quyền khoá công khai để đòi hỏi khoá công khai của B.
2. Người thẩm quyền gửi lại A một thông báo được mã dùng khoá riêng KRauth của thẩm quyền khoá công khai. Như vậy A có thể giải mã dùng khoá công khai của Người thẩm quyền. Thông báo bao gồm:
  - + Khoá công khai KUb của B mà A có thể dùng để mã thông báo cho B.
  - + Đòi hỏi nguyên bản của A để A xác định được đòi hỏi tương ứng trước đó và để xác nhận rằng đòi hỏi ban đầu không thay đổi trước khi được Người thẩm quyền chấp nhận.
  - + Tem thời gian nguyên bản, như vậy A có thể xác định rằng đây không phải là một thông báo cũ từ Người thẩm quyền chứa một khoá khác khoá công khai hiện tại của B.
3. A lưu giữ khoá công khai của B và dùng nó để mã một thông báo tới B chứa định danh ID<sub>A</sub> của A và một giá trị nonce (N1), được dùng để định danh duy nhất giao dịch này.
- 4.5. B thu được khoá công khai của A từ Người thẩm quyền theo cùng cách thức như A lấy được khoá công khai của B.
6. B gửi một thông báo tới A được mã bằng KUa và chứa giá trị N1 của A và N2 mới được sinh của B. Vì chỉ có B mới có thể giải mã thông báo (3) nên sự có mặt của N1 trong thông báo (6) đảm bảo với A rằng người đáp ứng là B.
7. A gửi trả lại N2 được mã bằng khoá công khai của B để đảm bảo rằng người đáp ứng là A.

Như vậy có 7 thông báo được đòi hỏi. Tuy nhiên 4 thông báo khởi tạo được dùng không thường xuyên vì cả A và B có thể lưu lại khoá công khai của người khác để dùng

lại. Định kỳ, một người dùng phải đòi hỏi một bản sao mới của các khoá công khai của các thành viên tương ứng.

*Ví dụ về cài đặt kỹ thuật thẩm quyền khoá công khai:*



**Hình 5.6: Mô hình cài đặt kỹ thuật thẩm quyền khoá công khai**

Giả sử trong một mạng máy tính có n máy cần trao đổi thông tin với nhau từng đôi một dùng kỹ thuật mật mã, ta quy ước gọi là máy khách hàng. Mỗi máy khách hàng cần nhận được khoá công khai của các máy khách hàng khác. Khi đó cần có thêm một máy tính đóng vai trò là Thẩm quyền khoá khai.

Người thiết kế hệ thống cần xây dựng bộ phần mềm có chức năng thực hiện kỹ thuật thẩm quyền khoá công khai. Bộ phần mềm này gồm có hai thành phần chính là thành phần chạy trên máy Thẩm quyền khoá công khai đóng vai trò người phân phối khoá, thành phần thứ hai được cài trên các máy khách hàng (n máy). Các máy này được kết nối với nhau qua kênh truyền công khai.

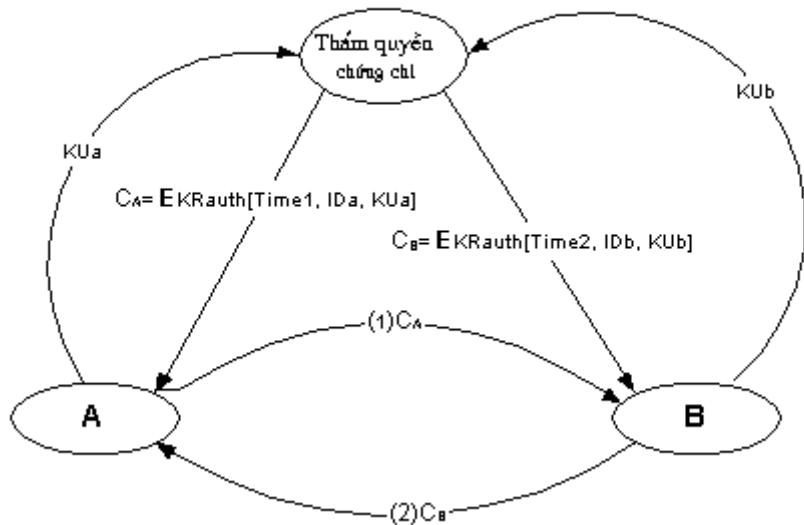
Các máy khách hàng đã có khoá công khai của Thẩm quyền khoá công khai được phân phối một cách tin cậy từ trước.

Giả sử hai máy khách hàng A và B bất kỳ muốn nhận được khoá công khai của nhau để thiết lập một kết nối an toàn. Mỗi máy có một định danh IDa và IDb. Để tiến hành phân phối khoá công khai các thành phần của bộ phần mềm của kỹ thuật Thẩm quyền khoá công khai trên 3 máy Thẩm quyền khoá công khai, máy A và máy B sẽ được kích hoạt và thực hiện theo đúng các bước như đã chỉ ra trong sơ đồ. Kết quả là các máy A và B đã xác thực lẫn nhau và cùng nhận được khoá công khai của nhau một cách tin cậy để thiết lập phiên kết nối an toàn. Tương tự như vậy đối với các cặp máy khách hàng khác.

#### *d. Chứng chỉ khoá công khai.*

Trong sơ đồ thẩm quyền khoá công khai còn có hạn chế khi người dùng thỉnh cầu người uỷ quyền cung cấp khoá công khai của tất cả các người dùng khác mà nó muốn

kết nối sẽ gây ra hiện tượng quá tải thắt cổ chai trong hệ thống. Sơ đồ trao đổi chứng chỉ khoá công khai tạo ra các chứng chỉ được dùng bởi các thành viên để trao đổi khoá không cần giao tiếp với người có thẩm quyền nhưng lại giống như khoá được trao trực tiếp từ người có thẩm quyền khoá công khai.



**Hình 5.7: Sơ đồ chứng chỉ khoá công khai**

Mỗi chứng chỉ chứa khoá công khai và thông tin liên quan được tạo ra bởi người có thẩm quyền và được giao cho thành viên với khoá bí mật phù hợp tương ứng. Thành viên này truyền thông tin khoá của mình đến thành viên khác chính là chứng chỉ của mình. Thành viên khác có thể kiểm tra xem chứng chỉ có được tạo ra bởi người có thẩm quyền không bằng cách dùng khoá công khai của người có thẩm quyền để giải mã chứng chỉ.

Đây thực ra là quá trình xác thực chữ ký của người có thẩm quyền vì ai cũng giải mã được thông tin trong chứng chỉ nhưng phải dùng khoá công khai của người có thẩm quyền.

Với thành viên A, người có thẩm quyền cung cấp chứng chỉ sau :

$C_A = E_{KRauth}[T, ID_A, KU_A]$ . Ở đó KRauth là khoá bí mật được dùng bởi Người có thẩm quyền. Sau đó A có thể chuyển chứng chỉ của mình tới thành viên bất kỳ khác, anh ta đọc và xác nhận như sau :

$$D_{KUauth}[C_A] = D_{KUauth}[E_{KRauth}[T, ID_A, KU_A]] = T, ID_A, KU_A .$$

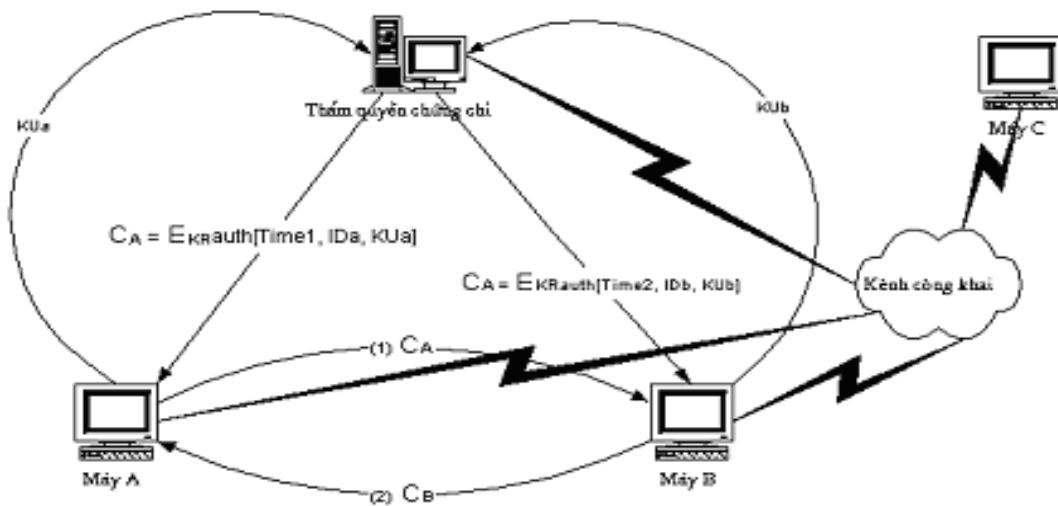
Người nhận dùng khoá công khai  $KUauth$  của Người có thẩm quyền để giải mã chứng chỉ. Vì chứng chỉ có thể đọc bằng cách dùng khoá công khai của Người có thẩm quyền, điều này xác nhận rằng chứng chỉ đến từ Người có thẩm quyền chứng chỉ. Các thành phần  $ID_A$ ,  $KU_A$  cung cấp cho người nhận tên và khoá công khai của người giữ chứng chỉ. Cuối cùng tem thời gian  $T$  xác nhận tính hợp lệ của thời gian lưu hành của chứng chỉ. Tem thời gian chống lại tình huống sau :

Một kẻ xâm nhập được khoá bí mật của A. Khi A sinh ra một cặp khoá mới và xin người thẩm quyền cấp một chứng chỉ mới, kẻ xâm nhập lại chứng chỉ cũ của A và gửi cho B. Nếu B mã thông báo dùng khoá công khai cũ thì kẻ xâm nhập có thể đọc các thông báo này.

Các đòi hỏi đối với sơ đồ trên là:

- Một thành viên bất kỳ có thể đọc một chứng chỉ để xác định tên và khoá công khai của người chủ của chứng chỉ.
- Thành viên bất kỳ có thể xác nhận rằng chứng chỉ là nguyên bản từ người thẩm quyền chứng chỉ và không bị giả.
- Chỉ người thẩm quyền chứng chỉ mới có thể đọc và cập nhật chứng chỉ.
- Thành viên bất kỳ có thể xác nhận thời gian lưu hành của chứng chỉ.

*Ví dụ về cài đặt kỹ thuật chứng chỉ khoá công khai:*



**Hình 5.8: Mô hình cài đặt sơ đồ chứng chỉ khoá công khai**

Giả sử trong một mạng máy tính có n máy cần trao đổi thông tin với nhau từng đôi một dùng kỹ thuật mật mã, ta quy ước gọi là máy khách hàng. Mỗi máy khách hàng cần nhận được khoá công khai của các máy khách hàng khác. Khi đó cần có thêm một máy tính đóng vai trò là Chứng chỉ khoá công khai.

Người thiết kế hệ thống cần xây dựng bộ phần mềm có chức năng thực hiện kỹ thuật Chứng chỉ khoá công khai. Bộ phần mềm này gồm có hai thành phần chính là thành phần chạy trên máy Chứng chỉ khoá công khai đóng vai trò người tạo ra chứng chỉ khoá công khai, thành phần thứ hai được cài trên các máy khách hàng (n máy). Các máy này được kết nối với nhau qua kênh truyền công khai.

Các máy khách hàng đã có khoá công khai của Chứng chỉ khoá công khai được phân phối một cách tin cậy từ trước.

Giả sử hai máy khách hàng A và B bất kỳ muốn nhận được khoá công khai của nhau để thiết lập một kết nối an toàn.

Mỗi máy có một định danh IDa và IDb. Để tiến hành phân phối khoá công khai các thành phần của bộ phần mềm của kỹ thuật Chứng chỉ khoá công khai trên 3 máy Chứng chỉ khoá công khai, máy A và máy B sẽ được kích hoạt và thực hiện theo đúng các bước như đã chỉ ra trong sơ đồ. Kết quả là các máy A và B đã xác thực lẫn nhau và cùng nhận được khoá công khai của nhau một cách tin cậy để thiết lập phiên kết nối an toàn. Tương tự như vậy đối với các cặp máy khách hàng khác.

#### e. Dùng mật mã khoá công khai phân phối khoá bí mật.

Trong phần trước chúng ta đã thấy được hạn chế của mật mã khoá bí mật trong việc phân phối khoá. Khi số đầu mối tăng lên, số lượng khoá chủ tăng theo hàm mũ của số lượng đầu mối. Một khía cạnh khác do thao tác với mật mã khoá công khai tồn tại nguyên tính toán nên người ta không dùng khoá công khai để mã hoá thông báo. Một trong những vai trò chính của mật mã khoá công khai là phân phối khoá bí mật dùng trong lập mã truyền thông.

##### \*. Phân phối khoá bí mật đơn giản.

Trong sơ đồ hình 2.19, hai thành viên A và B muốn truyền thông với nhau dùng mật mã khoá bí mật. A muốn B gửi cho A một khoá phiên  $K_s$  bằng cách dùng mật mã khoá công khai.



**Hình 5.9: Sơ đồ dùng mật mã khoá công khai để trao đổi khoá phiên**

Thủ tục trao đổi khoá như sau:

- A sinh ra một cặp khoá công khai/bí mật  $\{K_{UA}, K_{RA}\}$  và truyền thông báo (1) tới B bao gồm  $K_{UA}$  và định danh  $ID_A$  của A.
- B sinh một khoá bí mật  $K_s$ , mã  $K_s$  bằng khoá công khai của A và gửi cho B bản mã  $E_{K_{UA}}[K_s]$ .
- A tính  $D_{K_{RA}}[E_{K_{UA}}[K_s]]$  để khôi phục khoá bí mật  $K_s$ . Vì chỉ có A là có thể giải mã thông báo nên chỉ có A và B biết khoá  $K_s$ .
- A huỷ  $K_{UA}$  và  $K_{RA}$  và B huỷ  $K_{UA}$ .

Bây giờ A và B có thể truyền thông an toàn dùng khoá phiên  $K_s$ . Kết thúc phiên liên lạc, cả A và B huỷ  $K_s$ .

Cách phân phối này đơn giản, không có thông tin nào tồn tại trước và sau khi truyền thông. Chính vì vậy rủi ro về dàn xếp khoá là nhỏ. Tuy nhiên, cách phân phối này dễ dàng bị tấn công xen vào giữa thực hiện thành công.

Nếu có một kẻ tấn công E điều khiển được kênh truyền thông ở giữa thì E có thể làm tổn thương phiên truyền thông bằng cách sau:

- A sinh ra một cặp khoá công khai/bí mật  $\{K_{Ua}, K_{Ra}\}$  và truyền một thông báo tới B bao gồm  $K_{Ua}$  và định danh  $ID_A$  của A.

- E chặn thông báo, tạo cặp khoá công khai/bí mật  $\{K_{Ue}, K_{Re}\}$  của nó và truyền  $K_{Ue} \parallel ID_A$  cho B.

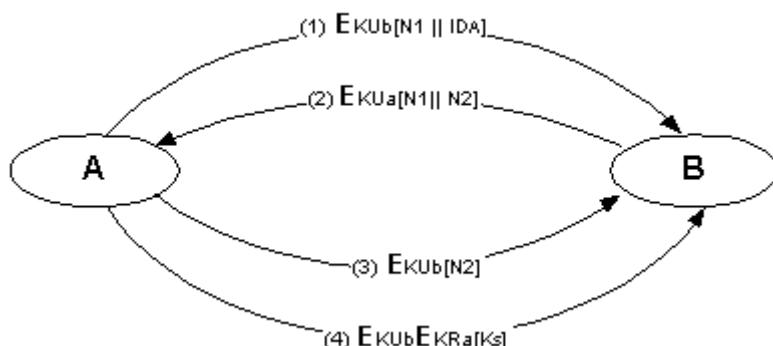
- B sinh một khoá bí mật  $K_s$ , và gửi  $E_{K_{Ue}}[K_s]$  cho A.

- E chặn thông báo, thu được  $K_s$  bằng cách tính  $D_{K_{Re}}[E_{K_{Ue}}[K_s]]$ .

- E truyền  $E_{K_{Ua}}[K_s]$  cho A.

Kết quả là cả A và B đều biết  $K_s$  và không biết rằng  $K_s$  đã bị E phát hiện. Nay giờ A và B có thể trao đổi thông báo dùng  $K_s$ . Bằng cách nghe trộm, E có thể giải mã tất cả các thông báo và cả A và B đều không biết điều này. Như vậy giao thức này đơn giản và chỉ có thể được dùng trong môi trường mà mối đe dọa là nghe trộm.

\* Phân phối khoá bí mật có bí mật và xác thực.



**Hình 5.10 : Dùng khoá công khai phân phối khoá bí mật**

Trong sơ đồ hình 2.20 hai thành viên A và B muốn truyền thông với nhau dùng mật mã khoá bí mật. A muốn B gửi cho A một khoá phiên  $K_s$  một cách bí mật và xác thực bằng cách dùng mật mã khoá công khai. Giả thiết A và B đã trao đổi với nhau về khoá công khai trước đó. Các bước được tiến hành như sau:

1. A dùng khoá công khai của B là  $E_{Kub}$  lập mã thông báo có định danh  $ID_A$  và nonce  $N_1$  là giá trị ngẫu nhiên không lặp lại do A sinh ra và gửi bản mã  $E_{Kub}[N_1 \parallel ID_A]$  cho B.

2. B gửi cho A bản mã  $K_{Ua}[N_1 \parallel N_2]$ , trong đó có giá trị nonce  $N_2$  của B. Vì chỉ có B mới có thể giải mã thông báo (1), nên sự có mặt của  $N_1$  trong thông báo (2) đảm bảo với A rằng người đáp ứng là B.

3. A gửi trả lại  $N_2$  được mã bằng khoá công khai của B để đảm bảo với B rằng người đang đáp ứng là A.

4. A chọn một khoá bí mật  $K_s$  và gửi cho B:  $E_{Kub}E_{Kra}[K_s]$ .

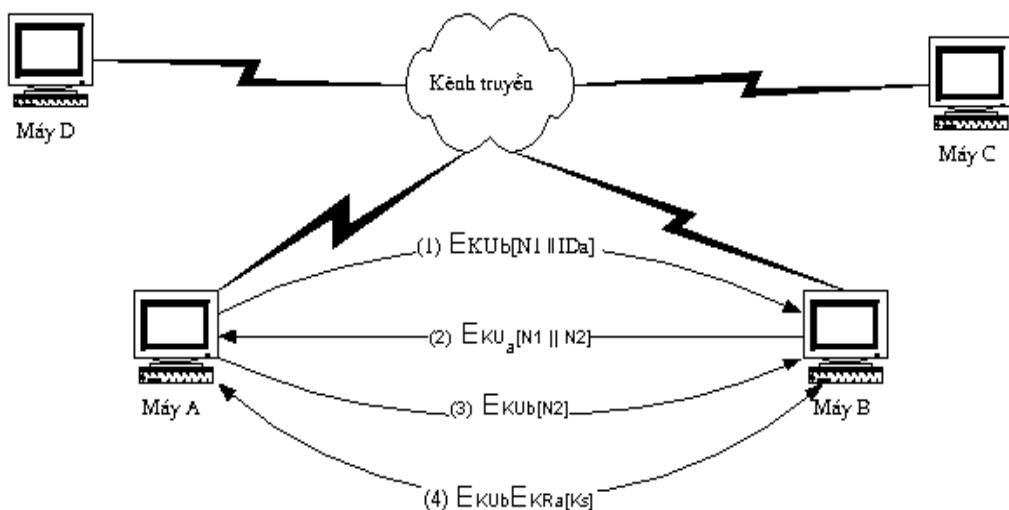
Trong đó  $K_{Ra}$  là khoá bí mật của A và  $K_s$  là khoá chung giữa A và B.

Sơ đồ này đảm bảo cả bí mật và xác thực khoá bí mật  $K_s$ .

Vì mã băm khoá công khai của B nên chỉ có B mới có thể đọc và mã băm khoá bí mật của A nên đảm bảo rằng chỉ có A là người gửi nó.

5. B tính  $D_{KUa}[D_{KRb}[M]]$  để khôi phục khoá bí mật.

*Ví dụ về cài đặt sơ đồ dùng khoá công khai phân phối khoá bí mật :*



**Hình 5.11: Mô hình cài đặt sơ đồ dùng khoá công khai phân phối khoá bí mật**

Giả sử trong một mạng máy tính có n máy cần trao đổi thông tin với nhau từng đôi một dùng một mật mã khoá bí mật. Người thiết kế hệ thống cần xây dựng bộ phần mềm có chức năng thực hiện sơ đồ dùng mật mã khoá công khai phân phối khoá bí mật và được cài đặt trên tất cả các máy. Các máy này được kết nối với nhau qua kênh truyền công khai.

Giả sử hai máy A và B bất kỳ muốn có một khoá phiên chung để thiết lập một kết nối an toàn với nhau (thường dùng cho kỹ thuật mã khóa như DES, IDEA, ...).

Trước đó A và B đã có khoá công khai của nhau. Mỗi máy có một định danh IDa và IDb. Để tiến hành thỏa thuận khoá phiên bộ phần mềm của sơ đồ phân phối khoá trên máy A và máy B sẽ được kích hoạt và thực hiện theo đúng các bước như đã chỉ ra trong sơ đồ. Kết quả là các máy A và B đã xác thực lẫn nhau và cùng được phân phối một khoá phiên để thiết lập phiên kết nối an toàn. Tương tự như vậy đối với các cặp máy khác.

\* Sơ đồ phân phối lai ghép.

Trong sơ đồ này có một trung tâm KDC phân phối khoá chủ bí mật với mỗi người dùng. Các khoá chủ được dùng để phân phối khoá phiên. Các khoá công khai được dùng để phân phối các khoá chủ. Sơ đồ này là sơ đồ ba mức phân cấp và khoá công khai chỉ được dùng để cập nhật các khoá chủ giữa người dùng và trung tâm. Điều này làm tăng năng suất hoạt động cho các sơ đồ đòi hỏi thay đổi khoá phiên thường xuyên. Sơ đồ lai ghép dễ dàng lắp ghép vào sơ đồ trung tâm phân phối khoá đang tồn tại. Do đó có ít trực trặc nhất và ít thay đổi phần mềm. Nó tạo cho tính tương thích ngược trở lại tăng lên. Sơ đồ này phù hợp cho các ứng dụng cần thay đổi khoá phiên thường xuyên. Việc phân phối

khoá phiên bằng mật mã khoá công khai sẽ làm giảm hiệu năng hệ thống bởi vì chúng đòi hỏi gánh nặng tính toán cho mã và giải mã.

#### \* Thuật toán trao đổi khoá Diffie-Hellman.

Trong các sơ đồ phân phối khoá phiên dùng mật mã khoá công khai ở trên, khoá phiên được tạo bởi một bên tham gia truyền thông sau đó được mã bởi khoá công khai để truyền cho bên kia. Điều này có thể dẫn đến việc lộ khoá tại bên sinh khoá hoặc trên đường truyền.

Trong thuật toán Diffie-Hellman hai bên truyền thông cùng cung cấp cho nhau các thông tin bí mật để tạo ra một khoá phiên chung. Đây là thuật toán đầu tiên của mật mã khoá công khai được ra đời vào năm 1976 được gọi là trao đổi khoá Diffie-Hellman. Mục đích của thuật toán này là giúp hai người dùng trao đổi khoá một cách an toàn để dùng lập mã các thông báo sau này. Thuật toán này có độ an toàn mật mã dựa trên bài toán tính Logarit rời rạc trên trường hữu hạn.

Dưới đây là thuật toán trao đổi khoá Diffie-Hellman.

Trong mạng truyền thông công cộng, khi một người dùng A muốn truyền tin mật cho một người dùng B thì họ thực hiện giao thức Diffie-Hellman để trao đổi một khoá K. Khoá K là bí mật (chỉ A và B biết), nhưng các thành phần để tạo nên K là công khai.

Giao thức được thực hiện như sau:

Giả sử đã chọn trước một số nguyên tố p và một phần tử nguyên thuỷ  $\alpha$  của  $Z_p$ . Các bước của giao thức là:

- A chọn ngẫu nhiên  $X_a$ ,  $0 \leq X_a \leq p - 2$ , giữ kín  $X_a$ , tính  $Y_a = \alpha^{X_a} \text{ mod } p$  và gửi  $Y_a$  cho B.
- B chọn ngẫu nhiên  $X_b$ ,  $0 \leq X_b \leq p - 2$ , giữ kín  $X_b$ , tính  $Y_b = \alpha^{X_b} \text{ mod } p$  và gửi  $Y_b$  cho A.
- Cả A và B đều tính được khoá chung  $K = \alpha^{X_a X_b} \text{ mod } p$ :
  - + A tính  $K = Y_b^{X_a} \text{ mod } p$
  - + B tính  $K = Y_a^{X_b} \text{ mod } p$

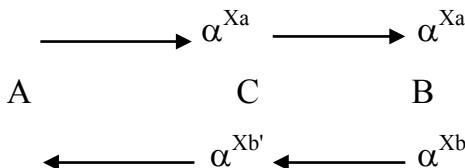
Kẻ địch muốn có khoá K phải tính được  $X_a$  hoặc  $X_b$ , do đó phải đối mặt với bài toán tính Logarit rời rạc trên mod p. Đây là sơ đồ nguyên thuỷ nên nó có tính chất là một khi  $Y_a$ ,  $Y_b$  đã xác định thì K không thay đổi.

Thuật toán Diffie-Hellman có hai đặc trưng chính sau:

- Các khóa bí mật chỉ được tạo khi cần thiết. Không cần phải chứa các khóa bí mật trong một khoảng thời gian dài.
  - Việc thỏa thuận dựa trên các tham số chung.
- Tuy nhiên Thuật toán Diffie-Hellman có một số điểm yếu sau:
- Nó không cung cấp thông tin bất kỳ về các định danh của các bên.

- Nó an toàn đối với việc tấn công thụ động nghĩa là một người thứ ba biết  $Y_a$ ,  $Y_b$  sẽ không tính được K. Tuy nhiên giao thức là không an toàn đối với việc tấn công chủ động bằng cách đánh tráo giữa đường hay còn gọi là kiểu tấn công “Người đàn ông ở giữa”. Trong đó một người C thứ ba mạo danh là B trong khi truyền thông với A và mạo danh A trong khi truyền thông với B. Cả A và B đều thỏa thuận khóa với C, sau đó C có thể nghe các thông tin được trao đổi giữa A và B.

Kiểu tấn công “Người đàn ông ở giữa” tiến hành như sau:



- B gửi khóa công khai  $Y_b = \alpha^{Xb}$  trong một thông báo tới A.

- C chặn thông báo này. C giữ lại khóa công khai  $Y_b$  của B và gửi một thông báo tới A với định danh người dùng là B nhưng chứa khóa công khai  $Y_c = \alpha^{Xb'}$  của C. Thông báo này được gửi trong cách thức như nó được gửi từ hệ thống của B. A nhận thông báo của C và chứa khóa công khai  $Y_c$  của C với định danh người dùng của B.

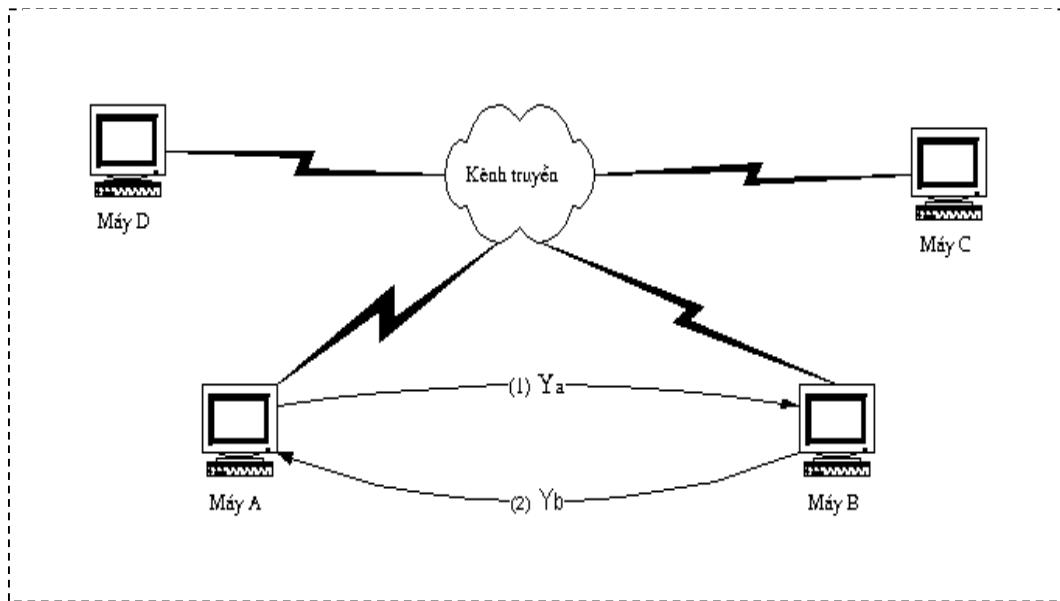
Tương tự C gửi một thông báo tới B với khóa công khai của C, nhưng với định danh người dùng là A.

- B tính một khóa bí mật K1 dựa trên khóa bí mật của B và  $Y_c$ . A tính một khóa bí mật K2 dựa trên khóa bí mật của A và  $Y_c$ . C tính K1 dùng khóa bí mật  $X_e$  của C và  $Y_b$  và tính khóa K2 dùng  $X_e$  và  $Y_a$ .

- Từ bây giờ C có thể chuyển tiếp các thông báo từ A tới B và từ B tới A, thay đổi một cách phù hợp nội dung bản mã bằng cách dùng các khóa chung K1, K2. Trong khi cả A và B đều không biết rằng chúng đang truyền thông với C.

**Chú ý:** Để chống lại tấn công trên thì A phải khẳng định  $Y_b$  là khoá công khai của B và B phải khẳng định là  $Y_a$  là khoá công khai của A.

Ví dụ về cài đặt thuật toán phân phối khoá Diffie – Hellman: Hình 2.22 là ví dụ về mô hình cài đặt thuật toán trao đổi khoá Diffie – Hellman. Giả sử trong một mạng máy tính có n máy cần trao đổi thông tin với nhau từng đôi một dùng mật mã khoá bí mật. Người thiết kế hệ thống cần xây dựng bộ phần mềm có chức năng thực hiện thuật toán phân phối khoá Diffie – Hellman và được cài đặt trên tất cả các máy. Các máy này được kết nối với nhau qua kênh truyền công khai.



**Hình 5.12 : Mô hình cài đặt thuật toán trao đổi khoá Diffie – Hellman.**

Giả sử hai máy A và B bất kỳ muốn có một khoá phiên chung để thiết lập một kết nối an toàn với nhau (thường dùng cho kỹ thuật mã khối như DES, IDEA, ...).

Trước đó A và B đã có khoá công khai của nhau. Mỗi máy có một định danh IDa và IDb. Để tiến hành trao đổi khoá phiên bộ phần mềm của thuật toán trao đổi khoá Diffie - Hellman trên máy A và máy B sẽ được kích hoạt và thực hiện theo đúng các bước của thuật toán. Kết quả là các máy A và B đã trao đổi một khoá phiên để thiết lập phiên kết nối an toàn. Tương tự như vậy đối với các cặp máy khác.

### 5.3. Chữ ký số

#### 5.3.1. Khái niệm.

Chữ ký số là một bộ phận dữ liệu không thể giả mạo mà nó xác nhận một người đã viết ra hoặc đồng ý với một tài liệu mà chữ ký đó được gắn vào. Chữ ký số sử dụng công nghệ mã hóa trên nền tảng khoá công khai để bảo đảm tính đúng đắn và toàn vẹn về nội dung của một dữ liệu hoặc một thông điệp điện tử và các yêu cầu khác đối với các mục đích của một chữ ký.

Chữ ký số có các tính chất sau:

Xác thực: người nhận kiểm tra được rằng người gửi đã ký vào tài liệu hay không.

Không thể giả mạo được: chữ ký số xác nhận với người nhận tài liệu là không ai khác ngoài người ký đã ký tài liệu đó.

Không thể dùng lại được: chữ ký số có kèm theo nội dung và thời gian gắn trên tài liệu nên không thể chuyển chữ ký từ tài liệu này sang tài liệu khác được.

Đảm bảo sự toàn vẹn và đúng đắn của tài liệu: tài liệu đã được ký là không thể giả mạo, sửa đổi hoặc thêm bớt.

Không thể chối bỏ: chữ ký và tài liệu là chứng cứ về mặt pháp lý và có khả năng xác thực sự đồng nhất giữa người ký và tài liệu được ký. Do đó, người gửi không thể phủ nhận rằng họ đã ký tài liệu đó.

Chữ ký số có ba ứng dụng chủ yếu trong đảm bảo an toàn thông tin trên mạng, đó là: xác thực, toàn vẹn và không thể chối bỏ.

Xác thực: khi người nhận thực hiện việc kiểm tra chữ ký trên một thông báo nhận được và cho kết quả chữ ký đúng, hợp lệ, người nhận có thể xác thực người ký, vì chỉ người ký mới sở hữu khóa bí mật (khóa riêng) để tạo chữ ký, những người khác không thể giả mạo để tạo ra chữ ký như thế.

Toàn vẹn: Việc kiểm tra chữ ký thành công còn cho phép người nhận tin tưởng rằng thông báo được ký đã không bị thay đổi trên đường truyền. Bất kỳ một thay đổi nào đó đối với thông báo cũng sẽ thể hiện bởi kết quả kiểm tra chữ ký là không đúng.

Không thể chối bỏ: Do chữ ký cho phép xác thực người ký trên một tài liệu được ký, người ký không thể chối bỏ việc đã ký lên tài liệu đó.

Chữ ký số được tạo ra trên các tài liệu phải dựa trên một lược đồ chữ ký số nhất định. Một lược đồ chữ ký số là một bộ năm phần tử ( $P, A, K, S, V$ ) thỏa mãn các điều kiện dưới đây:

- $P$ : là một tập hữu hạn các thông điệp có thể
- $A$ : là tập hữu hạn các chữ ký có thể
- $K$ : không gian khóa, là tập hữu hạn các khóa có thể
- Với mỗi  $k \in K$ , có một thuật toán ký  $\text{sig}_k \in S$  và thuật toán kiểm tra tương ứng  $\text{ver}_k \in V$

$$\text{sig}_k : P \rightarrow A: \text{sig}_k(x) = y; \text{ trong đó } x \in P \text{ và } y \in A$$

$$\text{ver}_k : P \times A \rightarrow \{\text{true}, \text{false}\} \text{ sao cho } \forall x \in P \text{ và } \forall y \in A \text{ thì}$$

$$\text{ver}(x, y) = \text{True} \text{ nếu } y = \text{sig}(x)$$

$$\text{ver}(x, y) = \text{False} \text{ nếu } y \neq \text{sig}(x)$$

Chữ ký số đối với một văn bản là một tập dữ liệu, đảm bảo tính nguồn gốc và tính toàn vẹn của văn bản đó.

+ Tính nguồn gốc: Người ký đồng ý với nội dung văn bản.

+ Tính toàn vẹn: Nội dung văn bản không bị thay đổi.

Như vậy chữ ký số là thông tin được mã hoá bằng khoá riêng của người gửi, được gửi kèm theo văn bản nhằm đảm bảo cho người nhận định danh, xác thực đúng nguồn gốc và tính toàn vẹn của tài liệu nhận được. Chữ ký số thể hiện văn bản gửi đi đã được ký bởi chính người sở hữu một Khoá riêng tương ứng với một chứng chỉ số.

Các chữ ký số được chia làm hai loại : chữ ký số trực tiếp và chữ ký số phân xử.

### 5.3.2. Chữ ký số trực tiếp.

Chữ ký số trực tiếp chỉ liên quan đến các bên truyền thông (Bên gửi và bên nhận). Bên gửi ký chữ ký số vào thông báo và bên nhận trực tiếp kiểm tra chữ ký số của bên gửi.

Mô hình thích hợp nhất cho cơ chế này là mô hình chữ ký số dùng mật mã khoá công khai. Chỉ có bên gửi có thể ký chữ ký số còn bất kỳ bên nào cũng có thể kiểm tra chữ ký số của bên gửi. Khi có tranh chấp xảy ra bên thứ ba cần phải biết nội dung thông báo và chữ ký số trên nó. Có hai tình huống xảy ra. Một là ký chữ ký số cho thông báo

rồi mới lập mã thông báo. Hai là chữ ký số được tính trên bản mã của thông báo. Khi đó bên thứ ba cũng cần cả khóa giải mã để đọc nội dung của thông báo. Trong trường hợp thứ hai thì người nhận có thể lưu giữ bản rõ thông báo và chữ ký số của nó để sau này giải quyết tranh chấp.

Các sơ đồ trực tiếp có những yếu điểm chung. Tính hợp lệ của sơ đồ phụ thuộc vào khóa mật của người gửi. Do đó người gửi có thể tuyên bố khóa bị lộ hoặc bị mất hoặc ai đó giả mạo chữ ký của anh ta. Để khắc phục tình trạng trên. Nhà chức trách kiểm soát yêu cầu người ký chữ ký số đưa vào thông báo nhãn thời gian và phải báo cáo ngay khi khóa bí mật bị bại lộ.

Vấn đề thứ hai là thời gian. Khóa bí mật bị lộ tại thời điểm T của bên liên lạc X. Khi đó kẻ chống phá hoại X có thể tạo ra các thông báo với chữ ký của X với tem thời gian trước khi hoặc cùng thời gian điểm T. Các vấn đề của chữ ký trực tiếp có thể được giải quyết bởi chữ ký phân xử.

### 5.3.3. Chữ ký số phân xử.

Cơ chế của chữ ký phân xử như sau : mỗi thông báo được ký từ người gửi X đến người nhận Y đầu tiên đi đến người phân xử A. Anh ta thử thông báo và chữ ký số một số lần để kiểm tra nguồn gốc và nội dung của nó. Thông báo được định ngày và gửi đi cho Y với dấu hiệu rằng nó đã được kiểm tra và thỏa mãn được yêu cầu của người phân xử. Sự có mặt của người phân xử giải quyết được hạn chế của sơ đồ chữ ký trực tiếp trong đó X có thể không thừa nhận mình là người chủ của thông báo.

Thông thường với sơ đồ chữ ký số phân xử người ta dùng mật mã truyền thống. Khi đó chúng ta nhận thấy vai trò quá lớn của người phân xử vì anh ta có thể đọc được mọi thông tin giữa X và Y. Do đó các bên liên lạc phải tin cậy cao độ vào người phân xử.

Để hạn chế người phân xử người ta có thể lập mã thông tin từ X đến Y và người phân xử không đọc được thông tin này nữa. Cho dù vậy anh ta vẫn có thể câu kết với người gửi để từ chối thông báo đã ký hoặc với người nhận để giả mạo chữ ký của người gửi.

Sử dụng mật mã khóa công khai sẽ giúp chúng ta giải quyết các vấn đề còn vướng mắc.

Trong sơ đồ chữ ký số phân xử dùng mật mã khóa công khai không cần chia sẻ thông tin trước khi liên lạc tránh được sự câu kết để lừa đảo.

Thứ nữa là thông báo định ngày sai không thể gửi đi được thậm chí cả khi khóa mật của người gửi X bị lộ khi mà khóa mật của người phân xử không bị lộ.

Nội dung thông báo đi từ X đến Y không thể đọc được đối với A và bất kỳ ai khác.

Dưới đây là một số sơ đồ chữ ký số:

(a) *Dùng mật mã khoá bí mật, người phân xử biết nội dung thông báo.*

1.  $X \rightarrow A : R \parallel E_{Kxa}[IDx \parallel H(R)]$

2.  $A \rightarrow Y : E_{Kay}[IDx \parallel R \parallel E_{Kxa}[IDx \parallel H(R)], T]$

(b) *Dùng mật mã khoá bí mật, người phân xử không biết nội dung thông báo.*

1.  $X \rightarrow A : IDx \parallel E_{Kxy}[R] \parallel E_{Kxa}[Idx \parallel H(E_{Kxy}[R])]$
2.  $A \rightarrow Y : E_{Kay} [Idx \parallel E_{Kxy}[R] \parallel E_{Kxa}[Idx \parallel H(E_{Kxy}[R])], T]$

(c) Dùng mật mã khoá công khai, người phân xử không biết nội dung thông báo.

1.  $X \rightarrow A : IDx \parallel E_{KR_x}[Idx \parallel E_{KR_x}(E_{KU_y}[R])]$
2.  $A \rightarrow Y : E_{KR_a}[Idx \parallel E_{KR_x}(E_{KU_y}[R]) \parallel T]$

Với các ký hiệu sau :

X- người gửi thông tin.

A- người phân xử.

Y- người nhận thông tin.

$ID_x$  - định danh của người gửi X.

R- thông báo gửi từ X đến Y.

$E_{KR_x}$  [ ] lập mã nội dung thông tin dùng khóa bí mật  $KR_x$  của X.

$KU_y$  - khóa công khai của Y.

T - thời gian do A ấn định.

$KR_a$  - khóa bí mật của người phân xử A.

$\parallel$  - nối các phần thông báo với nhau.

Người phân xử đóng vai trò quyết định trong sơ đồ và tất cả các thành viên phải tuyệt đối tin cậy rằng cơ chế phân xử đang làm việc đúng đắn.

Trong sơ đồ (a), mật mã khoá bí mật được dùng. Nó thừa nhận rằng người gửi X và người phân xử A cùng chia sẻ một khoá bí mật Kxa, Y và A chia sẻ khoá bí mật Kay. X tạo thông báo R và tính  $H(R)$ . Sau đó X gửi thông báo cùng với chữ ký cho A. Chữ ký bao gồm một định danh của X, giá trị mã Hash, tất cả được mã dùng Kxa. A giải mã chữ ký và kiểm tra giá trị Hash để xác nhận tính hợp lệ của thông báo. Sau đó A gửi thông báo tới Y, được mã bởi Kay. Thông báo bao gồm  $ID_x$ , thông báo nguyên bản từ X, chữ ký và tem thời gian. Y có thể giải mã bản mã này để khôi phục thông báo và chữ ký. Tem thời gian đảm bảo với Y rằng thông báo là đúng thời hạn và không bị truyền lại. Y có thể lưu giữ R và chữ ký.

Trong trường hợp tranh cãi, Y tuyên bố đã nhận R từ X và gửi thông báo tới A:  $E_{Kay} [Idx \parallel R \parallel E_{Kxa}[Idx \parallel H(R)]]$

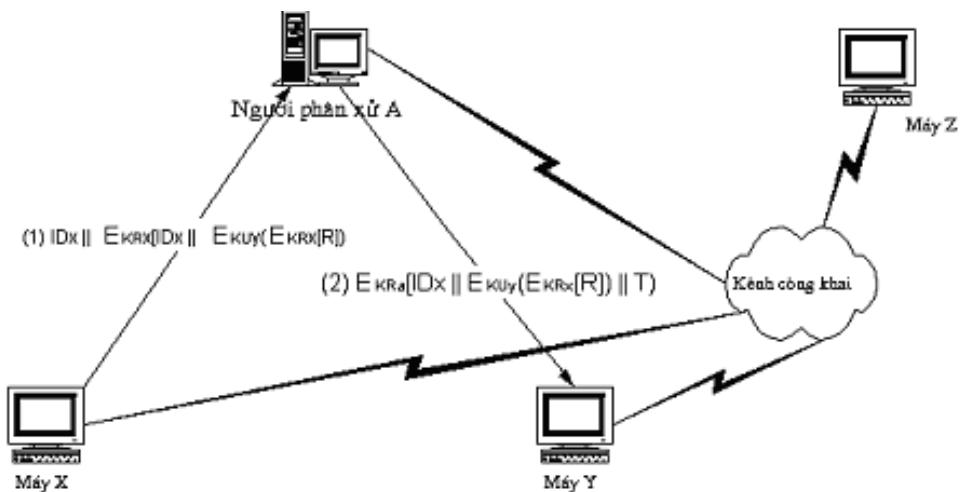
Người phân xử dùng Kay để khôi phục  $ID_x$ , R và chữ ký, sau đó dùng Kxa để giải mã chữ ký và xác nhận mã hash. Trong sơ đồ này, Y không thể kiểm tra trực tiếp chữ ký của X, chữ ký là bằng chứng duy nhất để dàn xếp tranh cãi. Y coi thông báo từ X là xác thực vì nó đến thông qua A. Trong sơ đồ này cả hai phải tin tưởng vào A:

- X phải tin tưởng A không để lộ Kxa và không sinh ra chữ ký lỗi  $E_{Kxa}[Idx \parallel H(R)]$
- Y phải tin tưởng A là người gửi  $E_{Kay} [Idx \parallel R \parallel E_{Kxa}[Idx \parallel H(R)]]$
- Cả X và Y tin tưởng A giải quyết tranh cãi đúng.

Trong sơ đồ (b), cả X và Y chia sẻ khoá bí mật Kxy. X gửi một định danh, một bản sao của thông báo được mã với Kxy và chữ ký tới A. Chữ ký bao gồm định danh và giá trị Hash của thông báo được mã. Tất cả được mã dùng Kxa. A giải mã chữ ký và kiểm

tra giá trị Hash để xác nhận tính hợp lệ của thông báo. Sau đó A gửi cho Y thông báo nhận được từ X, cùng với tem thời gian, tất cả được mã bởi Kay.

Mặc dù không thể đọc được thông báo, người phân xử vẫn giữ vai trò ngăn chặn sự gian lận của X hoặc Y. Có một vấn đề là trong sơ đồ đầu tiên, người phân xử có thể liên minh với người gửi để từ chối thông báo đã ký hoặc với người nhận để giả mạo chữ ký của người gửi. Tất cả các vấn đề trên có thể được giải quyết bằng sơ đồ (c) dùng mật mã khoá công khai. Trong trường hợp này X mã đúp một thông báo R đầu tiên với khoá bí mật KR<sub>x</sub> của X, sau đó với khoá công khai KU<sub>y</sub> của Y. Đây là một phiên bản bí mật được ký của thông báo. Thông báo được ký này cùng với định danh của X được mã lần nữa với KR<sub>x</sub>, cùng với ID<sub>x</sub> được gửi cho A. Thông báo mã đúp này là an toàn đối với A. Tuy nhiên A có thể giải mã để thừa nhận rằng thông báo phải đến từ X (bởi vì chỉ có X có KR<sub>x</sub>). A kiểm tra để đảm bảo rằng cặp khoá của A vẫn đúng và xác thực thông báo. Sau đó A truyền thông báo tới Y được mã với KR<sub>a</sub>. Thông báo bao gồm ID<sub>x</sub>, thông báo mã đúp và một tem thời gian.



**Hình 5.13: Mô hình cài đặt sơ đồ chữ ký số phân xử dùng mật mã khoá công khai**

Trên một mạng có n máy có nhu cầu sử dụng sơ đồ chữ ký số có phân xử. Khi đó cần có thêm một máy Người phân xử đóng vai trò phân xử chữ ký số.

Giả sử máy X cần gửi cho máy Y một thông báo R có chữ ký như trong sơ đồ (c).

Trước đó Người phân xử đã có khoá công khai KUX của X, máy X đã có khoá công khai KUY của Y, máy Y đã có khoá công khai của Người phân xử.

Để gửi thông báo R cho Y, máy X tính và gửi thông báo (1) cho A. Sau đó A tính và gửi thông báo (2) cho Y.

Kết quả là X đã gửi cho Y thông báo R đảm bảo bí mật và có chữ ký số. Tương tự đối với các cặp máy khác.

## CHƯƠNG 6: QUẢN TRỊ VÀ KIỂM SOÁT

### 6.1. Các nguyên tắc an ninh:

Bảo mật hệ thống thông tin dựa trên các đòi hỏi của pháp luật hiện hành, các tiêu chuẩn, các tài liệu phương pháp chuẩn, được bảo đảm bằng tổ hợp các thiết bị kỹ thuật – chương trình và các biện pháp tổ chức trợ giúp chúng ở tất cả các giai đoạn công nghệ của xử lý thông tin và trong tất cả các chế độ hoạt động của các thiết bị kể cả khi sửa chữa và niêm cát.

Các thiết bị kỹ thuật – chương trình của bảo vệ không được gây ảnh hưởng xấu tới các đặc trưng hoạt động cơ bản của hệ thống (độ tin cậy, tính linh hoạt, khả năng thay đổi cấu hình...). Một trong những phần không thể bỏ qua của công việc về an toàn, bảo mật hệ thống là việc đánh giá hiệu quả của các thiết bị bảo vệ, được tiến hành theo phương pháp có tính tới toàn bộ các đặc trưng kỹ thuật của đối tượng được đánh giá kể cả các giải pháp kỹ thuật và sự thực hiện trên thực tế các thiết bị bảo vệ. Việc bảo vệ hệ thống phải đi kèm sự kiểm soát hiệu quả các thiết bị bảo vệ được đề xuất định kỳ bởi người dùng hoặc bởi cơ quan kiểm tra.

Những đòi hỏi nêu trên có thể thực hiện nhờ 7 nguyên tắc gồm: Nguyên tắc tính hệ thống, nguyên tắc tổng thể, nguyên tắc bảo vệ liên tục, nguyên tắc đầy đủ hợp lý, nguyên tắc mềm dẻo hệ thống, nguyên tắc công khai của thuật toán và cơ chế bảo vệ, nguyên tắc đơn giản trong sử dụng.

#### a. Nguyên tắc tính hệ thống

Tiếp cận hệ thống trong bảo mật HTTT nói rằng: cần phải kiểm kê tất cả các yếu tố, các điều kiện và các nhân tố có quan hệ với nhau, có tương tác với nhau và có biến đổi theo thời gian:

- Trong tất cả các dạng hoạt động thông tin và thể hiện thông tin.
- Với tất cả các thành tố của hệ thống.
- Trong tất cả các chế độ hoạt động.
- Ở tất cả các giai đoạn của chu kỳ sống.
- Trong sự tương tác của đối tượng bảo vệ với môi trường bên ngoài.

Khi thực hiện bảo mật hệ thống cần phải tính tới tất cả các điểm xung yếu, các vị trí dễ tổn thương của hệ thống xử lý thông tin, và cả đặc trưng, các đối tượng tiềm năng, các hướng của các tấn công và hệ thống từ phía những kẻ phá hoại (đặc biệt kẻ ác ý có trình độ chuyên môn cao), các con đường xâm nhập vào các hệ thống phân tán và các kenh tiếp cận trái phép tới thông tin. Hệ thống bảo vệ phải được thiết lập không chỉ tính tới tất cả các kenh xâm nhập đã biết, mà còn cả khả năng xuất hiện các kenh hoàn toàn mới của các nguy cơ an toàn.

#### b. Nguyên tắc tổng thể

Trong tay các chuyên gia an toàn, bảo mật máy tính có rất nhiều biện pháp, phương pháp và thiết bị bảo vệ hệ thống máy tính. Các thiết bị tính toán hiện đại, các hệ điều

hành, các thiết bị chương trình ứng dụng và chỉ dẫn đều có cài đặt các yếu tố bảo vệ khác nhau. Sử dụng tổng thể đồng bộ các yếu tố này yêu cầu sự tương thích đồng bộ của các thiết bị khác loại khi xây dựng hệ thống toàn diện để bịt kín tất cả các khe hở xâm nhập của các hiểm họa và không chứa các vị trí xung yếu ở nơi tiếp giáp của các thành tố của hệ thống.

#### c. Nguyên tắc bảo vệ liên tục

Bảo vệ thông tin - đó không phải là biện pháp một vài lần và thậm chí đó không phải là tập hợp cụ thể của các biện pháp đã thực hiện và các thiết bị đã cài đặt, mà đó là một quá trình liên tục hướng tới mục tiêu, yêu cầu phải đưa ra các giải pháp phù hợp ở tất cả các giai đoạn của chu kỳ sống của hệ thống (bắt đầu ngay từ lúc thiết kế chứ không phải chỉ trong khi khai thác hệ thống). Thiết kế hệ thống bảo vệ phải được tiến hành song song với thiết lập chính hệ thống được bảo vệ.

Phần lớn các thiết bị bảo vệ kỹ thuật và vật lý cần có sự trợ giúp thường xuyên của các biện pháp tổ chức (hành chính) để thực hiện có hiệu quả các chức năng của chúng (ví như sự thay đổi kịp thời, bảo quản chặt chẽ và ứng dụng linh hoạt các tên, mật khẩu, các khoá mã, sự phân quyền v.v...). Sự gián đoạn (hoặc ngừng tạm thời) trong hoạt động của các thiết bị bảo vệ có thể bị kẻ ác lợi dụng để đưa vào các chương trình đặc biệt, các thiết bị cài bẫy và các phương tiện khác để qua mặt hệ thống bảo vệ khi hệ thống làm việc trở lại.

#### d. Nguyên tắc đầy đủ hợp lý

Thiết lập một hệ bảo vệ tuyệt đối không chọc thủng được là một điều không tưởng vì rằng với đầy đủ điều kiện và phương tiện có thể vượt qua mọi hệ bảo vệ. Ví dụ, các phương tiện bảo vệ mật mã trong phần lớn các trường hợp không bảo đảm độ bền vững tuyệt đối, mà chúng chỉ bảo đảm sự bí mật thông tin trong điều kiện bị tấn công mã thám liên tục bằng các máy tính hiện đại, trong một khoảng thời gian phù hợp với yêu cầu bảo vệ mà thôi. Do đó cần nói về một độ bảo vệ vừa đủ nào đó. Một hệ thống bảo vệ hiệu quả có chi phí khá đắt. Nó sử dụng công suất đáng kể của máy tính và các tài nguyên đi kèm và do đó nó có thể gây thêm cho người dùng hệ thống một sự bất tiện và rắc rối đáng kể. Điều quan trọng là phải lựa chọn đúng mức độ bảo vệ cần thiết, mà trong đó các chi phí, độ mạo hiểm và phạm vi các thiệt hại có thể là chấp nhận được (bài toán phân tích độ mạo hiểm).

#### e. Nguyên tắc mềm dẻo hệ thống

Thông thường phải thiết lập hệ bảo vệ trong các điều kiện bất định khá lớn. Cho nên các biện pháp thực hiện và các thiết bị lắp đặt cho bảo vệ, nhất là ở giai đoạn đầu đi vào hoạt động, có thể bảo đảm hoặc là một độ bảo vệ quá mức hoặc là quá thấp. Do vậy để có thể điều chỉnh độ bảo vệ, các thiết bị như vậy phải có sự mềm dẻo nhất định. Đặc biệt điều này quan trọng khi mà hệ bảo vệ được đưa vào một hệ thống đang làm việc mà không được phép phá vỡ quá trình hoạt động bình thường của nó. Ngoài ra, điều kiện bên ngoài, các yêu cầu bảo vệ theo thời gian cũng có thay đổi. Trong những tình huống như

vậy, tính chất mềm dẻo hệ thống bảo vệ sẽ giúp cho việc nâng cấp hệ thống dễ dàng mà không phải thay thế mới toàn bộ máy móc thiết bị của hệ thống.

#### f. Nguyên tắc công khai của thuật toán và cơ chế bảo vệ

Bản chất của nguyên tắc này là ở chỗ, sự bảo vệ không được chỉ dựa vào bí mật của cơ cấu tổ chức và các thuật toán hoạt động của các tiêu hệ (bộ phận). Dù có biết thuật toán làm việc của hệ thống bảo vệ thì cũng không thể qua mặt được nó (thậm chí cả tác giả của hệ thống bảo vệ cũng vậy).

#### g. Nguyên tắc đơn giản trong sử dụng

Các cơ chế bảo vệ phải dễ hiểu và đơn giản trong sử dụng. Việc áp dụng các thiết bị bảo vệ không được buộc phải biết các ngôn ngữ đặc biệt hoặc buộc phải thực hiện các thao tác khó khăn đối với người dùng hợp pháp, kể cả việc thực hiện các thao tác khó hiểu rắc rối.

### 6.2. Đánh giá rủi ro

Rủi ro (risk) là xác suất xảy ra thiệt hại đối với hệ thống.

Rủi ro bao gồm 2 yếu tố: Khả năng xảy ra rủi ro và thiệt hại do rủi ro gây ra. Có những rủi ro có khả năng xảy ra rất cao nhưng mức độ thiệt hại thì thấp và ngược lại.

Ví dụ: rủi ro mất thông tin trên hệ thống không có cơ chế bảo vệ tập tin, chẳng hạn như Windows 98. Windows 98 không có cơ chế xác thực người sử dụng nên bất cứ ai cũng có thể sử dụng máy với quyền cao nhất. Nếu trên đó chỉ có chứa các tập tin văn bản không có tính bí mật thì việc mất một tập tin thì thiệt hại gây ra chỉ là mất công sức đánh máy văn bản đó. Đây là dạng rủi ro có xác suất xảy ra cao nhưng thiệt hại thấp.

Một ví dụ khác: trên máy chủ cung cấp dịch vụ có một phần mềm có lỗi tràn bộ đệm, và nếu khai thác được lỗi này thì kẻ tấn công có thể chiếm được quyền điều khiển toàn bộ hệ thống.

Tuy nhiên, đây là phần mềm không phổ biến và để khai thác được lỗi này, kẻ tấn công phải có những kỹ năng cao cấp. Rủi ro hệ thống bị chiếm quyền điều khiển được đánh giá là có khả năng xảy ra thấp, nhưng nếu có xảy ra, thì thiệt hại sẽ rất cao.

Cần chú ý phân biệt giữa nguy cơ và rủi ro. Nguy cơ là những hành vi, những sự kiện hoặc đối tượng có khả năng gây hại cho hệ thống. Rủi ro là những thiệt hại có khả năng xảy ra đối với hệ thống.

Ví dụ: Tấn công từ chối dịch vụ là một nguy cơ (threat). Đây là một sự kiện có khả năng xảy ra đối với bất kỳ hệ thống cung cấp dịch vụ nào. Thiệt hại do tấn công này gây ra là hệ thống bị gián đoạn hoạt động, đây mới là rủi ro (risk). Tuy nhiên, không phải bất kỳ tấn công từ chối dịch vụ nào xảy ra cũng đều làm cho hệ thống ngưng hoạt động, và hơn nữa, tấn công từ chối dịch vụ không phải là nguồn gốc duy nhất gây ra gián đoạn hệ thống; những nguy cơ khác như lỗi hệ thống (do vận hành sai), lỗi phần mềm (do lập trình), lỗi phần cứng (hư hỏng thiết bị, mất điện, ...) cũng đều có khả năng dẫn đến gián đoạn hệ thống.

Một ví dụ khác, xét trường hợp lưu trữ tập tin trên một máy tính chạy hệ điều hành Windows 98 đã nói ở trên. Nguy cơ đối với hệ thống là các hành vi sửa hoặc xoá tập tin trên máy người khác. Những người hay sử dụng máy tính của người khác cũng được xem là nguy cơ đối với hệ thống. Rủi ro đối với hệ thống trong trường hợp này là việc tập tin bị mất hoặc bị sửa.

Trong thực tế, việc đề ra chính sách bảo mật cho một hệ thống thông tin phải đảm bảo được sự cân bằng giữa lợi ích của việc bảo đảm an toàn hệ thống và chi phí thiết kế, cài đặt và vận hành các cơ chế bảo vệ chính sách đó.

Công việc quản lý rủi ro trên một hệ thống là quy trình cần thiết để nhận diện tất cả những rủi ro đối với hệ thống, những nguy cơ có thể dẫn đến rủi ro và phân tích lợi ích / chi phí của giải pháp ngăn chặn rủi ro. Quy trình phân tích rủi ro bao gồm các bước:

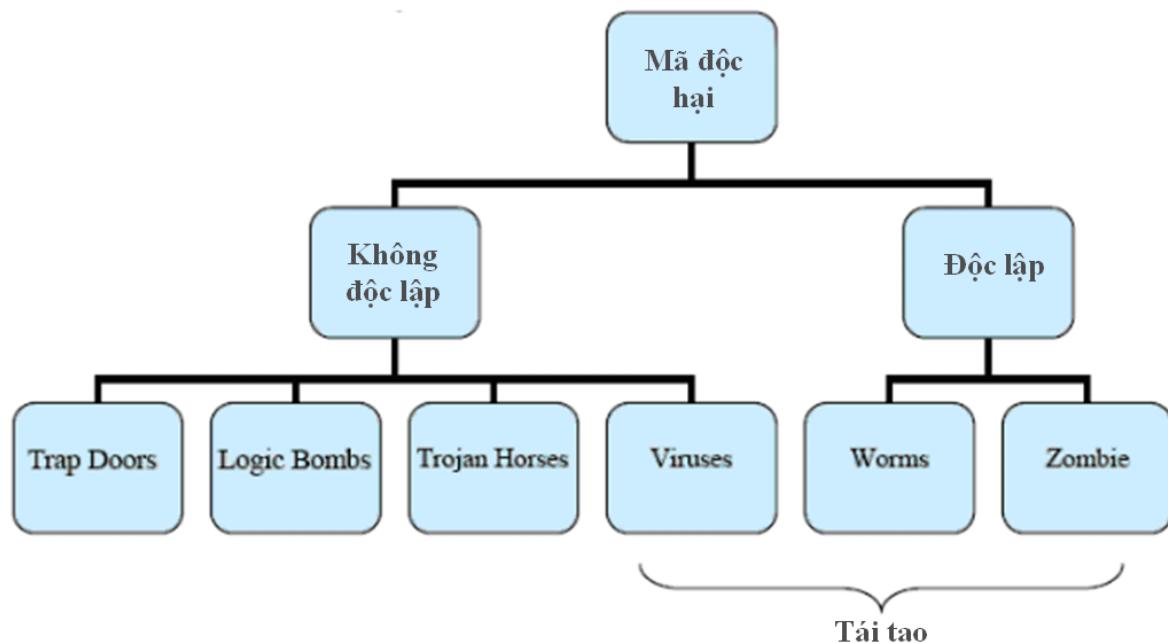
- Nhận dạng các rủi ro đối với hệ thống
- Chọn lựa và thực hiện các giải pháp để giảm bớt rủi ro.
- Theo dõi và đánh giá thiệt hại của những rủi ro đã xảy ra, làm cơ sở cho việc điều chỉnh lại hai bước đầu

## CHƯƠNG 7: PHẦN MỀM MÃ ĐỘC

Trong khi những người dùng ác ý hay hacker có thể tấn công hệ thống, các chương trình được phát hành bởi cùng một nhóm người này thường sẽ thành công hơn trong việc tiếp cận các thành phần bảo vệ của tổ chức. Mã độc hại được định nghĩa là bất kỳ đoạn mã trong một phần mềm hay kịch bản (script) được thiết kế với mục đích làm cho một hệ thống thực hiện bất kỳ hành động nào theo ý của người sử dụng (mã độc hại) với quyền của chủ sở hữu hệ thống. Một trong những cách nhanh nhất để đưa mã độc vào mạng được bảo vệ của một tổ chức mục tiêu là gửi các mã độc hại thông qua thư điện tử.

Có rất nhiều loại khác nhau của mã độc và có nhiều tiêu chí để phân loại mã độc hại, dưới đây là hai cách phân loại dựa vào hình thức lây nhiễm và theo phân loại của NIST-National Institute of Standard and Technology (Viện tiêu chuẩn – công nghệ quốc gia Hoa Kỳ). Sự phân loại này chỉ mang tính chất tương đối.

### 7.1 Theo hình thức lây nhiễm



Hình 7.1 Phân loại mã độc hại

Theo hình trên mã độc hại gồm 2 loại chính: một loại cần vật chủ để tồn tại và lây nhiễm, vật chủ ở đây có thể là các file dữ liệu, các file ứng dụng, hay các file chương trình thực thi... và một loại là tồn tại độc lập.

Độc lập nghĩa đó là chương trình độc hại có thể được lập lịch và chạy trên hệ điều hành.

Không độc lập (needs host program) là một đoạn chương trình đặc biệt thuộc một chương trình nào đó không thể thực thi độc lập như một chương trình thông thường hay tiện ích nào đó mà bắt buộc phải có bước kích hoạt chương trình chủ trước đó thì chương trình đó mới chạy.

### 7.1.1 Trap Door

Trap Door còn được gọi là Back Door. Trong đời sống thường, Trap Door mang ý nghĩa “cánh cửa” để vào một tòa nhà. Trap door là một điểm bí mật trong một chương trình, cho phép một ai đó có thể truy cập lại hệ thống mà không phải vượt qua các hàng rào an ninh như thông thường. Trap door được sử dụng bởi những nhà lập trình với mục đích dò lỗi, kiểm tra chương trình.

Trong các cuộc tấn công trap door là phần mềm độc hại thường trú và đợi lệnh điều khiển từ các cổng dịch vụ TCP hoặc UDP. Trap door khi chạy trên máy bị nhiễm, nó sẽ thường trú trong bộ nhớ và mở một cổng cho phép kẻ tấn công truy nhập vào máy nạn nhân thông qua cổng mà nó đã mở và kẻ tấn công có toàn quyền điều khiển máy bị nhiễm.

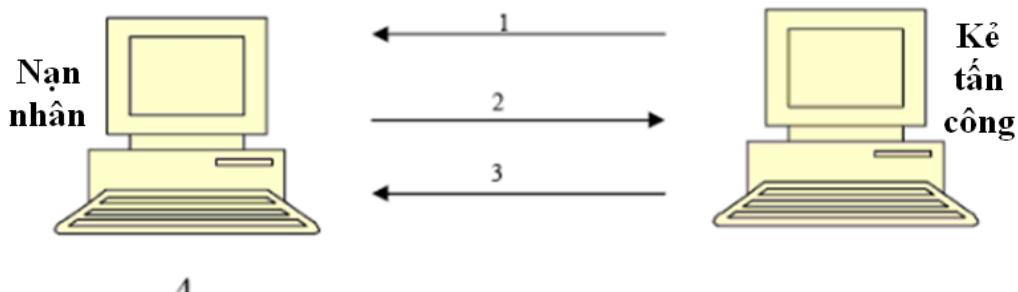
Trap door nguy hiểm ở chỗ nó hoàn toàn chạy ẩn trong máy. Nhiều Trap door được hẹn trước giờ để kết nối ra ngoài (đến một giờ nhất định mới mở một cổng để hacker đột nhập vào) nên rất khó phát hiện ngay cả quét cổng.

Ví dụ: Back door W32/TDSS là một chương trình chiếm quyền điều khiển từ xa, bí mật điều khiển hệ thống máy tính bị nhiễm.

### 7.1.2 Logic Bombs

Logic bomb là đoạn mã độc được nhúng vào một chương trình hợp pháp mà chúng có thể thực thi khi có một sự kiện nào đó xảy ra. Các đoạn mã thường được chèn vào các ứng dụng hoặc các hệ điều hành để thực hiện việc phá hủy hệ thống hoặc phá hủy các chức năng an toàn của hệ thống.

Logic bomb có thể gửi thông báo tới kẻ tấn công khi người dùng truy nhập Internet và sử dụng một chương trình đặc biệt nào đó như bộ xử lý văn bản. Từ đó kẻ tấn công có thể chuẩn bị cho các cuộc tấn công (chẳng hạn kết hợp với các máy tính khác bị nhiễm để bắt đầu một cuộc tấn công từ chối dịch vụ).



Hình 0.2 Mô hình hoạt động Logic bomb

1. Kẻ tấn công đưa bom logic vào máy nạn nhân thông qua một chương trình ứng dụng nào đó.
2. Máy nạn nhân cài đặt chương trình ứng dụng.
3. Kẻ tấn công gửi thông điệp tấn công.
4. Bom logic đã cài đặt trên máy nạn nhân.

### 7.1.3 Trojan Horses

Trojan Horse là loại mã độc hại được đặt theo sự tích “Ngựa thành Troy”. Trojan horse không có khả năng tự nhân bản tuy nhiên nó lây vào hệ thống với biểu hiện rất bình thường nhưng thực chất bên trong có ẩn chứa các đoạn mã với mục đích gây hại. Trojan có thể gây hại theo ba cách sau:

Tiếp tục thực thi các chức năng của chương trình mà nó bám vào, bên cạnh đó thực thi các hoạt động gây hại một cách riêng biệt (ví dụ như gửi một trò chơi cho người dùng sử dụng, bên cạnh đó là một chương trình đánh cắp mật khẩu).

Tiếp tục thực thi các chức năng của chương trình mà nó bám vào, nhưng sửa đổi một số chức năng để gây tổn hại (ví dụ như một trojan giả lập một cửa sổ đăng nhập để lấy mật khẩu) hoặc che giấu các hành động phá hoại khác (ví dụ như trojan che giấu cho các tiến trình độc hại khác bằng cách tắt các hiển thị của hệ thống).

Thực thi luôn một chương trình gây hại bằng cách nấp dưới danh một chương trình không có hại (ví dụ như một trojan được giới thiệu như là một trò chơi hoặc một công cụ trên mạng, người dùng chỉ cần kích hoạt file này là lập tức dữ liệu trên máy tính sẽ bị xoá hết).

Có 7 loại trojan chính:

Trojan truy cập từ xa: Được thiết kế để cho kẻ tấn công có khả năng từ xa chiếm quyền điều khiển của máy bị hại. Các trojan này thường được giấu vào các trò chơi và các chương trình nhỏ làm cho người dùng mất cảnh giác.

**Trojan gửi dữ liệu:** Nó thực hiện việc lấy và gửi dữ liệu nhạy cảm như mật khẩu, thông tin thẻ tín dụng, các tệp nhật ký, địa chỉ email... cho kẻ tấn công. Trojan này có thể tìm kiếm cụ thể thông tin hoặc cài phần mềm đọc trộm bàn phím và gửi toàn bộ các phím bấm về cho kẻ tấn công.

**Trojan hủy hoại:** Thực hiện việc xóa các tệp tin. Loại trojan này giống với virus và thường có thể bị phát hiện bởi các chương trình diệt virus.

**Trojan kiểu proxy:** Sử dụng máy tính bị hại làm proxy, qua đó có thể sử dụng máy bị hại để thực hiện các hành vi lừa gạt hay đánh phá các máy tính khác.

**Trojan FTP:** Được thiết kế để mở cổng 21 và cho phép tin tặc kết nối vào máy bị hại sử dụng FTP.

**Trojan tắt phần mềm an ninh:** Thực hiện việc dừng hoặc xóa bỏ chương trình an ninh như phần mềm chống virus hay tường lửa mà người dùng không nhận ra.

**Trojan DoS:** Được sử dụng trong các cuộc tấn công từ chối dịch vụ. Ví dụ các con bot sử dụng trong DDoS cũng có thể coi là một loại trojan.

Ví dụ trojan có tên Zeus, Clampi đã mang về cho tội phạm hàng triệu USD bằng cách ghi lại thông tin tài khoản để làm thẻ giả hoặc chuyển tiền vào tài khoản của một bên trung gian - gọi là Mule. Mule sau đó được trả công để đảm nhận việc gửi tiền ra nước ngoài. Mule được thuê thông qua các trang tìm kiếm việc làm và họ không hề biết rằng số tiền họ nhận gửi đi là bất hợp pháp.

#### 7.1.4 Virus

Virus là một loại mã độc hại có khả năng tự nhân bản và lây nhiễm chính nó vào các file, chương trình hoặc máy tính. Virus phải luôn bám vào vật chủ (có thể là file dữ liệu hoặc file ứng dụng) để lây lan. Virus có thể làm bất cứ việc gì mà các chương trình khác có thể làm. Virus chỉ khác ở điểm nó tự đính kèm vào một chương trình và thực thi bí mật khi chương trình mang virus được thực thi. Khi virus được thực thi nó có thể làm bất kỳ việc gì trên hệ thống như xóa file, chương trình. Vòng đời virus gồm 4 giai đoạn:

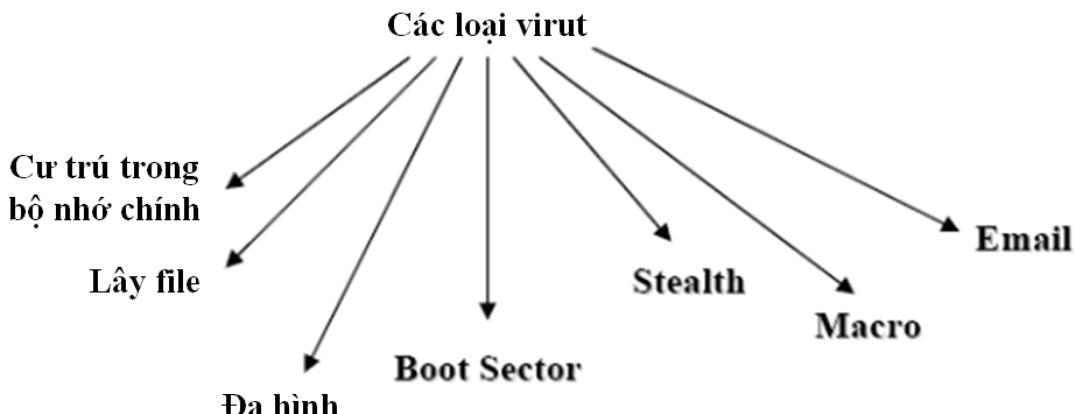
**Dormant (nằm im):** Trong giai đoạn này virus không làm gì cho đến khi được kích hoạt bởi một ai đó hay một sự kiện nào đó.

**Propagation (lây lan):** Trong giai đoạn này virus thực hiện việc sao chép chính nó tới các chương trình, vị trí khác trong ổ đĩa.

**Triggering (kích hoạt):** Trong giai đoạn này virus được kích hoạt để thực thi chức năng của nó.

**Execution** (thực thi): Chức năng của virus được thực thi. Chức năng có thể là vô hại như gửi một thông điệp nào đó tới màn hình, hoặc một chức năng có hại như phá hủy các chương trình, các file hệ thống.

Virus gồm 7 loại chính:



Hình 0.3 Phân loại virus

**Memory – resident virus** (virus cư trú trong bộ nhớ chính): Cư trú rong bộ nhớ chính như là một phần của chương trình hệ thống. Theo đó virus sẽ gây ảnh hưởng mỗi khi chương trình được thực thi.

**Program file virus** (virus lây file): Gây ảnh hưởng đến các file chương trình như exe/com/sys.

**Polymorphic virus** (virus đa hình): Loại virus này tự thay đổi hình thức của nó, gây khó khăn cho các chương trình diệt virus. Virus “Tequila” là loại virus đa hình đầu tiên xuất hiện năm 1991.

**Boot Sector virus**: Là loại virus đầu tiên trên thế giới được phổ biến rộng rãi và được viết vào năm 1986. Boot virus lợi dụng tiến trình boot của máy tính để thực hiện việc kích hoạt mình. Khi máy tính được khởi động, nó luôn tìm đến master boot record được lưu trữ tại địa chỉ head 0, track 0, sector 1 để đọc thông tin. Boot Sector virus lây lan sang đĩa cứng khi khởi động hệ thống từ đĩa mềm bị nhiễm.

**Stealth virus**: Đây là loại virus có khả năng tự che giấu không để cho hệ điều hành và phần mềm chống virus biết. Nó nằm trong bộ nhớ để ngăn chặn sử dụng hệ điều hành và che giấu những thay đổi về kích thước các tập tin. Những virus này chỉ bị phát hiện khi chúng còn ở trong bộ nhớ. Có nhiều boot sector virus có khả năng Stealth. Ví dụ virus "The Brain" được tạo ra tại Pakistan bởi Basit và Amjad. Chương trình này nằm trong phần khởi động (boot sector) của một đĩa

mềm 360Kb và nó sẽ lây nhiễm tất cả các ổ đĩa mềm. Đây là loại "stealth virus" đầu tiên.

Macro virus: Là tập lệnh được thực thi bởi một ứng dụng nào đó. Macro virus phổ biến trong các ứng dụng Microsoft Office khi tận dụng khả năng kiểm soát việc tạo và mở file để thực thi và lây nhiễm. Ví dụ: virus Baza, Laroux và một số virus Staog xuất hiện năm 1996 tấn công các file trong hệ điều hành Windows 95, chương trình bảng tính Excel và cả Linux. Virus Melisa cũng là một trong những Macro virus nổi tiếng.

Thư điện tử virus: Là những virus được phát tán qua thư điện tử. Ví dụ virus Melissa được đính kèm trong thư điện tử. Nếu người dùng mở file đính kèm Macro được kích hoạt sau đó thư điện tử virus này tự động gửi chính nó tới tất cả những hòm thư có trong danh sách thư của người đó.

#### 7.1.5 Worm

Worm (sâu mạng) là chương trình độc hại có khả năng tự nhân bản và tự lây nhiễm trong hệ thống, nó có khả năng “tự đóng gói”, điều đó có nghĩa là nó không cần file chủ để mang nó khi nhiễm vào hệ thống. Như vậy Worm không bám vào một file hoặc một vùng nào đó trên đĩa cứng, vì vậy không thể dùng các chương trình quét file để diệt Worm.

Mục tiêu của Worm là làm lãng phí băng thông của mạng, phá hoại hệ thống như xóa file, tạo back door, thả keylogger...

Tấn công của Worm có đặc trưng là lan rộng cực kỳ nhanh chóng do không cần tác dụng của con người (như khởi động máy, sao chép tập tin hay đóng/mở file).

Worm có thể chia làm 2 loại:

Network Service Worm lan truyền bằng cách lợi dụng các lỗ hổng bảo mật của mạng, của hệ điều hành hoặc của ứng dụng. Ví dụ Sasser (W32.Sasser.Worm) bắt đầu lây lan trên mạng vào 1/5/2004 có thể tự động lây lan bất cứ máy tính nào kết nối Internet. Nó lợi dụng lỗ hổng bảo mật Local Security Authority Subsystem Service (LSASS - lỗ này đã được Microsoft công bố và phát hành bản sửa lỗi ngày 13-4-2004) để tấn công các máy cài Windows 2000/ XP/ Máy chủ 2003.

Mass Mailing Worm là một dạng tấn công qua dịch vụ mail, tuy nhiên nó tự đóng gói để tấn công và lây nhiễm chứ không bám vào vật chủ là email. Khi sâu này lây nhiễm vào hệ thống, nó thường cố gắng tìm kiếm số địa chỉ và tự gửi bản

thân nó đến các địa chỉ thu nhặt được. Việc gửi đồng thời cho toàn bộ các địa chỉ thường gây quá tải cho mạng hoặc cho máy chủ mail. Ví dụ Mydoom là một trong những loại Worm khiến nhiều hệ thống máy tính bị tê liệt và gây thiệt hại tài chính lên đến hàng tỷ đôla.

### 7.1.6 Zombie

Zombie là chương trình độc hại bí mật liên kết với một máy tính khác ngoài internet để nghe lệnh từ các máy tính đó. Các Zombie thường sử dụng trong các cuộc tấn công từ chối dịch vụ DDoS để tấn công vào một website nào đó.

Kiểu thông dụng nhất của Zoombie là các agent dùng để tổ chức một cuộc tấn công DDoS. Khi tấn công có thể cài Zoombie vào một số lượng lớn các máy tính rồi ra lệnh tấn công cùng một lúc.

Ví dụ Trinoo và Tribe Flood Network là hai Zoombie nổi tiếng được sử dụng như các công cụ để thực hiện tấn công DDoS.

## 7.2 *Phân loại của NIST*

### 7.2.1 Virus

Với cách định nghĩa, phân loại này, virus là một loại mã độc hại (Malicious code) có khả năng tự nhân bản và lây nhiễm chính nó vào các file, chương trình hoặc máy tính. Như vậy virus máy tính phải luôn luôn bám vào một vật chủ (đó là file dữ liệu hoặc file ứng dụng) để lây lan. Các chương trình diệt virus dựa vào đặc tính này để thực thi việc phòng chống và diệt virus, để quét các file trên thiết bị lưu, quét các file trước khi lưu xuống ổ cứng... Điều này cũng giải thích vì sao đôi khi các phần mềm diệt virus tại PC đưa ra thông báo “phát hiện ra virus nhưng không diệt được” khi thấy có dấu hiệu hoạt động của virus trên PC, bởi vì “vật mang virus” lại nằm ở máy khác nên không thể thực thi việc xoá đoạn mã độc hại đó.

Compiled Virus là virus mà mã thực thi của nó đã được dịch hoàn chỉnh bởi một trình biên dịch để nó có thể thực thi trực tiếp từ hệ điều hành. Các loại boot virus như (Michelangelo và Stoned), file virus (như Jerusalem) rất phổ biến trong những năm 80 là virus thuộc nhóm này, compiled virus cũng có thể là pha trộn bởi cả boot virus và file virus trong cùng một phiên bản.

Interpreted Virus là một tổ hợp của mã nguồn mà chỉ thực thi được dưới sự hỗ trợ của một ứng dụng cụ thể hoặc một dịch vụ cụ thể trong hệ thống. Một cách đơn giản, virus kiểu này chỉ là một tập lệnh, cho đến khi ứng dụng gọi thì nó mới được thực thi. Macro virus, scripting virus là các virus nằm trong dạng này. Macro

virus rất phổ biến trong các ứng dụng Microsoft Office khi tận dụng khả năng kiểm soát việc tạo và mở file để thực thi và lây nhiễm. Sự khác nhau giữa macro virus và scripting virus là: Macro virus là tập lệnh thực thi bởi một ứng dụng cụ thể, còn scripting virus là tập lệnh chạy bằng một service của hệ điều hành. Melisa là một ví dụ điển hình về Macro virus, Love Stages là ví dụ cho scripting virus.

### 7.2.2 Worm

Giống với Worm trong tiêu chí phân loại phía trên.

### 7.2.3 Trojan Horse

Giống với Trojan Horse trong tiêu chí phân loại phía trên.

### 7.2.4 Malicious Mobile Code

Là một dạng mã phần mềm có thể được gửi từ xa vào để chạy trên một hệ thống mà không cần đến lời gọi thực hiện của người dùng hệ thống đó. Malicious Mobile Code được coi là khác với virus, worm ở đặc tính là nó không nhiễm vào file và không tìm cách tự phát tán. Thay vì khai thác một điểm yếu bảo mật xác định nào đó, kiểu tấn công này thường tác động đến hệ thống bằng cách tận dụng các quyền ưu tiên ngầm định để chạy mã từ xa. Các công cụ lập trình như Java, ActiveX, JavaScript, VBScript là môi trường tốt cho Malicious mobile code. Một trong những ví dụ nổi tiếng của kiểu tấn công này là Nimda, sử dụng JavaScript.

Kiểu tấn công này của Nimda thường được biết đến như một tấn công hỗn hợp (Blended Atatck). Cuộc tấn công có thể đi tới bằng một thư điện tử khi người dùng mở một thư điện tử đọc bằng trình duyệt web. Sau khi nhiễm vào máy này, Nimda sẽ cố gắng sử dụng số địa chỉ thư điện tử của máy đó để phát tán tới các máy khác. Mặt khác, từ máy đã bị nhiễm, Nimda cố gắng quét các máy khác trong mạng có thư mục chia sẻ mà không bảo mật, Nimda sẽ dùng dịch vụ NetBIOS như phương tiện để chuyển file nhiễm virus tới các máy đó. Đồng thời Nimda cố gắng dò quét để phát hiện ra các máy tính có cài dịch vụ IIS có điểm yếu bảo mật của Microsoft. Khi tìm thấy, nó sẽ sao chép bản thân nó vào máy chủ. Nếu một máy trạm web có điểm yếu bảo mật tương ứng kết nối vào trang web này, client đó cũng bị nhiễm (lưu ý rằng bị nhiễm mà không cần “mở thư điện tử bị nhiễm virus”). Quá trình nhiễm virus sẽ lan tràn theo cấp số nhân.

### 7.2.5 Tracking Cookie

Là một dạng lạm dụng cookie để theo dõi một số hành động duyệt web của người sử dụng một cách bất hợp pháp. Cookie là một file dữ liệu chứa thông tin

về việc sử dụng một trang web cụ thể nào đó của web-client. Mục tiêu của việc duy trì các cookie trong hệ thống máy tính nhằm căn cứ vào đó để tạo ra giao diện, hành vi của trang web sao cho thích hợp và tương ứng với từng web-client. Tuy nhiên tính năng này lại bị lạm dụng để tạo thành các phần mềm gián điệp (spyware) nhằm thu thập thông tin riêng tư về hành vi duyệt web của cá nhân.

#### 7.2.6 Phần mềm gián điệp

Spyware (phần mềm gián điệp) Là loại phần mềm chuyên thu thập các thông tin từ các máy chủ (thông thường vì mục đích thương mại) qua mạng Internet mà không có sự nhận biết và cho phép của chủ máy. Một cách điển hình, spyware được cài đặt một cách bí mật như là một bộ phận kèm theo của các phần mềm miễn phí (freeware) và phần mềm chia sẻ (shareware) mà người ta có thể tải về từ Internet. Một khi đã cài đặt, spyware điều phối các hoạt động của máy chủ trên Internet và lặng lẽ chuyển các dữ liệu thông tin đến một máy khác (thường là của những hàng chuyên bán quảng cáo hoặc của các tin tức). Phần mềm gián điệp cũng thu thập tin tức về địa chỉ thư điện tử và ngay cả mật khẩu cũng như là số thẻ tín dụng. Khác với worm và virus, Spyware không có khả năng tự nhân bản.

#### 7.2.7 Phần mềm quảng cáo

Phần mềm quảng cáo (Adware) rất hay có ở trong các chương trình cài đặt tải từ trên mạng. Một số phần mềm vô hại, nhưng một số có khả năng hiển thị thông tin lên màn hình, cưỡng chế người sử dụng.

#### 7.2.8 Attacker Tool

Là những bộ công cụ tấn công có thể sử dụng để đẩy các phần mềm độc hại vào trong hệ thống. Các bộ công cụ này có khả năng giúp cho kẻ tấn công có thể truy nhập bất hợp pháp vào hệ thống hoặc làm cho hệ thống bị lây nhiễm mã độc hại. Khi được tải vào trong hệ thống bằng các đoạn mã độc hại, Attacker tool có thể chính là một phần của đoạn mã độc đó (ví dụ như trong một trojan) hoặc nó sẽ được tải vào hệ thống sau khi nhiễm. Ví dụ như một hệ thống đã bị nhiễm một loại worm, worm này có thể điều khiển hệ thống tự động kết nối đến một trang web nào đó, tải attacker tool từ trang đó và cài đặt Attacker tool vào hệ thống. Có rất nhiều loại Attacker tool, trong đó thường gặp nhất là backdoor và keylogger

- Backdoor là một thuật ngữ chung chỉ các phần mềm độc hại thường trú và đợi lệnh điều khiển từ các cổng dịch vụ TCP hoặc UDP. Một cách đơn giản nhất, phần lớn các backdoor cho phép một kẻ tấn công thực thi một số hành động trên

máy bị nhiễm như truyền file, dò mật khẩu, thực hiện mã lệnh... Backdoor cũng có thể được xem xét dưới 2 dạng: Zoombie và Remote Administration Tool

+ Zoombie (có thể đôi lúc gọi là bot) là một chương trình được cài đặt lên hệ thống nhằm mục đích tấn công hệ thống khác. Kiểu thông dụng nhất của Zoombie là các Agent dùng để tổ chức một cuộc tấn công DDoS. Kẻ tấn công có thể cài Zoombie vào một số lượng lớn các máy tính rồi ra lệnh tấn công cùng một lúc. Trinoo và Tribe Flood Network là hai Zoombie nổi tiếng.

+ Remote Administration Tool (RAT) là các công cụ có sẵn của hệ thống cho phép thực hiện quyền quản trị từ xa. Tuy nhiên hacker cũng có thể lợi dụng tính năng này để xâm hại hệ thống. Tấn công kiểu này có thể bao gồm hành động theo dõi mọi thứ xuất hiện trên màn hình cho đến tác động vào cấu hình của hệ thống. Ví dụ về công cụ RAT là: Back Orifice, SubSeven...

- Keylogger là phần mềm được dùng để bí mật ghi lại các phím đã được nhấn bằng bàn phím rồi gửi tới hacker. Keylogger có thể ghi lại nội dung của thư điện tử, của văn bản, tên đăng nhập, mật khẩu, thông tin bí mật...Ví dụ một số loại keylogger như: KeySnatch, Spyster...

- Rootkits là tập hợp của các file được cài đặt lên hệ thống nhằm biến đổi các chức năng chuẩn của hệ thống thành các chức năng tiềm ẩn các tấn công nguy hiểm. Ví dụ như trong hệ thống Windows, Rootkit có thể sửa đổi, thay thế file, hoặc thường trú trong bộ nhớ nhằm thay thế, sửa đổi các lời gọi hàm của hệ điều hành. Rootkit thường được dùng để cài đặt các công cụ tấn công như cài backdoor, cài keylogger. Ví dụ về Rootkit là: LRK5, Knark, Adore, Hack Defender.

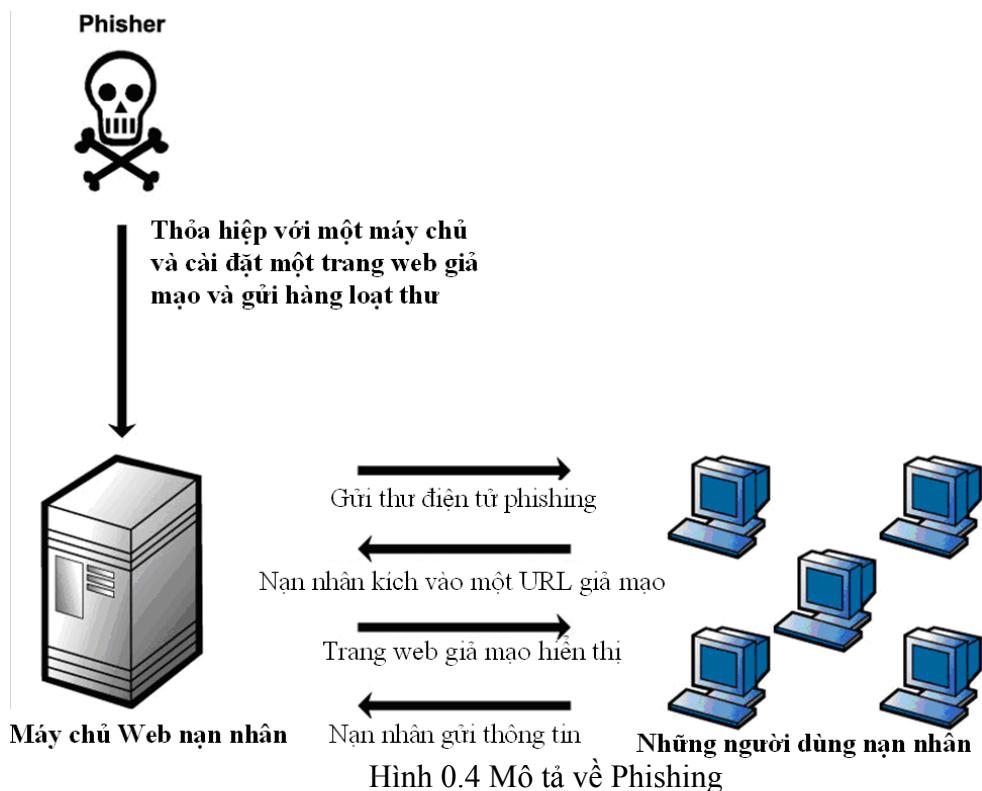
- Web Browser Plug-in là phương thức cài mã độc hại thực thi cùng với trình duyệt web. Khi được cài đặt, kiểu mã độc hại này sẽ theo dõi tất cả các hành vi duyệt web của người dùng (ví dụ như tên trang web đã truy nhập) sau đó gửi thông tin ra ngoài. Một dạng khác là phần mềm gián điệp có chức năng quay số điện thoại tự động, nó sẽ tự động kích hoạt modem và kết nối đến một số điện thoại ngầm định mặc dù không được phép của chủ nhân.

- Thư điện tử Generator là những chương trình cho phép tạo ra và gửi đi một số lượng lớn các thư điện tử. Mã độc hại có thể gieo rắc các Thư điện tử generator vào trong hệ thống. Các chương trình gián điệp, thư rác, mã độc hại có thể được đính kèm vào các thư điện tử được sinh ra từ Thư điện tử generator và gửi tới các địa chỉ có trong sổ địa chỉ của máy bị nhiễm.

- Attacker Toolkit là các bộ công cụ có thể được tải xuống và cài vào hệ thống khi hệ thống đã bị khống chế bởi phần mềm độc hại. Các công cụ kiểu như các bộ dò quét cổng, bộ phá mật khẩu, bộ dò quét gói tin chính là các Attacker Toolkit thường hay được sử dụng.

### 7.2.9 Phishing

Phishing Là một hình thức tấn công thường có thể xem là kết hợp với mã độc hại. Phishing là phương thức dụ người dùng kết nối và sử dụng một hệ thống máy tính giả mạo nhằm làm cho người dùng tiết lộ các thông tin bí mật về danh tính (ví dụ như mật khẩu, số tài khoản, thông tin cá nhân...). Kẻ tấn công phishing thường tạo ra trang web hoặc thư điện tử có hình thức giống hệt như các trang web hoặc thư điện tử mà nạn nhân thường hay sử dụng như trang của ngân hàng, của công ty phát hành thẻ tín dụng... Thư điện tử hoặc trang web giả mạo này sẽ đề nghị nạn nhân thay đổi hoặc cung cấp các thông tin bí mật về tài khoản, về mật khẩu... Các thông tin này sẽ được sử dụng để trộm tiền trực tiếp trong tài khoản hoặc được sử dụng vào các mục đích bất hợp pháp khác.



### 7.2.10 Virus Hoax

Là các cảnh báo giả về virus. Các cảnh báo giả này thường nấp dưới dạng một yêu cầu khẩn cấp để bảo vệ hệ thống. Mục tiêu của cảnh báo virus giả là cố gắng lôi kéo mọi người gửi cảnh báo càng nhiều càng tốt qua email. Bản thân

cảnh báo giả là không gây nguy hiểm trực tiếp nhưng những thư gửi để cảnh báo có thể chứa mã độc hại hoặc trong cảnh báo giả có chứa các chỉ dẫn về thiết lập lại hệ điều hành, xoá file làm nguy hại tới hệ thống. Kiểu cảnh báo giả này cũng gây tốn thời gian và quấy rối bộ phận hỗ trợ kỹ thuật khi có quá nhiều người gọi đến và yêu cầu dịch vụ.