**TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN VÀ TRUYỀN THÔNG**

**BỘ MÔN AN TOÀN HỆ THỐNG THÔNG TIN**

-------------o0o------------

**BÀI GIẢNG MÔN HỌC**

**MẬT MÃ HỌC VÀ ỨNG DỤNG**

Biên soạn: Ths. Trịnh Minh Đức

Hệ đào tạo: Chính quy

Thái Nguyên 12-2015

***Chương 1*. CƠ SỞ TOÁN HỌC**

Lý thuyết mật mã là một ngành khoa học được xây dựng dựa trên cơ sở

toán học, đặc biệt là lý thuyết số học. Chương này sẽ hệ thống lại *một số kiến thức*

*toán học* cần thiết, được sử dụng trong lý thuyết mật mã và An toàn dữ liệu.

**1.1. MỘT SỐ KHÁI NIỆM TRONG SỐ HỌC**

1.1.1. Ước chung lớn nhất, bội chung nhỏ nhất

***1.1.1.1. Khái niệm***

**1). Ước số và bội số**

Cho hai số nguyên a và b, b ≠ 0. Nếu có một số nguyên q sao cho a = b\*q, thì

ta nói rằng a ***chia hết*** cho b, kí hiệu **b\a**. Ta nói b là ***ước*** của a, và a là ***bội*** của b.

**Ví dụ:**

Cho a = 6, b = 2, ta có 6 = 2\*3, ký hiệu 2 \ 6. Ở đây 2 là ước của 6 và 6 là bội của 2.

Cho các số nguyên a, b ≠ 0, tồn tại cặp số nguyên (q, r) (0 ≤ r < /b/) duy nhất sao cho **a**  = **b** \* **q** + **r**. Khi đó **q** gọi là ***thương nguyên***,  **r** gọi là ***số dư*** của phép chia a cho b. Nếu **r** = 0 thì ta có phép chia hết.

**Ví dụ:**

Cho a = 13, b = 5, ta có 12 = 5\*2 + 3. Ở đây thương là q = 2, số dư là r = 3.

**2). Ước chung lớn nhất, bội chung nhỏ nhất**

Số nguyên **d** được gọi là ***ước chung*** của các số nguyên a1, a2, …, an , nếu nó là ***ước*** của tất cả các số đó.

Số nguyên **m** được gọi là ***bội chung*** của các số nguyên a1, a2, …, an , nếu nó là

***bội*** của tất cả các số đó.

Một ước chung **d** >0 của các số nguyên a1, a2, …, an , trong đó mọi ước chung của a1, a2, …, an đều là ước của **d**, thì **d** được gọi là ***ước chung lớn nhất*** (UCLN) của a1, a2, …, an . Ký hiệu **d** = **gcd** (a1, a2, …, an) hay **d** = **UCLN**(a1, a2, …, an).

Nếu **gcd**(a1, a2, …, an) = 1, thì các số a1, a2, …, an được gọi là ***nguyên tố cùng nhau***.

Một bội chung **m** >0 của các số nguyên a1, a2, …, an , trong đó mọi bội chung của a1, a2, …, an đều là bội của **m**, thì **m** được goi là ***bội chung nhỏ nhất*** (BCNN) của a1, a2, …, an . Ký hiệu **m** = **lcm**(a1, a2, …, an) hay **m** = **BCNN**(a1, a2, …, an).

**Ví dụ:**

Cho a =12, b =15, **gcd**(12,15) = 3, **lcm**(12,15) = 60.

Hai số 8 và 13 là ***nguyên tố cùng nhau***, vì **gcd**(8, 13) = **1**.

***Ký hiệu:***

**Zn** = {0, 1, 2, .. . , n-1} là tập các số nguyên không âm < n.

 **=** {e ∈ **Zn**, e là nguyên tố cùng nhau với n}. Tức là e # 0.

**Ví dụ:**

**Z 7** = {0, 1, 2, 3, 4, 5, 6}. Khi đó số phần tử của **Z 7** là /**Z 7** /= 7.

**Z 7** \* = {1, 2, 3, 4, 5, 6}. Khi đó số phần tử của **Z 7** \*  là / **Z 7** \*/= 7.

***1.1.1.2. Tính chất***

1). **d** = gcd(a1, a2, …, an) khi và chỉ khi tồn tại các số x1,x2,…, xn sao cho:

**d** = a1x1+a2x2+…+anxn

Hệ quả: a1, a2, …, an nguyên tố cùng nhau ⇔ tồn tại các số x1,x2,…, xn

sao cho: **1** = a1x1+a2x2+…+anxn

2). **d** = gcd(a1, a2, …, an) ⇔ gcd(a1/d, a2/d,…, an/d) =1.

3). **m** = lcm(a1, a2, …, an) ⇔ gcd(m/a1, m/a2,…, m/an) =1.

4). gcd(**m** a1, **m** a2, …, **m** an) = **m** \* gcd(a1, a2, …, an) (với m ≠ 0).

5). Nếu gcd(a, b) =1 thì lcm(a, b) = a \* b

6). Nếu b>0, a = bq+r thì gcd(a,b) = gcd(b, r).

***1.1.1.3. Thuật toán Euclide tìm ước chung lớn nhất***

**1). Bài toán**

***\* Dữ liệu vào*:** Cho hai số nguyên không âm a, b, a ≥ b.

***\* Kết quả*: gcd**(a,b).

**2). Thuật toán** (*Mô phỏng bằng ngôn ngữ Pascal*).

Readln(a, b);

While b>0 do

begin

r := a mod b; a := b; b := r ;

end;

Writeln(a); **gcd**(a,b) = a;

**3). Ví dụ:** a=30, b=18; gcd(30,18) = gcd(18,12) = gcd(12,6) = gcd(6,0) = 6

|  |  |  |  |
| --- | --- | --- | --- |
| A | b | r | a = b.q + r |
| 30 | 18 | 12 | 30 = 18 \* 1+12 |
| 18 | 12 | 6 | 18 = 12 \* 1+6 |
| 12 | 6 | 0 | 12 = 6 \* 2 + 0 |

***1.1.1.4. Thuật toán Euclide mở rộng***

**1). Bài toán:**

***\* Dữ liệu vào*:** Cho hai số nguyên không âm a, b, a ≥ b.

***\* Kết quả*: d = gcd** (a,b)và hai số **x**, **y** sao cho:a **x +** b **y** = **d.**

**2). Thuật toán:** (*Mô phỏng bằng ngôn ngữ Pascal*)

Readln(a, b);

IF b=0 THEN

Begin

d := a; x := 1; y := 0;

writeln(d, x, y);

End

ELSE

Begin

x2 := 1; x1 := 0; y2 := 0; y1 := 1;

While b>0 Do

begin

q := a div b; r := a mod b;

x := x2-q\*x1; y := y2-q\*y1;

a := b; b := r; x2 := x1; x1 := x; y2 := y1; y1 := y;

end;

d := a; x := x2; y := y2;

writeln(d, x1, x2);

End;

**1.1.2. Quan hệ “Đồng dư”**

***1.1.2.1. Khái niệm***

Cho các số nguyên a, b, **m** (m > 0). Ta nói rằng ***a*** và ***b “đồng dư” với nhau***

theo **modulo m**, nếu chia a và b cho **m**, ta nhận được cùng một số dư.

Ký hiệu: **a ≡ b** (mod **m**).

**Ví dụ:**

17 ≡ 5 (mod 3) vì chia 17 và 5 cho 3, được cùng số dư là 2.

**Nhận xét:** Các mệnh đề sau đây là tương đương:

1). a ≡ b (mod m)

2). m \ (a – b)

3). Tồn tại số nguyên **t** sao cho a = b + m **t**

***Chứng minh:***

1) ⇒ 2):

Nếu có 1), thì theo định nghĩa: a, b chia cho m, phải có cùng số dư, do đó:

a = mqa + r; b = mqb + r; Suy ra (a – b) = m (qa - qb), tức là m \ (a - b).

2) ⇒ 3):

Nếu có 1), tức là m \ (a – b). Nghĩa là có t  Z sao cho (a - b) = mt hay a = b + mt

3) ⇒ 1):

Nếu có 1), tức là tồn tại số nguyên **t** sao cho a = b + m **t**.

Lấy a chia cho m, giả sử thương là qa và dư r, hay **a** = mqa + r (0 ≤ r <m), do đó:

b + m **t** = a = mqa + r hay **b** = m(qa - **t**) + r (0 ≤ r < m). Điều đó chứng tỏ khi chia a và b cho m được cùng số dư r, hay a ≡ b (mod m).

***1.1.2.2. Các tính chất của quan hệ “đồng dư”***

***1). Quan hệ “đồng dư” là quan hệ tương đương trong Z:***

Với mọi số nguyên dương m ta có:

a ≡ a (mod m) với mọi a  Z; *(tính chất phản xạ).*

a ≡ b (mod m) thì b ≡ a (mod m); *(tính chất đối xứng).*

a ≡ b (mod m) và b ≡ c (mod m) thì a ≡ c (mod m); *(tính chất bắc cầu).*

***2). Tổng hay hiệu các “đồng dư”:***

(a+b) (mod n) ≡ [(a mod n) + (b mod n)] (mod n)

(a- b) (mod n) ≡ [(a mod n) - (b mod n)] (mod n)

***Tổng quát***:

Có thể cộng hoặc trừ từng vế nhiều đồng dư thức theo cùng một modulo **m**,

ta được một đồng dư thức theo cùng modulo **m**, tức là:

Nếu ai ≡ bi (mod m) , i = 1...k, thì  với ti = ± 1.

***3). Tích các “đồng dư”:***

(a\* b) (mod n) ≡ [(a mod n) \* (b mod n)] (mod n)

***Tổng quát***:

Có thể nhân từng vế nhiều đồng dư thức theo cùng một modulo **m**,

ta được một đồng dư thức theo cùng modulo **m**, tức là: 

Nếu ai ≡ bi (mod m) với i=1..k, thì ta có: 

***Hệ quả:***

\* Có thể cộng hoặc trừ cùng một số vào hai vế của một đồng dư thức.

\* Có thể chuyển vế các số hạng của đồng dư thức bằng cách đổi dấu các số hạng đó.

\* Có thể cộng vào một vế của đồng dư thức một ***bội***  của modulo:

a ≡ b (mod m) → a+km ≡ b (mod m) với mọi k  Z

\* Có thể nhân hai vế của một đồng dư thức với cùng một số:

a ≡ b (mod m) → ac ≡ bc (mod m) với mọi c  Z

\* Có thể nâng lên ***lũy thừa*** bậc nguyên không âm cho 2 vế của một đồng dư thức: a ≡ b (mod m) → an ≡ bn (mod m) với mọi n  Z+

\* Có thể ***chia***  2 vế đồng dư thức cho một ước chung nguyên tố với modulo:

c\a, c\b, (c,m)=1, a ≡ b (mod m) ⇒ a/c ≡ b/c (mod m)

\* Có thể ***nhân*** 2 vế đồng dư thức và modulo với cùng một số nguyên dương,

Nếu a ≡ b (mod m), c >0 ⇒ ac ≡ bc (mod mc)

\* Có thể ***chia*** 2 vế đồng dư thức và modulo cho cùng một số nguyên dương

là ước chung của chúng:

Nếu c \ (a, b, m) ⇒ a/c ≡ b/c (mod m/c)

\* a ≡ b (mod m) ⇒ a ≡ b (mod k) với k \ m

\* a ≡ b (mod m) ⇒ gcd(a, m) = gcd(b, m)

***Các lớp thặng dư***

Quan hệ “đồng dư” theo modulo **m** trên tập Z (tập các số nguyên) là một quan hệ tương đương (vì có tính chất phản xạ, đối xứng, bắc cầu), do đó nó tạo ra trên tập Z một phân hoạch gồm các lớp tương đương: hai số nguyên thuộc cùng một lớp tương đương khi và chỉ khi chúng có cùng một số dư khi chia cho **m**.

Mỗi lớp tương đương đại diện bởi một số duy nhất trong tập **Z­m** ={0, 1,…, m-1} là số dư khi chia các số trong lớp cho **m**, ký hiệu một lớp được đại diện bởi số **a**

là **[a]m** .

Như vậy [a]m = [b]m ⇔ a ≡ b (mod m)

Vì vậy ta có thể đồng nhất **Zm** với tập các lớp tương đương theo modulo **m**.

**Z­m** ={0, 1, 2,…, **m**-1} được gọi là tập các “***thặng dư đầy đủ***” theo modulo **m**. Mọi số nguyên bất kỳ đều có thể tìm được trong **Zm** một số đồng dư với mình theo modulo **m**. [30]

**1.1.3. Số nguyên tố**

***1.1.2.1. Khái niệm***

Số nguyên tố là số tự nhiên lớn hơn 1 và chỉ có hai ước là 1 và chính nó.

**Ví dụ:**

Các số 2, 3, 5, 7, 11, 13, 17, 19, 23, 29, 31, 37 là số nguyên tố.

Số 2 là số nguyên tố ***chẵn*** duy nhất.

Số nguyên tố có vai trò và ý nghĩa to lớn trong số học và lý thuyết mật mã.

Bài toán kiểm tra tính nguyên tố của một số nguyên dương n và phân tích một số n ra thừa số nguyên tố là các bài toán rất được quan tâm.

**Ví dụ: *10 số nguyên tố lớn đã được tìm thấy*** [33]

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **rank** | **Prime** | **Digits** | **Who** | **when** | **reference** |
| [1](http://primes.utm.edu/primes/page.php?id=78470) | **232582657-1** | [9808358](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M32582657) | [G9](http://primes.utm.edu/bios/code.php?code=G9) | 2006 | Mersenne 44?? |
| [2](http://primes.utm.edu/primes/page.php?id=76545) | **230402457-1** | [9152052](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M30402457) | [G9](http://primes.utm.edu/bios/code.php?code=G9) | 2005 | Mersenne 43?? |
| [3](http://primes.utm.edu/primes/page.php?id=73550) | **225964951-1** | [7816230](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M25964951) | [G8](http://primes.utm.edu/bios/code.php?code=G8) | 2005 | Mersenne 42?? |
| [4](http://primes.utm.edu/primes/page.php?id=70545) | **224036583-1** | [7235733](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M24036583) | [G7](http://primes.utm.edu/bios/code.php?code=G7) | 2004 | Mersenne 41?? |
| [5](http://primes.utm.edu/primes/page.php?id=67522) | **220996011-1** | [6320430](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M20996011) | [G6](http://primes.utm.edu/bios/code.php?code=G6) | 2003 | Mersenne 40?? |
| [6](http://primes.utm.edu/primes/page.php?id=1) | **213466917-1** | [4053946](http://www.isthe.com/chongo/tech/math/prime/mersenne.html#M13466917) | [G5](http://primes.utm.edu/bios/code.php?code=G5) | 2001 | Mersenne 39 |
| [7](http://primes.utm.edu/primes/page.php?id=80385) | **19249·213018586+1** | 3918990 | [SB10](http://primes.utm.edu/bios/code.php?code=SB10) | 2007 |  |
| [8](http://primes.utm.edu/primes/page.php?id=74836) | **27653·29167433+1** | 2759677 | [SB8](http://primes.utm.edu/bios/code.php?code=SB8) | 2005 |  |
| [9](http://primes.utm.edu/primes/page.php?id=73145) | **28433·27830457+1** | 2357207 | [SB7](http://primes.utm.edu/bios/code.php?code=SB7) | 2004 |  |
| [10](http://primes.utm.edu/primes/page.php?id=82804) | **33661·27031232+1** | 2116617 | [SB11](http://primes.utm.edu/bios/code.php?code=SB11) | 2007 |  |

***1.1.2.2. Định lý về số nguyên tố***

**1). Định lý: về số nguyên dương > 1.**

Mọi số nguyên dương **n** > 1 đều có thể biểu diễn được ***duy nhất*** dưới dạng:

 , trong đó:

**k**, **ni** ( i =1,2,..,k) là các số tự nhiên, **Pi** là các số nguyên tố, từng đôi một khác nhau.

**2). Định lý: Mersenne.**

Cho **p** = **2k -1**, nếu **p** là số nguyên tố, thì **k** phải là số nguyên tố.

***Chứng minh***

Bằng phản chứng, giả sử **k** không là nguyên tố. Khi đó k = a.b với 1< a, b < k. Như vậy  **p** = 2k -1 = 2ab -1 = (2a)b -1= (2a -1).E

(Trong đó E là một biểu thức nguyên - áp dụng công thức nhị thức Niu-tơn).

Điều này mâu thuẫn giả thiết **p** là nguyên tố. Vậy giả sử là sai, hay k là số nguyên tố.

**3). Hàm Euler:**

Cho số nguyên dương **n**, ***số lượng*** các số nguyên dương bé hơn n và ***nguyên tố***

***cùng nhau*** với n được ký hiệu **(n)** và gọi là hàm ***Euler***.

***Nhận xét***: Nếu **p** là số nguyên tố, thì **(p)** = **p-1**

**Ví dụ:**

Tập các số nguyên không âm nhỏ hơn 7 là **Z 7** = {0, 1, 2, 3, 4, 5, 6}.

Do **7** là ***số nguyên tố***, nên Tập các số nguyên dương nhỏ hơn 7 và *nguyên tố*

*cùng nhau* với 7 là **Z 7** \* ={1, 2, 3, 4, 5, 6}. Khi đó /**Z**/ = **(p)** = **p-1** = 7 - 1 = 6**.**

**Định lý: về Hàm Euler.**

Nếu n là tích của hai số nguyên tố **n** = **p.q**, thì **(n)** = (p).(q) = (p-1).(q-1).

**(n)** = / **Z n** \* /

***1.1.2.3. Phương pháp kiểm tra tính nguyên tố***

Kiểm tra tính nguyên tố của một số nguyên dương là bài toán nảy sinh trong nhiều ứng dụng, đặc biệt là trong lý thuyết mật mã. Năm 1975 Pratt đã chứng minh nó thuộc lớp NP và thuộc lớp co-NP NP, đây là bài toán “khó”.

***1). Phương pháp cổ điển.***

***Ý tưởng***:

Kiểm tra tính nguyên tố của một số nguyên dương **n** theo định nghĩa:

Thử lần lượt tìm các ***ước*** của **n**, từ 2 đến n / 2.

Nếu không tìm được ước nào thì kết luận **n** là nguyên tố.

***Thuật toán***:

KT := 1;

for i := 2 to sqrt(n) do

if (n mod i) = 0 then

begin

KT := 0; Break;

end;

IF KT = 1 THEN Writeln (‘n nguyên tố ‘) ELSE Writeln (‘n không nguyên tố ‘);

*Chú ý*

Trong thuật toán trên, vòng lặp có số lần lặp là n1/ 2/2 = n.

Nếu n = 10150 thì thuật toán trên phải tính khoảng 10150 phép tính.

Nếu dùng 1 máy tính nhanh nhất hiện nay (2007): khoảng 500 000 tỷ = 5 \* 1014 phép tính trong 1 giây, thì thời gian thực hiện là khoảng 5 \* 10136  giây.

Mỗi ngày có khoảng 24 giờ \* 60 phút \* 60 giây ≈ 10 5 giây.

Mỗi năm có khoảng 365 ngày \* 10 5 giây ≈ 10 8 giây.

Như vậy nếu n = 10150 thì thuật toán trên phải tính khoảng **10128**năm.

***2). Phương pháp “xác suất“.***

Trên cơ sở các định lý về số nguyên tố, hiện nay người ta có các phương pháp “xác suất“để kiểm tra tính nguyên tố của một số nguyên dương n.

Ví dụ như các phương pháp: Solovay-Strassen, Lehmann-Peralta, Miller-Rabin.

***Định lý Ferma*:**

Nếu p là số nguyên tố, a là số nguyên, thì **a p**≡ **a** (mod p).

Nếu p không chia hết a, thì **a p-1** ≡ **1** (mod p).

**Ví dụ**: 47 ≡ 4 (mod 7); 4 7-1 ≡ 1 (mod 7).

***Định lý Euler*:**

Nếu gcd(a, m) = 1 thì a(m)≡ 1 (mod m).

Trường hợp m là số nguyên tố, ta có định lý Ferma.

**Ví dụ:** m = 10, (m) = (2).(5) = 1 \* 4 = 4.

Ta có 74 ≡ 1 (mod 10), 94  ≡ 1 (mod 10), 214 ≡ 1 (mod 10).

**Hệ quả 1:**

Nếu gcd(c, m) = 1 và a ≡ b (mod (m)) với a, b là các số tự nhiên, thì

ca ≡ cb (mod m) và suy ra ca mod m = ca mod(m) mod m.

*Chứng minh*: a ≡ b (mod (n)) nên a = b + k(m), k  Z và vì vậy

ca = cb+k(m) = cb.(c(m))k ≡ 1 (mod m), theo theo định lý Euler.

***Nhận xét***: Hệ quả trên giúp giảm nhẹ việc tính toán đồng dư của lũy thừa bậc cao.

Ví dụ: Ta thấy (15) = (5).(3) = 4.2 = 8 và 1004 ≡ 4 (mod 8).

Do đó **21004** (mod 15) = 24 (mod 15) = 16 (mod 15) = 1.

**Hệ quả 2:**

Nếu các số nguyên e, d thỏa mãn **e.d ≡ 1** (mod **(n)**), thì với mọi số **c** nguyên tố cùng nhau với m, ta có **(ce)d** ≡ **c** (mod m).

*Chứng minh*: Đặt a = ed và b = 1, từ hệ quả 1 ta có hệ quả 2.

Hệ quả này đóng vai trò then chốt trong việc thiết lập các hệ mã mũ (VD RSA).

***1.1.2.4. Tính toán đồng dư của “ lũy thừa” lớn***

**1). Trường hợp a > (m)**:

Trong trường hợp a > (m), khi ấy b < a. Người ta dùng Hệ quả 1 để tính “đồng dư” của “ lũy thừa” lớn.

**2). Trường hợp (m) > a**:

Trong thực tế tính toán thường gặp m lớn, do đó (m) lớn, thậm chí > a, khi ấy người ta dùng kỹ thuật khác, ví dụ Phương pháp bình phương liên tiếp.

***\* Phương pháp bình phương liên tiếp***.

**Ví dụ**: Tính **8743** (mod **103**).

Khai triển số mũ 43 dưới dạng cơ số 2:

43 = 32+8+2+1 = 25 +23 + 21 + 20 (\*)

Tính liên tiếp các “đồng dư” bình phương như sau:

**87** (mod 103) = **87** (ứng với 20)

**872** (mod 103) = **50** (ứng với 21)

874 (mod 103) = 502 (mod 103) = 28

**878** (mod 103) = 282 (mod 103) = **63** (ứng với 23)

8716 (mod 103) = 632 (mod 103) = 55

**8732** (mod 103) = 552 (mod 103) = **38** (ứng với 25)

Theo khai triển (\*), lấy tích của các lũy thừa bậc 25 , 23 , 21 , 20

(rút gọn theo modulo 130), thu được kết quả:

8743 (mod 103) = 38 \* 63 \* 50 \* 87 (mod 103) = **85**

***\* Định lý về Số dư (ĐL Trung Quốc):***

Cho tập số nguyên tố cùng nhau từng đôi một m1, m2,…mr .

Với mỗi bộ số nguyên bất kỳ a1, a2,…ar , hệ phương trình đồng dư:

**x ≡ ai (mod mi)**, (i =1, 2, …, r), luôn có ***nghiệm duy nhất*** theo modulo **m**,

**m** = m1.m2.…mr . Nghiệm này có thể tính theo công thức:

**x** = a1m2 m3 … mr b1 + m1 a2 m3 … mr b2 + m1 m2 a3 m3 … mr b3 + …+ m1 m2 … mr-1 ar br (mod m1.m2.…mr),

trong đó bi = ( m1.m2…mi-1mi+1…mr)-1 (mod mi), với mọi i =1, 2,…, r.

**Nhận xét:**

Định lý số dư Trung Quốc cho phép tính đồng dư theo modulo của một số lớn (tích của nhiều số nguyên tố cùng nhau), thông qua tính toán đồng dư theo modulo

các số nhỏ (từng thừa số).

**Ví dụ:** Tìm nghiệm của hệ phương trình:



Vì các số 5353, 391, 247 nguyên tố cùng nhau, nên theo định lý Trung Quốc

về số dư hệ, có nghiệm duy nhất theo modulo **m** = 5353\*391\*247 = 516976681.

Để tìm **x** mod **m** ta tính:

m1 = m/5353 = 96577 → y1 = 96577-1 mod 5353 = 5329

m2= m/391 = 1322191 → y2 = 1322191-1 mod 391= 16

m3 = m/247 = 2093023 → y3 = 2093023-1 mod 247 = 238

**x** = 31188.96577.5329 + 139.1322191.16 + 239.2093023.238 (mod m)

= 13824 **(mod** m**)**

**1.2. MỘT SỐ KHÁI NIỆM TRONG ĐẠI SỐ**

**1.2.1. Cấu trúc Nhóm**

***1).***  ***Khái niệm Nhóm***

***Nhóm***là một bộ (**G**, \*), trong đó **G** ≠ ∅, \* là ***phép toán hai ngôi*** trên **G**

thoả mãn ba tính chất sau:

+ Phép toán có tính kết hợp: (x\*y)\* z = x\*(y\*z) với mọi x, y, z ∈ G.

+ Có phần tử phần tử ***trung lập*** **e** ∈ G: x\***e** = **e**\*x = x với mọi x ∈ G.

+ Với mọi x∈ G, có phần tử nghịch đảo x’∈ G: x \* x’ = x’ \* x = e.

***Cấp của nhóm*** **G** được hiểu là số phần tử của nhóm, ký hiệu là .

Cấp của nhóm có thể là ∞ nếu **G** có vô hạn phần tử.

***Nhóm Abel*** là nhóm (**G**, \*), trong đó phép toán hai ngôi \* có tính giao hoán.

*Tính chất:*Nếu a \* b = a \* c, thì b = c.

Nếu a \* c = b \* c, thì a = b.

**Ví dụ:**

\* Tập hợp các số nguyên Z cùng với phép cộng (+) thông thường là nhóm giáo hoán,

có phần tử đơn vị là số 0. Gọi là ***nhóm cộng*** các số nguyên.

\* Tập Q\*  các số hữu tỷ khác 0 (hay tập R\* các số thực khác 0), cùng với phép nhân (\*) thông thường là nhóm giao hoán. Gọi là ***nhóm nhân*** các số hữu tỷ (số thực) khác 0.

\* Tập các vectơ trong không gian với phép toán cộng vectơ là nhóm giao hoán.

***2). Nhóm con*** của nhóm (**G**, \*).

Nhóm con của G là tập S ⊂ G, S ≠ φ, và thỏa mãn các tính chất sau:

+ Phần tử trung lập e của G nằm trong S.

+ S khép kín đối với phép tính (\*) trong G, tức là x\*y ∈ S với mọi x, y∈ S.

+ S khép kín đối với phép lấy nghịch đảo trong G, tức x -1 ∈ S với mọi x∈S.

**1.2.2. Nhóm Cyclic**

***1). Khái niệm*** ***Nhóm Cyclic***

Nhóm (**G**, \*) được gọi là ***Nhóm* *Cyclic*** nếu nó được sinh ra bởi một trong

các phần tử của nó.

Tức là có phần tử **g** ∈ **G** mà với mỗi **a** ∈ **G**, đều tồn tại số **n** ∈ N để

**g n** = **g** \* **g** \* … \* **g** = **a**. (Chú ý g \* g \* … \* g là **g** \* **g** với **n** lần).

Khi đó **g** được gọi là ***phần tử sinh*** hay ***phần tử nguyên thuỷ***  của nhóm **G**.

Nói cách khác: **G** được gọi là Nhóm Cyclic nếu tồn tại **g** ∈ **G** sao cho mọi

phần tử trong **G** đều là một ***luỹ thừa nguyên***nào đó của **g**.

Ví dụ: Nhóm (**Z** + , **+**) gồm các số nguyên dương là *Cyclic* với phần tử sinh **g** = **1**.

***2). Cấp của Nhóm Cyclic:***

Cho (**G**, \*) là ***Nhóm* *Cyclic*** với phần tử sinh **g**. và phần tử trung lập **e.**

Nếu tồn tại số tự nhiên ***nhỏ nhất* n** mà **g n**= **e**, thì **G** sẽ chỉ gồm có **n** phần tử

khác nhau: **e, g, g2 , g3 , . . . , g n - 1**. Khi đó **G** được gọi là *nhóm Cyclic* hữu hạn ***cấp* n**.

Nếu không tồn tại số tự nhiên **n** để **g n**= **e**, thì **G** có ***cấp*** ∞.

**Ví dụ**: (**Z** + , **+**) gồm các số nguyên dương là *Cyclic* với phần tử sinh **g** = **1**, **e** = 0.

Đó là Nhóm Cyclic vô hạn, vì không tồn tại số tự nhiên **n** để **g n**= **e**,

***3). Cấp của một phần tử trong Nhóm Cyclic:***

Phần tử **α** ∈ **G** được gọi là có ***cấp d***, nếu **d** là số nguyên dương ***nhỏ nhất***

sao cho **α d** = **e**, trong đó **e**  là phần tử trung lập của **G**.

Như vậy phần tử **α** có ***cấp 1***, nếu **α** = **e**.

**1.2.3. Nhóm**  (**Zn \*** ,  **phép nhân mod n** )

***1). Khái niệm Tập thặng dư thu gọn theo modulo***

\* Kí hiệu**Zn** = {0, 1, 2, .. . , n-1} là tập các số nguyên không âm < n.

**Zn** và phép cộng (+) lập thành ***nhóm Cyclic*** có phần tử sinh là **1**, pt trung lập **e** = 0.

(**Zn** , + ) gọi là nhóm cộng, đó là nhóm hữu hạn có cấp n.

\* Kí hiệu **Zn \*  =** {**x** ∈ **Zn**, **x** là nguyên tố cùng nhau với **n**}. Tức là **x** phải ≠ 0.

**Zn \***được gọi là ***Tập* *thặng dư thu gọn theo mod n***, có số phần tử là **φ(n)**.

**Zn \*** với *phép nhân mod n* lập thành một *nhóm* (nhóm nhân), pt trung lập **e** = 1.

Tổng quát (**Zn \*** , *phép nhân mod n* ) không phải là nhóm Cyclic.

Nhóm nhân **Zn \*** là Cyclic chỉ khi **n** có dạng: 2, 4, pk, hay 2pk với p là nguyên tố lẻ.

***2). Một số kết quả đã được chứng minh***

\* ***Định lý Lagrange****:*Nếu **G** là nhóm cấp **n** và **α** ∈ **G**, thì ***Cấp*** của **α**  *là*  ***ước*** *của* **n**.

\* ***Hệ quả****:* Giả sử **α** ∈  có ***Cấp*** m, thì m *là* ***ước*** *của* φ(n).

\* ***Định lý****:*Nếu p là số nguyên tố thì  là nhóm Cyclic.

Nếu b ∈  thì b φ(n) ≡ 1 (mod n). Nếu p là số nguyên tố thì φ(p) = p-1.

Do đó với b ∈  (tức b nguyên tố với p), thì bφ(p) ≡ 1 (mod n), hay bp -1 ≡ 1 (mod n).

***Chú ý***

Theo định nghĩa, phần tử **α** ∈ **** có ***cấp* d**nếu **d** là số nguyên dương nhỏ nhất sao cho **α d** = **e** trong . Như vậy trong  ta hiểu là **α d** ≡ **e** (mod n).

***Định lý*:** Nhóm con của một nhóm Cyclic là một nhóm Cyclic.

***3). Phần tử nghịch đảo đối với phép nhân***

***\* Định nghĩa*:**

Cho **a** ∈ Zn , nếu tồn tại **b** ∈ Zn sao cho **a** **b** ≡ 1 (mod **n**), ta nói **b** là ***phần tử nghịch đảo*** của **a** trong Zn và ký hiệu **a -1**.

Một phần tử có phần tử nghịch đảo, gọi là khả nghịch.

***\* Định lý***: UCLN (a, **n**) = 1 ⇔ Phần tử a ∈ Zn  có phần tử nghịch đảo.

**Chứng minh:**

Nếu a a-1 ≡ 1 (mod n) thì a a-1 = 1 + kn ↔ a a-1 - kn = 1 → (a, n) =1.

Nếu (a, n) = 1, ta có a a-1 + kn = 1 → a a-1 = 1+kn, do đó a a-1 ≡ 1 (mod n).

***\* Hệ quả***: Mọi phần tử trong **Zn\***  đều có phần tử nghịch đảo.

***\* Tìm phần tử nghịch đảo bằng Thuật toán Euclid mở rộng.***

Input: **a** ∈ Zn , **n**, Output: Phần tử nghịch đảo của **a**.

Procedure Invert(**a**, **n**);

Begin

g0:=n; g1:=a; u0:=1; u1:=0; v0:=0; v1:=1;

i:=1;

while gi ≠ 0 do

begin

**y** := gi-1 div gi; gi+1 := gi+1 - **y**.gi;

ui+1 := ui+1 - **y**.ui; vi+1 := vi+1 - **y**.vi;

i := i+1;

end;

**t** := vi+1;

if **t** > 0 then **a-1**:= **t** else **a-1**:= **t** + **n**;

End;

**Ví dụ:** Tìmphần tử nghịch đảo của 3 trong **Z7**

Tức là phải giải phương trình 3 **x** ≡ 1 (mod **7**), x sẽ là phần tử nghịch đảo của 3.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **I** | **gi** | **ui** | **vi** | **y** |
| 1 | 7 | 1 | 0 |  |
| 1 | 3 | 0 | 1 | 2 |
| 2 | 1 | 1 | -2 | 3 |
| 3 | 0 |  |  |  |

Vì **t** = v2 = -2 < 0 do đó **x** = **a-1**:= **t** + **n** = -2 + 7 = 5.

Vậy 5 là phần tử nghịch đảo của 3 trong **Z7**

***Chú ý***

***Định lý*  (**Euler tổng quát): Nếu (a, n) = 1 thì **a φ(n)** mod n = 1

***Hệ quả****:* Nếu **p** là số nguyên tố và (**a**, **p**) = 1, thì **a p-1** (mod p) = 1

***4). Khái niệm Logarit rời rạc***

Cho **p** là số nguyên tố, **g** là phần tử nguyên thuỷ của **Zp** , β ∈ 

**“*Logarit rời rạc***” chính là việc giải phương trình **x** = **log g β** (mod **p**) với ẩn **x**.

Hay phải tìm số **x** duy nhất sao cho: **g x** ≡ **β** (mod **p**).

***5). Thặng dư bậc hai và ký hiệu Legendre.***

***\* Thặng dư bậc hai:***

Cho **p** là số nguyên tố lẻ, x là một số nguyên dương ≤p-1.

**x** được gọi là “***thặng dư bậc hai***” mod **p**, nếu phương trình

**y 2** ≡ **x** mod **p** có lời giải.

***\* Kí hiệu Legendre:***

Cho p là số nguyên tố lẻ, và a là một số nguyên dương bất kỳ.

ký hiệu Legendre như sau:

a 0, nếu a ≡ 0 mod p

= 1, nếu a là thặng dư bậc hai mod p

b 1, trong các trường hợp còn lại.

**1.3. KHÁI NIỆM ĐỘ PHỨC TẠP CỦA THUẬT TOÁN**

1.3.1. Khái niệm Thuật toán

***1.3.1.1. Khái niệm Bài toán***

Bài toán được diễn đạt bằng hai phần:

***Input***: Các dữ liệu vào của bài toán.

***Output***: Các dữ liệu ra của bài toán (kết quả).

Không mất tính chất tổng quát, giả thiết các dữ liệu trong bài toán đều là số nguyên.

***1.3.1.2. Khái niệm Thuật toán***

”***Thuật toán***” được hiểu đơn giản là cách thức để giải một bài toán. Cũng có thể được hiểu bằng hai quan niệm: Trực giác hay Hình thức như sau:

**1). Quan niệm trực giác về** ”***Thuật toán***”.

Một cách trực giác, Thuật toán được hiểu là một dãy hữu hạn các qui tắc (Chỉ thị, mệnh lệnh) mô tả một quá trình tính toán, để từ dữ liệu đã cho (Input) ta nhận được

kết quả (Output) của bài toán.

**2). Quan niệm toán học về** ”***Thuật toán***”.

Một cách hình thức, người ta quan niệm thuật toán là một máy Turing.

***Thuật toán***  được chia thành hai loại: Đơn định và không đơn định.

**Thuật toán đơn định** (*Deterministic*):

Là thuật toán mà kết quả của mọi phép toán đều được xác định duy nhất.

**Thuật toán không đơn định** (*NonDeterministic*):

Là thuật toán có ít nhất một phép toán mà kết quả của nó là không duy nhất.

***1.3.1.3. Hai mô hình tính toán***

Hai quan niệm về thuật toán ứng với hai mô hình tính toán.

Ứng với hai mô hình tính toán có hai cách biểu diễn thuật toán.

**1). Mô hình ứng dụng:** Thuật toán được biểu diễn bằng ngôn ngữ tựa Algol.

+ Đơn vị nhớ: Một ô nhớ chứa trọn vẹn một dữ liệu.

+ Đơn vị thời gian: Thời gian để thực hiện một phép tính cơ bản trong số học hay logic như cộng, trừ, nhân, chia, ...

**2). Mô hình lý thuyết:** Thuật toán được biểu diễn bằng ngôn ngữ máy Turing.

+ Đơn vị nhớ: Một ô nhớ chứa một tín hiệu. Với mã nhị phân thì đơn vị nhớ là 1 bit.

+ Đơn vị thời gian: Thời gian để thực hiện một bước chuyển hình trạng

1.3.2. Khái niệm Độ phức tạp của thuật toán

***1). Chi phí của thuật toán*** *(Tính theo một bộ dữ liệu vào):*

Chi phí phải trả cho một quá trình tính toán gồm chi phí về thời gian và bộ nhớ.

***Chi phí thời gian*** của một quá trình tính toán là thời gian cần thiết để thực hiện một quá trình tính toán. Với thuật toán tựa Algol: Chi phí thời gian là số các phép tính cơ bản

thực hiện trong quá trình tính toán .

***Chi phí bộ nhớ***của một quá trình tính toán là số ô nhớ cần thiết để thực hiện một quá trình tính toán.

Gọi A là một thuật toán, e là dữ liệu vào của bài toán đã được mã hoá bằng cách nào đó. Thuật toán A tính trên dữ liệu vào e phải trả một giá nhất định. Ta ký hiệu:

**t A** (e) là giá thời gian và **l A**(e) là giá bộ nhớ.

***2). Độ phức tạp về bộ nhớ*** (*Trong trường hợp xấu nhất*):

**LA(n) = max{ l A (e), với |e| ≤ n},**  **n** là “kích thước” đầu vào của thuật toán.

***3). Độ phức tạp thời gian*** (*Trong trường hợp xấu nhất*):

**TA(n) = max { t A (e), với |e| ≤ n}**.

***4). Độ phức tạp tiệm cận*:** Độ phức tạp PT(n) được gọi là ***tiệm cận tới hàm f(n)***, ký hiệu O(f(n)) nếu ∃ các số n0 , c mà PT(n) ≤ c.f(n) , ∀n ≥ n0.

***5). Độ phức tạp đa thức:***

Độ phức tạp PT(n) được gọi ***đa thức***, nếu nó ***tiệm cận tới đa thức p(n)****.*

***6). Thuật toán đa thức***: Thuật toán được gọi là ***đa thức***, nếu độ phức tạp về thời gian (trong trường hợp xấu nhất) của nó là ***đa thức***.

**Nói cách khác**:

+ Thuật toán ***thời gian đa thức*** là thuật toán có độ phức tạp thời gian **O(n t ),**

trong đó **t** là hằng số.

+ Thuật toán ***thời gian hàm mũ*** là thuật toán có độ phức tạp thời gian **O(t f (n) )**,

trong đó **t** là hằng số và **f(n)** là đa thức của n.

\* ***Thời gian chạy của các lớp thuật toán khác nhau:***

|  |  |  |
| --- | --- | --- |
| Độ phức tạp | Số phép tính(n=106) | Thời gian(106ptính/s) |
| O(1) | 1 | 1micro giây |
| O(n) | 106 | 1 giây |
| O(n2) | 1012 | 11,6 ngày |
| O(n3) | 1018 | 32 000 năm |
| O(2n) | 10301 030 | 10301 006 tuổi của vũ trụ |

***Chú ý***

- Có người cho rằng ngày nay máy tính với tốc độ rất lớn, không cần quan tâm nhiều

tới thuật toán nhanh, chúng tôi xin dẫn một ví dụ đã được kiểm chứng.

- Bài toán xử lý n đối tượng, có ba thuật toán với 3 mức phức tạp khác nhau sẽ chịu

3 hậu quả như sau: ***Sau 1 giờ:***

Thuật toán A có độ phức tạp O(n) : 3,6 triệu đối tượng.

Thuật toán B có độ phức tạp O(n log n) : 0,2 triệu đối tượng.

Thuật toán C có độ phức tạp O(2n) : 21 đối tượng.

**1.3.3. Phân lớp bài toán theo độ phức tạp**

***1.3.3.1.*** ***Các khái niệm*.**

**1)*.* Khái niệm *"Dẫn về được".***

Bài toán **B** được gọi là "***Dẫn về được***” bài toán A một cách ***đa thức***, ký hiệu:

**B ∝ A,** nếu có thuật toán đơn định đa thức để giải bài toán A, thì cũng có thuật toán đơn định đa thức để giải bài toán **B**.

*Nghĩa là:* Bài toán A "khó hơn" bài toán **B**, hay **B** "dễ” hơn **A**, **B** được diễn đạt bằng ngôn ngữ của bài toán **A,** hay có thể hiểu **B** là trường hợp riêng của **A**.

Vậy nếu giải được bài toán **A** thì cũng sẽ giải được bài toán **B**.

Quan hệ ∝ có tính chất bắc cầu: Nếu C ∝ B và B ∝ A thì C ∝ A.

**2). Khái niệm *"Khó tương đương".***

Bài toán A gọi là “khó tương đương” bài toán B, ký hiệu **A ~ B**,

nếu : A ∝ B và B ∝ A

***1.3.3.2.*** ***Các lớp bài toán*.**

**1). Lớp bài toán P, NP**.

Ký hiệu:

**P** là lớp bài toán giải được bằng thuật toán đơn định, đa thức (Polynomial).

**NP** là lớp bài toán giải được bằng thuật toán không đơn định, đa thức.

Theo định nghĩa ta có P ⊂ NP.

Hiện nay người ta chưa biết được **P ≠ NP ?**

**2). Lớp Bài toán NP- Hard.**

Bài toán **A** được gọi là ***NP - hard*** (NP- khó) nếu ∀ **L** ∈ NP đều là  **L ∝ A**.

Lớp bài toán NP - hard bao gồm tất cả những bài toán NP - hard.

Bài toán NP – hard có thể nằm ***trong*** hoặc ***ngoài*** lớp NP.

**3). Lớp bài toán NP- Complete*.***

**(1) *Bài toán NP- Complete.***

Bài toán A được gọi là NP - Complete (NP- đầy đủ) nếu A là ***NP – Hard*** và A∈ **NP**.

Bài toán NP – Complete là bài toán NP - hard nằm trong lớp NP.

Lớp bài toán NP - Complete bao gồm tất cả những bài toán NP - Complete.

Lớp NP – Complete là có thực, vì Cook và Karp đã chỉ ra BT đầu tiên thuộc lớp này. đó là bài toán “thỏa được”: SATISFYABILITY.

**(2) *Chứng minh bài toán là NP – Hard.***

**Cách 1: Theo định nghĩa**

Bài toán **A** được gọi là ***NP - hard*** (NP- khó) nếu ∀ **L** ∈ NP đều là  **L ∝ A**.

Chứng minh theo định nghĩa gặp nhiều khó khăn vì phải chứng minh:

Mọi bài toán trong NP đều “***dễ hơn***” A.

Theo cách 1, năm 1971 Cook và Karp đã chỉ ra bài toán đầu tiên thuộc lớp

NP - Hard, đó là bài toán “thoả được”: SATISFYABILITY.

**Cách 2**

Để chứng minh bài toán **A** là NP – hard, trong thực tế người ta thường dựa vào bài toán **B** nào đó đã được biết là NP - Hard và chứng minh rằng **B ∝ A**.

Theo tính chất bắc cầu của quan hệ ***“dẫn về***”, A thoả mãn định nghĩa

NP -hard. Theo cách hiểu trực quan: **B** đã ***“khó”*** thì A càng ***“khó”***.

***1.3.3.3.***  ***Phân lớp các bài toán*.**

1). Cho một bài toán, có 2 khả năng xảy ra: Đã có lời giải hoặc chưa có lời giải.

2). Cho một bài toán đã có lời giải, có 2 khả năng xảy ra:

Giải được bằng thuật toán hay không giải được bằng thuật toán.

3). Cho một bài toán giải được bằng thuật toán, cũng chia thành hai loại:

“Thực tế giải được” và “Thực tế khó giải”.

- Bài toán “thực tế giải được” hiểu là nó ***có thể*** giải được bởi thuật toán, xử lý trong thời gian đủ nhanh, thực tế cho phép, đó là thuật toán có độ phức tạp thời gian là

“***đa thức***”. Bài toán này thuộc loại “***dễ giải***”.

- Bài toán “thực tế khó giải” hiểu là nó ***chỉ có thể*** giải được bởi thuật toán, xử lý trong nhiều thời gian, thực tế khó chấp nhận, đó là thuật toán có độ phức tạp thời gian là trên đa thức (“***hàm mũ***”). Bài toán này thuộc loại “***khó giải***”.

**1.3.4.**   **Hàm một phía và hàm cửa sập một phía.**

1). Hàm f(x) được gọi là ***hàm một phía*** nếu tính “***xuôi***” y = f(x) thì “***dễ***”,

nhưng tính “**ngược**” x = f -1 (y) lại rất “**khó**”.

*Ví dụ:*

Hàm f(x) = **g** x (mod p), với p là số nguyên tố lớn, (**g** là phần tử nguyên thuỷ mod p)

là hàm một phía.

2). Hàm f(x) được gọi là ***hàm cửa sập một phía*** nếu tính y = f(x) thì “dễ”,

tính x = f -1 (y) lại rất “**khó**”. Tuy nhiên có ***cửa sập*** **z** để tính x = f -1 (y) là “dễ”.

*Ví dụ:*

Hàm f(x) = **x a** (mod n) (với n là tích của hai số nguyên tố lớn n = p\*q) là

hàm một phía. Nếu chỉ biết a và n thì tính x = f-1(y) rất “**khó**”, nhưng nếu biết

***cửa sập*** p và q, thì tính được f-1(y) là khá “***dễ***”.

***Chương 2.***  **CÁC HỆ MÃ HÓA**

**2. 1. TỔNG QUAN VỀ MÃ HÓA DỮ LIỆU**

**2.1.1. Khái niệm** **Mã hóa dữ liệu**

Để bảo đảm **An toàn thông tin** (**ATTT**) lưu trữ trong máy tính (giữ gìn thông tin cố định) hay bảo đảm An toàn thông tin trên đường truyền tin (trên mạng máy tính), người ta phải “***Che Giấu***” các thông tin này.

“***Che***” thông tin (dữ liệu) hay “***Mã hóa*** ” thông tin là ***thay đổi hình dạng*** thông tin gốc (Giấu đi ý nghĩa nghĩa TT gốc), và người khác “**khó**” nhận ra.

“***Giấu***” thông tin (dữ liệu) là ***cất giấu*** thông tin trong bản tin khác, và người khác cũng “**khó**” nhận ra (Giấu đi sự hiện diện TT gốc).

Trong chương này chúng ta bàn về “***Mã hóa*** ” thông tin.

***1). Hệ mã hóa:***

Việc mã hoá phải theo quy tắc nhất định, quy tắc đó gọi là ***Hệ mã hóa***.

Hệ mã hóa được định nghĩa là bộ năm (**P**, **C**, **K**, **E**, **D**), trong đó:

**P** là tập hữu hạn các ***bản rõ*** có thể*.* **C** là tập hữu hạn các ***bản mã***  có thể.

**K**là tập hữu hạn các ***khoá*** có thể*.*

**E** là tập các hàm lập mã*.* **D**là tập các hàm giải mã*.*

Với khóa lập mã **ke** ∈**K***,*  có hàm lập mã **eke**∈**E**, **eke**:**P**→ **C***,*

Với khóa giải mã **kd** ∈**K***,* có hàm giải mã **dkd** ∈**D**, **dkd**:**C***→* **P**,

sao cho **dkd (eke (x)) = x***,* ∀ **x** ∈ **P**.

Ơ đây **x** được gọi là ***bản rõ***, **eke (x)** được gọi là ***bản mã***.

***2). Mã hóa và Giải mã:***

|  |
| --- |
| Người gửi G → → **eke** (**T**) → → Người nhận N  (có khóa lập mã **ke**) (có khóa giải mã **kd**)  ↑  Tin tặc có thể trộm bản mã **eke** (**T**) |

Người gửi G muốn gửi bản tin **T** cho người nhận N. Để bảo đảm bí mật, G

mã hoá bản tin bằng khóa lập mã **ke**, nhận được bản mã **eke** (**T**), sau đó gửi cho N.

Tin tặc có thể trộm bản mã **eke** (**T**), nhưng cũng “***khó***”  hiểu được bản tin gốc **T** nếu không có khoá giải mã **kd**.

Người N nhận được bản mã, họ dùng khoá giải mã **kd**, để giải mã **eke** (**T**), sẽ nhận được bản tin gốc **T** = **dkd (eke (T))**.

**2.1.2.**  **Phân loại hệ mã hóa**

Có nhiều mã hoá tùy theo cách phân loại, sau đây xin giới thiệu một số cách.

***Cách 1: Phân loại mã hoá theo*  đoi xưng của khoá.**

**Hệ mã hóa khóa đối xứng** (Mã hoá khoá riêng, bí mật).

**Hệ mã hóa khóa phi đối xứng (Khóa công khai).**

Hiện có 2 loại mã hóa chính: mã hóa khóa đối xứng và mã hóa khoá công khai.

**Hệ mã hóa khóa đối xứng** có khóa lập mã và khóa giải mã “đối xứng nhau”, theo nghĩa biết được khóa này thì “***dễ***” tính được khóa kia. Vì vậy phải giữ bí mật cả 2 khóa.

**Hệ mã hóa khóa công khai** có khóa lập mã khác khóa giải mã (**ke** ≠ **kd**), biết được khóa này cũng “***khó***” tính được khóa kia. Vì vậy chỉ cần bí mật khóa giải mã, còn công khai khóa lập mã.

***Cách 2: Phân loại mã hoá theo*  đặc trưng xử lý bản rõ.**

Mã hoá khối, Mã hoá dòng

***Cách 3: Phân loại mã hoá theo*  ứng dụng đặc trưng.**

Mã hoá đồng cấu, Ma hoa xac suat, Ma hoa tat dinh

***2.1.2.1. Hệ mã hóa khóa đối xứng***

***Mã hóa khóa đối xứng*** là Hệ mã hóa mà biết được khóa lập mã thì có thể “***dễ***” tính được khóa giải mã và ngược lại. Đặc biệt một số Hệ mã hóa có khoá lập mã và khoá giải mã trùng nhau (**ke** = **kd**), như Hệ mã hóa “dịch chuyển” hay DES.

Hệ mã hóa khóa đối xứng còn gọi là ***Hệ mã hóa khoá bí mật***, hay ***khóa riêng***, vì phải giữ bí mật cả 2 khóa. Trước khi dùng Hệ mã hóa khóa đối xứng, người gửi và người nhận phải thoả thuận thuật toán mã hóa và ***khoá chung*** (lập mã hay giải mã), khoá phải được giữ bí mật. Độ an toàn của Hệ mã hóa loại này ***phụ thuộc vào khoá***.

***Ví dụ:***

***+ Hệ mã hóa cổ điển*** là Mã hóa khóa đối xứng: dễ hiểu, dễ thực thi, nhưng có độ an toàn không cao. Vì giới hạn tính toán chỉ trong phạm vi bảng chữ cái, sử dụng trong bản tin cần mã, ví dụ là Z26 nếu dùng các chữ cái tiếng Anh. Với hệ mã hóa cổ điển, nếu biết khoá lập mã hay thuật toán lập mã, có thể “dễ” xác định được bản rõ, vì “dễ” tìm được khoá giải mã.

+ ***Hệ mã hóa* DES** (1973) là Mã hóa khóa đối xứng ***hiện đại***, có độ an toàn cao.

**a). Đặc điểm của Hệ mã hóa khóa đối xứng.**

***Ưu điểm***:

Hệ mã hóa khóa đối xứng mã hóa và giải mã ***nhanh hơn*** Hệ mã hóa khóa công khai.

***Hạn chế***:

1). Mã hóa khóa đối xứng chưa thật an toàn với lý do sau:

Người mã hoá và người giải mã phải có “***chung***” **một khoá**. Khóa phải được giữ bí mật tuyệt đối, vì biết khoá này “***dễ***” xác định được khoá kia và ngược lại.

2). Vấn đề thỏa thuận khoá và quản lý khóa chung là khó khăn và phức tạp. Người gửi và người nhận phải luôn thống nhất với nhau về khoá. Việc thay đổi khoá là rất khó và dễ bị lộ. Khóa chung phải được gửi cho nhau trên kênh an toàn.

Mặt khác khi hai người (lập mã, giải mã) cùng biết “chung” một bí mật, thì càng khó giữ được bí mật !

**b). Nơi sử dụng Hệ mã hóa khóa đối xứng.**

Hệ mã hóa khóa đối xứng thường được sử dụng trong môi trường mà khoá chung có thể dễ dàng trao chuyển bí mật, chẳng hạn trong cùng một mạng nội bộ.

Hệ mã hóa khóa đối xứng thường dùng để mã hóa những bản tin lớn, vì tốc độ

mã hóa và giải mã nhanh hơn Hệ mã hóa khóa công khai.

***2.1.2.2. Hệ mã hóa khóa công khai***

**Hệ mã hóa khóa phi đối xứng** là Hệ mã hóacó khóa lập mã và khóa giải mã khác nhau (**ke** ≠ **kd**), biết được khóa này cũng “**khó**” tính được khóa kia.

Hệ mã hóa này còn được gọi là **Hệ mã hoá khóa công khai**, vì:

***Khoá lập mã*** cho ***công khai***, gọi là ***khoá công khai***  *(****Public key****)*.

***Khóa giải mã*** giữ bí mật, còn gọi là ***khóa riêng*** *(****Private key****)* hay***khóa bí mật***.

Một người bất kỳ có thể dùng khoá công khai để mã hoá bản tin, nhưng chỉ người nào có đúng khoá giải mã thì mới có khả năng đọc được bản rõ.

***Hệ mã hóa khoá công khai***hay ***Hệ mã hóa phi đối xứng*** do Diffie và Hellman phát minh vào những năm 1970.

**a). Đặc điểm của Hệ mã khoá công khai.**

***Ưu điểm***:

1). Hệ mã hóa khóa công khai có ưu điểm chủ yếu sau:

Thuật toán được viết một lần, công khai cho nhiều lần dùng, cho nhiều người dùng,

họ chỉ cần giữ bí mật khóa riêng của mình.

2). Khi biết các tham số ban đầu của hệ mã hóa, việc tính ra cặp khoá công khai và bí mật phải là “dễ”, tức là trong thời gian đa thức.

Người gửi có bản rõ P và khoá công khai, thì “dễ” tạo ra bản mã C.

Người nhận có bản mã C và khoá bí mật, thì “dễ” giải được thành bản rõ P.

3). Người mã hoá dùng khóa công khai, người giải mã giữ khóa bí mật. Khả năng lộ khóa bí mật khó hơn vì chỉ có một người giữ gìn.

Nếu thám mã biết khoá công khai, cố gắng tìm khoá bí mật, thì chúng phải đương đầu với bài toán “khó”.

4). Nếu thám mã biết khoá công khai và bản mã C, thì việc tìm ra bản rõ P cũng là bài toán “khó”, số phép thử là vô cùng lớn, không khả thi.

***Hạn chế***:

Hệ mã hóa khóa công khai: mã hóa và giải mã ***chậm hơn*** hệ mã hóa khóa đối xứng.

**b). Nơi sử dụng Hệ mã hóa khoá công khai.**

Hệ mã hóa khóa công khai thường được sử dụng chủ yếu trên các mạng công khai như Internet, khi mà việc trao chuyển khoá bí mật tương đối khó khăn.

Đặc trưng nổi bật của hệ mã hoá công khai là khoá công khai (public key) và bản mã (ciphertext) đều có thể gửi đi trên một kênh truyền tin  ***không an toàn***.

Có biết cả khóa công khai và bản mã, thì thám mã cũng không dễ khám phá được bản rõ.

Nhưng vì có tốc độ mã hóa và giải mã ***chậm***, nên hệ mã hóa khóa công khai chỉ dùng để mã hóa những bản tin ngắn, ví dụ như mã hóa khóa bí mật gửi đi.

Hệ mã hóa khóa công khai thường được sử dụng cho cặp người dùng thỏa thuận khóa bí mật của Hệ mã hóa khóa riêng.

**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 21, 22

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 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 | 23 | 24 | 25 | 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

***Ví dụ***

\* Bản rõ chữ: **T O I N A Y T H A V I R U S**

\* Chọn khóa **k** *=* **3***.*

\* Bản rõ số: **19 14 8 26 13 0 24 26 19 7 0 26 21 8 17 20 18**

\* Với phép mã hóa **y** = **ek**(**x**) = (**x** + **k**) mod 26 = (**x** + **3**) mod 26, ta nhận được:

\* Bản mã số: **22 17 11 3 16 3 1 3 22 10 3 3 24 11 20 23 21**

\* Bản mã chữ: **W R L D Q D B D W K D D Y L U X V**

* Với phép giải mã **x** = **dk**(**y**) = (**y** – **k**) mod 26 = (**y** – **3**) mod 26, ta nhận lại được bản rõ số, sau đó là bản rõ chữ.

***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)**

***Ví dụ***

\* Bản rõ chữ: **T O I N A Y T H A V I R U S**

**\*** Chọn khóa **k** *=* **π** là hoán vị:

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

\* Mã hóa theo công thức **y** = **eπ (x) = π (x)**:

\* Bản mã chữ: **E J P Z K Y V Z E Q Y Z C P G D F**

\* Giải mã theo công thức **x =** **dπ (y)** = **π -1** **(y),** ta nhận lại được bản rõ chữ.

***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

***Ví dụ***

\* Bản rõ chữ: **CHIEUNAYOVUONHOA**

\* Chọn khóa **k** = (**a**, **b**) = (**3**, 6).

\* Bản rõ số: **x** = 2 7 8 4 20 13 **0** 24 14 21 20 14 13 7 14 **0**

Mã hóa theo công thức **y** = **ek**(**x**) = (**a** **x** + **b**) mod 26 = (**3** **x** + **6**) mod 26

\* Bản mã số: **y** = 12 1 4 18 14 19 **6** 0 22 17 14 22 19 1 22 **6**

\* Bản mã chữ: **MBESOTGAWROWTBWG**

Giải mã theo công thức **x** = **dk**(**y**) = **a** **-1**(**y – b**) mod 26

= **3** **-1**(**y – 6**) mod 26 = 9 \* (**y – 6**) 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**.

***Ví dụ***

\* Bản rõ chữ **CX = SHESEL ISSEAS HELLSB YTHESE ASHO**

Đặt **P** = **C** = **Z26 m** , trong đó **m** = **6**.

Chọn khoá **k** là một hoán vị **π** của (1, 2, 3, 4, 5, 6):

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 3 | 5 | 1 | 6 | 4 | 2 |

Hoán vị ngược là **π**-1 là :

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 1 | 2 | 3 | 4 | 5 | 6 |
| 3 | 6 | 1 | 5 | 2 | 4 |

\* Mã hóa: Tách bản rõ thành từng nhóm 6 kí tự:

**SHESEL | ISSEAS | HELLSB | YTHESE | ASHORE**

Với mỗi nhóm 6 ký tự, sắp xếp lại các chữ theo hoán vị **π**, ta nhận được:

**EESLSH | SALSES | LSHBLE | HSYEET | HRAEOS**

\* Bản mã chữ: **CY** = **EESLSHSALSES LSHBLEHSYEETHRAE**

\* Dùng hoán vị ngược **π**-1, ta sẽ thu được bản rõ **CX**.

***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.

**2. 3. HỆ MÃ HÓA ĐỐI XỨNG DES**

**2. 3.1. Hệ mã hoá DES**

***2.3.1.1. Giới thiệu***

Hiện nay có nhiều hệ mã hóa đối xứng loại mới, mục này trình bày Chuẩn mã

hóa dữ liệu **DES** (Data Encryption Standard).

15/05/ 1973, Uỷ ban tiêu chuẩn quốc gia Mỹ (NBS) (được sự thẩm định của Cục an ninh QG (NAS) đã công bố một khuyến nghị về hệ mã hoá chuẩn.

- Hệ mã hoá phải có độ an toàn cao.

- Hệ mã hoá phải được định nghĩa đầy đủ và dễ hiểu.

- Độ an toàn của Hệ mã hoá phải phải nằm ở Khoá, không nằm ở thuật toán.

- Hệ mã hoá phải sẵn sàng cho mọi người dùng ở các lĩnh vực khác nhau.

- Hệ mã hoá phải xuất khẩu được.

DES được IBM phát triển, là một cải biên của hệ mật LUCIPHER DES, nó

được công bố lần đầu tiên vào ngày 17/03/1975. Sau nhiều cuộc tranh luận công khai,

cuối cùng DES được công nhận như một chuẩn liên bang vào ngày 23/11/1976 và

được công bố vào ngày 15/01/1977.

Năm 1980, “Cách dùng DES ” được công bố. Từ đó chu kỳ 5 năm DES được

xem xét lại một lần bởi Uỷ ban tiêu chuẩn quốc gia Mỹ, lần gần đây nhất là 2009.

***2.3.1.2. Qui trình mã hóa theo DES***.

Giai đoạn 1 : Bản Rõ chữ =====🡺 Bản Rõ số (Dạng nhị phân)

Chia thành

Giai đoạn 2 : Bản Rõ số =====🡺 Các đoạn 64 bit Rõ số

***Giai đoạn 3*** : 64 bit Rõ số =====🡺 64 bit Mã số

Kết nối

Giai đoạn 4 : Các đoạn 64 bit Mã số =====🡺 Bản Mã số (Dạng nhị phân)

Giai đoạn 5 : Bản Mã số =====🡺 Bản Mã chữ

**2.3.2. Lập mã và Giải mã DES**

***2.3.2.1. Qui trình lập mã DES***

Thuật toán DES tập trung thực hiện ***Giai đoạn 3*** .của qui trình mã hóa.

Đó là chuyển đổi bản rõ số với ***64*** bit thành bản mã với ***64*** bit.

***Sơ đồ***

IP

**Bản mã**: 64 bit

**L16 = R15**

**R16 = L15f (R15, k16)**

**IP-1**

**f**

**R15 =L14f(R14, k15)**

**L15 = R14**

**R1 = L0f ( R0, k1)**

**L1 = R0**

**f**

**R2 = L1f ( R1, k2)**

**L2 = R1**

**L0**

**R0**

**Bản rõ**:64 bit,K

**f**

**k1**

**k2**

**k16**

***2.3.2.2. Thực hiện mã hóa DES theo Sơ đồ***

\* Bản rõ là xâu **x** , Bản mã là xâu **y**, Khoá là xâu **K**, đều có độ dài 64 bit.

\* Thuật toán mã hóa DES thực hiện qua *3 bước* chính như sau:

***Bước 1***: Bản rõ **x** được hoán vị theo phép hoán vị **IP**, thành **IP** (**x**).

**IP** (**x**) = **L0 R0**, trong đó **L0** là 32 bit đầu (Left), **R0** là 32 bit cuối (Right).

(**IP** (**x**) tách thành **L0 R0**).

***Bước 2***: Thực hiện **16** vòng mã hoá với những phép toán giống nhau.

Dữ liệu được kết hợp với khoá thông qua hàm **f** :

**L i** = **R i -1**, **R i**= **L i -1 ⊕ f (R i -1 , k i ),**  trong đó:

**⊕** là phép toán ***hoặc loại trừ*** của hai xâu bit (cộng theo modulo 2).

**k1, k2, ..., k16** là các ***khoá con*** (48 bit) được tính từ khóa gốc **K**.

***Bước 3***: Thực hiện phép hoán vị ngược **IP-1** cho xâu **R16L16** , thu được bản mã **y**.

**y** = **IP -1**(**R16 , L16**). (Lưu ý thứ tự bit **R16** và **L16**)

\* Bảng hoán vị ban đầu **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 |

+ bit 1 của **IP**(x) là bit **58** của x.

+ bit 2 của **IP**(x) là bit **50** của x.

\* Bảng hoán vị cuối cùng **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 |

***2.3.2.3. Tính các khóa con*** **k1 , k2, … , k16** ***từ khóa gốc*** **K**.

***Sơ đồ***

K

**PC - 1**

|  |  |
| --- | --- |
| **C0** | **D0** |

**LS1**

**LS1**

**LS2**

**LS2**

**PC - 2**

**k1**

**PC - 2**

**k2**

|  |  |
| --- | --- |
| **C1** | **D1** |

|  |  |
| --- | --- |
| **C2** | **D2** |

**……………………………**

**LS16**

**LS16**

**PC - 2**

**k16**

|  |  |
| --- | --- |
| **C 16** | **D 16** |

\****Tính khoá*** **k i** (**48** bit):

1). Khoá **K** là xâu dài 64 bit, trong đó 56 bit là khoá và 8 bit để kiểm tra tính chẵn lẻ nhằm phát hiện sai, các bit này không tham gia vào quá trình tính toán.

Các bit kiểm tra tính chẵn lẻ nằm ở vị trí 8, 16, 24,…, 64 được xác định, sao cho mỗi byte chứa ***một số lẻ*** các số **1**. Bởi vậy mỗi sai sót đơn lẻ được xác định trong mỗi nhóm 8 bit.

2). Tính khoá **k i** như sau:

+ Với khoá **K** độ dài 64 bit, ta loại bỏ các bit kiểm tra tính chẵn lẻ, hoán vị 56 bit còn lại theo phép hoán vị **PC-1**:

**PC-1 (K )** = **C0 D0**

Trong đó **C0** là 28 bit đầu, **D­0** là 28 bit cuối cùng của **PC-1( K )**.

+ Với i = 1, 2, ... , 16, ta tính: **Ci** = **LSi** ( **Ci-1** ), **Di** = **LSi**( **Di-1** ).

Trong đó **LS i**  là phép chuyển dịch vòng sang trái:

Dịch **1 *vị trí***  nếu i = 1, 2, 9, 16. Dịch **2 *vị trí***  với những giá trị **i** khác.

+ Với i = 1, 2, ... , 16, khóa **k i** được tính theo phép hoán vị **PC-2** từ **Ci Di**:

**k i**  = **PC-2** (**Ci Di** ) (**48**  bit).

\* Phép hoán vị **PC - 1**: \* Phép hoán vị **PC - 2**:

***2.3.2.4. Tính*** ***hàm***  **f (R i -1 , k i )**

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

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

***Sơ đồ***

R**i-1**

## k i

**E**(R**i-1**)

f(Ri-1, ki)

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **B1** | **B2** | **B3** | **B4** | **B5** | **B6** | **B7** | **B8** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **S1** | **S2** | **S3** | **S4** | **S5** | **S6** | **S7** | **S8** |

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| **C1** | **C2** | **C3** | **C4** | **C5** | **C6** | **C7** | **C8** |

***\* Tính*** ***hàm***  **f (R i -1 , k i )**

Để cho đơn giản, ta không ghi chỉ số **i-1**, **i**, và mô tả cách tính **f (R, k)**:

1). Mở rộng xâu **R** (32 bit) thành xâu 48 bit, theo hàm mở rộng **E**:

**E: R** (32 bit) **--- > E(R)** (48 bit).

**E(R)** gồm 32 bit của cũ của **R** và 16 bit của **R** xuất hiện lần thứ 2.

2). Tính **E**(**R**) ⊕ **k**, trong đó **E(R)** (48 bit) và **k** (48 bit).

Kết quả gồm 8 xâu **Bj**, mỗi xâu **Bj** có 6 bit (8\*6 = 48):

**B** = **B1 B2 B3 B4 B5 B6 B7 B8**.

3). Tính **Cj** = **Sj**  (**Bj**), j = 1,… , 8. Dùng 8 bảng **S1, S2, …, S8**.

**Sj** là bảng cố định với **r** \* **c** số nguyên từ 0 -> 15, (0  **r** 3, 0  **c**  15).

**Sj** thể hiện việc thay thế mỗi **Bj** thành **Cj** (**Cj**  là xâu 4 bit) theo qui tắc sau:

\* Giả sử **Bj** = **b1 b2 b3 b4 b5 b6**. (6 bit).

+ **b1 b6** xác định biểu diển nhị phân của hàng **r** trong **Sj** (0  **r** 3 ).

+ **b2 b3 b4 b5** xác định biểu diển nhị phân của cột **c** trong **Sj** (0  **c**  15 ).

Xâu **Cj** (4 bit) được định nghĩa là biểu diển nhị phân của phần tử **S j**(**r, c**).

4). Thực hiện 8 lần bước 3), ta nhận được xâu **C** = **C1 C2 … C8**  (32 bit).

Sau hoán vị **P**, cho kết quả **P** (**C**), đó chính là **f** (**R**, **k**).

\* Phép hoán vị mở rộng **E**: \* Phép hoán vị **P**:

32 1 2 3 4 5

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

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

\* ***Các bảng*** **S1 , S2 , … , S8**:

**S1**

1 6 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 |14 4 13 1 2 15 11 8 3 10 6 12 **5** 9 0 7

0 1 | 0 15 7 4 14 2 13 1 10 6 12 11 9 5 3 8

1 0 | 4 1 14 8 13 6 2 11 15 12 9 7 3 10 5 0

1 1 |15 12 8 2 4 9 1 7 5 11 3 14 10 0 6 13

**S2**

7 12 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 |15 1 8 14 6 11 3 4 9 7 2 13 12 0 5 10

0 1 | 3 13 4 7 15 2 8 14 12 0 1 10 6 9 11 5

1 0 | 0 14 7 11 10 4 13 1 5 8 12 6 9 3 2 15

1 1 |13 8 10 1 3 15 4 2 11 6 7 12 0 5 14 9

**S3**

13 18 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 |10 0 9 14 6 3 15 5 1 13 12 7 11 4 2 8

0 1 |13 7 0 9 3 4 6 10 2 8 5 14 12 11 15 1

1 0 |13 6 4 9 8 15 3 0 11 1 2 12 5 10 14 7

1 1 | 1 10 13 0 6 9 8 7 4 15 14 3 11 5 2 12

**S4**

19 24 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 | 7 13 14 3 0 6 9 10 1 2 8 5 11 12 4 15

0 1 |13 8 11 5 6 15 0 3 4 7 2 12 1 10 14 9

1 0 |10 6 9 0 12 11 7 13 15 1 3 14 5 2 8 4

1 1 | 3 15 0 6 10 1 13 8 9 4 5 11 12 7 2 14

**S5**

25 30 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 | 2 12 4 1 7 10 11 6 8 5 3 15 13 0 14 9

0 1 |14 11 2 12 4 7 13 1 5 0 15 10 3 9 8 6

1 0 | 4 2 1 11 10 13 7 8 15 9 12 5 6 3 0 14

1 1 |11 8 12 7 1 14 2 13 6 15 0 9 10 4 5 3

**S6**

31 36 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 |12 1 10 15 9 2 6 8 0 13 3 4 14 7 5 11

0 1 |10 15 4 2 7 12 9 5 6 1 13 14 0 11 3 8

1 0 | 9 14 15 5 2 8 12 3 7 0 4 10 1 13 11 6

1 1 | 4 3 2 12 9 5 15 10 11 14 1 7 6 0 8 13

**S7**

37 42 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|------------------------------------------------------------------------------------

0 0 | 4 11 2 14 15 0 8 13 3 12 9 7 5 10 6 1

0 1 | 13 0 11 7 4 9 1 10 14 3 5 12 2 15 8 6

1 0 | 1 4 11 13 12 3 7 14 10 15 6 8 0 5 9 2

1 1 | 6 11 13 8 1 4 10 7 9 5 0 15 14 2 3 12

**S8**

43 48 | 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15

|-----------------------------------------------

0 0 |13 2 8 4 6 15 11 1 10 9 3 14 5 0 12 7

0 1 | 1 15 13 8 10 3 7 4 12 5 6 11 0 14 9 2

1 0 | 7 11 4 1 9 12 14 2 0 6 10 13 15 3 5 8

1 1 | 2 1 14 7 4 10 8 13 15 12 9 0 3 5 6 11

***\* Qui định lập bảng*** **Sj**:

- Mỗi hàng của bảng **S** phải là một hoán vị của 0, 1, …,15.

- Không có bảng S nào là hàm tuyến tính hay Apphin của các đầu vào của nó.

- Thay đổi 1 bit vào ở một bảng S, sẽ gây ra sự thay đổi ít nhất 2 bit ra của nó.

- Nếu 2 xâu vào của một bảng S giống nhau ở 2 bit đầu và 2 bit cuối,

thì 2 xâu ra phải khác nhau ít nhất tại 2 bit.

- Nếu 2 xâu vào của một bảng S khác nhau ở 2 bit đầu và giống nhau ở 2 bit cuối,

thì 2 xâu ra phải khác nhau.

- Với mỗi bảng S, nếu cố định 1 bit vào xét giá trị của 1 bit ra nào đó,

thì số các xâu vào tạo ra giá trị 0 ở bit ra đó cũng phải xấp xỉ bằng số các xâu vào

tạo ra giá trị 1 ở bit ra đó.

***2.3.2.5. Qui trình giải mã DES***

Qui trình giải mã của DES tương tự như qui trình lập mã, nhưng theo

dùng các khóa thứ tự ngược lại: **k16 , k15, … , k1** .

Xuất phát (đầu vào) từ bản mã **y**, kết quả (đầu ra) là bản ró **x**.

***2.3.2.6. Ví dụ***

Bản rõ **X** = **0123456789ABCDEF** = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001 1010 1011 1**1**00 1101 1**1**10 1111 **50 58**

***Bước 1***: Bản rõ **x** được hoán vị theo phép hoán vị **IP**, thành **IP** (**x**).

**IP** (**x**) = **L0 R0**, trong đó **L0** là 32 bit đầu (Left), **R0** là 32 bit cuối (Right).

(**IP** (**x**) tách thành **L0 R0**).

**L**0  = **11**00 1100 0000 0000 1100 1001 1111 1111 (32 bit).

**R**0  = 1111 0000 1010 1010 1111 0000 1010 1010 (32 bit).

Ví dụ: theo hoán vị **IP**, bit **1** của **L**0 là bit **58** của **x**, bit **2** của **L**0  là bit **50** của **x**.

***Bước 2***: Thực hiện **16** vòng mã hoá với những phép toán giống nhau.

Dữ liệu được kết hợp với khoá thông qua hàm **f** :

**L i** = **R i -1**, **R i**= **L i -1 ⊕ f (R i -1 , k i ),**  trong đó:

**k1, k2, ..., k16** là các ***khoá con*** (48 bit) được tính từ khóa gốc **K**.

***a). Tính khóa con*** **k1**(48 bit)từ khóa gốc **K** = **133457799BBCDFF1** (64 bit)=

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

\* Hoán vị **PC-1**: **K** 🡪 **C**0 **D**0 (Từ **K** qua PC-1, nhận được **C**0 **D**0).

**C**0  = 1111000 0110011 0010101 0101111 (28 bit)

**D**0  = 0101010 1011001 1001111 0001111 (28 bit)

**C1** = LS1(**C**0 ) =1110000 1100110 0101010 1011111 (28 bit)

**D1** = LS1(**D**0 ) =1010101 0110011 0011110 0011110 (28 bit)

\* Hoán vi **PC-2**: **C1 D1**  🡪 **k1**(48 bit)

**k1** = **000110 110000 001011 101111 111111 000111 000001 110010**

***b). Tính*** ***hàm***  **f (R 0 , k1 )**

+ Theo bước 1: **R**0  = 1111 0000 1010 1010 1111 0000 1010 1010 (32 bit).

1). Mở rộng xâu **R**0 (32 bit) thành xâu **E**(**R**0 ) (48 bit), theo hàm mở rộng **E**:

+ Hoán vị **E**: **R**0 🡪 **E**(**R**0 ):

**E**(**R**0 ) = 011110 100001 010101 010101 011110 100001 010101 010101 (48 bit).

+ Theo a):

**k1** = 000110 110000 001011 101111 111111 000111 000001 110010 (48 bit).

2). Tính  **E**(**R**0 ) ⊕ **k1**= **B1 B2 B3 B4 B5 B6  B7 B8** (48 bit)

011000 010001 100010 110010 100001 100110 010100 100111

3). Tính **C1** = **S1**  (**B1**), dùng bảng **S1**.

**S1** thể hiện việc thay thế **B1** (6 bit) thành **C1** (4 bit) theo qui tắc sau:

**B1** = **b1 b2 b3 b4 b5 b6** = 011000

+ **b1 b6** = (**00**)2  = (**00**)10 = Hàng **0** trong **S1**.

+ **b2 b3 b4 b5** = (**1100**)2 = (**12**)10 = Cột **12** trong **S1**.

Xâu **C1** (4 bit) được định nghĩa là biểu diển nhị phân của phần tử **S 1**(**0, 12**).

**C1** = **S 1**(**0, 12**) = (**5**)10 = (**0101**)2

+ Tương tự ta tính được **Cj** , j = 2, 3, …, 8.

4). Thực hiện 8 lần 3), ta nhận được xâu **C** = **C1 C2 … C8**  (32 bit).

**C** = 0101 1100 1000 0010 1011 0101 1001 0111

Sau hoán vị **P**, cho kết quả **P** (**C**), đó chính là **f** (**R**0, **k1**).

**f** (**R**0, **k1**) = **P**(**C**) = 0010 0011 0100 1010 1010 1001 1011 1011

***Bước 3***: Kết quả là bản mã **85E813540F0AB405**

**2.3.3. Độ an toàn của Hệ mã hóa DES**

1). Độ an toàn của Hệ mã hóa DES có liên quan đến các bảng **Sj**:

Ngoại trừ các bảng **S**, mọi tính toán trong DES đều tuyến tính, tức là việc tính phép hoặc loại trừ của hai đầu ra cũng giống như phép hoặc loại trừ của hai đầu vào, rồi tính toán đầu ra.

Các bảng S chứa đựng nhiều thành phần phi tuyến của hệ mật, là yếu tố quan trọng nhất đối với độ mật của hệ thống.

Khi mới xây dựng hệ mật DES, thì tiêu chuẩn xây dựng các hộp S không được biết đầy đủ. Và có thể các hộp S này có thể chứa các “***cửa sập***” được giấu kín. Và đó cũng là một điểm đảm bảo tính bảo mật của hệ DES.

2). Hạn chế của DES chính là kích thước không gian khoá:

Số khóa có thể là **256** , không gian này là nhỏ để đảm bảo an toàn thực sự. Nhiều thiết bị chuyên dụng đã được đề xuất nhằm phục vụ cho phép tấn công với bản rõ đã biết. Phép tấn công này chủ yếu thực hiện theo phương pháp “vét cạn”.

Tức là với bản rõ **x** và bản mã **y** tương ứng (64 bit), mỗi khoá có thể đều được kiểm tra cho tới khi tìm được một khoá **K** thoả mãn **eK** (**x**) = **y**.

**2. 4. HỆ MÃ HÓA KHÓA CÔNG KHAI**

**2.4. 1. Hệ mã hóa RSA.**

***Sơ đồ*** (Rivest, Shamir, Adleman đề xuất năm 1977)

***1/.Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **b*)*** :

Chọn bí mật số nguyên tố lớn **p**, **q**, tính **n** = **p** \* **q**, công khai **n**, đặt **P** = **C** = **Zn**

Tính bí mật **φ**(**n**) = (**p-1**).(**q-1**). Chọn khóa công khai **b** < **φ**(**n**),nguyên tố cung nhau với **φ(n)**.

Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod **φ**(n): **a**\***b** ≡ 1 (mod φ(n).

Tập cặp khóa (bí mật, công khai) **K** = {(**a**, **b**)/ a, b ∈ Zφ(n) , **a**\***b** ≡ 1 (mod φ(n))}.

Với  ***Bản rõ* x** ∈ **P** và ***Bản mã* y** ∈ **C**, định nghĩa:

2/. ***Mã hoá***: **y** = **ek** (**x**) = **x** **b** mod **n**

3/. ***Giải mã***: **x** = **dk** (**y**) = **y** **a** mod **n**

***Ví dụ***

\* Bản rõ chữ: **R E N A I S S A N C E**

\****Sinh khóa***:

Chọn bí mật số nguyên tố **p**= 53, **q**= 61, tính **n** = **p** \* **q** = **3233**, công khai **n**.

Đặt **P** = **C** = **Zn** , tính bí mật **φ(n)** = (**p-1**). (**q-1**) = 52 \* 60 = 3120.

+ Chọn khóa công khai **b** là nguyên tố với **φ(n)**, tức là ƯCLN(**b**, **φ(n)**) = 1,

ví dụ chọn **b** = 71.

+ Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod φ(n): **a**\***b** ≡ 1 (mod φ(n)).

Từ **a**\***b** ≡ 1 (mod **φ**(n)), ta nhận được khóa bí mật **a** = **791**.

\* Bản rõ số:

R E N A I S S A N C E (Dấu cách)

**17 04 13 00 08 18 18 00 13 02 04 26**

**m1 m2 m3 m4 m5 m6**

\* Theo phép lập mã: **ci** = **mi** **b** mod **n** = **mi** **71** mod **3233**, ta nhận được:

\* Bản mã số:

**c1 c2 c3 c4 c5 c6**

**3106 0100 0931 2691 1984 2927**

\* Theo phép giải mã: **mi** = **ci** **a** mod **n** = **ci** **791** mod 3233, ta nhận lại bản rõ.

***Độ an toàn***

1). Hệ mã hóa RSA là tất định, tức là với một bản rõ **x** và một khóa bí mật **a**, thì chỉ có một bản mã **y**.

2). Hệ mật RSA an toàn, khi giữ được bí mật khoá giải mã **a**, p, q, φ(n).

Nếu biết được p và q, thì thám mã dễ dàng tính được φ(n) = (q-1)\*(p-1).

Nếu biết được φ(n), thì thám mã sẽ tính được **a** theo thuật toán Euclide mở rộng.

Nhưng phân tích n thành tích của p và q là bài toán “khó”.

Độ an toàn của Hệ mật RSA dựa vào khả năng giải bài toán phân tích số nguyên dương n thành tích của 2 số nguyên tố lớn p và q.

**2. 4. 2. Hệ mã hóa Elgamal.**

***Sơ đồ*** (Elgamal đề xuất năm 1985)

1/. ***Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **h*)*** :

Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong **Zp** là “khó” giải. Chọn phần tử nguyên thuỷ **g** ∈ **Zp**\* . Đặt **P** = **Z p**\*, **C** = **Z p**\*  **Z p**\*.

Chọn khóa bí mật là **a** ∈ **Zp**\* . Tính khóa công khai **h** ≡ **g a** mod **p**.

Định nghĩa tập khóa:  = {(**p**, **g**, **a**, **h**): **h** ≡ **g a** mod **p**}.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

Với  ***Bản rõ* x** ∈ **P** và ***Bản mã* y** ∈ **C**, với khóa **k** ∈định nghĩa:

2/. ***Lập mã***: Chọn ngẫu nhiên bí mật **r** ∈ **Zp-1**, bản mã là **y** = **ek** (**x**, **r**) = (**y1, y2**)

Trong đó **y1** = **g** **r** mod **p** và **y2** = **x** \* **h****r**  mod **p**

3/. ***Giải mã***: **dk** (**y1, y2**) = **y2**(**y1 a**) **-1** mod **p**= **x**.

***Ví dụ*** \* Bản rõ **x** = **1299**.

Chọn **p** = 2579, **g** = **2**, **a** = 765. Tính khóa công khai **h** = **2** 765 mod 2579 = **949**.

\* Lập mã: Chọn ngẫu nhiên **r** = **853**. Bản mã là **y** = (**435, 2369**), trong đó

**y1** = 2853 mod 2579 = **435** và **y2** = 1299 \* 949 **853** mod 2579 = **2396**

\* Giải mã: **x** = **y2**(**y1 a**) **-1** mod **p** = 2369 \* (435 765) -1 mod 2579 = **1299**.

***Độ an toàn***

1). Hệ mã hóa Elgamal là không tất định, tức là với một bản rõ **x** và 1 khóa bí mật **a**, thì có thể có nhiều hơn một bản mã **y**, vì trong công thức lập mã còn có thành phần ngẫu nhiên **r**.

2). Độ an toàn của Hệ mật Elgamal dựa vào khả năng giải bài toán logarit rời rạc trong **Zp**. Theo giả thiết trong sơ đồ, thì bài toán này phải là “khó” giải.

Cụ thể như sau: Theo công thức lập mã: **y** = **ek** (**x**, **r**) = (**y1, y2**), trong đó **y1** = **g** **r** mod **p** và **y2** = **x** \* **h****r**  mod **p**

Như vậy muốn xác định bản rõ **x** từ công thức **y2** , thám mã phải biết được **r**.

Giá trị này có thể tính được từ công thức **y1** , nhưng lại gặp bài toán logarit rời rạc.

***Chương 3*. CHỮ KÝ SỐ**

**3.1. TỔNG QUAN VỀ CHỮ KÝ SỐ**

**3.1.1. Khái niệm** “**Chữ ký số**”

***3.1.1.1. Giới thiệu***

Để chứng thực nguồn gốc hay hiệu lực của một tài liệu (ví dụ: đơn xin học, giấy báo nhập học, ... ), lâu nay người ta dùng chữ ký “***tay***”, ghi vào phía dưới của mỗi tài liệu. Như vậy người ký phải ***trực tiếp*** “***ký tay***“ vào tài liệu.

Ngày nay các tài liệu được số hóa, người ta cũng có nhu cầu chứng thực nguồn gốc hay hiệu lực của các tài liệu này. Rõ ràng không thể “ký tay“ vào tài liệu, vì chúng không được in ấn trên giấy. Tài liệu “số” ( hay tài liệu “điện tử”) là một xâu các bit (0 hay 1), xâu bít có thể rất dài (nếu in trên giấy có thể hàng nghìn trang). “Chữ ký” để chứng thực một xâu bít tài liệu cũng không thể là một xâu bit nhỏ đặt phía dưới xâu bit tài liệu. Một “chữ ký” như vậy chắc chắn sẽ bị kẻ gian sao chép để đặt dưới một tài liệu khác bất hợp pháp.

Những năm 80 của thế kỷ 20, các nhà khoa học đã phát minh ra “***chữ ký số***” để chứng thực một “***tài liệu số***”. Đó chính là “**bản mã**” của xâu bít tài liệu.

Người ta tạo ra “***chữ ký số***” (chữ ký điện tử) trên “***tài liệu số***” giống như tạo ra “**bản mã**” của tài liệu với “khóa lập mã”.

Như vậy “***ký số***” trên “***tài liệu số***” là “***ký***” trên từng bit tài liệu. Kẻ gian khó thể giả mạo “chữ ký số” nếu nó không biết “khóa lập mã”.

Để kiểm tra một “***chữ ký số***” thuộc về một “***tài liệu số***”, người ta giải mã

“***chữ ký số***” bằng “khóa giải mã”, và so sánh với tài liệu gốc.

Ngoài ý nghĩa để chứng thực nguồn gốc hay hiệu lực của các tài liệu số hóa,

“***chữ ký số***” còn dùng để kiểm tra tính toàn vẹn của tài liệu gốc.

Mặt mạnh của “***chữ ký số***” hơn “chữ ký tay” còn là ở chỗ người ta có thể “***ký***” vào tài liệu từ rất xa (trên mạng công khai). Hơn thế nữa, có thể “***ký***” bằng các thiết bị cầm tay (VD điện thoại di động) tại khắp mọi nơi (Ubikytous) và di động (Mobile), miễn là kết nối được vào mạng. Đỡ tốn bao thời gian, sức lực, chi phí.

“Ký số” thực hiện trên từng bit tài liệu, nên độ dài của “***chữ ký số***” ít nhất cũng bằng độ dài của tài liệu. Do đó thay vì ký trên tài liệu dài, người ta thường dùng “***hàm băm***” để tạo “***đại diện***” cho tài liệu, sau đó mới “Ký số” lên “***đại diện***” này.

***3.1.1.2. Sơ đồ chữ ký số***

Sơ đồ chữ ký là bộ năm (**P**, **A**, **K**, **S**, **V** ), trong đó:

**P**  là tập hữu hạn các văn bản có thể.

**A**  là tập hữu hạn các chữ ký có thể.

**K**  là tập hữu hạn các khoá có thể.

**S** là tập các thuật toán ký.

**V**  là tập các thuật toán kiểm thử.

Với mỗi khóa **k** ∈ **K**, có thuật toán ký **Sig k** ∈ **S**, **Sig** **k**: **P**→ **A**,

có thuật toán kiểm tra chữ ký **Ver k** ∈ **V**, **Ver k**: **P** × **A**→ {đúng, sai},

thoả mãn điều kiện sau với mọi **x** ∈ **P**, **y** ∈ **A**:

Đúng, nếu **y** = **Sig k**(**x**)

**Ver k** (**x**, **y**) =

Sai, nếu **y** ≠ **Sig k**(**x**)

***Chú ý***

Người ta thường dùng hệ mã hóa khóa công khai để lập “***Sơ đồ chữ ký số”***.

Ở đây khóa bí mật **a** dùng làm khóa “***ký***”, khóa công khai **b** dùng làm khóa kiểm tra “***chữ ký***”.

Ngược lại với việc mã hóa, dùng khóa công khai **b** để lập mã., dùng khóa bí mật **a** để giải mã.

Điều này là hoàn toàn tự nhiên, vì “***ký***” cần giữ bí mật nên phải dùng khóa bí mật **a** để “***ký***”. Còn “***chữ ký***” là công khai cho mọi người biết, nên họ dùng khóa công khai **b** để kiểm tra.

**3.1.2. Phân loại** “**Chữ ký số**”.

Có nhiều loại chữ ký tùy theo cách phân loại, sau đây xin giới thiệu một số cách. ***Cách 1: Phân loại chữ ký theo khả năng khôi phục thông điệp gốc*.**

1). **Chữ ký có thể khôi phục thông điệp gốc**:

Là loại chữ ký, trong đó người nhận có thể khôi phục lại được thông điệp gốc, đã được “***ký***” bởi “***chữ ký***” này.

Ví dụ: Chữ ký RSA là chữ ký khôi phục thông điệp, sẽ trình bày trong mục sau.

2). **Chữ ký không thể khôi phục thông điệp gốc**:

Là loại chữ ký, trong đó người nhận không thể khôi phục lại được thông điệp gốc, đã được “***ký***” bởi “***chữ ký***” này.

Ví dụ: Chữ ký Elgamal là chữ ký không thể khôi phục, sẽ trình bày trong mục sau.

***Cách 2: Phân loại chữ ký theo*  mức an toàn.**

1). **Chữ ký** “**không thể phủ nhận**”:

Để tránh việc chối bỏ chữ ký hay nhân bản chữ ký để sử dụng nhiều lần, người gửi chữ ký cũng tham gia trực tiếp vào việc kiểm thử chữ ký. Điều đó được thực hiện bằng một giao thức kiểm thử, dưới dạng một giao thức mời hỏi và trả lời.

Ví dụ: Chữ ký không phủ định (Chaum - van Antverpen), trình bày trong mục sau.

2). **Chữ ký** “**một lần**”:

Để bảo đảm an toàn, “Khóa ký” chỉ dùng 1 lần (one- time) trên 1 tài liệu.

Ví dụ: Chữ ký một lần Lamport. Chữ ký Fail - Stop (Van Heyst & Pedersen).

***Cách 3: Phân loại chữ ký theo*  ứng dụng đặc trưng.**

Chữ ký “mù” (Blind Signature).

Chữ ký “nhóm” (Group Signature).

Chữ ký “bội” (Multy Signature).

Chữ ký “mù nhóm” (Blind Group Signature).

Chữ ký “mù bội” (Blind Multy Signature).

**3. 2. CHỮ KÝ RSA**

**3.2.1. Sơ đồ chữ ký**

***Sơ đồ***  (Đề xuất năm 1978)

***1/. Tạo cặp khóa*** (***bí mật, công khai***) (**a**, **b**):

Chọn bí mật số nguyên tố lớn **p**, **q**, tính **n** = **p** \* **q**, công khai **n**, đặt **P** = A = **Zn**

Tính bí mật **φ**(**n**) = (**p-1**).(**q-1**).

Chọn khóa công khai **b** < **φ**(**n**),nguyên tố cung nhau với **φ(n)**.

Khóa bí mật **a** là phần tử nghịch đảo của **b** theo mod **φ**(n): **a**\***b** ≡ **1** (mod φ(n).

Tập cặp khóa (bí mật, công khai) **K** = {(**a**, **b**)/ a, b ∈ Zn , **a**\***b** ≡ 1 (mod φ(n))}.

2/. ***Ký số***: Chữ ký trên **x** ∈ **P** là **y** = **Sig k** (**x**) = **x a** (mod **n**), **y** ∈ **A**.(R1)

3/. ***Kiểm tra chữ ký***: **Verk**(**x**, **y**) = đúng ⇔ **x** ≡ **y b** (mod **n**). (R2)

***Chú ý***

- So sánh giữa sơ đồ chữ ký RSA và sơ đồ mã hóa RSA ta thấy có sự tương ứng.

- Việc ký chẳng qua là mã hoá, việc kiểm thử lại chính là việc giải mã:

Việc “ký số” vào **x** tương ứng với việc “mã hoá” tài liệu **x**.

Kiểm thử chữ ký chính là việc giải mã “chữ ký”, để kiểm tra xem tài liệu đã giải mã có đúng là tài liệu trước khi ký không. Thuật toán và khóa kiểm thử “chữ ký” là công khai, ai cũng có thể kiểm thử chữ ký được.

***Ví dụ***  Chữ ký trên **x** = **2**

\****Tạo cặp khóa*** (***bí mật, công khai***) (**a**, **b**):

Chọn bí mật số nguyên tố **p**=3, **q**=5, tính **n** = **p** \* **q** = 3\*5 = 15, công khai **n**.

Đặt **P** = **A** = **Zn** = **Zn** . Tính bí mật **φ**(**n**) = (**p-1**).(**q-1**) = 2 \* 4 = 8.

Chọn khóa công khai **b** = **3** < **φ**(**n**),nguyên tố với **φ(n) = 8**.

Khóa bí mật **a** = **3**, là phần tử nghịch đảo của **b** theo mod **φ**(n): **a**\***b** ≡ **1** (mod φ(n).

\* ***Ký số***: Chữ ký trên **x** = **2** ∈ **P** là

**y** = **Sig k** (**x**) = **x a** (mod **n**)= **2 3** (mod **15**) = **8**, **y** ∈ **A**.

\* ***Kiểm tra chữ ký***: **Verk**(**x**, **y**) = đúng ⇔ **x** ≡ **y b** (mod **n**)

⇔ **2** ≡ **8 3** (mod **15**).

**3.2.2. Độ an toàn của chữ ký RSA**

**\* Bài toán căn bản bảo đảm độ an toàn của Sơ đồ chữ ký RSA:**

Bài toán tách số nguyên n thành tích của 2 số nguyên tố: n = p\*q

Vì nếu giải được bài toán này thì có thể tính được khóa mật **a** từ khóa công khai b và phần tử công khai n.

1). Người gửi G gửi tài liệu **x** cùng chữ ký **y** đến người nhận N, có 2 cách xử lý:

***a). Ký trước, Mã hóa sau:***

G ký trước vào **x** bằng chữ ký **y** = **SigG**(**x**), sau đó mã hoá **x** và **y** nhận được

**z** = **eG** (**x**, **y**). G gửi **z**  cho N.

Nhận được **z**, N giải mã **z** để được **x**, **y**.

Tiếp theo kiểm tra chữ ký **VerN**(**x**, **y**) = true ?

***b). Mã hóa trước, Ký sau:***

G mã hoá trước **x** bằng **u** = **eG** (**x**), sau đó ký vào **u** bằng chữ ký **v** = **SigG**(**u**).

G gửi (**u**, **v**) cho N.

Nhận được (**u**, **v**), N giải mã **u** được **x**.

Tiếp theo kiểm tra chữ ký **VerN**(**u**, **v**) = true ?

2). Giả sử H lấy trộm được thông tin trên đường truyền từ G đến N.

+ Trong trường hợp **a**, H lấy được **z**. Trong trường hợp **b**, H lấy được (**u**, **v**).

+ Để tấn công **x**, trong cả hai trường hợp, H đều phải giải mã thông tin lấy được.

+ Để tấn công vào chữ ký, thay bằng chữ ký (giả mạo), thì xảy ra điều gì ?

- Trường hợp **a**, để tấn công chữ ký **y**, H phải giải mã **z**, mới nhận được **y**.

- Trường hợp **b**, để tấn công chữ ký **v**, H đã sẵn có **v**, H chỉ việc thay **v** bằng **v’**.

H thay chữ ký **v** trên **u**, bằng chữ ký của H là **v’** = SigH(**u**), gửi (**u**, **v’**) đến N.

Khi nhận được **v’**, N kiểm thử thấy sai, gửi phản hồi lại G.

G có thể chứng minh chữ ký đó là giả mạo.

G gửi chữ ký đúng **v** cho N, nhưng quá trình truyền tin sẽ bị chậm lại.

+ Như vậy trong trường hợp **b**, H có thể giả mạo chữ ký mà không cần giải mã.

Vì thế có lời khuyên: ***Hãy ký trước, sau đó mã hoá cả chữ ký.***

**3. 3. CHỮ KÝ ELGAMAL**

**3.3.1. Sơ đồ chữ ký Elgamal**

***Sơ đồ***  (Elgamal đề xuất năm 1985)

1/. ***Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **h*)*** :

Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong **Zp** là “khó” giải.

Chọn phần tử nguyên thuỷ **g** ∈ **Zp**\* . Đặt **P** = **Z p**\*, **A** = **Z p**\* x **Z p-1**.

Chọn khóa bí mật là **a** ∈ **Zp**\* . Tính khóa công khai **h** ≡ **g a** mod **p**.

Định nghĩa tập khóa:  = {(**p**, **g**, **a**, **h**): **h** ≡ **g a** mod **p**}.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

2/. ***Ký số***: Dùng 2 khóa ký: khóa **a** và khóa ngẫu nhiên bí mật **r** ∈ **Zp-1\*** .

(Vì **r** ∈ **Zp-1\*** , nên nguyên tố cùng **p** -1, do đó tồn tại **r -1** mod (**p** -1) ).

Chữ ký trên **x** ∈ **P** là  **y** = **Sig a**(**x**, **r**) = (**γ**, **δ**), **y** ∈ ***A*** (E1)

Trong đó **γ** ∈ **Z p**\*, **δ** ∈ **Zp-1**:

**γ** = **g r** mod **p** và **δ** = (**x** – **a** \* **γ** ) \* **r -1** mod (**p** -1)

3/. ***Kiểm tra chữ ký***:

**Ver k** (**x**, **γ**, **δ**) = đúng ⇔ **h γ** \* **γ δ**≡ **g x** mod **p**. (E2)

***Chú ý***: Nếu chữ ký được tính đúng, kiểm thử sẽ thành công vì

**h γ** \* **γ δ ≡ g a****γ**\* **g r** \* **δ**mod p **≡ g** (**a****γ** +  **r** \* **δ**) mod p ≡ **g x** mod **p**.

Do **δ** = (**x** – **a** \* **γ** ) \* **r -1** mod (p -1) nên (**a** \* **γ +**  **r** \***δ**) ≡ **x** mod (p-1).

***Ví dụ*** Chữ ký Elgamal trên dữ liệu **x** = **112**.

***1/. Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **h*)*** :

Chọn số nguyên tố **p** = **463**. Đặt **P** = **Z p**\*, **A** = **Z p**\* x **Z p-1**.

Chọn phần tử nguyên thuỷ **g** = **2** ∈ **Zp**\* .

Chọn khóa bí mật là **a** = 211 ∈ **Zp**\* .

Tính khóa công khai **h** ≡ **g a** mod **p** = **2** 211 mod 463 = **249**.

Định nghĩa tập khóa:  = {(**p**, **g**, **a**, **h**): **h** ≡ **g a** mod **p**}.

Các giá trị **p**, **g**, **h** được công khai, phải giữ bí mật **a**.

2/. ***Ký số***: Chọn ngẫu nhiên bí mật **r** = 235 ∈ **Zp-1\*** . Khóa ký là (**a**, **r** ).

Vì **r** ∈ **Zp-1\*** , nên nguyên tố cùng **p** -1, do đó tồn tại **r -1** mod (**p** -1). Cụ thể:

UCLN(**r**, **p**-1) = UCLN(235, 462) = 1, nên **r -1** mod (p-1) = 235 -1 mod 462 = 289.

Chữ ký trên dữ liệu **x** = **112** là ( **γ**, **δ** ) = (**16**, **108**), trong đó:

**γ** = **g r** mod **p** = **2** 235 mod 463 = **16**

**δ** = (**x** – **a** \* **γ** ) \* **r -1** mod (**p** -1) = (**112** - 211 \* 16) \* 289 mod 462 = **108**

3/. ***Kiểm tra chữ ký***: **Ver k** (**x**, **γ**, **δ**) = đúng ⇔ **h γ** \* **γ δ**≡ **g x** mod **p**.

**h γ** \* **γ δ**  = 249 16 \* 16 108 mod 463 = **132**

**g x** mod **p** = **2 112** mod 463 = **132**.

Hai giá trị đó bằng nhau, như vậy chữ ký là đúng.

**3.3.2. Độ an toàn của chữ ký Elgamal**

**\* Bài toán căn bản bảo đảm độ an toàn của Sơ đồ chữ ký Elgamal:**

Bài toán tính Logarit rời rạc:

Biết khóa công khai **h** ≡ **g a** mod **p**.

Nên có thể xác định khóa bí mật **a**  bằng cách tính Log **g** **h**.

***3.3.2.1. Vấn đề giả mạo chữ ký Elgamal***

***1). Trường hợp 1*: Giả mạo chữ ký không cùng với tài liệu được ký**.

+ H cố gắng giả mạo chữ ký trên **x**, mà không biết khóa bí mật **a**.

Như vậy, H phải tính được **γ** và **δ**.

\* Nếu chọn trước **γ**, H phải tính **δ** qua đẳng thức **h γ** \* **γ δ**≡ **g x** mod **p**  (E2)

Tức là **γ δ**≡ **g x** **h - γ** mod p hay **δ** ≡ log**γ**  **g x** **h - γ** mod p.

\* Nếu chọn trước **δ**, H phải tính **γ** qua phương trình: **h γ** \* **γ δ**≡ **g x** mod **p**.

Hiện nay chưa có cách hữu hiệu 2 trường hợp trên, nhưng phỏng đoán là khó hơn bài toán logarit rời rạc.

Có thể có cách tính **γ**, **δ** đồng thời với (**γ**, **δ**) là chữ ký ? Chưa có trả lời rõ !

\* Nếu chọn trước **γ**, **δ**, sau đó tính **x,**  H phải đối đầu với bài toán logarit rời rạc.

Ta có **h γ** \* **γ δ**≡ **g x** mod **p** (E2).

Như vậy  **x** ≡ log**g** **g x** ≡ log**g h γ** \* **γ δ**

***2). Trường hợp 2*: Giả mạo chữ ký cùng với tài liệu được ký**.

H có thể ký trên tài liệu ngẫu nhiên bằng cách chọn trước đồng thời **x**, **γ**, **δ**.

***Cách 1***

\* Chọn **x**, **γ**, **δ** thoả mãn điều kiện kiểm thử như sau:

Chọn các số nguyên i, j sao cho 0 ≤ i, j ≤ p-2, (j, p-1) = 1 và tính:

**γ** = g i h j mod p, **δ** = **- γ**  j -1 mod (p -1), **x** = **- γ** i j -1 mod (p -1).

Trong đó j -1 được tính theo mod (p -1) (nghĩa là j nguyên tố với p-1).

\* Chứng minh (**γ**, **δ**) là chữ ký trên **x,** bằng cách kiểm tra điều kiện kiểm thử:

**h γ γ δ** ≡ **h γ (g i** **h j** ) **-γ . j -1**mod p ≡ **h** **γ****g – i .** **γ .****j-1** **h -γ**mod p ≡ **g x** mod p

***Ví dụ***

\* Chọn các tham số của sơ đồ chữ ký Elgamal:

Số nguyên tố p = 463, phần tử sinh **g** = 2, Khóa bí mật a = 135.

Khóa công khai  **h** = **g** **a** mod p = 2 135 mod 463 = 272.

\* Chọn **x**, **γ**, **δ** thoả mãn điều kiện kiểm thử như sau:

Chọn i = 89, j = 125, 0 ≤ i, j ≤ p-2, (j, p-1) = 1. Tính j -1 mod (p-1) = 377.

**γ** = **g** i \* **h** j mod p = 289  \* 272125 mod 463 = 218

**δ** = - **γ** \* j -1 mod (p -1) = -218 \* 377 mod 462 = 50

**x** = -**γ**  \* i \* j -1 mod (p -1) = -218 \* 89 \* 377 mod 462 = **292**

\* (**γ**, **δ**) = (218, 50) là chữ ký trên **x = 292,** vì thỏa mãn điều kiện kiểm thử:

**h** **γ**\* **γ δ**= 272 218 \* 218 50 ≡ 322 (mod 463)

**g** x = 2 292 ≡ 322 (mod 467)

***Cách 2***

\* Nếu (**γ**, **δ**) là chữ ký trên tài liệu **x** có từ trước, thì có thể giả mạo chữ ký trên

tài liệu **x’** khác.

+ Chọn số nguyên k, i, j thỏa mãn 0 ≤ k, i, j ≤ p-2, (k **γ** - j **δ**, p-1) = 1 và tính:

**λ** = **γ** **k**g **i** h **j** mod p, **μ** = **δ λ** (**k γ - j δ**) **-1** mod (p -1),

**x’** = **λ** (**k x** + **i δ**) (**k γ - j δ**) **-1** mod (p -1)

\* (**λ**, **μ**) là chữ ký trên **x’,** vì thỏa mãn điều kiện kiểm thử:

**h λ λ μ** ≡ **g x’** mod p.

***Chú ý***

Cả hai cách giả mạo nói trên đều cho chữ ký đúng trên tài liệu tương ứng, nhưng đó không phải là tài liệu được chọn theo ý của người giả mạo. Tài liệu đó đều được tính sau khi tính chữ ký, vì vậy giả mạo loại này trong thực tế cũng không có ý nghĩa nhiều.

***3.3.2.2. Vấn đề Phá khóa theo sơ đồ Elgamal***

Khoá bí mật **a** có thể bị phát hiện, nếu khóa ngẫu nhiên **r** bị lộ, hoặc dùng **r** cho hai lần ký khác nhau.

***1). Trường hợp 1: Số ngẫu nhiên r bị lộ:***

Nếu **r** bị lộ, thám mã sẽ tính được khoá mật **a** = (**x** - **r** **δ**) γ -1 mod (p-1).

***2). Trường hợp 2: Dùng r cho hai lần ký khác nhau:***

Giả sử dùng **r** cho 2 lần ký trên **x1** và **x2**.

(**γ, δ1**) là chữ ký trên **x1**, (**γ, δ2**) là chữ ký trên **x2**.

Khi đó thám mã có thể tính được **a** như sau:

. 

Do đó ta có 

Đặt γ = α r , ta có 

tương đương với x1-x2 ≡ r (δ1 - δ2) mod (p-1) (1)

Đặt d = (δ1 - δ2, p -1). Khi đó d | (p-1), d | (δ1 - δ2) ⇒ d | (x1-x2).



Khi đó đồng dư thức (1) trở thành: x' ≡ **r** \* δ' (mod p')

Vì (δ', p') = 1 nên tính ε = (δ')-1 mod p' và tính  **r** = x'\*ε mod p'

⇒ **r** = x'\*ε + i\*p' mod (p-1), với i là giá trị nào đó, 0≤ i ≤ d-1.

Thử với giá trị đó, ta tìm được **r** (điều kiện thử để xác định **r** là γ = αr mod p).

Tiếp theo sẽ tính được **a** như trường hợp 1).

**3. 4. CHỮ KÝ DSS**

**3.4.1. Sơ đồ chữ ký DSS**

***3.4.1.1. Giới thiệu Chuẩn chữ ký số DSS***

Chuẩn chữ ký số (DSS: Digital Signature Standard) được đề xuất năm 1991, là cải biên của sơ đồ chữ ký ElGamal, và được chấp nhận là chuẩn vào năm 1994 để dùng trong một số lĩnh vực giao dịch ở USA.

Thông thường tài liệu số được mã hoá và giải mã 1 lần. Nhưng chữ ký lại liên quan đến ***pháp luật***, ***chữ ký*,** có thể phải kiểm thử sau nhiều năm đã ký. Do đó ***chữ ký***  phải được bảo vệ cẩn thận.

Số nguyên tố **p** phải đủ lớn (chẳng hạn dài cỡ 512 bit) để bảo đảm an toàn, nhiều người đề nghị nó phải dài 1024 bit. Tuy nhiên, độ dài chữ ký theo sơ đồ Elgamal là gấp đôi số bit của p, do đó nếu p dài 512 bit thì độ dài chữ ký là 1024 bit.

Trong ứng dụng dùng thẻ thông minh (Smart card) lại mong muốn có chữ ký ngắn, nên giải pháp sửa đổi là một mặt dùng p với độ dài từ 512 bit đến 1024 bit (bội của 64), mặt khác trong chữ ký (**γ**, **δ**), các số **γ**, **δ** có độ dài biểu diễn ngắn, ví dụ 160 bit. Khi đó chữ ký là 320 bit.

Điều này được thực hiện bằng cách dùng nhóm con cyclic **Zq**\* của **Zp**\* thay cho **Zp**\*, do đó mọi tính toán được thực hiện trong **Zp**\*, nhưng thành phần chữ ký lại thuộc **Zq**\*.

+ Trong sơ đồ ký Elgamal, công thức tính **δ** được sửa đổi thành

**δ** = (x + **a** \* **γ**) **r -1** mod q.

+ Điều kiện kiểm thử **h** **γ  γδ** ≡**g** **x** mod p được sửa đổi thành

.

Chú ý nếu UCLN(x + **g** \* **γ**, p-1) = 1 thì **δ -1** mod p tồn tại.

***3.4.1.2. Sơ đồ Chuẩn chữ ký số DSS***

***Sơ đồ***

***1/.Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **h*)*** :

+ Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong **Zp** là “khó” giải.

Chọn **q** là ước nguyên tố của p-1. Tức là p-1 = **t** \* **q** hay p = **t** \* q + 1.

(Số nguyên tố p cỡ 512 bit, q cỡ 160 bit).

+ Chọn **g** ∈ **Zp**\* là căn bậc **q** của **1** mod **p**, (**g** là phần tử sinh của **Zp**\* ).

Tính **α** = **g t** , chọn khóa bí mật **a** ∈ **Zp**\*, tính khóa công khai **h** ≡ **α a** mod **p**.

+ Đặt ***P*** = **Z q**\*, **A** = **Z q**\* x **Z q**\* , ***K*** = {(**p**, **q**, **α**, **a**, **h**)/ **a** ∈ **Zp**\*, **h** ≡ **α** **a**mod **p**}.

+ Với mỗi khóa (**p**, **q**, **α**, **a**, **h**), k’ = **a** bí mật, k” = (**p**, **q**, **α**, **h**) công khai.

***2/. Ký số***: Dùng 2 khóa ký: khóa **a** và khóa ngẫu nhiên bí mật **r** ∈ **Z q**\*.

Chữ ký trên **x** ∈ **Zp**\*  là **Sig k’**(**x**, **r**) = (**γ**, **δ**), trong đó

**γ** = (**α** **r** mod p) mod **q**, **δ** = ((**x** + **a** \* **γ** ) \* **r -1** mod **q**.

(Chú ý **r** ∈ **Z q**\*, để bảo đảm tồn tại **r -1** mod **q**).

***3/. Kiểm tra chữ ký***: Với **e1** = **x** \* **δ -1** mod **q**, **e2** = **γ** \* **δ -1** mod **q**.

**Ver k”** (**x**, **γ**, **δ**) = đúng ⇔ (**α** **e1** \*  **h** **e2** mod **p**) mod **q** = **γ**

***Ví dụ***

***1/. Tạo cặp khóa*** (***bí mật, công khai***)  ***(*a**, **h*)*** :

Chọn **p** = 7649, **q** = 239 là ước nguyên tố của p-1, **t** = 32.

Tức là p -1 = **t** \* **q** hay **p** = **t** \* q + 1 = 32\*q + 1 = 32\*239 + 1 = 7649.

Chọn **g** =3 ∈ **Z**7649  là phần tử sinh. **α** = **g t**  mod p = 3 32 mod 7649 = 7098.

Chọn khóa mật **a** = 85, khóa công khai **h** = **α a** mod p = 709885 mod 7649 = 5387.

***2/. Ký số***: Dùng 2 khóa ký: **a** và khóa ngẫu nhiên **r** = 58 ∈ **Z q**\*, **r -1** mod q = 136.

+ Chữ ký trên **x** = 1246 là **Sig k’**(**x**, **r**) = (**γ**, **δ**) = (115, 87) , trong đó

**γ** = (**α r** mod p) mod q = (709858 mod 7649)mod 239 = 593 mod 239 = 115.

**δ** = (**x** + **a** \* **γ** ) \* **r** -1 mod q = (1246 + 85 \* 115) \*136 mod 239 = 87.

***3/. Kiểm tra chữ ký***: (**γ**, **δ**) = (115, 87) là chữ ký đúng trên **x** = 1246.

e1 = x\* **δ -1** mod q = 1246 \* 11 mod q = 83, e2 = **γ** \* **δ -1** mod q = 115\*11 mod q = 70.

Điều kiện kiểm thử đúng ? (**α** **e1** \*  **h** **e2** mod **p**) mod **q** = **γ ,** với **δ** -1 = 11.

(709883 \*538770 mod 7649) mod 239 = 593 mod 239 = 115 = **γ**.

***Chú ý***

1). Liên quan tới các tính toán cụ thể trong sơ đồ:

+ Chú ý rằng phải có **δ** ≠ 0 (mod q) để bảo đảm có **δ -1**mod q trong điều kiện kiểm thử (tương đương UCLN(δ, p-1) = 1). Vì vậy nếu chọn **r** mà không được điều kiện trên, thì phải chọn **r** khác để có **δ** ≠ 0 (mod q).

Tuy nhiên khả năng **δ** ≡ 0 mod q là 2-160 , điều đó hầu như không xảy ra.

+ Một chú ý là là thay vì tính p trước rồi mới tính q, ta sẽ tính q trước rồi tìm p.

2). Liên quan chung tới DSS (1991):

+ Độ dài cố định của p là 512 bit. Nhiều người muốn p có thể thay đổi lớn hơn.

Vì thế NIST sửa đổi là p có độ dài thay đổi, là bội của 64: từ 512 đến 1024 bit.

+ Nếu dùng chữ ký RSA với thành phần kiểm thử chữ ký là nhỏ, thì việc kiểm thử nhanh hơn việc ký. Đối với DSS, ngược lại, việc ký nhanh hơn kiểm thử.

Điều này dẫn đến vấn đề:

Một tài liệu chỉ được ký một lần, nhưng nó lại được kiểm thử nhiều lần, nên người ta muốn thuật toán kiểm thử nhanh hơn.

Máy tính ký và kiểm thử như thế nào ? Nhiều ứng dụng dùng thẻ thông minh với khả năng có hạn, kết nối với 1 máy tính mạnh hơn, vì vậy nên xây dựng sơ đồ chữ ký ít liên quan đến thẻ.

Nhưng tình huống đặt ra là một thẻ thông minh có thể sinh ra chữ ký và cũng có thể kiểm thử chữ ký, do vậy rất khó kết luận ?

NIST trả lời rằng thời gian kiểm thử và sinh chữ ký, cái nào nhanh hơn không quan trọng, miễn là đủ nhanh.

**3. 5. CHỮ KÝ KHÔNG THỂ PHỦ ĐỊNH**

**3.5.1. Sơ đồ chữ ký**

***3.5.1.1. Giới thiệu chữ ký không thể phủ định***

Trong phần trước ta đã trình bày một số sơ đồ chữ ký điện tử. Trong các sơ đồ đó, việc kiểm thử tính đúng đắn của chữ ký là do người nhận thực hiện. Nhằm tránh việc nhân bản chữ ký để sử dụng nhiều lần, tốt nhất là để người gửi tham gia trực tiếp vào việc kiểm thử chữ ký. Điều đó được thực hiện bằng một giao thức kiểm thử, dưới dạng một giao thức mời hỏi và trả lời.

Giả sử tài liệu cùng chữ ký từ G gửi đến N. Khi N yêu cầu G cùng kiểm thử chữ ký, thì một vấn đề nảy sinh là làm sao để ngăn cản G chối bỏ một chữ ký mà anh ta đã ký, G có thể tuyên bố rằng chữ ký đó là giả mạo ?

Để giải quyết tình huống trên, cần có thêm giao thức chối bỏ, bằng giao thức này, G có thể chứng minh một chữ ký là giả mạo. Nếu G từ chối tham gia vào giao thức đó, thì có thể xem rằng G không chứng minh được chữ ký đó là giả mạo.

Như vậy sơ đồ chữ ký không phủ định được gồm 3 phần: một thuật toán ký, một giao thức kiểm thử, và một giao thức chối bỏ.

***3.5.1.2. Sơ đồ chữ ký không thể phủ định* (Chaum - van Antverpen)**

***\* Chuẩn bị các tham số***:

Chọn số nguyên tố **p** sao cho bài toán log rời rạc trong **Zp** là khó.

**p** = 2\*q+1, q cũng là số nguyên tố.

Gọi **P** là nhóm nhân con của **Zp**\* theo q (**P** gồm các thặng dư bậc hai theo mod p).

Chọn phần tử sinh **g** của nhóm **P** cấp q.

Đặt **P** = **A** = **P**, **K** = {(**p**, **g**, **a**, **h**): **a** ∈ **Z q\***, **h** ≡ **g a**mod **p** }

***1). Thuật toán ký:***  Dùng khoá bí mật k’ = **a** để ký lên **x**:

Chữ ký là **y** = **Sig k’** (**x**) = **x a** mod **p**.

***2). Giao thức kiểm thử:*** Dùng khoá công khai k” = (**p**, **g**, **h**).

Với **x**, **y** ∈ **P**, người nhận N cùng người gửi G thực hiện giao thức kiểm thử:

1/. N chọn ngẫu nhiên **e1**, **e2** ∈ **Zq\***

2/. N tính **c** = **y e1  h e2** mod p, và gửi cho G.

3/. G tính  và gửi cho N.

4/. N chấp nhận **y** là chữ ký đúng, nếu **d** ≡ **x e1  g e2** mod p

***3). Giao thức chối bỏ:***

1/. N chọn ngẫu nhiên **e1, e2** ∈ **Z­q\***

2/. N tính **c** = **y e1  h e2** mod p, và gửi cho G.

3/. G tính  và gửi cho N.

4/. N thử điều kiện **d** ≠ **x e1** **g e2** (mod p).

5/. N chọn ngẫu nhiên **f1, f2**  ∈ **Zq\***.

6/. N tính  và gửi cho G.

7/. G tính  và gửi cho N.

8/. N thử điều kiện **D** ≠ **x f1** **g f2** (mod p).

9/. N kết luận **y** là chữ ký ***giả mạo*** nếu:

 (thay ***α*** bằng ***g***).

***Ví dụ*** Ký trên **x** = 229

***\* Chuẩn bị các tham số***:

Chọn số nguyên tố **p** = 467 = 2 \* q + 1, q = 233 cũng là số nguyên tố.

Chọn phần tử sinh của nhóm **P** là **g** = 4, (**P** là nhóm nhân con cấp q của **Zp**\*).

Đặt **P** = **A** = **P**, **K** = {(**p**, **g**, **a**, **h**): **a** ∈ **Z q\***, **h** ≡ **g a**mod **p** }

Chọn khóa mật **a** = 121. Khóa công khai **h** ≡ **g a**mod **p** = 4121 mod 467 = 422.

***1). Thuật toán ký:***  Dùng khoá bí mật k’ = **a** để ký lên **x** = 229:

Chữ ký là **y** = **Sig k’** (**x**) = **x a** mod **p** = 229121 mod 467 = **9**.

***2). Giao thức kiểm thử:*** Dùng khoá công khai k” = (**p**, **g**, **h**) = (467, 4, 422).

1/. N chọn ngẫu nhiên **e1** = 48, **e2** = 213 ∈ **Zq\***

2/. N tính **c** = **y e1  h e2** mod p = 116 và gửi cho G.

3/. G tính  = 235 và gửi cho N.

4/. N chấp nhận **y** là chữ ký đúng, nếu **d** ≡ **x e1  g e2** mod p

N thử điều kiện **d** ≡ **x e1  g e2** mod p.

Rõ ràng 235 ≡ 229 48 \* 4 213 (mod 467).

N chấp nhận **y** = **9** đúng là chữ ký của G trên **x** = 229.

***3). Giao thức chối bỏ:***

Giả sử G gửi tài liệu **x** = 226 với chữ ký **y** = 183. Giao thức chối bỏ thực hiện:

1/. N chọn ngẫu nhiên **e1** = 47**, e2** = 137 ∈ **Z­q\***

2/. N tính **c** = **y e1  h e2** mod p = 306, và gửi cho G.

3/. G tính  = 184, và gửi cho N.

4/. N thử điều kiện **d** ≠ **x e1** **g e2** (mod p).

Điều kiện trên không đúng vì 184 ≠ 226 47 \* 4 137 ≡ 145 mod 467.

N lại tiếp tục thực hiện bước 5 của giao thức.

5/. N chọn ngẫu nhiên **f1** = 225**, f2** = 19 ∈ **Zq\***.

6/. N tính  = 348, và gửi cho G.

7/. G tính  = 426, và gửi cho N.

8/. N thử điều kiện **D** ≠ **x f1** **g f2** (mod p).

D = 426 trong khi **x f1** **g f2** (mod p) = 226 225 \* 4 19 ≡ 295 mod 467.

Điều kiện 8 là đúng, nên N thực hiện bước 9:

9/. N kết luận **y** là chữ ký ***giả mạo*** nếu:

 (thay ***α*** bằng ***g***).

N tính giá trị của 2 vế đồng dư ≡

(d\* α -e2)f1 ≡ (184 \* 4 -137)225 ≡ 79 mod 467

(D\* α -f2)e1 ≡ (426 \* 4 -19)47 ≡ 79 mod 467

Hai giá trị đó bằng nhau. Kết luận chữ ký **y** là ***giả mạo***.

**3. 6. ĐẠI DIỆN TÀI LIỆU VÀ HÀM BĂM**

**3.6.1. Vấn đề Đại diện tài liệu và Hàm băm**

***3.6.1.1. Một số vấn đề với*** “***chữ ký số***”

***Vấn đề 1***:

“Ký số” thực hiện trên từng bit tài liệu, nên độ dài của “***chữ ký số***” ít nhất cũng bằng độ dài của tài liệu. Một số chữ ký trên bản tin có kích thước gấp đôi bản tin gốc. Ví dụ khi dùng sơ đồ chữ ký DSS để ký vào bản tin có kích thước 160 bit, thì chữ ký số này sẽ có kích thước 320 bit.

Trong khi đó trên thực tế, ta cần phải ký vào các bản tin có kích thước rất lớn, ví dụ vài chục MegaByte (tương ứng hàng ngàn trang tin trên giấy). Như vậy phải tốn nhiều bộ nhớ để lưu trữ “chữ ký”, mặt khác tốn nhiều thời gian để truyền “chữ ký” trên mạng..

***Vấn đề 2***:

Với một số sơ đồ chữ ký “an toàn”, thì tốc độ ký lại chậm vì chúng dùng nhiều phép tính số học phức tạp như số mũ modulo.

***Vấn đề 3***:

Thực tế có thể xảy ra trường hợp: Với nhiều bản tin đầu vào khác nhau, sử dụng hệ mã hóa hay sơ đồ ký số giống nhau (có thể khác nhau), nhưng lại cho ra bản mã hay chữ ký giống nhau (đó là ánh xạ nhiều – một), như hìnhdưới. Điều này sẽ dẫn đến phức tạp cho việc xác thực thông tin.

|  |
| --- |
|  |

***3.6.1.2. Giải quyết các vấn đề trên như thế nào*** ?

***Cách 1***:

Một cách đơn giản để giải quyết các vấn đề trên với thông điệp có kích thước lớn là “***chặt***” bản tin thành nhiều đoạn nhỏ (VD 160 bit), sau đó ký lên các đoạn đó độc lập nhau. Nhưng biện pháp này gặp các vấn đề trên.

Hơn thế nữa còn gặp vấn đề nghiêm trọng hơn. Đó là kết quả sau khi ký, nội dung của thông điệp có thể bị xáo trộn các đoạn với nhau, hoặc một số đoạn trong chúng có thể bị mất mát. Ta cần phải bảo vệ tính toàn vẹn của bản tin gốc.

***Cách 2***:

Thay vì phải ký trên tài liệu dài, người ta thường dùng “***hàm băm***” để tạo “***đại diện***” cho tài liệu, sau đó mới “Ký số” lên “***đại diện***” này.

Các tài liệu (bản tin) có thể dưới dạng văn bản, hình ảnh, âm thanh, …và kích thước của chúng tùy ý (vài KB đến vài chục MB), qua các thuật toán băm: như MD4, MD5, SHA, các “***đại diện***” tương ứng của chúng có kích thước ***cố định***, ví dụ 128 bit với dòng MD, 160 bit với dòng SHA.

“***Đại diện***” của tài liệu chính là giá trị của “***hàm băm***” trên tài liệu, nó còn được gọi là “tóm lược” hay “bản thu gọn” của tài liệu.

Với mỗi tài liệu (đầu vào), qua “***hàm băm***” chỉ có thể tính ra được một “***đại diện***”- giá trị băm tương ứng - duy nhất. “***Đại diện***” của tài liệu được xem là “***đặc thù***” của tài liệu (thông điệp), giống như dấu vân tay của mỗi người.

Trên thực tế, hai tài liệu khác nhau có hai “đại diện” khác nhau. Như vậy khi đã có “***đại diện***” duy nhất cho một tài liệu, thì việc “ký” vào tài liệu, được thay bằng “ký” vào “***đại diện***” của nó là hoàn toàn hợp lý. Đó là chưa kể việc tiết kiệm bao nhiêu thời gian cho việc “ký số”, bộ nhớ lưu giữ “chữ ký”, thời gian truyền “chữ ký” trên mạng, …

Cơ chế gửi tài liệu cùng “chữ ký” trên nó sử dụng hàm băm được mô tả theo các hình sau.

|  |
| --- |
|  |

*Băm thông điệp.*

|  |
| --- |
|  |

*Ký trên bản băm thông điệp.*

|  |
| --- |
|  |

*Truyền thông điệp và chữ ký.*

**3.6.2. Tổng quan về Hàm băm**

### 3.6.2.1. Hàm băm (Hàm tạo đại diện tài liệu)

**1). Khái niệm Hàm băm**

***Hàm băm*** là thuật toán không dùng khóa để ***mã hóa*** (ở đây dùng thuật ngữ “băm” thay cho “mã hóa”), nó có nhiệm vụ “lọc” (băm) tài liệu (bản tin) và cho kết quả là một giá trị “băm” có kích thước cố định, còn gọi là “***đại diện tài liệu***” hay “đại diện bản tin”, “đại diện thông điệp”.

Hàm băm là ***hàm một chiều***, theo nghĩa giá trị của hàm băm là ***duy nhất***, và từ giá trị băm này, “**khó thể**” **suy ngược** lại được nội dung hay độ dài ban đầu của tài liệu gốc.

**2). Đặc tính của Hàm băm**

Hàm băm **h** là hàm một chiều (One-way Hash) với các đặc tính sau:

1). Với tài liệu đầu vào (bản tin gốc) **x**, chỉ thu được giá trị băm duy nhất **z** = **h**(**x**).

2). Nếu dữ liệu trong bản tin **x** bị thay đổi hay bị xóa để thành bản tin **x**’, thì giá trị băm **h**(**x**’) ≠ **h**(**x**).

Cho dù chỉ là một sự thay đổi nhỏ, ví dụ chỉ thay đổi 1 bit dữ liệu của bản tin gốc **x**,thì giá trị băm **h**(**x**) của nó cũng vẫn thay đổi. Điều này có nghĩa là: hai thông điệp khác nhau, thì giá trị băm của chúng cũng khác nhau.

3). Nội dung của bản tin gốc “khó” thể suy ra từ giá trị hàm băm của nó. Nghĩa là: với thông điệp **x** thì “dễ” tính được **z** = **h**(**x**), nhưng lại “khó” tính ngược lại được **x** nếu chỉ biết giá trị băm **h**(**x**) (Kể cả khi biết hàm băm **h**)..

**3). Ứng dụng của hàm băm**

1). Với bản tin dài x, thì chữ ký trên x cũng sẽ dài, như vậy tốn thời gian “ký”,

tốn bộ nhớ lưu giữ “chữ ký”, tốn thời gian truyền “chữ ký” trên mạng.

Người ta dùng hàm băm **h** để tạo đại diện bản tin **z** = **h**(**x**), nó có độ dài ngắn (VD 128 bit). Sau đó ký trên **z**, như vậy chữ ký trên **z** sẽ nhỏ hơn rất nhiều

so với chữ ký trên bản tin gốc x.

2). Hàm băm dùng để xác định tính toàn vẹn dữ liệu.

3). Hàm băm dùng để bảo mật một số dữ liệu đặc biệt, ví dụ bảo vệ mật khẩu, bảo vệ khóa mật mã, …

### 3.6.2.2. Các tính chất của Hàm băm

***Tính chất 1:*** Hàm băm ***h*** là ***không va chạm*** **yếu**.

***Ví dụ***: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 1.*

\* *Hình a*: *Cách đi đúng của thông tin:*  thông tin được truyền đúng từ A đến B:

|  |
| --- |
|  |

\* Hình b: Thông tin bị lấy trộm và bị thay đổi trên đường truyền:

|  |
| --- |
|  |

***\* Kiểu tấn công theo tính chất 1:***

+ Người A gửi cho B bản tin (**x**, **y**) với **y** = sigk (**h**(**x**)). B không nhận được (**x**, **y**) vì:

+ Trên đường truyền, tin bị lấy trộm. Tên trộm, bằng cách nào đó tìm được bản tin **x**’ ≠ **x**, nhưng **h**(**x**’) = **h**(**x**). Hắn thay thế **x** bằng **x**’, và chuyển tiếp (**x**’, **y**) cho B.

+ Người B nhận được (**x**’, **y**), và vẫn xác thực được thông tin đúng. Do đó, để tránh kiểu tấn công như trên, hàm **h** phải thỏa mãn tính chất: **không va chạm yếu**.

***\* Khái niệm:*** Hàm băm ***không va chạm*** **yếu**.

Hàm băm **h** được gọi là **không va chạm yếu**, nếu cho trước bức điện **x**, “khó” thể tính toán để tìm ra bức điện **x**’ ≠ **x** mà **h**(**x**’) = **h**(**x**).

***Tính chất 2***: Hàm băm **h** là ***không va chạm* mạnh**.

***Ví dụ***: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 2.*

+ Đầu tiên, tên giả mạo tìm được hai thông điệp khác nhau **x**’ và **x** (**x**’**x**) mà có **h**(**x**’) = **h**(**x**). (Ta coi bức thông điệp **x** là hợp lệ, còn **x**’ là giả mạo).

+ Tiếp theo, hắn thuyết phục ông A ký vào bản tóm lược **h**(**x**) để nhận được **y**. Khi đó (**x**’, **y**) là bức điện giả mạo, nhưng hợp lệ vì **h**(**x**’) = **h**(**x**).

Để tránh kiểu tấn công này, hàm **h** phải thỏa mãn tính chất: ***không va chạm*** **mạnh**.

***\* Khái niệm:*** Hàm băm ***không va chạm*** **mạnh**.

Hàm băm **h** được gọi là ***không va chạm*** **mạnh**, nếu “khó” thể tính toán

để tìm ra hai bức thông điệp khác nhau  **x**’và **x** (**x**’**x**)màcó **h**(**x**’) = **h**(**x**).

***Tính chất 3***: Hàm băm **h** là ***hàm một chiều***.

***Ví dụ***: Xét kiểu tấn công như sau: *Kiểu tấn công theo tính chất 3.*

+ Người A gửi cho B thông tin (**x**, **z**, **y**) với **z** = **h**(**x**), **y** = sigk (**z**).

+ Giả sử tên giả mạo tìm được bản tin **x’**, được tính ngược từ bản tóm lược

**z** = **h**(**x**).

+ Tên trộm thay thế bản tin **x** hợp lệ, bằng bản tin **x’** giả mạo, nhưng lại có

**z** = **h**(**x’**). Hắn ta ký số trên bản tóm lược **z** của **x’** bằng đúng chữ ký hợp lệ.

Nếu làm được như vậy, thì (**x’**, **z**, **y**) là bức điện giả mạo, nhưng hợp lệ.

Để tránh được kiểu tấn công này, hàm băm **h** cần thỏa mãn ***tính chất một chiều***.

***\* Khái niệm:*** Hàm băm **một chiều**.

Hàm băm **h**  được gọi là ***hàm một chiều*** nếu khi cho trước một bản tóm lược thông báo **z**thì “khó thể” tính toán để tìm ra thông điệp ban đầu **x** sao cho**h**(**x**)= **z**.

### 3.6.2.3. Các Hàm băm

Các hàm băm dòng MD (MD2, MD4, MD5) do Rivest đề xuất. Giá trị băm theo các thuật toán này có độ dài cố định là **128** bit. Hàm băm MD4 đưa ra vào năm 1990. Một năm sau phiên bản mạnh hơn là MD5 cũng được đề xuất.

Hàm băm an toàn SHA, phức tạp hơn nhiều, cũng dựa trên các phương pháp tương tự, được công bố trong Hồ sơ Liên bang năm 1992 và được chấp nhận làm tiêu chuẩn vào năm 1993 do Viện Tiêu Chuẩn và Công Nghệ Quốc Gia (NIST). Giá trị băm theo thuật toán này có độ dài cố định là **160** bit.

**3.6.2. Hàm băm MD4**

***3.6.2.1.* *Khái niệm “Thông điệp đệm”***

“***Thông điệp đệm***” (Messege Padding) là xâu bit có độ dài chia hết cho **512**.

“Thông điệp đệm” được lưu trong mảng **M** = M[0] M[1] … M[N-1].

Trong đó **M**[i] là xâu bit có độ dài 32 bit, gọi là *word*.

N ≡ 0 mod 16. (32 x 16 = 512).

**M** được xây dựng từ ***Bản tin gốc*** **a** bằng thuật toán:

|  |
| --- |
| 1. **d** = 447 – ( |**a**| mod 512). (= 512 nếu |**a**| mod 512 > 447). 2. Giả sử **l**là kí hiệu biểu diễn nhị phân của |**a**| mod 264, tl: | **l**| = **64**. 3. **M** = **a** || 1 || 0 **d** || **l** |

\* Độ dài của xâu **a** || 1 || 0**d** qui ước là |**a**| + 1 + **d** = 448 mod 512.

\* Độ dài của “Thông điệp đệm” **M**  là

448 mod 512 +| **l**| = 448 mod 512 + **64** = 512 mod 512.

Chú ý: Vì **M** = **a** || 1 || 0 **d** || **l** nên

**d** = |**M**| - ( |**a**| + 1 + **l** ) =

512 - ( |**a**| + 1 + **64** ) = 512 - (|**a**| +65) = 447 – ( |**a**| mod 512).

***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

Trong việc xây dựng M, ta gắn số **1** đơn lẻ vào sau **a**, sau đó thêm tiếp các số 0 vào đủ để độ dài của M đồng dư với 448 modulo 512. Cuối cùng nối thêm 64 bit (chính là | ***l*** |) chứa biểu diễn nhị phân về độ dài ban đầu của ***x*** (được rút gọn theo modulo 264 nếu cần).

Xâu kết quả M có độ dài chia hết cho 512. Vì thế khi chặt M thành các *word* 32 bit, số *word* nhận được là N sẽ chia hết cho 16.

Mục đích việc tạo ra mảng M (”thông điệp đệm”) là để các hàm băm

xử lý trên từng khối (block) 512 bit, tức là 16 *word*, cùng một lúc.

***3.6.2.2.* *Thuật toán băm MD5***

*INPUT*  : Thông điệp là một xâu ***a***  có độ dài ***b*** bit.

*OUTPUT* : Bản băm, đại diện cho thông điệp gốc, độ dài cố định **128** bit.

**a). Tóm tắt thuật toán**

***Bước 1****:* **Khởi tạo các thanh ghi**

Có 4 thanh ghi để tính toán nhằm đưa ra các đoạn mã: A, B, C, D. Bản tóm lược của thông điệp được xây dựng như sự kết nối của các thanh ghi. Mỗi thanh ghi có độ dài 32 bit. Các thanh ghi này được khởi tạo giá trị hecxa.

*word* A := **67 45 23 01** *word* B := **ef cd ab 89**

*word* C := **98 ba dc fe**  *word* D := **10 32 54 76**

***Bước 2****:*

Xử lý thông điệp ***a*** trong 16 khối *word*, có nghĩa là xử lý cùng một lúc 16 *word* = 16 \* 32 bit = 512 bit.

Chia mảng M thành các khối 512 bit, đưa từng khối 512 bit vào mảng T[j]. Mỗi lần xử lý một khối 512 bit. Lặp lại N/16 lần.

**b). Thuật toán MD4**

1/.A := **67 45 23 01** B := **ef cd ab 89**

C := **98 ba dc fe**  D := **10 32 54 76**

2/. **FOR** i := 0 **TO** N/16 - 1 **DO**

**for** j := 0 **to** 15  **do T**[j] = M[16 i + j];

AA := A; BB := B;

CC := C; DD := D;

Mỗi lần xử lý 16 từ, mỗi từ 32 bit, tl: 512 bit.

**3/. Vòng 1**

**Vòng 2**

**Vòng 3**

4/. A = A + AA; B = B + BB; C = C + CC; D = D + DD;

Gán giá trị cho 4 biến AA, BB, CC, DD bằng giá trị 4 thanh ghi A, B, C, D tương ứng.

**c). Các phép tính và các hàm dùng trong Thuật toán MD4**

***\* Các phép toán logic được sử dụng trong ba vòng.***

X  Y là phép toán AND theo bit giữa X và Y

X  Y là phép toán OR theo bit giữa X và Y

X  Y là phép toán XOR theo bit giữa X và Y

 X chỉ phép bù của X

X + Y là phép cộng theo modulo 232

X <<< ***s*** là phép dịch vòng trái X đi ***s*** vị trí (0  ***s***  31)

***\* Ba hàm F, G, H dùng tương ứng trong vòng 1, 2, 3.***

Mỗi hàm này là một hàm boolean tính theo bit.

F(X, Y, Z) = (XY)((X) Z)

G(X, Y, Z) = (XY)(XZ)(YZ)

H(X, Y, Z) = XYZ

Ba vòng trong MD4 là hoàn toàn khác nhau. Mỗi vòng (3.1, 3.2, 3.3) gồm một trong 16 *word* trong T được xử lý. Các phép toán được thực hiện trong ba vòng tạo ra các giá trị mới trong bốn *thanh ghi*. Cuối cùng, bốn *thanh ghi* được cập nhật ở 3.4 bằng cách cộng ngược các giá trị lưu trước đó ở 2.3. Phép cộng này được xác định là cộng các số nguyên dương, được rút gọn theo modulo 232.

**d). Ba vòng “băm”.**

*Vòng 1*

|  |
| --- |
| 1. A = (A + F(B, C, D) + T[0]) <<< 3 |
| 2. D = (D + F(A, B, C) + T[1]) <<< 7 |
| 3.  C = (C + F(D, A, B) + T[2]) <<< 11 |
| 4.  B = (B + F(C, D, A) + T[3]) <<< 19 |
| 5. A = (A + F(B, C, D) + T[4]) <<< 3 |
| 6. D = (D + F(A, B, C) + T[5]) <<< 7 |
| 7. C = (C + F(D, A, B) + T[6]) <<< 11 |
| 8. B = (B + F(C, D, A) + T[7]) <<< 19 |
| 9.  A = (A + F(B, C, D) + T[8]) <<< 3 |
| 10. D = (D + F(A, B, C) + T[9]) <<< 7 |
| 11. C = (C + F(D, A, B) + T[10]) <<< 11 |
| 12. B = (B + F(C, D, A) + T[11]) <<< 19 |
| 13. A = (A + F(B, C, D) + T[12]) <<< 3 |
| 14. D = (D + F(A, B, C) + T[13]) <<< 7 |
| 15. C = (C + F(D, A, B) + T[14]) <<< 11 |
| 16. B = (B + F(C, D, A) + T[15]) <<< 19 |

*Kết quả của VD* **a** *sau khi được xử lý qua vòng 1.*

|  |  |  |  |
| --- | --- | --- | --- |
| 1. 64B3DA82 | 5. 3D5E5934 | 9. 59798D5E | 13. 7551AAC6 |
| 2. 34D8EB03 | 6. 489D5140 | 10. D206302D | 14. 789B984F |
| 3. B7BCB118 | 7. CCD14D6C | 11. 753D6134 | 15. F55A1F31 |
| 4. 6D91B115 | 8. 454D0E92 | 12. F52AED08 | 16. ABA71E22 |

***Vòng 2***

|  |
| --- |
| 1. A = (A + G(B, C, D) + T[0] + 5A827999) <<< 3 |
| 2.  D = (D + G(A, B, C) + T[4] + 5A827999) <<< 5 |
| 3.  C = (C + G(D, A, B) + T[8] + 5A827999) <<< 9 |
| 4.  B = (B + G(C, D, A) + T[12] + 5A827999) <<< 13 |
| 5. A = (A + G(B, C, D) + T[1] + 5A827999) <<< 3 |
| 6.  D = (D + G(A, B, C) + T[5] + 5A827999) <<< 5 |
| 7.  C = (C + G(D, A, B) + T[9] + 5A827999) <<< 9 |
| 8.  B = (B + G(C, D, A) + T[13] + 5A827999) <<< 13 |
| 9.  A = (A + G(B, C, D) + T[2] + 5A827999) <<< 3 |
| 10. D = (D + G(A, B, C) + T[6] + 5A827999) <<< 5 |
| 11. C = (C + G(D, A, B) + T[10] + 5A827999) <<< 9 |
| 12. B = (B + G(C, D, A) + T[14] + 5A827999) <<< 13 |
| 13. A = (A + G(B, C, D) + T[3] + 5A827999) <<< 3 |
| 14. D = (D + G(A, B, C) + T[7] + 5A827999) <<< 5 |
| 15. C = (C + G(D, A, B) + T[11] + 5A827999) <<< 9 |
| 16. B = (B + G(C, D, A) + T[15] + 5A827999) <<< 13 |

Giá trị 5A827999 là một hằng số ở dạng hecxa có độ dài 32 bit

*Kết* *quả của VD* **a** *sau khi được xử lý qua vòng 2.*

|  |  |  |  |
| --- | --- | --- | --- |
| 1. 558C2E28 | 5. 558C2E28 | 9. 31E9FE4A | 13. B60A11E6 |
| 2. 5A0E08F9 | 6. 5A0E08F9 | 10. 6F68E462 | 14. 2DED6D8E |
| 3. F6A9B390 | 7. F6A9B390 | 11. D745F88A | 15. A2870B31 |
| 4. 7876BC8F | 8. 7876BC8F | 12. 7050BC10 | 16. 4384D178 |

*Vòng 3*

|  |
| --- |
| 1.  A = (A + H(B, C, D) + T[0] + 6ED9EBA1) <<< 3 |
| 2.  D = (D + H(A, B, C) + T[8] + 6ED9EBA1) <<< 9 |
| 3.  C = (C + H(D, A, B) + T[4] + 6ED9EBA1) <<< 11 |
| 4.  B = (B + H(C, D, A) + T[12] + 6ED9EBA1) <<< 15 |
| 5.  A = (A + H(B, C, D) + T[2] + 6ED9EBA1) <<< 3 |
| 6.  D = (D + H(A, B, C) + T[10] + 6ED9EBA1) <<< 9 |
| 7. C = (C + H(D, A, B) + T[6] + 6ED9EBA1) <<< 11 |
| 8. B = (B + H(C, D, A) + T[14] + 6ED9EBA1) <<< 15 |
| 9.  A = (A + H(B, C, D) + T[1] + 6ED9EBA1) <<< 3 |
| 10. D = (D + H(A, B, C) + T[9] + 6ED9EBA1) <<< 9 |
| 11. C = (C + H(D, A, B) + T[5] + 6ED9EBA1) <<< 11 |
| 12. B = (B + H(C, D, A) + T[13] + 6ED9EBA1) <<< 15 |
| 13. A = (A + H(B, C, D) + T[3] + 6ED9EBA1) <<< 3 |
| 14. D = (D + H(A, B, C) + T[11] + 6ED9EBA1) <<< 9 |
| 15. C = (C + H(D, A, B) + T[7] + 6ED9EBA1) <<< 11 |
| 16. B = (B + H(C, D, A) + T[15] + 6ED9EBA1) <<< 15 |

Giá trị 6ED9EBA1 là một hằng số ở dạng hecxa có độ dài 32 bit

*Kết* *quả của VD* **a** *sau khi được xử lý qua vòng 3.*

|  |  |  |  |
| --- | --- | --- | --- |
| 1. 98A7C489 | 5. F3031C80 | 9. C02E826B | 13. 03477E5E |
| 2. E70B031C | 6. 7D7A371B | 10. F38DC78B | 14. 77509F0A |
| 3. A96B2FFA | 7. 1C2487DE | 11. E3C7F63B | 15. FB3D792D |
| 4. 58BE9F94 | 8. F7767709 | 12. 812AB00F | 16. 23D73C06 |

**e). Kết quả “băm”**

Kết quả ra là đoạn mã có độ dài 128 bit, được thu gọn từ thông điệp **a** có độ dài ***b*** bit. Đoạn mã này thu được từ 4 thanh ghi A, B, C, D: bắt đầu từ byte thấp của thanh ghi A cho đến byte cao của thanh ghi D.

Với VD **a** = “ABC”, kết quả cuối cùng là Đại diện văn bản:

A = 6A8CA15F

B = 671E4A93

C = 93F85626

D = 3409907C

Chú ý: A = A + AA = 03477E5E

67452301

= 6A8CA15F

***Chương 4.*** **BẢO TOÀN VÀ XÁC THỰC DỮ LIỆU**

**4.1. BẢO TOÀN DỮ LIỆU**

**4.1.1. Tổng quan về Bảo toàn dữ liệu**

***4.1.1.1. Bài toán Bảo toàn dữ liệu hay Bảo đảm “toàn vẹn” dữ liệu***

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

N nhận được tài liệu **y**  ghi rõ địa chỉ của G.

***Bài toán 1***: Bài toán “***bảo toàn dữ liệu*** ” (bảo đảm toàn vẹn dữ liệu).

Kiểm tra để khẳng định được rằng tài liệu **y**  (ghi địa chỉ của G), chính là **x**.

Yêu cầu quan trọng nhất của bài toán 1 là khẳng định được **y ≡** **x** ?

Chưa quan tâm tới **y** là của G hay không ! Tức là chưa quan tâm tới nguồn gốc của **y**.

Câu hỏi: Hiện nay có các phương pháp nào để kiểm tra **y ≡** **x** ?

Điều đó có nghĩa là trên đường truyền tin, nếu có kẻ gian chặn bắt được **x**,

nếu nó thay đổi nội dung của **x**, hay thay **x** bằng **y**, thì

1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x**.

2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**.

***Bài toán 2***: Bài toán “***xác thực*** ***nguồn gốc dữ liệu***” (chứng thực nguồn gốc dữ liệu).

Kiểm tra để khẳng định được rằng tài liệu **y** (ghi địa chỉ của G), đúng là của G.

Chưa quan tâm tới **y ≡** **x** hay không ! Các phương pháp kiểm tra ?

Câu hỏi: Hiện nay có các phương pháp nào để kiểm tra nguồn gốc của **y** ?

***4.1.1.2. Phương pháp Bảo toàn dữ liệu***

**1). Dùng mã hóa hay giấu tin đề bảo toàn dữ liệu.**

Phương pháp mã hóa hay giấu tin dùng để “che giấu” tài liệu **x**, kẻ gian có chặn bắt được nó, thì cũng “khó” hiểu được nó, do đó “khó thể” thay đổi được nội dung của **x**. Như vậy phương pháp này thực hiện được khả năng 2/.

(2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**).

Chú ý rằng nếu tài liệu **x** được mã hóa, kẻ gian “khó” giải được mã, nếu tài liệu **x** được “giấu” trong tài liệu khác, kẻ gian “khó” tách ra được tài liệu gốc.

Nhưng nếu kẻ gian vẫn thay **x** bằng **y**, thì phương pháp mã hóa hay giấu tin không xác minh được ! Tức là phương pháp này không có khả năng 1/.

(1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x** ).

**2). Dùng “chữ ký số” đề bảo toàn dữ liệu.**

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng “chữ ký số” để bảo toàn **x**, thì G phải chuyển **x** và cả chữ ký trên **x** là **z** cho N. Như vậy N sẽ nhận được cặp tin (***tài liệu***, ***chữ ký***) = (**x**, **z**), **z** = Sig (**x**).

Như vậy phương pháp này chỉ thực hiện được khả năng 1/.

(1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x** ).

Nếu kẻ gian thay đổi nội dung của **x**, hay dùng **y** thay cho **x**, thì khi N

kiểm tra chữ ký của G, chắc chắn chữ ký **z** là sai, vì **x** cũ đã bị thay đổi.

Chú ý rằng phương pháp này không thực hiện được khả năng 2/.

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**).

**3). Dùng “thủy vân ký” đề bảo toàn dữ liệu.**

**Khái niệm *Giấu thông tin*** (Steganography): là giấu thông tin này vào trong một thông tin khác. Thông tin được giấu (nhúng) vào bên trong một thông tin khác, sẽ khó bị phát hiện, vì người ta khó nhận biết được là đã có một thông tin được ***giấu*** (nhúng) vào bên trong một thông tin khác (gọi là ***môi trường giấu tin***).

Nói cách khác, giấu tin giống như “***ngụy trang***” cho thông tin, không gây ra cho tin tặc sự nghi ngờ. Ví dụ một thông tin ***giấu*** vào bên trong một bức tranh, thì **sự vô hình** của thông tin chứa trong bức tranh sẽ “***đánh lừa***” được sự chú ý của tin tặc.

Theo nghĩa rộng, ***giấu tin*** cũng là ***hệ mật mã***, nhằm đảm bảo tính bí mật của thông tin.

Theo nghĩa rộng, “***Giấu tin***” nhằm thực hiện hai việc: ***Bảo vệ thông tin cần giấu*** và  ***Bảo vệ chính môi trường giấu tin.***

***Giấu*** (nhúng) thông tin mật vào một thông tin khác, sao cho người ta khó phát hiện ra thông tin mật đó. Đó là bảo vệ thông tin cần giấu. Loại giấu tin này được gọi là “***Steganography***”.

***Giấu*** (nhúng) thông tin vào một thông tin khác, nhằm bảo vệ chính đối tượng được dùng để giấu tin vào. Tức là giấu tin để bảo vệ chính môi trường giấu tin.

***Tin được giấu*** có vai trò như ***chữ ký*** hay ***con dấu*** dùng để xác thực (chứng nhận) thông tin (là môi trường giấu tin). Loại “giấu tin” này được gọi là thủy vân ký “***Watermarking***”.

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng “thủy vân ký” để bảo toàn **x**, thì G phải cho “ẩn giấu” vào **x** một “dấu hiệu đặc trưng” **C** của mình, như vậy **x** đã trở thành **x’** (vật mang tin **C**). Sau đó

G chuyển **x’** cho N trên mạng.

N tách **C** ra khỏi **x’** và nhận được tài liệu gốc là **x**.

Như vậy phương pháp này chỉ thực hiện được khả năng 1/.

(1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x** ).

Lý do:

Kẻ gian không biết được tài liệu gốc là **x**, vì nó chỉ chặn bắt được **x’**.

Nó tìm cách tách một dấu hiệu khả nghi **C’**  từ **x’**.

Khi đó tài liệu mà N nhận được không phải là **x’**, mà là **x’’**.

N tách **C** ra khỏi **x’’** , sẽ không nhận được “dấu hiệu” **C**.

Tức là N có thể phát hiện được sự thay đổi nội dung của **x**.

Chú ý rằng phương pháp này không thực hiện được khả năng 2/ (như PP Ký).

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**).

**4). Dùng “hàm băm” đề bảo toàn dữ liệu.**

\* ***Đặc điểm của hàm băm***:

Hàm băm **h** là hàm một chiều (One-way Hash) với các đặc tính sau:

1). Với tài liệu đầu vào (bản tin gốc) **x**, chỉ thu được giá trị băm duy nhất **z** = **h**(**x**).

2). Nếu dữ liệu trong bản tin **x** bị thay đổi hay bị thay hoàn toàn để thành bản tin **x**’,

thì giá trị băm **h**(**x**’) ≠ **h**(**x**).

Cho dù chỉ là một sự thay đổi nhỏ, ví dụ chỉ thay đổi 1 bit dữ liệu của bản tin gốc **x**,thì giá trị băm **h**(**x**) của nó cũng vẫn thay đổi. Điều này có nghĩa là: hai thông điệp khác nhau, thì giá trị băm của chúng cũng khác nhau.

Dựa vào đặc điểm trên của hàm băm, người ta bảo toàn dữ liệu như sau.

**\* *Bảo toàn dữ liệu dùng “hàm băm”***:

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng “hàm băm” để bảo toàn **x**, thì G phải chuyển **x** và cả giá trị băm trên **x** là **z** cho N. Như vậy N sẽ nhận được cặp tin (***tài liệu***, ***đại diện TL***) = (**x**, **z**), **z** = **h**(**x**).

(Chú ý **z** là giá trị băm trên **x**, còn được gọi là ***đại diện tài liệu***).

N sẽ băm lại **x**, và nhận được giá trị băm là **z’**. Nếu **z’** ≠ **z**, thì chắc chắn **x** đã bị thay đổi trên đường truyền tin. Nếu **z’** ≡ **z**, thì **x** được bảo toàn.

Như vậy phương pháp này chỉ thực hiện được khả năng 1/.

Tức là N có thể phát hiện được sự thay đổi nội dung của **x**.

(1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x** ).

Chú ý rằng phương pháp này không thực hiện được khả năng 2/(như PP Ký).

Tức là kẻ gian có thể thay được đổi nội dung của **x**.

(2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**).

**4.1.2. Bảo toàn dữ liệu bằng kết hợp các phương pháp**

***4.1.2.1. Kết hợp Mã hóa hay Giấu tin với ký số hay thủy vân ký.***

Trong mục trước ta đã biết rằng mỗi công cụ có mặt mạnh và mặt yếu.

Kết hợp các công cụ lại, sẽ nhận được các mặt mạnh của chúng, cụ thể như sau:

+ Phương pháp mã hóa hay giấu tin chỉ thực hiện được khả năng 2/ của bảo toàn

dữ liệu **x**, tức là không cho phép kẻ gian thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 1/, tức là người nhận N không phát hiện được sự thay đổi nội dung của **x**.

+ Ngược lại, phương pháp ký số hay thủy vân ký chỉ thực hiện được khả năng 1/ của bảo toàn dữ liệu **x**, tức là có thể phát hiện được sự thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 2/, tức là kẻ gian vẫn có thể thay đổi được

nội dung của **x**.

Kết hợp hai phương pháp trên, sẽ có được phương pháp thực hiện được cả khả năng 1/ và khả năng 2/ của việc bảo toàn dữ liệu **x**.

(1/. N có thể ***phát hiện được*** sự thay đổi nội dung của **x** ).

(2/. G có thể ***không cho phép***  kẻ gian thay đổi nội dung của **x**).

***\* Kết hợp Mã hóa và Ký số:***

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng “chữ ký số” để bảo toàn **x**, thì G phải chuyển **x** và cả chữ ký trên **x** là **z** cho N. Trước khi gửi cặp tin **T**= (***tài liệu***, ***chữ ký***) = (**x**, **z**), (**z** = Sig (**x**)), G mã hóa

**T**, sau đó mới gửi cho N.

Tương tự ta kết hợp Giấu tin với ký số, Mã hóa và với thủy vân ký, …

***4.1.2.2. Kết hợp Mã hóa hay Giấu tin với Hàm băm.***

+ Phương pháp mã hóa hay giấu tin chỉ thực hiện được khả năng 2/ của bảo toàn

dữ liệu **x**, tức là không cho phép kẻ gian thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 1/, tức là người nhận N không phát hiện được sự thay đổi nội dung của **x**.

+ Ngược lại, phương pháp hàm băm chỉ thực hiện được khả năng 1/ của bảo toàn

dữ liệu **x**, tức là có thể phát hiện được sự thay đổi nội dung của **x** !

Nhưng không thực hiện được khả năng 2/, tức là kẻ gian vẫn có thể thay đổi được

nội dung của **x**.

Kết hợp hai phương pháp trên, sẽ có được phương pháp thực hiện được cả khả năng 1/ và khả năng 2/ của việc bảo toàn dữ liệu **x**.

***\* Kết hợp Mã hóa và hàm băm:***

Người gửi G cần chuyển tài liệu **x** tới người nhận N trên mạng công khai.

Nếu dùng “hàm băm” để bảo toàn **x**, thì G phải chuyển **x** và cả giá trị băm của **x** là **z** cho N. Trước khi gửi cặp tin **T**= (***tài liệu***, ***đại diện TL***) = (**x**, **z**), **z** = **h**(**x**), G mã hóa

**T**, sau đó mới gửi cho N.

Tương tự ta kết hợp Giấu tin với hàm băm.

**4.2. BẢO ĐẢM XÁC THỰC**

4.2.1. Phân loại xác thực điện tử

Hiện nay có một số cách phân loại “xác thực” như sau:

*Cách 1*: Phân loại theo đối tượng cần xác thực.

Có 2 loại đối tượng chính: Dữ liệu và Thực thể.

1). Xác thực Dữ liệu: Văn bản, hình ảnh, âm thanh,..

+ Xác thực Thông điệp (Message Authentication).

+ Xác thực Giao dịch (Transaction Authentication).

+ Xác thực Khóa (Key Authentication).

2). Xác thực Thực thể: Người dùng, thiết bị đầu cuối,..

+ Xác thực Thực thể (Entity Authentication).

*Cách 2*: Phân loại theo công việc cần xác thực.

1). Xác thực Thông điệp (Message Authentication).

2). Xác thực Giao dịch (Transaction Authentication).

3). Xác thực Thực thể (Entity Authentication).

4). Xác thực Khóa (Key Authentication).

*Cách 3*: Phân loại theo đặc điểm xác thực.

1). Xác thực bảo đảm định danh nguồn gốc (Identification of Source).

2). Xác thực bảo đảm toàn vẹn dữ liệu (Data Integrity).

3). Xác thực bảo đảm tính duy nhất (Uniqueness).

4). Xác thực bảo đảm tính phù hợp về thời gian (Timeliness).

Tổng kết các loại xác thực

|  |  |  |  |
| --- | --- | --- | --- |
| ***Type of Authentication*** | ***Identification of Source*** | ***Data Integrity*** | ***Timeliness or Uniqueness*** |
| **Message authentication**  **Transaction authentication**  **Entity authentication**  **Key authentication** | yes  yes  yes  yes | yes  yes  -  yes | -  yes  yes  - |

Properties of Various Types of Authentication

4.2.2. Xác thực dữ liệu (Data Authentication)

***4.2.2.1. Xác thực thông điệp (Message Authentication)***

***Xác thực thông điệp*** hay ***Xác thực*** ***tính nguyên bản*** của dữ liệu

(Data Origin Authentication)là một kiểu ***xác thực đảm bảo*** **một thực thể** được

chứng thực là **nguồn gốc thực sự** tạo ra dữ liệu này ở một thời điểm nào đó.

Xác thực thông điệp bao hàm cả **tính toàn vẹn dữ liệu**, nhưng ***không đảm bảo***

tính duy nhất và phù hợp về thời gian của nó.

***4.2.2.2. Xác thực giao dịch (Transaction Authentication)***

Xác thực giao dịch là ***Xác thực thông điệp*** cộng thêm việc **đảm bảo** **tính**

**duy nhất** (Uniqueness) và phù hợp về **thời gian** (Timeliness) của nó.

Xác thực giao dịch liên quan đến việc sử dụng các tham số thời gian (**TVP** – Time Variant Parameters).

**Transaction Authentication = Message Authentication + TVP**

Xác thực giao dịch “mạnh hơn” Xác thực thông điệp.

***Chú ý***

Một thông điệp gửi đi có thể đã bị chặn và phát lại (tương tự như việc sử dụng lại nhiều lần một đồng tiền “số” (Double spending). Để ngăn chặn tình huống này, người gửi và người nhận có thể gắn vào thông điệp ***nhãn thời gian*** hay ***mã thông điệp***.

***Mã thông điệp*** là con số được gắn vào thông điệp. Nó có thế chỉ dùng một lần duy nhất, giá trị không lặp lại, hoặc dùng một dãy số (Sequence Numbers).

Thám mã không thể biết được các bit của con số này nằm ở vị trí nào trong thông điệp, hay không thể biết cách thay đổi các bit để tạo ra dạng mã hoá của số tiếp theo, hoặc không thể biết cách thay đổi các bit này mà không làm gián đoạn việc giải mã phần còn lại của thông báo.

Các số thông báo này không thể bị thay thế, thay đổi hoặc giả mạo. Người nhận phải duy trì việc đếm các số thông báo đã nhận được. Nếu 2 người sử dụng một tập các số thì người nhận có thể ngay lập tức biết được liệu có thông báo nào trước thông báo hiện thời đã bị mất hoặc bị chậm trễ, vì số được mã hoá của thông báo hiện thời phải lớn hơn số được mã hoá của thông báo trước.

Nếu người gửi có nhiều thông báo thì có thể số thông báo sẽ quá dài. Vì thế, người ta thường đặt lại bộ đếm số thông báo khi nó đạt tới giá trị lớn nào đó. Lúc này tất cả các bên thu phải được thông báo rằng, số thông báo được gửi tiếp theo sẽ được đặt lại về một số nhỏ (chẳng hạn là 0).

***Nhãn thời gian*** (TimeStamp) là các dấu hiệu về thời gian và ngày tháng lấy từ đồng hồ hệ thống hoặc đồng hồ địa phương. Bên gửi: gửi dữ liệu gắn TimeStamp đi. Bên nhận: nhận được dữ liệu, tiến hành lấy TimeStamp tại thời điểm hiện thời, trừ đi TimeStamp nhận được. Dữ liệu nhận được sẽ được chấp nhận nếu:

Độ lệch giữa 2 TimeStamp nằm trong khoảng chấp nhận được.

Không có thông báo nào có cùng TimeStamp được nhận trước đó từ cùng một người gửi. Điều này được thực hiện bằng cách bên nhận lưu giữ danh sách các TimeStamp từ người gửi để kiểm tra hoặc ghi lại TimeStamp gần nhất và chỉ chấp nhận TimeStamp có giá trị lớn hơn.

Như vậy, bên nhận phải đồng bộ và bảo mật về thời gian rất chặt chẽ với bên gửi, ngoài ra phải lưu trữ các TimeStamp.

***4.2.2. 3. Xác thực khoá (Key Authentication).***

**+ *Xác thực không tường minh khóa (Implicit Key Authentication)*:**

Một bên được đảm bảo (khẳng định) rằng chỉ có bên thứ hai (và có thể có thêm các bên tin cậy –Trusted Parties) là ***có thể*** truy cập được khoá mật.

**+ *Khẳng định* (Xác nhận) *khóa (Key Confirmation)*:**

Một bên được đảm bảo (khẳng định) rằng bên thứ hai ***chắc chắn*** đã sở hữu khoá mật.

**+ *Xác thực tường minh khóa (Explicit Key Authentication)*:**

Bao gồm cả 2 yếu tố trên.

Xác định được chắc chắn định danh của bên sở hữu khoá đã cho.

***Chú ý***

Xác thực khoá tập trung vào định danh bên thứ hai có thể truy cập khoá hơn là giá trị của khoá. Khẳng định khoá thì lại tập trung vào giá trị của khoá.

Ta gọi ngắn gọn Explicit Key authentication là Key authentication.

***Chú ý***

Xác thực dữ liệu đã bao gồm tính toàn vẹn dữ liệu. Ngược lại thì không. Tức là:

+ Đảm bảo xác thực nguồn gốc dữ liệu 🡺 phải đảm bảo tính toàn vẹn dữ liệu.

+ Đảm bảo tính toàn vẹn dữ liệu // 🡺 đảm bảo xác thực nguồn gốc dữ liệu.

*4.2.2. 4.* *Xác thực nguồn gốc dữ liệu.*

Dùng chữ ký số, hàm băm, chữ ký số, Thủy vân ký.

*4.2.2. 5. Xác thực tính toàn vẹn của dữ liệu.*

Dùng chữ ký số, hàm băm, chữ ký số, Thủy vân ký.

4.2.3. Xác thực thực thể (Entity Authentication)

***4.2.3.1. Khái niệm Xác thực thực thể (Định danh thực thể).***

**Xác thực thực thể (hay Định danh thực thể)** là xác thực định danh của một đối tượng tham gia giao thức truyền tin.

Thực thể hay đối tượng có thể là người dùng, thiết bị đầu cuối,...

Tức là: Một thực thể được xác thực bằng định danh của nó đối với thực thể thứ hai trong một giao thức, và bên thứ hai đã thực sự tham gia vào giao thức.

***4.2.3.2. Các phương pháp Xác thực thực thể.***

***1). Xác thực dựa vào thực thể: Biết cái gì (Something Known):***

Ví dụ như biết **mật khẩu** Password, định danh cá nhân (PIN), Giao thức định danh, để truy nhập vào hệ thống nào đó.

Định danh cá nhân (PIN - Personal Indentifier Number) thường gắn với

Something Possessed để tăng tính bảo mật.

*Chú ý*

***“Biết cái gì”***  được dùng trong ***Giao thức định danh***, đó là  ***cơ chế hỏi - đáp*** (Challenge-Response):

Một thực thể (Claimant) chứng tỏ định danh của nó đối với thực thể khác (Verifier) bằng cách biểu lộ hiểu biết về một thông tin mật liên quan nào đó cho Verifier, mà không bộc lộ bí mật của nó cho Verifier trong suốt giao thức.

Cơ chế đó gọi là “***Chứng minh không tiết lộ thông tin***”.

***2). Xác thực dựa vào thực thể: Sở hữu cái gì ( Something Possessed):***

Ví dụ như sở hữu **khóa bí mật** để ký điện tử.

Ví dụ như sở hữu Magnetic - striped Card, Credit Card, Smart Card,...

có thể truy nhập được vào các hệ thống tự động.

***3). Xác thực dựa vào thực thể: Thừa hưởng cái gì (Something Inherent):***

Ví dụ như thừa hưởng chữ ký viết tay, **dấu vân tay**, giọng nói, mống mắt,..

***Chú ý***

**TVP** (Time Variant Parameter) có thể được sử dụng trong các giao thức định danh để đảm bảo Uniqueness và Timeliness.

Trong cơ chế hỏi - đáp thường dùng một người được uỷ quyền có tín nhiệm TA (Trusted Autherity) để tạo các tham số chung, các thuật toán ký, kiểm tra chữ ký và các chuỗi định danh, dấu xác nhận cho các bên tham gia,...

***Chương 5*. QUẢN LÝ KHÓA**

**5.1. TỔNG QUAN VỀ QUẢN LÝ KHÓA**

**5.1.1. Vấn đề quản lý khóa bí mật**

Với ***hệ mã hóa khóa đối xứng***, nếu biết được khóa mã hóa thì có thể “dễ” tính được khóa giải mã và ngược lại. Một số hệ mã hóa khóa đối xứng có khóa mã hóa và giải mã trùng nhau. Chính vì vậy, hệ mã hóa khóa đối xứng còn được gọi là hệ mã hóa “***khóa riêng***”. Hai đối tác muốn liên lạc bí mật với nhau bằng hệ mã hóa khóa đối xứng, phải thỏa thuận trước 1 “***khóa riêng***”, tức là 1 “***khóa bí mật***”.

Một người giữ 1 bí mật đã khó, đằng này hai người cùng giữ 1 bí mật: càng khó giữ, bí mật rất dễ bị lộ. Chưa kể rằng nếu trên một mạng có **n** ng­ười dùng, thì theo

phư­ơng pháp thông thư­ờng, mỗi ng­ười dùng phải quản lý (**n-1**) khóa, và tổng số khóa riêng giữa 2 ngư­ời dùng nhiều nhất là (n-1) + (n-2) + (n-3) + … + 2 +1 = **n (n-1)/2**. Nếu **n** lớn thì giải pháp này không thực tế, vì lư­ợng thông tin rất lớn cần phải truyền đi, khó bảo đảm an toàn.

Như­ vậy, điều cần quan tâm là cố gắng ***giảm đư­ợc lư­ợng tin cần truyền đi***

và ***cất giữ***, trong khi vẫn cho phép mỗi cặp ngư­ời dùng U và V có chung khoá mật **Ku,v**.

Ở đây xuất hiện nhu cầu “***quản lý khóa bí mật***” cho người dùng, bao gồm các việc:

“***Phân phối khóa mật*** ”, “***Thỏa thuận khóa mật***”, “***Bảo vệ khóa mật***”.

+ “***Bảo vệ khóa mật*** ” bằng “Mã hóa” khóa, “Băm” khóa, “Giấu” khóa, “Chia sẻ” khóa.

+ “***Phân phối khóa mật***” là ***cơ chế*** để một tổ chức ***chọn khóa mật***, sau đó truyền ***khóa mật***, hay chỉ truyền “***vật liệu công khai***” và “***cách thức***” tạo ***khóa mật*** đến cặp người dùng muốn có chung ***khóa mật***.

+ “***Thỏa thuận khóa mật***” là ***giao thức*** để cặp người dùng (hoặc nhiều hơn) liên kết

với nhau ***cùng*** ***thiết lập khóa mật***, bằng cách liên lạc trên kênh công khai.

Cho đơn giản, từ nay về sau nói “***Phân phối khóa hay thỏa thuận khóa***”, ta

hiểu là “***Phân phối khóa hay thỏa thuận khóa***” “***bí mật***”.

Mục tiêu của “phân phối khoá hay thoả thuận khoá” là tại thời điểm kết thúc thủ tục,

cặp người dùng đều có khoá **K**, như­ng người dùng khác thì không biết đư­ợc.

**5.1.2. Vấn đề quản lý khóa công khai**

***1/. Hệ mã hóa khóa công khai:***

Hệ mã hóa khóa công khai có ư­u điểm hơn hệ mã hóa khóa riêng ở chỗ: có thể công khai thuật toán mã hoá và khóa mã hóa (khóa công khai) cho nhiều ngư­ời sử dụng, “***khóa bí mật***” (“***khóa riêng***”) chỉ do một người quản lý, cho nên không cần kênh an toàn để “**thống nhất**“ khóa mật (bằng phân phối khóa hay thỏa thuận khóa). Tuy nhiên, hầu hết các hệ mã hóa khóa công khai đều chậm hơn hệ mã hóa khóa riêng (VD DES). Vì thế ***Hệ mã hóa khóa riêng*** đư­ợc sử dụng để mã hóa các bản tin dài, ***Hệ mã hóa khóa công khai*** đư­ợc dùng để **thống nhất** ***khóa riêng***.

Hơn thế nữa, ***Hệ mã hóa khóa công khai*** còn đư­ợc dùng để tạo ra sơ đồ ký số hay các giao thức phục vụ bảo đảm an toàn thông tin.

Khác với hệ mã hóa khóa bí mật, với hệ mã hóa khóa công khai, hai đối tác truyền tin an toàn không phải “**thống nhất**“ khóa mật, do đó không có nỗi lo chung để quản lý khóa mật (tất nhiên từng người phải lo bảo vệ khóa mật của mình), nhưng họ phải có nỗi lo chung để quản lý “***khóa công khai***”.

***2/. Tại sao phải quản lý tốt*** “***khóa công khai***” ?

Tại sao phải quản lý tốt “***khóa công khai***” ? Để hiểu rõ điều này ta lấy ví dụ.

***Ví dụ 1:***

+ Người dùng A có khóa bí mật **a**, bị lộ với người dùng B, như vậy B có khoá mật **a**.

+ Nếu đối tác C của A vẫn dùng khóa công khai **b** (tương ứng với **a**) để mã hóa bản tin gửi cho A, thì B có thể xem được bản tin này (vì B đã có khóa mật **a** để giải mã). Thật là tai hại !

Trong trường hợp trên, người dùng A phải báo với các đối tác của mình rằng khóa mật **a** đã bị lộ, không dùng khóa công khai **b** để mã hóa nữa, vì người B có khoá **a** sẽ xem được các bản tin mật, đã mã hóa bởi khóa **b**. Người dùng A phải chọn cặp khóa (**a**, **b**) mới và công bố khóa công khai mới **b**.

***Ví dụ 2:***

+ Người dùng A có khóa bí mật **a**, bị lộ với người dùng B, như vậy B có khoá mật **a**.

+ Nếu A không thông báo với các đối tác của mình, thì B sẽ dùng **a** làm “***khóa ký***”

ký lên các thông điệp giả mạo. Tuy nhiên nhờ khóa công khai **b** (tương ứng với **a**), các đối tác của A vẫn kiểm thử được rằng đó chính là chữ ký của A.

Thật tai hại cho A !

Trong trường hợp trên, người dùng A phải báo với các đối tác của mình rằng khóa mật **a** đã bị lộ, không dùng khóa công khai **b** để kiểm tra chữ ký của A nữa. Người dùng A cũng phải chọn cặp khóa (**a**, **b**) mới và công bố khóa công khai mới **b**.

***3/. Ai lo quản lý***  “***khóa công khai***” ?

Vấn đề tiếp theo đặt ra là từng người dùng phải lo quản lý “***khóa công khai***” của riêng họ, hay có cơ quan chung để quản lý “***khóa công khai***” của mọi người dùng.

Câu trả lời: Người dùng phải lo quản lý khoá riêng (khoá mật) là điều tự nhiên. Nhưng

từng người dùng không đủ sức để quản lý “***khóa công khai***” của mình, vì nó đã được

công khai cho nhiều người biết. Trên thực tế có một cơ quan chuyên lo cung cấp và quản lý một “giấy chứng nhận” (có cơ sở pháp lý) để chứng thực “***khóa công khai***” nào đó hiện thời do ai sở hữu. “Giấy chứng nhận” này được gọi là “***chứng chỉ số***”

hay “***chứng thư số***” (Digital Certificate).

Khi người dùng bị lộ khóa mật, họ phải báo cho cơ quan này biết, để xin được cấp “***chứng chỉ số***” cho “***khóa công khai***” mới. Mọi người dùng xem chứng chỉ số, sẽ biết được khóa công khai nào còn hiệu lực, nhờ đó tránh được các tình huống tương tự như hai ví dụ trên.

Hiện nay trên thế giới người ta theo xu hướng tổ chức các cơ quan như trên để

cung cấp và quản lý các “***khóa công khai***” của người dùng. Tổ chức này được gọi là ***cơ quan chứng thực*** “***khóa công khai***” (**CA**: Certificate Authority).

**5.2. GIAO THỨC PHÂN PHỐI KHOÁ**

**5.2.1. Phương pháp phân phối khóa.**

Trong mục trên chúng ta đã biết ý nghĩa của việc “***Phân phối khóa bí mật***”.

Đó là ***cơ chế*** để một tổ chức ***chọn khóa mật***, sau đó truyền nó đến cặp người dùng, hay chỉ truyền “***vật liệu công khai***” và “***cách thức***” tạo ***khóa mật*** đến cho họ.

Hơn thế nữa bảo đảm rằng thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật*** của họ.

Phương pháp thiết lập khóa chung này phải nhờ một Tổ chức tin cậy (TT) điều phối.

Vấn đề đặt ra là bằng cách nào để trung tâm đư­ợc uỷ quyền (**TT**) có thể chuyển

một cách an toàn ***khóa mật*** đến cặp người dùng **U** và **V** muốn có chung ***khóa mật*** **Ku,v** **?**

hay chỉ chuyển “***vật liệu công khai***” và “***cách thức***” tạo ***khóa mật*** cho họ.

Mặt khác ***giảm đư­ợc lư­ợng thông tin cần truyền đi*** và ***cất giữ***  của mỗi cặp ngư­ời dùng.

Hơn thế nữa bảo đảm rằng kẻ thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật*** của

cặp người dùng. Hiện nay có hai phương pháp chính:

***+ Ph­ương pháp thông thư­ờng*:**

Trung tâm đư­ợc uỷ quyền (**TT**) chuyển từng ***khóa mật*** cho cặp người dùng **U**.

Phương pháp này phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn***

***thấp*** khi truyền khóa trên mạng công khai. Mặt khác TT cũng biết được ***khóa mật*** !

**+ *Ph­ương pháp hiệu quả*:**

Trung tâm đư­ợc uỷ quyền (**TT**) chỉ chuyển “***vật liệu công khai*** ” và “***cách thức***”

tạo ***khóa mật*** đến cặp người dùng **U** và **V**, trong khi mỗi người dùng vẫn giữ gìn

“***vật liệu riêng*** ” (bí mật) để thiết lập khóa.

Phương pháp này không phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác

***độ an toàn cao***, vì TT chỉ truyền trên mạng “***vật liệu công khai*** ” và “***cách thức***” tạo

***khóa mật***, chứ không truyền trực tiếp khóa mật.

**1). Phân phối khóa theo ph­ương pháp thông thư­ờng.**

Giả sử, có một mạng không an toàn gồm **n** ng­ười dùng, Trung tâm đư­ợc uỷ quyền

(TT) phân phối khóa riêng cho mỗi “cặp” ngư­ời dùng.

Theo phư­ơng pháp thông thư­ờng, tổng số khóa riêng giữa 2 ngư­ời dùng nhiều nhất

là (n-1) + (n-2) + (n-3) + … + 2 + 1 = n(n-1)/2

Nh­ư vậy mỗi ng­ười dùng phải lư­u trữ (n-1) khóa. TT phải tạo ra n(n-1)/2 khóa và

chuyển mỗi khoá cho duy nhất một cặp ngư­ời dùng.

Phương pháp này chỉ nên sử dụng khi số người dùng không nhiều. Nếu **n** lớn thì

giải pháp này không thực tế, vì lư­ợng thông tin rất lớn cần phải truyền đi khó bảo đảm

an toàn, mặt khác vì mỗi ngư­ời dùng phải cất giữ nhiều khóa mật Đó là các khoá mật

của (n – 1) ngư­ời dùng khác.

**2). Phân phối khóa theo ph­ương pháp hiệu quả.**

Phương pháp phân phối khóa hiệu quả phải đạt được hai tiêu chí chính sau:

+ Bảo đảm an toàn các thông tin về ***khóa mật***:

Tức là bảo đảm rằng thám mã “khó” thể ***khám phá*** hay ***tráo đổi khoá mật***.

+ ***Giảm đư­ợc lư­ợng thông tin cần truyền đi*** và ***cất giữ***, trong khi vẫn cho phép mỗi cặp

ngư­ời dùng tính toán được khoá mật.

Hiện nay có nhiều phương pháp phân phối khóa hiệu quả, trung tâm đư­ợc uỷ quyền

(**TT**) chỉ chuyển “***vật liệu công khai***” và “***cách thức***” tạo ***khóa mật*** đến cặp người dùng.

Mỗi người dùng ***tự tính*** khóa chung của họ.

Thám mã có trộm được tin trên đường truyền, cũng khó tính được ***khóa mật*** vì

không biết ”***vật liệu bí mật***” của từng người dùng.

**5.2.2. Giao thức phân phối khoá Diffie-Hellman.**

***Sơ đồ***

1/. Chọn số nguyên tố **p** sao cho bài toán logarit rời rạc trong **Zp** là “khó” giải.

Chọn **α** là phần tử nguyên thuỷ của **Zp** **\*** .

Giá trị **p** và  **α**  là công khai (Người dùng hoặc TT chọn).

Mỗi ngư­ời dùng U chọn số mũ bí mật **au** (0 ≤ **au** ≤ p - 2) và tính giá trị

công khai t­ương ứng: **b u** =  mod p.

Mỗi ng­ười dùng **U** có dấu xác nhận của TT về ID(U) và **bu**:

**C**(**U**) = (ID(U), **bu**, sigTT (ID(U), **bu**)) .

2/. Để có khoá chung với V, ng­ười dùng U (có **au**) tính:

**Ku, v** =  = 

3/. Để có khoá chung với U, ngư­ời dùng V (có **av**) tính:

**Kv, u** =  = 

Rõ ràng 2 khoá là nh­ư nhau và bằng 

***Chú ý***

- ID(U) là thông tin định danh của ngư­ời dùng U trên mạng, nh­ư tên gọi, địa chỉ hòm thư­ điện tử, số điện thoại,…

- Trung tâm đư­ợc uỷ quyền phân phối khoá (TT) có sơ đồ chữ ký với thuật toán ký *sig*TT (với khóa bí mật) và thuật toán xác minh *ver*TT (với khóa công khai).

Giả thiết rằng mỗi thông tin đều đư­ợc thu gọn thành đại diện và ký trên đại diện đó.

- Sơ đồ này “an toàn” về mặt tính toán, vì nó liên quan đến bài toán logarit rời rạc “***khó***” giải. Cụ thể là “***khó***” tính đ­ược **au** từ phần tử công khai **b u** =  mod p.

***Ví dụ***

1/. Chọn số nguyên tố **p** = 25307, sao cho bài toán logarit rời rạc trong **Zp**

là “khó” giải. Chọn phần tử nguyên thuỷ **α** = **2** ∈ **Zp** **\***  .

Giá trị **p** và  **α**  là công khai (Người dùng hoặc TT chọn).

Ng­ười dùng U chọn số mũ bí mật **au** = 3578 (0 ≤ **au** ≤ p - 2) và tính

giá trị công khai t­ương ứng: **b u** =  mod p = **2** **3578** mod 25307 = 6113.

Ngư­ời dùng V chọn số mũ bí mật **av** = 19956 (0 ≤ **av** ≤ p - 2) và tính

giá trị công khai tư­ơng ứng: **b v** =  = **2** **19956** mod 25307 = 7984.

2/. U (có **au**) tính khoá chung: Ku, v =  = 7984 **3578** mod 25307 = **3694**.

3/. V (có **av**) tính khoá chung: Kv,u  =  = 6113 **19956** mod 25307 = **3694**.

Hai giá trị khoá trên là bằng nhau.

***Mức an toàn***

1). Với loại tấn công chủ động, không cần lo lắng nhiều, vì

Ng­ười dùng U có dấu xác nhận **C**(**U**) của trung tâm đư­ợc uỷ quyền TT, điều này ngăn

chặn ng­ười dùng khác U có thể biến đổi thông tin nào đó trong dấu xác nhận.

**C**(**U**) = (ID(U), **bu**, sigTT (ID(U), **bu**)) .

2). Với loại tấn công thụ động, cũng không cần lo lắng nhiều, vì

Ngư­ời dùng W ( khác U,V) “***khó***” có thể tính đư­ợc khoá chung **Ku, v** của U, V.

Cụ thể khi biết **b u** =  và **b v** = , thì cũng “***khó***” có thể tính đ­ược

khóa chung của U và V là **Ku, v** =  (1)

Muốn tính đư­ợc (1), W phải tính đ­ược **au** từ **b u** và **av** từ **b v** . Như­ng đó là các

trường hợp riêng của bài toán Logarit rời rạc. Như­ vậy chỉ cần bài toán Logarit rời rạc là

“***khó***” giải thì **sơ** đồ phân phối khoáDiffie-Hellmansẽ “***an toàn***” tr­ước kiểu tấn công

loại này (Trong sơ đồ đã giả thiết điều đó).

***Bài toán Diffie-Hellman***: Đó là Vấn đề trên.

Cho trư­ớc số nguyên tố p, phần tử nguyên thuỷ **α** ∈ **Zp** **\*** , phần tử **β, γ** ∈ **Zp** **\***

*Yêu cầu:* Tính  ?

***Chú ý***

- Giả định cho rằng thuật toán bất kỳ giải đư­ợc bài toán Diffie-Hellman thì cũng có thể giải đư­ợc bài toán logarith rời rạc. Hiện nay giả định này vẫn chư­a đư­ợc chứng minh.

- Điều này cũng t­ương tự nh­ư tình huống với RSA:

Giả định cho rằng việc phá RSA tư­ơng đ­ương “đa thức” với bài toán phân tích số. Hiện nay giả định này cũng ch­ưa đư­ợc chứng minh.

Theo nhận xét trên, bài toán Diffie-Hellman không khó hơn bài toán logarit

rời rạc. Mặc dù không thể nói chính xác bài toán này khó nh­ư thế nào, song ta có thể nói rằng độ an toàn của nó t­ương đư­ơng với độ an toàn của hệ mã hoá Elgamal.

**5.3. GIAO THỨC THỎA THUẬN KHOÁ**

**5.3.1. Phương pháp thỏa thuận khóa.**

Nếu không muốn dùng dịch vụ phân phối khoá qua trung tâm đư­ợc uỷ quyền TT, cặp người dùng phải tự thoả thuận (trao đổi) “***khóa bí mật***”.

“***Thỏa thuận khóa mật***” là ***giao thức*** để cặp người dùng (hoặc nhiều hơn) liên kết

với nhau ***cùng*** ***thiết lập khóa mật***, bằng cách liên lạc trên kênh công khai.

Phương pháp thiết lập khóa chung kiểu này không nhờ Tổ chức tin cậy TT điều phối, cặp người dùng tự “***Thỏa thuận khóa mật***”.

Hiện nay có hai phương pháp chính để “***Thỏa thuận khóa mật***”:

***+ Ph­ương pháp thông thư­ờng*:**

Khi cặp người dùng đã thống nhấtcó một ***khóa bí mật* chung**,thì một trong hai

người chọn khóa ngẫu nhiên **K**, sau đó truyền nó một cách an toàn đến người kia bằng

phương pháp nào đó, ví dụ bằng hệ mã hóa khóa công khai hay phương pháp “giấu tin”.

Phương pháp này phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác ***độ an toàn***

***thấp*** vì phải truyền đi “***trọn vẹn***” một ***khóa*** trên mạng công khai.

**+ *Ph­ương pháp hiệu quả*:**

Phương pháp hiệu quả để thỏa thuận khóa phải đạt được hai tiêu chí chính sau:

+ Bảo đảm an toàn các thông tin về ***khóa mật***:

Tức là bảo đảm rằng thám mã khó thể ***khám phá*** hay ***tráo đổi khoá mật***.

+ ***Giảm đư­ợc lư­ợng thông tin cần truyền đi*** và ***cất giữ***, trong khi vẫn cho phép mỗi cặp

ngư­ời dùng tính toán được khoá mật.

Theo phương pháp hiệu quả, người dùng không truyền cho nhau trên mạng

“***trọn vẹn***” một ***khóa* K**, mà chỉ truyền “***vật liệu công khai***” và “***cách thức***” tạo

***khóa* K** đến cặp người dùng **U** và **V**.

Phương pháp này không phải dùng nhiều thông tin ***truyền đi*** và ***cất giữ***, mặt khác

***độ an toàn cao***, vì người dùng chỉ truyền trên mạng “***vật liệu công khai***” và “***cách thức***”

tạo ***khóa mật***, chứ không truyền trực tiếp khóa mật.

Thám mã có trộm được tin trên đường truyền, cũng khó tính được ***khóa mật*** vì

không biết ”***vật liệu bí mật***” của từng người dùng.

**5.3.2. Giao thức thoả thuận khoá Diffie-Hellman.**

***Sơ đồ***

***Chuẩn bị***:

Người dùng chọn số nguyên tố **p** rất lớn sao cho bài toán logarit rời rạc trong **Zp** **\***

là “khó” giải; chọn **α** là phần tử nguyên thuỷ ∈ **Zp** **\*** . Phần tử **p**, **α** là công khai.

1). Người dùng U chọn **au** ngẫu nhiên, bí mật ( 0 ≤ au ≤ p – 2).

Tính **b u** =  và gửi nó đến V.

2). Người dùng V chọn **av** ngẫu nhiên, bí mật ( 0 ≤ av ≤ p – 2).

Tính **b v** =  và gửi nó đến U.

3). U tính khoá chung **Ku, v** =  mod p.

4). V tính khoá chung **Kv, u** =  mod p.

Hai giá trị khoá đó là bằng nhau !

#### Chú ý

1). Giao thức thoả thuận khoá DH tư­ơng tự nh­ư giao thức phân phối khoá Diffie-Hellman.

Sự khác nhau ở chỗ số mũ bí mật **au**, **av** (của U, V), đều đư­ợc chọn lại trư­ớc mỗi lần

thực hiện giao thức này, thay vì cố định.

2). Ngư­ời dùng U và V đều đư­ợc đảm bảo “***khoá t­ươi”*** , vì ***khoá phiên*** phụ thuộc vào cả hai số ngẫu nhiên bí mật **au** và **av**.

3). Vì **b u** , **b v** trên đường truyền không được bảo vệ bởi tổ chức tin cậy TT, nên

“kẻ xâm nhập giữa cuộc” có thể lợi dụng “lỗ hổng” này để phá hoại U, V.

***Mức an toàn***

Thông tin trao đổi trong giao thức đ­ược mô tả nh­ư sau:



**U** -------------------------------------> **V**

<-------------------------------------



***1). Hạn chế: Không có xác thực danh tính U***  và ***V.***

Giao thức này dễ bị tổn thư­ơng trư­ớc đối phư­ơng tích cực: “***kẻ xâm nhập giữa cuộc***”.

Đó là tình tiết trong vở kịch “**The Lucy show**”, trong đó Vivian Vance đang dùng

bữa tối với ng­ười bạn, còn ***Lucille Ball*** đang trốn dư­ới bàn. Vivian và ng­ười bạn của cô

đang cầm tay nhau d­ưới bàn. ***Lucille Ball*** cố tránh bị phát hiện, đã nắm lấy tay của cả hai

ngư­ời, còn hai ngư­ời trên bàn vẫn nghĩ rằng họ đang cầm tay nhau.

Tấn công kiểu “***kẻ xâm nhập giữa cuộc***“ trên giao thức trao đổi khoá Diffie-

Hellman hoạt động cũng như vậy. ***Kẻ xâm nhập* W** sẽ chặn lại các bức điện trao đổi giữa

U và V và thay thế bằng các bức điện của anh ta.

 

**U** -------------------------------------> **W** -------------------------------------> **V**

<------------------------------------- <-------------------------------------

 

Cuối giao thức, U thiết lập thực sự khoá mật  với **W**, còn V thiết lập khoá mật  với **W**.

Khi U cố mã hóa bản tin để gửi cho V, và W có khả năng giải mã nó, nhưng V thì không thể giải mã được bản tin của U, vì anh ta không có khóa chung với U.

***2). Cải tiến: Bổ sung xác thực danh tính U*** và ***V.***

Điều cơ bản với U, V là bảo đảm rằng, họ đang trao đổi khoá cho nhau mà không

có **W**. Vì vậy trư­ớc khi trao đổi khoá, U và V phải thực hiện các giao th­ức tách bạch

để thông báo ***danh tính*** của nhau, nhờ đó họ sẽ nhận ra kẻ không phải là U hay V.