ĐẠI HỌC KINH TẾ QUỐC DÂN
KHOA CÔNG NGHỆ THÔNG TIN

Bài giảng

AN TOÀN VÀ BẢO MẬT THÔNG TIN

Chương 2: MÃ HOÁ KHOÁ ĐỐI XỨNG (MÃ HOÁ KHOÁ BÍ MẬT)

ThS. Nguyễn Quốc Thái

MỤC TIÊU CHƯƠNG

- 📚 **Kiến thức:**
  - Hiểu khái niệm: bản rõ, bản mã, khóa, mã hóa & giải mã
  - Trình bày các thuật toán mã hóa đối xứng cổ điển & hiện đại
  - Nắm tính chứng thực, tính không chối bỏ và cơ chế KDC
- 🛠️ **Kỹ năng:**
  - Phân tích, so sánh các thuật toán mã hóa đối xứng
  - Vận dụng thuật toán để mã hóa/giải mã dữ liệu đơn giản
  - Nhận diện tấn công thám mã và đề xuất biện pháp phòng chống

NỘI DUNG CHƯƠNG

1. Giới thiệu chung
2. Mã hoá đối xứng cổ điển
3. Mã hoá đối xứng hiện đại
4. Tính chứng thực (authentication) và tính không từ chối (non-repudiation)
5. Trao đổi khoá bí mật bằng trung tâm phân phối khoá (KDC)

1 – GIỚI THIỆU CHUNG

- Các khái niệm cơ bản
- Các yêu cầu để sử dụng an toàn hệ mã hóa
- Đặc trưng của hệ mã hóa
- Thám mã
- Độ an toàn của hệ mã hóa

1 – GIỚI THIỆU CHUNG

- **Các khái niệm cơ bản**
- Khái niệm: Mã hóa khóa đối xứng sử dụng cùng một khóa cho việc mã hóa và giải mã => Có thể nói mã đối xứng là mã một khoá (hay mã khóa riêng, mã khoá thỏa thuận).

=> Bản chất: Người gửi và người nhận chia sẻ khoá chung K, mà họ có thể trao đổi bí mật với nhau. Ta xét hai hàm ngược nhau: E là hàm biến đổi bản rõ thành bản mã và D là hàm biến đổi bản mã trở về bản rõ. Giả sử gọi X là văn bản cần mã hóa và Y là dạng văn bản đã được thay đổi qua việc mã hóa => Ta có ký hiệu: $Y = E_K(X)$, $X = D_K(Y)$.

1 – GIỚI THIỆU CHUNG

- **Các khái niệm cơ bản**

Văn bản rõ

Văn bản rõ (plaintext, clear text) là bất kỳ dữ liệu nào có thể đọc được và cần được bảo vệ

Văn bản rõ có thể được sử dụng mà không cần thuật toán giải mã

Văn bản mã hoá

Văn bản mã hoá(ciphertext) là kết quả sau khi mã hoá văn bản rõ sử dụng thuật toán mã hoá

Văn bản mã hoá không thể được sử dụng hoặc đọc trực tiếp – cần thuật toán giải mã

1 – GIỚI THIỆU CHUNG

- **Các khái niệm cơ bản**
  - Bản rõ (plain text): X được gọi là là bản tin gốc. Bản rõ có thể được chia nhỏ có kích thước phù hợp.
  - Bản mã (cipher text): Y là bản tin gốc đã được mã hoá. Ở đây ta thường xét phương pháp mã hóa mà không làm thay đổi kích thước của bản rõ, tức là chúng có cùng độ dài.
  - Mã: là thuật toán E chuyển bản rõ thành bản mã. Thông thường chúng ta cần thuật toán mã hóa mạnh, cho dù kẻ thù biết được thuật toán, nhưng không biết thông tin về khóa cũng không tìm được bản rõ.
  - Khoá (key): K là thông tin tham số dùng để mã hoá, chỉ có người gửi và người nhận biết. Khóa là độc lập với bản rõ và có độ dài phù hợp với yêu cầu bảo mật.
  - Mã hoá (encryption): là quá trình chuyển bản rõ thành bản mã, thường gồm việc áp dụng thuật toán mã hóa và một số quá trình xử lý thông tin kèm theo.
  - Giải mã (decryption): chuyển bản mã => bản rõ,quá trình ngược lại của mã hóa.

1 – GIỚI THIỆU CHUNG

- **Các khái niệm cơ bản**
  - Mô hình mã hoá đối xứng

Mã hoá (Encrypt) là quá trình chuyển văn bản rõ thành văn bản mã hoá

C = E(P)

Giải mã (Decrypt) là quá trình đưa văn bản mã hoá về lại văn bản gốc đầu vào

P = D(C)

Khoá (Key) là chuỗi dữ liệu cần thiết cho cả quá trình giải mã và mã hoá
Độc lập với bản rõ

P Bản rõ (Plaintext)

C Bản mã (Ciphertext)

1 – GIỚI THIỆU CHUNG

- **Các khái niệm cơ bản**
  - Mật mã: là chuyên ngành khoa học của KHMT nghiên cứu về các nguyên lý và phương pháp mã hoá. Hiện nay người ta đưa ra nhiều chuẩn an toàn cho các lĩnh vực khác nhau của CNTT.
  - Thám mã: nghiên cứu các nguyên lý và phương pháp giải mã mà không biết khoá. Thông thường khi đưa các mã mạnh ra làm chuẩn dùng chung giữa các người sử dụng, các mã đó được các kẻ thám mã cũng như những người phát triển mã tìm hiểu nghiên cứu các phương pháp giải một phần bản mã với các thông tin không đầy đủ.
  - Lý thuyết mã: gồm cả mật mã và thám mã, là một thể thống nhất, để đánh giá một mã mạnh hay không, đều phải xét từ cả hai khía cạnh.

1 – GIỚI THIỆU CHUNG

- **Các yêu cầu để sử dụng an toàn hệ mã hóa**
  - Thuật toán mã hoá mạnh: Có cơ sở toán học vững chắc đảm bảo rằng mặc dù công khai thuật toán, mọi người đều biết, nhưng việc thám mã là rất khó khăn và phức tạp nếu không biết khóa.
  - Khoá mật chỉ có người gửi và người nhận biết: Có kênh an toàn để phân phối khoá giữa các người sử dụng chia sẻ khóa, mối liên hệ giữa khóa và bản mã là không nhận biết được.
- **Đặc trưng của hệ mã hóa**
  - Kiểu của thao tác mã hoá được sử dụng trên bản rõ.
  - Số lượng khoá được sử dụng khi mã hóa: một khoá duy nhất (khoá riêng) hoặc hai khoá (khoá công khai).
  - Cách mà bản rõ được xử lý: Theo khối hay dòng.

1 – GIỚI THIỆU CHUNG

- **Thám mã**
  - Tấn công thám mã dựa trên thuật toán và một số thông tin về các đặc trưng chung về bản rõ nhằm khai phá các đặc trưng của thuật toán để tìm bản rõ cụ thể hoặc tìm khóa.
  - Tấn công duyệt toàn bộ (Brute-Force): kẻ tấn công tìm cách thử mọi khóa trên bản mã cho đến khi nhận được bản rõ.
  - Các kiểu tấn công thám mã:
    - Chỉ dùng bản mã: biết thuật toán bản mã, dùng phương pháp thống kê, xác định bản rõ.
    - Biết bản rõ: biết thuật toán, biết được bản mã/bản rõ tấn công tìm khóa.
    - Chọn bản rõ: chọn bản rõ và nhận được bản mã, biết thuật toán tấn công tìm khóa.
    - Chọn bản mã: chọn bản mã và có được bản rõ tương ứng, biết thuật toán tấn công tìm khóa.
    - Chọn bản tin: chọn được bản rõ hoặc mã và mã hoặc giải mã tương ứng, tấn công tìm khóa.

1 – GIỚI THIỆU CHUNG

- **Độ an toàn của hệ mã hoá**
  - An toàn không điều kiện: Không phụ thuộc vào máy tính,…
  - An toàn tính toán: Nguồn lực máy tính, thời gian có hạn.

1 – GIỚI THIỆU CHUNG

- **Phân loại mã hoá**
  - Mã hoá khoá đối xứng (Symmetric Cryptosystems)

$$K_E = K_D$$

Hệ thống mã hoá đối xứng chỉ sử dụng một khoá để mã hoá và giải mã

Có thể được gọi là hệ thống mật mã khoá riêng (Private key cryptosystem)

1 – GIỚI THIỆU CHUNG

- **Phân loại mã hoá**
  - Mã hoá khoá đối xứng (Symmetric Cryptosystems)

Mã hoá cổ điển (Classical cryptographic) → **Mã hoá hiện đại** (Modern cryptographic)

Sử dụng thuật toán cổ điển cơ bản:
thay thế (substitution), hoán vị (transposition)

- Mã Caesar
- Mã hoá Vigenère
- Mã hoá đơn bảng
- One-Time Pad (OTP)

Sử dụng thuật toán hiện đại:
Biểu diễn dữ liệu thông qua giá trị nhị phân

Mã dòng (Stream cipher)

- Mã A5/1
- Mã RC4

Mã khối (Block cipher)

- DES
- AES

1 – GIỚI THIỆU CHUNG

- **Phân loại mã hoá**
  - Mã hoá khoá bất đối xứng (Asymmetric Cryptosystems)

Văn bản rõ đầu vào

Văn bản mã hoá

Văn bản rõ đầu ra

$$K_E \neq K_D$$

Hệ thống mã hoá bất đối xứng sử dụng các khoá khác nhau, một khoá công khai và một khoá riêng để mã hoá và giải mã

Có thể được gọi là hệ thống mật mã khoá công khai (Public key cryptosystem)
Mã hoá: sử dụng khoá công khai
Giải mã: sử dụng khoá riêng

2 – MÃ HOÁ ĐỐI XỨNG CỔ ĐIỂN

- Giới thiệu
- Hệ mã hoá cổ điển thay thế
- Hệ mã hoá cổ điển hoán vị
- Một số vấn đề khác * (đọc nâng cao)

2 – MÃ HOÁ ĐỐI XỨNG CỔ ĐIỂN

- **Giới thiệu**
  - Mã hoá đối xứng cổ điển là phương pháp mã hoá đơn giản nhất xuất hiện đầu tiên trong lịch sử ngành mã hoá, thuật toán đơn giản và dễ hiểu, là cơ sở cho việc nghiên cứu và phát triển thuật toán mã hoá đối xứng được sử dụng ngày nay. Mã hoá cổ điển có hai phương pháp nổi bật là:
  - Mã hoá thay thế
  - Mã hoá hoán vị

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Khái niệm:** là phương pháp mà từng kí tự (nhóm kí tự) trong bản rõ được thay thế bằng một kí tự (một nhóm kí tự) khác để tạo ra bản mã. Bên nhận chỉ cần thay thế ngược lại trên bản mã để có được bản rõ ban đầu.
- **Một số hệ mã:**
  - Hệ mã ceasar
  - Các mã bảng chữ đơn
  - Hệ mã Playfair
  - Hệ mã Vigenere

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Caesar**
  - Phương pháp: Việc mã hoá được thực hiện đơn giản là thay mỗi chữ trong bản rõ bằng chữ thứ ba tiếp theo trong bảng chữ cái.
  - Ví dụ:
    - Mã hóa bản rõ: “Meet me after the toga party”
    - Bản mã hóa:“PHHW PH DIWHU WKH WRJD SDUWB”

  => Nghĩa là: ta thay chữ “m” bằng chữ đứng thứ 3 sau “m” là “p” (m, n, o, p); thay chữ “e” bằng chữ đứng thứ 3 sau “e” là “h” (e, f, g, h),

  - => Thám mã: Ta chỉ việc thay thế ngược lại mỗi chữ trong bản mã.

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Các mã bảng chữ đơn**
  - Phương pháp: Mã hoá các chữ không chỉ là dịch chuyển bảng chữ, mà có thể tạo ra các bước nhảy khác nhau cho các chữ. Trong một mã mỗi chữ của bản rõ được ánh xạ đến một chữ khác nhau của bản mã nên mỗi cách mã như vậy sẽ tương ứng với một hoán vị của bảng chữ và hoán vị đó là khoá của mã đã cho.
  - Ví dụ: Mã hóa bản rõ: ”ifwewishtoreplaceletters”, giả sử có bản mã tương ứng với bản rõ trong mã bảng chữ đơn như sau:
    - Plain: abcdefghijklmnopqrstuvwxyz
    - Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Các mã bảng chữ đơn**
  - Ví dụ: Mã hóa bản rõ: ”ifwewishtoreplaceletters”, giả sử có bản mã tương ứng với bản rõ trong mã bảng chữ đơn như sau:
    - Plain: abcdefghijklmnopqrstuvwxyz
    - Cipher: DKVQFIBJWPESCXHTMYAUOLRGZN

    => Ta có bản mã hóa của bản rõ là:

    - Plaintext: ifwewishtoreplaceletters
    - Ciphertext: WIRFRWAJUHYFTSDVFSFUUFYA

    => Thám mã: 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 như: 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.

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Playfair**
  - Phương pháp: Mỗi chữ sẽ được mã bằng một số chữ khác nhau tùy thuộc vào các chữ mà nó đứng cạnh. Trong mã Playfair, mỗi chữ có thể được mã bằng một trong 7 chữ khác nhau tùy vào chữ cặp đôi cùng nó trong bản rõ sử dụng ma trận khóa Playfair. Ma trận khóa Playfair được thiết lập như sau: Cho trước một từ làm khoá (với điều kiện trong từ khoá 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 như sau:
    - Trước hết viết các chữ của từ khoá vào các hàng của ma trận bắt đầu 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 (viết theo một trình tự qui ước như: từ đầu bảng chữ cái cho đến cuối).
    - Do có 26 chữ cái tiếng Anh, nên thiếu một ô nên ta sẽ dồn hai chữ nào đó vào một ô chung (chẳng hạn I và J).

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Playfair**
  - 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 như sau:
    - Trước hết viết các chữ của từ khoá vào các hàng của ma trận bắt đầu 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 (viết theo một trình tự qui ước như: từ đầu bảng chữ cái cho đến cuối).
    - Do có 26 chữ cái tiếng Anh, nên thiếu một ô nên ta sẽ dồn hai chữ nào đó vào một ô chung (chẳng hạn I và J).
    - Giả sử dùng từ khoá MONARCHY => Ta lập ma trận khoá Playfair:
      - MONAR
      - CHYBD
      - EFGIK
      - LPQST
      - UVWXZ

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Playfair**

  => Mã hóa và giải mã: Bản rõ được mã hoá 2 chữ cùng một lúc theo qui tắc:

  - 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".
  - 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"
  - 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"
  - 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)

  => Thám mã: Sử dụng phương pháp thống kê tần suất.

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Vigenere**
  - Phương pháp: 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.
  - Chi tiết như sau:
    - Giả sử khoá là một chữ có độ dài d được viết dạng K = K1 K2 …Kd , trong đó K i nhận giá trị nguyên từ 0 đến 25.
    - Khi đó 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 giãn tương đối đều. Giải mã 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.

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Vigenere**
  - Để sử dụng mã Vigenere với từ khóa và bản rõ, ta thực hiện các bước:
    - Viết bản rõ ra.
    - Viết từ khoá lặp nhiều lần phía trên tương ứng của nó.
    - Sử dụng mỗi chữ của từ khoá như khoá của mã Ceasar.
    - Mã chữ tương ứng của bản rõ với bước nhảy tương ứng.

2.1 – HỆ MÃ HOÁ CỔ ĐIỂN THAY THẾ

- **Hệ mã Vigenere**
  - Ví dụ: Sử dụng từ khoá deceptive mã hóa bản rõ như dưới đây?
    - key: deceptivedeceptivedeceptive
    - plaintext: wearediscoveredsaveyourself
    - ciphertext:ZICVTWQNGRZGVTWAVZHCQYGL
  - Thực hiện: Để mã chữ w, ta tìm chữ đầu của khóa là d, do đó w sẽ được mã trên bảng chữ tịnh tiến 3 (tức là a tịnh tiến vào d) nên chữ đầu w được mã bởi chữ Z. Chữ thứ hai trong từ khóa là e, có nghĩa là chữ thứ hai trong bản rõ sẽ được tịnh tiến 4 (từ a tịnh tiến đến e). Như vậy, chữ thứ hai trong bản rõ e sẽ được mã bởi chữ I,…. Tương tự như vậy cho đến hết bản rõ
  - => Thám mã: Sử dụng phương pháp thống kê tần suất.

2.2 – HỆ MÃ HOÁ CỔ ĐIỂN HOÁN VỊ

- **Khái niệm:** Các kí tự trong bản rõ vẫn được giữ nguyên, chúng chỉ được sắp xếp lại vị trí để tạo ra bản mã. Tức là các kí tự trong bản rõ hoàn toàn không bị thay đổi bằng kí tự khác mà chỉ đảo chỗ của chúng để tạo thành bản mã. Có một số hệ mã sau:
  - **Hệ mã Rail Fence**
  - **Hệ mã dịch chuyển dòng**

2.2 – HỆ MÃ HOÁ CỔ ĐIỂN HOÁN VỊ

- **Hệ mã Rail Fence**:
  - Phương pháp: Viết các chữ của bản rõ theo đường chéo (hình răng cưa) trên một số dòng. Sau đó viết các chữ theo từng dòng sẽ nhận được bản mã. Số dòng chính là khoá của mã (vì khi biết số dòng ta sẽ tính được số chữ trên mỗi dòng và lại viết bản mã theo các dòng sau đó lấy bản rõ bằng cách viết lại theo các cột). Quá trình giải mã được thực hiện ngược lại.
  - Ví dụ: Mã hóa bản rõ "meet me after the toga party " với khóa K = 2 (số dòng bằng 2)?
  - Giải: Ta viết bản rõ "meet me after the toga party" lần lượt trên 2 dòng:

m e m a t r h t g p r y

e t e f e t e o a a t

  - Sau đó ghép các chữ ở dòng thứ nhất với các chữ ở dòng thứ hai cho bản mã: MEMATRHTGPRYETEFETEOAAT
  - => Thám mã: Đơn giản.

2.2 – HỆ MÃ HOÁ CỔ ĐIỂN HOÁN VỊ

- **Hệ mã dịch chuyển dòng:**
  - Phương pháp: Viết các chữ của bản rõ theo các dòng với số cột xác định. Sau đó thay đổi thứ tự các cột theo một dãy số khoá cho trước, rồi đọc lại chúng theo các cột để nhận được bản mã. Quá trình giải mã được thực hiện ngược lại.
  - Ví dụ: Thực hiện mã hóa bản rõ (số cột bằng 7) với khóa như dưới đây?
  - Giải: 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

  Sau đó, ta viết theo thứ tự các cột từ 1 đến 7 để nhận được bản mã: Ciphertext: TTNAAPTMTSUOAODWCOIXKNLYPETZ

  => Thám mã: Đơn giản.

2.2 – HỆ MÃ HOÁ CỔ ĐIỂN HOÁN VỊ

- **Điểm yếu của mã cổ điển:**
- Phương pháp mã hoá cổ điển có thể dễ dàng bị giải mã bằng cách đoán chữ dựa trên phương pháp thống kê tần suất xuất hiện các chữ cái trên mã và so sánh với bảng thống kê quan sát của bản rõ.
- Để dùng được mã hoá cổ điển thì bên mã hoá và bên giải mã phải thống nhất với nhau về cơ chế mã hoá cũng như giải mã.

3 – MÃ HOÁ ĐỐI XỨNG HIỆN ĐẠI

- Giới thiệu
- Mã hoá dòng (Stream Cipher)
- Mã hoá khối (Block Cipher)

3 – MÃ HOÁ ĐỐI XỨNG HIỆN ĐẠI

- **Giới thiệu**
  - Đối tượng mã hóa của các phương pháp mã hoá đối xứng cổ điển chỉ là các chữ cái, trong khi đó bản tin hiện nay có thể là hình ảnh, video, âm thanh, ký tự đặc biệt,.. => Ngoài ra, rất quan tâm đến vấn đề chống phá mã trong các hệ mã nên các hệ mã hóa hiện đại được phát triển.
  - Để minh họa mã hóa đối xứng hiện đại, ta sử dụng bản rõ là các chữ cái của một ngôn ngữ gồm có 8 chữ cái: A, B, C, D, E, F, G, H; trong đó mỗi chữ cái được biểu diễn bằng 3 bít nhị phân như bảng sau:

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

3 – MÃ HOÁ ĐỐI XỨNG HIỆN ĐẠI

- **Giới thiệu**
  - Giả sử bản rõ là các chữ cái của một ngôn ngữ gồm 8 chữ cái: A, B, C, D, E, F, G, H; lần lượt được biểu diễn bằng 4 bít như bảng sau:
  - Nếu bản rõ là "HEAD" thì biểu diễn nhị phân: 111100000111
  - Sử dụng khoá K gồm 4 bít để mã hoá bản rõ là: 0101
  - Mã hoá bản rõ bằng phép **XOR** (⊕) với khoá K:

1111 0000 0111 ⊕ 0101 0101 0101 = 1010 0101 0110 (FBCG)

  - Giải mã: lấy bản mã XOR với khoá để thu được bản rõ

⇒ Phương pháp XOR như trên khá đơn giản, do đó ra đời mã hoá phức tạp hơn

⇒ Có 2 phương pháp mã hoá đối xứng hiện đại: **Mã hoá dòng (Stream Cipher)** và **mã hoá khối (Block Cipher)**

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

3.1 – MÃ DÒNG

- **Mã dòng (Stream Cipher)**
- **Quá trình mã hoá:**

(1) Bản rõ được chia thành các đơn vị mã hoá (mỗi đơn vị có kích thước k bit):

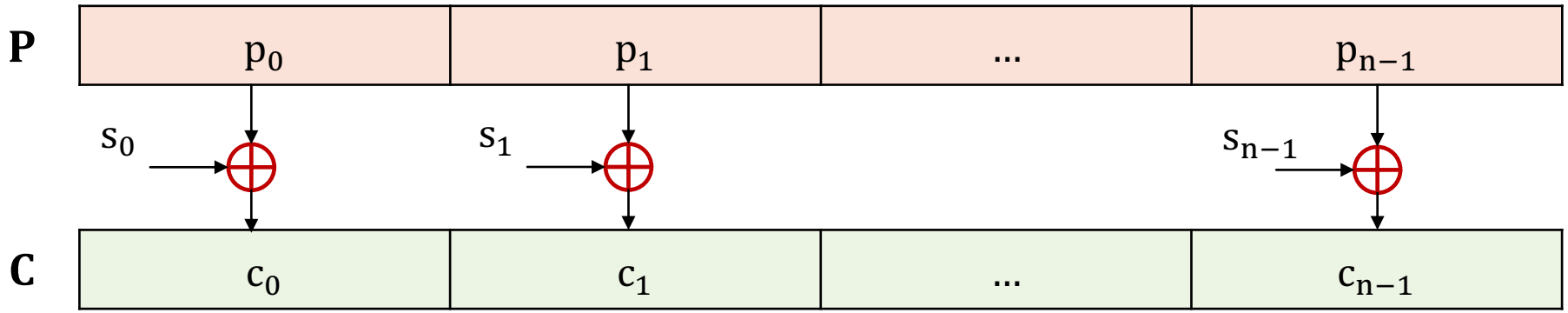
$$\mathbf{P} \rightarrow p_0 p_1 p_2 \dots p_{n-1} \qquad (p_i : k\,\text{bit})$$

(2) Một bộ sinh dãy số ngẫu nhiên: Dùng khoá **K** ban đầu để sinh ra các số ngẫu nhiên có kích thước bằng kích thước đơn vị mã hoá:

$$\text{StreamCipher}(K) \rightarrow S = s_0 s_1 s_2 \dots s_{n-1} \qquad (s_i : k\,\text{bit})$$

(3) Mỗi số ngẫu nhiên được **XOR** với đơn vị mã hoá của bãn rõ để được bản mã:

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

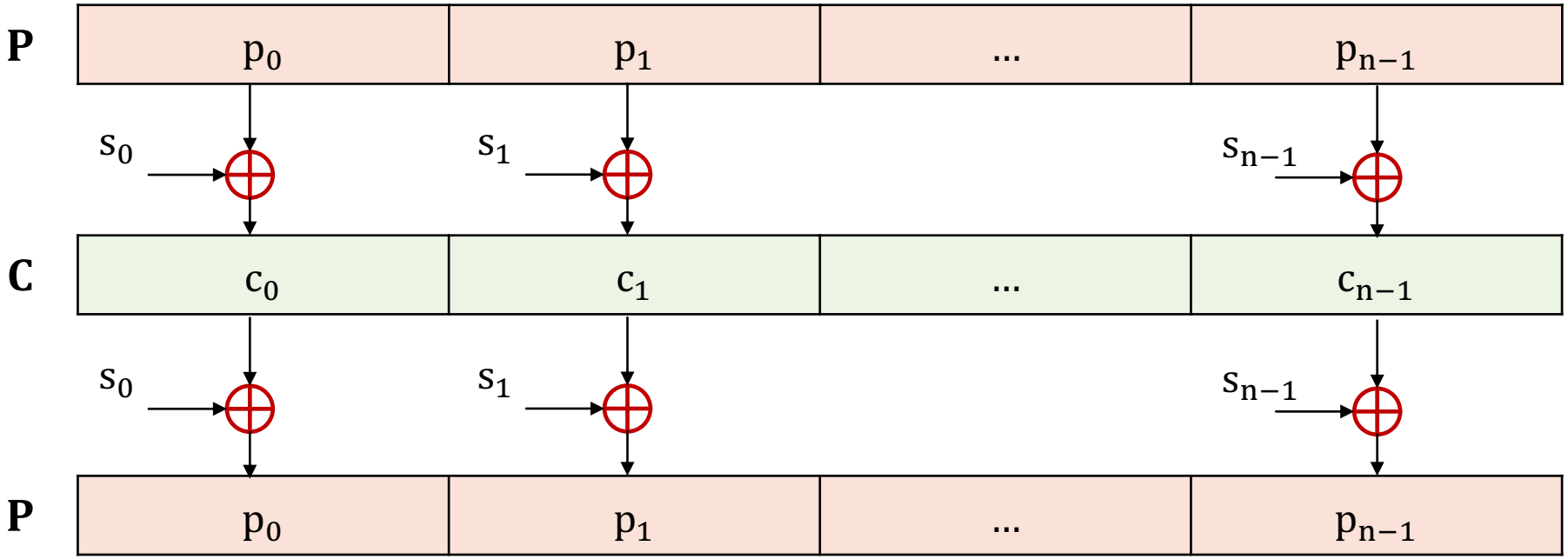


3.1 – MÃ DÒNG

- **Mã dòng (Stream Cipher)**
- **Quá trình giải 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:

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

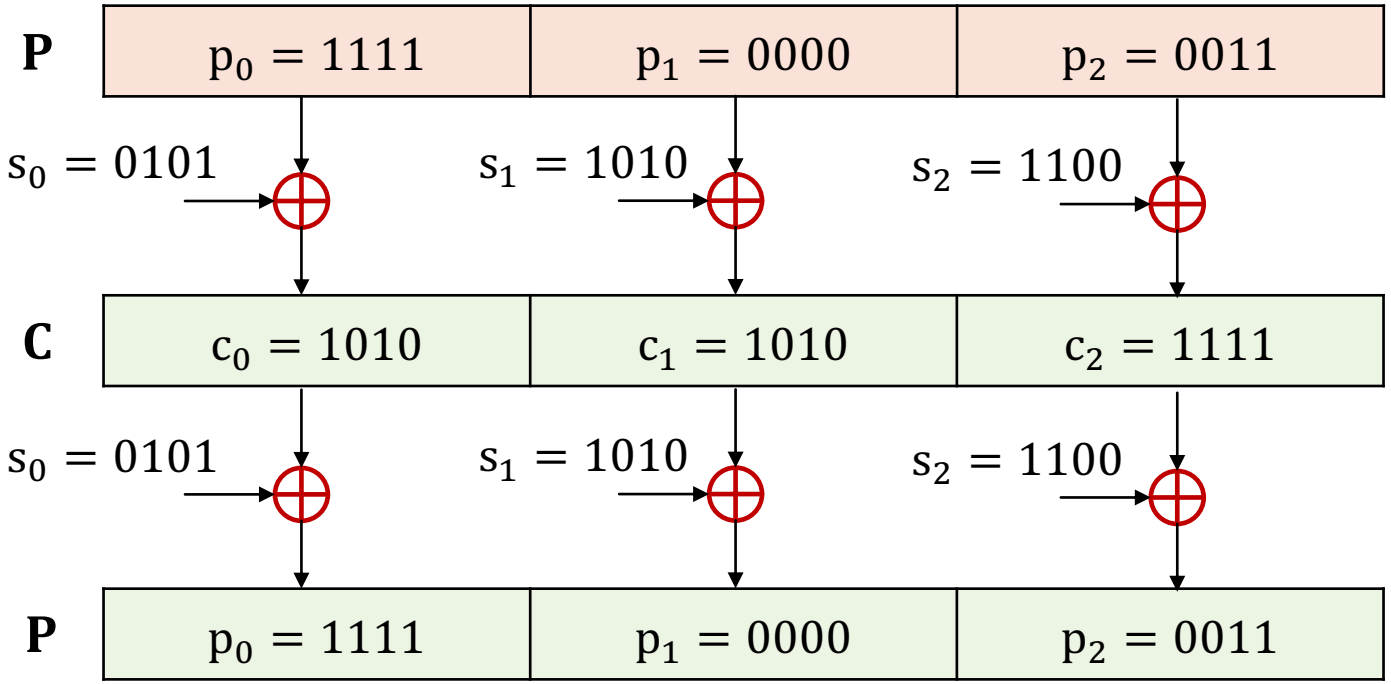


3.1 – MÃ DÒNG

- **Mã dòng (Stream Cipher)**
- Ví dụ 1:

Cho P=111100000011 (Chữ "HEAD") đơn vị mã hoá có chiều dài k = 4 bít và n =3

$$s_0 = 0101, s_1 = 1010, s_2 = 1100$$



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

3.1 – MÃ DÒNG

- **Hệ mã A5/1**

  Giới thiệu:

  - A5/1 được dùng trong mạng điện thoại GSM, để bảo mật dữ liệu trong quá trình liên lạc giữa máy tính điện thoại và trạm thu phát sóng vô tuyến
  - Đơn vị mã hoá của A5/1 là 1 bít. Bộ sinh số mỗi lần sẽ sinh ra hoặc bít 0 hoặc bít 1 để sử dụng trong phép XOR

  ⇒ Trong bài học giới hạn xét mô hình thu nhỏ của A5/1, gọi tắt là TinyA5/1

3.1 – MÃ DÒNG

- **Hệ mã TinyA5/1**
  - Bộ sinh số gồm 3 thanh ghi X, Y, Z:
    - X gồm 6 bít $(x_0, x_1, \dots, x_5)$
    - Y gồm 8 bít $(y_0, y_1, \dots, y_7)$
    - Z gồm 9 bít $(z_0, z_1, \dots, z_8)$
  - Khoá K có chiều dài 23 bít là được phân bổ vào các thanh ghi: $K \longrightarrow XYZ$
  - Các thanh ghi X, Y, Z được biến đổi theo các **quy tắc quay**

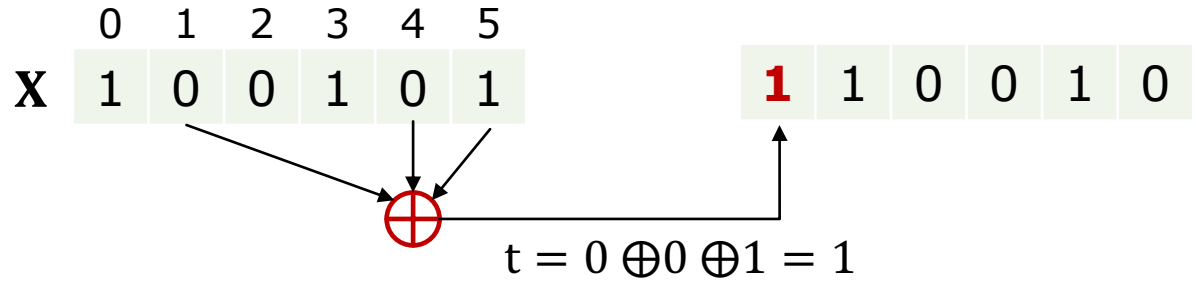
3.1 – MÃ DÒNG

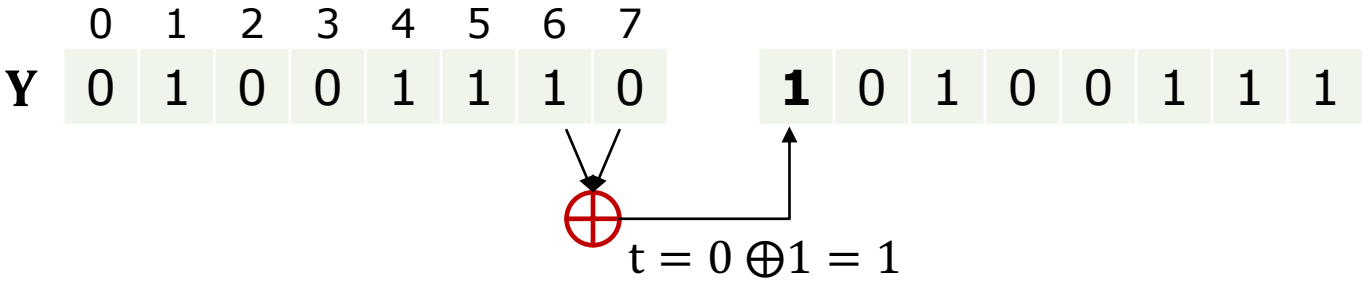
- **Hệ mã TinyA5/1**
  - Đầu vào: P, K
  - Nội dung thuật toán:

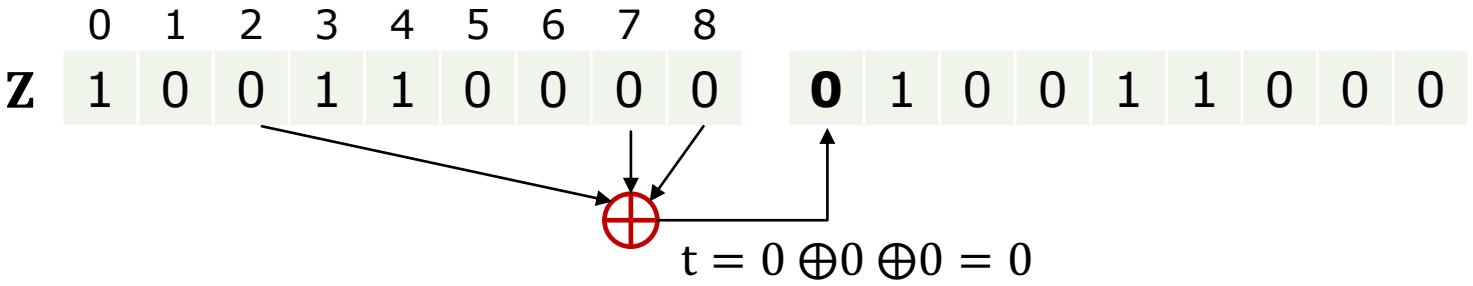
    Phân bổ K vào các thanh ghi XYZ
    Tại bước sinh số thứ i, thực hiện các phép tính:
    Tính m = maj(x_1, y_3, z_3) là ”hàm chiếm đa số”. Nếu trong 3 bít x_1, y_3, z_3 có từ hai bít 0 trở lên thì hàm trả về giá trị 0; Ngược lại, hàm trả về giá trị 1.
    Kiểm tra:
    Nếu $x_1 = m$ thì thực hiện **Quay X**;
    Nếu $y_3 = m$ thì thực hiện **Quay Y**;
    Nếu $z_3 = m$ thì thực hiện **Quay Z**;
    Tính $s_i = x_5 \oplus y_7 \oplus z_8$
    Bản mã $C = P \ \mathbf{XOR} \ S$
  - Đầu ra: C

3.1 – MÃ DÒNG

- **Hệ mã TinyA5/1**
  - Quay X gồm các thao tác:
    - $t = x_2 \oplus x_4 \oplus x_5$
    - $x_j = x_{j-1}$ với $j = 5, 4, 3, 2, 1$
    - $x_0 = t$
  - Quay Y gồm các thao tác:
    - $t = y_6 \oplus y_7$
    - $y_j = y_{j-1}$ với $j = 7, 6, \dots, 1$
    - $y_0 = t$
  - Quay Z gồm các thao tác:
    - $t = z_2 \oplus z_7 \oplus z_8$
    - $z_j = z_{j-1}$ với $j = 8, 7, \dots, 1$
    - $z_0 = t$







3.1 – MÃ DÒNG

- **Hệ mã TinyA5/1**
- **Bài tập**: Cho bản rõ P = 111 (chữ H), khoá K = 10010101001110100110000

Phân bổ X 100101 Y 01001110 Z 100110000

- **Bước 0**: $x_1 = 0, y_3 = 0, z_3 = 1 \rightarrow m = maj(0,0,1) = 0 \rightarrow$ quay X, quay Y

X 100101 → 110010

Y 01001110 → 10100111

Z 100110000 → $s_0 = 0 \oplus 1 \oplus 0 = 1$

- **Bước 1**: $x_1 = 1, y_3 = 0, z_3 = 1 \rightarrow m = maj(1,0,1) = 1 \rightarrow$ quay X, quay Z

X 110010 → 111001

Z 100110000 → 010011000

Y 10100111 → $s_1 = 1 \oplus 1 \oplus 0 = 0$

- **Bước 2**: $x_1 = 1, y_3 = 0, z_3 = 0 \rightarrow m = maj(1,0,0) = 0 \rightarrow$ quay Y, quay Z

(Còn tiếp)

3.1 – MÃ DÒNG

- **Hệ mã TinyA5/1**
- **Bài tập**: Cho bản rõ P = 111 (chữ H), khoá K = 1001010100111010011000 0

Phân bổ X 100101 Y 01001110 Z 100110000

- **Bước 2**: $x_1 = 1, y_3 = 0, z_3 = 0 \rightarrow m = maj(1,0,0) = 0 \rightarrow$ quay Y, quay Z

Y 10100111 → 01010011

Z 0100110 00 → 101001100

X 111001 → $s_2 = 1 \oplus 1 \oplus 0 = 0$

- Vậy bản mã là **C = 111 ⊕ 100 = 011** (chữ D)
- Giải mã: **P = C ⊕ S = 011 ⊕ 100 = 111**

3.1 – MÃ DÒNG

- **Hệ mã A5/1 (Tổng quát)**
- **Đặc điểm**: Nguyên tắc bộ số A5/1 hoạt động giống TinyA5/1 nhưng kích thước thanh ghi X, Y, Z là 19, 22 và 23 bít
- Hàm maj được tính trên 3 bít: m = maj(x_8, y_{10}, z_{10})
- Quay X, Y, Z với các bít như sau:
  - Quay X :
    - $t = x_{13} \oplus x_{16} \oplus x_{17}, \oplus x_{18}$
    - $x_j = x_{j-1}$ với $j = 18, 17, \dots, 1$
    - $x_0 = t$
  - Quay Y:
    - $t = y_{20} \oplus y_{21}$
    - $y_j = y_{j-1}$ với $j = 21, 20, \dots, 1$
    - $y_0 = t$
  - Quay Z:
    - $t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$
    - $z_j = z_{j-1}$ với $j = 22, 21, \dots, 1$
    - $z_0 = t$
- Sau khi quay bít xong thì bít sinh ra: $s_i = x_8 \oplus y_{10} \oplus z_{10}$
- A5/1 được thực hiện dễ dàng bằng các thiết bị phần cứng, tốc độ nhanh

3.1 – MÃ DÒNG

- **Hệ mã A5/1 (Tổng quát)**
- **Đặc điểm**: Nguyên tắc bộ số A5/1 hoạt động giống TinyA5/1 nhưng kích thước thanh ghi X, Y, Z là 19, 22 và 23 bít
- Hàm maj được tính trên 3 bít: m = maj(x_8, y_{10}, z_{10})
- Quay X, Y, Z với các bít như sau:
  - Quay X :
    - $t = x_{13} \oplus x_{16} \oplus x_{17}, \oplus x_{18}$
    - $x_j = x_{j-1}$ với $j = 18, 17, \dots, 1$
    - $x_0 = t$
  - Quay Y:
    - $t = y_{20} \oplus y_{21}$
    - $y_j = y_{j-1}$ với $j = 21, 20, \dots, 1$
    - $y_0 = t$
  - Quay Z:
    - $t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$
    - $z_j = z_{j-1}$ với $j = 22, 21, \dots, 1$
    - $z_0 = t$
- Sau khi quay bít xong thì bít sinh ra: $s_i = x_8 \oplus y_{10} \oplus z_{10}$
- A5/1 được thực hiện dễ dàng bằng các thiết bị phần cứng, tốc độ nhanh

3.1 – MÃ DÒNG

- **Hệ mã RC4**
- **Giới thiệu**: RC4 là hệ mã được sử dụng trong giao thức SSL để bảo mật dữ liệu trong quá trình truyền dữ liệu giữa Web Server và trình duyệt Web
  - RC4 còn được sử dụng trong mã hoá WEP của mạng Wireless LAN
  - Trong giới hạn bài học xét mô hình thu nhỏ của RC4, gọi tắt là **TinyRC4**
  - **Hệ mã TinyRC4**:
    - Đơn vị mã hoá của TinyRC4 là 3 bít
    - TinyRC4 dùng 2 mảng là S và T mỗi mảng gồm 8 số nguyên 3 bít (từ 0 đến 7)
    - Khoá là một dãy gồm N số nguyên 3 bít với N có thể lấy giá trị từ 1 đến 8. Bộ sinh số mỗi lần sinh ra 3 bít để sử dụng trong phép XOR.
    - Quá trình sinh số của **TinyRC4** gồm 2 giai đoạn: **Khởi tạo** và **Sinh số**

3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Giai đoạn khởi tạo**
  - Trước tiên, dãy S gồm các số nguyên 3 bít từ 0 đến 7 được sắp thứ tự tăng dần
  - Dựa trên các phần tử của khoá K, các phần tử của S được hoán vị lẫn nhau đến một mức độ ngẫu nhiên nhất định

```
# Khoi tao day so S va T
for i = 0 to 7 do
     S[i] = i;
     T[i] = K[i mod N];
next i
# Hoan vi day S
j=0
for i = 0 to 7 do
     j = (j + S[i] + T[i]) mod 8;
     Swap(S[i], S[j]);
next j
```

3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Giai đoạn sinh số**
  - Các phần tử của S tiếp tục được hoán vị. Tại mỗi bước sinh số, hai phần tử của dãy S được chọn để tính ra số k 3 bít là số được dùng để XOR với đơn vị mã hoá của bản rõ.

```
i, j = 0;
while (true):
     i = (i + 1) mod 8;
     j = (j + S[i]) mod 8;
     Swap(S[i], S[j]);
     t = (S[i] + S[j]) mod 8;
     k = S[t];
end while
```

3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Xét ví dụ:** Cho bản rõ P = 001000110 (từ "BAG"), khoá K gồm 3 số 2, 1, 3 (N=3)
- **Giai đoạn khởi tạo:** S và T

S	0	1	2	3	4	5	6	7

T	2	1	3	2	1	3	2	1

K

```
# Khoi tao day so S va T
for i = 0 to 7 do
     S[i] = i;
     T[i] = K[i mod N];
next i
# Hoan vi day S
j=0
for i = 0 to 7 do
     j = (j + S[i] + T[i]) mod 8;
     Swap(S[i], S[j]);
next j
```

3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Xét ví dụ:** Cho bản rõ P = 001000110 (từ "BAG"), khoá K gồm 3 số 2, 1, 3 (N=3)
- **Giai đoạn khởi tạo:** S và T

S	0	1	2	3	4	5	6	7
T	2	1	3	2	1	3	2	1

```
# Khoi tao day so S va T
for i = 0 to 7 do
     S[i] = i;
     T[i] = K[i mod N];
next i
# Hoan vi day S
j=0
for i = 0 to 7 do
     j = (j + S[i] + T[i]) mod 8;
     Swap(S[i], S[j]);
next j
```

- **Hoán vị S**

i=0

T	2	1	3	2	1	3	2	1
	j S[i]+T[i]=2							
S	2	1	0	3	4	5	6	7

Swap(S[i], S[j])

i=1

T	2	1	3	2	1	3	2	1
		j S[i]+T[i]=2						
S	2	4	0	3	1	5	6	7

Swap(S[i], S[j])

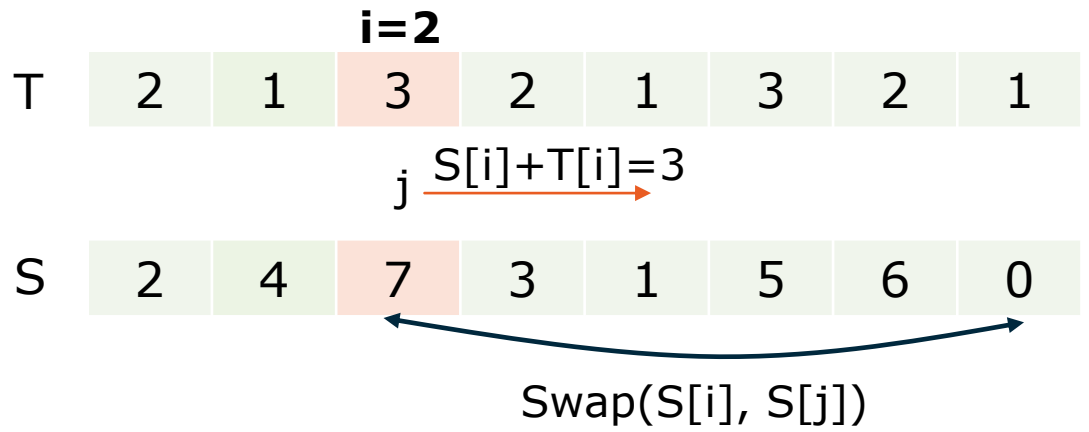
3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Xét ví dụ:** Cho bản rõ P = 001000110 (từ "BAG"), khoá K gồm 3 số 2, 1, 3 (N=3)
- **Giai đoạn khởi tạo:** S và T

S	0	1	2	3	4	5	6	7
T	2	1	3	2	1	3	2	1

```
# Khoi tao day so S va T
for i = 0 to 7 do
     S[i] = i;
     T[i] = K[i mod N];
next i
# Hoan vi day S
j=0
for i = 0 to 7 do
     j = (j + S[i] + T[i]) mod 8;
     Swap(S[i], S[j]);
next j
```

- **Hoán vị S**



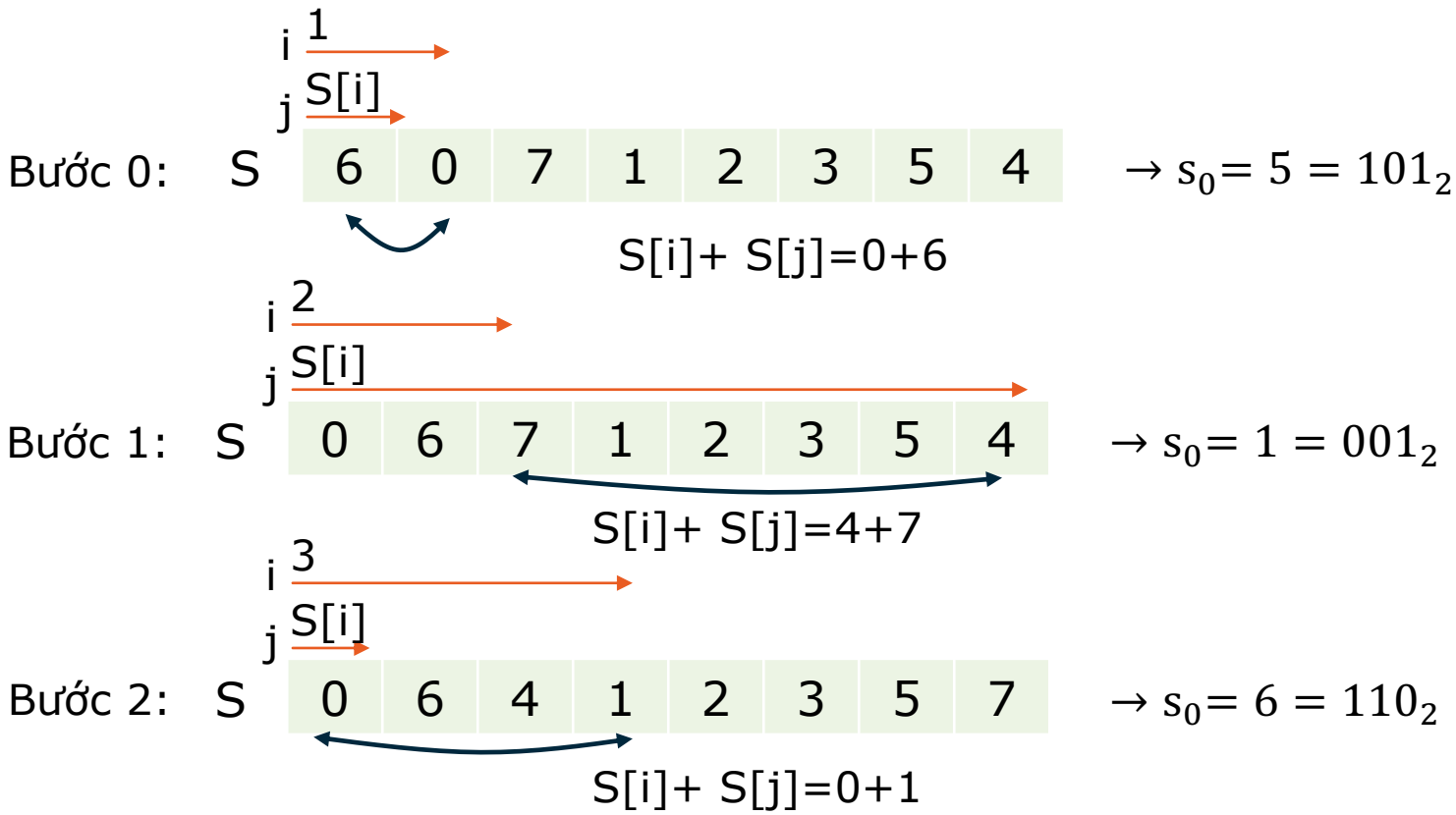
- Quá trình thực hiện đến khi i = 7
- Kết quả của dãy S:

S	6	0	7	1	2	3	5	4

3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Xét ví dụ:** Cho bản rõ P = 001000110 (từ "BAG"), khoá K gồm 3 số 2, 1, 3 (N=3)
- **Giai đoạn sinh số:**

```
i, j = 0;
while (true):
      i = (i + 1) mod 8;
      j = (j + S[i]) mod 8;
      Swap(S[i], S[j]);
      t = (S[i] + S[j]) mod 8;
      k = S[t];
end while
```



3.1 – MÃ DÒNG

- **Hệ mã TinyRC4**
- **Xét ví dụ:** Cho bản rõ P = 001000110 (từ "BAG"), khoá K gồm 3 số 2, 1, 3 (N=3)
- **Giai đoạn sinh số:**

```
i, j = 0;
while (true):
      i = (i + 1) mod 8;
      j = (j + S[i]) mod 8;
      Swap(S[i], S[j]);
      t = (S[i] + S[j]) mod 8;
      k = S[t];
end while
```

Bước 0:	S	6	0	7	1	2	3	5	4	$\rightarrow s_0 = 5 = 101_2$
Bước 1:	S	0	6	7	1	2	3	5	4	$\rightarrow s_0 = 1 = 001_2$
Bước 2:	S	0	6	4	1	2	3	5	7	$\rightarrow s_0 = 6 = 110_2$

- Vậy bản mã là **C = 001 000 110 ⊕ 101 001 110 = 100 001 000** (từ EBA)
- Giải mã: **P = C ⊕ S = 100 001 000 ⊕ 101 001 110 = 001 000 110**

3.1 – MÃ DÒNG

- **Hệ mã RC4 (Tổng quát)**
- Cơ chế hoạt động của RC4 cũng giống TinyRC4 với các đặc điểm sau:
  - Đơn vị mã hoá của RC4 là một byte 8 bít
  - Mảng S và T gồm 256 số nguyên 8 bít
  - Khoá K là một dãy gồm N số nguyên 8 bít với N có thể lấy các giá trị từ 1 đến 256
  - Bộ sinh số mỗi lần sinh ra một byte để sử dụng trong phép XOR
  - RC4 cũng sinh dãy số ngẫu nhiên, vì vậy RC4 đạt được mức độ an toàn cao.

3.1 – MÃ DÒNG

- **Hệ mã RC4 (Tổng quát)**
- Hai giai đoạn của RC4:

Giai đoạn khởi tạo

```
# Khoi tao day so S va T
for i = 0 to 255 do
     S[i] = i;
     T[i] = K[i mod N];
next i
# Hoan vi day S
j=0
for i = 0 to 255 do
     j = (j + S[i] + T[i]) mod 256;
     Swap(S[i], S[j]);
next j
```

Giai đoạn sinh số

```
i, j = 0;
while (true):
     i = (i + 1) mod 256;
     j = (j + S[i]) mod 256;
     Swap(S[i], S[j]);
     t = (S[i] + S[j]) mod 256;
     k = S[t];
end while
```

3.2 – MÃ KHỐI

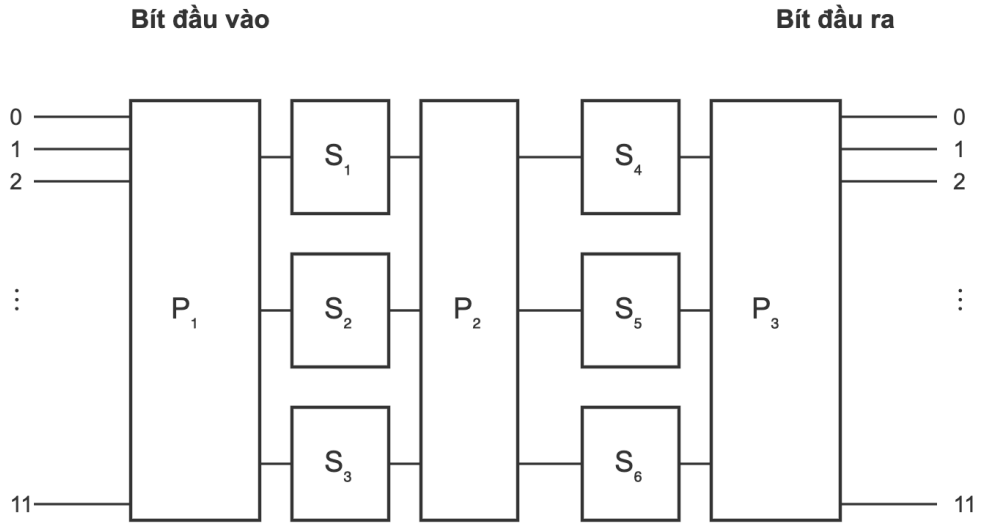
- **Mã khối an toàn lý tưởng**
  - Hạn chế của phép toán XOR: Chỉ cần biết một cặp khối bản rõ và bản mã, ta có thể suy ra được khóa và dùng khóa đó để giải các khối bản mã khác (known-plaintext attack)

  => Do đó, để chống phá mã trong trường hợp known-plaintext, ta chỉ có thể làm cho bản rõ P và bản mã C không có mối liên hệ toán học với nhau.

  - Nếu người phá mã chỉ biết một số cặp bản rõ - bản mã thì cũng không suy ra được các bản rõ cho các bản mã còn lại (nghĩa là: muốn phá mã thì phải biết được tất cả các cặp bản rõ và bản mã).
  - Nếu ta chọn kích thước của khối là 64 bít thì số dòng của bảng khóa là 2^{64} (một số rất lớn) nên người phá mã không thể nắm được tất cả các cặp bản rõ-bản mã của bảng khóa và trường hợp này được gọi là mã khối an toàn lý tưởng.

3.2 – MÃ KHỐI

- **Mạng SPN (Substitution-Permutation Network)**
  - Để xấp xỉ độ an toàn của mã khối lý tưởng, ta kết hợp hai hay nhiều mã hóa đơn giản lại với nhau để tạo thành một mã hóa tổng (product cipher) và mã hóa tổng này an toàn hơn nhiều so với các mã hóa thành phần.
  - Các mã hóa đơn giản thường là phép thay thế (substitution, S-box) và hoán vị (Permutation, P-box) nên người ta gọi mã hóa tổng này là mạng SPN (Substitution-Permutation Network).
  - Mô hình một mạng SPN (là kết hợp các S-box và P-box).



3.2 – MÃ KHỐI

- **Mô hình mã Feistel:**
  - Mô hình mã Feistel là một tiếp cận khác với mạng SPN. Mô hình này do Horst Feistel đề xuất, là sự kết hợp các phép thay thế và hoán vị.
  - Quá trình mã hóa của Feistel: Bản rõ P sẽ được biến đổi qua một số vòng để cho ra bản mã: $P \xrightarrow{K_1} C_1 \xrightarrow{K_2} C_2 \rightarrow \ldots \rightarrow \xrightarrow{K_{n-1}} C_n$
    - Trong đó bản rõ P và các bản mã C_i được chia thành nửa trái và nửa phải:

      $P = (L_0, R_0)$ $\qquad\qquad$ $C_i = (L_i, R_i)$ i = 1, 2, ...n
    - Quy tắc biến đổi các nửa trái phải này qua các vòng được thực hiện như sau:

      $L_i = R_{i-1}$ $\qquad\qquad$ $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$
    - K_i là một khóa con cho vòng thứ i. Khóa con này được sinh ra từ khóa K ban đầu theo một thuật toán sinh khóa con (key schedule): $K \rightarrow K_1 \rightarrow K_2 \rightarrow \ldots \rightarrow K_n$
    - F là một hàm mã hóa dùng chung cho tất cả các vòng. Hàm F đóng vai trò như là phép thay thế còn việc hoán đổi các nửa trái phải có vai trò hoán vị. Bản mã C được tính từ kết xuất của vòng cuối cùng: $C = C_n = (L_n, R_n)$

3.2 – MÃ KHỐI

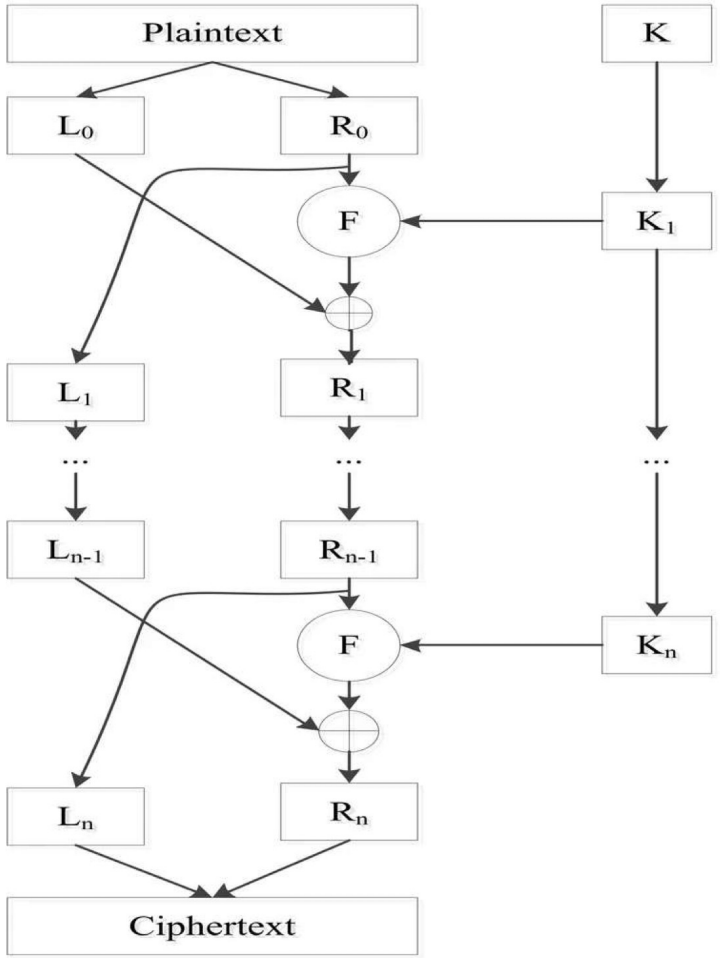
- **Mô hình mã Feistel:**

=> Sơ đồ tính toán của hệ mã Feistel (*hình bên*):

- Quá trình giải mã của Feistel:

Ta thực hiện qua các vòng theo thứ tự ngược lại như dưới đây:

$C \rightarrow L_n, R_n$
$R_{i-1} = L_i$ (theo mã hóa $L_i = R_{i-1}$)
$L_{i-1} = R_i \oplus F(R_{i-1}, K_i)$ (theo mã hóa $R_i = L_{i-1} \oplus F(R_{i-1}, K_i)$)
Cuối cùng được bản rõ: $P = (L_0, R_0)$.

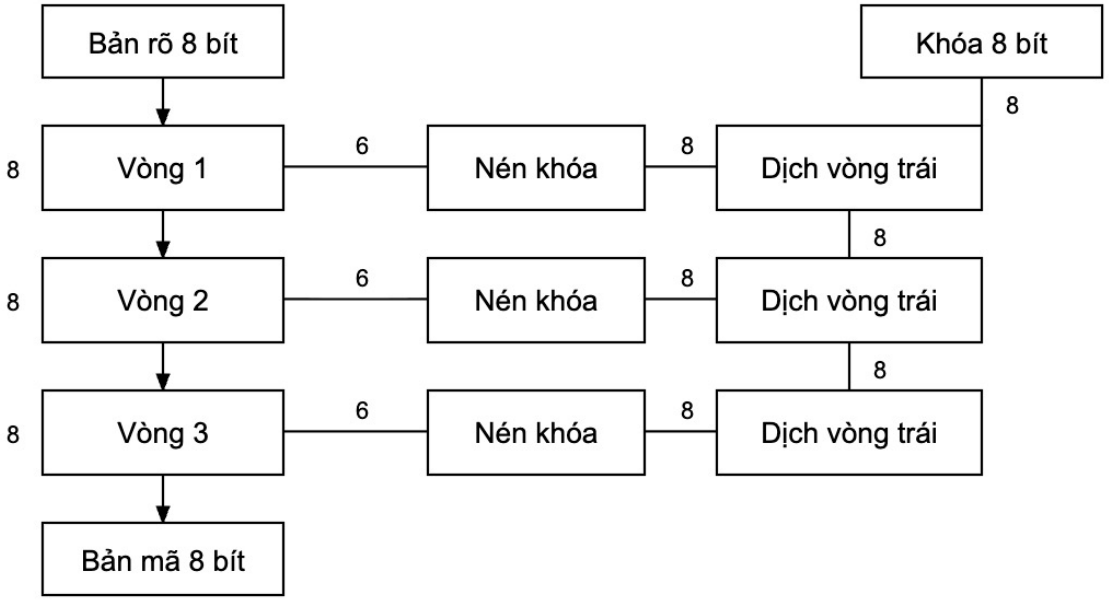


3.2 – MÃ KHỐI

- **Hệ mã TinyDES**

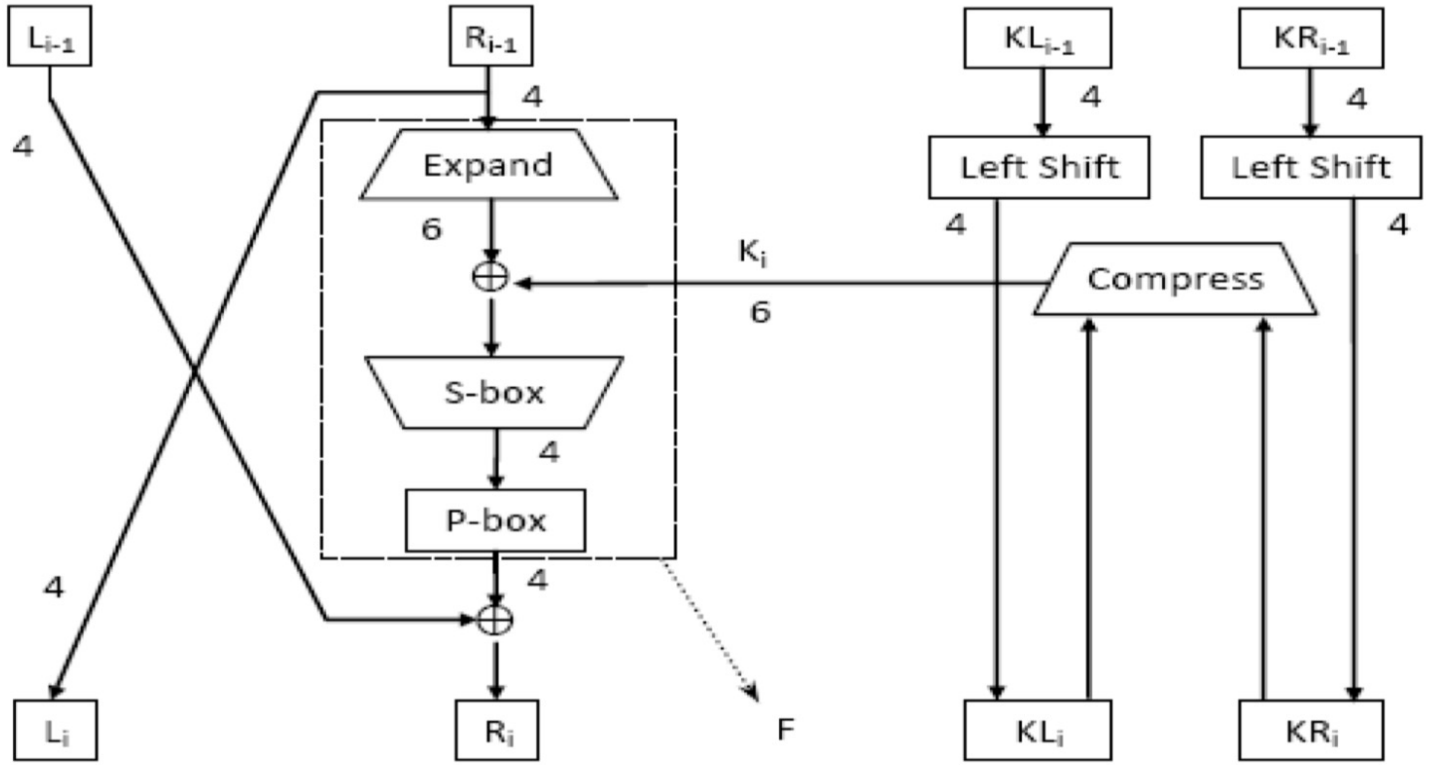
  Ta sẽ xem xét chi tiết mô hình thu nhỏ của hệ mã DES (Data Encryption Standard) là TinyDES.

  - Tính chất và các vòng Feistel của hệ mã TinyDES:
    - Tính chất của hệ mã Tiny DES:
      - Là mã thuộc hệ mã Feistel gồm 3 vòng.
      - Kích thước của khối là 8 bít.
      - Kích thước khóa là 8 bít.
      - Mỗi vòng của TinyDES dùng khóa con có kích thước 6 bít được trích ra từ khóa chính.
    - Các vòng Feistel của hệ mã TinyDES:
  - Hệ mã TinyDES gồm 2 phần: Các vòng Feistel và thuật toán sinh khóa con.



3.2 – MÃ KHỐI

- **Hệ mã TinyDES**
- Các vòng của TinyDES:
  - Cấu trúc 1 vòng của Feistel của TinyDES



3.2 – MÃ KHỐI

- **Hệ mã TinyDES**
- Các vòng của TinyDES:
- Mô tả các hàm:
  - *Expand:* gọi 4 bít của R_{i-1} là $b_0b_1b_2b_3$. Hàm Expand hoán vị và mở rộng 4 bít thành 6 bít cho ra kết quả: $b_2b_3b_1b_2b_1b_0$.
    Ví dụ: $R_0 = 0110 \Rightarrow \text{Expand}(R_0) = 101110$
  - *S-box:* Gọi $b_0b_1b_2b_3b_4b_5$ là 6 bít đầu vào của S-box, ứng với mỗi trường hợp của 6 bít đầu vào sẽ có 4 bít đầu ra. Việc tính các bít đầu ra dựa trên bảng sau:

b_0b_5	$b_1b_2b_3b_4$															
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

3.2 – MÃ KHỐI

- **Hệ mã TinyDES**
- Các vòng của TinyDES:
- Mô tả các hàm:
  - Hai bít b_0b_5 xác định thứ tự hàng, bốn bít $b_1b_2b_3b_4$ xác định thứ tự cột của bảng. Từ đó dựa vào bảng tính được 4 bít đầu ra. Để cho đơn giản, ta có thể viết lại bảng trên dưới dạng số thập lục phân.

b_0b_5	$b_1b_2b_3b_4$															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
1	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
2	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
3	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

  - Ví dụ: X = 101010. Tra bảng ta có S-box(X) = 0110. (Tức giá trị bằng 6 trong số thập lục phân)
  - *P-box:* thực hiện hoán vị 4 bít đầu $b_0b_1b_2b_3$ cho ra kết quả $b_2b_0b_3b_1$.

3.2 – MÃ KHỐI

- **Hệ mã TinyDES**
- Thuật toán sinh khoá con của TinyDES
  - Khóa K 8 bít ban đầu được chia thành 2 nửa trái và phải là KL_0 và KR_0 (mỗi nửa có kích thước 4 bít).
  - Tại vòng thứ nhất KL_0 và KR_0 được dịch vòng trái 1 bít để có được KL_1 và KR_1.
  - Tại vòng thứ hai KL_1 và KR_1 được dịch vòng trái 2 bít để có được KL_2 và KR_2.
  - Tại vòng thứ 3 KL_2 và KR_2 được dịch vòng trái 1 bít để có KL_3 và KR_3.
  - Cuối cùng khóa K_i của mỗi vòng được tạo ra bằng cách hoán vị và nén (compress) 8 bít của KL_i và $KR_i(k_0k_1k_2k_3k_4k_5k_6k_7)$ thành kết quả gồm 6 bít: $k_5k_1k_3k_2k_7k_0$.
  - **BTVD:** Hãy mã hóa bản rõ P = 01011100 (5C) (ở hệ cơ số 16) với khóa K = 10011010 ?

3.2 – MÃ KHỐI

Hệ mã TinyDES

- **BTVD:** Hãy mã hóa bản rõ P = 01011100 (5C) (ở hệ cơ số 16) với khóa K = 10011010 ?
- **Giải:** $L_0 = 0101$, $R_0 = 1100$, $KL_0 = 1001$, $KR_0 = 1010$
- **Vòng 1:**

  - $L_1 = R_0 = 1100$, $Expand(R_0) = 001011$
  - $KL_1 = KL_0 << 1 = 0011$, $KR_1 = KR_0 << 1 = 0101$
  - $K_1 = Compress(KL_1KR_1) = 101110$
  - $Expand(R_0) \oplus K_1 = 100101$
  - S-box(100101) = 1000
  - $F_1 = P\text{-box}(1000) = 0100$
  - $R_1 = L_0 \oplus F_1 = 0001$

- **Vòng 2:**

  - $L_2 = R_1 = 0001$, $Expand(R_1) = 010000$
  - $KL_2 = KL_1 << 2 = 1100$, $KR_2 = KR_1 << 2 = 0101$
  - $K_2 = Compress(KL_2KR_2) = 110011$
  - $Expand(R_1) \oplus K_2 = 100011$
  - S-box(100011) = 1100
  - $F_2 = P\text{-box}(1100) = 0101$
  - $R_2 = L_1 \oplus F_2 = 1001$

Vòng 3:

- $L_3 = R_2 = 1001$, $Expand(R_2) = 010001$
- $KL_3 = KL_2 << 1 = 1001$, $KR_3 = KR_2 << 1 = 1010$
- $K_3 = Compress(KL_3KR_3) = 001001$
- $Expand(R_2) \oplus K_3 = 011000$
- S-box(011000) = 0101
- $F_3 = P\text{-box}(0101) = 0011$
- $R_3 = L_2 \oplus F_3 = 0010$

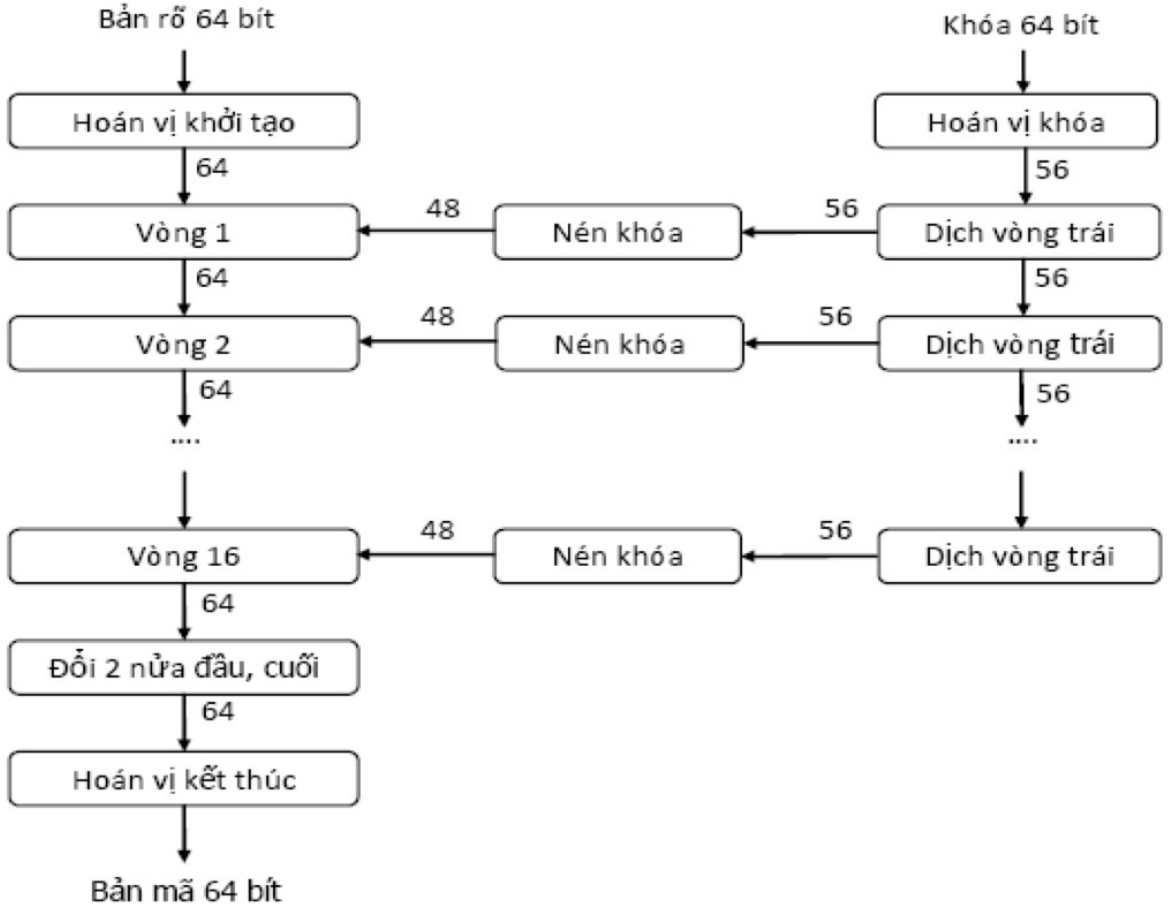
- **Kết quả C = L_3R_3 = 1001.0010 (hệ thập lục phân: 92)**

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
  - Tính chất và các vòng Feistel của hệ mã DES:
    - Tính chất của hệ mã DES:
      - Là mã thuộc hệ mã Feistel gồm 16 vòng, ngoài ra DES có thêm một hoán vị khởi tạo trước khi vào vòng 1 và một hoán vị kết thúc sau vòng 16.
      - Kích thước của khối là 64 bít. Ví dụ bản rõ "meetmeafterthetogaparty" (biểu diễn theo mã ASCII) thì mã DES sẽ mã hóa làm 3 lần (mỗi lần 8 chữ cái (64 bít) là: meetmeaf - tertheto - gaparty).
      - Kích thước khóa là 56 bít.
      - Mỗi vòng của DES dùng khóa con có kích thước 48 bít được trích ra từ khóa chính.

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
  - Tính chất và các vòng Feistel của hệ mã DES:
    - Các vòng Feistel của hệ mã DES
    - Hệ mã DES gồm 3 phần:
      - Hoán vị khởi tạo và hoán vị kết thúc;
      - 85 vòng Feistel;
      - Thuật toán sinh khóa con



3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Hoán vị khởi tạo và hoán vị kết thúc:
  - Đánh số các bít của khối 64 bít theo thứ tự từ trái sang phải là 0, 1,…, 62, 63 là: b0 b1 b2 …b62 b63
  - Hoán vị khởi tạo hoán đổi các bít theo quy tắc (hình trái):
  - Hoán vị kết thúc hoán đổi các bít theo quy tắc (hình phải):

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
56	48	40	32	24	16	8	0
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6

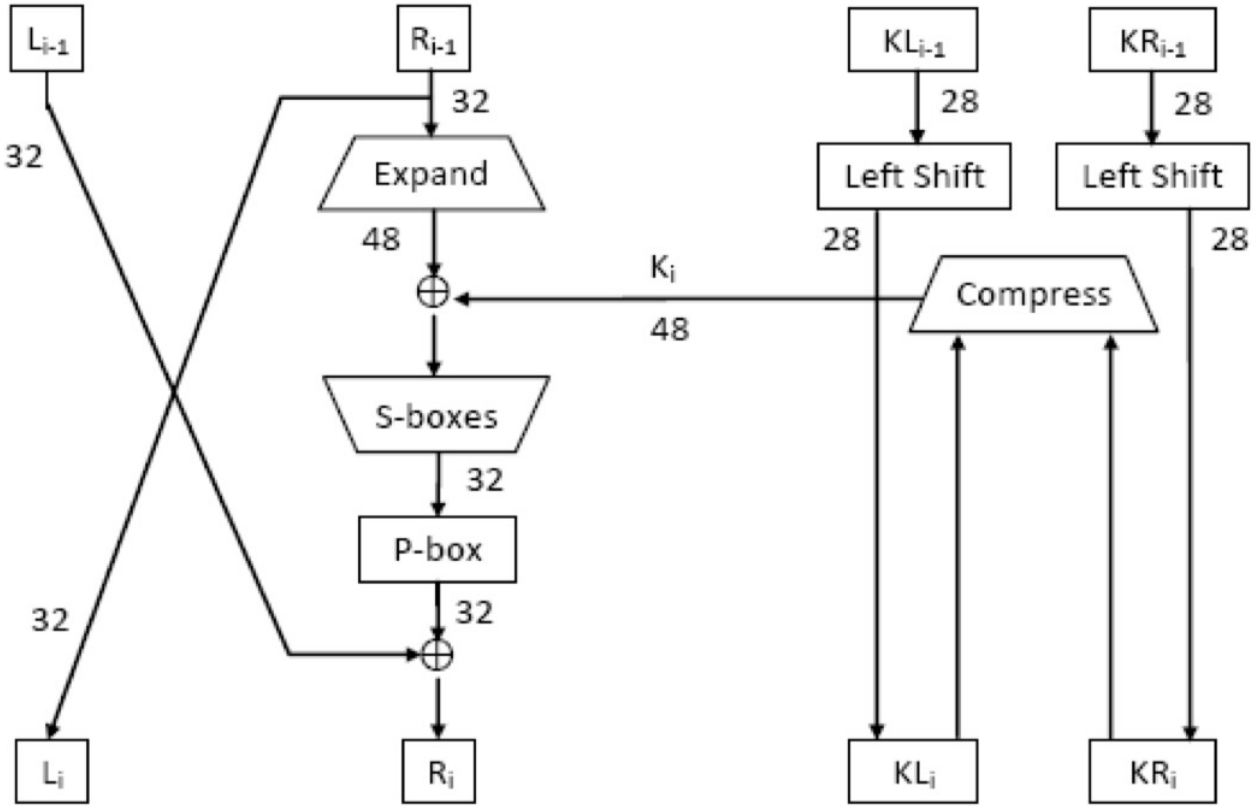
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25
32	0	40	8	48	16	56	24

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Các vòng của DES:
  - Cấu trúc 1 vòng của Feistel của DES:
  - Trong DES, hàm F của Feistel là:

$F(R_{i-1}, K_i) = P\text{-box}(S\text{-boxes}(\text{Expand}(R_{i-1}) \oplus K_i))$

trong đó: hàm Expand vừa mở rộng vừa hoán vị R_{i-1} từ 32 bít lên 48 bít, hàm S-boxes nén 48 bít lại còn 32 bít, hàm P-box là một hoán vị 32 bít.



3.2 – MÃ KHỐI

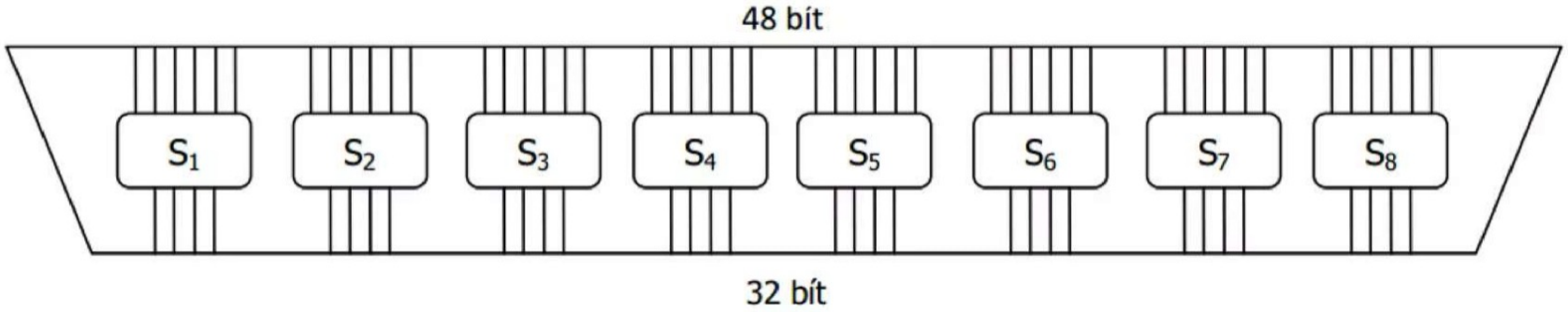
- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Các vòng của DES:
  - Mô tả các hàm:

+ Hàm Expand: Đánh số các bít của R i-1 theo thứ tự từ trái sang phải là 0, 1, 2, …, 31. Hàm Expand thực hiện vừa hoán vị vừa mở rộng 32 bít thành 48 bít theo quy tắc (hình bên):

31	0	1	2	3	4
3	4	5	6	7	8
7	8	9	10	11	12
11	12	13	14	15	16
15	16	17	18	19	20
19	20	21	22	23	24
23	24	25	26	27	28
27	28	29	30	31	0

48 bit

+ Hàm S-boxes: Biến đổi một số 48 bít thành một số 32 bít. Để giảm kích thước của bảng tra cứu, S-boxes được chia thành 8 hàm S-box con (mỗi hàm biến đổi số 6 bít thành số 4 bít (hình bên dưới))



3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Các vòng của DES:
  - Mô tả các hàm (tiếp):

+ Hàm S-boxes (tiếp…): Hàm S-box đầu tiên, hộp S1, giống S-box của TinyDES như dưới đây. Còn các hộp S-box còn lại (Xem ở phần ***Phụ Lục 1***)

b_0b_5	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
00	1110	0100	1101	0001	0010	1111	1011	1000	0011	1010	0110	1100	0101	1001	0000	0111
01	0000	1111	0111	0100	1110	0010	1101	0001	1010	0110	1100	1011	1001	0101	0011	1000
10	0100	0001	1110	1000	1101	0110	0010	1011	1111	1100	1001	0111	0011	1010	0101	0000
11	1111	1100	1000	0010	0100	1001	0001	0111	0101	1011	0011	1110	1010	0000	0110	1101

+ Hàm P-box: Thực hiện hoán vị 32 bít đầu vào theo quy tắc:

15	6	19	20	28	11	27	16
0	14	22	25	4	17	30	9
1	7	23	13	31	26	2	8
18	12	29	5	21	10	3	24

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Thuật toán sinh khoá con của DES:
  - Khóa K 64 bít ban đầu được rút trích và hoán vị thành một khóa 56 bít (chỉ sử dụng 56 bít) theo quy tắc (hình bên):

56	48	40	32	24	16	8
0	57	49	41	33	25	17
9	1	58	50	42	34	26
18	10	2	59	51	43	35
62	54	46	38	30	22	14
6	61	53	45	37	29	21
13	5	60	52	44	36	28
20	12	4	27	19	11	3

56 bít

  - Khóa 56 bít này được chia thành 2 nửa trái phải KL_0 và KR_0 (mỗi nửa có kích thước 28 bít).
    Tại vòng thứ i (i = 1, 2, 3,...,16), KL_{i-1} và KR_{i-1} được dịch vòng trái r_i bít để có được KL_i và KR_i, với r_i được định nghĩa:
    r_i= { 1 nếu i ∈ {1,2,9,16}; 2 với những i khác }
  - Cuối cùng, khóa K_i của mỗi vòng được tạo ra bằng cách hoán vị và nén 56 bít của KL_i và KR_i thành 48 bít theo quy tắc:

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

48 bít

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Phụ lục 1: Chi tiết các S-box của mã hoá DES

DES S-box 1

b_0b_5 \ $b_1b_2\,b_3b_4$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	E	4	D	1	2	F	B	8	3	A	6	C	5	9	0	7
1	0	F	7	4	E	2	D	1	A	6	C	B	9	5	3	8
2	4	1	E	8	D	6	2	B	F	C	9	7	3	A	5	0
3	F	C	8	2	4	9	1	7	5	B	3	E	A	0	6	D

DES S-box 2

b_0b_5 \ $b_1b_2\,b_3b_4$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	F	1	8	E	6	B	3	4	9	7	2	D	C	0	5	A
1	3	D	4	7	F	2	8	E	C	0	1	A	6	9	B	5
2	0	E	7	B	A	4	D	1	5	8	C	6	9	3	2	F
3	D	8	A	1	3	F	4	2	B	6	7	C	0	5	E	9

DES S-box 3

b_0b_5 \ $b_1b_2\,b_3b_4$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	A	0	9	E	6	3	F	5	1	D	C	7	B	4	2	8
1	D	7	0	9	3	4	6	A	2	8	5	E	C	B	F	1
2	D	6	4	9	8	F	3	0	B	1	2	C	5	A	E	7
3	1	A	D	0	6	9	8	7	4	F	E	3	B	5	2	C

DES S-box 4

b_0b_5 \ $b_1b_2\,b_3b_4$	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	7	D	E	3	0	6	9	A	1	2	8	5	B	C	4	F
1	D	8	B	5	6	F	0	3	4	7	2	C	1	A	E	9
2	A	6	9	0	C	B	7	D	F	1	3	E	5	2	8	4
3	3	F	0	6	A	1	D	8	9	4	5	B	C	7	2	E

3.2 – MÃ KHỐI

- **Hệ mã DES (Data Encryption Standard) (tổng quát)**
- Phụ lục 1: Chi tiết các S-box của mã hoá DES

DES S-box 5

b_0b_5	$b_1b_2\,b_3b_4$															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	2	C	4	1	7	A	B	6	8	5	3	F	D	0	E	9
1	E	B	2	C	4	7	D	1	5	0	F	A	3	9	8	6
2	4	2	1	B	A	D	7	8	F	9	C	5	6	3	0	E
3	B	8	C	7	1	E	2	D	6	F	0	9	A	4	5	3

DES S-box 6

b_0b_5	$b_1b_2\,b_3b_4$															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	C	1	A	F	9	2	6	8	0	D	3	4	E	7	5	B
1	A	F	4	2	7	C	9	5	6	1	D	E	0	B	3	8
2	9	E	F	5	2	8	C	3	7	0	4	A	1	D	B	6
3	4	3	2	C	9	5	F	A	B	E	1	7	6	0	8	D

DES S-box 7

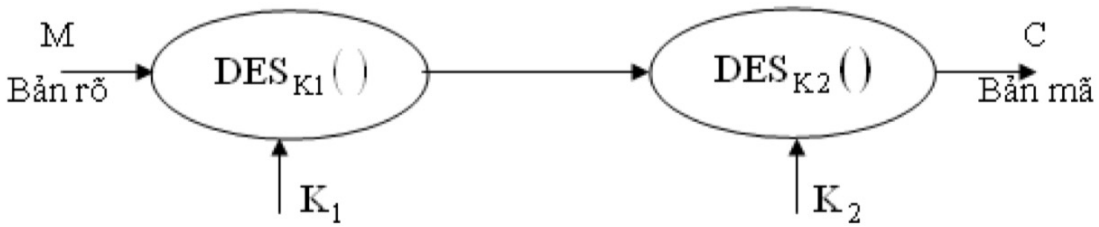
b_0b_5	$b_1b_2\,b_3b_4$															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	4	B	2	E	F	0	8	D	3	C	9	7	5	A	6	1
1	D	0	B	7	4	9	1	A	E	3	5	C	2	F	8	6
2	1	4	B	D	C	3	7	E	A	F	6	8	0	5	9	2
3	6	B	D	8	1	4	A	7	9	5	0	F	E	2	3	C

DES S-box 8

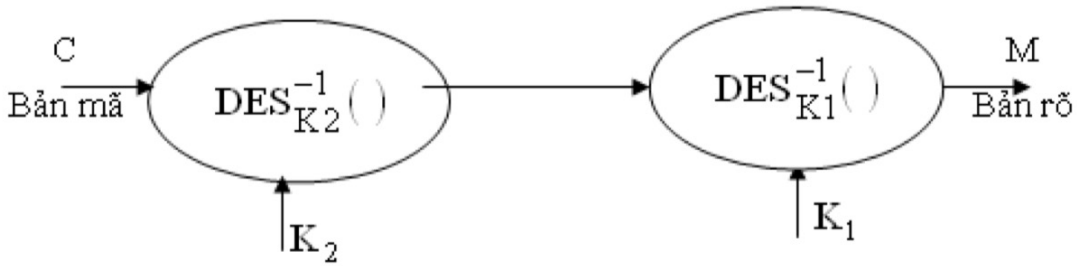
b_0b_5	$b_1b_2\,b_3b_4$															
	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	D	2	8	4	6	F	B	1	A	9	3	E	5	0	C	7
1	1	F	D	8	A	3	7	4	C	5	6	B	0	E	9	2
2	7	B	4	1	9	C	E	2	0	6	A	D	F	3	5	8
3	2	1	E	7	4	A	8	D	F	C	9	0	3	5	6	B

3.2 – MÃ KHỐI

- **Một số phương pháp mã khối khác**
- **Double DES (DDES):**
  - Đặc điểm: Dùng DES 2 lần với 2 khóa K_1, K_2 khác nhau:
  - Mã hóa: $C = DES_{K2}[DES_{K1}(M)]$
  - Giải mã: $M = DES^{-1}{}_{K1}[DES^{-1}{}_{K2}(C)]$ (với: M là bản rõ, DES là hàm mã, DES^{-1} là hàm giải mã)
  - đồ (hình bên):



a. Mã hóa DES bội hai

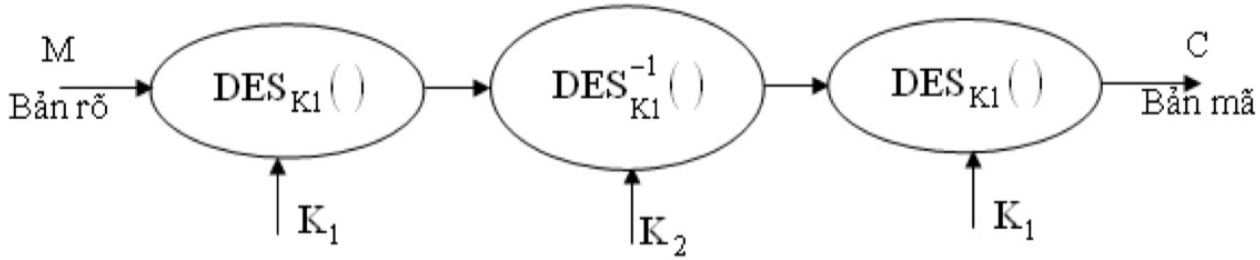


3.2 – MÃ KHỐI

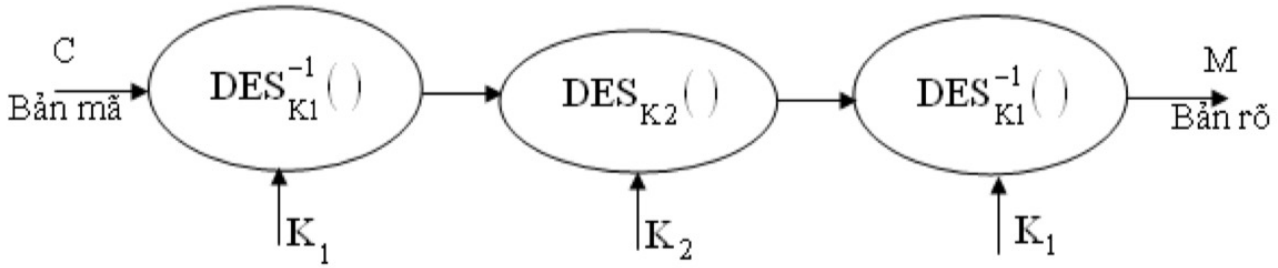
- **Một số phương pháp mã khối khác**
- **Triple DES (TDES):**
  - Đặc điểm: Dùng DES 3 lần với 2 khóa K_1, K_2 khác nhau:
    - Mã hóa:

      $C = DES_{K1}[DES^{-1}_{K2}[DES_{K1}(M)]]$
    - Giải mã:

      $M = DES^{-1}_{K1}[DES_{K2}[DES^{-1}_{K1}(C)]]$
  - Sơ đồ (hình bên):



a. Mã hóa TDES với hai khóa



b. Giải mã TDES với hai khóa

3.2 – MÃ KHỐI

- **Một số phương pháp mã khối khác**
- **Chuẩn mã hóa nâng cao (AES) (Advanced Encryption Standard):**
  - Đặc điểm 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 nâng cao AES - Rijndael: Rijndael được chọn là chuẩn mã nâng cao, được thiết kế bởi Rijmen – Daemen (Bỉ), có các đặc trưng:

    + Có 128/192/256 bit khoá và 128 bit khối dữ liệu.

    + Lặp (khác đôi chút so với Feistel) là: Chia dữ liệu thành 4 nhóm (mỗi nhóm 4 byte), thao tác trên cả khối mỗi vòng.

    + Xử lý khối dữ liệu 128 bit như 4 nhóm của 4 byte: 128 = 4*4*8 bit (mỗi nhóm nằm trên 1 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.

    + Có tùy chọn 9/11/13 vòng, trong đó mỗi vòng gồm: Phép thế byte (dùng 1 hộp 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 thực hiện với phép XOR và bảng tra cứu nên rất nhanh và hiệu quả.
  - Các mô hình ứng dụng mã khối *: Electronic Codebook – ECB, Counter – CTR, Cipher Block Chaining – CBC, Output Feedback – OFB, Cipher Feedback – CFB 95 (Xem thêm trong giáo trình và tài liệu tham khảo).

4 – TÍNH CHỨNG THỰC VÀ KHÔNG TỪ CHỐI

- **Tính chứng thực (Authentication)**
- Mã hóa đối xứng có thể chống lại các hình thức tấn công:
  - Mạo danh: Kẻ thứ ba mạo danh, nghĩa là: A gửi thông điệp (bản rõ) cho B mà B lại nghĩ rằng thông điệp đó là do C gửi.
  - Sửa đổi nội dung thông điệp: Nếu A chặn được bản mã của B và sửa đổi thông điệp thì xác suất để bản rõ là văn bản có nghĩa cũng rất nhỏ và C biết được bản rõ đã bị sửa đổi.
  - Phát lại thông điệp (replay attack): A gửi bản mã cho B, B nhận được và giải mã để có bản rõ. Tuy nhiên, C chặn được bản mã và sau đó mạo danh A gửi bản mã cho B thêm một lần nữa. B giải mã và cũng có được bản rõ => Như vậy, B nhận được cùng một bản rõ 2 lần. Tại lần thứ 2, B không có cơ sở xác định là A muốn gửi lại hay là do A gửi.

  => Do đó, để đảm bảo tính chứng thực, dùng mã chứng thực thông điệp (MAC).

4 – TÍNH CHỨNG THỰC VÀ KHÔNG TỪ CHỐI

- **Tính không từ chối(Non-repudiation)**
- Mã hóa đối xứng có thể chống lại các hình thức tấn công nhưng mã hóa đối xứng lại không thực hiện được tính không từ chối do tính bí mật của khóa (vì khóa K bí mật có hai người biết nên nếu K bị tiết lộ thì không có cơ sở để quy trách nhiệm cho A hay B trong 2 người làm lộ khóa. Do đó, A có thể từ chối là đã gửi thông điệp.
- => Để khắc phục: Nghiên cứu và phát triển các hệ mã hóa khóa công khai

5 – TRAO ĐỔI KHOÁ BÍ MẬT BẰNG TRUNG TÂM PHÂN PHỐI KHOÁ

- **Vấn đề**

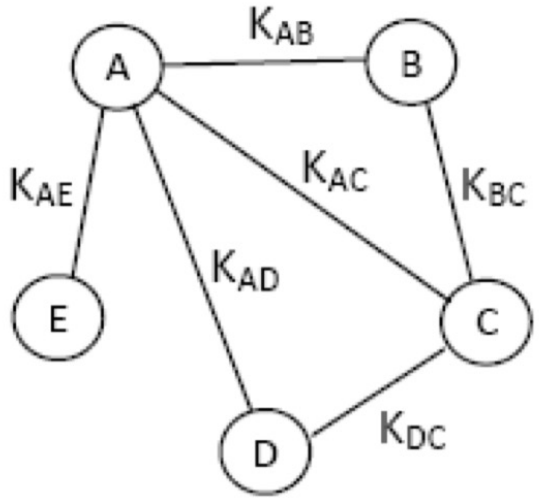
- Giả sử có N người sử dụng, trao đổi dữ liệu bằng mã hóa đối xứng, mỗi cặp người sử dụng cần có một khóa bí mật riêng, dẫn đến cần có N(N-1)/2 khóa bí mật. Việc thiết lập các khóa bí mật này sẽ gây ra khó khăn cho các người sử dụng vì mỗi người cần thiết lập N-1 khóa => Để khắc phục, sử dụng phương pháp trao đổi khóa bằng trung tâm phân phối khóa (Key Distribution Center – KDC).
- => Trong mô hình sử dụng KDC, mỗi người sử dụng chỉ cần có một khóa bí mật với KDC, còn khóa dùng để trao đổi dữ liệu giữa các người sử dụng sẽ do KDC cung cấp.

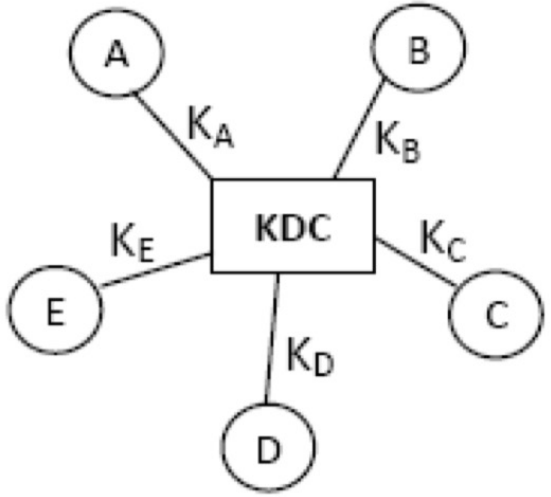
5 – TRAO ĐỔI KHOÁ BÍ MẬT BẰNG TRUNG TÂM PHÂN PHỐI KHOÁ

- **Quá trình thiết lập khóa chung K_{AB}:** Giả sử A có khóa bí mật K_A với KDC và B có khóa bí mật K_B với KDC. Bây giờ, A muốn trao đổi dữ liệu với B, gồm các bước:
  - B1: A gửi yêu cầu muốn trao đổi dữ liệu với B cho KDC.
  - B2: KDC tạo một khóa bí mật K_{AB} và mã hóa thành hai bản mã.

  Một bản mã được mã hóa bằng khóa bí mật của A là $E(K_{AB}, K_A)$ và 1 bản mã được mã hóa bằng khóa bí mật của B là $E(K_{AB}, K_B)$.
  - B3: A giải mã $E(K_{AB}, K_A)$ để có K_{AB}.
  - B4: KDC gửi $E(K_{AB}, K_B)$ cho B, B giải mã để có được K_{AB}.
  - B5: A và B trao đổi dữ liệu qua khóa bí mật K_{AB}.

=> Như vậy, khóa K_{AB} chỉ có KDC, A và B biết. Trách nhiệm của KDC là giữ bí mật khóa này. A và B dùng khóa K_{AB} để mã hóa dữ liệu. Khi kết thúc quá trình thì khóa K_{AB} bị hủy.





TÓM TẮT NỘI DUNG CHƯƠNG 2

- Trao đổi/thảo luận nhóm
- Bài tập/Thực hành
- Tóm tắt nội dung
- Câu hỏi và bài tập ôn tập