TRƯỜNG CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG

KHOA KHOA HỌC MÁY TÍNH

AN TOÀN VÀ BẢO MẬT THÔNG TIN

(Information security)

Chương 3: Các phương pháp tạo mật mã

A. Tổng quan về mật mã

Mật mã học

- Mật mã học (Cryptology)
  - Mật mã (Cryptography)
  - Mã thám (Cryptanalysis)
- Mật mã
  - Tăng cường các tính chất *Bí mật* và *Toàn vẹn* thông tin: các phép mã hóa
  - Xây dựng các kỹ thuật trao đổi thông tin bí mật: các giao thức mật mã
- Mã thám
  - Phá mã

Các giai đoạn hình thành mật mã

- **Giai đoạn "Tiền sử"** (~ 2000, TCN)
  - Những dấu hiệu đầu tiên của Mật mã xuất hiện ở bên bờ sông Nile, Ai Cập
- **Giai đoạn "Mật mã thủ công"** (~ 50, TCN)
  - Phép mã hóa Ceasar
- **Giai đoạn "Mật mã cơ học"** (cho đến Thế chiến 2)
  - Máy Enigma ở Đức
  - Các nghiên cứu về Mã thám ở Anh
- **Giai đoạn "Mật mã điện tử"**
  - Dựa vào Toán học và Tin học
  - Được đặt nền móng bởi Shanon, Diffie và Hellman
  - Khóa bí mật (DES, AES,…), Khóa công khai (RSA,….)

Các khái niệm cơ bản

- **Hệ mã** (**C**ryptosystem): một phương pháp ngụy trang bản rõ. Nghệ thuật tạo ra và sử dụng các hệ mật mã được gọi là thuật mật mã hóa hay mật mã học (Cryptography)
- **Bản rõ** (**P**laintext): thông tin gốc cần chuyển, được ghi bằng hình ảnh, âm thanh, chữ số, chữ viết…
- **Bản mật/ Bản mã** (**C**iphertext): "ngụy trang" bản rõ thành một dạng khác để người "ngoài cuộc" không thể đọc được
- **Mật mã hóa/ lập mã** (**E**ncryption): quá trình biến đổi bản rõ thành bản mật
- **Giải mật mã/ giải mã** (**D**ecryption): quá trình biến đổi bản mật thành bản rõ

Các khái niệm cơ bản

- **Phân tích mã/thám mã** (**Cryptanalysis**)
- Nỗ lực phá vỡ các hệ mật mã
- Lý do thám mã là cần thiết
  - Không tồn tại chứng minh toán học về tính an toàn cho bất kỳ hệ mật mã thực tế nào
  - Cách duy nhất đảm bảo tính an toàn của một hệ mật mã là thử phá vỡ nó (và thất bại)

Các khái niệm cơ bản

❖ **Các PP Thám mã**

- Thám mã cổ điển
  - Khoa học phát hiện nguyên bản hoặc khóa
  - Các tấn công thám mã
    - + Khai thác cấu trúc bên trong của phương pháp mã hóa
  - Các tấn công vét cạn
    - + Coi giải thuật mã hóa như một hộp đen và thử tất cả các khóa có thể
- Các tấn công cài đặt

Các khái niệm cơ bản

- **Giải thuật mã hóa** (***Encryption algorithm***) là giải thuật dùng để mã hóa thông tin
- **Giải thuật giải mã** (***Decryption algorithm***) dùng để giải mã thông tin.
- **Khóa/Chìa** (***Key***) là một chuỗi được sử dụng trong giải thuật mã hóa và giải mã.
- **Không gian khóa** (***Keyspace***) là tổng số khóa có thể có của một hệ mã hóa.

  VD: nếu sử dụng khóa kích thước 64 bit thì không gian khóa là 2^{64}.

3 đặc tính của hệ mật mã hóa

(1) Kiểu của các thao tác được dùng để biến đổi bản rõ thành bản mật

(2) Số khóa được sử dụng

(3) Cách thức bản rõ được xử lý

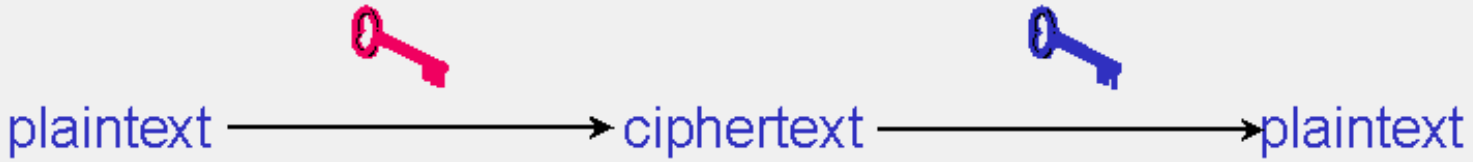
(1) Kiểu của các thao tác được dùng để biến đổi bản rõ thành bản mật

- Các giải thuật mã hóa dựa trên các nguyên lý:
  - *Thay thế* (substitution): mỗi thành phần trong bản rõ (bit, ký tự, nhóm các bit hay các ký tự) được ánh xạ đến thành phần khác.
  - *Chuyển vị* (transposition): các thành phần trong bản rõ được sắp xếp lại
- Yêu cầu cơ bản: thông tin không bị mất (*Phần lớn các hệ mã kết hợp cả 2 nguyên lý qua nhiều bước*)

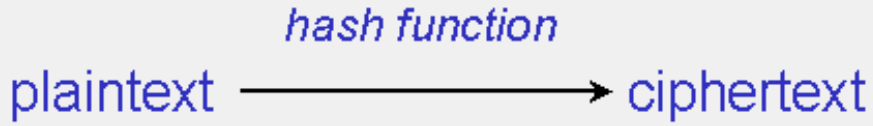
(2) Số khóa được sử dụng

- **1 Khóa**: người gởi và người nhận sử dụng chung khóa (Khóa đối xứng)
- **2 Khóa**: Khóa bí mật/Khóa công khai. Mật hóa dùng 1 khóa và giải mật mã dùng 1 khóa khác (Khóa bất đối xứng)

A) Secret key (symmetric) cryptography. SKC uses a single key for both encryption and decryption.



B) Public key (asymmetric) cryptography. PKC uses two keys, one for encryption and the other for decryption.

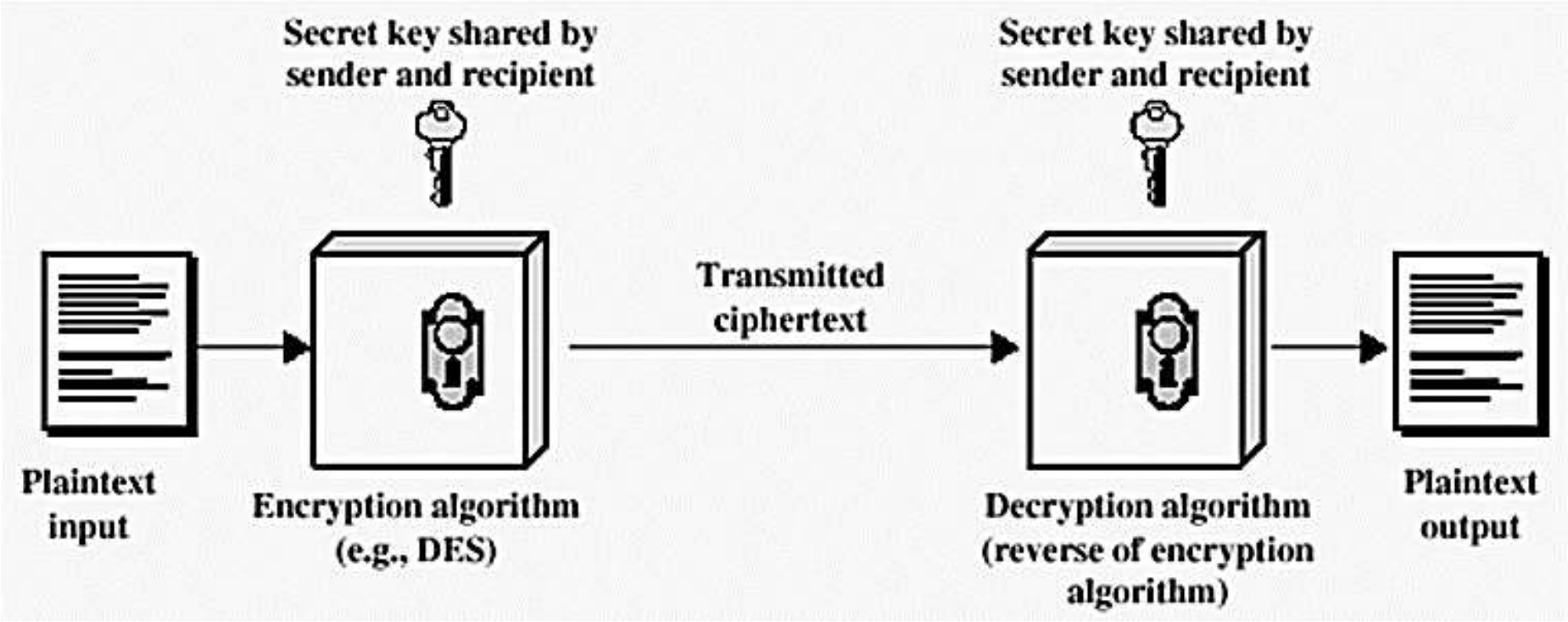


C) Hash function (one-way cryptography). Hash functions have no key since the plaintext is not recoverable from the ciphertext.

Sơ đồ ba phương pháp mã hóa cơ bản

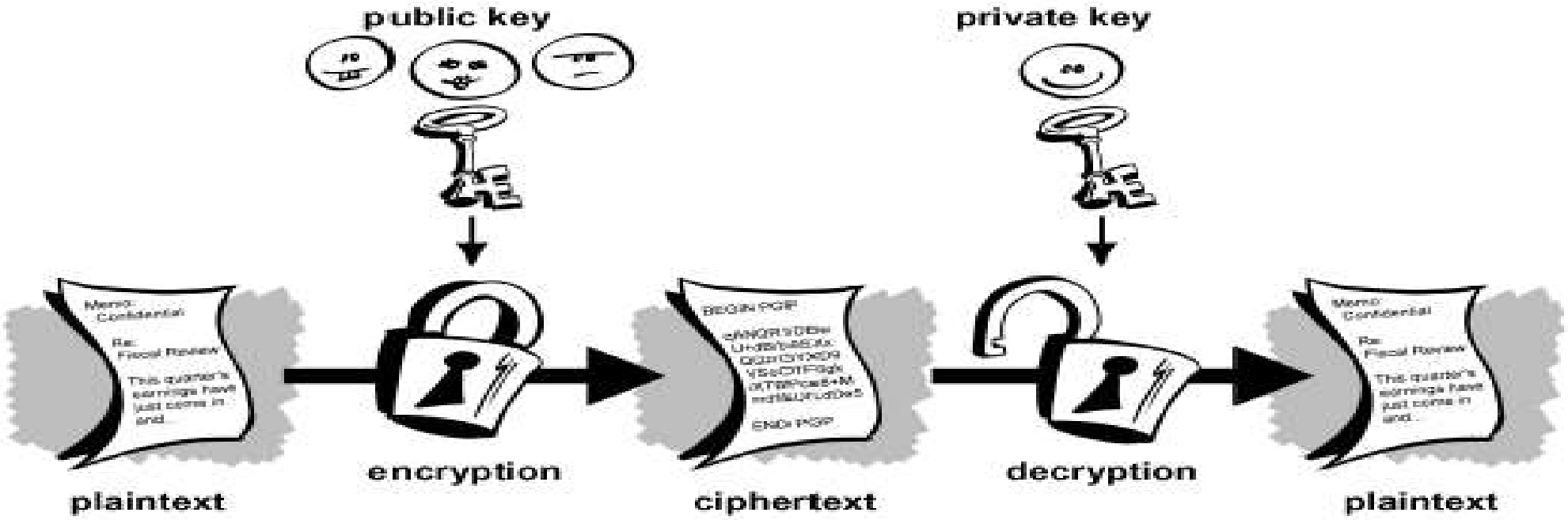
Mã hóa khóa với khóa đối xứng

- Hay mã hóa khóa bí mật (**Secret key cryptography**). Là dạng mã hóa trong đó một khóa được sử dụng cho cả giải thuật mã hóa và giải mã.



Mã hóa khóa với khóa bất đối xứng

- Hay còn được gọi là **mã hóa khóa công khai** (Public key cryptography)
- Trong đó *một cặp khóa* được sử dụng:
  - Khóa công khai (public key) dùng để mã hóa
  - Khóa riêng (private key) dùng để giải mã

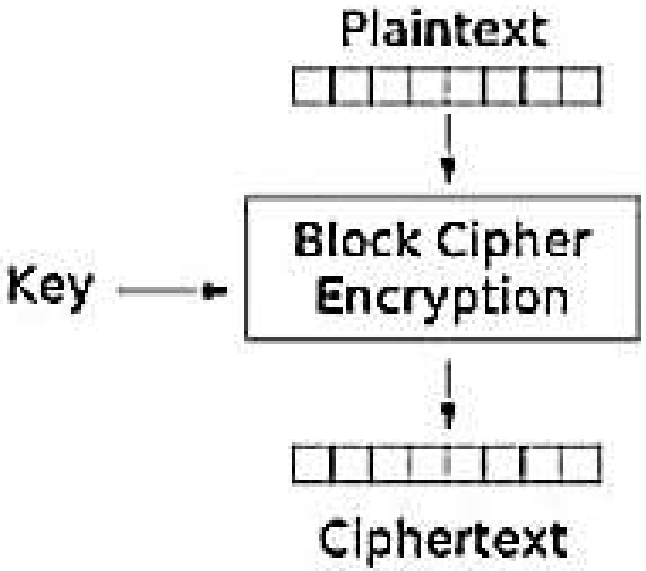


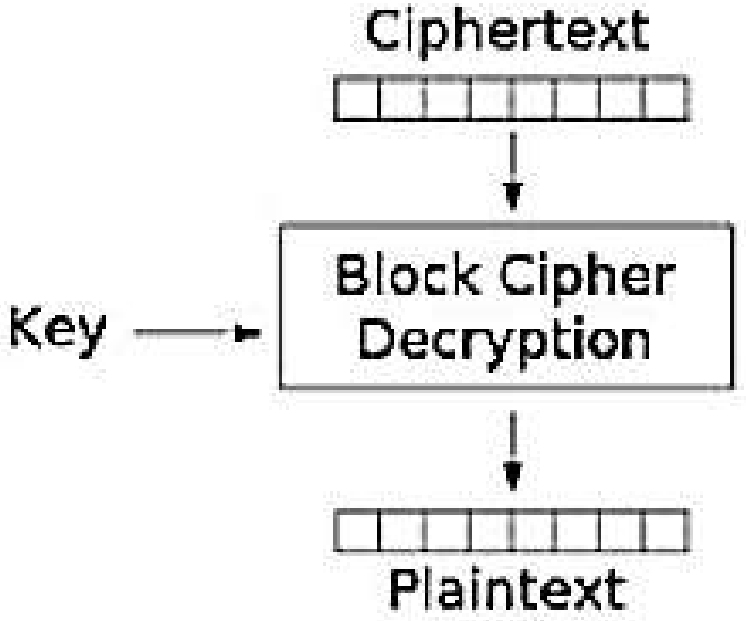
Hàm băm (*Hash function*)

- Là một ánh xạ chuyển các dữ liệu có kích thước thay đổi về dữ liệu có kích thước cố định.
- **Hàm băm 1 chiều** (One-way hash function) là hàm băm, trong đó việc thực hiện mã hóa tương đối đơn giản, còn việc giải mã thường có độ phức tạp rất lớn, hoặc không khả thi về mặt tính toán.

(3) Cách thức bản rõ được xử lý:

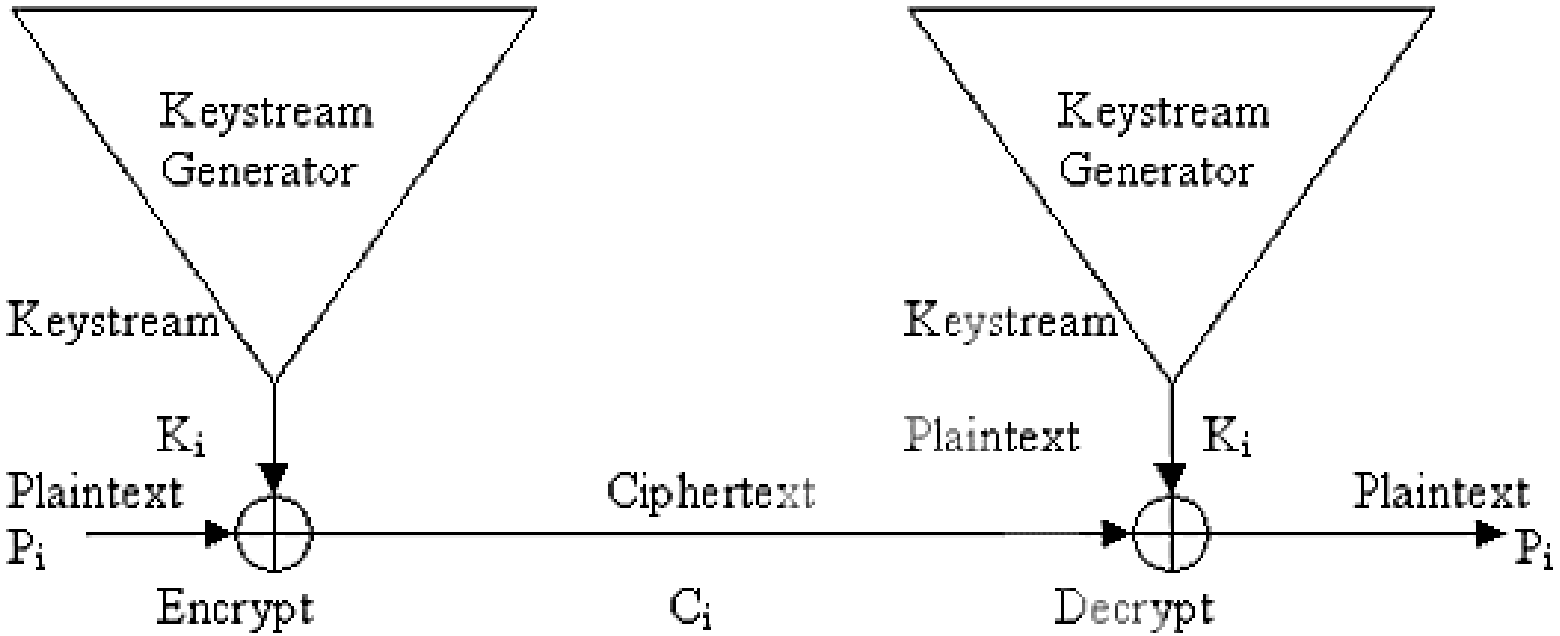
- **Mã hóa khối** (block cipher) là kiểu mã hóa mà *dữ liệu được chia ra thành từng khối có kích thước cố định* để mã hóa và giải mã.
- VD: DES, 3-DES, IDEA, AES với kích thước khối 64, 128 bit





(3) Cách thức bản rõ được xử lý

Mã hóa luồng (stream cipher) Là kiểu mã hóa mà **từng bit**, hoặc **ký tự** của **bản rõ** được kết hợp (XOR) với **từng bit**, hoặc **ký tự** tương ứng **của khóa** để tạo thành bản mã.



B. Các kỹ thuật mã hóa

I. Mã hóa đối xứng căn bản

1. Mã hóa Ceasar
2. Mã hóa thay thế (Đơn, Đa kí tự, Đa bảng)
3. One-Tim Pad
4. Mã hóa hoán vị (Permutation)

1. Mã hóa Ceasar

Mật mã đơn ký tự

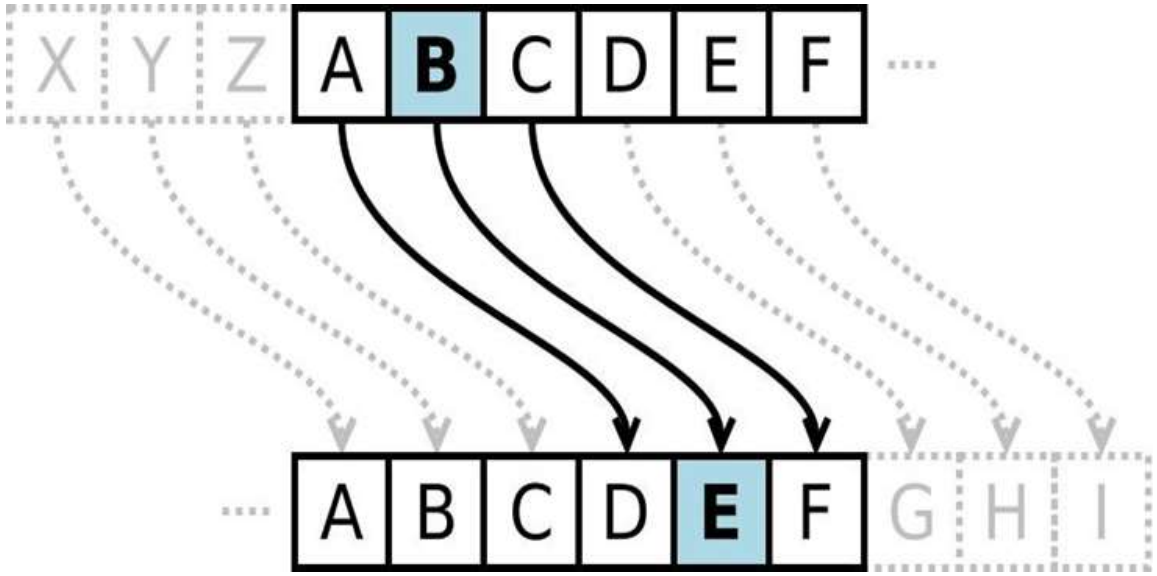
Thay thế mỗi ký tự bằng một ký tự khác cách nhau một khoảng cách nhất định trong bảng chữ cái

Biểu diễn toán học:

$$C = E(p) = (p+k) \bmod 26$$

$$p = D(c) = (c - k) \bmod 26$$

Với **k**: khóa của giải thuật



1. Ceasar – Ví dụ

Áp dụng mật mã Ceasar mật mã hóa các bản rõ sau với khóa k = 4

actions speak louder than words

Đoán khóa **k** và giải mật cho bản mật sau:

ST RFS HFS XJWAJ YBT RFXYJWX

1. Mã hóa Ceasar

Có thể bị tấn công theo kiểu **vét cạn** (**brute-force**) bằng cách thử hết tất cả 25 khóa.

	TIPGKFXIRGYPZEYZJKFIPZJZEKVIVJKZEX		TIPGKFXIRGYPZEYZJKFIPZJZEKVIVJKZEX
1	UJQHLGYJSHZQAFZAKLGJQAKAFLWJWKLAFY	14	HWDUYTLWFUMDNSMNXYTWDNXNSYJWJXYNSL
2	VKRIMHZKTIARBGABLMHKRBLBGMXKXLMBGZ	15	IXEVZUMXGVNEOTNOYZUXEOYOTZKXKYZOTM
3	WLSJNIALUJBSCHBCMNILSCMCHNYLYMNCHA	16	JYFWAVNYHWOFPUOPZAVYFPZPUALYLZAPUN
4	XMTKOJBMVKCTDICDNOJMTDNDIOZMZNODIB	17	KZGXBWOZIXPGQVPQABWZGQAQVBMZMABQVO
5	YNULPKCNWLDUEJDEOPKNUEOEJPANAOPEJC	18	LAHYCXPAJYQHRWQRBCXAHRBRWCNANBCRWP
6	ZOVMQLDOXMEVFKEFPQLOVFPFKQBOBPQFKD	19	MBIZDYQBKZRISXRSCDYBISCSXDOBOCDSXQ
7	APWNRMEPYNFWGLFGQRMPWGQGLRCPCQRGLE	20	NCJAEZRCLASJTYSTDEZCJTDTYEPCPDETYR
8	BQXOSNFQZOGXHMGHRSNQXHRHMSDQDRSHMF	21	ODKBFASDMBTKUZTUEFADKUEUZFQDQEFUZS
9	CRYPTOGRAPHYINHISTORYISINTERESTING	22	PELCGBTENCULVAUVFGBELVFVAGRERFGVAT
10	DSZQUPHSBQIZJOIJTUPSZJTJOUFSFTUJOH	23	QFMDHCUFODVMWBVWGHCFMWGWBHSFSGHWBU
11	ETARVQITCRJAKPJKUVQTAKUKPVGTGUVKPI	24	RGNEIDVGPEWNXCWXHIDGNXHXCITGTHIXCV
12	FUBSWRJUDSKBLQKLVWRUBLVLQWHUHVWLQJ	25	SHOFJEWHQFXOYDXYIJEHOYIYDJUHUIJYDW
13	GVCTXSKVETLCMRLMWXSVCMWMRXIVIWXMRK		

3. Mã hóa thay thế đơn bản

- Mỗi ***hoán vị*** của 26 chữ cái là một khóa => **Có 26!=4.1026 khóa.**
- Ví dụ:

Chữ ban đầu: a b c d e f g h i j k l m n o p q r s t u v w x y z

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

- Như vậy bản rõ : **meet me after the toga party**
- Được mã hóa thành: **NJJU NJ ZRUJM UKJ UVSZ DZMUE**
- Quá trình giải mã được tiến hành ngược lại để cho ra bản rõ ban đầu.

Nhận xét: tần số xuất hiện của các chữ cái trong các ngôn ngữ là cố định

3. Mã hóa thay thế đơn bản

Bảng liệt kê tần suất chữ cái tiếng Anh

Chữ cái (%)		Cụm 2 chữ (%)		Cụm 3 chữ (%)		Từ (%)	
E	13.05	TH	3.16	THE	4.72	THE	6.42
T	9.02	IN	1.54	ING	1.42	OF	4.02
O	8.21	ER	1.33	AND	1.13	AND	3.15
A	7.81	RE	1.30	ION	1.00	TO	2.36
N	7.28	AN	1.08	ENT	0.98	A	2.09
I	6.77	HE	1.08	FOR	0.76	IN	1.77
R	6.64	AR	1.02	TIO	0.75	THAT	1.25
S	6.46	EN	1.02	ERE	0.69	IS	1.03
H	5.85	TI	1.02	HER	0.68	I	0.94
D	4.11	TE	0.98	ATE	0.66	IT	0.93
L	3.60	AT	0.88	VER	0.63	FOR	0.77
C	2.93	ON	0.84	TER	0.62	AS	0.76
F	2.88	HA	0.84	THA	0.62	WITH	0.76
U	2.77	OU	0.72	ATI	0.59	WAS	0.72
M	2.62	IT	0.71	HAT	0.55	HIS	0.71
P	2.15	ES	0.69	ERS	0.54	HE	0.71
Y	1.51	ST	0.68	HIS	0.52	BE	0.63
W	1.49	OR	0.68	RES	0.50	NOT	0.61
G	1.39	NT	0.67	ILL	0.47	BY	0.57
B	1.28	HI	0.66	ARE	0.46	BUT	0.56
V	1.00	EA	0.64	CON	0.45	HAVE	0.55
K	0.42	VE	0.64	NCE	0.45	YOU	0.55
X	0.30	CO	0.59	ALL	0.44	WHICH	0.53
J	0.23	DE	0.55	EVE	0.44	ARE	0.50
Q	0.14	RA	0.55	ITH	0.44	ON	0.47
Z	0.09	RO	0.55	TED	0.44	OR	0.45

4. Mã hóa thay thế đa ký tự

1. Mật mã Playfair
2. Mật mã Hill
3. Mật mã Affine

- Bài tập nhóm:

1. Giải thuật
2. Sinh viên nộp cài đặt giải thuật bằng ngôn ngữ Python **(Nộp lên hệ thống Elaerning)**

Mật mã Playfair

- **2 ký tự đứng sát nhau** là một đơn vị mã hóa (chúng được thay thế cùng lúc bằng hai ký tự khác)
- Giải thuật dựa trên một ma trận vuông **5x5** được xây dựng từ một khóa (chuỗi các ký tự)
- **Xây dựng ma trận khóa**:
  - Lần lượt thêm từng ký tự của khóa vào ma trận
  - Nếu ma trận chưa đầy, thêm các ký tự còn lại trong bảng chữ cái vào ma trận theo thứ tự A – Z
  - **I** và **J** được điền vào cùng 1 ô
  - Các ký tự trong ma trận không được trùng nhau

Mật mã Playfair

Ví dụ: Xây dựng MT khóa

- **Key:PLAYFAIR EXAMPLE**
- **Key: MONARCHY**

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

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

Mật mã Playfair

❖ **Giải thuật mật mã hóa**:

- Mã hóa từng cặp "**2 ký tự**" liên tiếp nhau.
- Nếu 2 ký tự này giống nhau thì thêm một ký tự '**X**' vào giữa.

  VD: **balloon** tách thành **balxlo** on (vì ll →lx l)
- Nếu dư 1 ký tự thì thêm vào ký tự 'q' vào cuối.

  VD: hat → ha tq
- Nếu 2 ký tự nằm **cùng dòng** được thay thế bằng 2 ký tự tương ứng **bên phải**. Ký tự ở cột cuối cùng được thay bằng ký tự ở cột đầu tiên

  VD: AR → RM

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

Mật mã Playfair

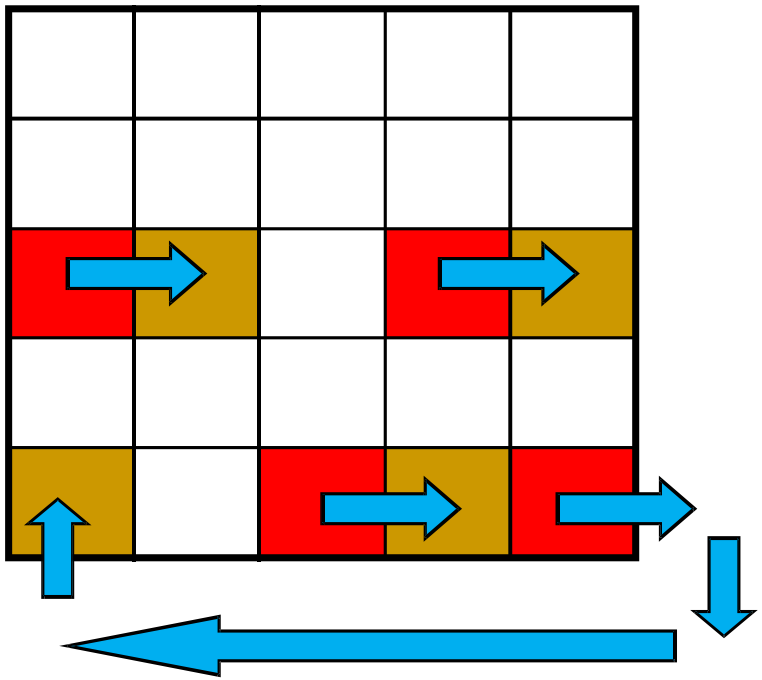
❖ **Giải thuật mật mã hóa**:

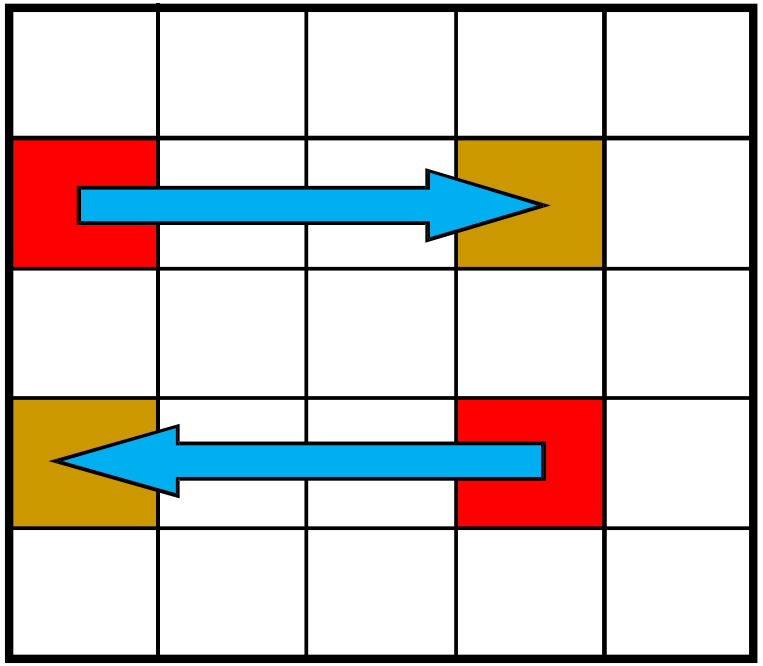
- Nếu hai ký tự trong cặp thuộc **cùng một cột**, thì được thay bằng hai ký tự tiếp theo trong cột. Nếu đến **cuối cột** thì quay về đầu cột. Ví dụ cặp **OV** được mã hóa thành **HO**.
- Nếu hai ký tự được mã hóa sẽ tạo thành **đường chéo** *của một hình chữ nhật* và được thay bằng 2 ký tự trên đường chéo kia. Ví dụ: HS → BP (B cùng dòng với H và P cùng dòng với S);

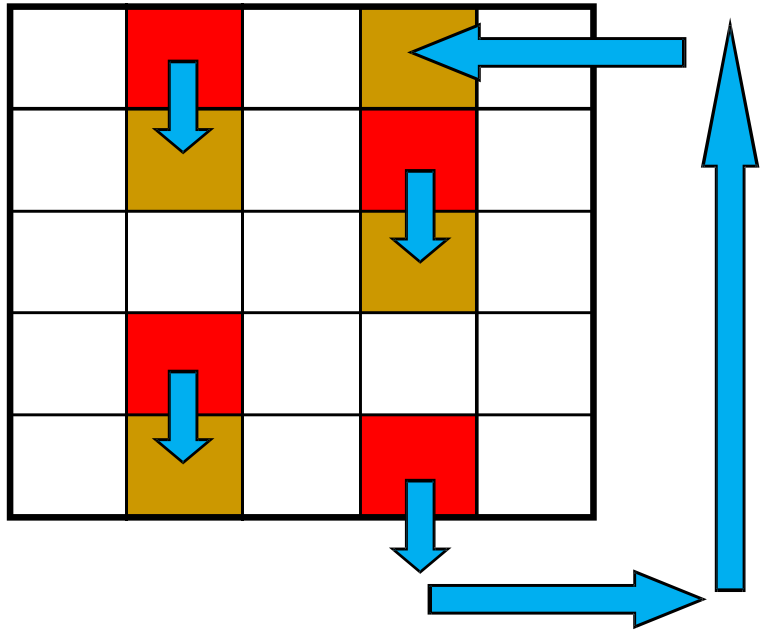
  EA → IM (hoặc JM)

  Nhận xét: Playfair có **26x26=676** cặp – ít bị chênh lệnh về tần suất

Mật mã Playfair







Mật mã Playfair

Bài tập:

- Mật mã hóa bản rõ sau: **hide the gold in the tree stump**

Mật mã Hill

$$\begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} k_{11} & k_{12} & k_{13} \\ k_{21} & k_{22} & k_{23} \\ k_{31} & k_{32} & k_{33} \end{bmatrix} \begin{bmatrix} p_1 \\ p_2 \\ p_3 \end{bmatrix} \ mod\ 26$$

- Giải thuật sử dụng **m** ký tự liên tiếp của bản rõ (ký hiệu *p*1, *p*2,…,p_m) và thay thế **m** ký tự khác trong bản mật (ký hiệu *c1, c2,…,*c_m)..

$$c_1 = k_{11}p_1 + k_{12}p_2 + k_{13}p_3 \ mod\ 26$$
$$c_2 = k_{21}p_1 + k_{22}p_2 + k_{23}p_3 \ mod\ 26$$
$$c_3 = k_{31}p_1 + k_{32}p_2 + k_{33}p_3 \ mod\ 26$$

- Việc thay thế được thực hiện bởi **m** phương trình tuyến tính. Giả sử m=3

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

Mật mã Hill

- **Mã hóa**: $C = KP \bmod 26$

Với P và C là vector đại diện cho bản rõ và bản mã, còn K là ma trận dùng làm khóa.

- **Giải mã**: $P = K^{-1}C \bmod 26$

Ví dụ:

- Cho P = **PAYMOREMONEY**
- Ma trận khóa:

$$K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

- **Mã hóa**:

- Khóa K là ma trận 3x3 nên m=3

- Bản rõ P được chia thành các cụm 3 chữ cái

+ Mã hóa: PAY (15,0,24) = $\begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix} \begin{bmatrix} 15 \\ 0 \\ 24 \end{bmatrix}$

mod 26

$$= \begin{bmatrix} 11 \\ 13 \\ 18 \end{bmatrix} = \textbf{LNS}$$

- **Thực hiện tương tự => C = LNSHDLEWMTRW**

Mật mã Hill

- C = **LNS**HDL**EWM**TRW Tìm -> P
- Ma trận khóa:

$$K = \begin{pmatrix} 17 & 17 & 5 \\ 21 & 18 & 21 \\ 2 & 2 & 19 \end{pmatrix}$$

- **Giải mã**:

- Chia bản mật thành 3 cụm từ và giải theo $P = K^{-1}C \bmod 26$

- **Tính ma trận nghịch đảo K^{-1} :** $K^{-1} = \begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix}$

- **Giải mã: -** LNS (11,13,18) = $\begin{pmatrix} 4 & 9 & 15 \\ 15 & 17 & 6 \\ 24 & 0 & 17 \end{pmatrix} \begin{bmatrix} 11 \\ 13 \\ 18 \end{bmatrix} \bmod 26 = \begin{bmatrix} 15 \\ 0 \\ 24 \end{bmatrix}$ = **PAY**

- **Các từ còn lại tương tự** P = **PAYMOREMONEY**

Kiến thức bổ trợ

1. Tìm ma trận nghịch đảo A

Cách 1: Dùng ma trận phụ hợp

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

Cách 2: dùng phép biến đổi sơ cấp

$(A \mid E) \rightarrow (E \mid A^{-1})$

Hệ mã Affine

- Gọi **N** là số phần tử của bảng chữ cái, ta có **N=26**.
- **C** và **P** lần lượt là không gian **bản mã** và **bản rõ**
- Đánh số các chữ cái từ **0** đến **N-1**
- Không gian khóa: K = { (a,b): a,b $\in$Z, (a,N)=1}
- Với mỗi khóa **k**=(a,b) $\in$ **K**, **Mã hóa**: $E_k(x)$ **= (a*x + b) mod N**
  - **Giải mã: $D_k(y) = a^{-1} *(y-b) \bmod N$**
  - x,y $\in Z_N$

Hệ mã Affine

- **Ví dụ:**
- Mã hóa P= "TAINGUYEN" với k=(a,b) =(5,7)

- **Mã hóa**: $E_k(x)$ **= (5*x + 7) mod N**

x_i	T	A	I	N	G	U	Y	E	N
	19	0	8	13	6	20	24	4	13

⇒**"T" mã hóa =** 5*19+7 mod 26 =24 **= Y**

⇒**"A" mã hóa =** 5*0+7 mod 26 =7 **= H**

⇒ **SV tính các ký tự còn lại C= YHVULDXVU**

Hệ mã Affine

- **Ví dụ:**
- **Giải mã C= YHVULDXVU** với k=(a,b) =(5,7)
- Tìm nghịch đảo của a và P

Giải mã: $D_k(y) = a^{-1} *(y-b) \bmod N = (5^{-1} *(y - 7)) \bmod 26$

y_i	Y	H	V	U	L	D	X	V	U
	24	7	21	20	11	3	23	21	20

- Ta có 5^{-1} =21 => $D_k(y) = 21*(y - 7)) \bmod 26$

"Y" = 21*(24-7) mod 26 =19 ="T"

"H" =21*(7-7) mod 26 =0 ="A"

⇒ SV tính các ký tự còn lại => P= "TAINGUYEN"

Bài tập

Trong hệ mã Affine, cho $N = 26$, khóa $k = (a, b) = (7,3)$. Hãy:

- Tìm a^{-1} bằng phương pháp Euclide mở rộng.
- Trình bày cách mã hóa bản rõ P = *"NEVER SAY NEVER"* thành bản mã C.
- Trình bày cách giải mã khi nhận được bản mã C.

II. Mã hóa đối xứng hiện đại

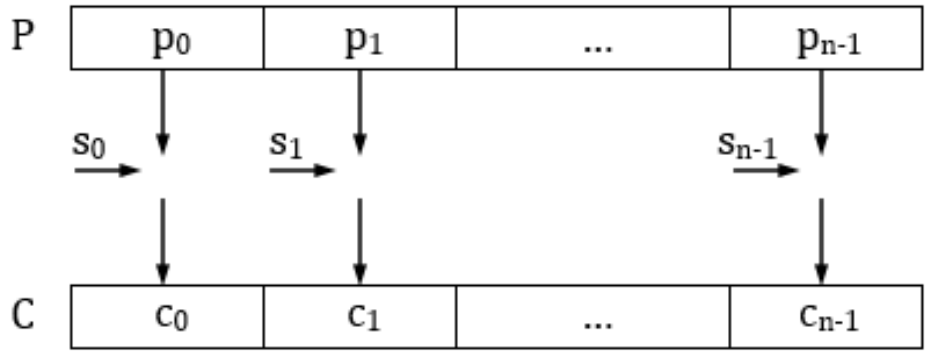
- Mã dòng (Stream Cipher)
- Mã khối (Block Cipher)
- Mã DES (Data Encryption Standard)
- Mã hóa AES (Advanced Encryption Standard)

Giới thiệu

- Xử lý dữ liệu dưới dạng nhị phân
- Mã hóa hiện đại quan tâm đến vấn đề *chống phá mã trong các trường hợp biết trước bản rõ (known-plaintext), hay bản rõ được lựa chọn (chosen-plaintext)*.
- Ví dụ: Sử dụng 8 chữ cái A, B, C, D, E, F, G, H (3 b
  - Plaintext : **head** $\rightarrow$ 111100000011
  - Khóa **K**: **0101**
  - Mã hóa bằng **XOR** $\oplus$
    - *Bản rõ*: 1111 0000 0011 (head)
    - *Khóa*: 0101 0101 0101
    - Bản mã: 1010 0101 0110 (FBCG)
  - Giải mã: **bản mã** $\oplus$ **khóa = bản rõ**
- => Nhận xét: Đơn vị mã hóa là khối 4 bit

Chữ cái	Nhị phân
A	000
B	001
C	010
D	011
E	100
F	101
G	110
H	111

1. Mã dòng (Stream Cipher)



- Kích thước một đơn vị mã hóa: gồm **k** bít.
- **Bản rõ** được chia thành các đơn vị mã hóa: $P \rightarrow p_0 p_1 p_2 \ldots p_{n-1}$ (p_i : k bit)
- **Một bộ sinh dãy số ngẫu nhiên**: dùng một khóa **K** ban đầu để sinh ra các số ngẫu nhiên có kích thước bằng kích thước đơn vị mã hóa: StreamCipher (K) $S \rightarrow s_0 s_1 s_2 \ldots s_{n-1}$ (s_i: k bit)
- Mỗi số ngẫu nhiên được **XOR** với đơn vị mã hóa của bản rõ để có được bản mã.
- Quá trình giải mã được thực hiện ngược lại, bản mã C được XOR với dãy số ngẫu nhiên S để cho ra lại bản rõ ban đầu:
-

$$c_0 = p_0 \oplus s_0,\ c_1 = p_1 \oplus s_1 \ldots \quad ; C = c_0 c_1 c_2 \ldots c_{n-1}$$

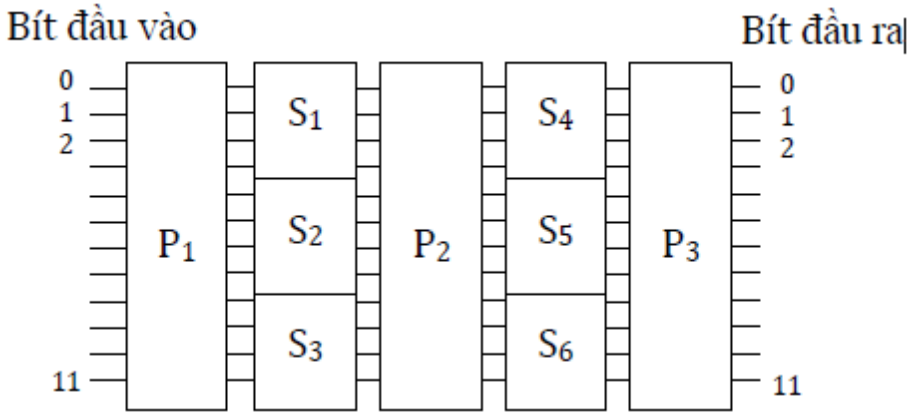
$$p_0 = c_0 \oplus s_0,\ p_1 = c_1 \oplus s_1 \ldots$$

- Một số phương pháp mã hóa dòng tiêu biểu là **A5/1** và **RC4**.

2. Mã khối (Block Cipher)

❑ Mạng SPN (*Substitution-Permutation Network*)

- Sử dụng phép thay thế (substitution, **S-box**) và hoán vị (Permutation, **P-box**).



- Việc kết hợp các S-box và P-box tạo ra hai tính chất quan trọng của mã hóa là tính **khuếch tán** (***diffusion***) và **tính gây lẫn** (***confusion)***.

2. Mã khối (Block Cipher)

❑ Yêu cầu:

- **Sự khuếch tán** (***diffusion***):
  - Một bít của bản rõ tác động đến tất cả các bít của bản mã, **hay** một bít của bản mã chịu tác động của tất cả các bít trong bản rõ.
  - Sử dụng **P-box** *kết hợp* **S-box: Nhằm l**àm giảm tối đa ***mối liên quan*** giữa bản rõ và bản mã, ngăn chặn việc ***suy ra lại khóa***.
- **Sự gây lẫn /Sự hỗn loạn *(confusion)***:
  - Dựa vào sử dụng S-box: Làm phức tạp hóa mối liên quan giữa bản mã và khóa. Do đó cũng ngăn chặn việc ***suy ra lại khóa.***

⇒ **là cơ sở của tất cả các mã khối hiện nay**

Mạng Feistel

- Được mô tả đầu tiên bởi Horst Feistel của công ty IBM vào năm 1973
- Horst Feistel thực sự thành công trong việc thực hiện hóa ý tưởng SPN của Claude Shannon
- Nguyên lý hoạt động: Dựa trên kỹ thuật hoán vị và thay thế nhiều lần trên khối dữ liệu gốc.

Mạng Feistel

❖MÃ HÓA:

- Khối Plaintext (**P)** sẽ chạy qua **n** vòng mã hóa để sinh ra một khối Ciphertext (**C**)
- Chia **P** làm 2 phần: ***P* = (*L0, R0*)**
- Khóa **K** sẽ được sử dụng để sinh ra n khóa con (Subkey):

$$K_i \ (1 \le i \le n)$$

- Tại vòng thứ **i**:
  - Input: R_{i-1} , L_{i-1} *(kết quả của vòng thứ i-1)*
  - Output: $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$

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

- **C = (*Ln, Rn*)**

Mạng Feistel

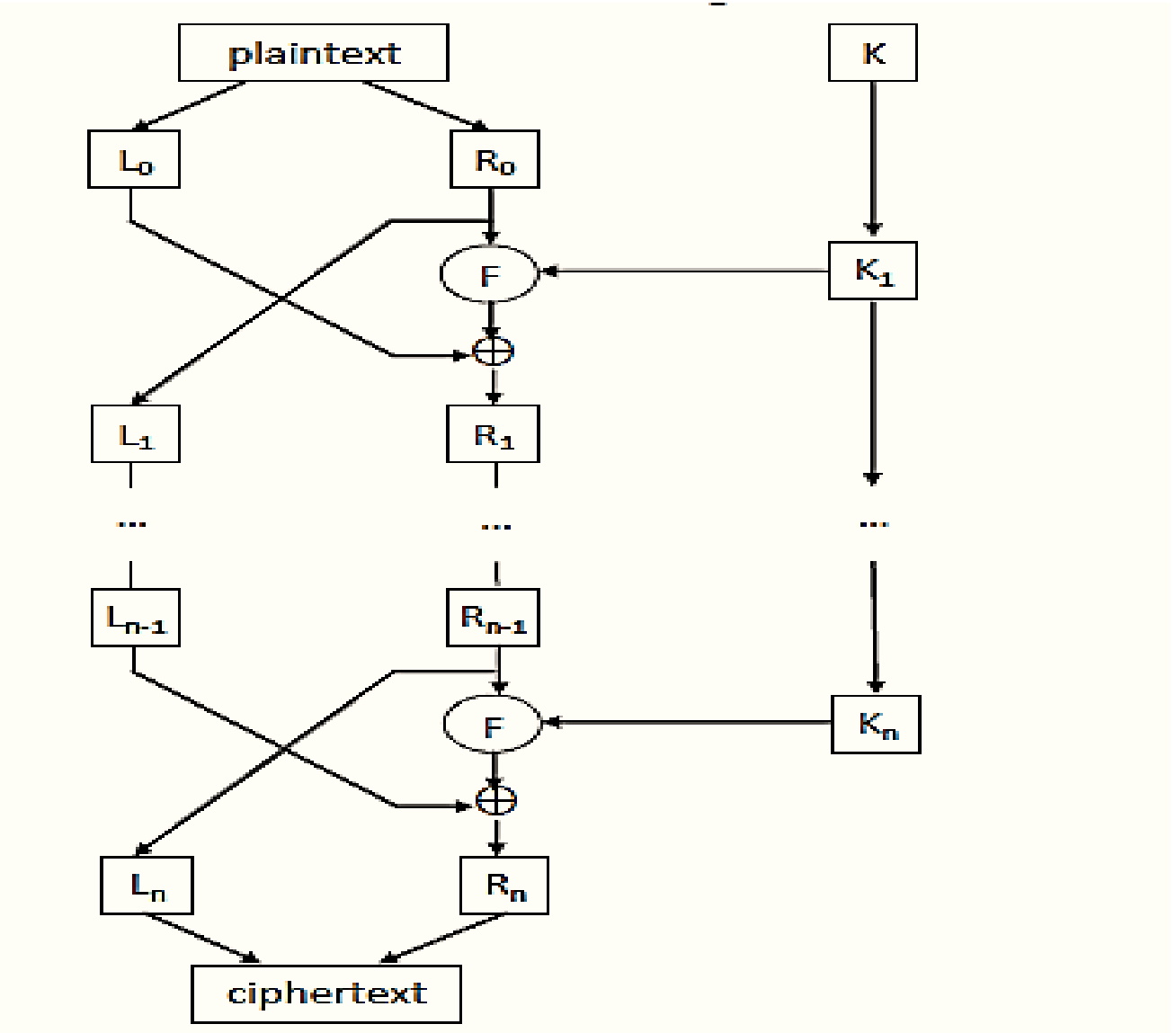
❖GIẢI MÃ:

- Giống với mã hóa nhưng theo chiều ngược lại:
  - **Ciphertext** được sử dụng làm input
  - Các khóa con $\mathbf{k}_i$ được sử dụng theo trình tự ngược lại:

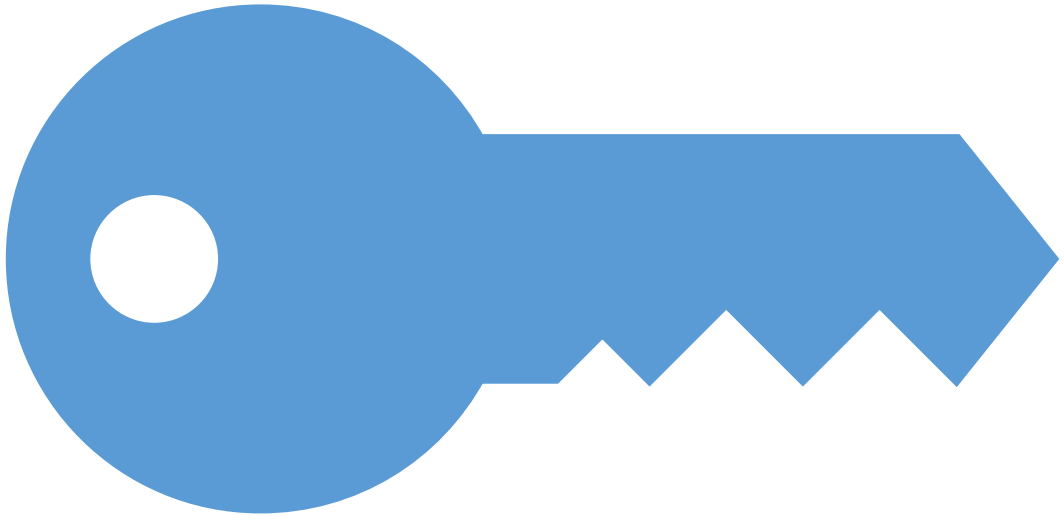
$K_n, k_{n-1}, \ldots, k_2, k_1$

  - $C \rightarrow Ln, Rn$
  - $Ri\text{-}1 = Li$ (*theo mã hóa* $Li = Ri\text{-}1$)
  - $Li\text{-}1 = Ri \oplus F(Ri\text{-}1, Ki)$ (*theo mã hóa* $Ri = Li\text{-}1\ F(Ri\text{-}1, Ki)$)

The Feistel structure

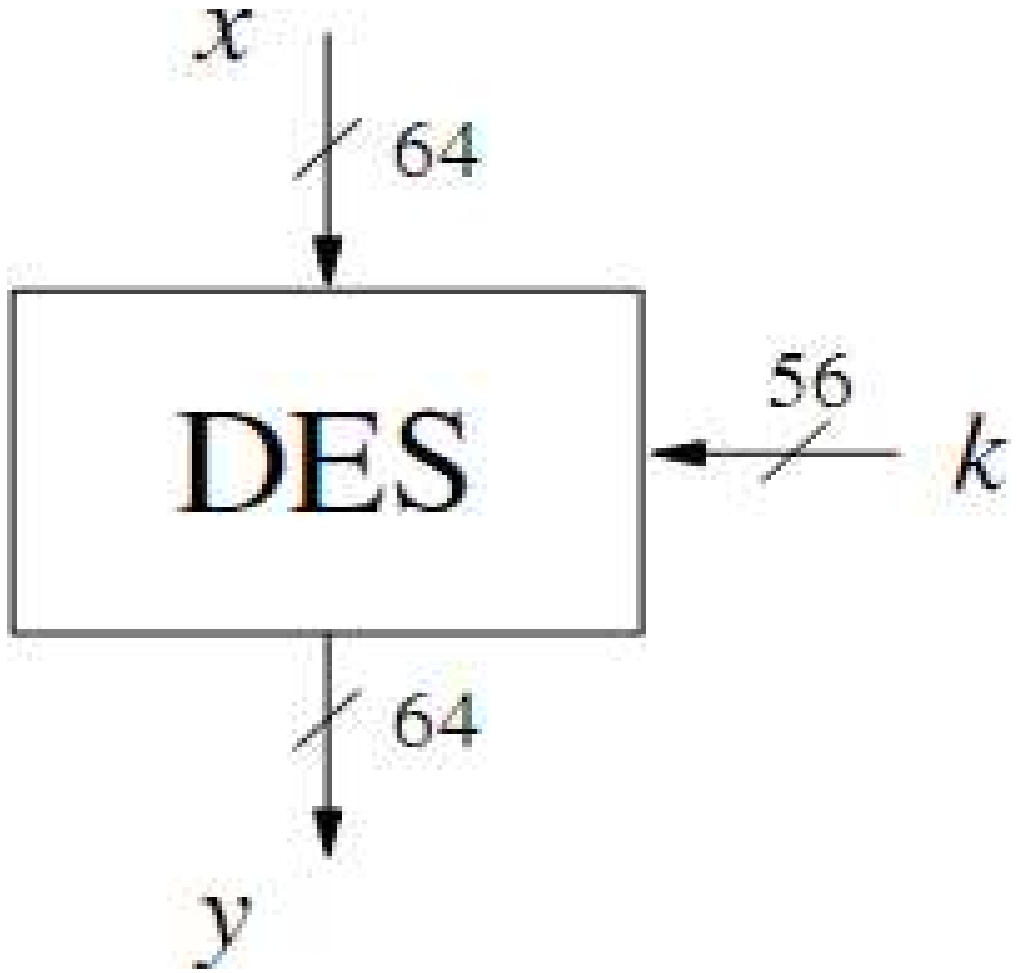


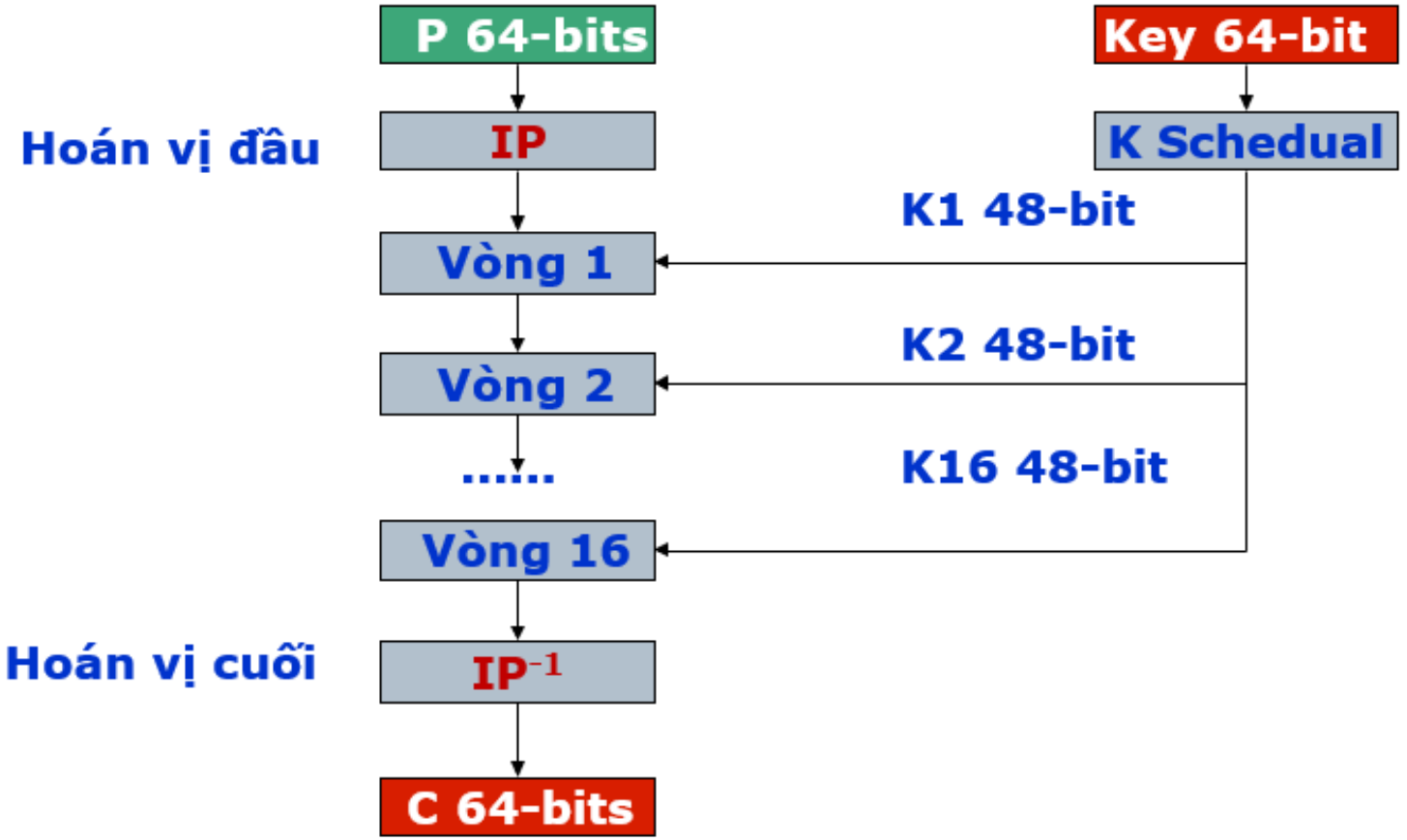
Data Encryption Standard (DES)



Data Encryption Standard (DES)

- Công bố 1977, chuẩn hóa 1979
- Key: 64 bit =8-bit được dung kiểm tra chẵn lẻ +56-bit key
- 64 bit input, 64 bit output
- DES is a symmetric cipher.
- An iterative algorithm.



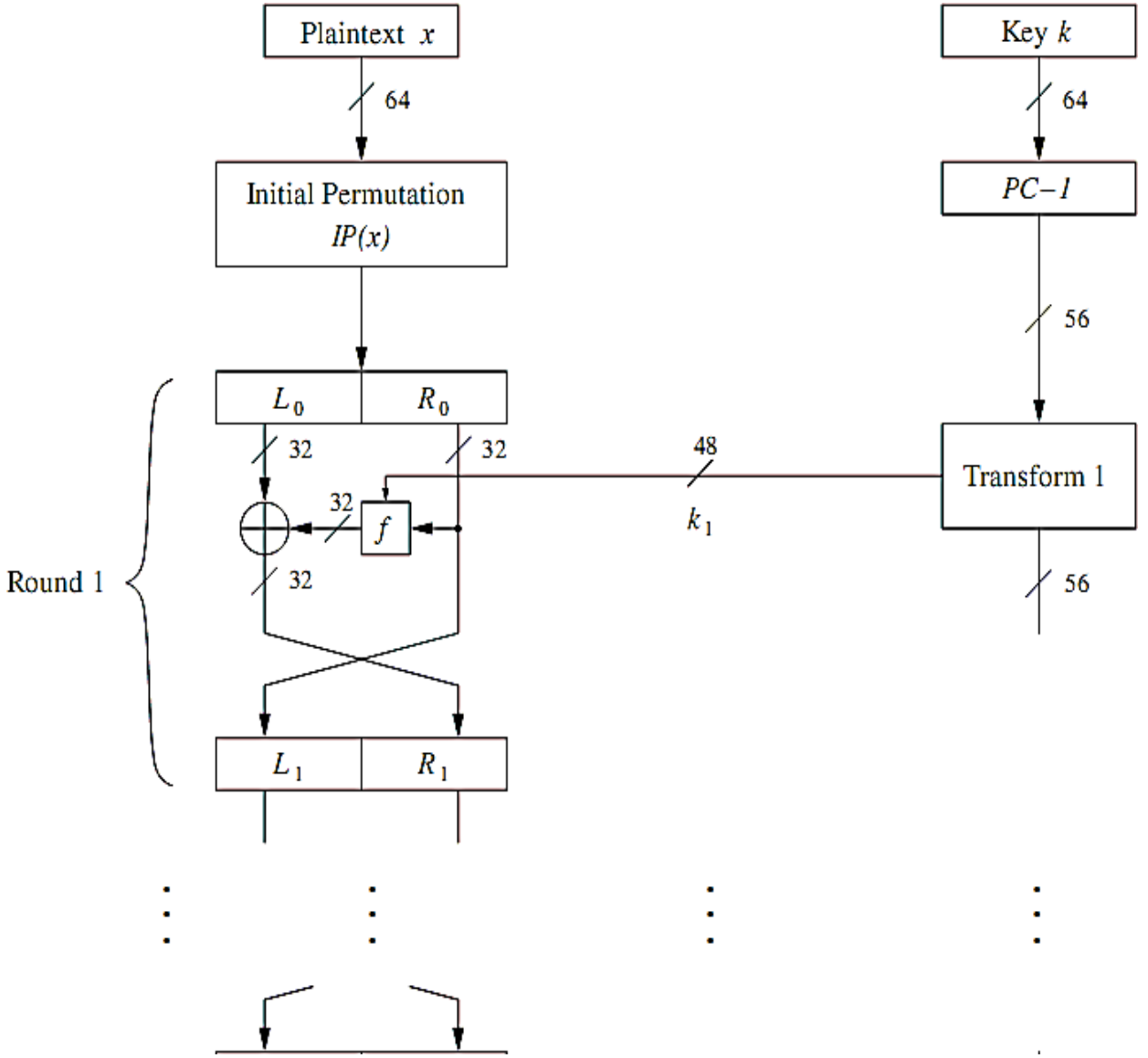


DES Encryption

DES có cấu trúc tương tự như cấu trúc của Feistel nếu bỏ đi hàm **IP** và $\mathbf{IP}^{-1}$ đầu và cuối

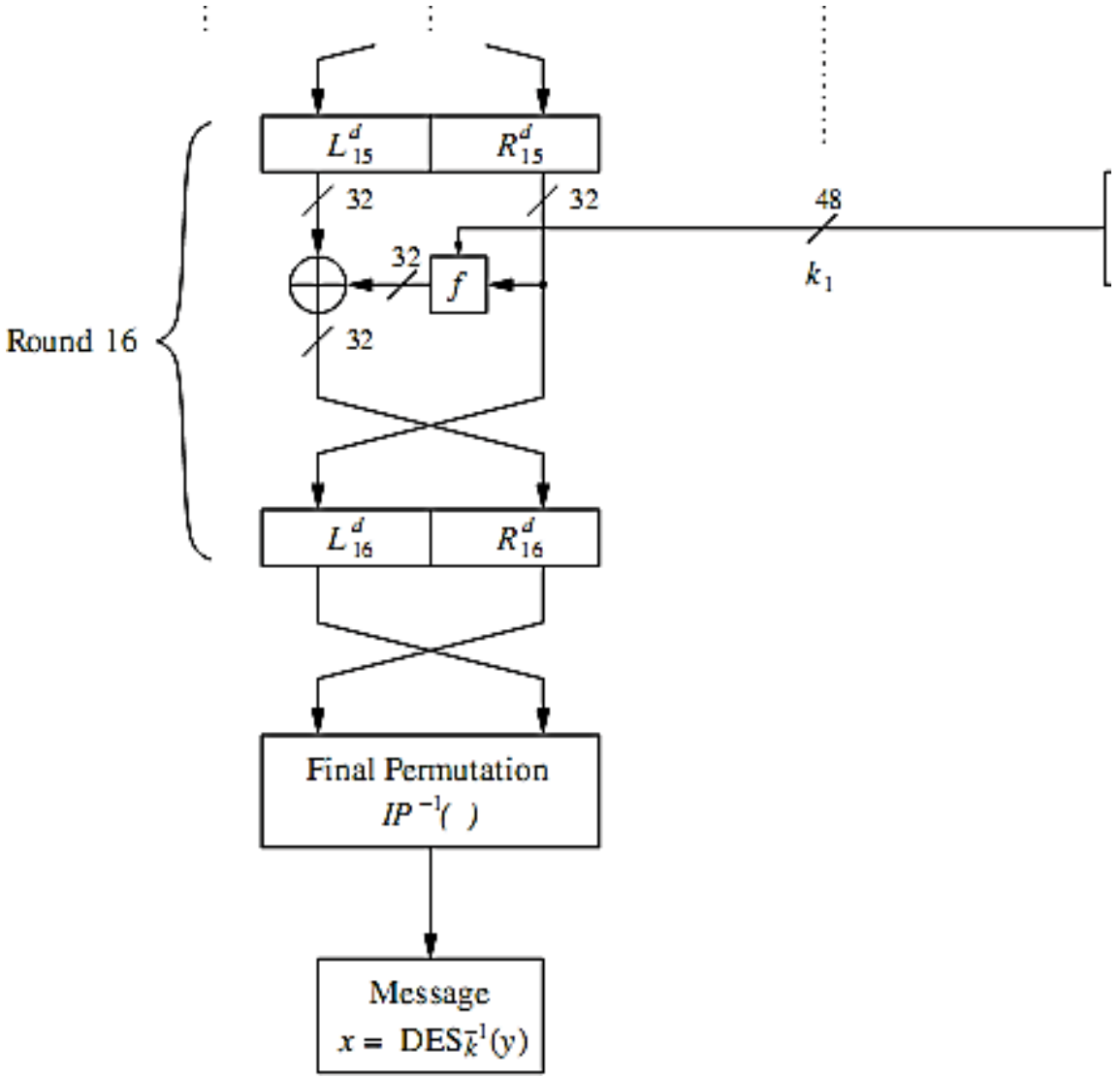
DES - Encryption

- Khối 64 bit của bản rõ được thực hiện biến đổi ở khối hoán vị khởi đầu (IP),.
- Thực hiện 16 vòng biến đổi với cùng một chức năng, trong đó sử dụng các toán tử thay thế và hoán vị.
- Hoán vị cuối (IP-1)



Decryption

- Áp dụng ngược lại với mã hóa:
  - Vòng 1 của giải mã dung khóa của vòng cuối của mã hóa
- Mỗi vòng:
  - Input: R_{n+1}, L_{n+1}
  - Output: P

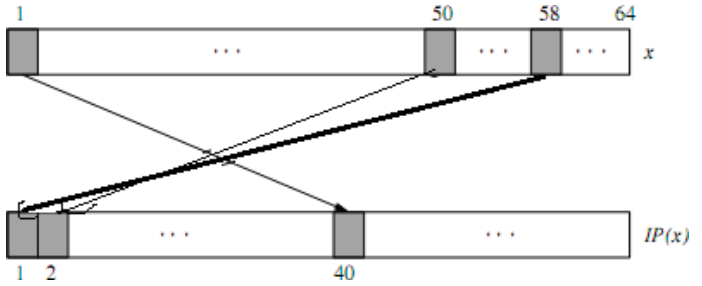


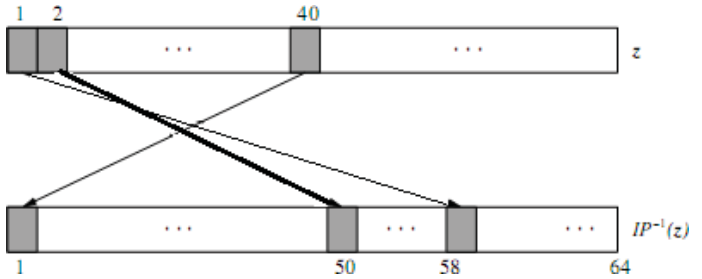
Initial Permutation (IP)

- **IP** là bước đầu tiên của cấu trúc DES
- Bản chất của hàm **IP** là tạo ra một hoán vị của input
- Hàm hoán vị được định nghĩa theo cấu trúc bảng
- Với một bảng **IP** sẽ có bảng tương ứng mô tả trong **IP^{-1}**

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

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





Initial Permutation (IP)

- Đánh số các bít của khối 64 bít (P) theo thứ tự từ trái sang phải:1, …, 63, 64: $b_1b_2…b_{63}b_{64}$
- Bảng 8x8 mô tả hàm IP có chiều dài 64 bits :

Plaintext: => IP

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

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

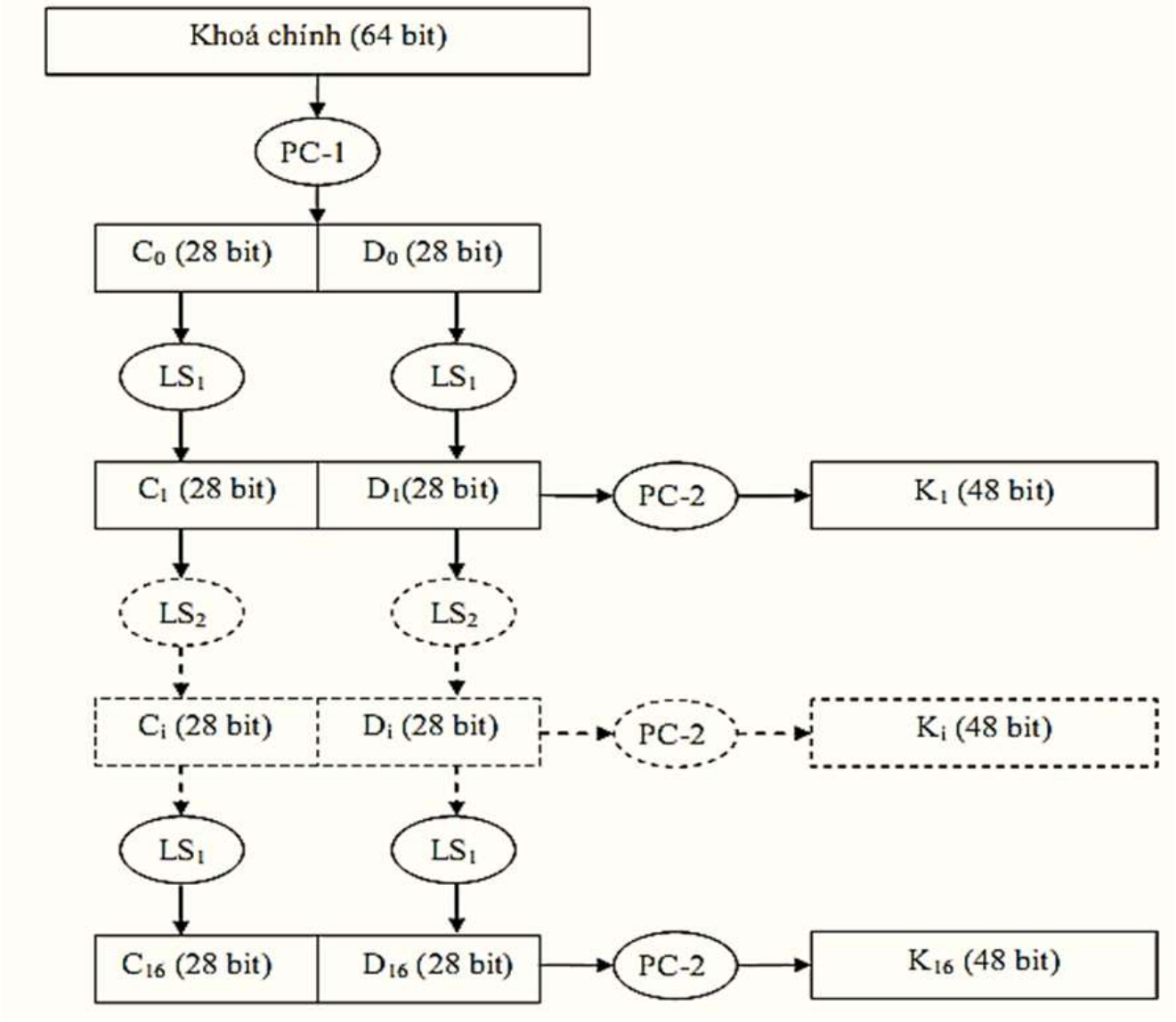
$\Rightarrow IP(b_1b_2…b_{64}) = b_{58}b_{50}…b_7$

IP^{-1}

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

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

Sinh khóa - Key generation



Hàm PC1 (Permuted choice 1)

- Input: 64 bit -> 56 bits
- Bít thứ 8 mỗi byte đc lấy ra

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

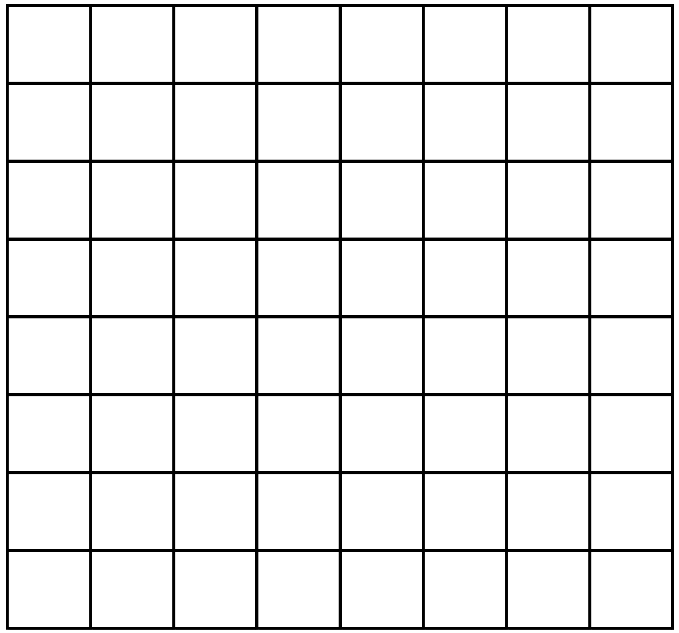
57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Hàm PC1 (Example)

- VD: k = *133457799BBCDFF0 (Hex)*

0001 0011 0011 0100 0101 0111 0111 1001 1001 1011 1011 1100 1101 1111 1111 0000

PC1(k) = 1111 0000 1100 1100 1010 1010 1111 0101 0101 0110 0110 0111 1000 1111



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

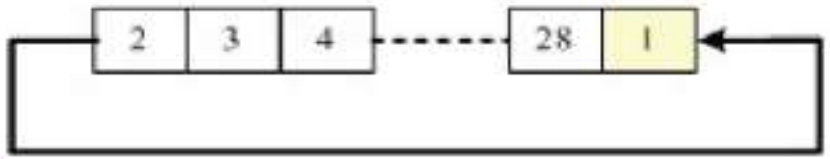
PC1(k)= F0 CC AA F5 56 67 8F

$\Rightarrow C_0$=F0 CC AA F

$\Rightarrow D_0$= 5 56 67 8F

Hàm Left shift:

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1



Single left shift

Two left shift

$\mathbf{PC1}(k) =$ 1111 0000 1100 1100 1010 1010 1111 0101 0101 0110 0110 0111 1000 1111

$C_1 =$ 111 0000 1100 1100 1010 1010 11111

$D_1 =$ 101 0101 0110 0110 0111 1000 11110

left shift

i	Si	PC1K	
		C_i	D_i
0		1111 0000 1100 1100 1010 1010 1111	0101 0101 0110 0110 0111 1000 1111
1	1	1110 0001 1001 1001 0101 0101 1111	1010 1010 1100 1100 1111 0001 1110
2	1	1100 0011 0011 0010 1010 1011 1111	0101 0101 1001 1001 1110 0011 1101
3	2	0000 1100 1100 1010 1010 1111 1111	0101 0110 0110 0111 1000 1111 0101
4	2	0011 0011 0010 1010 1011 1111 1100	0101 1001 1001 1110 0011 1101 0101
5	2	1100 1100 1010 1010 1111 1111 0000	0110 0110 0111 1000 1111 0101 0101
6	2	0011 0010 1010 1011 1111 1100 0011	1001 1001 1110 0011 1101 0101 0101
7	2	1100 1010 1010 1111 1111 0000 1100	0110 0111 1000 1111 0101 0101 0110
8	2	0010 1010 1011 1111 1100 0011 0011	1001 1110 0011 1101 0101 0101 1001
9	1	0101 0101 0111 1111 1000 0110 0110	0011 1100 0111 1010 1010 1011 0011
10	2	0101 0101 1111 1110 0001 1001 1001	1111 0001 1110 1010 1010 1100 1100
11	2	0101 0111 1111 1000 0110 0110 0101	1100 0111 1010 1010 1011 0011 0011
12	2	0101 1111 1110 0001 1001 1001 0101	0001 1110 1010 1010 1100 1100 1111
13	2	0111 1111 1000 0110 0110 0101 0101	0111 1010 1010 1011 0011 0011 1100
14	2	1111 1110 0001 1001 1001 0101 0101	1110 1010 1010 1100 1100 1111 0001
15	2	1111 1000 0110 0110 0101 0101 0111	1010 1010 1011 0011 0011 1100 0111
16	1	1111 0000 1100 1100 1010 1010 1111	0101 0101 0110 0110 0111 1000 1111

i	Ki
1	1B02EFFC7072
2	79AED9DBC9E5
3	55FC8A42CF99
4	72ADD6DB351D
5	7CEC07EB53A8
6	63A53E507B2F
7	EC84B7F618BC
8	F78A3AC13BFB
9	E0DBEBEDE781
10	B1F347BA464F
11	215FD3DED386
12	7571F59467E9
13	97C5D1FABA41
14	5F43B7F2E73A
15	BF918D3D3F0A
16	CB3D8B0E17F5

Hàm PC2

- Hàm PC2:

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

- $K_i \leftarrow PC2(C_i, D_i)$

$C_1 = 1110\ 0001\ 1001\ 1001\ 0101\ 0101\ 1111$

$D_1 = 1010\ 1010\ 1100\ 1100\ 1111\ 0001\ 1110$

$k_1 = 0001\ 1011\ 0000\ 0010\ 1110\ 1111\ 1111\ 1100\ 0111\ 0000\ 0111\ 0010$

PC2 - Example

$C_1 = 1110\ 0001\ 1001\ 1001\ 0101\ 0101\ 1111$

$D_1 = 1010\ 1010\ 1100\ 1100\ 1111\ 0001\ 1110$

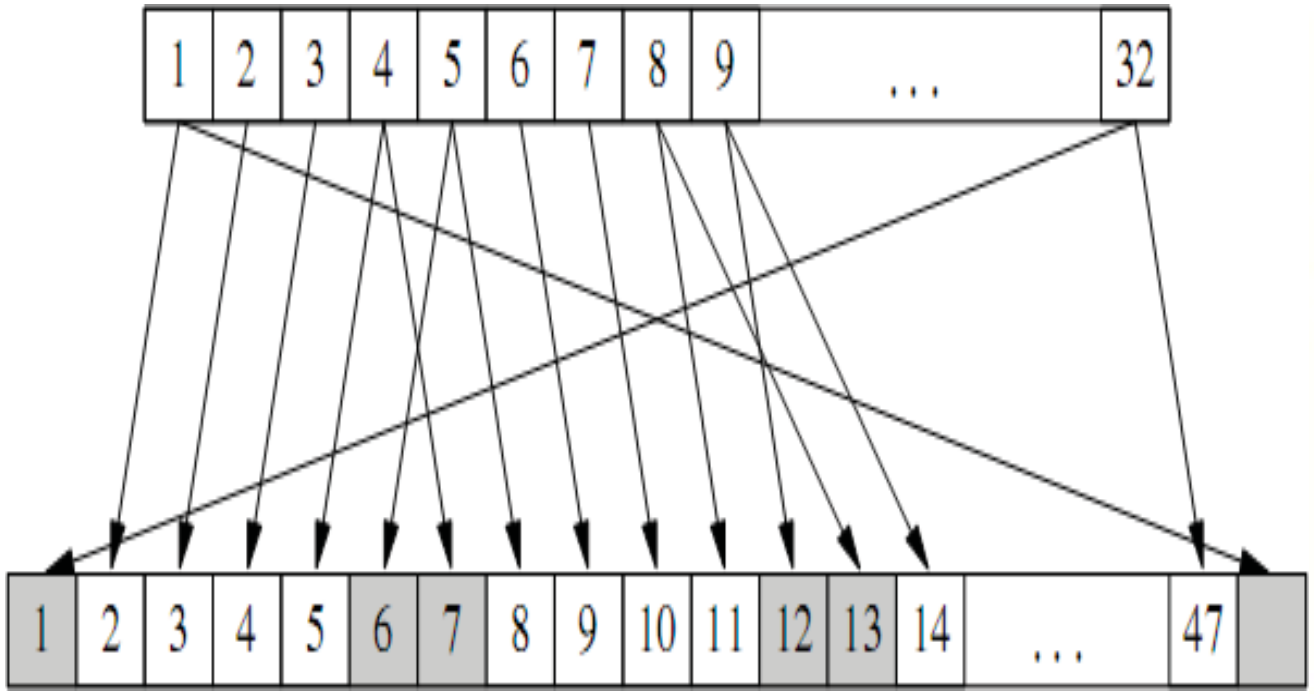
$k_1 = 0001\ 1011\ 0000\ 0010\ 1110\ 1111\ 1111\ 1100\ 0111\ 0000\ 0111\ 0010$

1	2	3	4	5	6	7	8	9	10	11	12	13	14
1	1	1	0	0	0	0	1	1	0	0	1	1	0
15	16	17	18	19	20	21	22	23	24	25	26	27	28
0	1	0	1	0	1	0	1	0	1	1	1	1	1
29	30	31	32	33	34	35	36	37	38	39	40	41	42
1	0	1	0	1	0	1	0	1	1	0	0	1	1
43	44	45	46	47	48	49	50	51	52	53	54	55	56
0	0	1	1	1	1	0	0	0	1	1	1	1	0

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

HÀM MỞ RỘNG E

- Hàm **E**(Expansion)



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

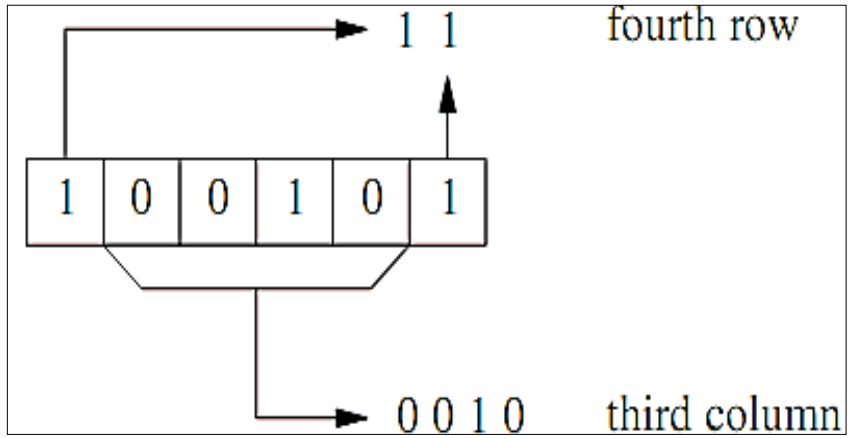
- Hàm E được mô tả bằng một bảng hoán vị, có sự mở rộng bằng cách lặp lại 16 bits

HỘP S-BOX

- Hàm F chứa **8 S-box**
- Mỗi hộp S-Box có **6 bit đầu vào** và **4 bit đầu ra**
- Mỗi hàng trong mỗi hộp là hoán vị của các số nguyên từ 0 đến 15
- Cách tra bảng:
  - 2 bits đầu cuối: Chỉ số dòng
  - 4 bits giữa: Chỉ số cột

S-box S_1

S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13



Decoding of the input 100101_2 by S-box 1

HỘP P (Permutation)

- Hoán vị này mang tính đơn ánh, 1 bit đầu vào sẽ cho ra 1 bit ở đầu ra, không bit nào được sử dụng 2 lần hay bị bỏ qua.
- Cấu tạo hàm Swap: Hoán đổi vị trí hai nửa của chuỗi bits input

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

S-box S_1

S_1	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	14	04	13	01	02	15	11	08	03	10	06	12	05	09	00	07
1	00	15	07	04	14	02	13	01	10	06	12	11	09	05	03	08
2	04	01	14	08	13	06	02	11	15	12	09	07	03	10	05	00
3	15	12	08	02	04	09	01	07	05	11	03	14	10	00	06	13

S-box S_2

S_2	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	15	01	08	14	06	11	03	04	09	07	02	13	12	00	05	10
1	03	13	04	07	15	02	08	14	12	00	01	10	06	09	11	05
2	00	14	07	11	10	04	13	01	05	08	12	06	09	03	02	15
3	13	08	10	01	03	15	04	02	11	06	07	12	00	05	14	09

S-box S_3

S_3	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	10	00	09	14	06	03	15	05	01	13	12	07	11	04	02	08
1	13	07	00	09	03	04	06	10	02	08	05	14	12	11	15	01
2	13	06	04	09	08	15	03	00	11	01	02	12	05	10	14	07
3	01	10	13	00	06	09	08	07	04	15	14	03	11	05	02	12

S-box S_4

S_4	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	07	13	14	03	00	06	09	10	01	02	08	05	11	12	04	15
1	13	08	11	05	06	15	00	03	04	07	02	12	01	10	14	09
2	10	06	09	00	12	11	07	13	15	01	03	14	05	02	08	04
3	03	15	00	06	10	01	13	08	09	04	05	11	12	07	02	14

S-box S_5

S_5	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	02	12	04	01	07	10	11	06	08	05	03	15	13	00	14	09
1	14	11	02	12	04	07	13	01	05	00	15	10	03	09	08	06
2	04	02	01	11	10	13	07	08	15	09	12	05	06	03	00	14
3	11	08	12	07	01	14	02	13	06	15	00	09	10	04	05	03

S-box S_6

S_6	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	12	01	10	15	09	02	06	08	00	13	03	04	14	07	05	11
1	10	15	04	02	07	12	09	05	06	01	13	14	00	11	03	08
2	09	14	15	05	02	08	12	03	07	00	04	10	01	13	11	06
3	04	03	02	12	09	05	15	10	11	14	01	07	06	00	08	13

S-box S_7

S_7	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	04	11	02	14	15	00	08	13	03	12	09	07	05	10	06	01
1	13	00	11	07	04	09	01	10	14	03	05	12	02	15	08	06
2	01	04	11	13	12	03	07	14	10	15	06	08	00	05	09	02
3	06	11	13	08	01	04	10	07	09	05	00	15	14	02	03	12

S-box S_8

S_8	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	13	02	08	04	06	15	11	01	10	09	03	14	05	00	12	07
1	01	15	13	08	10	03	07	04	12	05	06	11	00	14	09	02
2	07	11	04	01	09	12	14	02	00	06	10	13	15	03	05	08
3	02	01	14	07	04	10	08	13	15	12	09	00	03	05	06	11

DES Example

Round	K_i	L_i	R_i
IP		5a005a00	3cf03c0f
1	1e030f03080d2930	3cf03c0f	bad22845
2	0a31293432242318	bad22845	99e9b723
3	23072318201d0c1d	99e9b723	0bae3b9e
4	05261d3824311a20	0bae3b9e	42415649
5	3325340136002c25	42415649	18b3fa41
6	123a2d0d04262a1c	18b3fa41	9616fe23
7	021f120b1c130611	9616fe23	67117cf2
8	1c10372a2832002b	67117cf2	c11bfc09
9	04292a380c341f03	c11bfc09	887fbc6c
10	2703212607280403	887fbc6c	600f7e8b
11	2826390c31261504	600f7e8b	f596506e
12	12071c241a0a0f08	f596506e	738538b8
13	300935393c0d100b	738538b8	c6a62c4e
14	311e09231321182a	c6a62c4e	56b0bd75
15	283d3e0227072528	56b0bd75	75e8fd8f
16	2921080b13143025	75e8fd8f	25896490
IP^{-1}		da02ce3a	89ecac3b

Sự an toàn của DES

- Không gian khóa quá nhỏ, tức là thuật toán dễ bị tổn thương trước các cuộc tấn công brute-force.
- Các tiêu chí thiết kế của S-box được giữ bí mật và có thể đã tồn tại một cuộc tấn công phân tích khai thác các tính chất toán học của S-box, nhưng chỉ có các nhà thiết kế DES mới biết.

Advanced Encryption Standard (AES)

Advanced Encryption Standard (AES)

- 1997, NIST kêu gọi xây dựng một hệ mật mã mới để thay thế DES
- Hệ Rijndael của Daemen và Rijmen được lựa chọn
- **2001**, hệ **Rijndael** được chuẩn hóa thành AES
  - Dựa trên lý thuyết "Trường Galois"
  - **Input, Output**: Khối 128 bit
  - **Cipher Key**: 128, 192, 256 bit
  - **n** vòng lặp mã hóa, phụ thuộc vào chiều dài khóa
    - Khóa 128 bit, **n = 10**
    - Khóa 192 bit, **n = 12**
    - Khóa 256 bit, **n = 14**
- Mỗi vòng kết hợp **Hoán vị + Thay thế**

Các hàm trong AES

AddRoundKey()

MixColumns(); Inv MixColumns()

ShiftRows(); InvShiftRows()

SubBytes(); InvSubBytes()

SubWord()

RotWord()

Rcon[]

Nb: Số các cột (AES sử dụng Nb=4)

Nr: Số lượng vòng lặp của thuật toán

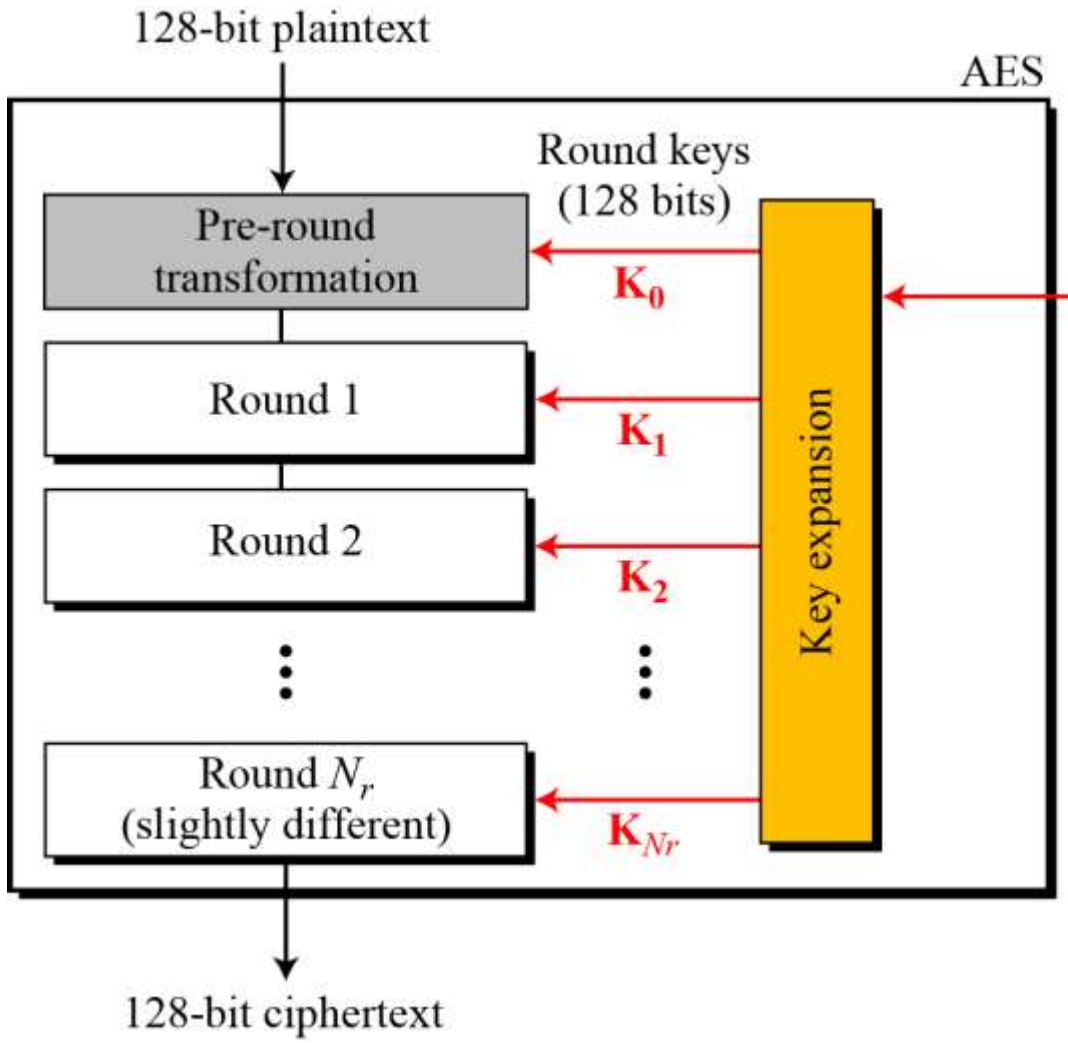
⊕: Phép cộng XOR

• : Phép nhân trên trường GF

Encryption - AES

- Các khóa vòng (**Round key**) dùng trong các vòng lặp được sinh ra từ khóa chính AES sử dụng thủ tục sinh khóa **Rijndael**.
- Bản rõ (Input) được copy vào mảng state
- Vòng khởi tạo (Initial Round): Thực hiện hàm AddRoundKey- trong đó mỗi byte trong state được kết hợp với khóa vòng sử dụng phép XOR.
- Các vòng lặp chính (**Rounds**): Thực hiện **Nr** vòng (10,12, hoặc 14).
  - 4 hàm (SubBytes; ShiftRows; MixComumns; AddRoundKey) **N_{r-1} vòng**
  - 3 hàm (SubBytes; ShiftRows; AddRoundKey) cho vòng cuối

Encryption - AES



Cipher key
(128, 192, or 256 bits)

Nr	Key size
10	128
12	192
14	256

Relationship between number of rounds and cipher key size

Bytes - *AES*

Bit	Ký tự	Bit	Ký tự	Bit	Ký tự	Bit	Ký tự
0000	0	0100	4	1000	8	1100	C
0001	1	0101	5	1001	9	1101	D
0010	2	0110	6	1010	a	1110	E
0011	3	0111	7	1011	b	1111	f

- **là Chuỗi 8 bit**, chúng là các phần tử trên trường hữu hạn và được biểu diễn thành đa thức:
- $b_7x^7 + b_6x^6 + b_5x^5 + b_4x^4 + b_3x^3 + b_2x^2 + b_1x^1 + b_0x^0 = \sum_{i=0}^{7} b_i x^i$
- Ví dụ: {01100011} được biểu diễn thành: **$x^6 + x^5 + x + 1$**
- Các byte được biểu diễn bằng hai ký tự, mỗi ký tự là ký hiệu của hệ **HEX** cho 4 bit
- Ví dụ: byte {**0010 1010**} được biểu diễn thành {**2a**}

Arrays of bytes- *AES*

- Mảng các byte được biểu diễn dưới dạng: $a_0\, a_1\, a_2 \ldots a_{15}$
- Các bytes và thứ tự bit trong byte được lấy từ **Input** 128 bit

***input*0 *input*1 *input*2 … *input*126 *input*127**

Trong đó:

*a*0 = {*input*0, *input*1, …, *input*7};
*a*1 = {*input*8, *input*9, …, *input*15};
…
*a*15 = {*input*120, *input*121, …, *input*127}.

Input bit sequence	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20	21	22	23	…
Byte number	0								1								2								…
Bit numbers in byte	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	7	6	5	4	3	2	1	0	…

State - AES

- Là mảng 2 chiều các bytes: mỗi hàng có **Nb** byte (Nb: Số lượng các word 32-byte = input chia cho 32)
- Kí hiệu là s:

$s[r, c] = in[r + 4c]$ với $0 \le r < 4$ and $0 \le c < Nb$,

$out[r + 4c] = s[r, c]$ Với $0 \le r < 4$ and $0 \le c < Nb$.

input bytes

in_0	in_4	in_8	in_{12}
in_1	in_5	in_9	in_{13}
in_2	in_6	in_{10}	in_{14}
in_3	in_7	in_{11}	in_{15}

→

State array

$s_{0,0}$	$s_{0,1}$	$s_{0,2}$	$s_{0,3}$
$s_{1,0}$	$s_{1,1}$	$s_{1,2}$	$s_{1,3}$
$s_{2,0}$	$s_{2,1}$	$s_{2,2}$	$s_{2,3}$
$s_{3,0}$	$s_{3,1}$	$s_{3,2}$	$s_{3,3}$

→

output bytes

out_0	out_4	out_8	out_{12}
out_1	out_5	out_9	out_{13}
out_2	out_6	out_{10}	out_{14}
out_3	out_7	out_{11}	out_{15}

Cơ sở toán

1. Addition (⊕)

Trong AES, phép cộng bit sử dung phép XOR

Ví dụ:

$(x^6 + x^4 + x^2 + x + 1) + (x^7 + x + 1) = x^7 + x^6 + x^4 + x^2$

{01010111} ⊕ {10000011} = {11010100}

{57} ⊕ {83} = {d4} (hexadecimal notation).

Cơ sở toán (tt)

2. Multiplication (•) GF(2^8)

- Tương ứng nhân các đa thức với đa thức bất khả qui

m(x) = $(x^8 + x^4 + x^3 + x + 1)$.
Or {01} {1b}

VD : {57} • {83} = {c1}

$$(x^6+x^4+x^2+x+1)(x^7+x+1) = x^{13}+x^{11}+x^9+x^8+x^7+$$
$$x^7+x^5+x^3+x^2+x+$$
$$x^6+x^4+x^2+x+1$$
$$= x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1$$

and

$$x^{13}+x^{11}+x^9+x^8+x^6+x^5+x^4+x^3+1 \text{ modulo } (x^8+x^4+x^3+x+1)$$
$$= x^7+x^6+1.$$

Cơ sở toán (tt)

3. **Multiplication by *x*** ***{đa thức }* • {02}**

- *Nếu* $b_7 = 0$ - Left shift đa thức **1 bit**
- *Nếu* $b_7 = 1$ - **Left shift** đa thức **1 bit** và **XOR** m(x) hay {1b}

Ví dụ: {57} • {13} = {fe}

{57} • {02} = xtime({57}) = {ae}

{57} • {04} = xtime({ae}) = {47}

{57} • {08} = xtime({47}) = {8e}

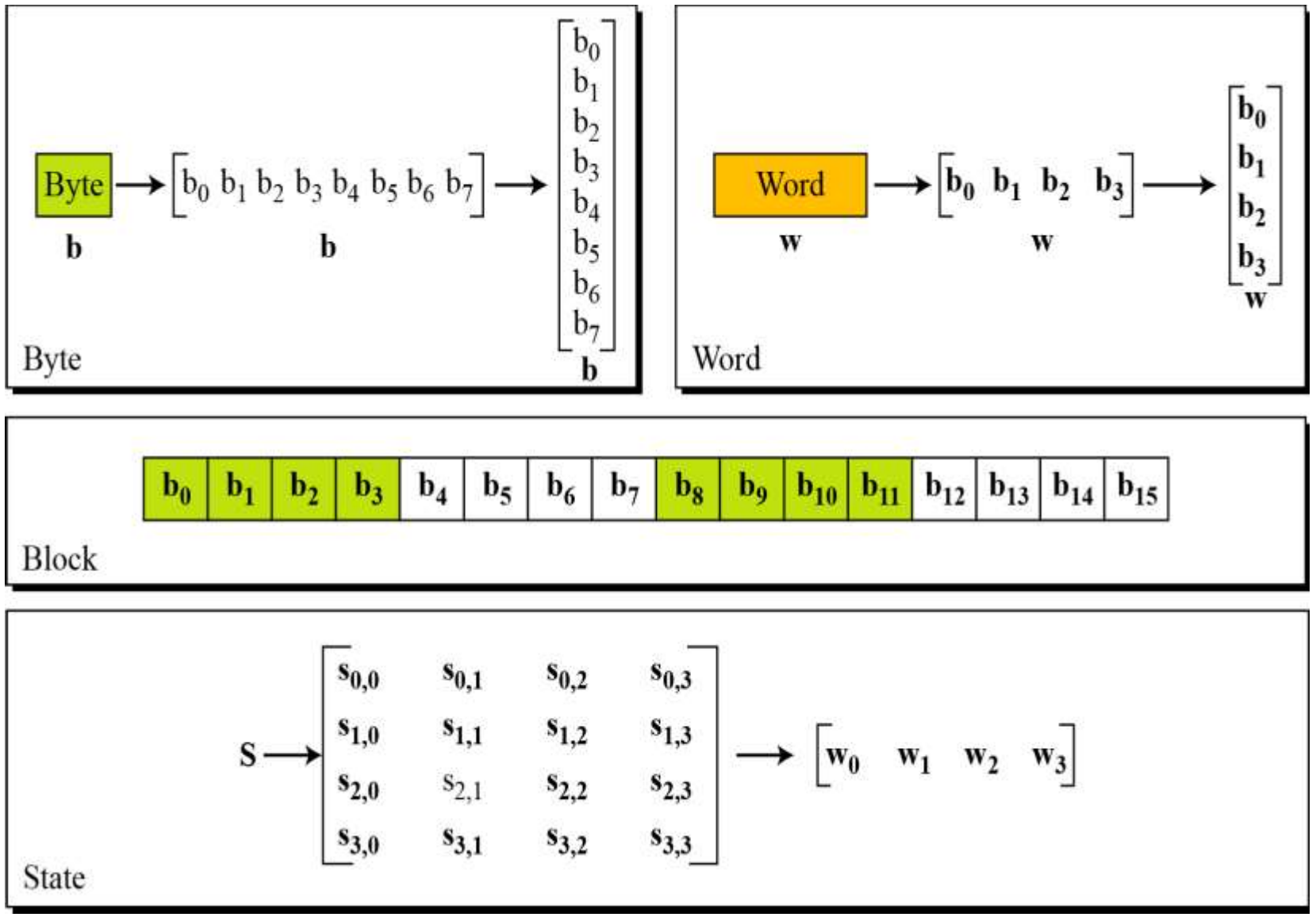
{57} • {10} = xtime({8e}) = {07}

=> {57} • {13} = {57} • ({01} ⊕ {02} ⊕ {10})

= {57} ⊕ {ae} ⊕ {07}

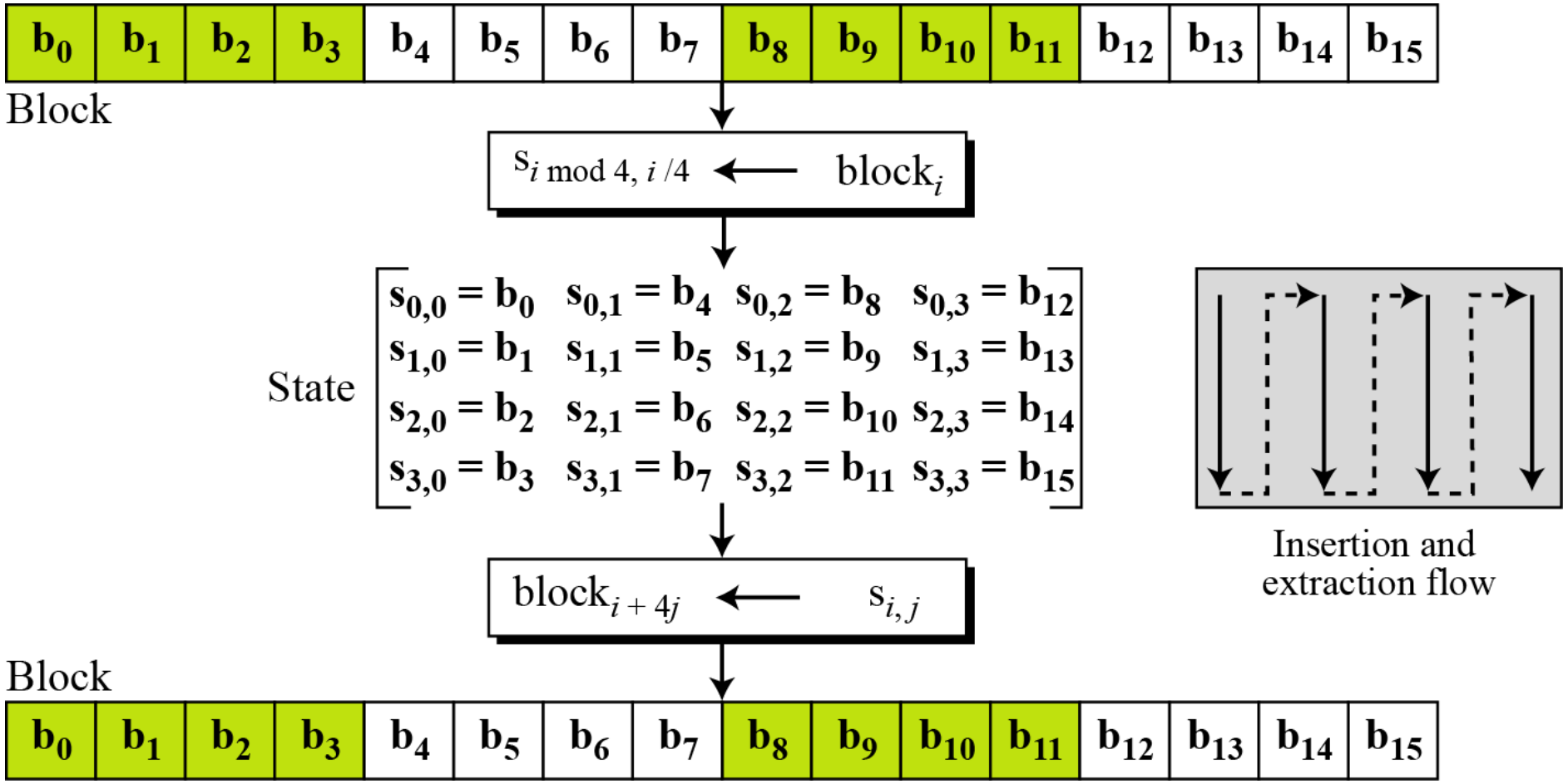
= {fe}.

State - AES


Byte
b
b0 b1 b2 b3 b4 b5 b6 b7
b
b0
b1
b2
b3
b4
b5
b6
b7
b
Byte
Word
w
b0 b1 b2 b3
w
b0
b1
b2
b3
w
Word
b0 b1 b2 b3 b4 b5 b6 b7 b8 b9 b10 b11 b12 b13 b14 b15
Block
S
s0,0 s0,1 s0,2 s0,3
s1,0 s1,1 s1,2 s1,3
s2,0 s2,1 s2,2 s2,3
s3,0 s3,1 s3,2 s3,3
w0 w1 w2 w3
State

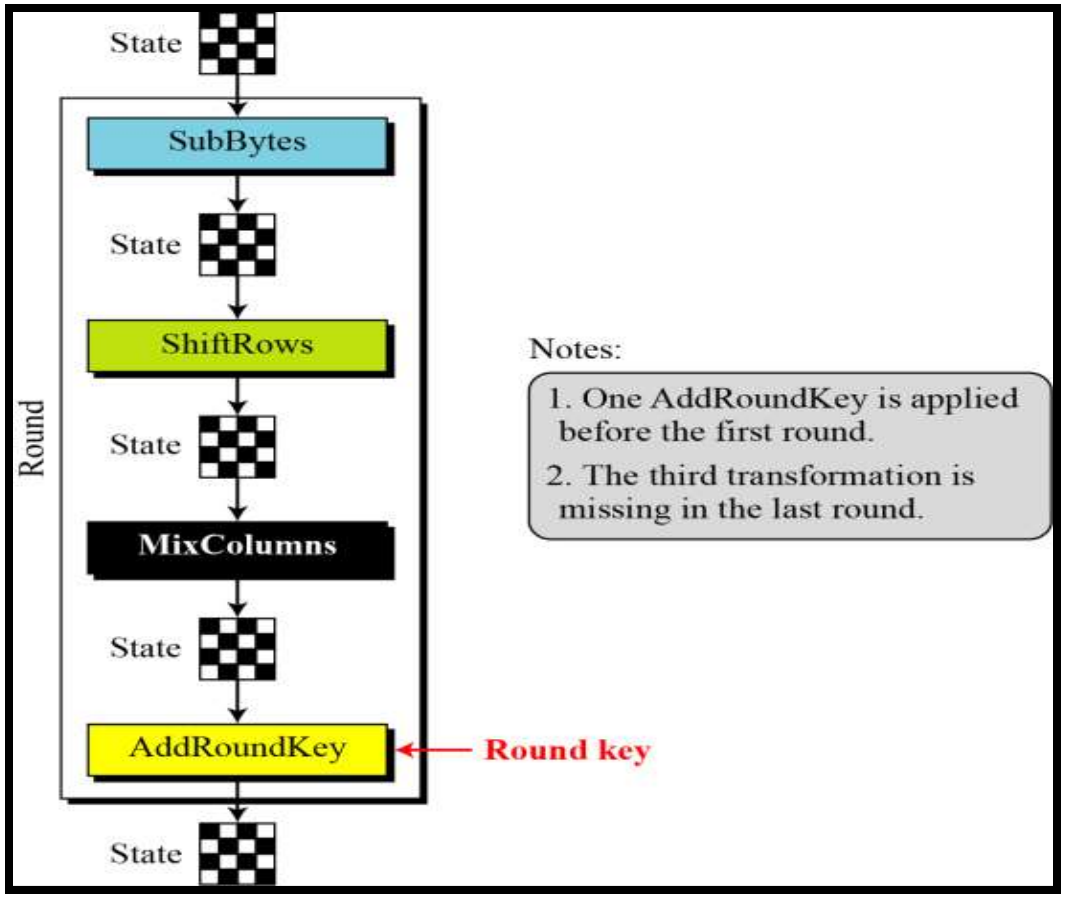
State - AES

Block → state** and **state → block



Insertion and extraction flow

One round- AES



Chi tiết một vòng lặp (từ vòng 1 đến N – 1)

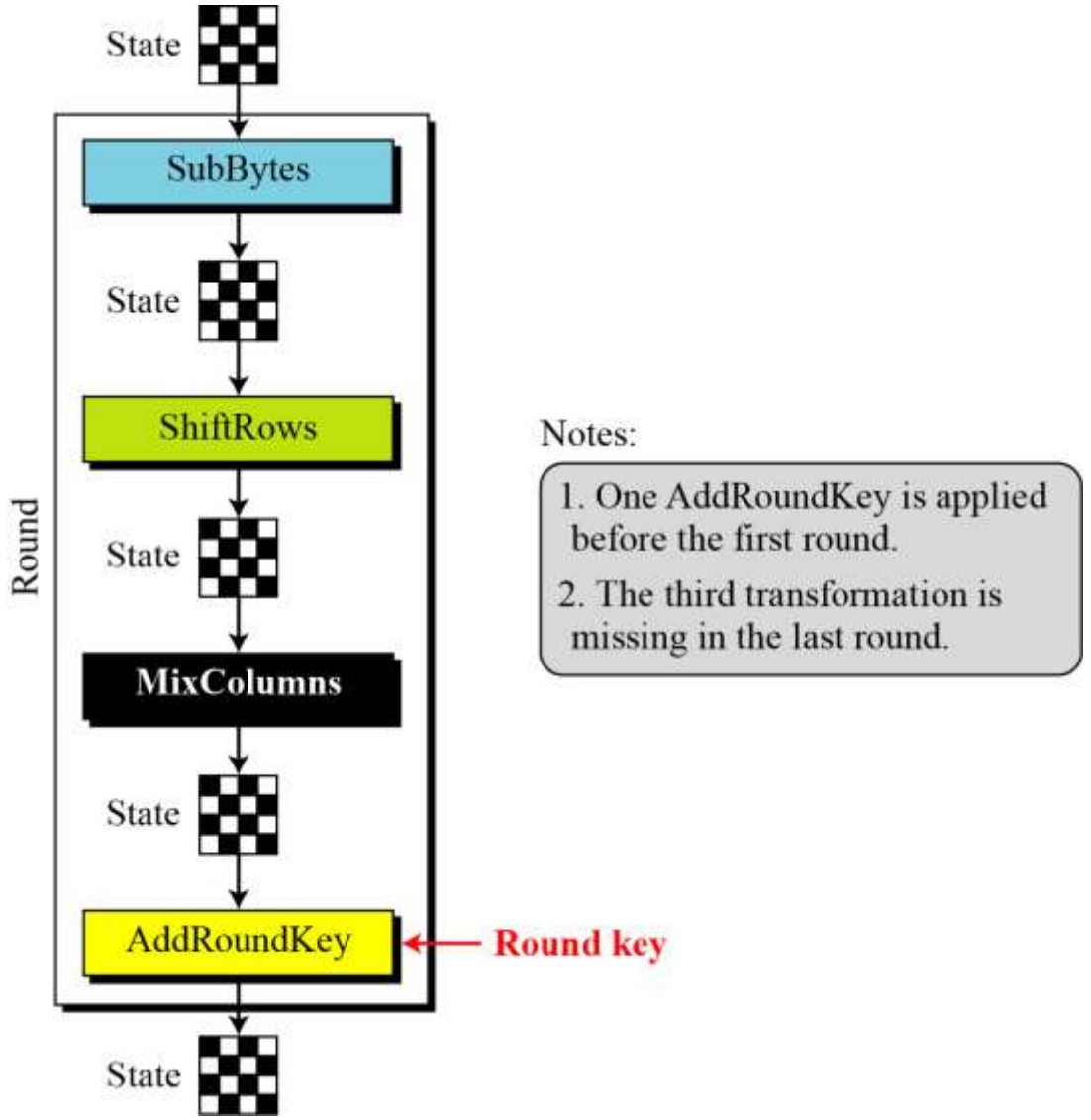
1. **Substitute bytes**
2. **ShiftRows**
3. **MixColumns**
4. **AddRoundKey**

Riêng vòng thứ N không có phép **MixColumns.**

Khóa (bit)	128	192	256
Input (bit)	128	128	128
Số vòng lặp	10	12	14
Khóa vòng lặp (bit)	128	128	128
Khóa mở rộng (bytes)	176	208	240

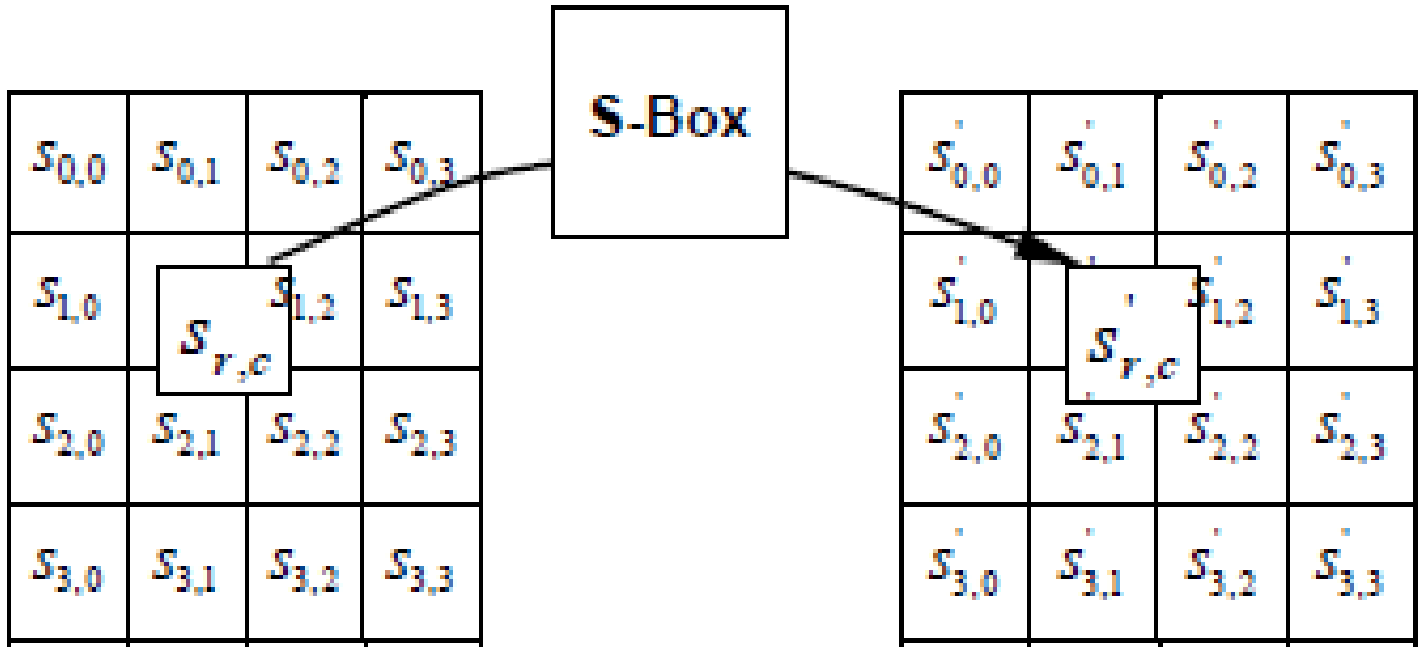
AddRoundkey- AES

- 128 bit của **state** XOR với 128 bit của khóa vòng



Subbyte- AES

- Hàm thay thế phi tuyến tính, trong đó mỗi byte trong *state* được thay thế bằng một byte khác sử dụng bảng tham chiếu **S-box** hay $\mathbf{b_{ij} = S(a_{ij})}$



SubBytes table- AES

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

VD- Subbyte

Ma trận trạng thái

19	a0	9a	e9
3d	f4	c6	f8
e3	e2	8d	48
be	2b	2a	08

SubBytes

Kết quả SubBytes

d4	e0	b8	1e
27	bf	b4	41
11	98	5d	52
ae	f1	e5	30

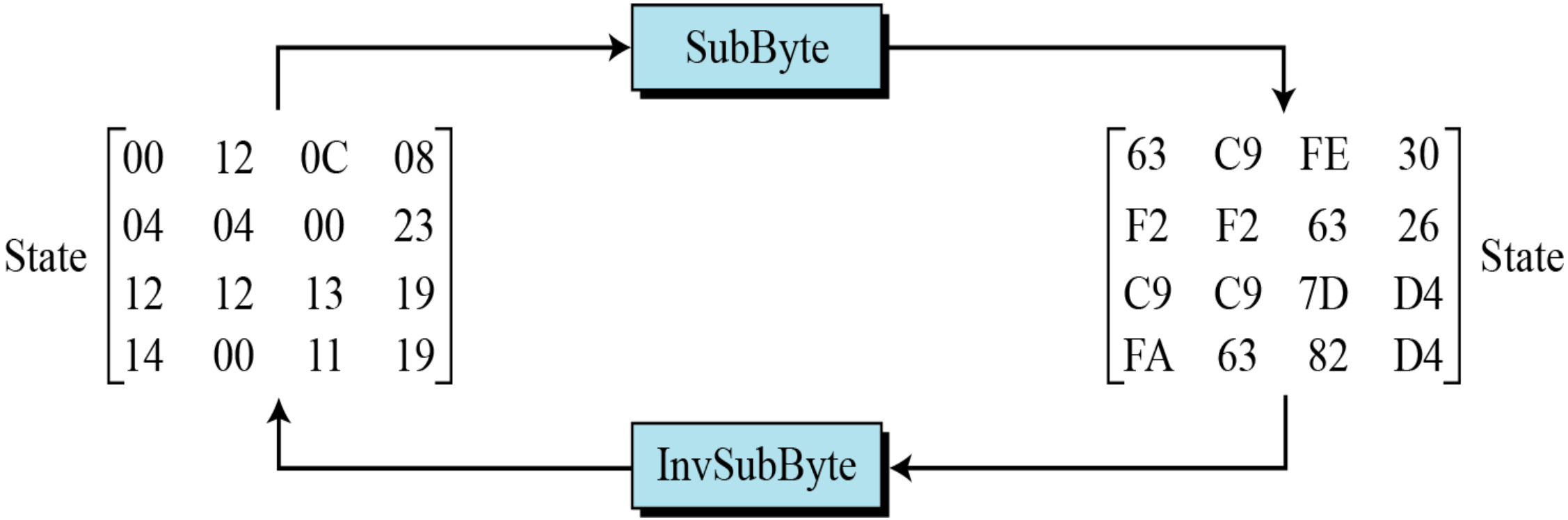
S-box

x \ y	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
0	63	7c	77	7b	f2	6b	6f	c5	30	01	67	2b	fe	d7	ab	76
1	ca	82	c9	7d	fa	59	47	f0	ad	d4	a2	af	9c	a4	72	c0
2	b7	fd	93	26	36	3f	f7	cc	34	a5	e5	f1	71	d8	31	15
3	04	c7	23	c3	18	96	05	9a	07	12	80	e2	eb	27	b2	75
4	09	83	2c	1a	1b	6e	5a	a0	52	3b	d6	b3	29	e3	2f	84
5	53	d1	00	ed	20	fc	b1	5b	6a	cb	be	39	4a	4c	58	cf
6	d0	ef	aa	fb	43	4d	33	85	45	f9	02	7f	50	3c	9f	a8
7	51	a3	40	8f	92	9d	38	f5	bc	b6	da	21	10	ff	f3	d2
8	cd	0c	13	ec	5f	97	44	17	c4	a7	7e	3d	64	5d	19	73
9	60	81	4f	dc	22	2a	90	88	46	ee	b8	14	de	5e	0b	db
a	e0	32	3a	0a	49	06	24	5c	c2	d3	ac	62	91	95	e4	79
b	e7	c8	37	6d	8d	d5	4e	a9	6c	56	f4	ea	65	7a	ae	08
c	ba	78	25	2e	1c	a6	b4	c6	e8	dd	74	1f	4b	bd	8b	8a
d	70	3e	b5	66	48	03	f6	0e	61	35	57	b9	86	c1	1d	9e
e	e1	f8	98	11	69	d9	8e	94	9b	1e	87	e9	ce	55	28	df
f	8c	a1	89	0d	bf	e6	42	68	41	99	2d	0f	b0	54	bb	16

invSubBytes table- AES

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

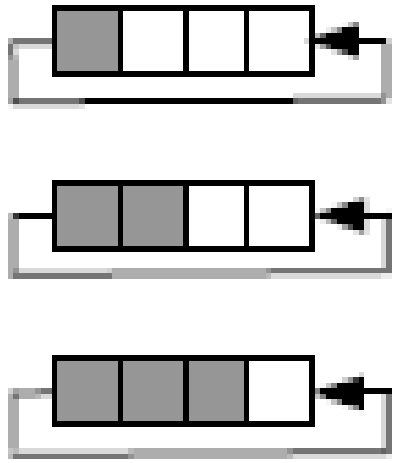
SubBytes table



Shiftrows - AES

- Là hàm đổi chỗ, trong đó mỗi dòng trong state được dịch một số bước theo chu kỳ;

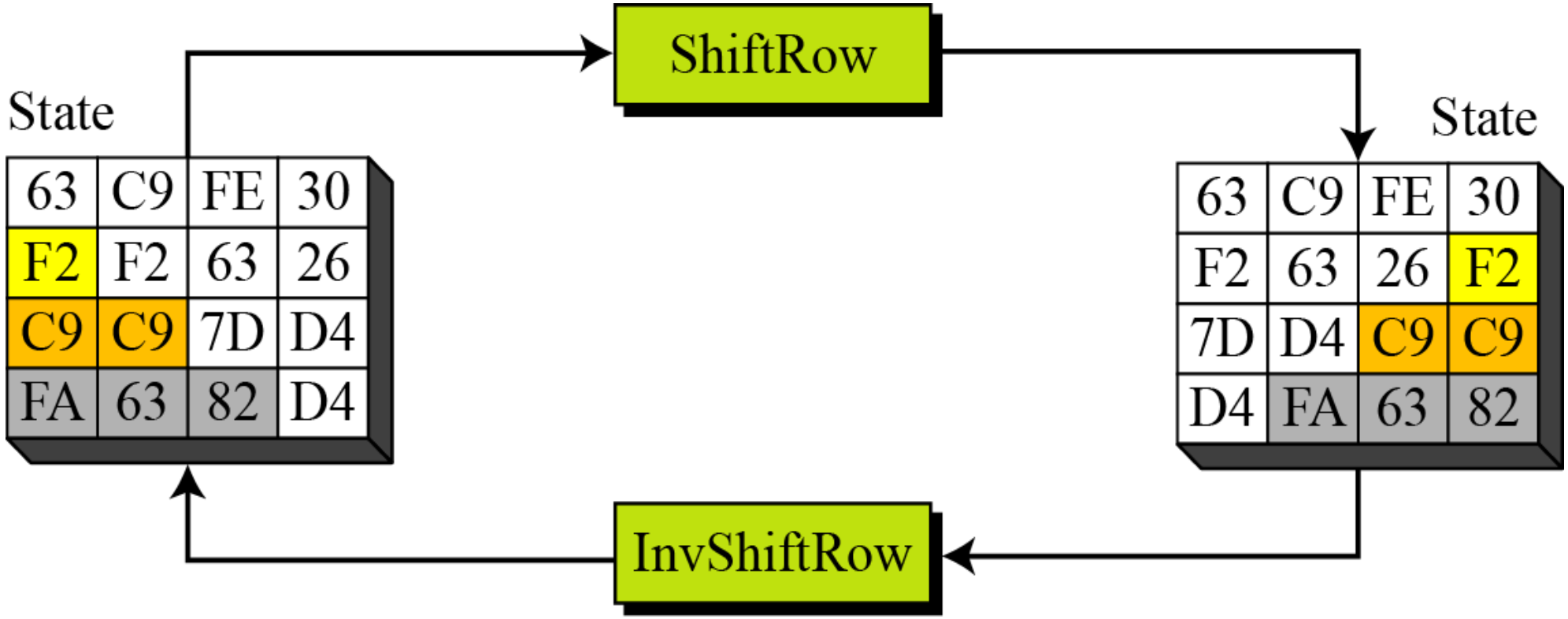
$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,0}$	$S_{1,1}$	$S_{1,2}$	$S_{1,3}$
$S_{2,0}$	$S_{2,1}$	$S_{2,2}$	$S_{2,3}$
$S_{3,0}$	$S_{3,1}$	$S_{3,2}$	$S_{3,3}$



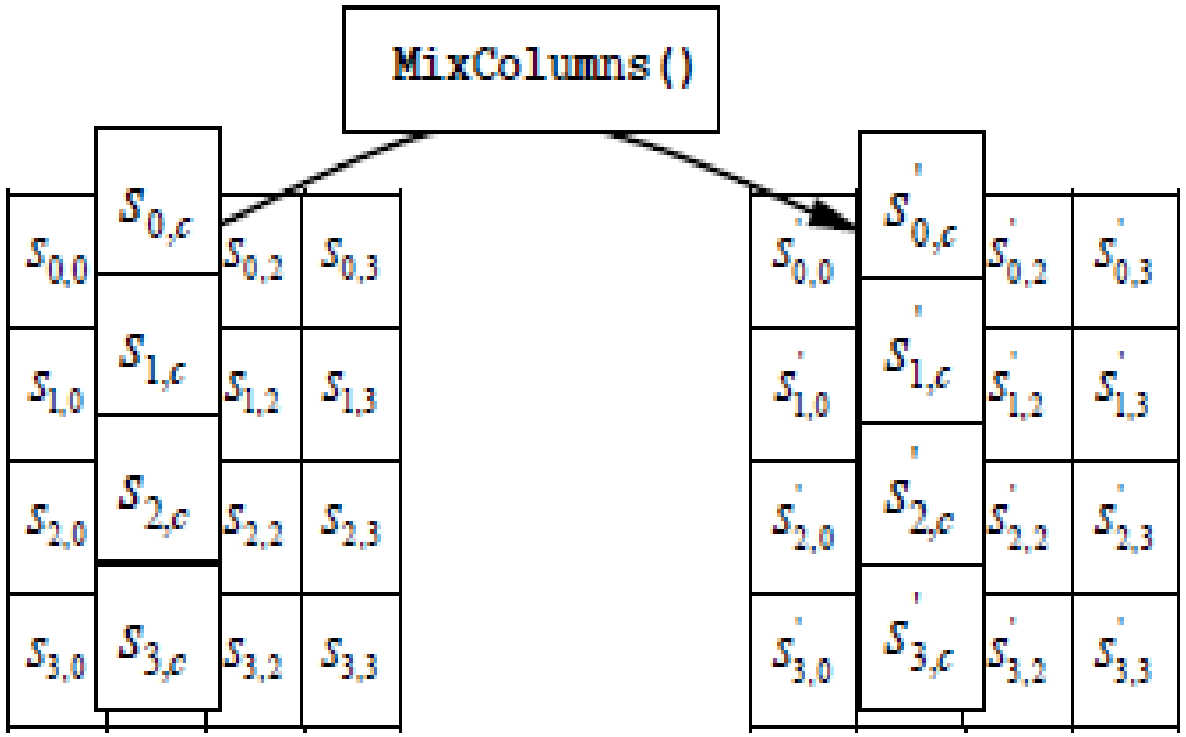
$S_{0,0}$	$S_{0,1}$	$S_{0,2}$	$S_{0,3}$
$S_{1,1}$	$S_{1,2}$	$S_{1,3}$	$S_{1,0}$
$S_{2,2}$	$S_{2,3}$	$S_{2,0}$	$S_{2,1}$
$S_{3,3}$	$S_{3,0}$	$S_{3,1}$	$S_{3,2}$

Shiftrows - AES

1. Hàng đầu không thay đổi.
2. Hàng hai, dịch vòng trái 1-byte.
3. Hàng ba, dịch vòng trái 2 byte.
4. Hàng tư, dịch vòng trái 3-byte.



MixColumns - AES



- Mỗi cột của state là một đa thức trên trường **GF(2^8)** và nhân modulo x^4+1 với đa thức cố định a(x)

$$a(x) = \{03\}x^3 + \{01\}x^2 + \{01\}x + \{02\}$$

MixColumns - AES

$$\begin{bmatrix} s'_{0,c} \\ s'_{1,c} \\ s'_{2,c} \\ s'_{3,c} \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,c} \\ s_{1,c} \\ s_{2,c} \\ s_{3,c} \end{bmatrix} \quad \text{for } 0 \le c < Nb.$$

$s'(x) = a(x) \otimes s(x)$:

$$s'_{0,c} = (\{02\} \bullet s_{0,c}) \oplus (\{03\} \bullet s_{1,c}) \oplus s_{2,c} \oplus s_{3,c}$$

$$s'_{1,c} = s_{0,c} \oplus (\{02\} \bullet s_{1,c}) \oplus (\{03\} \bullet s_{2,c}) \oplus s_{3,c}$$

$$s'_{2,c} = s_{0,c} \oplus s_{1,c} \oplus (\{02\} \bullet s_{2,c}) \oplus (\{03\} \bullet s_{3,c})$$

$$s'_{3,c} = (\{03\} \bullet s_{0,c}) \oplus s_{1,c} \oplus s_{2,c} \oplus (\{02\} \bullet s_{3,c}).$$

MixColumns - AES

- **MixColumns** được định nghĩa bằng phép nhân ma trận sau

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

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

→

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

MixColumns - AES

- MixColumns được định nghĩa bằng phép nhân ma trận sau

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

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

→

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

Kết quả cho cột 1

({02}.{87}) ⊕ ({03}.{6E}) ⊕ {46} ⊕ {A6} = {47}

{87} ⊕ ({02} .{6E}) ⊕ ({03}.{46}) ⊕ {A6} = {37}

{87} ⊕ {6E} ⊕ ({02}.{46}) ⊕ ({03}.{A6}) = {94}

({03} .{87}) ⊕ {6E} ⊕ {46} ⊕ ({02}.{A6}) = {ED}

MixColumns - AES

$$(\{02\}.\{87\}) \oplus (\{03\}.\{6E\}) \oplus \{46\} \oplus \{A6\} = \{47\}$$

- {87} = 1000.0111 ➔ {02} .{87} = 1.0000.1110 > (2^8 = 1.0000.0000)
- cần mod ($x^8 + x^4 + x^3 + x + 1$) tức là ⊕ 1.0001.1011)
- {02} .{87} = (1.0000.1110) ⊕ (1.0001.1011) = (0001 0101)
- {03} .{6E} = {6E} ⊕ ({02} .{6E}) = (0110 1110) ⊕ (1101 1100) = (1011 0010)

- {40}={f2}•{02}⊕ {4c}•{03}⊕ {e7}•{01}⊕ {8c}•{01}

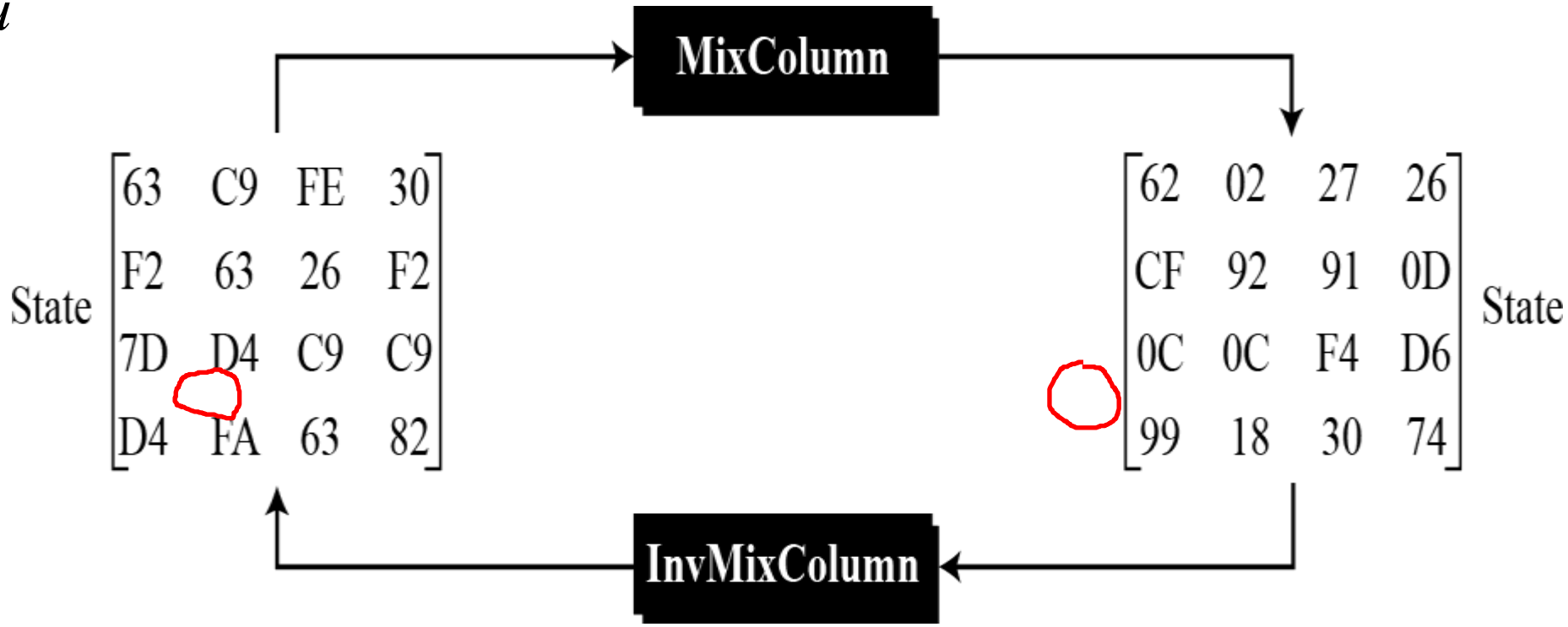
  {f2}•{02} = {1111 0010} •{02}

  ={1110 0100} ⊕ {1b} – do bit 7 của f2=1

  = {1111 1111}

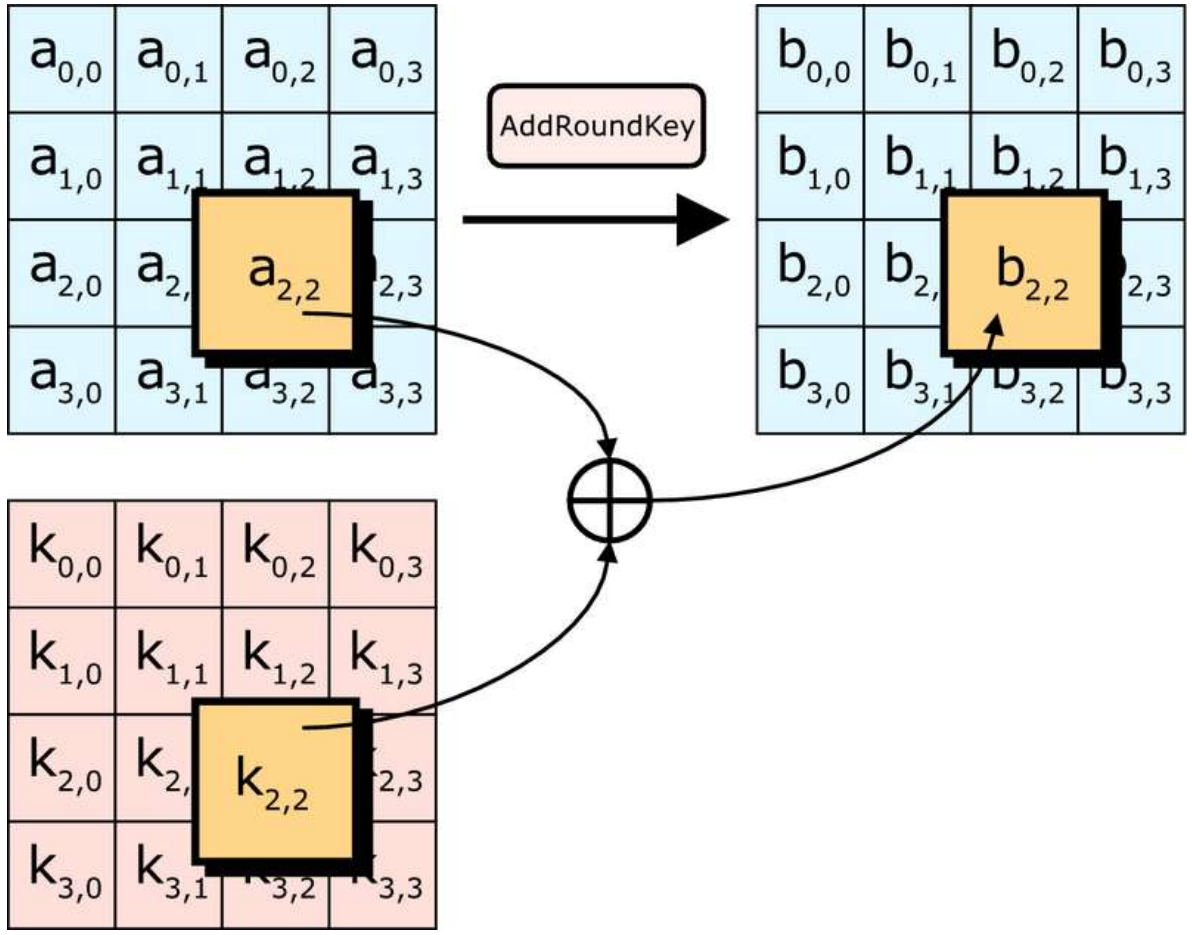
MixColumns - AES

- *Ví dụ*

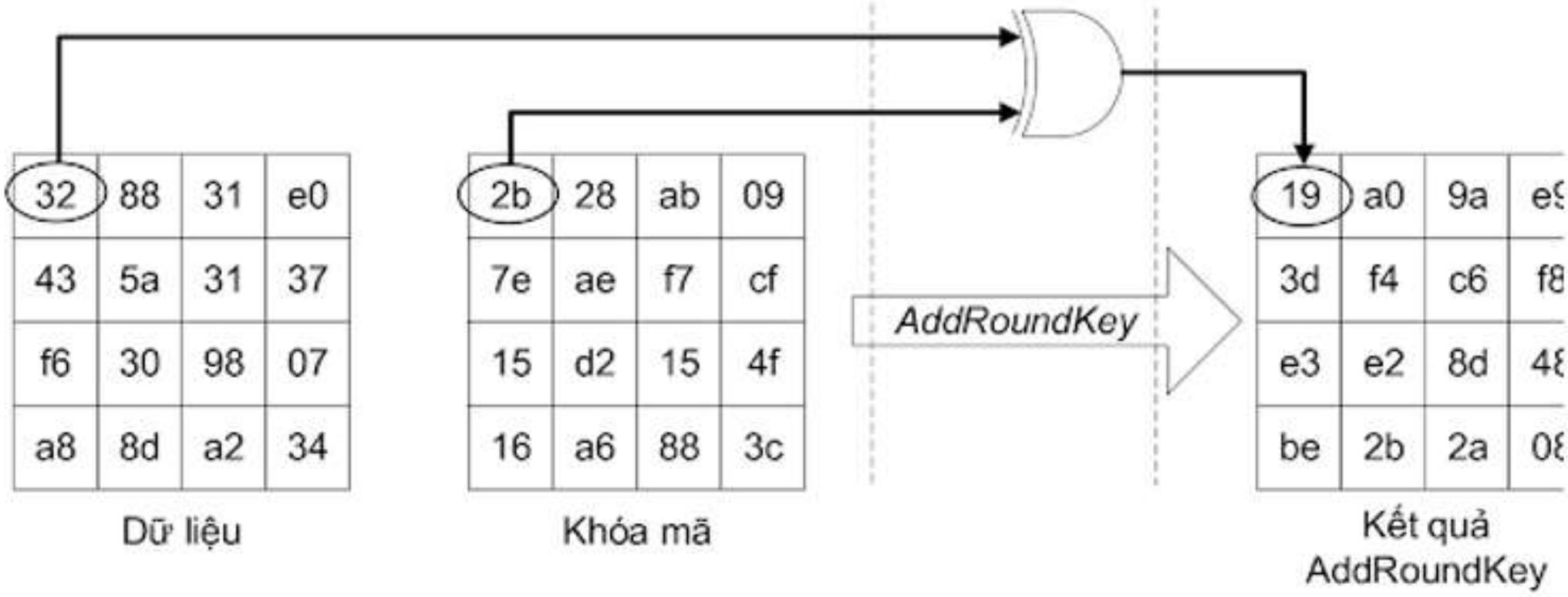


AddRoundKey - AES

Mỗi byte của state được kết hợp tương ứng với 1 byte của khóa vòng bằng sử dụng toán tử XOR.



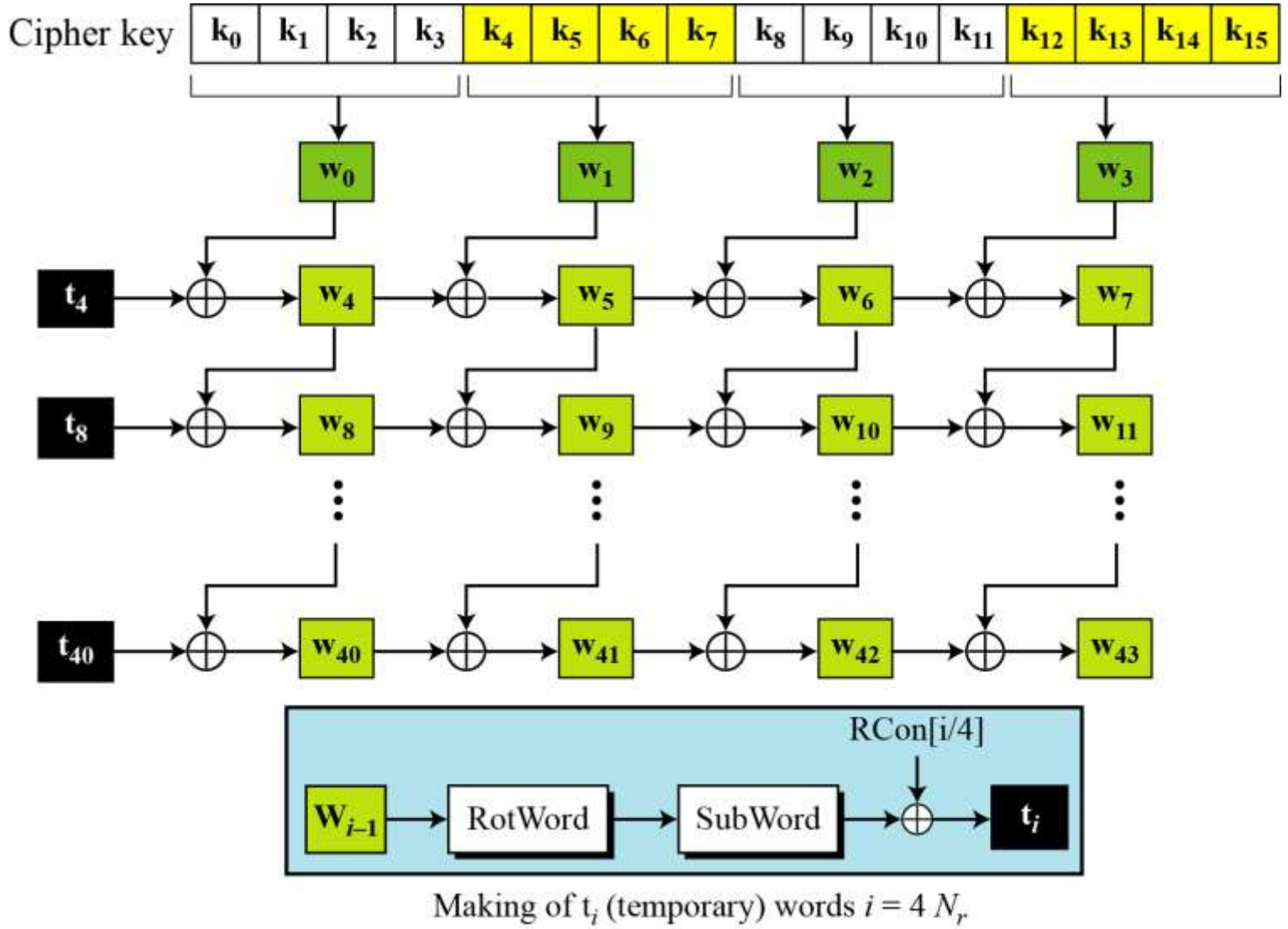
AddRoundKey - AES



Mở rộng khóa (Key Expansion):

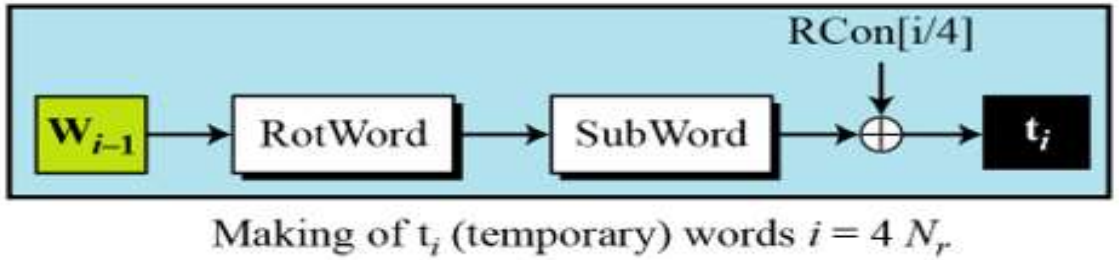
Round	*Words*			
Pre-round	$\mathbf{w}_0$	$\mathbf{w}_1$	$\mathbf{w}_2$	$\mathbf{w}_3$
1	$\mathbf{w}_4$	$\mathbf{w}_5$	$\mathbf{w}_6$	$\mathbf{w}_7$
2	$\mathbf{w}_8$	$\mathbf{w}_9$	$\mathbf{w}_{10}$	$\mathbf{w}_{11}$
. . .	. . .			
N_r	$\mathbf{w}_{4N_r}$	$\mathbf{w}_{4N_r+1}$	$\mathbf{w}_{4N_r+2}$	$\mathbf{w}_{4N_r+3}$

Mở rộng khóa (Key Expansion):



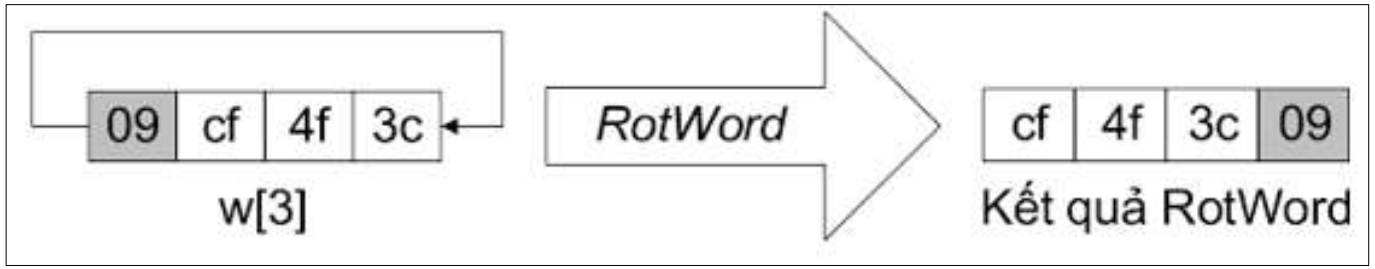
Making of t_i (temporary) words $i = 4\ N_r$

Mở rộng khóa (Key Expansion):

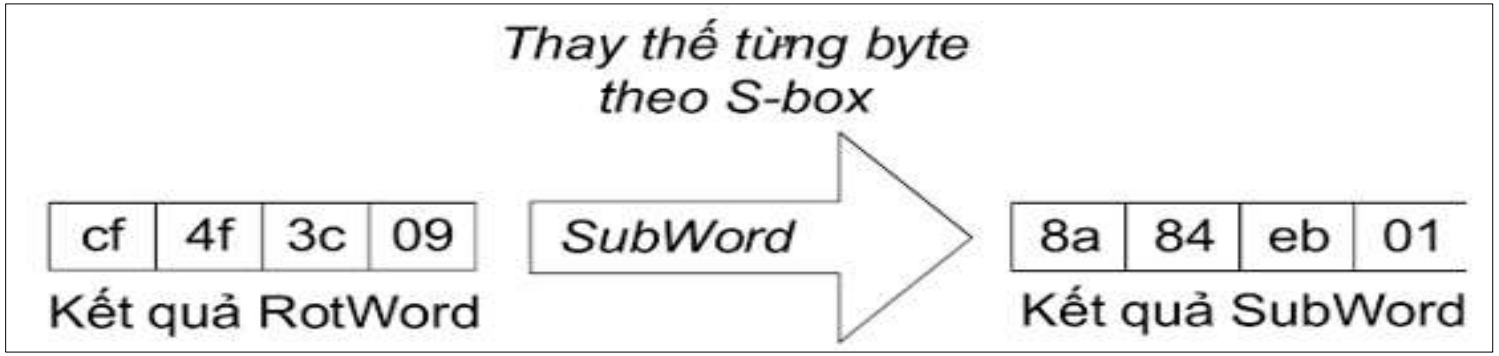


Making of t_i (temporary) words $i = 4\ N_r$

- **Rotword**: dịch vòng trái 1 byte



- **SubWord**: Thực hiện thay thế các phi tuyến từng byte của kết quả RotWord theo bảng S-box.



Ví dụ minh họa

- **Xem paper tham khảo**

Private Key Cryptography

1. Mã hóa César với chìa cố định
2. Mã hóa César với chìa bất định
3. Mã khối (Mã Hill)
4. Mật mã affine

Tính chất:

- Mã đối xứng vì mã hóa và giải mã cùng một khóa
- Khó thay đổi hoặc chia sẻ

Public Key Cryptography (PKC)

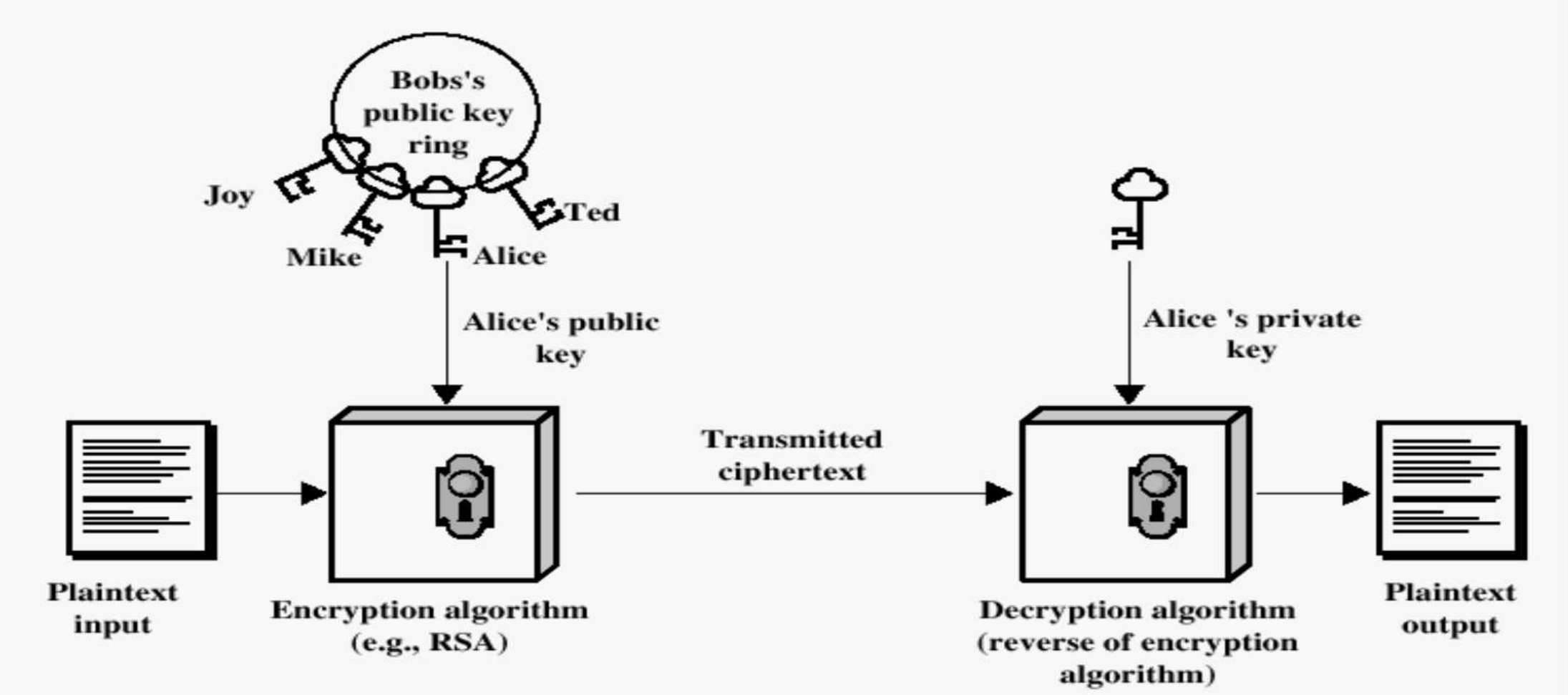
Public Key Cryptography (PKC)

❑ **Đặc trưng**:

- Mã hóa với khóa bất đối xứng (*Asymmetric Key Encryption*) dùng 2 khóa riêng biệt cho mã hóa và giải mã
  - Có một khóa được dùng công khai: *Khóa công khai - K_u (Public Key)*
  - Khóa được giữ riêng cho cá nhân*: Khóa riêng - K_r (Private Key)*
- Việc chọn K_u hay K_r cho quá trình mã hóa tạo ra hai ứng dụng:
  - Nếu dùng K_u để **mã hóa** và K_r để **giải mã** → ***ứng dụng bảo mật*** nội dung thông tin
  - Nếu dùng K_r để mã hóa và K_u để giải mã → ***ứng dụng xác thực*** nội dung và nguồn gốc thông tin
- Mã hóa khóa bất đối xứng còn được gọi là **mã hóa khóa công khai** (Public Key Encryption)

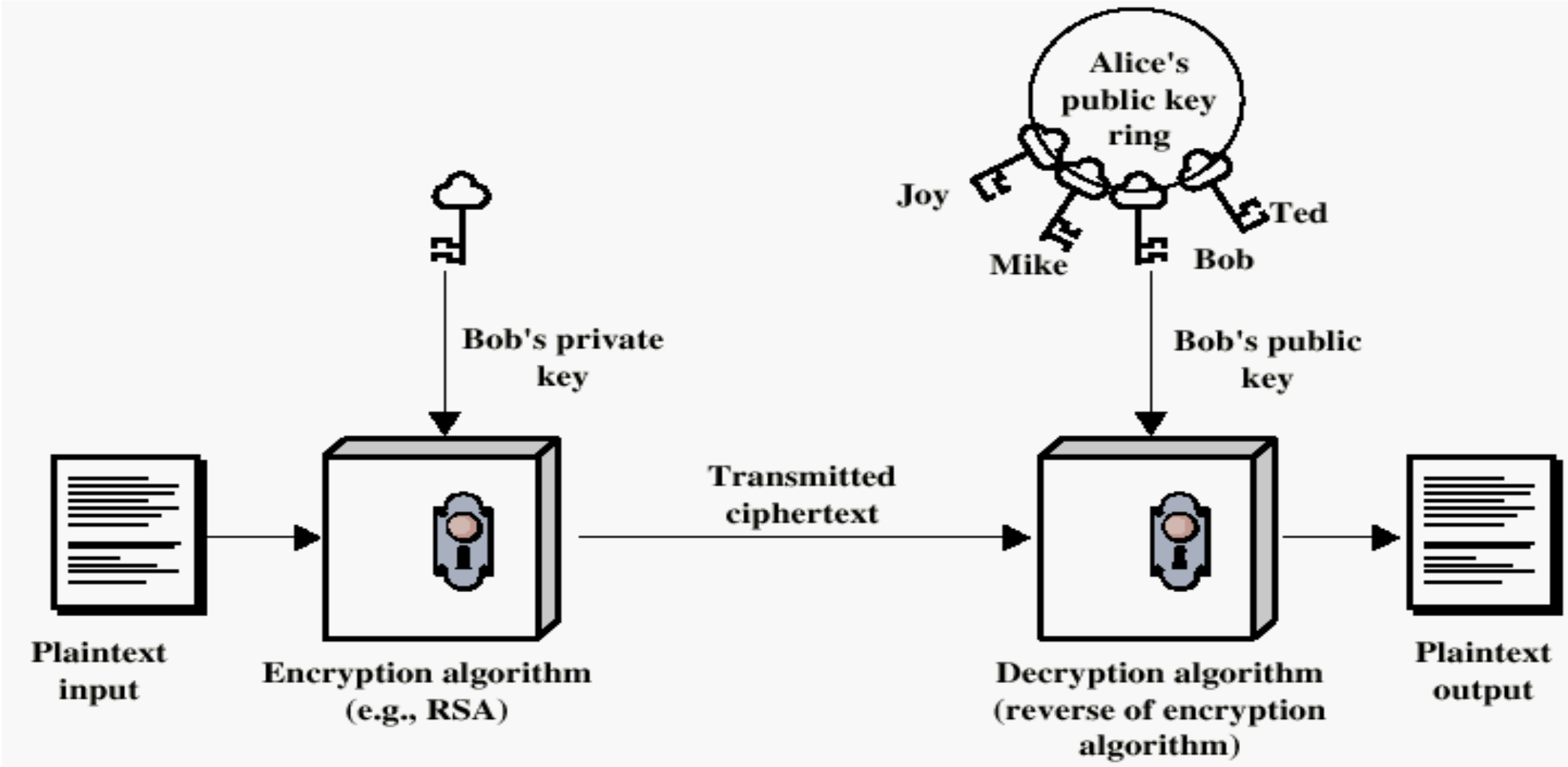
Public Key Cryptography (PKC)

Ứng dụng bảo mật thông tin:



Public Key Cryptography (PKC)

❑ Ứng dụng xác thực thông tin:

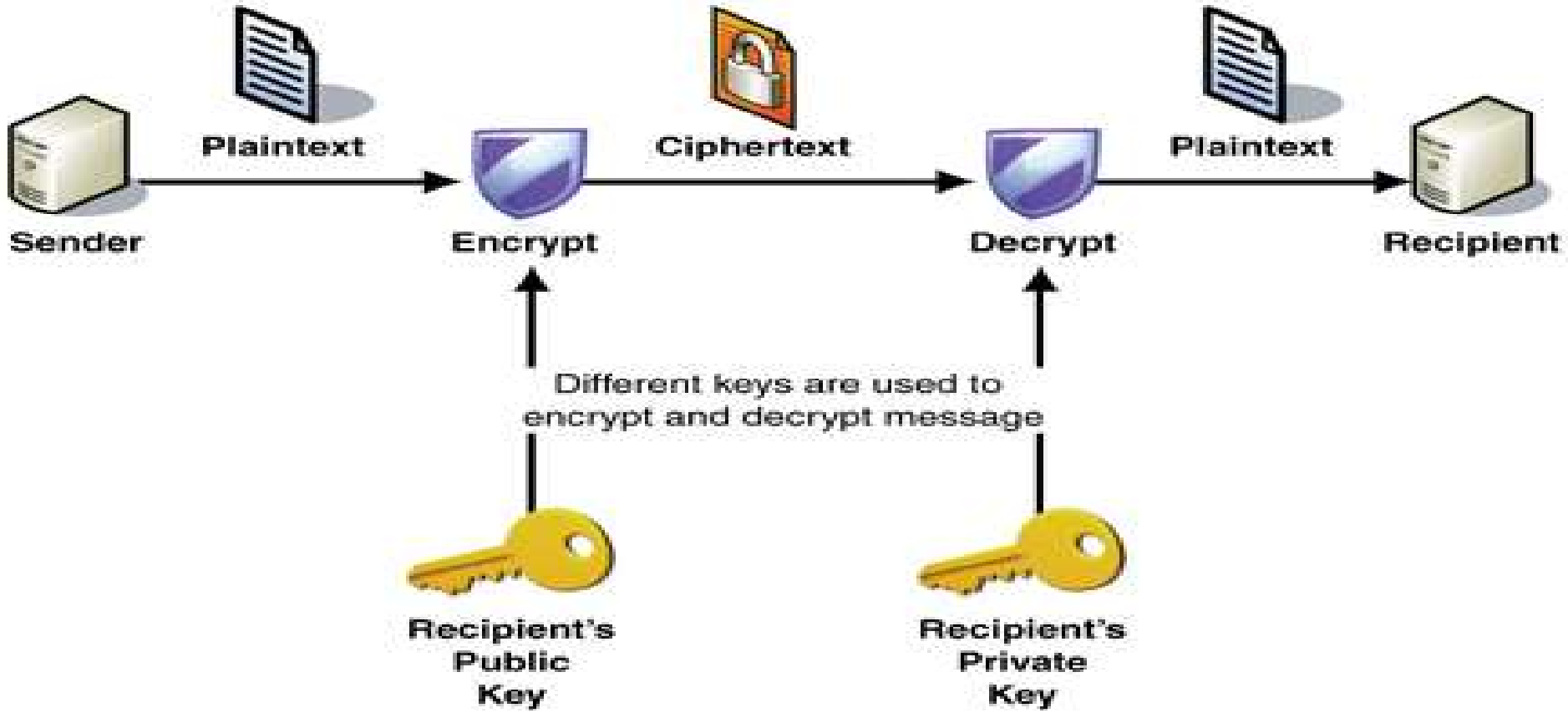


Public Key Cryptography

❑ **Hoạt động**:

- Mỗi người dùng tạo một cặp khóa để mã hóa và giải mã
- Mỗi người dùng đăng ký một trong 2 khóa làm khóa công khai sao cho mọi người đều có thể truy cập được. Khóa còn lại được giữ bí mật.
- Ví dụ:
  - Nếu **Bob** muốn gởi một thông điệp mật đến **Alice**, anh ta **mã hóa thông điệp** bằng **khóa công khai của Alice.**
  - Khi Alice nhận thông điệp, cô ta **giải mã thông điệp** bằng **khóa bí mật của mình**. Không ai ngoài Alice có khả năng giải mã vì chỉ Alice có khóa bí mật của mình

Public Key Cryptography



Public Key Cryptography

❑ **Yêu cầu đối với thuật toán PKC**:

- **Dễ dàng tính** được cặp khóa công khai K_u và bí mật K_r
- **Dễ dàng mã hóa** thông tin gốc **P** bằng một trong hai khóa thành thông tin mã hóa **C:**

  $C = E(P, K_u)$ hay $C = E(P, K_r)$
- **Dễ dàng giải mã** thông tin mã hóa C bằng K_u hay K_r ra **P**:

  $P=D(C, K_r) = D(E(P, K_u), K_r)$ hay

  $P=D(C, K_u) = D(E(P, K_r), K_u)$ hay
- **Không thể** tính được K_r từ K_u trong khoảng thời gian cho phép.
- **Không thể** tính được bảng rõ **P** từ khóa K_u và bản mã cho trước
- Mật **mã hóa** và **giải mã** được thực hiện theo một trong hai quá trình:

  $P=D(E(P, K_u), K_r)$

  $P=D(E(P, K_r), K_u)$

Public Key Cryptography

- **Giải thuật khóa công khai gồm 6 thành phần:**
  - Bản rõ: thông điệp có thể đọc, đầu vào của giải thuật
  - Giải thuật mật hóa
  - Khóa công khai **và** bí mật: một cặp khóa được chọn sao cho 1 khóa dùng để mật hóa và 1 khóa dùng để giải mật.
  - Bản mật: thông điệp đầu ra ở dạng không đọc được, phụ thuộc vào bản rõ và khóa. Nghĩa là với cùng một thông điệp, 2 khóa khác nhau sinh ra 2 bảng mã khác nhau
  - Giải thuật giải mật

Public Key Cryptography

- **Giải thuật khóa công khai gồm 6 thành phần:**
  - Bản rõ: thông điệp có thể đọc, đầu vào của giải thuật
  - Giải thuật mật hóa
  - Khóa công khai **và** bí mật: một cặp khóa được chọn sao cho 1 khóa dùng để mật hóa và 1 khóa dùng để giải mật.
  - Bản mật: thông điệp đầu ra ở dạng không đọc được, phụ thuộc vào bản rõ và khóa. Nghĩa là với cùng một thông điệp, 2 khóa khác nhau sinh ra 2 bảng mã khác nhau
  - Giải thuật giải mật

Mật mã Diffie-Hellman (1976)

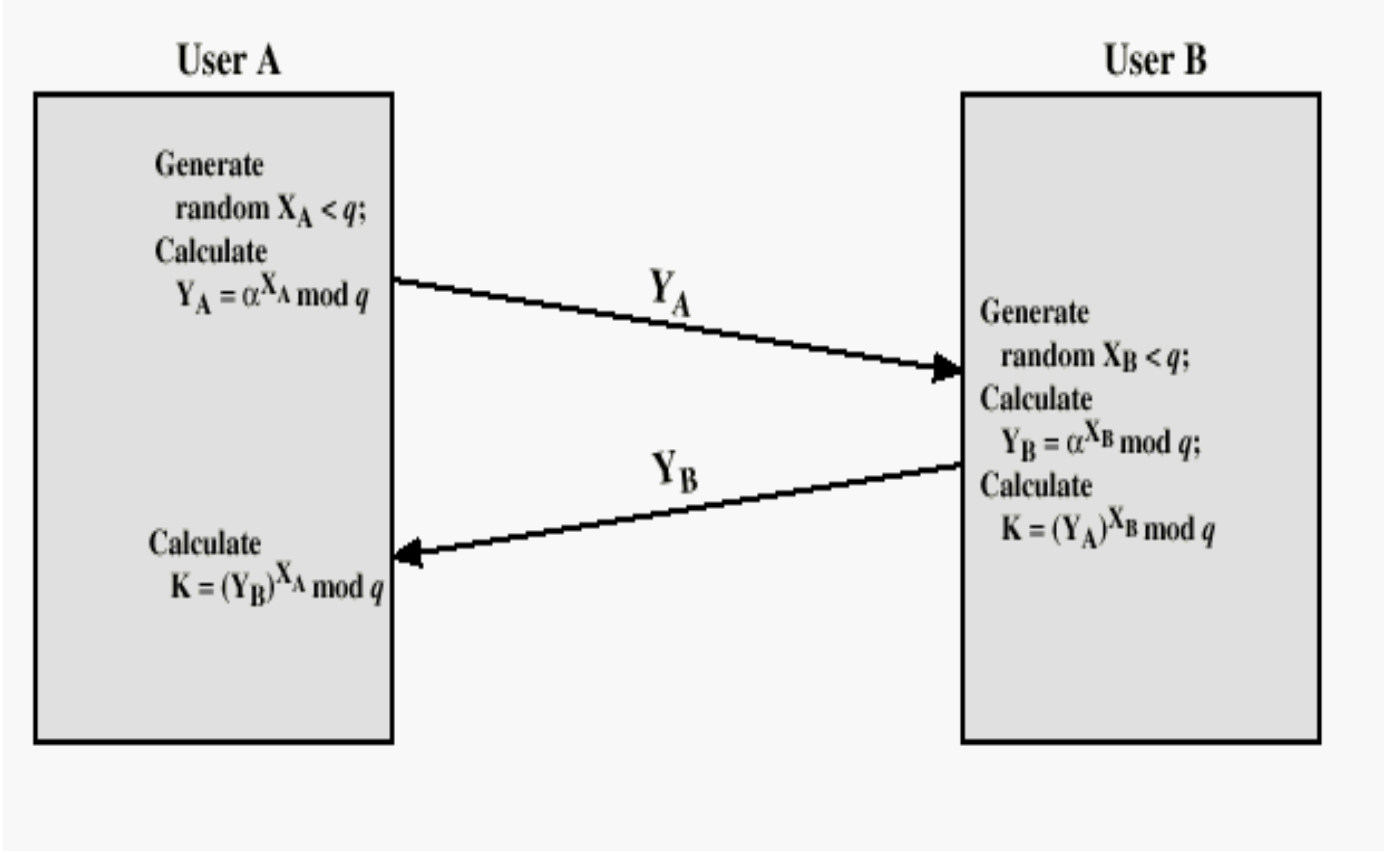
Whitfield Diffie (1944)  **Martin E. Hellman** (1954)

Mật mã Diffie-Hellman: Khái niệm

- Phương pháp khóa công khai (PKC).
- Chức năng: Trao đổi khóa mật mà không truyền trên kênh.
- Mô hình tiêu biểu: Trao đổi khóa (Key Exchange)

Giải thuật Diffie-Hellman

- α và **q** là tham số chọn trước
- User A có: X_A là khóa cá nhân và Y_A là khóa công khai.
- User B có: X_B là khóa cá nhân và Y_B là khóa công khai.
- Tìm X_A hay X_B là tính Logarithm rời rạc.
- Thời gian tính Logarithm rời rạc là lũy thừa.



Mật mã Diffie-Hellman : Ví dụ

Alice...

Chọn số ngẫu nhiên đủ lớn X_A.

Gửi cho Bob: q, α và Y_A

($Y_A = \alpha^{X_A} \bmod q$).

Nhận: Y_B

Tính: $K = Y_B^{X_A} \bmod q$

Ví dụ:

- Chọn: q=7, α =3 gửi Bob.
- Chọn: $X_A = 2$
- Gửi Bob: $Y_A = \alpha^{X_A} \bmod q = 2$
- Nhận: Y_B=6
- Tính: $K = 6^2 \bmod 7 = 1$

Bob...

Chọn số ngẫu nhiên đủ lớn X_B.

Gửi cho Alice:

($Y_B = \alpha^{X_B} \bmod q$).

Nhận Y_A

Tính: $K = Y_A^{X_B} \bmod q$

Ví dụ:

- Nhận: q=7, α =3 .
- Chọn: $X_B = 3$
- Gửi Alice: $Y_B = 3^3 \bmod 7 = 6$
- Nhận: $Y_A = 2$
- Tính $K = 2^3 \bmod 7 = 1$

Ron Rivest - Adi Shamir-Leonard Adleman(1978)

Ron Rivest – Adi Shamir – Leanard Adleman

Thuật toán RSA

- Phương pháp khóa công khai (**PKC**).
- **Chức năng chính**: Trao đổi khóa, Chữ ký điện tử, Tạo mật mã cho khối dữ liệu nhỏ.
- **Chức năng phổ biến**: Tạo mật mã cho khóa phiên giao dịch trong mật mã lai.
- **Nguyên lý tạo mật mã** của **P**:
  - **Tạo bản mật** với khóa công khai (**n,e**): $\mathbf{C \equiv P^e \pmod n}$
  - **Giải mật mã** với khóa cá nhân d: $\mathbf{P \equiv C^d \pmod n}$

Thuật toán RSA

Bob muốn mã hóa thông điệp số **P** $\rightarrow$ **C** để gửi cho **Alice**

1. **<u>Sinh khóa</u>:**

- **Alice:**

+ chọn **p** và **q**: hai số nguyên tố khác nhau (*2 số rất lớn*)

+ tính: **n = p.q**

+ tính phi –Euler **φ(n) = (p-1)(q-1)**.

(phi-Euler: số lượng các số nguyên tố cùng nhau với n).

+ Chọn 1< **e** < φ(n), sao cho: **gcd(φ(n),e) = 1**

+ Tính **d**: d.e = 1 (mod φ(n)) (hay: **$d = e^{-1} \bmod \varphi(n)$**)

K_u =(e,n) là *khóa công khai* dùng để mã hóa (gửi cho mọi người)

K_r=(d,n) là *khóa riêng* dùng để giải mã

Thuật toán RSA

2. <u>Phân phối khóa</u>:

Alice gửi khóa công khai **(e,n)** của mình cho **Bob**

3. <u>Mã hóa</u>:

Sau khi **Bob** nhận được khóa công khai **(e,n)** của **Alice, Bob** có thể mã hóa thông điệp **P** thành **C** và gửi cho **Alice**

Công thức: $C \equiv P^e \pmod n$

4. <u>Giải mã</u>:

Sau khi nhận được **C** của **Alice** có thể giải mã **C** bằng cách sử dụng khóa riêng **(d,n)** của mình để tính được **P**

Công thức: $P \equiv C^d \pmod n$

Hay $C^d \equiv (P^e)^d \pmod n = P^{e\,d} \pmod n = P^1 \pmod n = P$ $(P,n)=1$

Ví dụ 1 - Ứng dụng mã hóa bảo mật

Bob muốn gửi cho *Alice* 1 thông điệp bí mật *C* được mã hóa từ thông điệp gốc *P=65.*

1- <u>Sinh khóa</u>:

- **Alice:**

  + chọn **p=61** và **q=53 => n = p.q= 3233**

  + φ(3233) = 60*52 = 3120

  + Chọn **e** =17, sao cho: **gcd(φ(n),e) = 1**

  + Tính **d**: **d.17 = 1 (mod 3120)** =2753

  (17,3233) là *khóa công khai* của Alice

  (2753,3233) là *khóa riêng* của Alice

2- <u>Phân phối Khóa:</u>

Alice gửi cho Bob (17,3233)

Ví dụ 1 - tt

Bob muốn gửi cho *Alice* 1 thông điệp bí mật *C* được mã hóa từ thông điệp gốc *p=65.*

3- <u>Mã hóa:</u>

- Bob dựa vào **(17,3233) mã hóa** P=65 $\rightarrow$ C = 2790

- Và gửi cho **Alice** ($\mathbf{C \equiv P^e \pmod n} \equiv 65^{17} \pmod{3233} = 2790$)

4- <u>Giải mã</u>:

Alice dựa vào khóa riêng **(2753,3233) tìm P**

$$\mathbf{P \equiv C^d \pmod n} \equiv \mathbf{2790}^{2753} \pmod{3233} = 65$$

Tìm số nghịch đảo d

- ***Euclide mở rộng***
- *Đồng dư thức (tài liệu P109)*

Ví dụ 2:

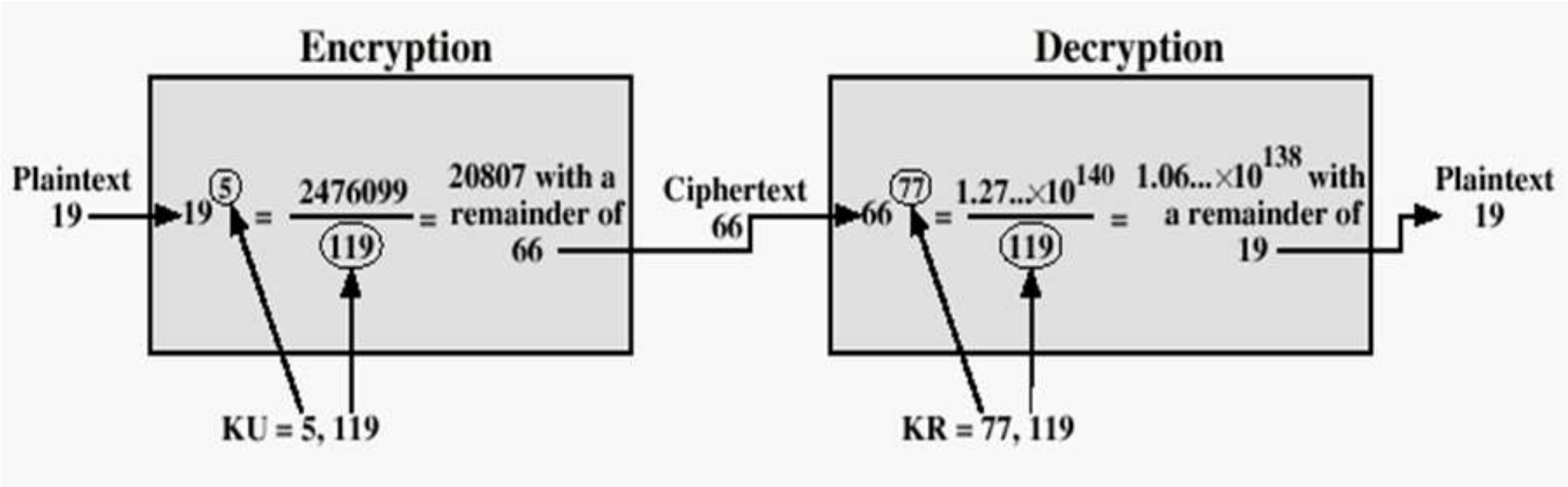
Các thông số: p = 7, q = 17 **Plaintext P=19;**

=> n = p.q=119, $\varphi(n)$ = 96 ;

e = 5, d = 77(e.d ≡ 1 mod 96)

- **Mã hóa**: P=19 => C≡ 19^5 (mod 119) ≡ 66

- **Giải mã**: C= 66 =>P ≡66^{77} (mod 119) ≡ 19



Ví dụ khác (SV tự kiểm tra)

Giả sử:

- Khóa công khai (n,e) = (15,11) và khóa cá nhân d = 3
- M = "**SECRET" = 0x83 69 67 82 69 84**.

Quá trình tạo mật mã và giải mật mã:

- Tạo mật mã RSA cho từng ký tự với khóa công khai: $C_i = M_i^{11} \bmod 15$.
- Kết quả tạo mật mã : C = **0x2c696d286924**.
- Giải mật mã từng ký tự C_i: $M_i = C_i^3 \bmod 15$.
- Kết quả giải mật mã: C -> M = **0x836967826984** = "**SECRET"**.

Chú ý:

- Do n không đủ lớn nên có một số ký tự không thay đổi,
- Chẳng hạn: ký tự E với mã ASCII là 69.

Mật mã khác

- Mật mã đường Ellip (Elliptic-Curve Cryptography - ECC)
  - Ưu điểm: sử dụng độ dài khóa nhỏ vf cps tốc độ tính toán nhanh.
  - Nhược điểm: Độ tin cậy thấp hơn so với RSA.