

MẬT MÃ ỨNG DỤNG TRONG AN TOÀN THÔNG TIN

(Applied Cryptography In Secure Information System)

ThS. Bùi Hữu Đông
buihuudong19@gmail.com
0903.82.36.46



Học viện Kỹ thuật Mật mã, khoa An toàn thông tin

Ngày 31 tháng 10 năm 2020

Buổi 03

MÃ HÓA ĐỐI XỨNG (Symmetric Cryptography)

MÃ HÓA ĐỐI XỨNG HIỆN ĐẠI (Modern Symmetric Cryptography)

Tổng quan

• Nhận xét:

- Đối tượng của phương pháp mã hóa cổ điển là các bản tin ngôn ngữ, một đơn vị mã hóa là các chữ cái và có thể áp dụng các hình thức thay thế (đơn hoặc đa bảng) hoặc hoán vị
- Với sự phát triển của máy tính, bản thông tin cần mã hóa đa dạng hơn như dữ liệu hình ảnh, âm thanh... mà dữ liệu này được biểu diễn trong máy tính dưới dạng số nhị phân

Thí dụ:

Bản tin: attack

Mã ASCII: 97 116 116 97 99 107

Biểu diễn nhị phân: 01100001 01110100 01110100 01100001 01100011 01101011

Tổng quan

- Dù thông tin tồn tại dưới dạng nhị phân nhưng kẻ thám mã vẫn có thể phá mã
- Mã hóa hiện đại quan tâm tới vấn đề *chống phá mã* khi biết trước bản rõ

Mã hóa đối xứng hiện đại

- Thí dụ:

- Giả sử ta có bảng mã tương ứng 8 chữ cái với số nhị phân 3 bit:

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

- Ta có bản rõ $P = \text{"HEAD"} = \text{"111100000011"}$ và khóa $K = \text{"0101"}$.

Mã hóa đối xứng hiện đại

- Để mã hóa ta dùng phép toán XOR, cụ thể:

bản rõ: 1111 0000 0011 (head)

khóa: 0101 0101 0101

bản mã: 1010 0101 0110 (FBCG)

- ▷ Trong phép mã hóa trên ta không mã hóa theo từng chữ cái mà là một khối gồm 4 bit.
- ▷ Để giải mã ta thực hiện phép XOR ngược giữa bản mã và khóa với nhau.

Mã hóa đối xứng hiện đại

- Thực hiện mã hóa bằng XOR có nhược điểm:
 - ▷ Khóa lặp lại nhiều lần (giống mã hóa Vigenere) để khắc phục người ta dùng bộ sinh số ngẫu nhiên để tạo khóa dài (giả lập One-Time Pad) đây là cơ sở cho *mã dòng (stream cipher)*
 - ▷ Một khối được mã hóa bằng XOR không an toàn vì chỉ cần biết *một cặp khối* - bản rõ & bản mã, kẻ phá mã có thể tìm được khóa => cần phương pháp mã hóa phức tạp hơn và đây là cơ sở của *mã khối (block cipher)*

MÃ DÒNG (Stream Cipher)

Mã dòng - Stream Cipher

• Đặc điểm:

- Kích thước một đơn vị mã hóa gồm k bit. Bản rõ chia thành các đơn vị mã hóa, có nghĩa $P \rightarrow p_0, p_1 p_2 \dots p_{n-1}$ với mỗi $p_i : k$ 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:

$$SC(K) \rightarrow Z = z_0 z_1 z_2 \dots z_{n-1} \text{ với } z_i \text{ là } k \text{ bit.}$$

- Mỗi số ngẫu nhiên được XOR với đơn vị mã hóa của bản rõ để được bản mã:

$$c_0 = p_0 \oplus z_0, c_1 = p_1 \oplus z_1, \dots, C = c_0 c_1 c_2 \dots c_{n-1}$$

- Quá trình giải mã được làm ngược lại. Tức thực hiện phép XOR giữa khóa và bản mã

Mã dòng - Stream Cipher

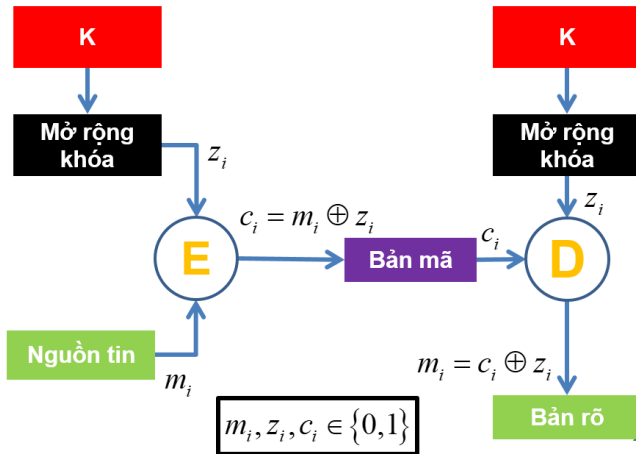
- Thí dụ mã hóa dưới đây có phải mã dòng?

bản rõ: 1111 0000 0011 (head)

khóa: 0101 0101 0101

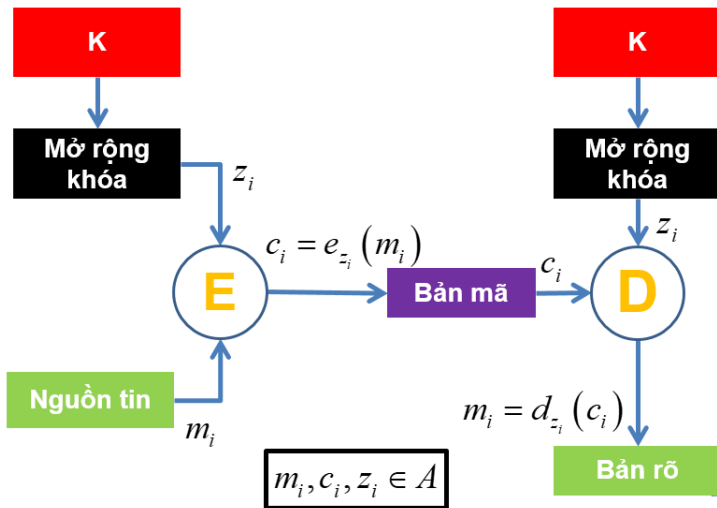
bản mã: 1010 0101 0110 (FBCG)

Mã dòng - Stream Cipher



Theo bạn mã hóa dạng mã dòng quan trọng nhất là gì?

Mã dòng - Stream Cipher



Mã dòng - Stream Cipher

- Phần mở rộng khóa:

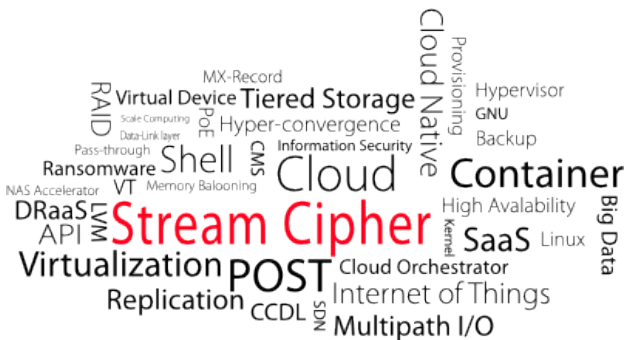
- Là bộ sinh số giả ngẫu nhiên (PRNG)
- Là quan trọng nhất
- Quyết định độ an toàn của mã dòng

- Phân loại:

- z_i chỉ phụ thuộc vào khóa K thì gọi là mã dòng đồng bộ
- z_i phụ thuộc $c_{i-n}, c_{i-n+1}, \dots, c_{i-1}$: gọi là mã dòng tự đồng bộ

Ứng dụng của Mã dòng

- Mã dòng có tốc độ cao do cơ chế sinh dòng khóa và mã hóa khá đơn giản so với mã khối
- Mã dòng có thể mã hóa lượng dữ liệu bất kì, không cần phải chờ đợi kích thước đầu vào đạt đến giá trị nhất định như mã khối



Phương pháp Mã dòng A5/1

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

Để hiểu được A5/1 ta tìm hiểu dạng thu nhỏ là **TinyA5/1**. Với cơ chế như sau:

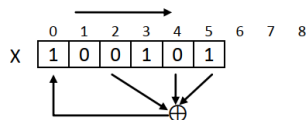
- Bộ sinh số gồm 3 thanh ghi X, Y, Z .
 - X gồm 6 bit (x_0, x_1, \dots, x_5)
 - Y gồm 8 bit (y_0, y_1, \dots, y_7)
 - Z gồm 9 bit (z_0, z_1, \dots, z_8)
- Khóa K có chiều dài 23 bit và được phân bổ vào các thanh ghi $K \rightarrow XYZ$

Phương pháp Mã dòng A5/1

- Các thanh ghi X,Y,Z được biến đổi theo 3 quy tắc:

1) **Quay X** gồm các thao tác:

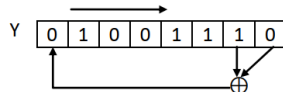
- $t = x_2 \oplus x_4 \oplus x_5$
- $x_j = x_{j-1}$ với $j = 5, 4, 3, 2, 1$
- $x_0 = t$



Ví dụ: giả sử X là 100101, dẫn đến $t = 0 \oplus 0 \oplus 1 = 1$, vậy sau khi quay giá trị của X là 110010.

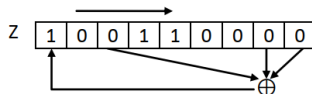
2) **Quay Y**: tương tự như quay X, quay Y là như sau:

- $t = y_6 \oplus y_7$
- $y_j = y_{j-1}$ với $j = 7, 6, 5, \dots, 1$
- $y_0 = t$



3) **Quay Z**:

- $t = z_2 \oplus z_7 \oplus z_8$
- $z_j = z_{j-1}$ với $j = 8, 7, 6, \dots, 1$
- $z_0 = t$



Phương pháp Mã dòng A5/1

- Cho 3 bit x, y, z ta định nghĩa hàm $maj(x, y, z)$, nếu trong 3 bit này có từ 2 bit 0 trở lên thì hàm trả về 0, ngược lại trả về 1.
- Tại bước sinh số thứ i , các phép tính sau được thực hiện:
 - $m = maj(x_1, y_3, z_3)$
 - 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

Và bit được sinh ra là: $s_i = x_5 \oplus y_7 \oplus z_8$

Bit s_i được XOR với bit thứ i trong bản rõ để có được bit thứ i trong bản mã theo quy tắc của mã dòng.

Phương pháp Mã dòng A5/1

- Thí dụ:

Mã hóa bản rõ $P = 111$ (chữ h) với khóa K là
100101.01001110.100110000

Ban đầu giá trị của các thanh ghi X, Y, Z là:

$$X = 1\underline{0}0101$$

$$Y = 010\underline{0}1110$$

$$Z = 100\underline{1}10000$$

Bước 0: $x_1 = 0, y_3 = 0, z_3 = 1 \rightarrow m = 0 \rightarrow$ quay X, quay Y

$$X = 1\underline{1}0010$$

$$Y = 101\underline{0}0111 \quad \rightarrow s_0 = 0 \oplus 1 \oplus 0 = 1$$

$$Z = 100\underline{1}10000$$

Phương pháp Mã dòng A5/1

○ Thí dụ <tiếp>:

Bước 1: $x_1 = 1, y_3 = 0, z_3 = 1 \rightarrow m = 1 \rightarrow$ quay X, quay Z

$$X = 1\underline{1}1001$$

$$Y = 101\underline{0}0111$$

$$\rightarrow s_1 = 1 \oplus 1 \oplus 0 = 0$$

$$Z = 010\underline{0}11000$$

Bước 2: $x_1 = 1, y_3 = 0, z_3 = 0 \rightarrow m = 0 \rightarrow$ quay Y, quay Z

$$X = 111001$$

$$Y = 01010011$$

$$\rightarrow s_2 = 1 \oplus 1 \oplus 0 = 0$$

$$Z = 001001100$$

Vậy bản mã là $C = 111 \oplus 100 = 011$ (chữ D)

Phương pháp Mã dòng A5/1

• Phương pháp mã dòng A5/1:

Về nguyên tắc, bộ sinh số **A5/1** hoạt động như **TinyA5/1** nhưng các thanh ghi có size lần tương ứng X , Y , Z là 19, 22 và 23 bit. Các bước quay cụ thể như sau:

1) Quay X:

- $t = x_{13} \oplus x_{16} \oplus x_{17} \oplus x_{18}$
- $x_j = x_{j-1}$ với $j = 18, 17, 16, \dots, 1$
- $x_0 = t$

2) Quay Y:

- $t = y_{20} \oplus y_{21}$
- $y_j = y_{j-1}$ với $j = 21, 20, 19, \dots, 1$
- $y_0 = t$

3) Quay Z:

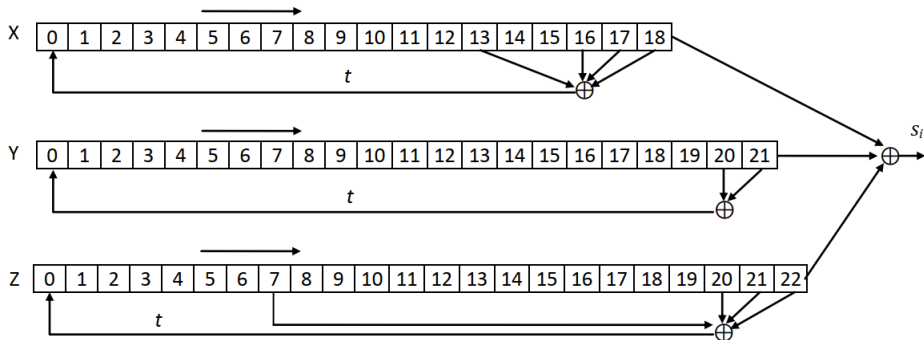
- $t = z_7 \oplus z_{20} \oplus z_{21} \oplus z_{22}$
- $z_j = z_{j-1}$ với $j = 22, 21, 20, \dots, 1$
- $z_0 = t$

Phương pháp Mã dòng A5/1

Hàm *maj* được tính trên 3 bit x_8, y_{10}, z_{10} , với bit sinh ra là:

$$S_i = x_{18} \oplus y_{21} \oplus z_{22}$$

Mô hình sinh mã dòng của A5/1:



Ứng dụng của phương pháp Mã dòng A5/1

- A5/1 đã từng được sử dụng để mã hóa các dữ liệu real-time như các dãy بیت audio
- A5/1 được sử dụng để mã hóa dữ liệu cuộc gọi trong mạng điện thoại GSM.

Giới thiệu phương pháp Mã dòng RC4

- RC4 được thiết kế để đạt hiệu năng cao khi cài đặt bằng phần mềm
- Xây dựng bởi Ron Rivest năm 1987 nhưng đến năm 1994 mới được tiết lộ
- Được ứng dụng rộng rãi
- Kích thước khóa: 40-2048 bit

Để đơn giản, ta sẽ tìm hiểu mô hình thu nhỏ của **RC4** đó là **TinyRC4**

Phương pháp Mã dòng TinyRC4

- Giới thiệu:

- ◇ Khác với A5/1, đơn vị mã hóa của TinyRC4 là 3 bit và nó dùng hai mảng S và T , mỗi mảng gồm 8 số nguyên 3 bit (từ 0 đến 7)
- ◇ Khóa là một dãy gồm N số nguyên 3 bit với N có thể lấy giá trị từ 1 đến 8
- ◇ Bộ sinh số mỗi lần sinh ra 3 bit để sử dụng trong phép XOR. Quá trình sinh số của TinyRC4 gồm hai giai đoạn:

Phương pháp Mã dòng TinyRC4

◇ Quá trình sinh số của TinyRC4 gồm hai giai đoạn:

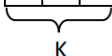
a) Giai đoạn khởi tạo:

```
/* Khởi tạo dãy số S và T */  
for i = 0 to 7 do  
    S[i] = i;  
    T[i] = K[i mod N];  
next i  
/* Hoán vị dãy S */  
j = 0;  
for i = 0 to 7 do  
    j = (j + S[i] + T[i]) mod 8;  
    Swap(S[i], S[j]);  
next i
```

Phương pháp Mã dòng TinyRC4

- Mô tả giai đoạn khởi tạo:
 - Dãy S gồm các số nguyên 3 bit từ 0 đến 7 được sắp thứ tự tăng dần
 - Dựa trên các phần tử của khóa K , các phần tử của S được hoán vị lẫn nhau đến một mức độ ngẫu nhiên nào đó.
- Thí dụ: mã hóa bản rõ $P = 001000110$ (từ "bag") với khóa K gồm 3 số 2, 1, 3 ($N = 3$)
 - Khởi tạo S và T

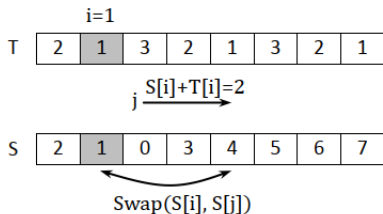
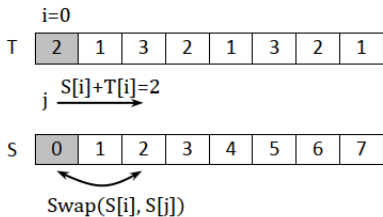
S	0	1	2	3	4	5	6	7
T	2	1	3	2	1	3	2	1



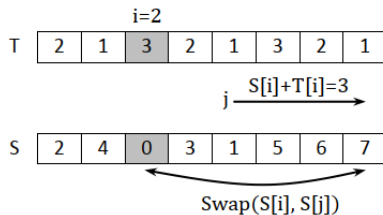
K

Phương pháp Mã dòng TinyRC4

- Hoán vị S



Phương pháp Mã dòng TinyRC4



Quá trình thực hiện đến khi $i = 7$ và lúc đó dãy S là 6 0 7 1 2 3 5 4

Phương pháp Mã dòng TinyRC4

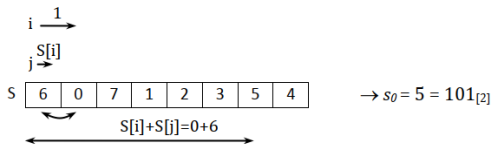
b) Giai đoạn sinh số:

```
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;
```

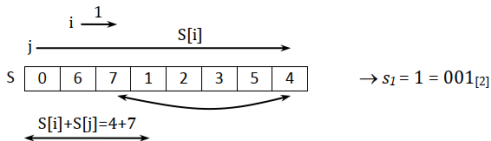
Phương pháp Mã dòng TinyRC4

- Mô tả 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 bit là số được dùng để XOR với đơn vị mã hóa của bản rõ
- Thí dụ: quá trình sinh số mã hóa bản rõ "bag" thực hiện như sau:

Bước 0:

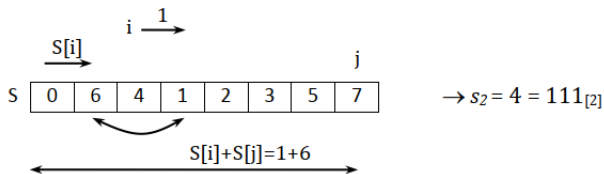


Bước 1:



Phương pháp Mã dòng TinyRC4

Bước 2:



Vậy bản mã là $C = 001.000.110 \oplus 101.001.111 = 100.001.001$ (từ EBB)

Phương pháp Mã dòng RC4

- Giới thiệu:

Cơ chế hoạt động của **RC4** cũng giống như **TinyRC4** với các đặc tính sau:

- Đơn vị mã hóa của RC4 là một byte 8 bit.
- Mảng S và T gồm 256 số nguyên 8 bit
- Khóa K là một dãy gồm N số nguyên 8 bit với N có thể lấy giá trị từ 1 đến 256
- Bộ sinh số mỗi lần sinh ra một byte để sử dụng trong phép XOR.

Phương pháp Mã dòng RC4

Hai giai đoạn của RC4 là:

a) Giai đoạn khởi tạo:

```
/* Khởi tạo day S và 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 i
```

Phương pháp Mã dòng RC4

b) 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;
```

Đánh giá phương pháp Mã dòng RC4

- ◇ Thuật toán đơn giản, rõ ràng
- ◇ Kích thước từ có thể thay đổi (ví dụ, dùng 4 bít thay vì 8 bít)
- ◇ Quá trình sinh số của RC4 cũng sinh ra dãy số ngẫu nhiên, khó đoán trước
- ◇ RC4 đạt được mức độ an toàn cao theo tinh thần của mã hóa One-Time Pad
- ◇ Mã hóa RC4 hoàn toàn được thực hiện trên các số nguyên một byte do đó tối ưu cho việc thiết lập bằng phần mềm và tốc độ thực hiện nhanh hơn so với mã khối

Ứng dụng của mã dòng RC4

- ◇ RC4 được dùng trong giao thức SSL
- ◇ RC4 được sử dụng trong mã hóa WEP (Wired Equivalent Privacy) của mạng Wireless LAN.
- ◇ BitTorrent protocol encryption
- ◇ Opera Mini
- ◇ Remote Desktop Protocol
- ◇ PDF
- ◇ Skype
- ◇ Secure shell*

MÃ KHỐI (Block Cipher)

Mã khối an toàn lý tưởng

- Phép toán XOR đảm bảo cho tốc độ mã hóa, nhưng chỉ cần biết cặp khối (bản rõ và bản mã) thì có thể xác định được khóa và có thể dùng khóa này để giải các khối mã hóa khác.
- Một cách tổng quát, nếu giữa bản rõ P và bản mã C có mối liên hệ toán học thì kẻ phá mã có thể tìm được khóa
- Vậy để an toàn ta phải làm cho P và C không có mối quan hệ toán học nào bằng cách lập bảng tra cứu ngẫu nhiên giữa bản rõ và bản mã

Mã khối an toàn lý tưởng

- Thí dụ: bảng ngẫu nhiên giữa bản rõ và bản mã:

Bản rõ	Bản mã
0000	1110
0001	0100
0010	1101
0011	0001
0100	0010
0101	1111
0110	1011
0111	1000
1000	0011
1001	1010
1010	0110
1011	1100
1100	0101
1101	1001
1110	0000
1111	0111

Mã khối an toàn lý tưởng

- Khóa lúc này chính là bảng trên
- Kẻ phá mã chỉ có thể biết một phần bảng map trên, không thể biết hết từng mã như trên
- Nếu ta tăng kích thước của khối lên 64 bit \Rightarrow càng khó để phá mã vì số lượng bảng khóa rất lớn - cỡ $2^{64} \Rightarrow$ mã khối an toàn lý tưởng
- Tuy nhiên, nếu size của khóa tăng thì sẽ khó khăn cho việc lưu trữ và trao đổi. Bảng khóa có 2^{64} dòng mỗi dòng 64 bit do đó kích thước khóa sẽ là $64 \times 2^{64} = 2^{70} \approx 10^{21}$ bit \Rightarrow mã khối an toàn lý tưởng là không khả thi trong thực tế.

Nguyên lý thuyết kế mã khối

- Nguyên tắc Kerckhoffs: được Auguste Kerckhoffs đưa ra trong thế kỷ 19:
 - A cryptosystem should be secure even if everything about the system, except the key, is public knowledge
 - The enemy knows the system (Claude Shannon)
- Nguyên lý Khuếch tán và xáo trộn:
 - Xáo trộn (Confusion): mỗi bit của bản mã phải phụ thuộc vào một phần của khóa và sao cho che dấu được mối liên quan giữa khóa và bản mã
 - Khuếch đại (Diffusion): nếu ta thay đổi một bit của bản rõ thì một nửa số bit trong bản mã sẽ thay đổi và ngược lại.