TRƯỜNG ĐẠI HỌC NGUYỄN TẤT THÀNH
KHOA CÔNG NGHỆ THÔNG TIN

Bài giảng môn học: AN TOÀN THÔNG TIN

Chương 4:

HỆ MẬT MÃ KHÓA ĐỐI XỨNG HIỆN ĐẠI

Số tín chỉ: 3
Số tiết: 60 tiết
(30 LT + 30 TH)

Biên soạn: ThS. **Nguyễn Thị Phong Dung**
Email : ntpdung@ntt.edu.vn

Bài 4: MẬT MÃ KHÓA ĐỐI XỨNG HIỆN ĐẠI

- Hệ mật mã hiện đại
- Mật mã dòng (Stream Cipher)
- Mật mã khối (Block Cipher)
- Hệ mật mã Feistel
- Data Encryption Standard (DES)
- Chuẩn mật mã 3-DES và EAS

HỆ MẬT MÃ HIỆN ĐẠI

- **Các đặc tính của Hệ mật mã hiện đại:**
  - Mã hóa thông tin có dung lượng lớn.
  - Áp dụng các thuật toán hiện đại, phức tạp.
  - Không gian khóa lớn => gây khó khăn trong việc phá mã bằng kỹ thuật vét cạn (*brute force*).
  - Dùng **máy tính** để mã hóa và giải mã:
    - Dữ liệu được **số hóa** thành ***bits*** (*dữ liệu số*).
    - Mã hóa và Giải mã trên dữ liệu số.
    - Chuyển dữ liệu số thành thông tin.
  - Thuật toán chạy bằng máy tính => tốc độ nhanh.

HỆ MẬT MÃ HIỆN ĐẠI

- **Các mô hình mã hóa dữ liệu:**
  - **Mã hóa dòng** (*Stream Cipher*): từng bit (hay byte) dữ liệu đầu vào được xử lý mã hóa cho đến khi kết thúc luồng dữ liệu.
  - **Mã hóa khối** (*Block Cipher*): dữ liệu được chia thành từng khối để mã hóa (hoặc giải mã).
- **Các hình thức sử dụng khóa mã:**
  - **Khóa đối xứng** (*Symmetric key*): dùng khóa giống nhau cho tác vụ Mã hóa (*Encrypt*) và Giải mã (*Decrypt*).
  - **Khóa bất đối xứng** (*Asymmetric key*): Mã hóa bằng một khóa, giải mã bằng một khóa khác.

HỆ MÃ HÓA DÒNG (Stream Cipher)

- ***Stream Cipher* theo bit:**
  - Giả định:
    - Gọi **P** (*Plain text*) là tập **bản rõ** gồm các *bits* $\mathbf{x_i}$ (i = 1, 2, 3…).
    - Gọi **K** (*Key*) là tập **khoá** bao gồm các *bits* $\mathbf{z_i}$ (i = 1, 2, 3 …).
    - Gọi **C** (*Cipher text*) là tập **bản mã** gồm các *bits* $\mathbf{y_i}$ (i = 1, 2, 3 …).
  - Mã hóa và giải mã:
    - Mã hóa (*Encryption*) từng ***bit*** đầu vào của dòng dữ liệu:

$$y_i = e_{z_i}(x_i) = x_i + z_i \bmod 2$$

    - Giải mã (*Decryption*) từng ***bit*** của dòng dữ liệu

$$x_i = e_{z_i}(y_i) = y_i + z_i \bmod 2$$

    - Ghi chú: ***mod*** là phép *modulo* (lấy **số dư**) của phép chia.

HỆ MÃ HÓA DÒNG (Stream Cipher)

- **Tinh gọn:**
  - Biểu thức: *(a + b) mod 2* cũng chính là phép ***XOR*** của *a* và b:

a	b	c = a + b	c mod 2	a XOR b
0	0	0	**0**	**0**
0	1	1	**1**	**1**
1	0	1	**1**	**1**
1	1	10	**0**	**0**

  - VD: mã hóa và giải mã ký tự '**A**' (**0100 0001**) với Key = **0010 1100**

Encryption		ASCII
P $\{x_1 .. x_7\}$	0100 0001	A
K $\{x_1 .. x_7\}$	0010 1100	
C $\{x_1 .. x_7\}$	0110 1101	m

Decryption		ASCII
C $\{x_1 .. x_7\}$	0110 1101	m
K $\{x_1 .. x_7\}$	0010 1100	
P' $\{x_1 .. x_7\}$	0100 0001	A

MẬT MÃ KHỐI (Block Cipher)

- **Dẫn nhập:**
  - Các yếu điểm của *Stream cipher*:
    - Tốc độ chậm.
    - Không thích hợp mã hóa dữ liệu lớn.
    - Dễ bị phá mã.
  - Mã hóa khối (*Block cipher*):
    - Dữ liệu chia thành các khối với kích thước (*block size*) bằng nhau.
    - Dựa theo khóa **K**, mỗi *block* $\mathbf{P_i}$ sẽ được mã hóa thành $\mathbf{C_i}$.
    - Nhiều *block* có thể được mã hóa cùng lúc.
    - Độ an toàn cao nếu *Block size* và *Key space* đủ lớn.

MẬT MÃ KHỐI (Block Cipher)

- **Minh họa *Block Cipher* cơ bản:**
  - Định nghĩa trước bảng ma trận mật mã, có: *block size* = ***3*** *bit* theo ***5*** *key* khác nhau (*key space* = ***3*** *bit*).

block / key	000	001	010	011	100	101	110	111
0	001	111	110	000	100	010	101	011
1	001	110	111	100	011	010	000	101
2	001	000	100	101	110	111	010	011
3	100	101	110	111	000	001	010	011
4	101	110	100	010	011	001	000	111

  - Theo bảng này, *plaintext* **010100110111** sau khi chia thành các *block* = *3 bits* sẽ được mã hóa thành:
    - Dùng ***key=1***: 010 100 110 111 → **111 011 000 101**
    - Dùng ***key=4***: 010 100 110 111 → **100 011 000 111**

MẬT MÃ KHỐI (Block Cipher)

- **Nhận xét Block Cipher cơ bản:**
  - Tính an toàn:
    - Nếu có quá *ít khóa* => dễ dàng phá mã bằng phương pháp *Brute Force*.
    - *Block size* quá nhỏ => dễ đoán plaintext bằng phương pháp thống kê. (ví dụ: nếu có khối $\boldsymbol{C}$ = **001** sẽ dễ dàng đoán **P** sẽ là **000** hoặc **101**)
  - Độ lớn của bảng ma trận mật mã:
    - Nếu *Block size* dùng **n** bits => ma trận sẽ có $\mathbf{2^n}$ cột.
    - Nếu *Key size* dùng **m** bits => ma trận sẽ có $\mathbf{2^m}$ dòng.
    - Nếu *Block size* = **32** bits và *Key size* = **32** bits thì: => bảng ma trận sẽ có $\mathbf{2^{32}}$ x $\mathbf{2^{32}}$ phần tử, mỗi phần tử là *32 bits*.
    - QUÁ LỚN !!!

MẬT MÃ KHỐI (Block Cipher)

- **Giảm độ lớn của ma trận mật mã:**
  - Nguyên lý:
    - Dùng *Key size* lớn => không gian khóa lớn.
    - Từ không gian khóa, chọn ra các keys sẽ dùng => số lượng dòng ít.
    - Lựa chọn dòng Key nào sử dụng bằng thuật toán sinh khóa.
  - Thuật toán sinh khóa:
    - Từ khóa **K** ban đầu => tạo ra **n** khóa $\mathbf{K_i}$ con (*Sub-key*).
    - Nếu khóa **K** ban đầu giống nhau => các khóa $\mathbf{K_i}$ con (sinh ra bởi thuật toán) phải có **giá trị** giống nhau (không thay đổi).
  - Minh họa: sinh khóa bằng "*Rotate Right*" (**step =1**)
    - Khóa ban đầu **K** = **1001 0110**
    - Khóa con $\mathbf{K_1}$ = **0100 1011**
    - Khóa con $\mathbf{K_2}$ = **1010 0101**
    - Khóa con $\mathbf{K_3}$ = **1101 0010**

MẬT MÃ KHỐI (Block Cipher)

- **Nguyên lý tăng tính an toàn cho mật mã khối:**
  - Tạo tính hỗn loạn (*Confusion*) cho dữ liệu (**P** hoặc **C**) nhằm gây khó trong việc dò tìm quy luật để phá mã.
  - Tạo tính khuếch tán / rườm rà (*Diffusion*): đưa dữ liệu gây nhiễu vào **P** hay **C**, gây khó khăn cho phương pháp thống kê.
  - Thuật toán phần mềm cần đảm bảo tính mềm dẻo và giá thấp.
  - Về phần cứng: thuật toán phải đảm bảo tốc độ cao và kinh tế .

MẬT MÃ KHỐI (Block cipher)

- **Nguyên lý tăng tính an toàn cho mật mã khối:**
  - *Minh họa 1*: tạo tính hỗn loạn cho **P** = "**computer security**" bằng phép thay thế (*substitution*):
    - Viết các ký tự theo ma trận 5 cột.
    - Tạo lại chuỗi hỗn loạn bằng cách viết lại theo thứ tự cột.
    - Chuỗi kết quả: **C T C Y O E U M R R P S I U E T**

c	**o**	**m**	**p**	**u**
t	**e**	**r**	**s**	**e**
c	**u**	**r**	**i**	**t**
y				

  - *Minh họa 2*: gây nhiễu bằng phương pháp mã hóa nhiều lần, mỗi lần mã hóa bằng key khác nhau.
    - *Plaintext* **010100110111** mã theo ***key=1*** thành:**111 011 000 101**
    - Lấy kết quả, tiếp tục mã theo ***key=4*** thành: **111 110 110 000**

MÔ HÌNH MẬT MÃ FEISTEL

- **Nguyên lý hệ mật mã Feistel:**
  - Kết hợp giữa **thay thế** (*Substitution*) và **chuyển vị** (*Permutation*).
  - Dùng **thay thế** liên tục để tạo tính khuếch tán (*Diffusion*):
    - *Plaintext* (P) sẽ biến đổi qua ***n*** vòng để được *Cipher text*. Mỗi vòng sử dụng khóa K_i khác nhau.

$$P \xrightarrow{K_1} C_1 \xrightarrow{K_2} C_2 \xrightarrow{K_3} \dots \xrightarrow{K_{n-1}} C_n$$

    - K_i là khóa của vòng ***i***, là khóa con, sinh ra từ khóa **K** ban đầu theo một *thuật toán sinh khóa* nào đó.
  - Dùng **chuyển vị** liên tục để tạo tính hỗn loạn (*Confusion*):
    - Khối *plaintext* **P** được chia thành L_0 và R_0 (*Initial Permutation* - IP)
    - Sau mỗi vòng mã hóa, 2 phần L_i và R_i của *Cipher* C_i sẽ đảo chỗ cho nhau.

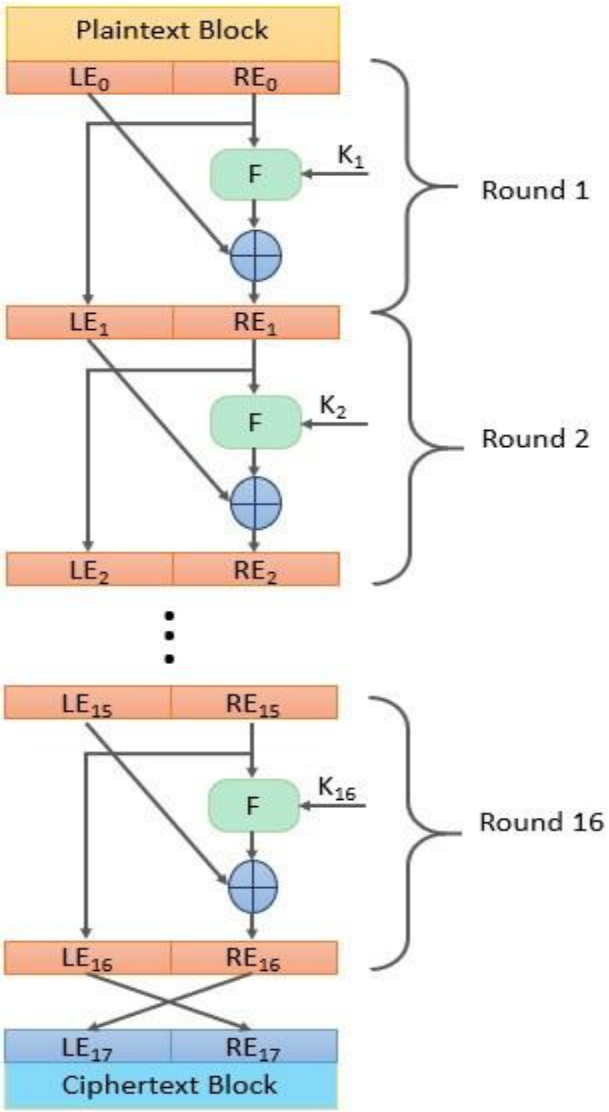
Mô hình mã Feistel

- **Quy trình mã hóa:**
  - Tại mỗi vòng lặp, thực hiện:

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

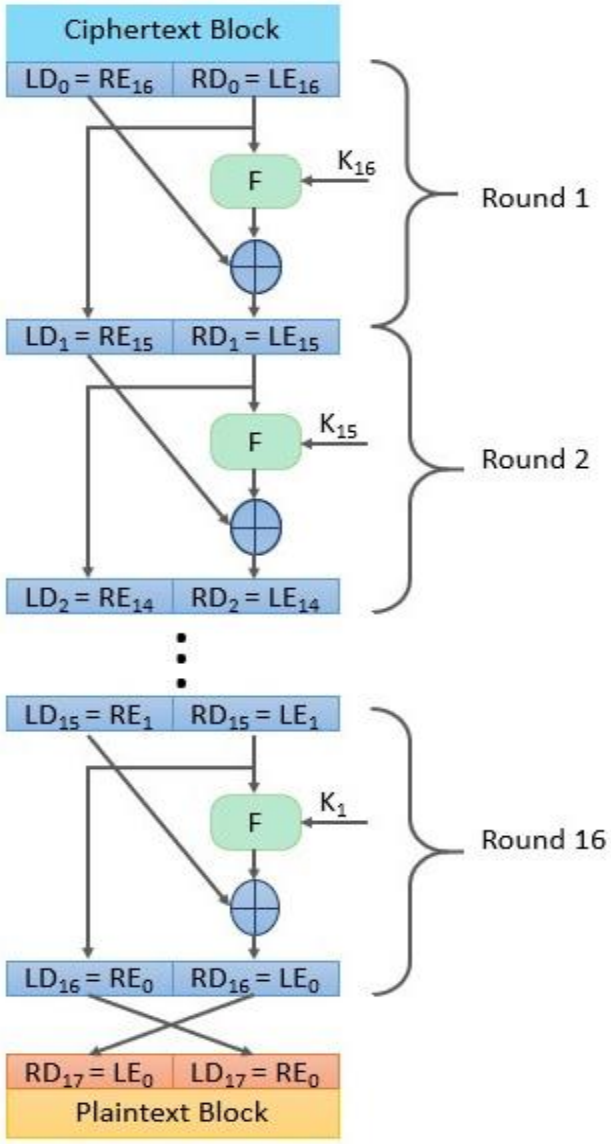
$$R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$$

  - Diễn giải:
    - Phần dữ liệu **L** hiện tại (L_i) lấy từ **R** kết quả trước đó (R_{i-1})
    - **F** là hàm mã hóa dữ liệu R_{i-1} theo khóa K_i.
    - R_i là kết quả **XOR** giữa L_{i-1} với kết quả của hàm **F**.
  - Thực hiện **n** lần.
  - Kết quả: **C** = (L_n , R_n)



Mô hình mã Feistel

- **Quy trình giải mã:**
  - Từ **C** = (**L_n** , **R_n**), thực hiện lại qui trình mã hóa.
    - *Sub-key*: dùng từ **K_{16}** đến **K_1**.
    - Cùng hàm **F**, cùng số vòng lặp.
  - Kết quả nhận được **P** = (**L_0** , **R_0**).
- **Nhận xét về mã Feistel:**
  - Tăng *Block size*/ *Key size*/ *số vòng* => tăng an toàn / làm giảm tốc độ.
  - Hàm **F** / thuật toán sinh khóa khác nhau => mã khác nhau.
  - Mã hóa và giải mã đều dùng cùng hàm **F**



Data Encryption Standard (DES)

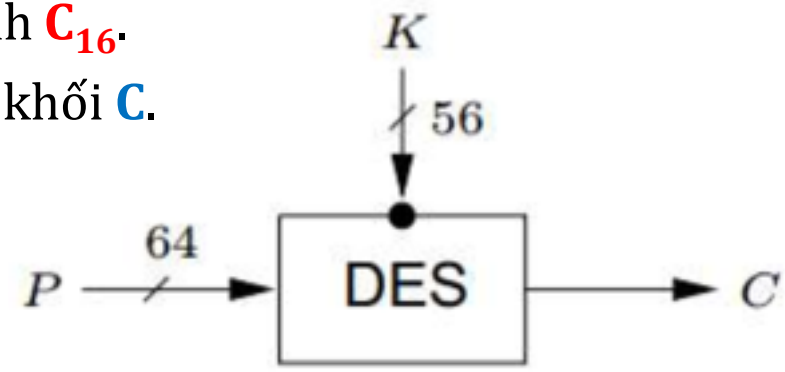
- **Tổng quan hệ mật mã DES:**
  - Thiết kế bởi IBM (1975), chuẩn NIST (1977)
  - **DES** là dạng mã hóa **Khóa đối xứng** (*Symmetric key*).
  - Dùng phương pháp mã khối dựa theo hệ mã ***Feistel***.
    - *Block size* = **64** bit (*L-Block* = *R-Block* = **32** bit).
    - *Key size* = **56** bit (*Key* dùng **48** bit, 8 bit dùng kiểm tra)
    - Số vòng lặp: **16** vòng.
  - Mã DES hiện nay không còn được coi là an toàn:
    - 1998: "*DES Cracker*" phá mã DES trong **56 giờ**
    - 2006: COPACOBANA phá mã DES trong **9 ngày**
    - 2006: Hệ thống khoảng 10.000 PC phá mã DES trong **1 đêm**

Data Encryption Standard (DES)

- **Qui trình mã hóa khối 64 bits của DES:**
  - *Bước 1*: hoán vị khởi tạo (*Initial Permutation* - **IP**):
    - Hoán vị **64** bit **P** (64 bit = ma trận 8 x 8).
    - Chia khối 64 bit P thành 2 phần L_0 và R_0.
  - *Bước 2*: thực hiện qui trình mật mã ***Feistel*** với *khóa* **48** bit.
    - Chạy thuật toán sinh khóa con (*sub-key* = **48** bit).
    - Thực hiện vòng lặp *Feistel* với hàm **F** riêng.
    - Số vòng lặp là: **16** vòng

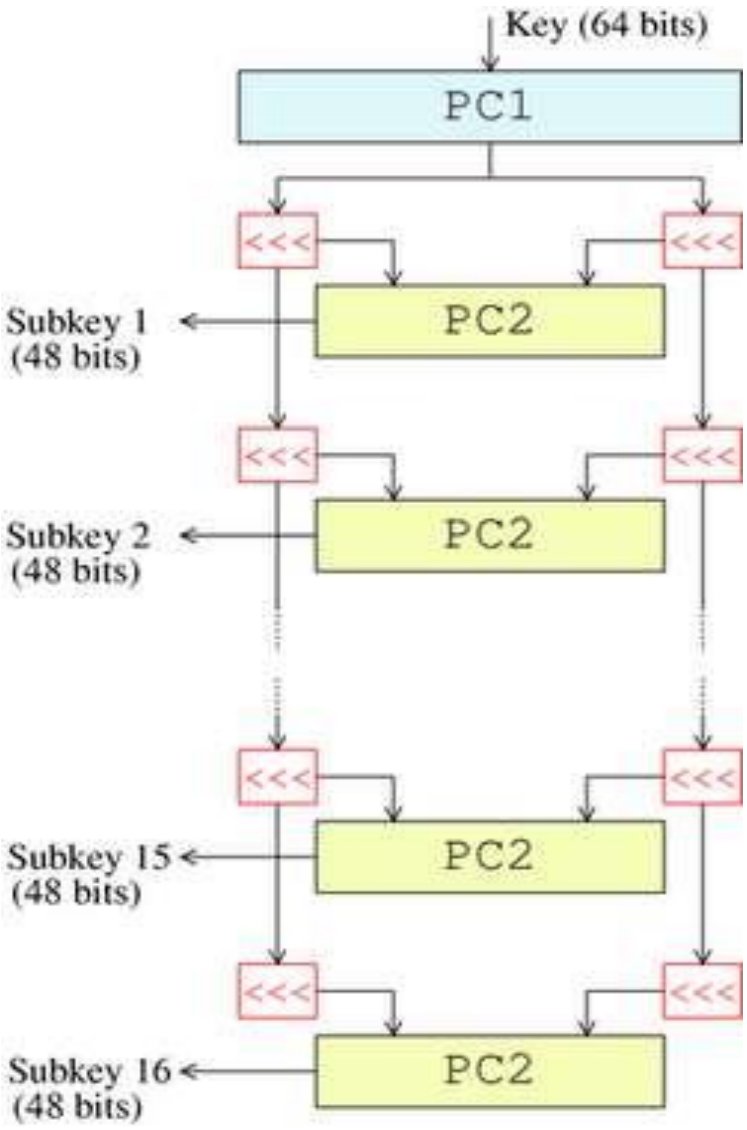
$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1}, K_i)$$

  - *Bước 3*: hoán vị kết thúc (*Final Permutation* - **FP**).
    - Sau vòng lặp 16: ghép L_{16} và R_{16} thành C_{16}.
    - Hoán vị nghịch đảo C_{16} => có bản mã khối **C**.



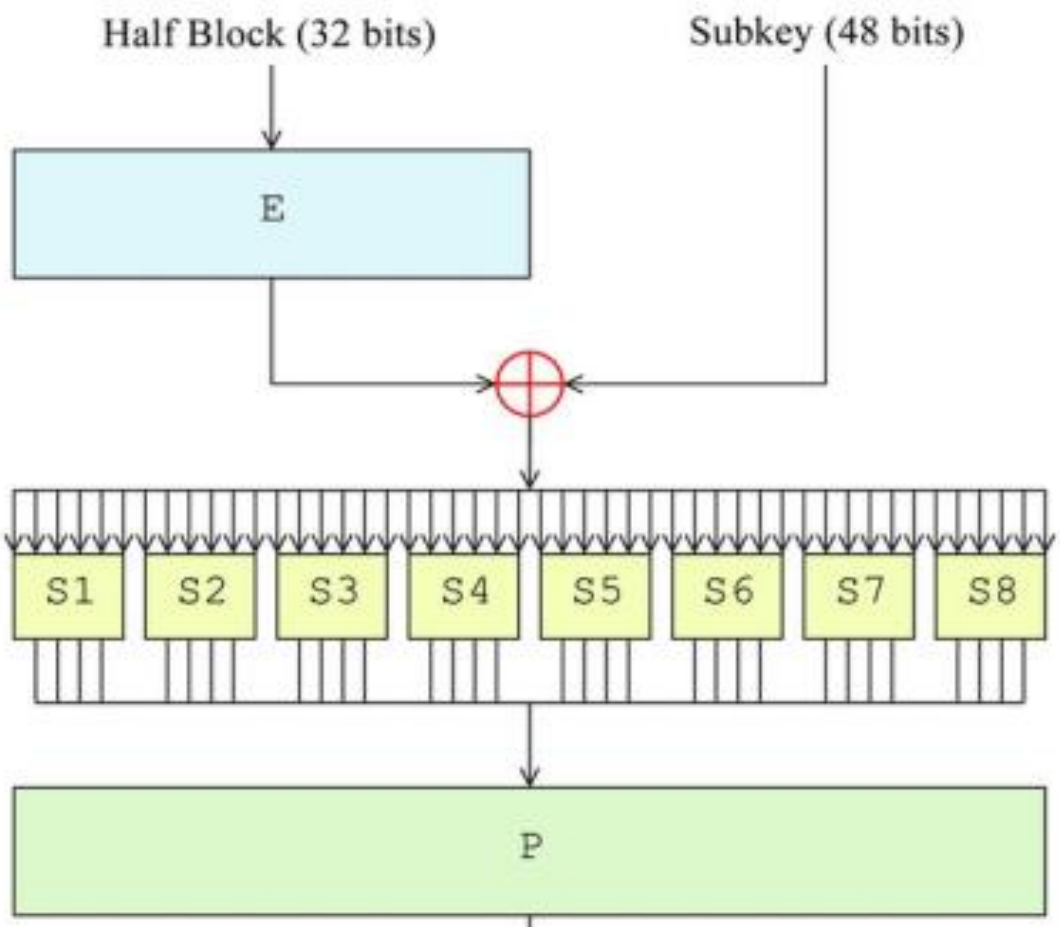
Data Encryption Standard (DES)

- **Thuật toán sinh khóa của DES:**
  - *Permuted Choice 1*: chọn **56** bit từ khóa ban đầu 64 bit.
  - 56 bit **PC1** chia thành 2 phần **28** bit.
    - Mỗi phần thực hiện xoay trái (*Left rotation*) 1 (hoặc 2 bit tùy thuật toán)
    - Ghép 2 phần đã xoay thành 48 bit (**PC2**).
    - Trích PC2 ra làm *Sub-key 1*.
  - Lặp lại bước trên để tạo 15 *Sub-key* còn lại.



Data Encryption Standard (DES)

- **Hàm F (Fiestel) với khối 32 bít của DES:**
  - **E** = ***Expansion*** (mở rộng **32** bits -> **48** bits)
  - ⊕ = ***XOR*** với *Subkey*.
  - **S1**..**S8** = ***Substitution*** (thay thế **6** bits -> **4** bits)

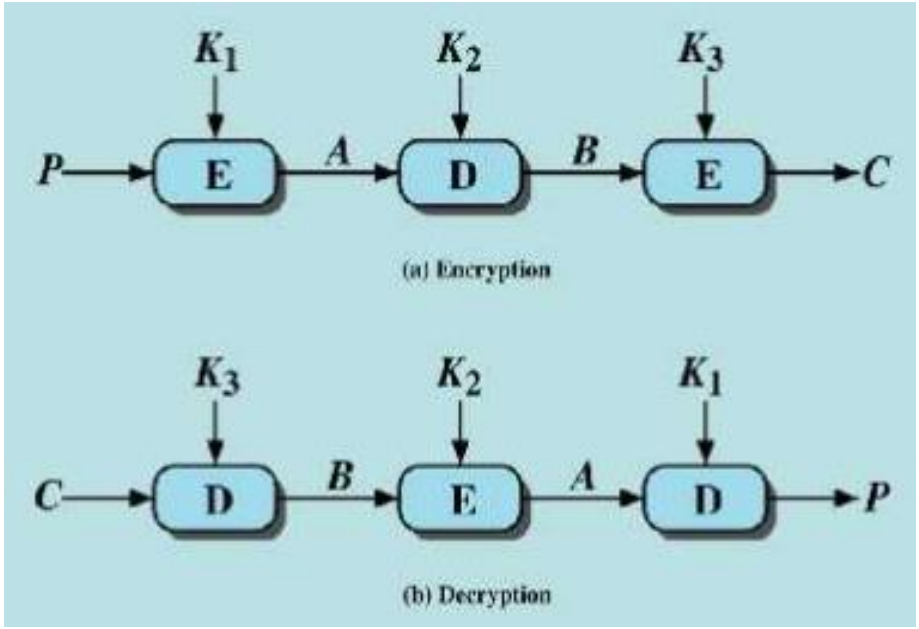


Data Encryption Standard (DES)

- **Hàm F (Fiestel) với khối 32 bít của DES:**
  - 1. ***Expansion*** (mở rộng):
    - Dùng phương thức **$E(R_{i-1})$** để mở rộng **R_{i-1}** từ **32** bit thành **48** bit.
  - 2. ***XOR***:
    - Tính: **$E(R_{i-1}) \oplus Ki = B$** (block **48** bits)
  - 3. ***Substitution*** (thay thế):
    - Khối **B** (**48** bit) được chia thành **8** khối con **B1..B8** (**B_i** = **6** bít)
    - Tra **6** bít của **B_i** vào *Bảng ma trận mật mã khối* **S** = (16 x 4) để nhận được **4** bit mật mã
      - ➢ Lấy **2** bit thứ ***1*** và ***6*** của **B_i** làm mã dòng (*row*).
      - ➢ Lấy **4** bit thứ ***2*** đến ***3*** của **B_i** làm mã cột (*column*).
    - **8** khối con **B1..B8** sau khi đi qua **S** còn lại **32** bit (8 x 4 bit)

Triple DES (3DES)

- **Tổng quát về 3-DES:**
  - DES FIBS PUB 46-3 (1999).
  - Thực hiện 3 lần mã DES
- **Mã hóa:**
  - **C = E{K3, D[K2, E(K1, P)]}**
- **Giải mã:**
  - **P = D{K1, E[K2, D(K3, C)]}**
- **Đánh giá:**
  - Độ an toàn dự kiến (NIST)
    - 3DES (2 khóa) đến 2009
    - 3DES (3 khóa) đến 2030
  - Hiệu quả cao nhưng tốc độ chậm
  - Mã AES sẽ thay thế 3DES



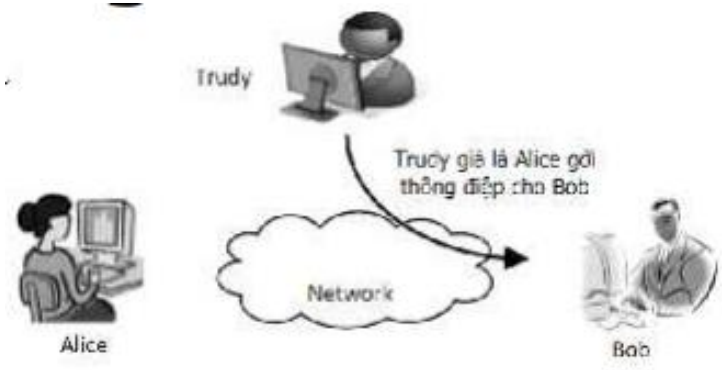
Nếu dùng 2 khóa thì ***K1 = K3***

Advanced Encryption Standard (AES)

- **Tính chất của AES:**
  - Chuẩn NIST, FIBS PUB 197 (2001)
  - Thiết kế bởi Vincent Rijmen và Joan Daemen, sau đó là Rijdael
  - **Block size** = **128** bit và Khóa **K** = **128/192/256** bit
  - Không dựa trên mã Feistel.
  - Số lượng vòng có thể thay đổi từ 10 đến 14 vòng
- **Hiệu quả:**
  - Độ an toàn bằng hoặc cao hơn 3DES => Thay thế cho 3DES
  - Từ 2003, AES được sử dụng ở Mỹ cho các dữ liệu mật và tuyệt mật
  - Chống lại được mọi sự thám mã cho đến nay!

Ứng dụng mã khóa đối xứng

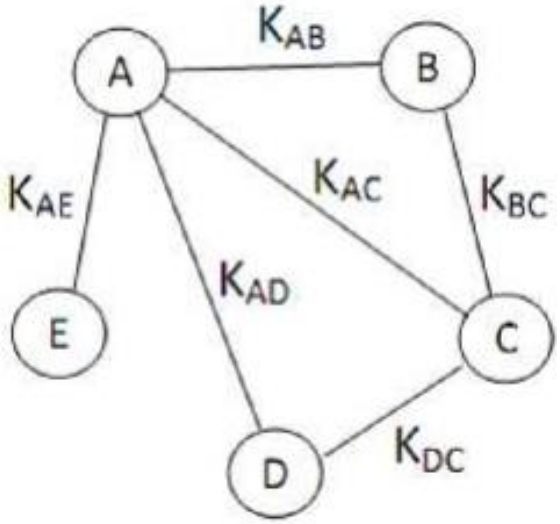
- **Tính xác thực (Authentication) của mã đối xứng:**
  - Tình huống
    - Giả sử *Alice* và *Bob* thỏa thuận khóa bí mật K
    - *Alice* và *Trudy* cùng gởi thông điệp đã mã hóa đến Bob.



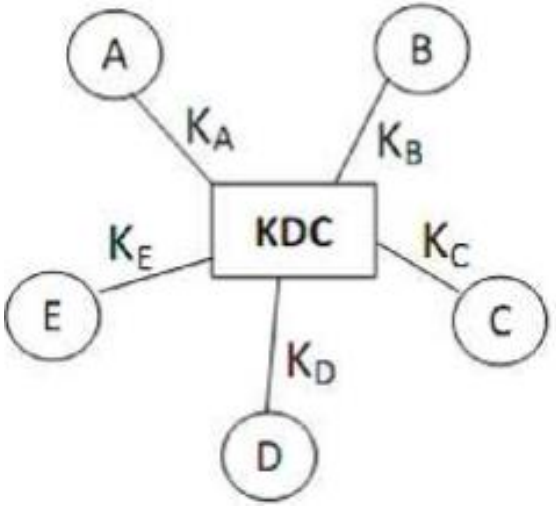
    - Khi nhận thông điệp, *Bob* cần xác thực thông điệp nào là của *Alice*?
  - Cách giải quyết:
    - Bob nhận "MHFSLHIFSNIQU", dùng khóa K để giải = "meetmetonight" –> cụm từ này *có nghĩa*! => đúng là gởi từ *Alice* !!!
    - Bob một thông điệp, "FNERIUF", dùng khóa K (vì nghĩ là Alice gửi) để giải mã thành "gfhntqs" –> *vô nghĩa*! => Không phải là gửi từ *Alice !!!*
  - Thực tế: tính "*có nghĩa*" là khó xác định với máy tính.

Ứng dụng mã khóa đối xứng

- **Vấn đề trao đổi khóa bí mật:**
  - Người gửi và người nhận phải thỏa thuận cùng một khóa bí mật.
    - Khó quản lý khóa.
    - Nếu có N người cần trao đổi với nhau mỗi người cần **N(N-1)/2** khóa.



  - Dùng trung tâm phân phối khóa (Key Distribution Center - KDC)
    - Mỗi người giữ một khóa bí mật với KDC.
    - Khóa bí mật giữa 2 người do KDC cung cấp.



Cám ơn !