
KSTP.Ebook

CHƯƠNG 2

CÁC HỆ MẬT KHÓA BÍ MẬT

Nội dung chính

- 2.1. Giới thiệu về hệ mật khóa bí mật
- 2.2. Các hệ mật thay thế đơn giản
- 2.3. Các hệ mật thay thế đa biểu
  - 2.3.1. Hệ mật thay thế đa biểu
  - 2.3.2. Hệ mật Playfair
  - 2.3.3. Hệ mật Hill
  - 2.3.4. Hệ mật Vigenere
  - 2.3.5. Hệ mật Beaufort
  - 2.3.6. Khoảng giải mã duy nhất của các hệ mật thay thế đa biểu tuần hoàn

Nội dung chính

- 2.4. Các hệ mật thay thế không tuần hoàn
  - 2.4.1. Hệ mật khoá chạy
  - 2.4.2. Hệ mật Vernam
- 2.5. Các hệ mật chuyển vị
- 2.6. Các hệ mật tích
- 2.7. Chuẩn mã dữ liệu (DES)
  - 2.7.1. Thuật toán DES
  - 2.7.2. Các chế độ hoạt động của DES
  - 2.7.3. Double DES và Triple DES
- 2.8. Chuẩn mã dữ liệu tiên tiến (AES)

2.1. Giới thiệu về hệ mật khóa bí mật

- Mã hóa cổ điển là phương pháp mã hóa đơn giản nhất xuất hiện đầu tiên trong lịch sử ngành mã hóa. Thuật toán đơn giản và dễ hiểu. Những phương pháp mã hóa này là cơ sở cho việc nghiên cứu và phát triển thuật toán mã hóa đối xứng được sử dụng ngày nay.
- Mọi thuật toán cổ điển đều là mã khóa đối xứng, vì ở đó thông tin về khóa được chia sẻ giữa người gửi và người nhận. MĐX là kiểu duy nhất trước khi phát minh ra khóa công khai (hệ mã không đối xứng) vào những năm 1970.

2.1. Giới thiệu về hệ mật khóa bí mật

- Mật mã đối xứng sử dụng cùng một khóa cho việc mã hóa và giải mã. Có thể nói MĐX là mã một khóa hay mã khóa riêng hay mã thỏa thuận.
- Hiện nay các MĐX và công khai tiếp tục phát triển và hoàn thiện. Mã công khai ra đời hỗ trợ mã đối xứng chứ không thay thế nó, do đó mã đối xứng đến nay vẫn được sử dụng rộng rãi.
- Có ba phương pháp chính trong mật mã khoá bí mật (mật mã khoá riêng hay mật mã cổ điển):
  - Hoán vị
  - Thay thế
  - Xử lý bit (chủ yếu nằm trong các ngôn ngữ lập trình)
  - Ngoài ra còn có phương pháp hỗn hợp thực hiện kết hợp các phương pháp trên mà điển hình là chuẩn mã dữ liệu (DES – Data Encryption Standard) của Mỹ.

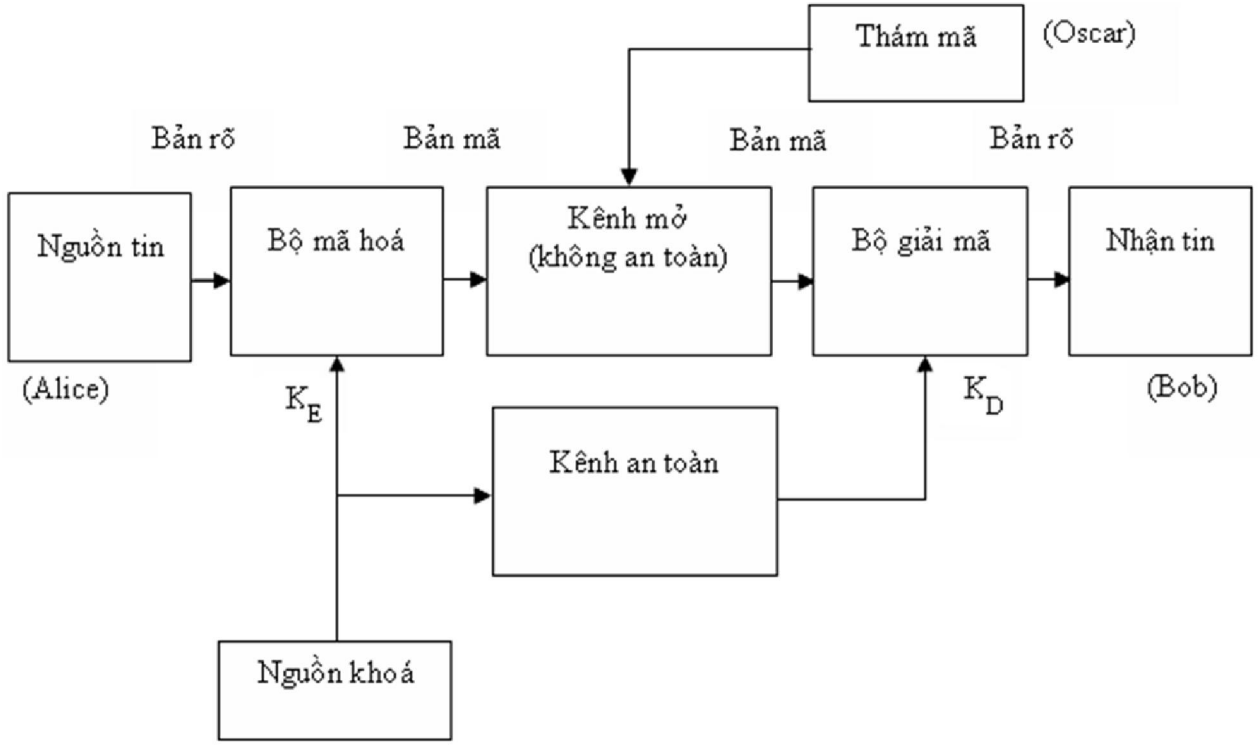
2.1. Giới thiệu về hệ mật khóa bí mật

- Định nghĩa 2.1: *Một hệ mật là bộ 5* $(\mathcal{P}, C, \mathcal{K}, \mathcal{E}, \mathcal{D})$ *thoả mãn các điều kiện sau:*
  - *1)* $\mathcal{P}$ *là tập hữu hạn các bản rõ có thể*
  - *2)* C *là tập hữu hạn các bản mã có thể*
  - *3)* $\mathcal{K}$ *là tập hữu hạn các khoá có thể*
  - *Đối với mỗi* $k \in \mathcal{K}$ *có một quy tắc mã hoá* $e_k \in \mathcal{E}$, $e_k : \mathcal{P} \to C$

  và một quy tắc giải mã tương ứng: $d_k \in \mathcal{D}$, $d_k : C \to \mathcal{P}$ *,sao cho:* $d_k(e_k(x)) = x$ *với* $\forall\ x \in \mathcal{P}$.

2.1. Giới thiệu về hệ mật khóa bí mật

- Sơ đồ khối một hệ mật truyền tin mật:



2.2. Các hệ mật thay thế đơn giản

- Các HMTT đơn biểu
  - Khi khóa đã được chọn thì mỗi kí tự của bản rõ được ánh xạ đến một kí tự duy nhất của bản mã. Do mỗi cách mã hóa như vậy sẽ tương ứng với một hoán vị của bảng chữ và hoán vị đó chính là khóa của mã đã cho. Như vậy độ dài của khóa ở đây là 26 và số khóa có thể có là 26!.
  - *Ví dụ:* Ta có bản mã tương ứng với bản rõ trong bảng chữ đơn như sau:

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

2.2. Các hệ mật thay thế đơn giản

- Mật mã dịch vòng (MDV):

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

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

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

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

Ký 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
Ký 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

2.2. Các hệ mật thay thế đơn giản

- Xét ví dụ: **k =5**; bản rõ: **meetmeatsunset**
- **B1:** Biến bản rõ thành dãy số nguyên theo bảng trên, ta được dãy:

  12.4.4.19.12.4.0.19.18.20.13.18.4.19
- **B2:** Cộng 5 vào mỗi giá trị trên và rút gọn tổng theo mod 26. Ta được dãy:

  17.9.9.24.17.9.5.24.23.25.18.23.9.24
- **B3:** Biến dãy số ở B2 thành kí tự tương ứng. Ta được bản mã: **RJJYRJFYXZSXJY**

2.2. Các hệ mật thay thế đơn giản

- Mã thay thế (MTT)

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

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

và $$d_\pi(y) = \pi^{-1}(y)$$

trong đó π^{-1} là hoán vị ngược của π

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

- *Ví dụ:* với phép TT trên, từ bản rõ: **meetmeatsunset.**
  Ta thu được bản mã: **THHMTHXMVUSVHM**

2.2. Các hệ mật thay thế đơn giản

- Tính an toàn của mã trên bảng chữ đơn. Tổng cộng có 26! Xấp xỉ khoảng $4x10^{26}$ khóa. Với khá nhiều khóa vậy nhiều người nghĩ rằng mã trên bảng chữ đơn sẽ an toàn. Nhưng không phải vậy!
- Vấn đề ở đây là do:
  - Các đặc trưng về ngôn ngữ, tần suất xuất hiện của các chữ trong bản rõ và chữ tương ứng trong bản mã là như nhau

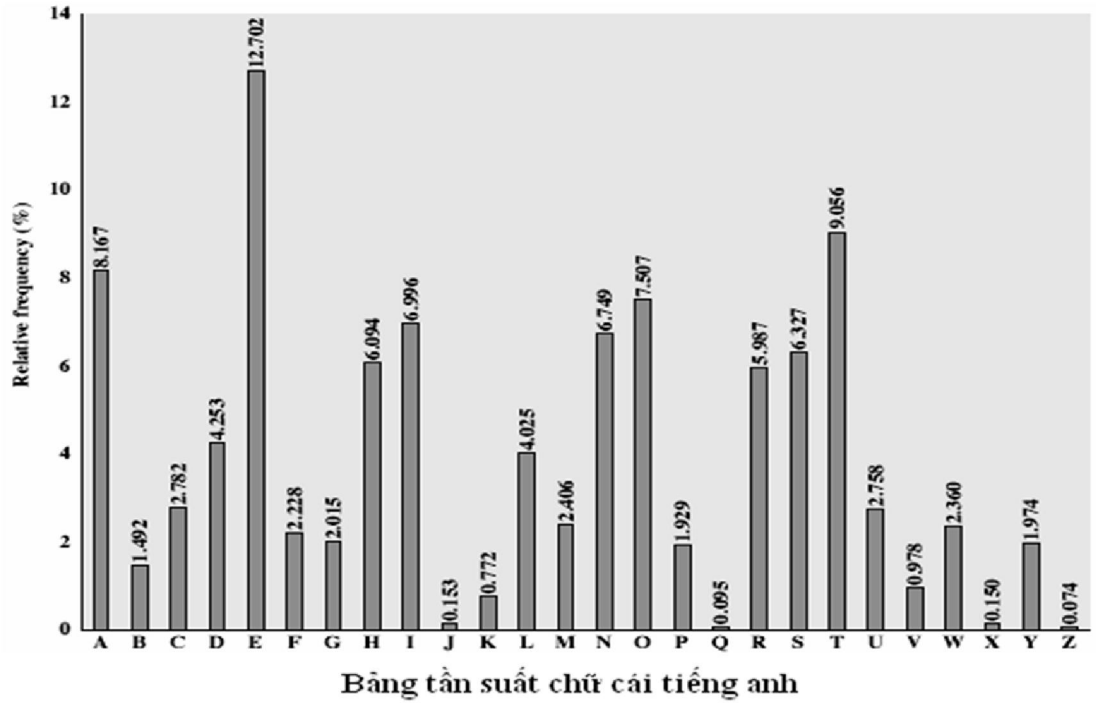
  $\Rightarrow$ Nên thám mã có thể suy đoán được ánh xạ của một số chữ và từ đó dò tìm ra chữ mã cho các chữ khác.

2.2. Các hệ mật thay thế đơn giản

- Tính dư thừa của ngôn ngữ và thám mã. Ngôn ngữ của loài người là dư thừa.Có một số chữ hoặc các cặp chữ hoặc bộ ba chữ được dùng thường xuyên hơn các bộ chữ cùng độ dài khác. Chẳng hạn như các bộ chữ sau đây trong tiếng Anh "th lrd s m shphrd shll nt wnt".
- Tóm lại trong nhiều ngôn ngữ các chữ không được sử dụng thường xuyên như nhau. Trong tiếng Anh chữ E được sử dụng nhiều nhất; sau đó đến các chữ T, R, N, I, O, A, S. Một số chữ rất ít dùng như: Z, J, K, Q, X.
- Bằng phương pháp thống kê, ta có thể xây dựng các bảng các tần suất các chữ đơn, cặp chữ, bộ ba chữ.

2.2. Các hệ mật thay thế đơn giản

- Sử dụng bảng tần suất vào việc thám mã vì mã thế trên bảng chữ đơn không làm thay đổi tần suất tương đối của các chữ, có nghĩa là ta vẫn có bảng tần suất trên nhưng đối với bảng chữ mã tương ứng



2.2. Các hệ mật thay thế đơn giản

- Do đó có cách thám mã trên bảng chữ đơn như sau:
  - Tính toán tần suất của các chữ trong bản mã
  - So sánh với các giá trị đã biết
  - Tìm kiếm các chữ đơn hay dùng A-I-E, bộ đôi NO và bộ ba RST; và các bộ ít dùng JK, X-Z..
  - Trên bảng chữ đơn cần xác định các chữ dùng các bảng bộ đôi và bộ ba trợ giúp
- *Ví dụ:* Thám mã bản mã trên bảng chữ đơn, cho bản mã:

  wklv phvvdjh lv qrw wrr kdug wr euhdn

  - Dự đoán các bộ kí tự hay xuất hiện

  Đoán w và r là T và O.

  Khi đó ta có dự đoán 1 số vị trí trong xâu kí tự rõ là:

  T--- ------- -- -OT TOO ---- TO -----

  ⇒ Suy luận tiếp tục ta có bản rõ:

  THIS MESSAGE IS NOT TOO HARD TO BREAK

2.3. Các hệ mật thay thế đa biểu

- 2.3.1. Hệ mật thay thế đa biểu
- 2.3.2. Hệ mật Playfair
- 2.3.3. Hệ mật Hill
- 2.3.4. Hệ mật Vigenere
- 2.3.5. Hệ mật Beaufort
- 2.3.6. Khoảng giải mã duy nhất của các hệ mật thay thế đa biểu tuần hoàn

2.3.1. Hệ mật thay thế đa biểu

- Yếu điểm của các mã pháp đơn biểu là phân bố tần suất của chúng phản ánh phân bố của bảng chữ cái cơ sở. Một mã pháp an toàn hơn về mặt mật mã sẽ thể hiện phân bố bằng phẳng hơn, điều này sẽ không cho kẻ thám mã chút thông tin nào.
- Một hướng khác làm tăng độ an toàn cho mã trên bảng chữ là sử dụng nhiều bảng chữ để mã. Mỗi chữ sẽ được mã bằng bất kì chữ nào trong bản mã tùy thuộc vào ngữ cảnh khi mã hóa. Làm như vậy để trải bằng tần suất các chữ xuất hiện trong bản mã. Do đó làm mất bớt cấu trúc của bản rõ được thể hiện trên bản mã và làm cho mã thám đa bảng khó hơn.

2.3.1. Các hệ mật thay thế đa biểu

- Ví dụ: Để san bằng phân bố ta kết hợp các chữ cái có phân bố cao với các chữ có phân bố thấp. Nếu chữ cái T đôi lúc được mã là a và lúc khác lại được mã thành b, và X đôi lúc được mã thành a và đôi lúc lại được mã thành b thì tần suất cao của T sẽ trộn với tần suất thấp của X sẽ tạo ra phân bố vừa phải hơn đối với a và b

- Ta sử dụng khóa để chỉ rõ chọn bảng nào được dùng cho từng chữ trong bản tin.
- Độ dài khóa là chu kì lặp của các bảng chữ. Độ dài càng lớn và nhiều chữ khác nhau được sử dụng trong từ khóa thì càng khó thám mã.

2.3. 2. Hệ mật Playfair

- Mã Playfair
  - Như chúng ta đã thấy không phải số khoá lớn trong mã bảng chữ đơn đảm bảo an toàn mã. Một trong các hướng khắc phục là mã bộ các chữ, tức là mỗi chữ sẽ được mã bằng một số chữ khác nhau tùy thuộc vào các chữ mà nó đứng cạnh.
  - Playfair là một trong các mã như vậy, được sáng tạo bởi Charles Wheastone vào năm 1854 và mang tên người bạn là Baron Playfair.
  - Ma trận khoá Playfair. Cho trước một từ làm khoá, với điều kiện trong từ khoá đó không có chữ cái nào bị lặp. Ta lập ma trận Playfair là ma trận cỡ 5 x 5 dựa trên từ khoá đã cho và gồm các chữ trên bảng chữ cái, được sắp xếp theo thứ tự nhất định.

2.3. 2. Hệ mật Playfair

- Quy tắc sắp xếp:
  - Trước hết viết các chữ của từ khoá vào các hàng của ma trận bắt từ hàng thứ nhất.
  - Nếu ma trận còn trống, viết các chữ khác trên bảng chữ cái chưa được sử dụng vào các ô còn lại. Có thể viết theo một trình tự qui ước trước, chẳng hạn từ đầu bảng chữ cái cho đến cuối.
  - Vì có 26 chữ cái tiếng Anh, nên thiếu một ô. Thông thường ta dồn hai chữ nào đó vào một ô chung, chẳng hạn I và J.

2.3. 2. Hệ mật Playfair

- Giả sử sử dụng từ khoá **MONARCHY**. Lập ma trận khoá Playfair tương ứng như sau:

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

- Cách mã hóa và giải mã:
  - Chia bản rõ thành từng cặp chữ. Nếu một cặp nào đó có hai chữ như nhau, thì ta chèn thêm một chữ lọc chẳng hạn X. Ví dụ, trước khi mã "**balloon**" biến đổi thành "**ba lx lo on**".

2.3. 2. Hệ mật Playfair

- Nếu cả hai chữ trong cặp đều rơi vào cùng một hàng, thì mã mỗi chữ bằng chữ ở phía bên phải nó trong cùng hàng của ma trận khóa (cuộn vòng quanh từ cuối về đầu), chẳng hạn "**ar**" biến đổi thành "**RM**"

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

- Nếu cả hai chữ trong cặp đều rơi vào cùng một cột, thì mã mỗi chữ bằng chữ ở phía bên dưới nó trong cùng cột của ma trận khóa (cuộn vòng quanh từ cuối về đầu), chẳng hạn "**mu**" biến đổi thành "**CM**"

2.3. 2. Hệ mật Playfair

- Trong các trường hợp khác, mỗi chữ trong cặp được mã bởi chữ cùng hàng với nó và cùng cột với chữ cùng cặp với nó trong ma trận khóa. Chẳng hạn, **"hs"** mã thành **"BP"**, và **"ea"** mã thành **"IM"** hoặc **"JM"** (tuỳ theo sở thích)

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

2.3. 2. Hệ mật Playfair

- An toàn của mã Playfair:
  - An toàn được nâng cao so hơn với bảng đơn, vì ta có tổng cộng 26 x 26 = 676 cặp. Mỗi chữ có thể được mã bằng 7 chữ khác nhau, nên tần suất các chữ trên bản mã khác tần suất của các chữ cái trên văn bản tiếng Anh nói chung.
  - Muốn sử dụng thống kê tần suất, cần phải có bảng tần suất của 676 cặp để thám mã (so với 26 của mã bảng đơn). Như vậy phải xem xét nhiều trường hợp hơn và tương ứng sẽ có thể có nhiều bản mã hơn cần lựa chọn. Do đó khó thám mã hơn mã trên bảng chữ đơn.
  - Mã Playfair được sử dụng rộng rãi nhiều năm trong giới quân sự Mỹ và Anh trong chiến tranh thế giới thứ 1. Nó có thể bị bẻ khoá nếu cho trước vài trăm chữ, vì bản mã vẫn còn chứa nhiều cấu trúc của bản rõ.

2.3.3. Hệ mật Hill

- Ý tưởng: là lấy m tổ hợp tuyến tính của m kí tự của một phần tử bản rõ để tạo ra một phần tử m kí tự bản mã.
- Mô tả:

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

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

Với một khoá $k \in \mathcal{K}$, ta xác định:

$$e_k(x) = xk$$

và

$$d_k(y) = yk^{-1}$$

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

2.3.3. Hệ mật Hill (tiếp)

- Ví dụ: Giả sử cho khóa: $k = \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$ sử dụng mật mã Hill với bản rõ "**July**"
  - Ta thấy rằng ma trận có cỡ 2 × 2 nên bản rõ sẽ được chia thành các phần tử, mỗi phần tử chứa 2 kí tự như sau: "**ju**" tương ứng với (x1, x2) = (9, 20) và "**ly**" tương ứng với (x3, x4) = (11, 24)
  - Mã hóa:

$$y = (y_1, y_2) = (x_1, x_2)k = \begin{pmatrix} 9 & 20 \end{pmatrix}\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = \begin{pmatrix} 99+60 & 72+140 \end{pmatrix} = \begin{pmatrix} 3 & 4 \end{pmatrix}$$

$$y = (y_3, y_4) = (x_3, x_4)k = \begin{pmatrix} 11 & 24 \end{pmatrix}\begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix} = \begin{pmatrix} 121+72 & 88+168 \end{pmatrix} = \begin{pmatrix} 11 & 22 \end{pmatrix}$$

2.3.3. Hệ mật Hill (tiếp)

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

    Kiểm tra thấy rằng det(k) = 11 × 7 – 3 × 8 mod 26 = 1, rõ ràng ucln(26, det(k)) = 1, vậy k khả nghịch trên Z_{26}
  - Khi đó:

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

2.3.3. Hệ mật Hill (tiếp)

- Nếu như tấn công hệ mật Hill chỉ biết bản mã thì rất khó nhưng nếu tấn công biết bản rõ thì lại không khó.
- Trước tiên hãy giả sử kẻ tấn công đã biết được giá trị m. Giả sử anh ta có ít nhất m cặp bản rõ và bản mã khác nhau là:

  $x_j = (x_{1j}, x_{2j}, \ldots, x_{mj})$ và

  $yj = (y_{1j}, y_{2j}, \ldots, y_{mj})$

  Sao cho $y_j = e_K(x_j)$, $1 \le j \le m$.
- Nếu xây dựng hai ma trận m × m là $X = (x_{i,j})$ và $Y = (y_{i,j})$, thì chúng ta có phương trình ma trận $Y = XK$, trong đó ma trận khóa K cỡ m × m chưa biết
- Ma trận X có nghịch đảo và kẻ tấn công có thể tính $K = X^{-1}Y$ và do đó phá được hệ mật.

  $\Rightarrow$ Kẻ tấn công sẽ làm gì khi không biết m?

2.3.4. Hệ mật Vigenere

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

2.3.4. Hệ mật Vigenere (tiếp)

- Cách làm:
  - Giả sử khoá là một chữ có độ dài d được viết dạng
    $K = K_1K_2\ldots K_d$, trong đó K_i nhận giá trị nguyên từ 0 đến 25.
  - Ta chia bản rõ thành các khối gồm d chữ. Mỗi chữ thứ i trong khối chỉ định dùng bảng chữ thứ i với tịnh tiến là K_i giống như trong mã Ceasar.
  - Trên thực tế khi mã ta có thể sử dụng lần lượt các bảng chữ và lặp lại từ đầu sau d chữ của bản rõ. Vì có nhiều bảng chữ khác nhau, nên cùng một chữ ở các vị trí khác nhau sẽ có các bước nhảy khác nhau, làm cho tần suất các chữ trong bản mã dãn tương đối đều.
  - Giải mã đơn giản là quá trình làm ngược lại. Nghĩa là dùng bản mã và từ khoá với các bảng chữ tương ứng, nhưng với mỗi chữ sử dụng bước nhảy lui lại về đầu

2.3.4. Hệ mật Vigenere (tiếp)

- *Ví dụ:*
  - Giả sử d=6 và từ khóa là **CIPHER**, từ khóa này tương ứng với dãy số:

$$k = (2, 8, 15, 7, 4, 17)$$

  - Giả sử bản rõ: **meetmeatsunset**. Chuyển các kí tự rõ thành mã trên Z_{26} rồi cộng với từ khóa

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

  - Ta nhận được bản mã tương ứng:

OMTAQVCBHBRJGB

2.3.4. Hệ mật Vigenere (tiếp)

- Trên thực tế để hỗ trợ mã Vigenere, người ta đã tạo ra trang Saint – Cyr để trợ giúp cho việc mã và giải mã thủ công.
- Đó là một bảng cỡ 26 x 26 có tên tương ứng là các chữ cái trong bảng chữ tiếng Anh. Hàng thứ i là tịnh tiến i chữ của bảng chứ cái. Khi đó chữ ở cột đầu tiên chính là khoá của bảng chữ ở cùng hàng. Do đó chữ mã của một chữ trong bản rõ nằm trên cùng cột với chữ đó và nằm trên hàng tương ứng với chữ khoá.

2.3.4. Hệ mật Vigenere (tiếp)

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

Bảng Saint Cyr

2.3.4. Hệ mật Vigenere (tiếp)

- **An toàn của mã Vigenere.**
  - Như vậy có chữ mã khác nhau cho cùng một chữ của bản rõ. Suy ra tần suất của các chữ bị là phẳng, nghĩa là tần suất xuất hiện các chữ trên bản mã tương đối đều nhau. Tuy nhiên chưa mất hoàn toàn, do độ dài của khoá có hạn, nên có thể tạo nên chu kỳ vòng lặp.
  - Kẻ thám mã bắt đầu từ tần suất của chữ để xem có phải đây là mã đơn bảng chữ hay không. Giả sử đây là mã đa bảng chữ, sau đó xác định số bảng chữ trong từ khoá (dùng phương pháp Kasiski) và lần tìm từng chữ. Như vậy cần tăng độ dài từ khoá để tăng số bảng chữ dùng khi mã để "là" tần suất của các chữ.

2.3.4. Hệ mật Vigenere (tiếp)

- Phương pháp Kasiski:
  - Dựa trên quy luật tiếng anh: không chỉ các chữ cái mà các nhóm chữ cái lẫn các từ đầy đủ đều lặp lại
    - Ví dụ:
      - Các từ kết thúc bằng: s, -th, -ed, -ion, -tion, …
      - Bắt đầu bằng kí tự: im-, in-, un-,…
      - Các từ: of, and, with, are, is, that, … xuất hiện với tần suất cao
  - Tuân theo quy tắc: nếu một thông báo được mã bằng n bảng chữ cái luân phiên theo chu kì, và nếu một từ hay một nhóm chữ cái cụ thể xuất hiện k lần trong một thông báo rõ thì nó sẽ được mã xấp xỉ k/n lần từ cùng một bảng chữ cái.

2.3.4. Hệ mật Vigenere (tiếp)

- Ví dụ:
  - Nếu từ khóa dài 6 kí tự thì chỉ có 6 cách khác nhau để đặt từ khóa trên từ của các bản rõ.
  - Một từ hay nhóm kí tự trong bản rõ mà xuất hiện hơn 6 lần phải được mã ít nhất 2 lần theo cùng vị trí từ khóa và những lần xuất hiện đó đều được mã như nhau.
- Biện pháp Kasiskis được dùng trên các đoạn đúp trong bản mã. Để cụm từ của bản rõ được mã 2 lần theo cùng cách, khóa phải đi hết toàn bộ số vòng quay và trở ngược đến cùng điểm. Bởi vậy khoảng cách giữa các mẫu lặp lại phải là bội của độ dài từ khóa.
- Để dùng phương pháp này ta phải nhận diện tất cả các mẫu lặp trong bản mã. Thường xét mẫu lặp có trên 3 kí tự.

2.3.4. Hệ mật Vigenere (tiếp)

- Phương pháp Kasiski gồm các bước sau:
  - Nhận diện các mẫu bị lặp có 3 kí tự hoặc hơn.
  - Với mỗi mẫu, ghi ra vị trí bắt đầu mỗi thể hiện của mẫu.
  - Tính khoảng cách giữa các điểm bắt đầu của các thể hiện kế tiếp nhau.
  - Xác định tất cả các tham số cho mỗi khoảng cách.
  - Nếu dùng phép thế đa biểu, độ dài khóa sẽ là một trong các tham số thường xuất hiện trong bước 4.

2.3.5. Hệ mật Beaufort

- Hệ mật Beaufort do Francis Beaufort tạo ra.
- HM này cũng dùng bảng aphabe giống HM Vigenere
- Tuy nhiên thuật toán mã hóa thì khác với Vigenere

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

2.3.5. Hệ mật Beaufort (tiếp)

- Thuật toán:
  - Để mã hóa 1 từ trong bản rõ, ta phải tìm từ rõ đó trên hàng đầu tiên của bảng anphabe. Giả sử x(0,j) là vị trí (hàng thứ 0, cột thứ j) của từ rõ x cần mã.
  - Ta dóng thẳng xuống theo cột j cho tới khi gặp từ khóa. Giả sử từ khóa ở vị trí (hàng thứ i, cột thứ j) k(i,j).
  - Khi đó để tìm từ mã tương ứng của từ rõ x(0,j) ta dóng ngang sang bên trái, vị trí (hàng thứ i, cột thứ 0) y(i,0) chính là từ mã tương ứng với từ rõ x(0,j) cần tìm.

2.3.5. Hệ mật Beaufort (tiếp)

- **Ví dụ:** Key: **FORTIFICATION**

  Bản rõ: **DEFEND THE EAST WALL OF THE CASTLE**

Khóa	F	O	R	T	I	F	I	C	A	T	I	O	N	F	O	R	T	I	F	I	C	A	T	I	O	N	F	O
Bản rõ	D	E	F	E	N	D	T	H	E	E	A	S	T	W	A	L	L	O	F	T	H	E	C	A	S	T	L	E

- Kí tự rõ "D" trong bản mã nằm tại vị trí (0,3), dóng xuống theo cột 3 cho tới khi gặp kí tự "F" của khóa tại vị trí (2,3). Khi đó từ mã tương ứng của "D" sẽ là từ "C" tại vị trí (2,0). Tương tự như vậy ta sẽ thu được bản mã tương ứng là:

  CKMPVCPVWPIWUJOGIUAPVWRIWUUK

2.3.6. Khoảng giải mã duy nhất của các HMTT đa biểu tuần hoàn

- Ta có $|K| = 26!$; $H(K)_{max} = \log 26! = 88{,}4$ bit
- Khi nào thì thám mã thu được lượng thông tin đủ để phát hiện ra khóa?
  - Để giải mã $\Rightarrow$ thám mã căn cứ vào thông tin nhận được từ bản mã C. Khi lượng thông tin $\geq H(K) \Rightarrow$ thám mã thành công
- Trong hệ mật thay thế thì độ dư của bản rõ = bản mã.

2.3.6. Khoảng giải mã duy nhất …

- Ta có: $H(P)_{max} = \log 26 = 4.76$ bit (đạt được khi ghép ngẫu nhiên các kí tự)
- Trong ngôn ngữ tiếng anh:
  - H(P) = 4.19 bit; $H(C) = \lim_{n \to \infty} \frac{H(P^n)}{n}$; $H(P^n) = \underbrace{H(P.P.P...P)}_{n \text{ lần}}$
  - Ngôn ngữ độc lập: $H(P^n) = nH(P)$
  - 1 bit < H(C) < 1.5 bit
  - Độ dư: $D = H(P)_{max} - H(C) \approx 3.5$ bit

$\Rightarrow$ Khoảng duy nhất: $N = \frac{H(K)_{max}}{D} = 25$ kí tự

2.4. Các HMTT không tuần hoàn

- 2.4.1. Hệ mật khoá chạy
- 2.4.2. Hệ mật Vernam

2.4. Các HMTT không tuần hoàn

- Phép thế lý tưởng sẽ dùng nhiều bảng cái để không thể nhận diện được phân bố và không có mẫu trong suốt đối với việc lựa chọn một bảng chữ cái tại một điểm cụ thể.
- Điều gì xảy ra nếu một văn bản được mã bằng số bảng không hạn chế?
- Một dãy vô hạn *không lặp lại* sẽ làm hỏng phương pháp Kasiski.
  - Thứ nhất: Một câu bản rõ lặp lại sẽ không được mã theo cùng cách hai lần vì không có sự lặp lại mẫu trong việc chọn các bảng chữ cái.
  - Thứ hai: Giả sử một đoạn bản mã nào đó là đoạn lặp lại của đoạn trước đó, đoạn lặp lại hầu như chắc chắn là tình cờ. Khoảng cách giữa hai đoạn đúp rõ sẽ không chỉ rõ chu kì trong mẫu mã hóa vì không có mẫu nào và vì thế không có chu kì.

2.4. Các HMTT không tuần hoàn

- Như vậy, việc lựa chọn không lặp các bảng chữ cái mã gây khó khăn cho phân tích mã.
- Trong phần này ta sẽ đi tìm hiểu một số hệ mật thay thế "hoàn hảo".

2.4.1. Hệ mật khoá chạy

- Lý tưởng nhất là ta có khoá dài như bản tin.
- Vigenere đề xuất khoá tự động sinh cho bằng độ dài bản tin như sau:
  - Từ khoá được nối tiếp bằng chính bản rõ để tạo thành khoá. Sau đó dùng mã Vigenere để mã bản rõ đã cho.
  - Khi đó biết từ khoá có thể khôi phục được một số chữ ban đầu của bản rõ. Sau đó tiếp tục sử dụng chúng để giải mã cho văn bản còn lại.
  - Sự cải tiến này làm mất khái niệm chu kỳ, gây khó khăn cho việc thám mã, nhưng vẫn còn đặc trưng tần suất để tấn công.

2.4.1. Hệ mật khoá chạy

- **Ví dụ:** Cho key: **deceptive**

  Plaintext: **wearediscoveredsaveyourself**

Key	d	e	t	e	c	t	i	v	e	w	e	a	r	e	d	i	s	c	o	v	e	r	e	d	s	a	v
Plaintext	w	e	a	r	e	d	i	s	c	o	v	e	r	e	d	s	a	v	e	y	o	u	r	s	e	l	f
Ciphertext	Z	I	C	V	T	W	Q	N	G	K	Z	E	I	I	G	A	S	X	S	T	S	L	V	V	W	L	A

- Ta nhận được bản mã tương ứng là:

ZICVTWQNGKZEIIGASXSTSLVVWLA

2.4.2. Hệ mật Vernam

- Hệ mật Vernam (OTP)
  - Giả sử n 1 là một số nguyên và P = C = K = $(Z_2)^n$. Với K ϵ $(Z_2)^n$, ta xác định $e_K(x)$ là tổng vec tơ theo modulo 2 của K và x (tương đương với phép hoặc loại trừ của hai dãy bit). Như vậy, nếu x = (x1,x2,…,xn) và K= (K1,K2,…,Kn) thì:

$$e_K(x) = (x1+K1,x2+K2,\ldots,xn+Kn) \bmod 2$$

  - Phép mã hóa là đồng nhất với phép giải mã, tức là nếu y = (y1, y2, …, yn) thì:

$$d_K(y) = (y1 + K1, y2 + K2, \ldots, yn + Kn) \bmod 2.$$

2.5. Các hệ mật chuyển vị

- Trong các mục trước chúng ta đã xét một số MTT, ở đó các chữ của bản rõ được thay thế bằng các chữ khác của bản mã.
- Bây giờ chúng ta xét đến loại mã khác, mã hoán vị (MHV), các chữ trong bản rõ không được thay thế bằng các chữ khác mà chỉ thay đổi vị trí, tức là việc mã hoá chỉ dịch chuyển vị trí tương đối giữa các chữ trong bản rõ.
- Như vậy, nó dấu bản rõ bằng cách thay đổi thứ tự các chữ, nó không thay đổi các chữ thực tế được dùng. Do đó bản mã có cùng phân bố tần suất xuất hiện các chữ như bản gốc. Như vậy có thể thám mã để phát hiện được.

2.5. Các hệ mật chuyển vị

- Định nghĩa MHV:

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

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

Đối với một khoá π (tức là một phép hoán vị nào đó), ta xác định:

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

và $$d_\pi = (x_1, ..., x_m) = (y_{\pi^{-1}(1)}, ..., y_{\pi^{-1}(m)})$$

trong đó π^{-1} là phép hoán vị ngược của π

2.5. Các hệ mật chuyển vị

- **Ví dụ 1:** m =6; khóa là phép hoán vị sau:

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

Khi đó phép HV ngược:

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

Bản rõ: **asecondclasscarriageonthetrain**

2.5. Các hệ mật chuyển vị

- Mã hóa:
  - **B1:** Nhóm bản rõ thành các nhóm 6 kí tự

asecon	dclass	carria	geonth	etrain

  - **B2:** Mỗi nhóm 6 kí tự sẽ được sắp xếp lại theo theo phép HV π (3, 5, 1, 6, 4, 2), ta có:

EOANCS	LSDSAC	RICARA	OTGHNE	RIENAT

  - Khi đó ta có bản mã:

EOANCSLSDSACRICARAOTGHNERIENAT

2.5. Các hệ mật chuyển vị

- Giải mã:
  - **B1:** Nhóm bản mã thành các nhóm 6 kí tự

EOANCS	LSDSAC	RICARA	OTGHNE	RIENAT

  - **B2:** Mỗi nhóm 6 kí tự sẽ được sắp xếp lại theo theo phép HV π^{-1} (3, 6, 1, 5, 2, 4), ta có:

asecon	dclass	carria	geonth	etrain

  - Khi đó ta có bản rõ tương ứng:

asecondclasscarriageonthetrain

2.5. Các hệ mật chuyển vị

- **Ví dụ 2:** m = 7

  Key: 4 3 1 2 5 6 7 (khóa là thứ tự các cột)

  Bản rõ: **attackpostponeduntiltwoam**

- Cách mã:
  - Viết các chữ của bản tin theo các dòng với số cột xác định. Nếu các chữ trên dòng không trải đủ số cột thì ta thêm các kí tự khác vào cho đủ chẳng hạn x, y, z, …
  - Thay đổi thứ tự các cột theo dãy số khóa cho trước
  - Đọc lại chúng theo các cột để nhận được bản mã.

2.5. Các hệ mật chuyển vị

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

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

TTNAAPTMTSUOAODWCOIXKNLYPETZ

- Cách giả mã được thực hiện ngược lại

2.6. Các hệ mật tích

- Một phát minh do Shannon đưa ra năm 1949
- 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 này được gọi là tự đồng cấu).

2.6. Các hệ mật tích

- Giả sử: $S_1 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_1, \mathcal{E}_1, \mathcal{D}_1)$ và $S_2 = (\mathcal{P}, \mathcal{P}, \mathcal{K}_2, \mathcal{E}_2, \mathcal{D}_2)$ là 2 hệ mật tự đồng cấu có cùng không gian bản rõ và mã
- Khi đó, tích của S_1 và S_2 (kí hiệu là $S_1 x S_2$) được xác định là hệ mật sau:

$$(\mathcal{P}, \mathcal{P}, \mathcal{K}_1 \times \mathcal{K}_2, \mathcal{E}, \mathcal{D})$$

2.6. Các hệ mật tích

- Hệ mật tích: $(\mathcal{P}, \mathcal{P}, \mathcal{K}_1 \times \mathcal{K}_2, \mathcal{E}, \mathcal{D})$
- Trong đó:
  - Khóa của HMT có dạng $k = (k_1, k_2)$, với $k_1 \in \mathcal{K}_1$ và $k_2 \in \mathcal{K}_2$
  - Các quy tắc mã và giải mã của HMT 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 hoá x bằng e_{k1} rồi mã lại bản kết quả bằng e_{k2}. Quá trình giải mã tương tự nhưng thực hiện theo thứ tự ngược lại

2.6. Các hệ mật tích

- Phân bố xác suất cho không gian khoá K của hệ mật tích được xác định như sau:

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

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

2.6. Các hệ mật tích

- Định nghĩa HMT:

Giả sử $\mathcal{P} = \mathcal{C} = Z_{26}$ và giả sử:

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

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

và $d_a(y) = a^{-1} y \bmod 26$

2.6. Các hệ mật tích

- **Ví dụ:** Cho M là HMT và S là MDV (với các khóa chọn đồng xác suất). Chứng minh rằng MxS và SxM chính là hệ mã Affine (cùng với các khóa chọn đồng xác suất).
- Trước khi đi vào CM ta sẽ tìm hiểu về mật Affine:
  - Mã Affine là một trường hợp đặc biệt của MTT. Hàm mã có dang: $e(x) = ax + b \bmod 26$. Với a, b $\in Z_{26}$. Khi a =1 ta có MDV.

2.6. Các hệ mật tích

- Ta có định nghĩa về mã Affine như sau:

Cho $\mathcal{P} = \mathcal{C} = Z_{26}$ và giả sử:

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

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

$$e_k(x) = ax + b \bmod 26$$

và

$$d_k(y) = a^{-1}(y - b) \bmod 26$$

2.6. Các hệ mật tích

- **Ví dụ:** k = (7, 3). Bản rõ "**hot**"

  Ta có hàm mã: $e_k(x) = 7x + 3$

  Hàm giải mã tương ứng: $d_k(x) = 15(y - 3)$

  (vì $7^{-1} \bmod 26 = 15$)
- Biến đổi các chữ của bản rõ thành các thặng dư theo modulo 26 ta được các giá trị tương ứng là 7, 14, 19.
- Bây giờ ta sẽ mã:

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

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

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

- Khi đó 3 kí hiệu của bản mã là 0, 23, 6 tương ứng với xâu kí tự "**AXG**"

2.6. Các hệ mật tích

- Trở lại với ví dụ chứng minh MxS và SxM là hệ mật Affine.
- Chứng minh MxS là hệ mật Affine:
  - Một khóa DV là phần tử $k \in Z_{26}$ và quy tắc mã tương ứng: $e_k(x) = x + k \bmod 26$.
  - Khóa trong HMT là phần tử $a \in Z_{26}$ sao cho:

    UCLN(a, 26) = 1. Quy tắc mã tương ứng là: $e_a(x) = a \bmod 26$.
  - Bởi vậy khóa trong mã tích MxS có dạng (a, k) trong đó:

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

$\Rightarrow$ Đây chính là ĐN về khóa trong hệ Affine

2.6. Các hệ mật tích

- Hơn nữa, xác suất của một khóa trong hệ mã Affine là 1/312 = (1/12)x(1/26). Đó là tích của xác suất tương ứng của các khóa a và k.

⇒ *Bởi vậy MxS là hệ mã Affine.*

- Chứng minh SxM là hệ mật Affine:
  - Một khóa trong hệ mã này có dạng (k,a), trong đó: $e_{(k, a)}(x) = a(x+k) = ax + ak \bmod 26$. Như vậy khóa (k, a) của mã tích SxM đồng nhất với khóa (a, ak) của hệ mã Affine

2.6. Các hệ mật tích

- Ta cần chứng tỏ rằng mỗi khóa của mã Affine xuất hiện cùng với xác suất 1/312 như trong mã tích SxM.
- Thấy rằng $ak = k_1 \Leftrightarrow k = a^{-1}k_1$, (vì UCLN(a, 26) = 1, bởi vậy a có phần tử nghịch đảo). Nói cách khác, khóa (a, k_1) của hệ mã Affine tương đương với khóa $(a^{-1}k_1, a)$ của mã tích SxM. Bởi vậy, ta có một song ánh giữa hai không gian khóa. Vì mỗi khóa là đồng xác suất nên có thể thấy rằng SxM thực sự là mã Affine.

2.7. Chuẩn mã dữ liệu (DES)

- 2.7.1. Thuật toán DES
- 2.7.2. Các chế độ hoạt động của DES
- 2.7.3. Double DES và Triple DES

2.7.1. Thuật toán DES

- DES được IBM phát triển và được công bố lần đầu tiên vào 1975.
- Mô tả DES:
  - 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.
  - Thuật toán tiến hành theo 3 bước:
    - **B1:** 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: $x_0 = IP(x) = L_0R_0$, trong đó L_0 gồm 32 bit đầu và R_0 là 32 bit cuối.

2.7.1. Thuật toán DES

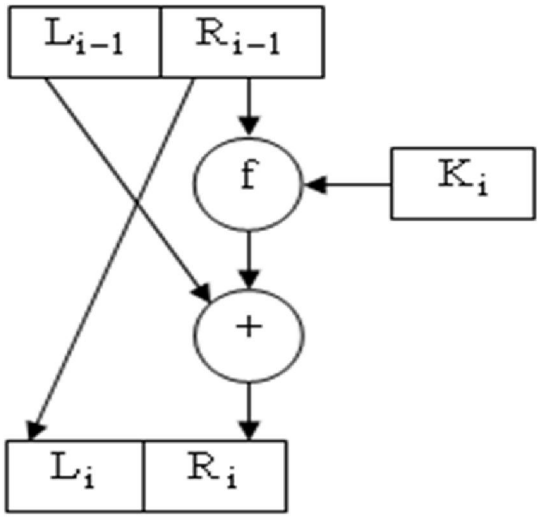
- **B2:** Sau đó tính toán 16 lần lặp theo một hàm xác định. Ta sẽ tính L_iR_i, $1 \leq i \leq 16$ theo quy tắc sau:

$$L_i = R_{i-1}; R_i = L_{i-1} \oplus f(R_{i-1}, k_i)$$

- Trong đó:
  - $\oplus$ là phép loại trừ của hai xâu bit
  - f là một hàm sẽ được mô tả ở sau
  - $k_1, k_2, \ldots, k_{16}$ là các xâu bit có độ dài 48 được tính như 1 hàm của khóa k (k_i chính là một phép chọn hoán vị bit trong k).

2.7.1. Thuật toán DES

- Một vòng của phép mã hóa được mô tả như sau:



- **B3**: 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}

2.7.1. Thuật toán DES

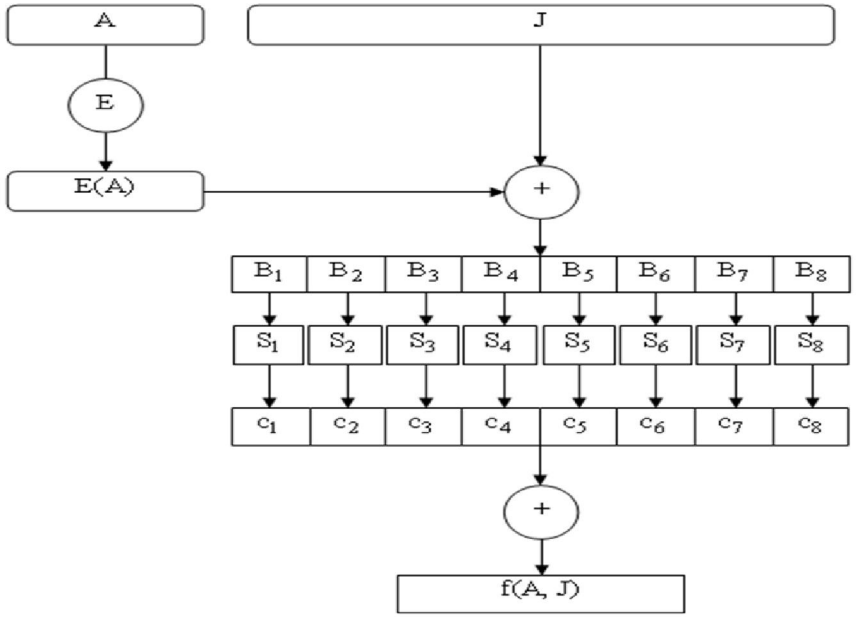
- Mô tả hàm f:
  - Hàm f có 2 biến vào:
    - Xâu bit A có độ dài 32
    - Xâu bit J có độ dài 48
  - Đầu ra của f là xâu bit có độ dài 32.
  - Các bước thực hiện:
    - **B1:** Biến thứ nhất A được mở rộng thành một xâu bit độ dài 48 theo một hàm mở rộng cố định E. E(A) gồm 32 bit của A (được hoán vị theo cách cố định) với 16 bit xuất hiện hai lần.
    - **B2:** Tính E(A) $\oplus$ J và viết kết quả thành một chuỗi 8 xâu 6 bit là $B_1B_2B_3B_4B_5B_6B_7B_8$.

2.7.1. Thuật toán DES

- **B3:** Bước tiếp theo dùng 8 bảng $S_1S_2,\ldots,S_8$ (được gọi là các hộp S). Với mỗi S_i là một bảng 4×16 cố định có các hàng là các số nguyên từ 0 đến 15. Với xâu bit có độ dài 6 (kí hiệu $B_i = b_1\, b_2\, b_3\, b_4\, b_5\, b_6$), ta tính $S_j(B_j)$ như sau:
  - Hai bit b_1b_6 xác định biểu diễn nhị phân hàng r của S_j $(0\leq r \leq 3)$
  - Bốn bit $(b_2\, b_3\, b_4\, b_5)$ 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
  - Bằng cách tương tự tính các $C_j = S_j(B_j)$, $(1 \leq j \leq 8)$.

2.7.1. Thuật toán DES

- **B4:** Xâu bit $C = C_1 C_2 \ldots C_8$ 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).



2.7.1. Thuật toán DES

- Phép hoán vị ban đầu IP:

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 ý nghĩa là bit thứ 58 của x là bit đầu tiên của IP(x); bit thứ 50 của x là bit thứ 2 của IP(x)

2.7.1. Thuật toán DES

- Phép hoán vị ngược IP^{-1}:

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

2.7.1. Thuật toán DES

- 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

2.7.1. Thuật toán DES

- Tám hộp S:

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

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

2.7.1. Thuật toán DES

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

2.7.1. Thuật toán DES

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

2.7.1. Thuật toán DES

- Phép hoán vị P:

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

2.7.1. Thuật toán DES

- Mô tả tính bảng khóa từ khóa k.
  - Trên thực tế k là một xâu bit độ dài 64, trong đó có 56 bit khóa và 8 bit 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 khóa.

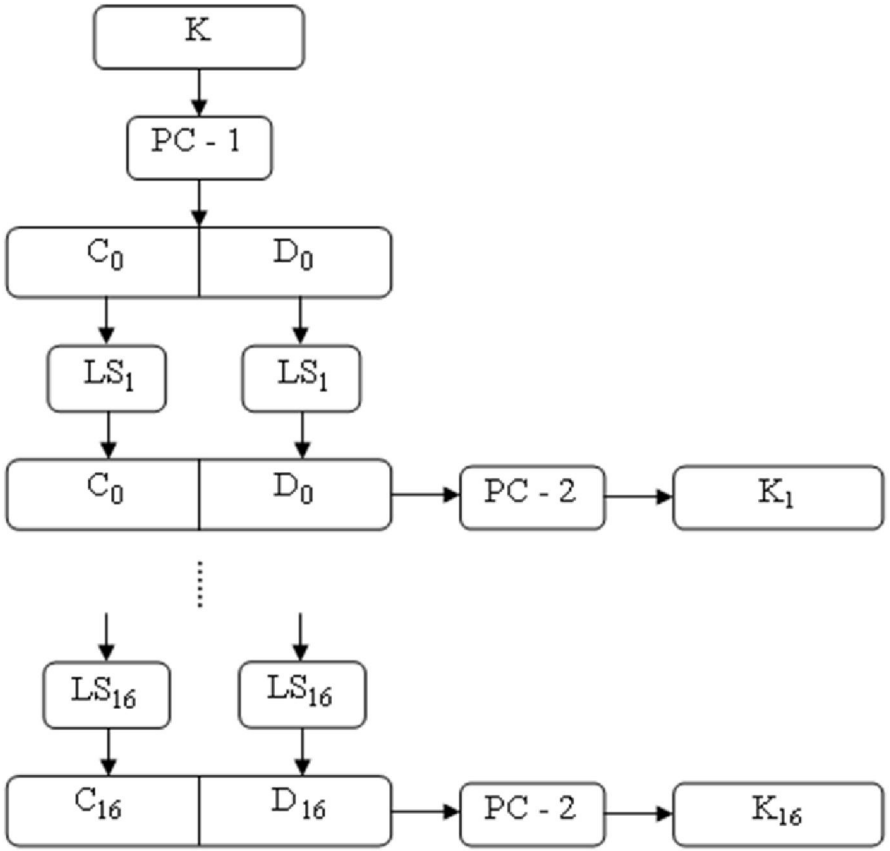
2.7.1. Thuật toán DES

- Các bước tính bảng khóa DES:
  - 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_0D_0
  - Với i thay đổi từ 1 đến 16:

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

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

2.7.1. Thuật toán DES



Tính bảng khóa DES

2.7.1. Thuật toán DES

- Các hoán vị PC-1 và PC-2:

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

2.7.1. Thuật toán DES

- Tính chất của DES
  - Tác dụng đồng loạt: Khi ta thay đổi 1 bit trong khoá sẽ gây ra tác động đồng loạt làm thay đổi nhiều bit trên bản mã. Đây là tính chất mong muốn của khoá trong thuật toán mã hoá. Nếu thay đổi 1 bít đầu vào hoặc khoá sẽ kéo theo thay đổi một nửa số bít đầu ra. Do đó không thể đoán khoá được. Co thể nói rằng DES thể hiện tác động đồng loạt mạnh
  - Sức mạnh của DES – kích thước khoá: Độ dài của khoá trong DES là 56 bít có $2^{56} = 7.2 \times 10^{16}$ giá trị khác nhau. Đây là con số rất lớn nên tìm kiếm duyệt rất khó khăn

2.7.1. Thuật toán DES

- **Sức mạnh của DES – tấn công thời gian:** Đây là dạng tấn công vào cài đặt thực tế của mã. Ở đây sử dụng hiểu biết về quá trình cài đặt thuật toán mà suy ra thông tin về một sô khoá con hoặc mọi khoá con. Đặc biệt sử dụng kết luận là các tính toán chiếm khoảng thời gian khác nhau phụ thuộc vào giá trị đầu vào của nó. Do đó kẻ thám mã theo dõi thời gian thực hiện mà phán đoán về khoá. Có thể kẻ thám mã sáng tạo ra các loại card thông minh phán đoán khoá, mà còn phải bàn bạc thêm về chúng.
- **Sức mạnh của DES – tấn công thám mã:** Có một số phân tích thám mã trên DES, từ đó đề xuất xây dựng một số cấu trúc sâu về mã DES. Rồi bằng cách thu thập thông tin về mã, có thể đoán biết được tất cả hoặc một số khoá con đang dùng. Nếu cần thiết sẽ tìm duyệt những khoá còn lại. Nói chung, đó là những tấn công dựa trên phương pháp thống kê bao gồm: thám mã sai phân, thám mã tuyến tính và tấn công khoá liên kết.

2.7.1. Thuật toán DES

- Thám mã sai phân: đây là phương pháp mạnh để phân tích mã khối
  - Thám mã sai phân so sánh hai cặp mã có liên quan với nhau
    - Với sự khác biệt đã biết ở đầu vào
    - Khảo sát sự khác biệt ở đầu ra
    - Khi với cùng khoá con được dùng
    - Trong công thức sau với hai đầu vào khác nhau, vế trái là sự khác biệt mã ở cùng vòng thứ i được biểu diễn qua sự khác biệt mã ở vòng trước đó i-1 và sự khác biệt của hàm f trong ngoặc vuông.

$$\Delta m_{i+1} = m_{i+1} \oplus m'_{i+1}$$

$$= \left[m_{i-1} \oplus \mathrm{f}(m_i, K_i)\right] \oplus \left[m'_{i-1} \oplus \mathrm{f}(m'_i, K_i)\right]$$

$$= \Delta m_{i-1} \oplus \left[\mathrm{f}(m_i, K_i) \oplus \mathrm{f}(m'_i, K_i)\right]$$

2.7.1. Thuật toán DES

- Sự khác biệt ở đầu vào cho sự khác biệt ở đầu ra với một xác suất cho trước.
  - Nếu tìm được một thể hiện đầu vào - đầu ra với xác suất cao. Thì có thể luận ra khoá con được sử dụng trong vòng đó
  - Sau đó có thể lặp lại cho nhiều vòng (với xác suất giảm dần)
  - Cặp đúng cho bít khoá như nhau
  - Cặp sai cho giá trị ngẫu nhiên
  - Đối với số vòng lớn, xác suất để có nhiều cặp đầu vào 64 bít thoả mãn yêu cầu là rất nhỏ.

2.7.1. Thuật toán DES

- Qui trình thám mã như sau: thực hiện mã hoá lặp lại với cặp bản rõ có XOR đầu vào biết trước cho đến khi nhận được XOR đầu ra mong muốn
- Khi đó có thể tìm được
  - nếu vòng trung gian thỏa mãn XOR yêu cầu thì có cặp đúng
  - nếu không thì có cặp sai, tỷ lệ sai tương đối cho tấn công đã biết trước dựa vào thống kê.
- Sau đó có thể tạo ra các khoá cho các vòng theo suy luận sau

2.7.1. Thuật toán DES

- Thám mã tuyến tính: nó cũng dùng phương pháp thống kê. Cơ sở của phương pháp dựa trên tìm xấp xỉ tuyến tính
  - Tìm xấp xỉ tuyến tính với xác suất p != ½

    $P[i_1, i_2,..., i_a] (+) C[j_1, j_2,..., i_b] = K[k_1, k_2,..., k_c]$

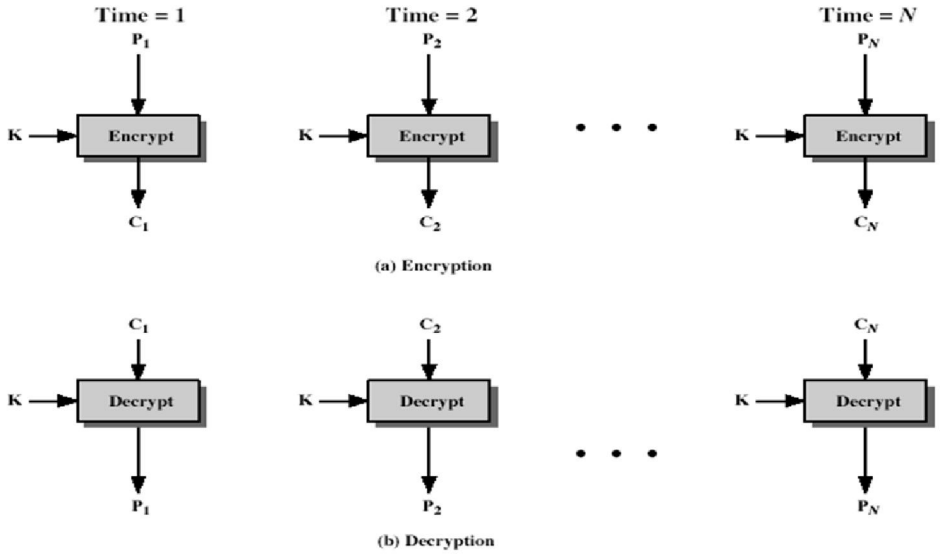
    trong đó i_a, j_b, k_c là các vị trí bit trong bản rõ, mã, khoá.
  - Điều kiện trên cho phương trình tuyến tính của các bít khoá.
  - Để nhận được 1 bít khoá sử dụng thuật toán lân cận tuyến tính
    - Sử dụng một số lớn các phương trình thử nghiệm. Hiệu quả cho bởi $|p - 1/2|$
  - Trong quá trình tìm hiểu DES người ta đã hệ thống lại các tiêu chuẩn thiết kế DES. Như báo cáo bởi Copperscmith trong [COPP94]:
    - Có 7 tiêu chuẩn đối với S box được cung cấp để đảm bảo
      - tính phi tuyến tính
      - chống tham mã sai phân
      - Rối loạn tốt
    - Có 3 tiêu chuẩn cho hoán vị P để tăng độ khuếch tán

2.7.2. 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)
  - Chế độ phản hồi đầu ra (OFB)

2.7.2. Các chế độ hoạt động của DES

- Chế độ quyển mã điện tử (ECB)
  - Mẫu tin được chia thành các khối độc lập, sau đó mã từng khối
  - Mỗi khối là giá trị cần thay thế như dùng sách mã, do đó có tên như vậy
  - Mỗi khối được mã độc lập với các mã khác $C_i = DES_{K1}(Pi)$
  - Khi dùng: truyền an toàn từng giá trị riêng lẻ

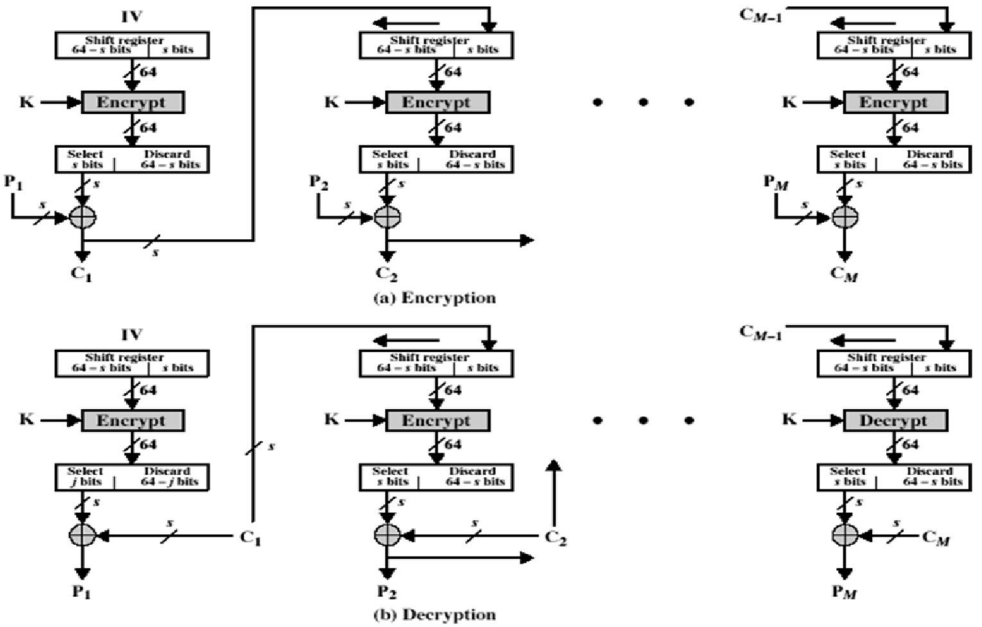


2.7.2. Các chế độ hoạt động của DES

- Ưu và nhược của ECB:
  - Lặp trên bản mã được chỉ rõ lặp trên bản tin
    - Nếu dóng đúng khối
    - Đặc biệt với hình ảnh
    - Hoặc với bản tin mà thay đổi rất ít sẽ trở thành đối tượng để thám mã
  - Nhược điểm là các khối được mã độc lập
  - Được sử dụng chủ yếu khi gửi một ít dữ liệu

2.7.2. Các chế độ hoạt động của DES

- Chế độ phản hồi mã (CFB)
  - Bản tin coi như dòng các bít
  - Bổ sung vào đầu ra của mã khối
  - Kết quả phản hồi trở lại cho giai đoạn tiếp theo, vì vậy có tên như vậy.



  - Nói chung cho phép số bít phản hồi là 1, 8, 64, hoặc tuỳ ý: ký hiệu tương ứng là CFB1, CFB8, CFB64,…
  - Thường hiệu quả sử dụng cả 64 bít

$$C_i = P_i \text{ XOR } DES_{K1}(C_{i-1}); C_{-1} = IV$$

  - Được dùng cho mã dữ liệu dòng, xác thực

2.7.2. Các chế độ hoạt động của DES

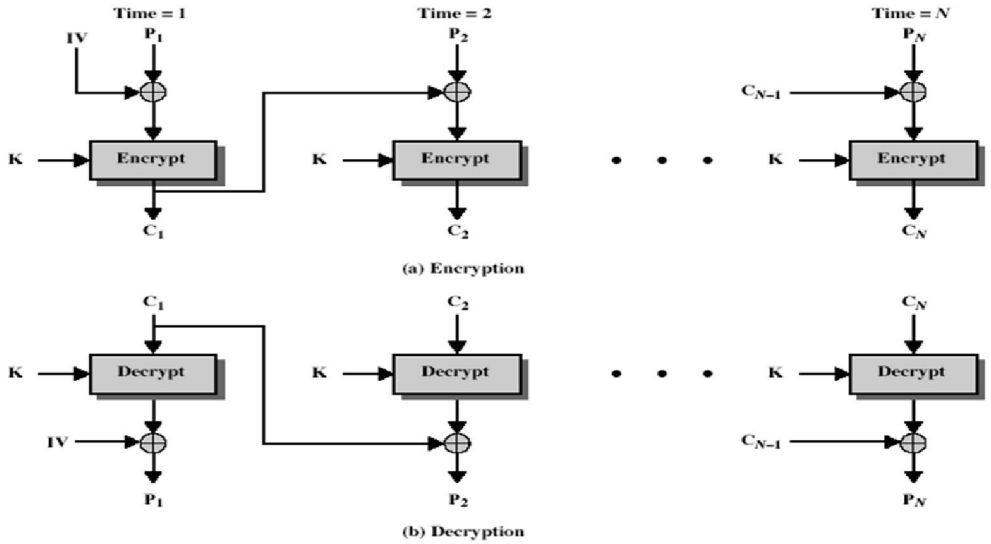
- **Ưu và nhược điểm của CFB**
  - Được dùng khi dữ liệu đến theo byte/bit
  - Chế độ dòng thường gặp nhất
  - Hạn chế là cần ngăn chuồng khi mã khối sau mỗi n bit
  - Nhận xét là mã khối được dùng ở chế độ mã ở cả hai đầu
  - Lỗi sẽ lan ra một vài block sau lỗi

2.7.2. Các chế độ hoạt động của DES

- Chế độ liên kết khối mã (CBC)
  - Các mẫu tin được chia thành các khối
  - Nhưng chúng được liên kết với nhau trong quá trình mã hoá
  - Các block được sắp thành dãy, vì vậy có tên như vậy
  - Sử dụng véctơ ban đầu IV để bắt đầu quá trình

$$C_i = DES_{K1}(P_i \text{ XOR } C_{i-1}); C_{-1} = IV$$

  - Dùng khi: mã dữ liệu lớn, xác thực



2.7.2. Các chế độ hoạt động của DES

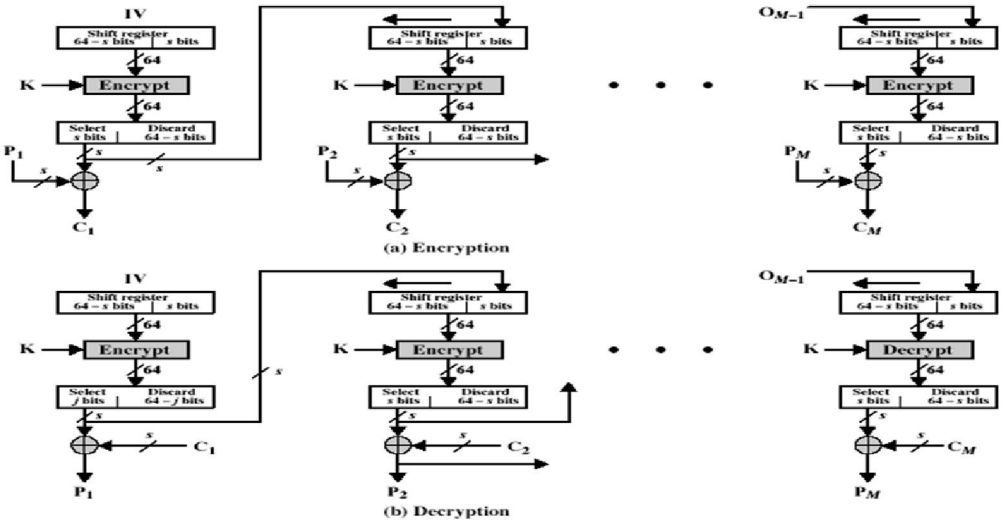
- **Ưu và nhược của CBC**
  - Mỗi khối mã phụ thuộc vào tất cả các khối bản rõ
  - Sự thay đổi của bản tin ở đâu đó sẽ kéo theo sự thay đổi của mọi khối mã
  - Cần giá trị véc tơ ban đầu IV được biết trước bởi người gửi và người nhận
    - Tuy nhiên nếu IV được gửi công khai, kẻ tấn công có thể thay đổi bít đầu tiên và thay đổi cả IV để bù trừ
    - Vậy IV cần phải có giá trị cố định trước hoặc mã hoá trong chế độ ECB và gửi trước phần còn lại của mẩu tin
  - Ở cuối bản tin, để kiểm soát các block ngắn còn lại
    - Có thể bổ sung các giá trị không phải dữ liệu như NULL
    - Hoặc dùng bộ đệm cuối với số byte đếm kích thước của nó.
      - Ví dụ: [b1 b2 b3 0 0 0 0 5] <- 3 data bytes, vậy có 5 bytes dành cho đệm và đếm

2.7.2. Các chế độ hoạt động của DES

- Chế độ phản hồi đầu ra (OFB)
  - Mẩu tin xem như dòng bit
  - Đầu ra của mã được bổ sung cho mẩu tin
  - Đầu ra do đó là phản hồi, do đó có tên như vậy
  - Phản hồi ngược là độc lập đối với bản tin
  - Có thể được tính trước

$$C_i = P_i \text{ XOR } O_i;\ O_i = DES_{K1}(O_{i-1});\ O_{-1} = IV$$

  - Được dùng cho mã dòng trên các kênh âm thanh



2.7.2. Các chế độ hoạt động của DES

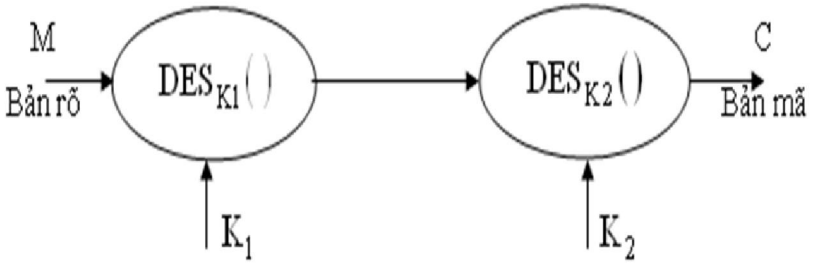
- **Ưu điểm và nhược điểm của OFB**
  - Được dùng khi lỗi phản hồi ngược lại hoặc ở nơi cần mã trước khi mẩu tin sẵn sàng
  - Rất giống CFB
  - Nhưng phản hồi là từ đầu ra của mã và độc lập với mẩu tin
  - Là biến thể của mã Vernam, suy ra không sử dụng lại với cùng một dãy (Key + IV)
  - Người gửi và người nhận phải đồng bộ, có phương pháp khôi phục nào đó là cần thiết để đảm bảo việc đó.
  - Nguyên bản chỉ rõ m bit phản hồi ngược theo các chuẩn
  - Các nghiên cứu tiếp theo chỉ ra rằng chỉ có OFB64 là dùng được

2.7.3. Double DES và Triple DES

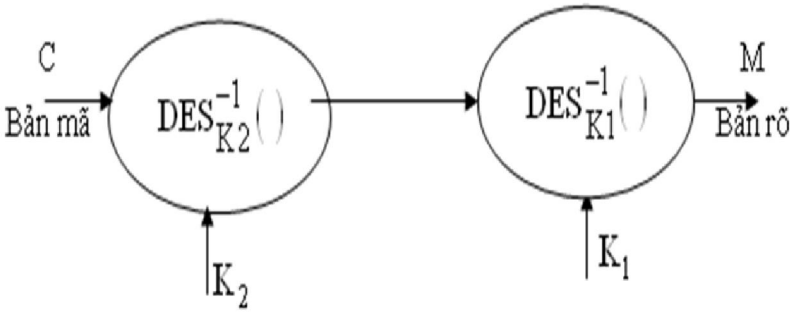
- DES bội hai
  - **Mã hóa:**

    $C = DES_{K2}[DES_{K1}(M)]$
  - **Giải mã:**

    $M = DES_{K1}^{-1}[DES_{K2}^{-1}(C)]$
  - Có 2^{56} sự lựa chọn cho khóa K_1 và 2^{56} sự lựa chọn cho khóa K_2. Bởi vậy có 2^{112} sự lựa chọn cho cặp khóa (K_1, K_2)



a. Mã hóa DES bội hai



b. Giải mã DES bội hai

Mã hóa và giải mã DES bội hai

2.7.3. Double DES và Triple DES

- DES bội ba
  - **Mã hóa:**

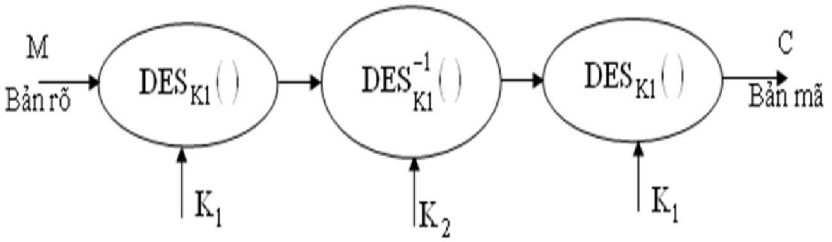
$$C = DES_{K1}\left\{DES^{-1}_{K2}\left[DES_{K1}(M)\right]\right\}$$

  - **Giải mã:**

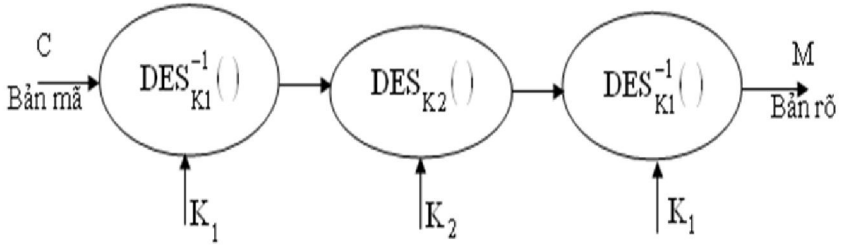
$$M = DES^{-1}_{K1}\left\{DES_{K2}\left[DES^{-1}_{K1}(C)\right]\right\}$$

  - Với TDES việc tìm khóa vét cạn yêu cầu khoảng:

$2^{112} = 5{,}1923.10^{23}$ phép tính TDES, bởi vậy thực tế khó có thể thám mã thành công.



a. Mã hóa TDES với hai khóa



b. Giải mã TDES với hai khóa

Mã hóa và giải mã TDES với hai khóa

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- **Nguồn gốc:**
  - Rõ ràng cần phải thay thế DES, vì có những tấn công về mặt lý thuyết có thể bẻ được nó.
  - Do đó Viện chuẩn quốc gia Hoa kỳ US NIST ra lời kêu gọi tìm kiếm chuẩn mã mới vào năm 1997. Sau đó có 15 đề cử được chấp nhận vào tháng 6 năm 1998. Và được rút gọn còn 5 ứng cử viên vào tháng 6 năm 1999. Đến tháng 10 năm 2000, mã Rijndael được chọn làm chuẩn mã nâng cao và được xuất bản là chuẩn FIPS PUB 197 vào 11/2001.
- **Yêu cầu của AES**
  - Là mã khối đối xứng khoá riêng.
  - Kích thước khối dữ liệu 128 bit và độ dài khoá là tùy biến: 128, 192 hoặc 256 bit.
  - Chuẩn mã mới phải mạnh và nhanh hơn Triple DES. Mã mới có cơ sở lí thuyết mạnh để thời gian sống của chuẩn khoảng 20-30 năm (cộng thêm thời gian lưu trữ).
  - Khi đưa ra thành chuẩn yêu cầu cung cấp chi tiết thiết kế và đặc tả đầy đủ. Đảm bảo rằng chuẩn mã mới cài đặt hiệu quả trên cả C và Java.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- 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:**
    - $A = (a_1\ a_2\ a_3\ a_4\ a_5\ a_6\ a_7\ a_8)$; $B = (b_1\ b_2\ b_3\ b_4\ b_5\ b_6\ b_7\ b_8)$
    - $C = A + B = (c_1\ c_2\ c_3\ c_4\ c_5\ c_6\ c_7\ c_8)$, trong đó: $C_i = a_i + b_i \bmod 2$, $1 \le i \le 8$.
    - **Ví dụ:** tổng của $A = 73_H$; $B = 4E_H$ là:
      - Dạng cơ số Hexa: $73_H + 4E_H = 3D_H$
      - Dạng nhị phân: 01110011 + 01001110 = 00111101
      - Dạng đa thức:

      $(x^6 + x^5 + x^4 + x + 1) + (x^6 + x^3 + x^2 + x) = (x^5 + x^4 + x^3 + x^2 + 1)$

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- **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 $a(x) = x^7 + x^6 + x + 1$ và $b(x) = x^7 + x^2 + 1$. Khi đó: C= A.B

    $c(x) = a(x).b(x) \bmod (x^8 + x^4 + x^3 + x + 1)$

    $c(x) = x^7 + x^5 + x^3 + x^2 + x$ hay $C = AE_H = 10101110$

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

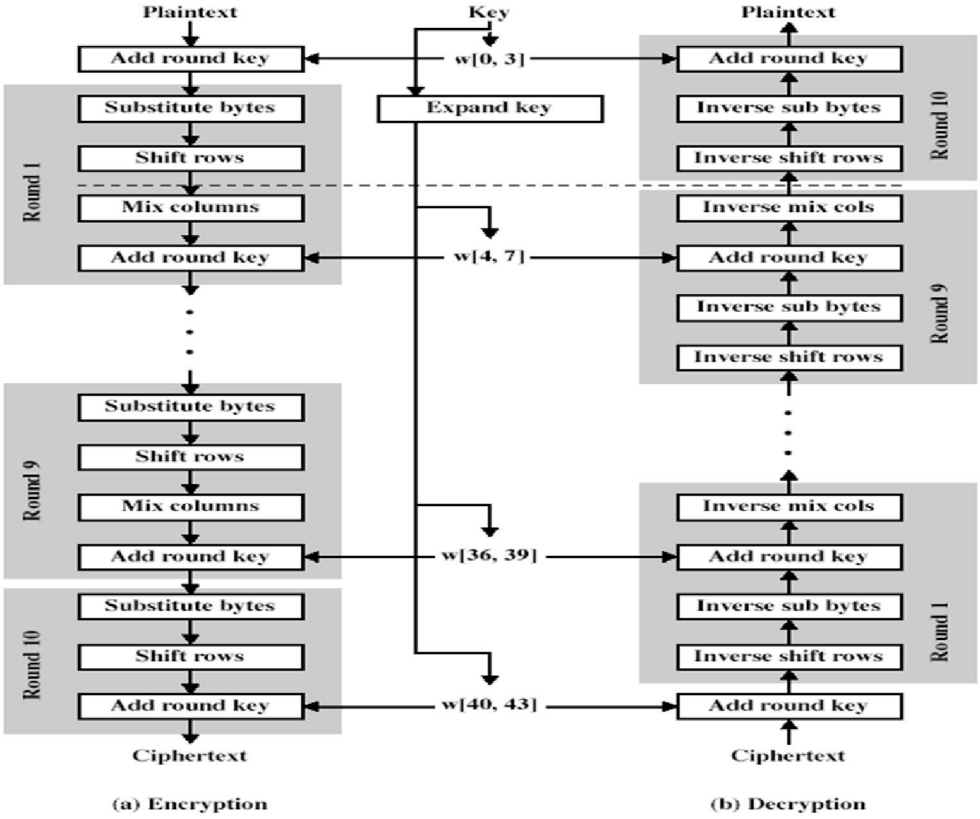
- **Chuẩn mã nâng cao AES – Rijndael**: có các đặc trưng sau:
  - Có 128/192/256 bit khoá và 128 bit khối dữ liệu.
  - Lặp hơi khác với Fiestel
    - Chia dữ liệu thành 4 nhóm – 4 byte
    - Thao tác trên cả khối mỗi vòng
    - Thiết kế để:
      - chống lại các tấn công đã biết
      - tốc độ nhanh và nén mã trên nhiều CPU
      - Đơn giản trong thiết kế

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

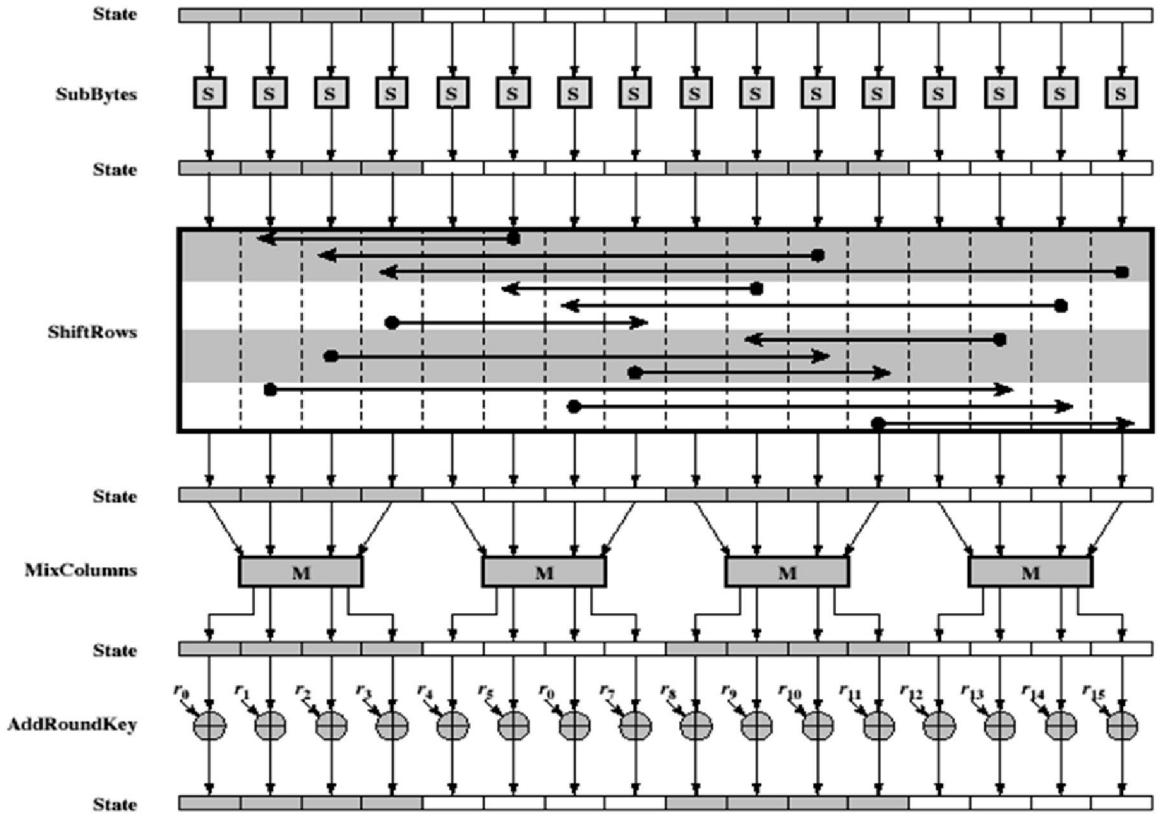
- Xử lý khối dữ liệu 128 bit như 4 nhóm của 4 byte: 128 = 4*4*8 bit. Mỗi nhóm nằm trên một hàng. Ma trận 4 hàng, 4 cột với mỗi phần tử là 1 byte coi như trạng thái được xử lý qua các vòng mã hoá và giải mã.
- Khoá mở rộng thành mảng gồm 44 từ 32 bit w[i].
- Có tùy chọn 9/11/13 vòng, trong đó mỗi vòng bao gồm
  - Phép thế byte (dùng một S box cho 1 byte)
  - Dịch hàng (hoán vị byte giữa nhóm/cột)
  - Trộn cột (sử dụng nhân ma trận của các cột)
  - Cộng khoá vòng (XOR trạng thái dữ liệu với khoá vòng).
  - Mọi phép toán được thực hiện với XOR và bảng tra, nên rất nhanh và hiệu quả.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Sơ đồ Rijndael



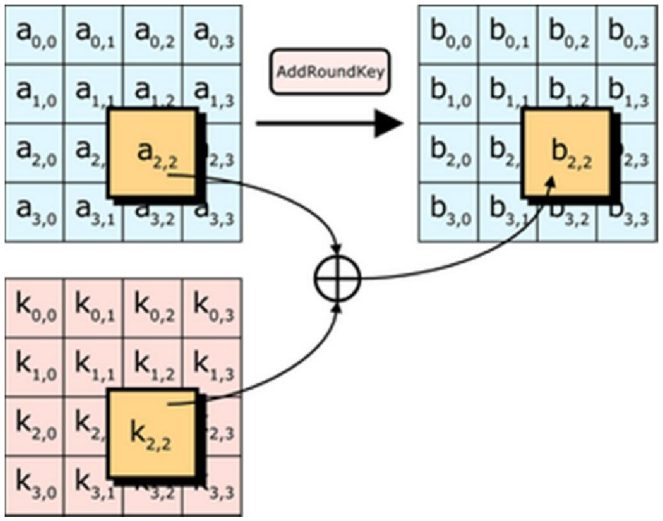
2.8. Chuẩn mã dữ liệu tiên tiến (AES)



Một vòng AES

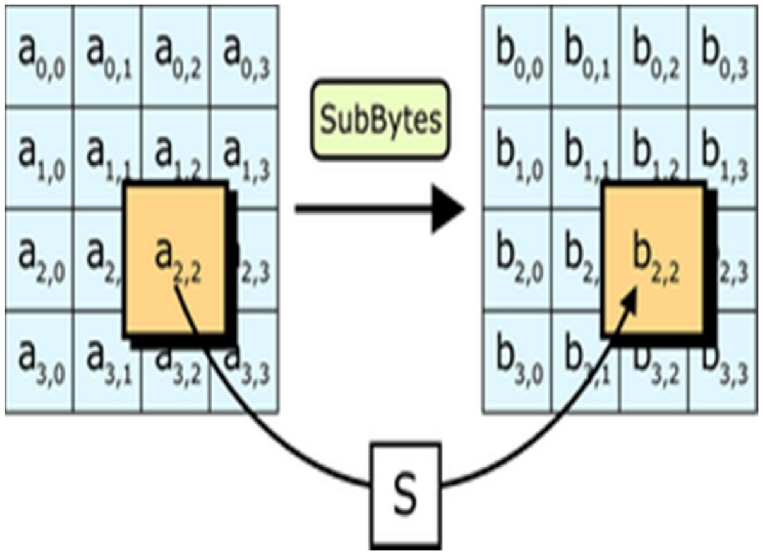
2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- **Sau đây ta xét chi tiết hơn các quá trình mã hoá, sinh khoá và giải mã AES:**
  - Quá trình mã gồm 4 bước sau:
    - 1. AddRoundKey - mỗi byte của khối được kết hợp với khóa con, các khóa con này được tạo ra từ quá trình tạo khóa con Rijndael



2.8. Chuẩn mã dữ liệu tiên tiến (AES)

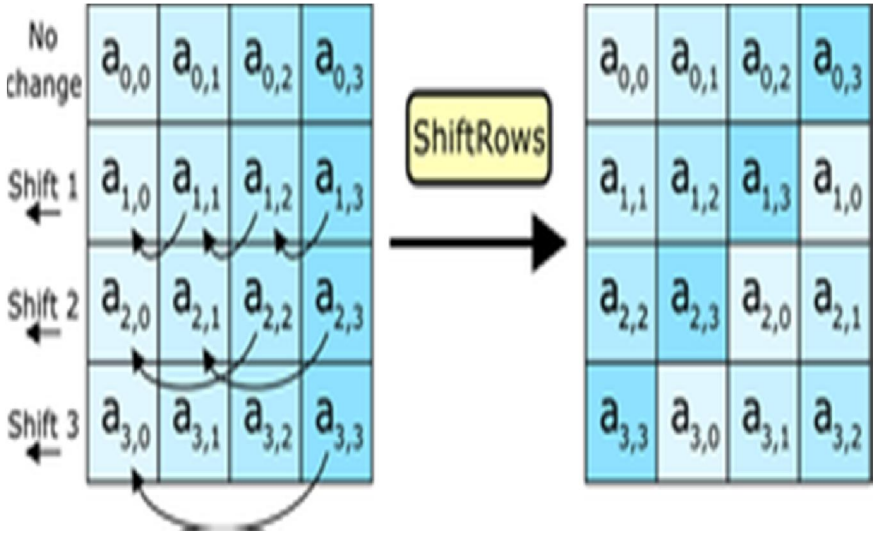
- 2. SubBytes - đây là quá trình thay thế (phi tuyến) trong đó mỗi byte sẽ được thay thế bằng một byte khác theo bảng tra
  - Phép thê byte đơn giản
  - Sử dụng một bảng 16 x 16 byte chứa hoán vị của tất cả 256 giá trị 8 bit



  - Mỗi byte trạng thái được thay bởi byte trên hàng xác định bởi 4 bit trái và cột xác định bởi 4 bit phải.
  - Chẳng hạn {95} được thay bởi hàng 9, cột 5, mà giá trị sẽ là {2A}.
  - S box được xây dựng sử dụng hoán vị các giá trị trong GF(28) đã được xác định trong chương trước.
  - Thiết kế để chống mọi tấn công đã biết

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

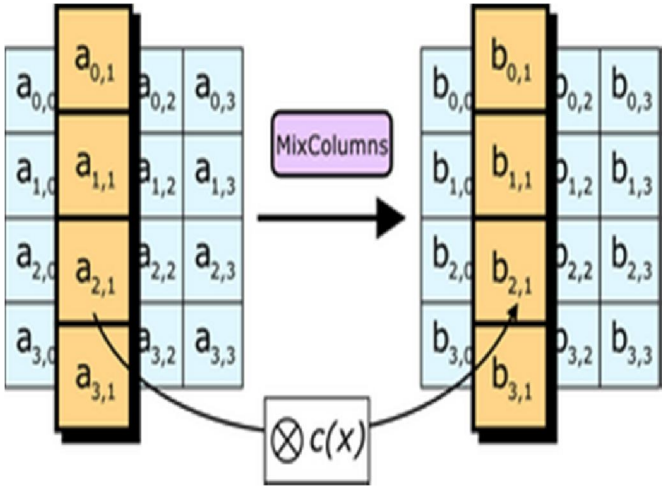
- 3. ShiftRows - đổi chỗ, các hàng trong khối được dịch vòng
  - Dịch hàng vòng quanh trên mỗi hàng
  - Hàng 1 không đổi
  - Hàng 2 dịch vòng quanh 1 byte sang trái



  - Hàng 3 dịch vòng quanh 2 byte sang trái
  - Hàng 4 dịch vòng quanh 3 byte sang trái
  - Giải mã thực hiện dịch ngược lại sang phải
  - Vì trạng thái được xử lý bởi cột, bước này thực chất là hoán vị byte giữa các cột.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- 4. MixColumns - quá trình trộn làm việc theo các cột trong khối theo một chuyển đổi tuyến tính.
  - Có thể biểu diễn mỗi cột mới là nghiệm của 4 phương trình
    - để tìm ra byte mới trong mỗi cột
  - Mã yêu cầu sử dụng ma trận nghịch đảo
    - Với hệ số lớn thì tính toán khó khăn hơn



- Có các đặc trưng khác của cột như sau:
  - Mỗi cột là một đa thức bậc 3 gồm 4 số hạng
  - Với mỗi phần tử là một byte tương ứng với phần tử trong GF(28).
  - Các đa thức nhân tính theo Modulo (x4+1).

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Bốn byte trong từng cột được kết hợp lại theo một phép biến đổi tuyến tính khả nghịch. Mỗi khối 4 byte đầu vào sẽ cho một khối 4 byte ở đầu ra với tính chất là mỗi byte ở đầu vào đều ảnh hưởng tới cả 4 byte đầu ra.
- Cùng với bước ShiftRows, MixColumns đã tạo ra tính chất khuếch tán cho thuật toán. Mỗi cột được xem như một đa thức trong trường hữu hạn và được nhân với đa thức

$$c(x) = 3x^3 + x^2 + x + 2 \text{ (modulo } x^4 + 1)$$

Vì thế, bước này có thể được xem là phép nhân ma trận trong trường hữu hạn.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- **Mở rộng khoá AES**
  - Dùng khoá 128 bit (16 byte) và mở rộng thành mảng gồm 44/52/60 từ 32 bit.
  - Bắt đầu bằng việc copy khoá vào 4 từ đầu
  - Sau đó tạo quay vòng các từ mà phụ thuộc vào giá trị ở các vị trí trước và 4 vị trí sau
    - 3 trong 4 trường hợp chỉ là XOR chúng cùng nhau
    - Mỗi cái thứ 4 có S box kết hợp quay và XOR với hằng số trước đó, trước khi XOR cùng nhau
    - Thiết kế chống các tấn công đã biết

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Giải mã AES
  - Giải mã ngược lại không duy nhất vì các bước thực hiện theo thứ tự ngược lại.
  - Nhưng có thể xác định mã ngược tương đương với các bước đã làm đối với mã
    - Nhưng sử dụng ngược lại với từng bước
    - Với khoá con khác nhau
  - Thực hiện được vì kết quả không thay đổi khi
    - Đổi lại phép thế byte và dịch các hàng
    - Đổi lại việc trộn các cột và bổ sung khoá vòng
  - Lý do mở rộng khoá: các tiêu chuẩn thiết kế bao gồm
    - Giả sử biết một phần khoá, khi đó không đủ để biết nhiều hơn, tức là các khoá con khác hoặc khoá nói chung.
    - Phép biến đổi nghịch đảo được.
    - Nhanh đối với nhiều kiểu CPU.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Sử dụng hằng số vòng để làm mất tính đối xứng
- Khuếch tán bit khoá thành khoá con cho các vòng
- Có đủ tính phi đối xứng để chống thám mã
- Đơn giản trong việc giải mã

- Các khía cạnh cài đặt:
  - Có thể cài đặt hiệu quả trên CPU 8 bit
  - Phép thế byte làm việc trên các byte sử dụng bảng với 256 đầu vào.
  - Dịch hàng là phép dịch byte đơn giản
  - Cộng khoá vòng làm việc trên byte XOR
  - Các cột hỗn hợp yêu cầu nhân ma trận trong GF(28) mà làm việc trên giá trị các byte, có thể đơn giản bằng cách tra bảng

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- có thể cài đặt hiệu quả trên CPU 32 bit
- Xác định lại các bước để sử dụng từ 32 bit
- Có thể tính trước 4 bảng với 256 đầu vào
- Sau đó mỗi cột trong mỗi vòng có thể tính bằng cách tra 4 bảng và 4 XOR
- Cần 16 Kb để lưu các bảng
- Những nhà thiết kế tin tưởng rằng việc cài đặt rất hiệu quả này là yếu tố cơ bản trong việc chọn nó là mã AES

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Độ an toàn của AES:
  - Thiết kế và độ dài khóa của thuật toán AES (128, 192 và 256 bít) là đủ an toàn để bảo vệ các thông tin được xếp vào loại TỐI MẬT (secret). Các thông tin TUYỆT MẬT (top secret) sẽ phải dùng khóa 192 hoặc 256 bít.
  - Một vấn đề khác nữa là cấu trúc toán học của AES. Không giống với các thuật toán mã hóa khác, AES có mô tả toán học khá đơn giản. Tuy điều này chưa dẫn đến mối nguy hiểm nào nhưng một số nhà nghiên cứu sợ rằng sẽ có người lợi dụng được cấu trúc này trong tương lai.

2.8. Chuẩn mã dữ liệu tiên tiến (AES)

- Vào thời điểm năm 2006, dạng tấn công lên AES duy nhất thành công là tấn công kênh bên (side channel attack).
- Tấn công kênh bên không tấn công trực tiếp vào thuật toán mã hóa mà thay vào đó, tấn công lên các hệ thống thực hiện thuật toán có sơ hở làm lộ dữ liệu