**2. 2. HỆ MÃ HÓA ĐỐI XỨNG - CỔ ĐIỂN**

***Khái niệm***

Hệ mã hóa đối xứng đã được dùng từ rất sớm, nên còn gọi là ***Hệ mã hóa dối xứng - cổ điển*** (gọi ngắn gọn là ***Hệ mã hóa dối xứng cổ điển***).

Bản mã hay bản rõ là dãy các ký tự Latin.

***Lập mã:*** thực hiện theo các bước sau:

1/. Nhập bản rõ ký tự: RÕ\_CHỮ. 2/. Chuyển RÕ\_CHỮ ==> RÕ\_SỐ.

3/. Chuyển RÕ\_SỐ ==> MÃ\_SỐ. 4/. Chuyển MÃ\_SỐ ==> MÃ\_CHỮ.

***Giải mã***: thực hiện theo các bước sau:

1/. Nhập bản mã ký tự: MÃ\_CHỮ. 2/. Chuyển MÃ\_CHỮ ==> MÃ\_SỐ.

3/. Chuyển MÃ\_SỐ ==> RÕ\_SỐ. 4/. Chuyển RÕ\_SỐ ==> RÕ\_CHỮ.

Để chuyển từ CHỮ sang SỐ hay ngược lại từ SỐ trở về CHỮ, người ta theo một qui ước nào đó, ví dụ chữ cái thay bằng số theo **modulo 26** như sau:

Thiếu cột 25 , W = 23

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| A | B | C | D | E | F | G | H | I | J | K | L | M | N | O | P | Q | R | S | T | U | V | X | Y |  |
| 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | 10 | 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 | 22 | 24 | 26 |

Để thực hiện mã hóa hay giải mã với các “***số***”, người ta dùng các phép toán số học theo **modulo 26**.

***Các hệ mã hóa cổ điển***

Mã hóa cổ điển gồm nhiều hệ, ví dụ:

Hệ mã hóa dịch chuyển: Khóa có 1 “chìa”. (Thể hiện bằng 1 giá trị).

Hệ mã Affine: Khóa có 2 “chìa”. (Thể hiện bằng 2 giá trị).

Hệ mã hóa thay thế: Khóa có 26 “chìa”. (Thể hiện bằng 16 giá trị).

Hệ mã hóa VIGENERE: Khóa có m “chìa”. (Thể hiện bằng m giá trị).

Hệ mã hóa HILL: Khóa có ma trận “chìa” (chùm chìa khóa).

**2.2.1. Hệ mã hóa: Dịch chuyển**

***Sơ đồ***

Đặt **P** = **C** = **K** = **Z26**.  ***Bản mã* y** và ***bản rõ* x** ∈ **Z26**.

Với khóa **k** ∈ **K**, ta định nghĩa:

Hàm Mã hóa: **y** = **ek**(**x**) = (**x** + **k**)mod26

Hàm Giải mã: **x** = **dk**(**y**) = (**y** – **k**)mod 26

***Độ an toàn Độ an toàn của mã dịch chuyển: Rất thấp.***

Tập khóa **K** chỉ có 26 khóa, nên việc phá khóa (thám mã) có thể thực hiện dễ dàng bằng cách thử kiểm tra từng khóa: **k** = 1, 2, 3, ..., 26.

**2.3.2. Hệ mã hóa: Thay thế (Hoán vị toàn cục)**

***Sơ đồ***

Đặt **P** = **C** = **Z26**. ***Bản mã* y** và ***bản rõ* x** ∈ **Z26**.

Tập khóa **K** là tập mọi hoán vị trên **Z26**.

Với khóa **k** *=* **π** ∈ **K**, tức là 1 hoán vị trên **Z26**, ta định nghĩa:

Mã hóa: **y** = **eπ (x) = π (x)**

Giải mã: **x =** **dπ (y)** = **π -1** **(y)**

***Độ an toàn*** Độ an toàn của mã thay thế: ***Thuộc loại cao.***

Tập khóa K có 26 ! khóa ( > 4. 1026 ), nên việc phá khóa (thám mã) có thể thực hiện bằng cách duyệt tuần tự **26 !** hoán vị của 26 chữ cái.

Để kiểm tra tất cả **26 !** khóa, tốn rất nhiều thời gian !

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

**2.3.3. Hệ mã hóa: AFFINE**

***Sơ đồ***

Đặt **P** = **C** = **Z26**. ***Bản mã* y** và ***bản rõ* x** ∈ **Z26**.

Tập khóa **K** = {(**a**, **b**), với **a, b**∈ **Z26**, UCLN(**a**, 26) = 1}

Với khóa  **k** = **(a**, **b**)∈ **K**, ta định nghĩa:

Phép Mã hóa **y** = **ek**(**x**) = (**a** **x + b**) mod 26

Phép Giải mã **x** = **dk**(**y**) = **a** **-1** ( **y - b**) mod 26

***Độ an toàn*** Độ an toàn của Hệ mã hóa Affine***: Rất thấp.***

+ Điều kiệnUCLN(**a**, 26) = 1 để bảo đảm **a** có phần tử nghịch đảo **a –1**mod 26, tức là thuật toán giải mã **dK** luôn thực hiện được.

+ Số lượng **a** ∈ **Z**26 nguyên tố với 26 là φ(26) = **12** , đó là

1, 3, 5, 7 ,9, 11, 15, 17, 19, 21, 23, 25

Các số nghịch đảo theo (mod 26) tương ứng: 1, 9, 21, 15, 3, 19, 7, 23, 11, 5, 17, 25

+ Số lượng **b** ∈ **Z**26 là **26** .

+ Số các khoá (**a, b**) có thể là **12** \* **26** = **312.** Rất ít !

Như vậy việc dò tìm khóa mật khá dễ dàng.

**2.3. 4. Hệ mã hóa : VIGENERE**

***Sơ đồ***

Đặt **P** = **C** = **K** = (**Z26**)**m** , **m** là số nguyên dương, các phép toán thực hiện trong **Z26**.

***Bản mã* Y** và ***bản rõ* X** ∈ (**Z26**)**m**. Khoá **k** = (**k1**, **k2**, …., **km**) gồm **m** phần tử.

Mã hóa **Y** =(**y1, y2, …, ym**)= **ek**(**x1, x2, …, xm**)=(**x1 + k1, x2 + k2, …, xm + km**) mod **m**.

Giải mã **X** =(**x1, x2, …, xm**)= **dk** (**y1, y2, …, ym**)=(**y1 - k1, y2 - k2, …, ym – km**) mod **26**.

***Ví dụ***

\* Bản rõ chữ: **THISISACRYPTOSYSTEM**

Chọn khoá: **k** = “**KWORD**” = {**10, 22, 14, 17, 3**} với độ dài **m**=5.

\* Bản rõ số: S**X** = 19 7 8 18 8 18 0 2 17 24 15 19 14 18 24 18 19 4 12

\* Mã hóa:

Chia bản rõ S**X** thành các đoạn, mỗi đoạn gồm **m** =5 số.

Với mỗi đoạn, áp dụng công thức mã hóa, ta nhận được bản mã số.

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 19  10 | 7  22 | 8  14 | 18  17 | 8  3 | 18  10 | 0  22 | 2  14 | 17  17 | 24  3 |
| 3 | 3 | 22 | 9 | 11 | 2 | 22 | 16 | 8 | 1 |

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 15  10 | 19  22 | 14  14 | 18  17 | 24  3 | 18  10 | 19  22 | 4  14 | 12  17 |
| 25 | 15 | 2 | 9 | 1 | 2 | 15 | 18 | 3 |

\* Bản mã số: S**Y** = **3 3 22 9 11 2 22 16 8 1 25 15 2 9 1 2 15 18 3**

\* Bản mã chữ: **DDWJL CWQIB ZPCJB CPSD**

***Độ an toàn*** Độ an toàn của mã VIGENERE***: Tương đối cao.***

Nếu khoá gồm **m** ký tự khác nhau, mỗi ký tự có thể được ánh xạ vào 1 trong **m** ký tự có thể, do đó hệ mật này được gọi là hệ ***thay thế đa biểu***.

Như vậy số khoá (độ dài **m**) có thể có trong mật Vigenere là **26 m** .

Nếu dùng phương pháp “tấn công vét cạn”, thám mã phải kiểm tra **26 m** khóa.

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

**2.3.5. Hệ mã hóa: Hoán vị cục bộ.**

***Sơ đồ***

Đặt **P** = **C** = **Z26 m** , **m** là số nguyên dương. ***Bản mã* Y** và ***bản rõ* X** ∈ (**Z26**)**m**.

Tập khóa *K* là tập tất cả các hoán vị của {1, 2, …., **m**}.

Với mỗi khoá **k** = **π** *K* , **k** = (**k1**, **k2**, …., **km**) gồm **m** phần tử, ta định nghĩa:

\* Mã hóa **Y** = (**y1, y2, …, ym**) = **ek** (**x1, x2, …, xm**) = (**xk(1), xk(2) , … , xk(m)**)

\* Giải mã **X** = (**x1, x2, …, xm**) = **dk** (**y1, y2, …, ym**) = (**yk(1)-1, yk(2)-1, … , yk(m)-1**)

Trong đó **k** -1 = **π** -1 là hoán vị ngược của **π**.

***Độ an toàn***

Nếu dùng phương pháp “tấn công vét cạn”, thám mã phải kiểm tra số khóa có thể là:

**1 ! + 2! + 3 ! + … + m !**  trong đó m ≤ 26.

Hiện nay với hệ mã này, người ta có phương pháp thám mã khác nhanh hơn.

**2.3.6. Hệ mã hóa: HILL**

***Sơ đồ***  Lester S. Hill đưa ra năm 1929.

Đặt **P** = **C** = **Z26 m** , **m** là số nguyên dương. ***Bản mã* Y** và ***bản rõ* X** ∈ (**Z26**)**m**.

Tập khóa *K* = {**K** **Z 26 m\*m**/ (**det** (**K**), 26) = **1**}. (**K** phải có **K -1** ).

Mỗi khóa **K** là một “***Chùm chìa khóa***” (một Ma trận “Các chìa khóa” ).

Với mỗi **K**  *K* , định nghĩa:

\* Hàm lập mã: **Y** = (**y1, y2, …, ym**) = **ek** (**x1, x2, …, xm**) = (**x1, x2, …, xm**) \* **K**

\* Hàm giải mã: **X** = (**x1, x2, …, xm**) = **dk** (**y1, y2, …, ym**) = (**y1, y2, …, ym**) \* **K -1**

***Ví dụ***

\* Bản rõ chữ: **TUDO**

Chọn **m** = 2, khóa **K** = , bảo đảm UCLN (**det** (**K**), 26) = **1,** tính **K -1**

\* Bản rõ số: **19 20** | **3 14**

**x1** **x2**|**x1 x2**

Với mỗi bộ rõ số (**x1** , **x2**), theo hàm lập mã (**y1** , **y2**) = (**x1** , **x2**) \* **K**, ta tính được:

**y1** = **11 \* x1 + 3 \* x2** , **y2** = **8 \* x1 + 7 \* x2**

\* Bản mã số: **9 6 | 23 18**

\* Bản mã chữ: **FGXS**

***Độ an toàn***

Nếu dùng phương pháp “tấn công vét cạn”, thám mã phải kiểm tra số khóa có thể

với **m** lần lượt là **2, 3, 4, …** trong đó **m** lớn nhất là bằng độ dài bản rõ.

***Ví dụ***

Xâu đầu vào là **a** = “ABC”, xây dựng **M** như sau:

***a***: = “ABC” = "01000001 01000010 01000011". (Chú ý: ‘A’ =65).

\* Độ dài tính theo bit của xâu ***a*:** |***a***| = 24 bit

=> **d** = 447 – (|***a***| mod 512) = 423.

|***a***| + 1 + **d** = 24 + 1 + 423 = 448 mod 512.

\* Biểu diễn nhị phân của độ dài xâu **a** là ***l***:

***l*** = |***a***| mod 264 = 24 mod 264 = 24 = 16 + 8 = 24 + 23 = ( 11000 )2

=> Độ dài của l là | ***l*** | = |11000| = 59 + 5 = 64.

M = **a** || **1** || 0d || ***l***

=> M = 01000001 01000010 01000011 || **1** ||  || **11000**

M = M[0] M[1] … M[N-1], N  0 mod 16

M[0] = 01000001 01000010 01000011 10000000

M[1] = M[2] = ….. = M[13] = M[14] = 

M[15] = 00000000 00000000 00000000 00011000