CHƯƠNG 1

MẬT MÃ CỔ ĐIỂN

1.1 MỞ ĐẦU - MỘT SỐ HỆ MẬT ĐƠN GIẢN

Đối tượng cơ bản của mật mã là tạo ra khả năng liên lạc trên một kênh không mật cho hai người sử dụng (tạm gọi là Alice và Bob) sao cho đối phương (Oscar) không thể hiểu được thông tin được truyền đi. Kênh này có thể là một đường dây điện thoại hoặc một mạng máy tính. Thông tin mà Alice muốn gửi cho Bob (bản rõ) có thể là một văn bản tiếng Anh, các dữ liệu bằng số hoặc bất cứ tài liệu nào có cấu trúc tuỳ ý. Alice sẽ mã hoá bản rõ bằng một khoá được xác định trước và gửi bản mã kết quả trên kênh. Oscar có bản mã thu trộm được trên kênh song không thể xác định nội dung của bản rõ, nhưng Bob (người đã biết khoá mã) có thể giải mã và thu được bản rõ.

Ta sẽ mô tả hình thức hoá nội dung bằng cách dung khái niệm toán học như sau:

Định nghĩa 1.1

Một hệ mật là một bộ 5 $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{E},\mathcal{D})$ thoả mãn các điều kiện sau:

1. *$\mathcal{P}$ là một tập hữu hạn các bản rõ có thể.*
2. *$\mathcal{C}$ là một tập hữu hạn các bản mã có thể.*
3. *$\mathcal{K}$ (không gian khoá) là tập hữu hạn các khoá có thể.*
4. *Đối với mỗi $k \in \mathcal{K}$ có một quy tắc mã e_k: $\mathcal{P} \to \mathcal{C}$ và một quy tắcv giải mã tương ứng $d_k \in D$. Mỗi e_k: $\mathcal{P} \to \mathcal{C}$ và d_k: $\mathcal{C} \to \mathcal{P}$ là những hàm mà:*

$$d_k(e_k(x)) = x \text{ với mọi bản rõ } x \in \mathcal{P}.$$

Trong tính chất 4 là tính chất chủ yếu ở đây. Nội dung của nó là nếu một bản rõ x được mã hoá bằng e_k và bản mã nhận được sau đó được giải mã bằng d_k thì ta phải thu được bản rõ ban đầu x. Alice và Bob sẽ áp dụng thủ tục sau dùng hệ mật khoá riêng. Trước tiên họ chọn một khoá ngẫu nhiên K $\in \mathcal{K}$. Điều này được thực hiện khi họ ở cùng một chỗ và không bị Oscar theo dõi hoặc khi họ có một kênh mật trong trường hợp họ ở xa nhau. Sau đó giả sử Alice muốn gửi một thông baó cho Bob trên một kênh không mật và ta xem thông báo này là một chuỗi:

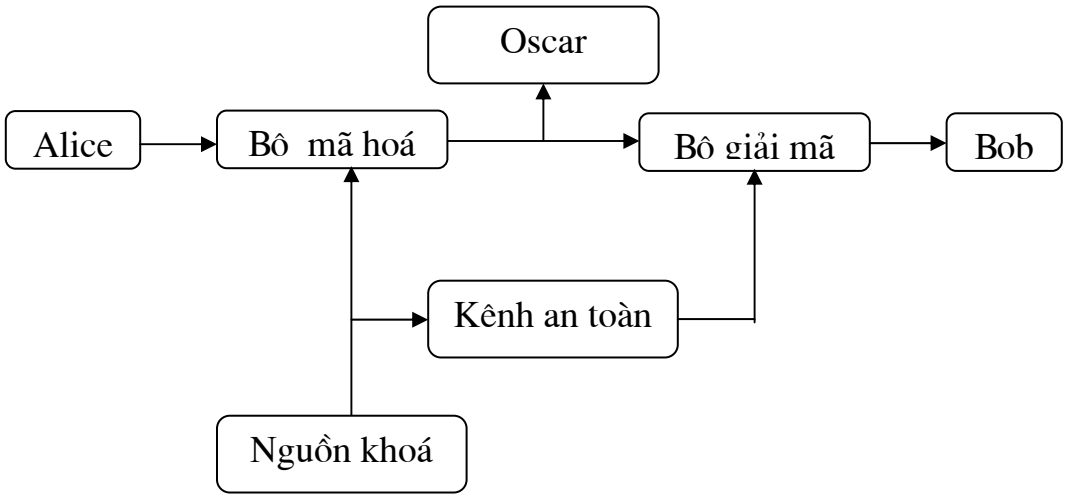
$$x = x_1,x_2 ,. . .,x_n$$

với số nguyên n ≥ 1 nào đó. Ở đây mỗi ký hiệu của mỗi bản rõ $x_i \in \mathcal{P}$, $1 \leq i \leq n$. Mỗi x_i sẽ được mã hoá bằng quy tắc mã e_k với khoá K xác định trước đó. Bởi vậy Alice sẽ tính $y_i = e_k(x_i)$, $1 \leq i \leq n$ và chuỗi bản mã nhận được:

$$y = y_1,y_2 ,. . .,y_n$$

sẽ được gửi trên kênh. Khi Bob nhận đươc $y_1,y_2 ,. . .,y_n$ anh ta sẽ giải mã bằng hàm giải mã d_k và thu được bản rõ gốc $x_1,x_2 ,. . .,x_n$. Hình 1.1 là một ví dụ về một kênh liên lạc

Hình 1.1. Kênh liên lạc



Rõ ràng là trong trường hợp này hàm mã hoá phải là hàm đơn ánh (tức là ánh xạ 1-1), nếu không việc giải mã sẽ không thực hiện được một cách tường minh. Ví dụ

$$y = e_k(x_1) = e_k(x_2)$$

trong đó $x_1 \neq x_2$, thì Bob sẽ không có cách nào để biết liệu sẽ phải giải mã thành x_1 hay x_2 . Chú ý rằng nếu $\mathcal{P} = \mathcal{C}$ thì mỗi hàm mã hoá là một phép hoán vị, tức là nếu tập các bản mã và tập các bản rõ là đồng nhất thì mỗi một hàm mã sẽ là một sự sắp xếp lại (hay hoán vị) các phần tử của tập này.

1.1.1 Mã dịch vòng (shift cipher)

Phần này sẽ mô tả mã dịch (MD) dựa trên số học theo modulo. Trước tiên sẽ điểm qua một số định nghĩa cơ bản của số học này.

Định nghĩa 1.2

Giả sử a và b là các số nguyên và m là một số nguyên dương. Khi đó ta viết a ≡ b (mod m) nếu m chia hết cho b-a. Mệnh đề a ≡ b (mod m) được gọi là " a đồng dư với b theo modulo m". Số nguyên m được gọi là mudulus.

Giả sử chia a và b cho m và ta thu được thương nguyên và phần dư, các phần dư nằm giữa 0 và m-1, nghĩa là $a = q_1m + r_1$ và $b = q_2m + r_2$ trong đó $0 \le r_1 \le m-1$ và $0 \le r_2 \le m-1$. Khi đó có thể dễ dàng thấy rằng a ≡ b (mod m) khi và chỉ khi $r_1 = r_2$. Ta sẽ dùng ký hiệu a mod m (không dùng các dấu ngoặc) để xác định phần dư khi a được chia cho m (chính là giá trị r_1 ở trên). Như vậy: a ≡ b (mod m) khi và chỉ khi a mod m = b mod m. Nếu thay a bằng a mod m thì ta nói rằng a được rút gọn theo modulo m.

Nhận xét: Nhiều ngôn ngữ lập trình của máy tính xác định a mod m là phần dư trong dải - m+1,.. ., m-1 có cùng dấu với a. Ví dụ -18 mod 7 sẽ là -4, giá trị này khác với giá trị 3 là giá trị được xác định theo công thức trên. Tuy nhiên, để thuận tiện ta sẽ xác định a mod m luôn là một số không âm.

Bây giờ ta có thể định nghĩa số học modulo m: Z_m được coi là tập hợp {0,1,. . .,m-1} có trang bị hai phép toán cộng và nhân. Việc cộng và nhân trong Z_m được thực hiện giống như cộng và nhân các số thực ngoài trừ một điểm làcác kết quả được rút gọn theo modulo m.

Ví dụ tính 11× 13 trong Z_{16} . Tương tự như với các số nguyên ta có 11 ×13 = 143. Để rút gọn 143 theo modulo 16, ta thực hiện phép chia bình thường: 143 = 8 × 16 + 15, bởi vậy 143 mod 16 = 15 trong Z_{16} .

Các định nghĩa trên phép cộng và phép nhân Z_m thảo mãn hầu hết các quy tắc quyen thuộc trong số học. Sau đây ta sẽ liệt kê mà không chứng minh các tính chất này:

1. Phép cộng là đóng, tức với bất kì $a,b \in Z_m$,$a+b \in Z_m$
2. Phép cộng là giao hoán, tức là với a,b bất kì $\in Z_m$
   $$a+b = b+a$$
3. Phép cộng là kết hợp, tức là với bất kì $a,b,c \in Z_m$
   $$(a+b)+c = a+(b+c)$$
4. 0 là phần tử đơn vị của phép cộng, có nghĩa là với a bất kì $\in Z_m$
   $$a+0 = 0+a = a$$

5. Phần tử nghịch đảo của phép cộngcủa phần tử bất kì ($a \in Z_m$) là m-a, nghĩa là a+(m-a) = (m-a)+a = 0 với bất kì $a \in Z_m$.
6. Phép nhân là đóng , tức là với a,b bất kì $\in Z_m$, $ab \in Z_m$.
7. Phép nhân là gioa hoán , nghĩa là với a,b bất kì $\in Z_m$, ab = ba
8. Phép nhân là kết hợp, nghĩa là với $a,b,c \in Z_m$, (ab)c = a(cb)
9. 1 là phần tử đơn vị của phép nhân, tức là với bất kỳ $a \in Z_m$

$$a \times 1 = 1 \times a = a$$

10. Phép nhân có tính chất phân phối đối với phép cộng, tức là đối với $a,b,c \in Z_m$, (a+b)c = (ac)+(bc) và a(b+c) = (ab) + (ac)

Các tính chất 1,3-5 nói lên rằng Z_m lập nên một cấu trúc đại số được gọi là một nhóm theo phép cộng. Vì có thêm tính chất 4 nhóm được gọi là nhóm Aben (hay nhóm gioa hoán).

Các tính chất 1-10 sẽ thiết lập nên một vành Z_m . Ta sẽ còn thấy nhiều ví dụ khác về các nhóm và các vành trong cuốn sách này. Một số ví dụ quên thuộc của vành là các số nguyên Z, các số thực R và các số phức C. Tuy nhiên các vành này đều vô hạn, còn mối quan tâm của chúng ta chỉ giới hạn trên các vành hữu hạn.

Vì phần tử ngược của phép cộng tồn tại trong Z_m nên cũng có thể trừ các phần tử trong Z_m . Ta định nghĩa a-b trong Z_m là a+m-b mod m. Một cách tương có thể tính số nguyên a-b rồi rút gon theo modulo m.

Ví dụ : Để tính 11-18 trong Z_{31}, ta tính 11+13 mod 31 = 24. Ngược lại, có thể lấy 11-18 được -7 rồid sau đó tính -7 mod 31 = 24.

Ta sẽ mô tả mã dịch vòng trên hình 1.2. Nó được xác định trên Z_{26} (do có 26 chữ cái trên bảng chữ cái tiếng Anh) mặc dù có thể xác định nó trên Z_m với modulus m tuỳ ý. Dễ dàng thấy rằng, MDV sẽ tạo nên một hệ mật như đã xác định ở trên, tức là $d_K(e_K(x)) = x$ với mọi $x \in Z_{26}$.

Hình 1.2: Mã dịch vòng

Giả sử $\mathcal{P} = \mathcal{C} = \mathcal{K} = Z_{26}$ với $0 \le k \le 25$, định nghĩa:

$$e_K(x) = x + K \bmod 26$$

và

$$d_K(x) = y - K \bmod 26$$

$(x,y \in Z_{26})$

Nhận xét: Trong trường hợp K = 3, hệ mật thường được gọi là mã Caesar đã từng được Julius Caesar sử dụng.

Ta sẽ sử dụng MDV (với modulo 26) để mã hoá một văn bản tiếng Anh thông thường bằng cách thiết lập sự tương ứnggiữa các kí tự và các thặng dư theo modulo 26 như sau: A $\leftrightarrow$ 0,B $\leftrightarrow$ 1, . . ., Z $\leftrightarrow$ 25. Vì phép tương ứng này còn dùng trong một vài ví dụ nên ta sẽ ghi lại để còn tiện dùng sau này:

A	B	C	D	E	F	G	H	I	J	K	L	M
0	1	2	3	4	5	6	7	8	9	10	11	12

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
13	14	15	16	17	18	19	20	21	22	23	24	25

Sau đây là một ví dụ nhỏ để minh hoạ

Ví dụ 1.1:

Giả sử khoá cho MDV là K = 11 và bản rõ là:

wewillmeetatmidnight

Trước tiên biến đổi bản rõ thành dãy các số nguyên nhờ dùng phép tương ứng trên. Ta có:

22	4	22	8	11	11	12	4	4	19
0	19	12	8	3	13	8	6	7	19

sau đó cộng 11 vào mỗi giá trị rồi rút gọn tổng theo modulo 26

7	15	7	19	22	22	23	15	15	4
11	4	23	19	14	24	19	17	18	4

Cuối cùng biến đổi dãy số nguyên này thành các kí tự thu được bản mã sau:

HPHTWWXPPELEXTOYTRSE

Để giả mã bản mã này, trước tiên, Bob sẽ biến đổi bản mã thành dãy các số nguyên rồi trừ đi giá trịcho 11 (rút gọn theo modulo 26) và cuối cùng biến đổi lại dãy nàythành các ký tự.

Nhận xét: Trong ví dụ trên , ta đã dùng các chữ in hoa ch o bản mã, các chữ thường cho bản rõ đêr tiện phân biệt. Quy tắc này còn tiếp tục sử dụng sau này.

Nếu một hệ mật có thể sử dụng được trong thực tế thì nó phảo thoả mãn một số tính chất nhất định. Ngay sau đây sé nêu ra hai trong số đó:

1. Mỗi hàm mã hoá e_K và mỗi hàm giải mã d_K phải có khả năng tính toán được một cách hiệu quả.

2. Đối phương dựa trên xâu bản mã phải không có khả năng xác định khoá K đã dùng hoặc không có khả năng xác định được xâu bản rõ x.

Tính chất thứ hai xác định (theo cách khá mập mờ) ý tưởng ý tưởng "bảo mật". Quá trình thử tính khoá K (khi đã biết bản mã y) được gọi là mã thám (sau này khái niệm này sẽ đực làm chính xác hơn). Cần chú ý rằng, nếu Oscar có thể xác định được K thì anh ta có thể giải mã được y như Bob bằng cách dùng d_K. Bởi vậy, việc xác định K chí ít cũng khó như việc xác định bản rõ x.

Nhận xét rằng, MDV (theo modulo 26) là không an toàn vì nó có thể bị thám theo phương pháp vét cạn. Do chỉ có 26 khoá nên dễ dàng thử mọi khoá d_K có thể cho tới khi nhận được bản rõ có nghĩa. Điều này được minh hoạ theo ví dụ sau:

Ví du 1.2

Cho bản mã

JBCRCLQRWCRVNBJENBWRWN

ta sẽ thử liên tiếp các khoá giải mã d_0 ,d_1 .. . và y thu được:

j b c r c l q r w c r v n b j e n b w r w n
i a b q b k p q v b q u m a i d m a v q v m
h z a p a j o p u a p t l z h c l z u p u l
g y z o z i n o t z o s k y g b k y t o t k
j x y n y h m n s y n r j e x f a j x s n s j
e w x m x g l m r x m q i w e z i w r m r i
d v w l w f k l q w l p h v o d y h v q l q h
c u v k v e j k p v k o g u c x g u p k p g
b t u j u d i j o u j n f t b w f o j o f
a s t i t c h i n t i m e s a v e s n i n e

Tới đây ta đã xác định được bản rõ và dừng lại. Khoá tương ứng K = 9.

Trung bình có thể tính được bản rõ sau khi thử 26/2 = 13 quy tắc giải mã.

Như đã chỉ ra trong ví dụ trên , điều kiện để một hệ mật an toàn là phép tìm khoá vét cạn phải không thể thực hiện được; tức không gian khoá phải rất lớn. Tuy nhiên, một không gian khoá lớn vẫn chưa đủ đảm bảo độ mật.

1.1.2 Mã thay thế

Một hệ mật nổi tiếng khác là hệ mã thay thế. Hệ mật này đã được sử dụng hàng trăm năm. Trò chơi đố chữ "*cryptogram*" trong các bài báo là những ví dụ về MTT. Hệ mật này được nếu trên hình 1.3.

Trên thực tế MTT có thể lấy cả $\mathcal{P}$ và $\mathcal{C}$ đều là bộ chữ cái tiếng anh, gồm 26 chữ cái. Ta dùng Z_{26} trong MDV vì các phép mã và giải mã đều là các phép toán đại số. Tuy nhiên, trong MTT, thích hợp hơn là xem phép mã và giải mã như các hoán vị của các kí tự.

Hình 1.3 Mã thay thế

Cho $\mathcal{P}=\mathcal{C}=Z_{26}$. $\mathcal{K}$ chứa mọi hoán vị có thể của 26 kí hiệu 0,1, . . . ,25
Với mỗi phép hoán vị $\pi \in \mathcal{K}$, ta định nghĩa:

$$e\pi(x) = \pi(x)$$

và

$$d\pi(y) = \pi^{-1}(y)$$

trong đó π^{-1} là hoán vị ngược của π.

Sau đây là một ví dụ về phép hoán vị ngẫu nhiên π tạo nên một hàm mã hoá (cũng nhưb trước, các kí hiệu của bản rõ được viết bằng chữ thường còn các kí hiệu của bản mã là chữ in hoa).

a	b	c	d	e	f	g	h	i	j	k	l	M
X	N	Y	A	H	P	O	G	Z	Q	W	B	T

n	o	p	q	r	s	t	u	v	w	x	y	Z
S	F	L	R	C	V	M	U	E	K	J	D	I

Như vậy, eπ (a) = X, eπ (b) = N,. . . . Hàm giải mã là phép hoán vị ngược. Điều này được thực hiện bằng cách viết hàng thứ hai lên trước rồi sắp xếp theo thứ tự chữ cái. Ta nhận được:

A	B	C	D	E	F	G	H	I	J	K	L	M
d	l	r	y	v	o	h	e	z	x	w	p	T

N	O	P	Q	R	S	T	U	V	W	X	Y	Z
b	g	f	j	q	n	m	u	s	k	a	c	I

Bởi vậy dπ (A) = d, dπ(B) = 1, . . .

Để làm bài tập, bạn đọc có giải mã bản mã sau bằng cách dùng hàm giải mã đơn giản:

M G Z V Y Z L G H C M H J M Y X S S E M N H A H Y C D L M H A.

Mỗĩ khoá của MTT là một phép hoán vị của 26 kí tự. Số các hoán vị này là 26!, lớn hơn 4×10^{26} là một số rất lớn. Bởi vậy, phép tìm khoá vét cạn không thể thực hiện được, thậm chí bằng máy tính. Tuy nhiên, sau này sẽ thấy rằng MTT có thể dễ dàng bị thám bằng các phương pháp khác.

1.1.3 Mã Affine

MDV là một trường hợp đặc biệt của MTT chỉ gồm 26 trong số 26! các hoán vị có thể của 26 phần tử. Một trường hợp đặc biệt khác của MTT là mã Affine được mô tả dưới đây. trong mã Affine, ta giới hạn chỉ xét các hàm mã có dạng:

$$e(x) = ax + b \bmod 26,$$

a,b $\in Z_{26}$. Các hàm này được gọi là các hàm Affine (chú ý rằng khi a = 1, ta có MDV).

Để việc giải mã có thể thực hiện được, yêu cầu cần thiết là hàm Affine phải là đơn ánh. Nói cách khác, với bất kỳ y $\in Z_{26}$, ta muốn có đồng nhất thức sau:

$$ax + b \equiv y \pmod{26}$$

phải có nghiệm x duy nhất. Đồng dư thức này tương đương với:

$$ax \equiv y-b \pmod{26}$$

Vì y thay đổi trên Z_{26} nên y-b cũng thay đổi trên Z_{26} . Bởi vậy, ta chỉ cần nghiên cứu phương trình đồng dư:

$$ax \equiv y \pmod{26} \qquad (y \in Z_{26}).$$

Ta biết rằng, phương tfình này có một nghiệm duy nhất đối với mỗi y khi và chỉ khi UCLN(a,26) = 1 (ở đây hàm UCLN là ước chung lớn nhất của các biến của nó). Trước tiên ta giả sử rằng, UCLN(a,26) = d >1. Khi đó, đồng dư thức $ax \equiv 0 \pmod{26}$ sẽ có ít nhất hai nghiệm phân biệt trong Z_{26} là x = 0 và x = 26/d. Trong trường hợp này, e(x) = ax + b mod 26 không phải là một hàm đơn ánh và bởi vậy nó không thể là hàm mã hoá hợp lệ.

Ví dụ, do UCLN(4,26) = 2 nên 4x +7 không là hàm mã hoá hợp lệ: x và x+13 sẽ mã hoá thành cùng một giá trị đối với bất kì $x \in Z_{26}$.

Ta giả thiết UCLN(a,26) = 1. Giả sử với x_1 và x_2 nào đó thảo mãn:

$$ax_1 \equiv ax_2 \pmod{26}$$

Khi đó

$$a(x_1 - x_2) \equiv 0 \pmod{26}$$

bởi vậy

$$26 \mid a(x_1 - x_2)$$

Bây giờ ta sẽ sử dụng một tính chất của phép chia sau: Nếu USLN(a,b)=1 và $a \mid bc$ thì $a \mid c$. Vì $26 \mid a(x_1 - x_2)$ và USLN(a,26) = 1 nên ta có:

$$26 \mid (x_1 - x_2)$$

tức là

$$x_1 \equiv x_2 \pmod{26}$$

Tới đây ta chứng tỏ rằng, nếu UCLN(a,26) = 1 thì một đồng dư thức dạng $ax \equiv y \pmod{26}$ chỉ có (nhiều nhất) một nghiệm trong Z_{26} . Do đó , nếu ta cho x thay đổi trên Z_{26} thì ax mod 26 sẽ nhận được 26 giá trị khác nhau theo modulo 26 và đồng dư thức $ax \equiv y \pmod{26}$ chỉ có một nghiệm y duy nhất.

Không có gì đặc biệt đối vơi số 26 trong khẳng định này. Bởi vậy, bằng cách tương tự ta có thể chứng minh được kết quả sau:

Định lí 1.1

Đồng dư thức $ax \equiv b \bmod m$ chỉ có một nghiệm duy nhất $x \in Z_m$ với mọi $b \in Z_m$ khi và chỉ khi UCLN(a,m) = 1.

Vì $26 = 2 \times 13$ nên các giá trị $a \in Z_{26}$ thoả mãn UCLN(a,26) = 1 là a = 1, 3, 5, 7, 9, 11, 13, 15, 17, 19, 21, 23 và 25. Tham số b có thể là một phần tử bất kỳ trong Z_{26}. Như vậy, mã Affine có $12 \times 26 = 312$ khoá có thể (dĩ nhiên con số này quá nhỉ để bảo đảm an toàn).

Bây giờ ta sẽ xét bài toán chung với modulo m. Ta cần một định nghĩa khác trong lý thuyết số.

Định nghĩa 1.3

Giả sử $a \geq 1$ và $m \geq 2$ là các số nguyên. UCLN(a,m) = 1 thì ta nói rằng a và m là nguyên tố cùng nhau. Số các số nguyên trong Z_m nguyên tố cùng nhau với m thường được ký hiệu là $\phi(m)$ (hàm này được gọi là hàm Euler).

Một kết quả quan trọng trong lý thuyết số cho ta giá trị của $\phi(m)$ theo các thừa số trong phép phân tích theo luỹ thừa các số nguyên tố của m. (Một số nguyên p >1 là số nguyên tố nếu nó không có ước dương nào khác ngoài 1 và p. Mọi số nguyên m >1 có thể phân tích được thành tích của các luỹ thừa các số nguyên tố theo cách duy nhất. Ví dụ $60 = 2^3 \times 3 \times 5$ và $98 = 2 \times 7^2$).

Ta sẽ ghi lại công thức cho $\phi(m)$ trong định lí sau:

Định lý 1.2. (thiếu)

Giả sử $m = \prod p_i$

Trong đó các số nguyên tố p_i khác nhau và $e_i > 0$,1

Định lý này cho thấy rằng, số khoá trong mã Affine trên Z_m bằng $m\phi(m)$, trong đó $\phi(m)$ được cho theo công thức trên. (Số các phép chọn của b là m và số các phép chọn của a là $\phi(m)$ với hàm mã hoá là $e(x) = ax + b$). Ví dụ, khi m = 60, $\phi(60) = 2 \times 2 \times 4 = 16$ và số các khoá trong mã Affine là 960.

Bây giờ ta sẽ xét xem các phép toán giải mã trong mật mã Affine với modulo m = 26. Giả sử UCLN(a,26) = 1. Để giải mã cần giải phương trình đồng dư $y \equiv ax+b \pmod{26}$ theo x. Từ thảo luận trên thấy rằng, phương trình

này có một nghiệm duy nhất trong Z_{26} . Tuy nhiên ta vẫn chưa biết một phương pháp hữu hiệu để tìm nghiệm. Điều cần thiết ở đây là có một thuật toán hữu hiệu để làm việc đó. Rất mayb là một số kết quả tiếp sau về số học modulo sẽ cung cấp một thuật toán giải mã hữu hiệu cần tìm.

Định nghĩa 1.4

Giả sử $a \in Z_m$. Phần tử nghịch đảo (theo phép nhân) của a là phần tử $a^{-1} \in Z_m$ sao cho $aa^{-1} \equiv a^{-1}a \equiv 1 \pmod m$.

Bằng các lý luận tương tự như trên, có thể chứng tỏ rằng a có nghịch đảo theo modulo m khi và chỉ khi UCLN(a,m) =1, và nếu nghịch đảo này tồn tại thì nó phải là duy nhất. Ta cũng thấy rằng, nếu $b = a^{-1}$ thì $a = b^{-1}$. Nếu p là số nguyên tố thì mọi phần tử khác không của Z_P đều có nghịch đảo. Một vành trong đó mọi phần tử đều có nghịch đảo được gọi là một trường.

Trong phần sau sẽ mô tả một thuật toán hữu hiệu để tính các nghịch đảo của Z_m với m tuỳ ý. Tuy nhiên, trong Z_{26} , chỉ bằng phương pháp thử và sai cũng có thể tìm được các nghịch đảo của các phần tử nguyên tố cùng nhau với 26: $1^{-1} = 1$, $3^{-1} = 9$, $5^{-1} = 21$, $7^{-1} = 15$, $11^{-1} = 19$, $17^{-1} = 23$, $25^{-1} = 25$. (Có thể dễ dàng kiểm chứng lại điều này, ví dụ: $7 \times 5 = 105 \equiv 1 \bmod 26$, bởi vậy $7^{-1} = 15$).

Xét phương trình đồng dư $y \equiv ax+b \pmod{26}$. Phương trình này tương đương với

$$ax \equiv y-b \pmod{26}$$

Vì UCLN(a,26) =1 nên a có nghịch đảo theo modulo 26. Nhân cả hai vế của đồng dư thức với a^{-1} ta có:

$$a^{-1}(ax) \equiv a^{-1}(y-b) \pmod{26}$$

Áp dụng tính kết hợp của phép nhân modulo:

$$a^{-1}(ax) \equiv (a^{-1}a)x \equiv 1x \equiv x.$$

Kết quả là $x \equiv a^{-1}(y-b) \pmod{26}$. Đây là một công thức tường minh cho x. Như vậy hàm giải mã là:

$$d(y) = a^{-1}(y-b) \bmod 26$$

Hình 1.4 cho mô tả đầy đủ về mã Affine. Sau đây là một ví dụ nhỏ

Hình 1.4 Mật mãA ffine

Cho $\mathcal{P} = \mathcal{C} = Z_{26}$ và giả sử

$\mathcal{P} = \{ (a,b) \in Z_{26} \times Z_{26} : UCLN(a,26) = 1 \}$

Với $K = (a,b) \in \mathcal{K}$, ta định nghĩa:

$$e_K(x) = ax + b \bmod 26$$

và

$$d_K(y) = a^{-1}(y-b) \bmod 26,$$

$x,y \in Z_{26}$

Ví dụ 1.3

Giả sử K = (7,3). Như đã nêu ở trên, $7^{-1} \bmod 26 = 15$. Hàm mã hoá là

$$e_K(x) = 7x+3$$

Và hàm giải mã tương ứng là:

$$d_K(x) = 15(y-3) = 15y - 19$$

Ở đây, tất cả các phép toán đều thực hiện trên Z_{26}. Ta sẽ kiểm tra liệu $d_K(e_K(x)) = x$ với mọi $x \in Z_{26}$ không?. Dùng các tính toán trên Z_{26} , ta có

$$\begin{aligned} d_K(e_K(x)) &= d_K(7x+3) \\ &= 15(7x+3)-19 \\ &= x + 45 - 19 \\ &= x. \end{aligned}$$

Để minh hoạ, ta hãy mã hoá bản rõ "*hot*". Trước tiên biến đổi các chữ h, o, t thành các thặng du theo modulo 26. Ta được các số tương ứng là 7, 14 và 19. Bây giờ sẽ mã hoá:

$$\begin{aligned} 7 \times 7 + 3 \bmod 26 &= 52 \bmod 26 = 0 \\ 7 \times 14 + 3 \bmod 26 &= 101 \bmod 26 = 23 \\ 7 \times 19 + 3 \bmod 26 &= 136 \bmod 26 = 6 \end{aligned}$$

Bởi vậy 3 ký hiệu của bản mã là 0, 23 và 6 tương ứng với xâu ký tự AXG. Việc giải mã sẽ do bạn đọc thực hiện như một bài tập.

1.1.4 Mã Vigenère

Trong cả hai hệ MDV và MTT (một khi khoá đã được chọn) mỗi ký tự sẽ được ánh xạ vào một ký tự duy nhất. Vì lý do đó, các hệ mật còn được gọi hệ thay thế đơn biểu. Bây giờ ta sẽ trình bày (trong hùnh 1.5) một hệ mật không phải là bộ chữ đơn, đó là hệ mã Vigenère nổi tiếng. Mật mã này lấy tên của Blaise de Vigenère sống vào thế kỷ XVI.

Sử dụng phép tương ứng A $\Leftrightarrow$ 0, B $\Leftrightarrow$ 1, . . . , Z $\Leftrightarrow$ 25 mô tả ở trên, ta có thể gắn cho mỗi khoa K với một chuỗi kí tự có độ dài m được gọi là từ khoá. Mật mã Vigenère sẽ mã hoá đồng thời m kí tự: Mỗi phần tử của bản rõ tương đương với m ký tự.

Xét một ví dụ nhỏ

Ví dụ 1.4

Giả sử m =6 và từ khoá là CIPHER. Từ khoá này tương ứng với dãy số K = (2,8,15,4,17). Giả sử bản rõ là xâu:

thiscryptosystemisnotsecure

Hình 1.5 Mật mã Vigenère

Cho m là một số nguyên dương cố định nào đó. Định nghĩa $\mathcal{P} = \mathcal{C} = \mathcal{K} = (Z_{26})^m$. Với khoá K = $(k_1, k_2, \ldots, k_m)$ ta xác định :

$$e_K(x_1, x_2, \ldots, x_m) = (x_1+k_1, x_2+k_2, \ldots, x_m+k_m)$$

và

$$d_K(y_1, y_2, \ldots, y_m) = (y_1-k_1, y_2-k_2, \ldots, y_m-k_m)$$

trong đó tất cả các phép toán được thực hiện trong Z_{26}

Ta sẽ biến đổi các phần tử của bản rõ thành các thặng dư theo modulo 26, viết chúng thành các nhóm 6 rồi cộng với từ khoá theo modulo 26 như sau:

19	7	8	18	2	17	24	15	19	14	18	24
2	8	15	7	4	17	2	8	15	7	4	17
21	15	23	25	6	8	0	23	8	21	22	15
18	19	4	12	8	18	13	14	19	18	4	2
2	8	15	7	4	17	2	8	15	7	4	17
20	1	19	19	12	9	15	22	8	15	8	19

20	17	4
2	8	15
22	25	19

Bởi vậy, dãy ký tự tương ứng của xâu bản mã sẽ là:

V P X Z G I A X I V W P U B T T M J P W I Z I T W Z T

Để giải mã ta có thể dùng cùng từ khoá nhưng thay cho cộng, ta trừ cho nó theo modulo 26.

Ta thấy rằng các từ khoá có thể với số độ dài m trong mật mã Vigenère là 26^m, bởi vậy, thậm chí với các giá trị m khá nhỏ, phương pháp tìm kiếm vét cạn cũng yêu cầu thời gian khá lớn. Ví dụ, nếu m = 5 thì không gian khoá cũng có kích thước lớn hơn $1{,}1 \times 10^7$. Lượng khoá này đã đủ lớn để ngaen ngừa việc tìm khoá bằng tay(chứ không phải dùng máy tính).

Trong hệ mật Vigenère có từ khoá độ dài m,mỗi ký tự có thể được ánh xạ vào trong m ký tự có thể có (giả sử rằng từ khoá chứa m ký tự phân biệt). Một hệ mật như vậy được gọi là hệ mật thay thế đa biểu (polyalphabetic). Nói chung, việc thám mã hệ thay thế đa biểu sẽ khó khăn hơn so việc thám mã hệ đơn biểu.

1.1.5 Mật mã Hill

Trong phần này sẽ mô tả một hệ mật thay thế đa biểu khác được gọi là mật mã Hill. Mật mã này do Lester S.Hill đưa ra năm 1929. Giả sử m là một số nguyên dương, đặt $\mathcal{P} = C = (Z_{26})^m$. Ý tưởng ở đây là lấy m tổ hợp tuyến tính của m ký tự trong một phần tử của bản rõ để tạo ra m ký tự ở một phần tử của bản mã.

Ví dụ nếu m = 2 ta có thể viết một phần tử của bản rõ là $x = (x_1,x_2)$ và một phần tử của bản mã là $y = (y_1,y_2)$. Ở đây, y_1cũng như y_2 đều là một tổ hợp tuyến tính của $x_1$và x_2. Chẳng hạn, có thể lấy

$$y_1 = 11x_1 + 3x_2$$
$$y_2 = 8x_1 + 7x_2$$

Tất nhiên có thể viết gọn hơn theo ký hiệu ma trận như sau

$$(y_1\ y_2) = (x_1\ x_2) \begin{pmatrix} 11 & 8 \\ 3 & 7 \end{pmatrix}$$

Nói chung, có thể lấy một ma trận K kích thước m × m làm khoá. Nếu một phần tử ở hàng i và cột j của K là $k_{i,,j}$ thì có thể viết $K = (k_{i,,j})$, với $x = (x_1, x_2, \ldots ,x_m) \in \mathcal{P}$ và $K \in \mathcal{K}$, ta tính $y = e_K(x) = (y_1, y_2, \ldots ,y_m)$ như sau:

$$(y_1,\ldots,y_m)\ (x_1, \ldots ,x_m) \begin{pmatrix} k_{1,1} & k_{1,2} & \ldots & k_{1,m} \\ k_{2,1} & k_{2,2} & \ldots & k_{2,m} \\ \ldots & \ldots & \ldots & \ldots \\ k_{m,1} & k_{m,2} & \ldots & k_{m,m} \end{pmatrix}$$

Nói một cách khác y = xK.

Chúng ta nói rằng bản mã nhận được từ bản rõ nhờ phép biến đổi tuyến tính. Ta sẽ xét xem phải thực hiện giải mã như thế nào, tức là làm thế nào để tính x từ y. Bạn đọc đã làm quen với đại số tuyến tính sẽ thấy rằng phải dùng ma trận nghịch đảo K^{-1} để giả mã. Bản mã được giải mã bằng công thức $y\ K^{-1}$.

Sau đây là một số định nghĩa về những khái niệm cần thiết lấy từ đại số tuyến tính. Nếu $A = (x_{i,j})$ là một ma trận cấp l × m và $B = (b_{1,k})$ là một ma trận cấp m × n thì tích ma trận $AB = (c_{1,k})$ được định nghĩa theo công thức:

$$c_{1,k} = \sum_{j=1}^{m} a_{i,j}\, b_{j,k}$$

Với $1 \le i \le l$ và $1 \le k \le l$. Tức là các phần tử ở hàng i và cột thứ k của AB được tạo ra bằng cách lấy hàng thứ i của A và cột thứ k của B, sau đó nhân tương ứng các phần tử với nhau và cộng lại. Cần để ý rằng AB là một ma trận cấp $l \times n$.

Theo định nghĩa này, phép nhân ma trận là kết hợp (tức (AB)C = A(BC)) nhưng noiâ chung là không giao hoán (không phải lúc nào AB = BA, thậm chí đố với ma trận vuông A và B).

Ma trận đơn vị $m \times m$ (ký hiệu là I_m) là ma trận cấp $m \times m$ có các số 1 nằm ở đường chéo chính và các số 0 ở vị trí còn lại. Như vậy ma trận đơn vị 2×2 là:

$$I_2 = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

I_m được gọi là ma trận đơn vị vì $AI_m = A$ với mọi ma trận cấp $l \times m$ và $I_mB = B$ với mọi ma trận cấp $m \times n$. Ma trận nghịch đảo của ma trận A cấp $m \times m$ (nếu tồn tại) là ma trận A^{-1} sao cho $AA^{-1} = A^{-1}A = I_m$. Không phải mọi ma trận đều có nghịch đảo, nhưng nếu tồn tại thì nó duy nhất.

Với các định nghĩa trên, có thể dễ dàng xây dựng công thức giải mã đã nêu: Vì y = xK, ta có thể nhân cả hai vế của đẳng thức với K^{-1} và nhận được:

$$yK^{-1} = (xK)K^{-1} = x(KK^{-1}) = xI_m = x$$

(Chú ý sử dụng tính chất kết hợp)

Có thể thấy rằng, ma trận mã hoá ở trên có nghịch đảo trong Z_{26}:

$$\begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix}^{-1} = \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix}$$

vì

$$\begin{bmatrix} 12 & 8 \\ 3 & 7 \end{bmatrix} \begin{bmatrix} 8 & 18 \\ 23 & 11 \end{bmatrix} = \begin{bmatrix} 11\times7+8\times23 & 11\times18+8\times11 \\ 3\times7+7\times23 & 3\times18+7\times11 \end{bmatrix}$$

$$= \begin{bmatrix} 261 & 286 \\ 182 & 131 \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$$

(Hãy nhớ rằng mọi phép toán số học đều được thực hiện theo modulo 26).

Sau đây là một ví dụ minh hoạ cho việc mã hoá và iải mã trong hệ mật mã Hill.

Via dụ 1.5

Giả sử khoá $K = \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix}$

Từ các tính toán trên ta có:

$$K^{-1} = \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix}$$

Giả sử cần mã hoá bản rõ "July". Ta có hai phần tử của bản rõ để mã hoá: (9,20) (ứng với Ju) và (11,24) (ứng với ly). Ta tính như sau:

$$(9,20) \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix} = (99+60, 72+140) = (3,4)$$

và

$$(11,24) \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix} = (121+72, 88+168) = (11,22)$$

Bởi vậy bản mã của July là DELW. Để giải mã Bob sẽ tính

$$(3,4) \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix} = (9,20)$$

và

$$(11,22 \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix} = (11,24)$$

Như vậy Bob đã nhận được bản đúng.

Cho tới lúc này ta đã chỉ ra rằng có thể thực hiện phép giải mã nếu K có một nghịch đảo. Trên thực tế, để phép giải mã là có thể thực hiện được, điều kiện cần là K phải có nghịch đảo. (Điều này dễ dàng rút ra từ đại số tuyến tính sơ cấp, tuy nhiên sẽ không chứng minh ở đây). Bởi vậy, chúng ta chỉ quan tâm tới các ma trận K khả nghịch.

Tính khả nghịch của một ma trận vuông phụ thuộc vào giá trị định thức của nó. Để tránh sự tổng quát hoá không cần thiết, ta chỉ giới hạn trong trường hợp 2×2.

Định nghĩa 1.5

Định thức của ma trận $A = (a_{i,j})$ *cấp* 2×2 *là giá trị*

$$det\ A = a_{1,1}\, a_{2,2} - a_{1,2}\, a_{2,1}$$

Nhận xét: Định thức của một ma trận vuông cấp mm có thể được tính theo các phép toán hằng sơ cấp: hãy xem một giáo trình bất kỳ về đại số tuyến tính.

Hai tính chất quan trọng của định thức là $\det I_m = 1$ và quy tắc nhân $\det(AB) = \det A \times \det B$.

Một ma trận thức K là có nghịch đảo khi và chỉ khi định thức của nó khác 0. Tuy nhiên, điều quan trọng cần nhớ là ta đang làm việc trên Z_{26} . Kết quả tương ứng là ma trận K có nghịch đảo theo modulo 26 khi và chỉ khi UCLN(det K,26) = 1.

Sau đây sẽ chứng minh ngắn gọn kết quả này.

Trước tiên, giả sử rằng UCLN(det K,26) = 1. Khi đó det K có nghịch đảo trong Z_{26} . Với $1 \le i \le m$, $1 \le j \le m$, định nghĩa $K_{i\,j}$ ma trận thu được từ K bằng cách loại bỏ hàng thứ i và cột thứ j. Và định nghĩa ma trận K^* có phần tử (i,j) của nó nhận giá trị(-1) $\det K_{j\,i}$ (K^* được gọi là ma trận bù đại số của K). Khi đó có thể chứng tỏ rằng:

$$K^{-1} = (\det K)^{-1} K^* .$$

Bởi vậy K là khả nghịch.

Ngược lại K có nghịch đảo K^{-1} . Theo quy tắc nhân của định thức

$$1 = \det I = \det (KK^{-1}) = \det K \det K^{-1}$$

Bởi vậy det K có nghịch đảo trong Z_{26} .

Nhận xét: Công thức đối với ở trên không phải là một công thức tính toán có hiệu quả trừ các trường hợp m nhỏ (chẳng hạn m = 2, 3). Vớim lớn, phương pháp thích hợp để tính các ma trận nghịch đảo phải dựa vào các phép toán hằng sơ cấp.

Trong trường hợp 2×2, ta có công thức sau:

Định lý 1.3

Giả sử $A = (a_{i\ j})$ là một ma trận cấp 2×2 trên Z_{26} sao cho $\det A = a_{1,1}a_{2,2} - a_{1,2}\ a_{2,1}$ có nghịch đảo. Khi đó

$$A^{-1} = (\det A)^{-1} \begin{bmatrix} a_{2,2} & -a_{1,2} \\ -a_{2,1} & a_{1,1} \end{bmatrix}$$

Trở lại ví dụ đã xét ở trên . Trước hết ta có:

$$\det \begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix} = 11 \times 7 - 8 \times 3 \bmod 2$$
$$= 77 - 24 \bmod 26 = 53 \bmod 26$$
$$= 1$$

Vì $1^{-1} \mod 26 = 1$ nên ma trận nghịch đảo là

$$\begin{bmatrix} 11 & 8 \\ 3 & 7 \end{bmatrix}^{-1} = \begin{bmatrix} 7 & 18 \\ 23 & 11 \end{bmatrix}$$

Đây chính là ma trận đã có ở trên.

Bây giờ ta sẽ mô tả chính xác mật mã Hill trên Z_{26} (hình 1.6)

Hình 1.6 Mật mã HILL

Cho m là một số nguyên dương có định. Cho $\mathcal{P} = \mathcal{C} = (Z_{26})^m$ và cho

$\mathcal{K}$ = { các ma trận khả nghịch cấp m × m trên Z_{26}}

Với một khoá $K \in \mathcal{K}$ ta xác định

$$e_K(x) = xK$$

và $$d_K(y) = yK^{-1}$$

Tất cả các phép toán được thực hiện trong Z_{26}

1.1.5 Mã hoán vị (MHV)

Tất cả các hệ mật thảo luận ở trên ít nhiều đều xoay quanh phép thaythế: các ký tự của bản rõ được thay thế bằng các ký tự khác trongbản mã. Ý tưởng của MHV là giữ các ký tự của bản rõ không thay đổi nhưng sẽ thay đôỉi vị trí của chúng bằng cách sắp xếp lại các ký tự này. MHV (còn được gọi là mã chuyển vị) đã được dùng từ hàng trăm năm nay. Thật ra thì sự phân biệt giữa MHV và MTT đã được Giovani Porta chỉ ra từ 1563. Định nghĩa hình thức cho MHV được nêu ra trên hình 1.7.

Không giống như MTT, ở đây không có các phép toán đại số nào cần thực hiện khi mã hoá và giải mã nên thích hợp hơn cả là dùng các ký tự mà không dùng các thặng dư theo modulo 26. Dưới đây là một ví dụ minh hoạ

Ví dụ 1.6

Giả sử m = 6 và khoá là phép hoán vị (π) sau:

1	2	3	4	5	6
3	5	1	6	4	2

Hình 1.7 Mã hoán vị

Cho m là mộ số nguyên dương xác định nào đó. Cho $\mathcal{P} = \mathcal{C} = (Z_{26})^m$ và cho $\mathcal{K}$ gồm tất cả các hoán vị của {1, . . ., m}. Đối một khoá π (tức là một hoán vị) ta xác định

$$e_\pi(x_1, \ldots, x_m) = (x_{\pi(1)}, \ldots, x_{\pi(m)})$$

và

$$d_\pi(x_1, \ldots, x_m) = (y_{\pi^{-1}(1)}, \ldots, y_{\pi^{-1}(m)})$$

trong đó π^{-1} là hoán vị ngược của π

Khi đó phép hoán vị ngược π^{-1} sẽ là:

1	2	3	4`	5	6
3	6	1	5	2	4

Bây giờ giả sử có bản rõ

Shesellsseashellsbytheseashore

Trước tiên ta nhóm bản rõ thành các nhóm 6 ký tự:

shesel | lsseas | hellsb | ythese | ashore

Bây giờ mỗi nhóm 6 chữ cái được sắp xếp lại theo phép hoán vị π, ta có:

EESLSH | SALSES | LSHBLE | HSYEET | HRAEOS

Như vậy bản mã là

EESLSH SALSES LSHBLE HSYEET HRAEOS

Như vậy bản mã đã được mã theo cách tương tự banừg phép hoán vị đảo π^{-1}.

Thực tế mã hoán vị là trường hợp đặc biệt của mật mã Hill. Khi cho phép hoán vị π của tập {1, . . . ,m}, ta có thể xác định một ma trận hoán vị m × m thích hợp $K_\pi = \{ k_{i,j}\}$ theo công thức:

$$k_{i,j} = \begin{cases} 1 & \text{nếu } j = \pi(i) \\ 0 & \text{với các trường hợp còn lại} \end{cases}$$

(ma trận hoán vị là ma trận trong đó mỗi hàng và mỗi cột chỉ có một số "1", còn tất cả các giá trị khác đều là số "0". Ta có thể thu được một ma trận hoán vị từ ma trận đơn vị bằng cách hoán vị các hàng hoặc cột).

Dễ dàng thấy rằng, phép mã Hill dùng ma trận K_π trên thực tế tương đương với phép mã hoán vị dùng hoán vị π. Hơn nữa $K^{-1}{}_\pi = K_\pi{}^{-1}$ tức ma trận nghịch đảo của K_π là ma trận hoán vị xác định theo hoán vị π^{-1}. Như vậy, phép giải mã Hill tương đương với phép giải mã hoán vị.

Đối với hoán vị π được dung trong ví dun trên, các ma trận hoán vị kết hợp là:

$$K_\pi = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 1 \end{pmatrix} \quad \text{và } K^{-1}{}_\pi = \begin{pmatrix} 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix}$$

Bạn đọc có thể kiểm tra để thấy rằng, tích của hai ma trạn này là một ma trận đơn vị.

1.1.7 Các hệ mã dòng

Trong các hệ mật nghiên cứu ở trên, cácb phần tử liên tiếp của bản rõ đều được mã hoá bằng cùng một khoá K. Tức xâu bản mã y nhạn được có dạng:

$$y = y_1y_2 \ldots = e_K(x_1)\, e_K(x_2) \ \ldots$$

Các hệ mật thuộc dạng này thường được gọi là các mã khối. Một quan điểm sử dụng khác là mật mã dòng. Ý tưởng cơ bản ở đây là tạo ra một dòng khoá $z = z_1z_2 \ldots$ và dùng nó để mã hoá một xâu bản rõ $x = x_1x_2 \ldots$ theo quy tắc:

$$y = y_1y_2 \ldots = e_{z1}(x_1)\, e_{z2}(x_1) \ldots$$

Mã dòng hoạt động như sau. Giả sử $K \in \mathcal{K}$ là khoá và $x = x_1x_2 \ldots$ là xâu bản rõ. Hàm f_i được dùng để tạo z_i (z_i là phần tử thứ i của dòng khoá) trong đó f_i là một hàm của khoá K và i-1 là ký tự đầu tiên của bản rõ:

$$z_i = f_i\,(K, x_1, \ldots, x_{i-1})$$

Phần tử z_i của dòng khoá được dùng để mã x_i tạo ra $y_i = e_{iz}(x_i)$. Bởi vậy, để mã hoá xâu bản rõ $x_1\, x_2 \ldots$ ta phải tính liên tiếp: z_1, y_1, z_2, y_2 ...

Việc giải mã xâu bản mã $y_1y_2 \ldots$ có thể được thực hiện bằng cách tính liên tiếp: z_1, x_1, z_2, x_2 ...

Sau đây làb định nghĩa dưới dạng toán học:

Định nghĩa 1.6.

Mật mã dòng là một bộ $(\mathcal{P},\mathcal{C},\mathcal{K},\mathcal{L},\mathcal{F},\mathcal{E},\mathcal{D})$ thoả mãn dược các điều kiện sau:

1. *$\mathcal{P}$ là một tập hữu hạn các bản rõ có thể.*
2. *$\mathcal{C}$ là tập hữu hạn các bản mã có thể.*
3. *$\mathcal{K}$ là tập hữu hạn các khoá có thể (không gian khoá)*
4. *$\mathcal{L}$ là tập hữu hạn các bộ chữ của dòng khoá.*
5. *$\mathcal{F} = (f_1 f_2 \ldots)$ là bộ tạo dòng khoá. Với $i \geq 1$*

$$f_i : \mathcal{K} \times \mathcal{P}^{i-1} \rightarrow \mathcal{L}$$

6. *Với mỗi $z \in \mathcal{L}$ có một quy tắc mã $e_z \in \mathcal{E}$ và một quy tắc giải mã tương ứng $d_z \in \mathcal{D}$. $e_z : \mathcal{P} \rightarrow \mathcal{C}$ và $d_z : \mathcal{C} \rightarrow \mathcal{P}$ là các hàm thoả mãn $d_z(e_z(x)) = x$ với mọi bản rõ $x \in \mathcal{P}$.*

Ta có thể coi mã khối là một trường hợp đặc biệt của mã dòng trong đó dùng khoá không đổi: $Z_i = K$ với mọi $i \geq 1$.

Sau đây là một số dạng đặc biệt của mã dòng cùng với các ví dụ minh hoạ. Mã dòng được gọi là đồng bộ nếu dòng khoá không phụ thuộc vào xâu bản rõ, tức là nếu dòng khoá đực tạo ra chỉ là hàm của khoá K. Khi đó ta coi K là một "mần" để mở rộng thành dòng khoá $z_1 z_2 \ldots$

Một hệ mã dòng được gọi là tuần hoàn với chu kỳ d nếu $z_{i+d} = z_i$ với số nguyên $i \geq 1$. Mã Vigenère với độ dài từ khoá m có thể coi là mã dòng tuần hoàn với chu kỳ m. Trong trường hợp này, khoá là $K = (k_1, \ldots k_m)$. Bản thân K sẽ tạo m phần tử đầu tiên của dòng khoá: $z_i = k_i$, $1 \leq i \leq m$. Sau đó dòng khoá sẽ tự lặp lại. Nhận thấy rằng, trong mã dòng tương ứng với mật mã Vigenère, các hàm mã và giải mã được dùng giống như các hàm mã và giải mã được dùng trong MDV:

$$e_z(x) = x+z \text{ và } d_z(y) = y-z$$

Các mã dòng thường được mô tả trong các bộ chữ nhi phân tức là $\mathcal{P}= \mathcal{C}=\mathcal{L}= Z_2$. Trong trường hợp này, các phép toán mã và giải mã là phép cộng theo modulo 2.

$$e_z(x) = x +z \bmod 2 \text{ và } d_z(x) = y +z \bmod 2.$$

Nếu ta coi "0" biểu thị giá trị "sai" và "1" biểu thị giá trị "đúng" trong đại số Boolean thì phép cộng theo moulo 2 sẽ ứng với phép hoặc có loại trừ. Bởi vậy phép mã (và giải mã) dễ dàng thực hiện bằng mạch cứng.

Ta xem xét một phương pháp tạo một dòng khoá (đồng bộ) khác. Giả sử bắt đầu với $(k_1, \ldots, k_m)$ và $z_i = k_i$, $1 \leq i \leq m$ (cũng giống như trước đây), tuy nhiên bây giờ ta tạo dòng khoá theo một quan hệ đệ quy tuyến tính cấp m:

$$z_{i+m} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2$$

trong đó $c_0, \ldots, c_{m-1} \in Z_2$ là các hằng số cho trước.

Nhận xét:

Phép đệ quy được nói là có bậc m vì mỗi số hạng phụ thuộc vào m số hạng đứng trước. Phép đệ quy này là tuyến tính bởi vì Z_{i+m} là một hàm tuyến tính của các số hạng đứng trước. Chú ý ta có thể lấy $c_0 = 1$ mà không làm mất tính tổng quát. Trong trường hợp ngược lại phép đệ quy sẽ là có bậc m-1.

Ở đây khoá K gồm 2m giá trị $k_1, \ldots, k_m, c_0, \ldots, c_{m-1}$. Nếu $(k_1, \ldots, k_m)= (0, \ldots, 0)$ thì dòng khoá sẽ chứa toàn các số 0. Dĩ nhiên phải tránh điều này vì khi đó bản mã sẽ đồng nhất với bản rõ. Tuy nhiên nếu chọn thích hợp các hằng số $c_0, \ldots, c_{m-1}$ thì một véc tơ khởi đầu bất kì khác $(k_1, \ldots, k_m)$ sẽ tạo nên một dòng khoá có chu kỳ 2^m -1. Bởi vậy một khoá ngắn sẽ tạo nên một dòng khoá có chu kỳ rất lớn. Đây là một tính chất rất đáng lưu tâm vì ta sẽ thấy ở phần sau, mật mã Vigenère có thể bị thám nhờ tận dụng yếu tố dòng khoá có chu kỳ ngắn.

Sau đây là một ví dụ minh hoạ:

Ví dụ 1.7

Giả sử m = 4 và dòng khoá được tạo bằng quy tắc:

$$z_{i+4} = z_i + z_{i+1} \bmod 2$$

Nếu dòng khoá bắt đầu một véc tơ bất kỳ khác với véc tơ (0,0,0,0) thì ta thu được dòng khoá có chu kỳ 15. Ví dụ bắt đầu bằng véc tơ (1,0,0,0), dòng khoá sẽ là:

1, 0, 0, 0, 1, 0, 0, 1, 1, 0, 1, 0, 1, 1, 1

Một véc tơ khởi đầu khác không bất kỳ khác sẽ tạo một hoán vị vòng (cyclic) của cùng dòng khoá.

Một hướng đáng quan tâm khác của phương pháp tạo dòng khoá hiệu quả bằng phần cứng là sử dụng bộ ghi dịch hồi tiếp tuyến tính (hay LFSR). Ta dùng một bộ ghi dịch có m tầng. Véc tơ $(k_1, \ldots, k_m)$ sẽ được dùng để khởi tạo (đặt các giá trị ban đầu) cho thanh ghi dịch. Ở mỗi đơn vị thời gian, các phép toán sau sẽ được thực hiện đồng thời.

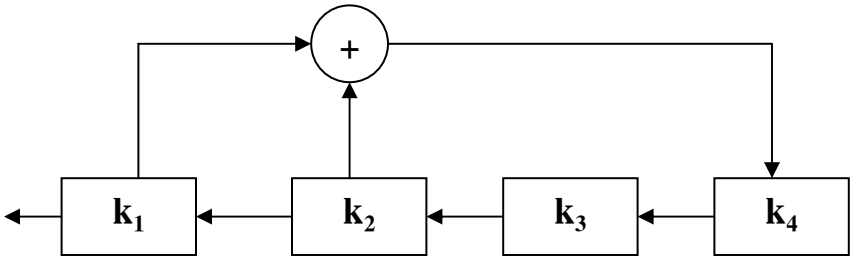
1. k_1 được tính ra dùng làm bit tiếp theo của dòng khoá.
2. $k_2, \ldots, k_m$ sẽ được dịch một tầng về phía trái.
3. Giá trị mới của sẽ được tính bằng:

$$\sum_{j=0}^{m-1} c_j k_{j+1}$$

(đây là hồi tiếp tuyến tính)

Ta thấy rằng thao tác tuyến tính sẽ được tiến hành bằng cách lấy tín hiệu ra từ một số tầng nhất định của thanh ghi (được xác định bởi các hằng số c_j có giá trị "1") và tính tổng theo modulo 2 (là phép hoặc loại trừ). Hình 1.8 cho mô tả của LFSR dùng để tạo dòng khoá cho ví dụ 1.7.

Hình 1.8 Thanh ghi dịch hồi tiếp tuyến tính (LFSR)



Một ví dụ về mã dòng không đồng bộ là mã khoá tự sinh được cho ở hình 1.9. Hình như mật mã này do Vigenère đề xuất.

Hình 1.9. Mật mã khoá tự sinh

Cho $\mathcal{P} = C = \mathcal{K} = \mathcal{L} = Z_{26}$
Cho $z_1 = K$ và $z_i = x_{i-1}$ $(i \geq 2)$
Với $0 \leq z \leq 25$ ta xác định
$e_z(x) = x + z \bmod 26$
$d_z(y) = y - z \bmod 26$
$(x,y \in Z_{26})$

Lý do sử dụng thuật nghữ "khoá tự sinh" là ở chỗ: bản rõ được dùng làm khoá (ngoài "khoá khởi thuỷ" ban đầu K).

Sau đây là một ví dụ minh hoạ

Ví dụ 1.8:

Giả sử khoá là k = 8 và bản rõ là *rendezvous*. Trước tiên ta biến đổi bản rõ thành dãy các số nguyên:

17 4 13 3 4 25 21 14 20 18

Dòng khoá như sau:

8 17 4 13 3 4 25 21 14 20

Bây giờ ta cộng các phần tử tương ứng rồi rút gọn theo modulo 26:

25 21 17 16 7 3 20 9 8 12

Bản mã ở dạng ký tự là: ZVRQHDUJIM
Bây giờ ta xem Alice giải mã bản mã này như thế nào. Trước tiên Alice biến đổi xâu kí tự thành dãy số:

25 21 17 16 7 3 20 9 8 12

Sau đó cô ta tính:

$$x_1 = d_8(25) = 25 - 8 \bmod 26 = 17$$

và $$x_2 = d_{17}(21) = 21 - 17 \bmod 26 = 4$$

và cứ tiếp tục như vậy. Mỗi khi Alice nhận được một ký tự của bản rõ, cô ta sẽ dùng nó làm phần tử tiếp theo của dòng khoá.

Dĩ nhiên là mã dùng khoá tự sinh là không an toàn do chỉ có 26 khoá.
Trong phần sau sẽ thảo luận các phương pháp thám các hệ mật mã mà ta đã trình bày.

1.2 MÃ THÁM CÁC HỆ MÃ CỔ ĐIỂN

Trong phần này ta sẽ bàn tới một vài kỹ thuật mã thám. Giả thiết chung ở đây là luôn coi đối phương Oscar đã biết hệ mật đang dùng. Giả thiết này được gọi là nguyên lý Kerekhoff. Dĩ nhiên, nếu Oscar không biết hệ mật được dùng thì nhiệm vụ của anh ta sẽ khó khăn hơn. Tuy nhiên ta không muốn độ mật của một hệ mật lại dựa trên một giả thiết không chắc chắn là Oscar không biết hệ mật được sử dụng. Do đó, mục tiêu trong thiết kế một hệ mật là phải đạt được độ mật dưới giả thiết Kerekhoff.

Trước tiên ta phân biệt các mức độ tấn công khác nhau vào các hệ mật. Sau đây là một số loại thông dụng nhất.

Chỉ có bản mã:

Thám mã chỉ có xâu bản mã y.

Bản rõ đã biết:

Thám mã có xâu bản rõ x và xâu bản mã tương ứng y.

Bản rõ được lựa chọn:

Thám mã đã nhận được quyền truy nhập tạm thời vào cơ chế mã hoá. Bởi vậy, thám mã có thể chọn một xâu bản rõ x và tạo nên xâu bản mã y tương ứng.

Bản mã được lựa chọn:

Thám mã có được quyền truy nhập tạm thời vào cơ chế giải mã. Bởi vậy thám mã có thể chọn một bản mã y và tạo nên xâu bản rõ x tương ứng.

Trong mỗi trường hợp trên, đối tượng cần phải xác định chính là khoá đã sử dụng. Rõ ràng là 4 mức tấn công trên đã được liệt kê theo độ tăng của sức mạnh tấn công. Nhận thấy rằng, tấn công theo bản mã được lựa chọn là thích hợp với các hệ mật khoá công khai mà ta sẽ nói tới ở chương sau.

Trước tiên, ta sẽ xem xét cách tấn công yếu nhất, đó là tấn công chỉ có bản mã. Giả sử rằng, xâu bản rõ là một văn bản tiếng Anh thông thường không có chấm câu hoặc khoảng trống (mã thám sẽ khó khăn hơn nếu mã cả dấu chấm câu và khoảng trống).

Có nhiều kỹ thuật thám mã sử dụng các tính chất thống kê của ngôn ngữ tiếng Anh. Nhiều tác giả đã ước lượng tần số tương đối của 26 chữ cái theo các tính toán thống kê từ nhiều tiểu thuyết, tạp chí và báo. Các ước lượng trong bảng 1.1 lấy theo tài liệu của Beker và Piper.

Bảng 1.1 Xác suất xuất hiện của 26 chữ cái:

Kí tự	Xác suất	Kí tự	Xác suất	Kí tự	Xác suất
A	.082	J	.002	S	.063
B	.015	K	.008	T	.091
C	.028	L	.040	U	.028
D	.043	M	.024	V	.010
E	.0127	N	.067	W	.023
F	.022	O	.075	X	.001
G	.020	P	.019	Y	.020
H	.061	Q	.001	Z	.001
I	.070	R	.060		

Từ bảng trên, Beker và Piper phân 26 chữ cái thành 5 nhóm như sau:

1. E: có xác suất khoảng 1,120
2. T, A, O, I, N, S, H, R : mỗi ký tự có xac suất khoảng 0,06 đến 0,09
3. D, L : mỗi ký tự có xác suất chừng 0,04
4. C, U, M, W, F, G, Y, P, B: mỗi ký tự có xác suất khoảng 0,015 đến 0,023
5. V, K, J, X, Q, Z mỗi ký tự có xác suất nhỏ hơn 0,01

Việc xem xét các dãy gồm 2 hoặc 3 ký tự liên tiếp (được gọi là bộ đôi - diagrams và bộ ba - Trigrams)cũng rất hữu ích. 30 bộ đôi thông dụng nhất (theo hứ tự giảm dần) là: TH, HE, IN, ER, AN, RE, ED, ON, ES, ST, EN, AT, TO, NT, HA, ND, OU, EA, NG, AS, OR, TI, IS, ET, IT, AR, TE, SE, HI và OF. 12 bộ ba thông dụng nhất (theo thứ tự giảm dần) là: THE, ING, AND, HER, ERE, ENT, THA, NTH, WAS, ETH, FOR và DTH.

1.2.1 Thám hệ mã Affine

Mật mã Affine là một ví dụ đơn giản cho ta thấy cách thám hệ mã nhờ dùng các số liệu thống kê. Giả sử Oscar đã thu trộm được bản mã sau:

Bảng 1.2: Tần suất xuất hiện của 26 chữ cái của bản mã

Kí tự	Tần suất	Kí tự	Tần suất	Kí tự	Tần suất	Kí tự	Tần suất
A	2	H	5	O	1	U	2
B	1	I	0	P	3	V	4
C	0	J	0	Q	0	W	0
D	6	K	5	R	8	X	2
E	5	L	2	S	3	Y	1
F	4	M	2	T	0	Z	0
G	0	N	1				

Ví Dụ 1.9:

Bản mã nhận được từ mã Affine:

FMXVEDRAPHFERBNDKRXRSREFMORUDSDKDVSHVUFEDKPKDLYEVLRHHRH

Phân tích tần suất của bản mã này được cho ở bảng 1.2

Bản mã chỉ có 57 ký tự. Tuy nhiên độ dài này cũng đủ phân tích thám mã đối với hệ Affine. Các ký tự có tần suất cao nhất trong bản mã là: R (8 lần xuất hiện), D (6 lần xuất hiện), E, H, K (mỗi ký tự 5 lần) và F, S, V (mỗi ký tự 4 lần).

Trong phỏng đoán ban đầu, ta giả thiết rằng R là ký tự mã của chữ e và D là kí tự mã của t, vì e và t tương ứng là 2 chữ cái thông dụng nhất. Biểu thị bằng số ta có: $e_K(4) = 17$ và $e_K(19) = 3$. Nhớ lại rằng $e_K(x) = ax + b$ trong đó a và b là các số chưa biết. Bởi vậy ta có hai phương trình tuyến tính hai ẩn:

$$4a + b = 17$$
$$19a + b = 3$$

Hệ này có duy nhất nghiệm a = 6 và b = 19 (trong Z_{26}). Tuy nhiên đây là một khoá không hợp lệ do UCLN(a,26) = 2 1. Bởi vậy giả thiết của ta là không đúng.

Phỏng đoán tiếp theo của ta là: R là ký tự mã của e và E là mã của t. Thực hiện như trên, ta thu được a =13 và đây cũng là một khoá không hợp lệ. Bởi vậy ta phải thử một lần nữa: ta coi rằng R là mã hoá của e và H là mã hoá của t. Điều này dẫn tới a = 8 và đây cũng là một khoá không hợp lệ. Tiếp tục, giả sử rằng R là mã hoá của e và K là mã hoá của t. Theo giả thiết này ta thu được a = 3 và b = 5 là khóa hợp lệ.

Ta sẽ tính toán hàm giải mã ứng với K = (3,5) và gải mã bản mã để xem liệu có nhận được xâu tiếng Anh có nghĩa hay không. Điều này sẽ khẳng định tính hợp lệ của khoá (3,5). Sau khi thực hiện các phép toán này, ta có $d_K(y) = 9y - 19$ và giải mã bản mã đã cho, ta được:

algorithmsarequitegeneraldefinitionsof
arithmeticprocesses

Như vậy khoá xác định trên là khoá đúng.

1.2.2. Thám hệ mã thay thế

Sau đâyta phân tích một tình huống phức tạp hơn, đó là thay thế bản mã sau

Ví dụ 1.10

Bản mã nhận được từ MTT là:

YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ
NZUCDRJXỷYMTMEYIFZWDYVZVYFZUMRZCRWNZDZJT
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDINZDIR

Phân tích tần suất của bản mã này đươch cho ở bảng 1.3.

Bảng 1.3. Tần suất xuất hiện của 26 chưz cái trong bản mã.

Ký tự	Tần suất	Ký tự	Tần suất	Ký tự	Tần suất	Ký tự	Tần suất
A	0	H	4	O	0	U	5
B	1	I	5	P	1	V	5
C	15	J	11	Q	4	W	8
D	13	K	1	R	10	X	6
E	7	L	0	S	3	Y	10
F	11	M	16	T	2	Z	20
G	1	N	9				

Do Z xuất hiện nhiều hơn nhiều so với bất kỳ một ký tự nào khác trong bản mã nên có thể phỏng đoán rằng, $d_Z(Z) = e$. các ký tự còn lại xuất hiện ít nhất 10 lần (mỗi ký tự) là C, D, F, J, R, M, Y. Ta hy vọng rằng, các ký tự này là mã khoá của (một tập con trong) t, a, c, o, i, n, s, h, r, tuy nhiên sự khác biệt về tần suất không đủ cho ta có được sự phỏng đoán thích hợp.

Tới lúc này ta phải xem xét các bộ đôi, đặc biệt là các bộ đôi có dạng -Z hoặc Z- do ta đã giả sử rằng Z sẽ giải mã thành e. Nhận thấy rằng các bộ đôi thường gặp nhất ở dạng này là DZ và ZW (4 lần mỗi bộ); NZ và ZU (3 lần mỗi bộ); và RZ, HZ, XZ, FZ, ZR, ZV, ZC, ZD và ZJ (2 lần mỗi bộ). Vì ZW xuất hiện 4 lần còn WZ không xuất hiện lần nào và nói chung W xuất hiện ít hơn so với nhiều ký tự khác, nên ta có thể phỏng đoán là $d_K(W) = d$. Vì DZ xuất hiện 4 lần và ZD xuất hiện 2 lần nên ta có thể nghĩ rằng $d_K(D) \in \{r,s,t\}$, tuy nhiên vẫn còn chưa rõ là ký tự nào trong 3 ký tự này là ký tự đúng.

Nêu tiến hành theo giả thiết $d_K(Z) = e$ và $d_K(W) = d$ thì ta phải nhìn trở lại bản mã và thấy rằng cả hai bộ ba ZRW và RZW xuất hiện ở gần đầu của bản mã và RW xuất hiện lại sau đó vì R thường xuất hiện trong bản mã và nd là một bộ đôi thường gặp nên ta nên thử $d_K(R) = n$ xem là một khả năng thích hợp nhất.

Tới lúc này ta có:

```
- - - - - - end - - - - - - - - - e - - - - ned- - - e - - - - - - - - -
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ
- - - - - - - - e- - - - e - - - - - - - - n - - d - - - en - - - - e - - - -e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ
- e - - - n - - - - - n - - - - - - ed - - - e - - - - - - ne - nd- e- e - -
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ
- ed - - - - - n - - - - - - - - - - e - - - ed - - - - - - - d - - - e - - n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Bước tiếp theo là thử $d_K(N) = h$ vì NZ là một bộ đôi thường gặp còn ZN không xuất hiện. Nếu điều này đúng thì đoạn sau của bản rõ ne - ndhe sẽ gợi ý rằng $d_K(C) = a$. Kết hợp các giả định này, ta có:

```
- - - - - -end- - - - - a- - -e -a - - nedh- -e- - - - - -a - - - - -
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ
h -  - - - - - - a- - - e - a- - - a - - - nhad - a - -en -a -  e - h- -e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ
he - a - n- - - - - - n - - - - - - ed - - - e- - - e - - neandhe  -e - -
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ
- ed  - a - - -nh - - - ha - - - a- e - - - - ed - - - - -a -d - - he- -n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Bây giờ ta xét tới M là ký tự thường gặp nhất sau Z. Đoạn bản mã RNM mà ta tin là sẽ giải mã thành nh- gợi ý rằng h- sẽ bắt đầu một từ, bởi vậy chắc là M sẽ biểu thị môt nguyên âm. Ta đã sử dụng a và e, bởi vậy, phỏng đoán rằng $d_K(M) = i$ hoặc o. Vì ai là bộ đôi thường gặp hơn ao nên bộ đôi CM trong bản mã gợi ý rằng, trước tiên nên thử $d_K(M) = i$. Khi đó ta có:

```
- - - - -iend- - - - - a -i - e -a -inedhi - e- - - - - -a - - -i  -
YIFQFMZRWQFYVECFMDZPCVMRZWNMDZVEJBTXCDDUMJ
h - - - - - i - ea - i - e  -a - - -a  - i -nhad -a - en - -a - e -hi -e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREXCHZUNMXZ
he  - a - n - - - - -in -i - - - -  ed - - -e - - - e - ineandhe - e - -
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ
- ed - a - - inhi - - hai - - a - e - i- -ed- - - - - a - d - - he - -n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR
```

Tiếp theo thử xác định xem chữ nào được mã hoá thành o. Vì o là một chữ thường gặp nên giả định rằng chữ cái tương ứng trong bản mã là một trong các ký tự D,F,J,Y. Y có vẻ thích hợp nhất, nếu không ta sẽ có các xâu dài các nguyên âm, chủ yếu là aoi (từ CFM hoặc CJM). Bởi vậy giả thiết rằng $d_K(Y) = o$.

Ba ký tự thường gặp nhất còn lại trong bản mã là D,F,J, ta phán đoán sẽ giải mã thành r,s,t theo thứ tự nào đó. Hai lần xuất hiện của bộ ba NMD gợi ý rằng $d_K(D) = s$ ứng với bộ ba his trong bản rõ (điều này phù hợp với giả định trước kia là $d_K(D) \in \{r,s,t\}$). Đoạn HNCMF có thể là bản mã của chair, điều này sẽ cho $d_K(F) = r$ (và $d_K(H) = c$) và bởi vậy (bằng cách loại trừ) sẽ có $d_K(J) = t$.
Ta có:

o- r - riend - ro - - arise - a - inedhise - - t - - - ass - it
YIFQFMZRWQFYVECFMDZPCVMRZNMDZVEJBTXCDDUMJ
hs - r - riseasi - e - a - orationhadta - - en - -ace - hi - e
NDIFEFMDZCDMQZKCEYFCJMYRNCWJCSZREZCHZUNMXZ
he - asnt - oo - in - i - o - redso - e - ore - ineandhesett
NZUCDRJXYYSMRTMEYIFZWDYVZVYFZUMRZCRWNZDZJJ
- ed - ac - inhischair - aceti - ted - - to - ardsthes - n
XZWGCHSMRNMDHNCMFQCHZJMXJZWIEJYUCFWDJNZDIR

Bây giờ việc xác định bản rõ và khoá cho ví dụ 1.10 không còn gì khó khăn nữa. Bản rõ hoàn chỉnh như sau:

Our friend from Pais examined his empty glass with surprise, as if evaporation had taen place while he wasn't looking. I poured some more wine and he settled back in his chair, face tilted up towards the sun.

1.2.3. Thám hệ mã Vigenère

Trong phần này chúng ta sẽ mô tả một số phương pháp thám hệ mã Vigenère. Bước đầu tiên là phải xác định độ dài từ khoá mà ta ký hiệu là m. Ở đây dùng hai kỹ thuật. Kỹ thuật thứ nhất là phép thử Kasiski và kỹ thuật thứ hai sử dụng chỉ số trùng hợp.

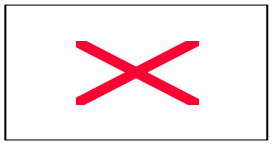
Phép thử Kasiski lần đầu tiên được Kasiski Friendrich mô tả vào năm 1863. Kỹ thuật này được xây dựng trên nhận xét là: hai đoạn giống nhau của bản rõ sẽ được mã hoá thành cùng một bản mã khi chúng xuất hiện trong bản rõ cách nhau x vị trí, trong đó $x \equiv o \bmod m$. Ngược lại, nếu ta thấy hai đoạn giống nhau của bản mã (mỗi đoạn có độ dài ít nhất là 3) thì đó là một dấu hiệu tốt để nói rằng chúng tương ứng với các đoạn bản rõ giống nhau.

Phép thử Kasiski như sau. Ta tòm trong bản mã các cặp gồm các đoạn như nhau có độ dài tối thiểu là 3 và ghi lại khoảng cách giữa các vị trí bắt đầu của hai đoạn. Nếu thu được một vài giá trị d_1, d_2 ,. . . thì có thể hy vọng rằng m sẽ chia hết cho ước chung lớn nhất của các d_i.

Việc xác minh tiếp cho giá trị của m có thể nhận được bằng chỉ số trùng hợp. Khái niệm này đã được Wolfe Friedman đưa ra vào 1920 như sau:

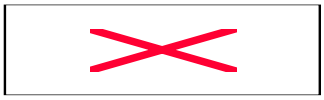
Định nghĩa 1.7.

Giả sử $x = x_1x_2 \ldots x_n$ là một xâu ký tự. Chỉ số trùng hợp của x (ký hiệu là $I_c(x)$) được định nghĩa là xác suất để hai phần tử ngẫu nhiên của x là đồng nhất. Nếu ký hiệu các tần suất của A,B,C,. . . ,Z trong x tương ứng là f_0,f_1 ,. . . f_{25} , có thể chọn hai phần tử của x theo ??? cách. Với mỗi i, $0 \le i \le 25$, có ??? cách chọn hai phần tử là i. Bởi vậy ta có công thức:



Ghi chú: Hệ số nhị thức ?????? xác định số cách chọn một tập con k đối tượng từ một tập n đối tượng.

Bây giờ, giả sử x là một xâu văn bản tiếng Anh. Ta kí hiệu các xác suất xuất hiện của các kí tự A,B,. . .,Z trong bảng 1.1 là p_0,...p_{25}. Khi đó:



do xác suất để hai phần tử ngẫu nhiên đều là A là p_0^2, xác suất để cả hai phần tử này đều bằng B bằng p_1^2 . . . Tình hình tương tự cũng xảy ra nếu x là một bản mã nhận được theo một hệ mã thay thế đơn bất kì. Trong trường hợp này, từng xác suất riêng rẽ sẽ bị hoán vị nhưng tổng ??? sẽ không thay đổi.

Bây giờ giả sử có một bản mã $y = y_1y_2. \ldots y_n$ được cấu trúc theo mật mã Vigenère. Ta xác định các xâu con m của $y(y_1,y_2,. \ldots ,y_m)$ bằng cách viết ra bản mã thành một hình chữ nhật có kích thước $m\times(n/m)$. Các hàng của ma trận này là các xâu con y_i, $1 \le i \le m$. Nếu m thực sự là độ dài khoá thì mỗi $I_c(y_i)$ phải xấp xỉ bằng 0,065. Ngược lại, nếu m không phải là độ dài khoá thì các xâu con y_i sẽ có vẻ ngẫu nhiên hơn vì chúng nhận được bằng cách mã dịch vòng với các khoá khác nhau. Xét thấy rằng, một xâu hoàn toàn ngẫu nhiên sẽ có:

Hai giá trị 0,065 và 0,038 đủ cách xa nhau để có thể xác định được độ dài từ khoá đúng (hoặc xác nhận giả thuyết đã được làm theo phép thử Kasiski). Hai kỹ thuật này sẽ được minh hoạ qua ví dụ dưới đây:

Ví dụ 1.11.

Bản mã nhận được từ mật mã Vigenère.

CHEEVOAHMAERATBTAXXWTNXBEEOPHBSBQMQEQERBW
RVXUOAKXAOSXXWEAHBWGJMMQMNKGRFVGXWTRZXWIAK
LXFPSKAUTEMNDCMGTSXMXBTUIADNGMGPSRELXNJELX
VRVPRTULHDNQWTWDTYGBPHXTFEALJHASVBFXNGLLCHR
ZBWELEKMSJIKNBHWRJGNMGJSGLXFEYPHAGNRBIEQJT
AMRVLCRREMNDGLXRRIMGNSNRWCHRQHAEYEVTAQEBBI
PEEWEVKAKOEWADREMXMTBHHCHRTKDNVRZCHRCLQOHP
WQAIIWXNRMGWOIIFKEE

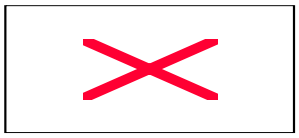
Trước tiên, ta hãy thử bằng phép thử Kasiski xâu bản mã CHR xuất hiện ở bốn vị trí trong bản mã, bắt đầu ở các vị trí 1, 166,236 và 286. Khoảng cách từ lần xuất hiện đầu tiên tới 3 lần xuất hiện còn lại tương ứng là 165,235 và 285. UCLN của 3 số nguyên này là 5, bởi vậy giá trị này rất có thể là độ dài từ khoá.

Ta hãy xét xem liệu việc tính các chỉ số trùng hợp có cho kết luận tương tự không. Với m = 1 chỉ số trùng hợp là 0,045. Với m = 2, có 2 chỉ số là 0,046 và 0,041. Với m = 3 ta có 0,043; 0,050; 0,047. Với m = 4 các chỉ số là 0,042; 0,039; 0,046; 0,040. Với m = 5 ta có các giá trị 0,063; 0,068; 0,069; 0,061 và 0,072. Điều này càng chứng tỏ rằng độ dại từ khoá là 5.

Với giả thiết trên, làm như thế nào để xác định từ khoá? Ta sẽ sử dụng khái niệm chỉ số trùng hợp tương hỗ của hai xâu sau:

Định nghĩa 1.8.

Giả sử $x = x_1x_2. . .x_n$ và $y = y_1y_2. . .y_{n'}$ là các xâu có n và n' kí tự anphabet tương ứng. Chỉ số trùng hợp tương hỗ của x và y (kí hiệu là $MI_c(x,y)$) được xác định là xác suất để một phần tử ngẫu nhiên của x giống với một phần tử ngẫu nhiên của y. Nếu ta kí hiệu các tần suất của A,B,. . .,Z trong x và y tương ứng là $f_0,f_1,. . .,f_{25}$ thì $MI_c(x,y)$ sẽ được tính bằng:



Với các giá trị m đã xác định, các xâu con y_i thu được bằng mã dịch vòng bản rõ. Giả sử K = (k_1,k_2,. . .,k_m) là từ khoá. Ta sẽ xem xét có thể đánh giá $MI_c(y_i,y_j)$ như thế nào. Xét một kí tự ngẫu nhiên trong y_i và một kí tự ngẫu nhiên trong y_j . Xác suất để cả hai kí tự là A bằng $p_{-ki}\, p_{-kj}$, xác suất để cả hai là B bằng $p_{1-ki}\, p_{1-kj}$,. . .(Cần chú ý rằng tất cả các chỉ số dưới đều được rút gọn theo modulo 26). Bởi vậy có thể ước lượng rằng:

$$MI_c(y_i, y_i) \approx \sum_{h=0}^{25} p_{h-ki}\, p_{h-kj} = \sum_{h=0}^{25} p_h\, p_{h+ki-kj}$$

Ta thấy rằng, giá trị ước lượng này chỉ phụ thuộc vào kiếu hiệu k_i-k_j mod 26 (được gọi là độ dịch tương đối của y_i và y_j). Cũng vậy, ta thấy rằng:

$$\sum_{h=0}^{25} p_h\, p_{h+1} = \sum_{h=0}^{25} p_h\, p_{h-1}$$

Bởi vậy độ dịch tương đối *l* sẽ dẫn đến cùng một ước lượng MI_c như độ dịch tương đối 26-*l* .

Ta lập bảng các ước lượng cho độ dịch tương đối trong phạm vi từ 0 đến 13.(Xem bảng 1.4).

Bảng 1.4. Các chỉ số trùng hợp tương hỗ tính được.

Độ dịch tương đối	**Giá trị tính được của MI_c**
0	0.065
1	0,039
2	0,032
3	0,034
4	0,044
5	0,033
6	0,036
7	0,039
8	0,034
9	0,034
10	0,038
11	0,045
12	0,039
13	0,043

Xét thấy rằng, nếu độ dịch tương đối khác 0 thì các ước lượng này thay đổi trong khoảng từ 0.031 đến 0,045; ngược lại nếu độ dịch tương đối bằng 0 thì ước lượng bằng 0,065. Có thể dùng nhận xét này để tạo nên một phỏng đoán thích hợp cho $l = k_i - k_j$ (độ dịch tương đối của y_i và y_j) như sau: Giả sử cố định y_i và xét việc mã hoá y_j bằng $e_0, e_1, e_2 \ldots$ Ta kí hiệu các kết quả bằng $y_j^0, y_j^1, \ldots$ Dễ dàng dùng các chỉ số $MI_c(y_i, y_j^g)$, $0 \le g \le 25$ theo công thức sau:

$$MI_c(x, y^g) = \frac{\sum_{i=0}^{25} f_i f'_{i-g}}{n.n'}$$

Khi $g = l$ thì MI_c phải gần với giá trị 0,065 vì độ dịch tương đối của y_i và y_j bằng 0. Tuy nhiên, với các giá trị $g \neq l$ thì MI_c sẽ thay đổi giữa 0,031 và 0,045.

Bằng kỹ thuật này, có thể thu được các độ dịch tương đối của hai xâu con y_i bất kỳ. Vấn đề còn lại chỉ là 26 từ khoá có thể và điều này dễ dàng tìm được bằng phương pháp tìm kiếm vét cạn.

Trở lại ví dụ 1.11 để minh hoạ.

Ví dụ 1.11(tiếp):

Ở trên đã giả định rằng, độ dài từ khoá là 5. Bây giờ ta sẽ thử tính các độ dịch tương đối. Nhờ máy tính, dễ dàng tính 260 giá trị $MI_c(y_i, y_j^g)$, trong đó $1 \le i \le j \le 5$; $0 \le g \le 25$. Các giá trị này được cho trên bảng 1.5. Với mỗi cặp (i,j), ta tìm các giá trị của $MI_c(y_i, y_j^g)$ nào gần với 0,065. Nếu có một giá trị duy nhất như vậy(Đối với mỗi cặp (i,j) cho trước), thì có thể phán đoán đó chính là giá trị độ dịch tương đối.

Trong bảng 1.5 có 6 giá trị như vậy được đóng khung. Chúng chứng tỏ khá rõ ràng là độ dịch tương đối của y_1 và y_2 bằng 9; độ dịch tương đối của y_2 và y_3 bằng 13; độ dịch tương đối của y_2 và y_5 bằng 7; độ dịch tương đối của y_3 và y_5 bằng 20; của y_4 và y_5 bằng 11. Từ đây có các phương trình theo 5 ẩn số K_1, K_2, K_3, K_4, K_5 như sau:

$$K_1 - K_2 = 9$$
$$K_1 - K_2 = 16$$
$$K_2 - K_3 = 13$$
$$K_2 - K_5 = 17$$
$$K_3 - K_5 = 20$$
$$K_4 - K_5 = 11$$

Điều này cho phép biểu thị các K_i theo K_1 ;

$$K_2 = K_1 + 17$$
$$K_3 = K_1 + 4$$
$$K_4 = K_1 + 21$$
$$K_5 = K_1 + 10$$

Như vậy khoá có khả năng là (K_1, K_1+17, K_1+4, K_1+21, K_1+10) với giá trị K_1 nào đó $\in Z_{26}$. Từ đây ta hy vọng rằng, từ khoá là một dịch vòng nào đó của AREVK. Bây giờ , không tốn nhiều công sức lắm cũng có thể xác định được từ khoá là JANET. Giải mã bản mã theo khoá này, ta thu được bản rõ sau:

The almond tree was in tentative blossom. The days were longer often ending with magnificient evenings of corrugated pink skies. The hunting seasun was over, with hounds and guns put away for six months. The vineyards were busy again as the well-organized farmers treated their vinesand the more lackadaisical neighbors hurried to do the pruning they have done in November.

Bảng 1.5. Các chỉ số trùng hợp tương hỗ quan sát được.

i	j	Giá trị của $MI_c(y_j,y_j^g)$
1	2	0,028 0,027 0,028 0,034 0,039 0,037 0,026 0,025 0,052 0,068 0,044 0,026 0,037 0,043 0,037 0,043 0,037 0,028 0,041 0,041 0,041 0,034 0,037 0,051 0,045 0,042 0,036
1	3	0,039 0,033 0,040 0,034 0,028 0,053 0,048 0,033 0,029 0,056 0,050 0,045 0,039 0,040 0,036 0,037 0,032 0,027 0,037 0,047 0,032 0,027 0,039 0,037 0,039 0,035
1	4	0,034 0,043 0,025 0,027 0,038 0,049 0,040 0,032 0,029 0,034 0,039 0,044 0,044 0,034 0,039 0,045 0,044 0,037 0,055 0,047 0,032 0,027 0,039 0,037 0,039 0,035
1	5	0,043 0,033 0,028 0,046 0,043 0,044 0,039 0,031 0,026 0,030 0,036 0,040 0,041 0,024 0,019 0,048 **0,070** 0,044 0,028 0,038 0,044 0,043 0,047 0,033 0,026
2	3	0,046 0,048 0,041 0,032 0,036 0,035 0,036 0,020 0,024 0,039 0,034 0,029 0,040 **0,067** 0,061 0,033 0,037 0,045 0,033 0,033 0,027 0,033 0,045 0,052 0,042 0,030
2	4	0,046 0,034 0,043 0,044 0,034 0,031 0,040 0,045 0,040 0,048 0,044 0,033 0,024 0,028 0,042 0,039 0,026 0,034 0,050 0,035 0,032 0,040 0,056 0,043 0,028 0,028
2	5	0,033 0,033 0,036 0,046 0,026 0,018 0,043 **0,080** 0,050 0,029 0,031 0,045 0,039 0,037 0,027 0,026 0,031 0,039 0,040 0,037 0,041 0,046 0,045 0,043 0,035 0,030
3	4	0,038 0,036 0,040 0,033 0,036 0,060 0,035 0,041 0,029 0,058 0,035 0,035 0,034 0,053 0,030 0,032 0,035 0,036 0,036 0,028 0,043 0,032 0,051 0,032 0,034 0,030
3	5	0,035 0,038 0,034 0,036 0,030 0,043 0,043 0,050 0,025 0,041 0,051 0,050 0,035 0,032 0,033 0,033 0,052 0,031 0,027 0,030 **0,072** 0,035 0,034 0,032 0,043 0,027
4	5	0,052 0,038 0,033 0,038 0,041 0,043 0,037 0,048 0,028 0,028 0,036 **0,061** 0,033 0,033 0,032 0,052 0,034 0,027 0,039 0,043 0,033 0,027 0,030 0,039 0,048 0,035

1.2.4.Tấn công với bản rõ đã biết trên hệ mật Hill.

Hệ mã Hill là một hệ mật khó pha hơn nếu tấn công chỉ với bản mã. Tuy nhiên hệ mật này dễ bị phá nếu tấn công bằng bản rõ đã biết. Trước tiên, giả sử rằng, thám mã đã biết được giá trị m đang sử dụng. Giả sử thám mã có ít nhất m cặp véc tơ khác nhau xj = $(x_{1,j}, x_{2,j}, , . . ., x_{m,j})$ và $y_j = (y_{1,j}, y_{2,j},...,y_{m,j})$ $(1 \le j \le m)$ sao cho $y_j = e_K(x_j)$, $1 \le j \le m$. Nếu xác định hai ma trận: $X = (x_{i,j})$ $Y = (y_{i,j})$ cấp m×m thì ta có phương trình ma trận Y = XK, trong đó ma trận K cấp m×m là khoá chưa biết. Với điều kiện ma trận Y là khả nghịch. Oscar có thể tính $K = X^{-1}Y$ và nhờ vậy phá được hệ mật. (Nếu Y không khả nghịch thì cần phải thử các tập khác gồm m cặp rõ - mã).

Ví dụ 1.12.

Giả sử bản rõ *Friday* được mã hoá bằng mã Hill với m = 2, bản mã nhận được là PQCFKU.

Ta có $e_K(5,17) = (15,16)$, $e_K(8,3) = (2,5)$ và $e_K(0,24) = (10,20)$. Từ hai cặp rõ - mã đầu tiên, ta nhận được phương trình ma trận:

$$\begin{pmatrix} 15 & 16 \\ 2 & 5 \end{pmatrix} = \begin{pmatrix} 5 & 17 \\ 8 & 3 \end{pmatrix} K$$

Dùng định lý 1.3, dễ dàng tính được:

$$\begin{pmatrix} 5 & 17 \\ 8 & 3 \end{pmatrix}^{-1} = \begin{pmatrix} 9 & 1 \\ 2 & 15 \end{pmatrix}$$

Bởi vậy:

$$K = \begin{pmatrix} 9 & 1 \\ 2 & 15 \end{pmatrix} \begin{pmatrix} 15 & 16 \\ 2 & 5 \end{pmatrix} = \begin{pmatrix} 7 & 19 \\ 8 & 3 \end{pmatrix}$$

Ta có thể dùng cặp rõ - mã thứ 3 để kiểm tra kết quả này.

Vấn đề ở đây là thám mã phải làm gì nếu không biết m?. Giả sử rằng m không quá lớn, khi đó thám má có thể thử với m = 2,3,. . . cho tới khi tìm được khoá. Nếu một giá trị giả định của m không đúng thì mà trận m×m tìm được theo thuật toán đã mô tả ở trên sẽ không tương thích với các cặp rõ - mã khác. Phương pháp này, có thể xác định giá trị m nếu chưa biết.

1.2.5. Thám mã hệ mã dòng xây dựng trên LFSR.

Ta nhớ lại rằng, bản mã là tổng theo modulo 2 của bản rõ và dòng khoá, tức $y_i = x_i + z_i \bmod 2$. Dòng khóa được tạo từ $(z_1,z_2,\ldots,z_m)$ theo quạn hệ đệ quy tuyến tính:

$$z_{m+1} = \sum_{j=0}^{m-1} c_j z_{i+1} \bmod 2$$

trong đó $c_0,\ldots,c_m \in Z_2$ (và $c_0 = 1$)

Vì tất cả các phép toán này là tuyến tính nên có thể hy vọng rằng, hệ mật này có thể bị phá theo phương pháp tấn công với bản rõ đã biết như trường hợp mật mã Hill. Giả sử rằng, Oscar có một xâu bản rõ $x_1x_2\ldots x_n$ và xâu bản mã tương ứng $y_1y_2\ldots y_n$. Sau đó anh ta tính các bít dòng khoá $z_i = x_i+y_i \bmod 2$, $1 \le i \le n$. Ta cũng giả thiết rằng Oscar cũng đã biết giá trị của m. Khi đó Oscar chỉ cần tính $c_0, \ldots, c_{m-1}$ để có thể tái tạo lại toàn bộ dòng khoá. Nói cách khác, Oscar cần phải có khả năng để xác định các giá trị của m ẩn số.

Với $i \ge 1$ bất kì ta có :

$$z_{m+1} = \sum_{j=0}^{m-1} c_j z_{i+j} \bmod 2$$

là một phương trình tuyến tính n ẩn. Nếu $n \ge 2n$ thì có m phương trình tuyến tính m ẩn có thể giải được.

Hệ m phương trình tuyến tính có thể viết dưới dạng ma trận như sau:

$$(z_{m+1}, z_{m+2},\ldots,z_{2m}) = (c_0, c_1,\ldots,c_{m-1}) \begin{bmatrix} z_1 & z_2 & \ldots & z_m \\ z_2 & z_3 & \ldots & z_{m+1} \\ \ldots & \ldots & \ldots & \ldots \\ z_m & z_{m+1} & \ldots & z_{2m-1} \end{bmatrix}$$

Nếu ma trận hệ số có nghịch đảo (theo modulo 2)thì ta nhận được nghiệm:

$$(c_0, c_1,\ldots,c_{m-1}) = (z_{m+1}, z_{m+2},\ldots,z_{2m}) \begin{bmatrix} z_1 & z_2 & \ldots & z_m \\ z_2 & z_3 & \ldots & z_{m+1} \\ \ldots & \ldots & \ldots & \ldots \\ z_m & z_{m+1} & \ldots & z_{2m-1} \end{bmatrix}^{-1}$$

Trên thực tế, ma trận sẽ có nghịch đảo nếu bậc của phép đệ quy được dùng để tạo dòng khoá là m.(xem bài tập). Minh hoạ điều này qua một ví dụ.

Ví dụ 1.13.

Giả sử Oscar thu được xâu bản mã

101101011110010

tương ứng với xâu bản rõ

011001111111001

Khi đó anh ta có thể tính được các bít của dòng khoá:

110100100001010

Ta cũng giả sử rằng, Oscar biết dòng khoá được tạo từ một thanh ghi dịch phản hồi (LFSR) có 5 tầng. Khi đó, anh ta sẽ giải phương trình mà trận sau (nhận được từ 10 bít đầu tiên của dòng khoá):

$$(0,1,0,0,0)=(c_0,c_1,c_2,c_3,c_4)\begin{bmatrix}1&1&0&1&0\\1&0&1&0&0\\0&1&0&0&1\\1&0&0&1&0\\0&0&1&0&0\end{bmatrix}$$

Có thể kiểm tra được rằng:

$$\begin{bmatrix}1&1&0&1&0\\1&0&1&0&0\\0&1&0&0&1\\1&0&0&1&0\\0&0&1&0&0\end{bmatrix}^{-1}=\begin{bmatrix}0&1&0&0&1\\1&0&0&1&0\\0&0&0&0&1\\0&1&0&1&1\\1&0&1&1&0\end{bmatrix}$$

Từ đó ta có:

$$(c_0,c_1,c_2,c_3,c_4)=(0,1,0,0,0)\begin{bmatrix}0&1&0&0&1\\1&0&0&1&0\\0&0&0&0&1\\0&1&0&1&1\\1&0&1&1&0\end{bmatrix}$$

$$= (1, 0, 0, 1, 0)$$

Như vậy phép đệ quy được dùng để tạo dòng khoá là:

$$z_{i+5} = z_i + z_{i+3} \bmod 2$$

1.3. CÁC CHÚ GIẢI VÀ TÀI LIỆU DẪN

Nhiều tài liệu về mật mã cổ điển đã có trong các giáo trình, chẳng hạn như giáo trình của Beker và Piper [BP82] và Denning [DE82]. Xác suất đánh

giá cho 26 kí tự được lấy của Beker và Piper. Cũng vậy, việc phân tích mã Vigenère được sửa đổi theo mô tả của Beker và Piper. Rosen [Ro93] là một tài liệu tham khảo tốt về lý thuyết số. Cơ sở của Đại số tuyến tính sơ cấp có thể tìm thấy trong sách của Anton [AN91]. Cuốn " Những người mã thám " của Kahn [KA67] là một cấu chuyện hấp dẫn và phong phú về mật mã cho tới năm 1967, trong đó Kahn khẳng định rằng mật mã Vigenère thực sự không phải là phát minh của Vigenère.

Mật mã Hill lần đầu tiên được mô tả trong [HI29]. Các thông tin về mật mã dòng có thể tìm được trong sách của Rueppel [RU86].

BÀI TẬP

1.1. Dưới đây là 4 bản mã thu được từ mã thay thế. Một bản thu được từ mã Vigenère, một từ mật mã Affine và một bản chưa xác định. Nhiệm vụ ở đây là xác định bản rõ trong mỗi trường hợp.

Hãy mô tả các bước cần thực hiện để giải mã mỗi bản mã (bao gồm tất cả các phân tích thống kê và các tính toán cần thực hiện).

Hai bản rõ đầu lấy từ cuốn " The diary of samuenl marchbanks " của Robertson Davies, Clack Iriwin,1947; bản rõ thứ tư lấy từ " Lake wobegon days" của Garrison Keillor, Viking Penguin, 1985.

a) Mã thay thế:
EMGLXUDCGDNCUSWYXFPHNSFCYKDPUMLWGYICOXYFIPJCK
QPKUGKMGOLICGINCGACKFNIFACYKZSCKXECJCKFHYFXCG
0IDPKZCNKSHICGIWYGKKGKGOLDSILKGOIUFIGLEDFPWZU
GFZCCNDGYYFFUSZCNXEOJNCGYEOWEUPXEZGACGNFGLKNF
ACIGOYCKXCJUCIUZCFZCCNDGYYSFEUEKUZCSOCSZCCNC
IACZEJNCFFZEJZEGMXCYHCJUMGKUSI

Chỉ dẫn: F sẽ giải mã thành W.

b) Hệ mã Vigenère
KCCPKBGUFDPHQTYAVINRRTMVGRKDNBVFDETDGLLTXRGUD
DKBTMBPVGEGLTGCKQRACQCWDNAWCRXIZAKSTLEWRPTYC
QKYVXCHKFTPONCQQRHJVAJUWETMCMSPKQDYHJVDAHCTRL
SVSKCGCZQQDZXGSFRLFWCWSJTBHAFSIASPRJAHKJRJUMP
FFSQNRWXCVYCGAONWDDKACKAWBBIKFTIOVKCGGHJVLNHI
CWHJVLNHIQIBTKHJVNPIST

c) Hệ mã Affine.

KQEREJEBCPPCJCRKIEACUZBKRVPKRBCIBQCARBJCVFCUP
KRIOFKPACUZQEPBKRXPEIIEABDKPBCPFCDCAFIEABDKP
BCPFEQPKAZBKRHAIBKAPCCIBURCCDKDCCJCIDFUIXPAFF
ERBICZDFKABICBBENEFCUPJCVKABPCYDCCDPKBCOCPERK
IVKSCPICBRKIJPKABI

d) Hệ mã chưa xác định được.

BNVSNSIHQCEELSSKKYERISJKXUMBGYKAMQLJTYAVFBKVT
DVBPVVRJYYLAOKYMPQSCGDLFLLPROYGEFEBUUALRWXM
MASAZLGLEDFJBZAVVPXWYCGJXASCBYEHOSNMULKCEAHTQ
OKMFLEBKFXLRRFDTZXCIWBJSICBGAWDVYDHAVFJXZIBKC
GJIWEAHTTOEWTUHKRQVVRGZBXYIREMMASCSPBNLHJGBLR
FFJELHWEYLWISTFVVYFJCMHYURUFSFMGESIGRLWALSWM
NUHSIMYYITCCQPZSICEHBCCMZFEGVJYOCDEMMPGHVAAMU
ELCMOEHVLTIPSUYILVGFLMVWDVYDBTHERAYISYSGKVSUU
HYHGGCKTMBLRX

1.2. a) Có bao nhiêu ma trận khả nghịch cấp 2×2 trên Z_{26} .

b) Giả sử p là số nguyên tố. Hãy chứng tỏ số các ma trận khả nghịch cấp 2×2 trên Z_p là $(p^2-1)(p^2-p)$.

Chỉ dẫn: Vì p là số nguyên tố nên Z_p là một trường. Hãy sử dụng khẳng định sau: Một ma trận là khả nghịch trên một trường là khả nghịch khi và chỉ khi các hàng của nó là các véc tơ độc lập tuyến tính (tức không tồn tại một tổ hợp tuyến tính các hàng khác 0 mà tổng của chúng là một véc tơ toàn số 0).

c) Với p là số nguyên tố và m là một số nguyên ≥ 2. Hãy tìm công thức tính số các ma trận khả nghịch cấp m×m trên Z_p.

1.3. Đôi khi chọn một khoá mà phép mã và giải mã là đồng nhất rất hữu ích. Trong trường hợp mất mã Hill, ta phải tìm các ma trận K sao cho $K = K^{-1}$ (ma trận này được gọi là ma trận đối hợp). Trên thực tế, Hill đã đề nghị sử dụng các ma trận này làm khoá trong các hệ mật của mình. Hãy xác định số các ma trận đối hợp trên Z_{26} trong trường hợp m = 2.

Chỉ dẫn: Hãy dùng công thức trong định lý 1.3 và để ý rằng detA = ± với một ma trận đối hợp trên Z_{26}.

1.4. Giả sử ta đã biết rằng bản rõ " *conversation* " sẽ tạo nên bản mã " HIARRTNUYTUS " (được mã theo hệ mã Hill nhưng chưa xác định được m). Hãy xác định ma trận mã hoá.

1.5. Hệ mã Affine - Hill là hệ mã Hill được sửa đổi như sau: Giả sử m là một số nguyên dương và $\mathcal{P} = C = (Z_{26})^m$. Trong hệ mật này, khoá K gồm các cặp (L,b), trong đó L là mọt ma trận khả nghịch cấp m×m trên Z_{26} và $b \in (Z_{26})^m$. Với $x = (x_1,\ldots,x_m) \in \mathcal{P}$ và $K = (L,b) \in \mathcal{K}$, ta tính $y = e_K(x) = (y_1,\ldots,y_m)$ theo công thức y = xL + b. Bởi vậy, nếu $L = (l_{i,j})$ và $b = (b_1,\ldots,b_m)$ thì:

$$(y_1,\ldots,y_m) = (x_1,\ldots,x_m)\begin{bmatrix} l_{1,1} & l_{1,2} & . & . & . & l_{1,m} \\ l_{2,1} & l_{2,2} & . & . & . & l_{2,m} \\ . & . & . & . & . & . \\ . & . & . & . & . & . \\ l_{m,1} & l_{m,2} & . & . & . & l_{m,m} \end{bmatrix} + (b_1,\ldots,b_m)$$

Giả sử Oscar đã biết bản rõ là "*adisplayedequation*" và bản mã tương ứng là " DSRMSIOPLXLJBZULLM". Oscar cũng biết m =3. Hình tính khoá và chỉ ra tất cả các tính toán cần thiết.

1.6. Sau đây là cách thám mã hệ mã Hill sử dụng phương pháp tấn công chỉ với bản mã. Giả sử ta biết m = 2. Chia các bản mã thành các khối có độ dài 2 kí tự (các bộ đôi). Mỗi bộ đôi này là bản mã của một bộ đôi của bản rõ nhờ dùng một ma trận mã hoá chưa biết. Hãy nhặt ra các bộ đôi thường gặp nhất trong bản mã và coi rằng đó là mã của một bộ đôi thường gặp trong danh sách ở bảng 1.1 (ví dụ TH và ST). Với mỗi giả định, hãy thực hiện phép tấn công với bản rõ đã biết cho tới khi tìm được ma trận giải mã đúng.

Sau đây là một ví dụ về bản mã để bạn giải mã theo phương pháp đã nêu:

LMQETXYEAGTXCTUIEWNCTXLZEWUAISPZYVAPEWLMGQWYA
XFTCJMSQCADAGTXLMDXNXSNPJQSYVAPRIQSMHNOCVAXFV.

1.7. Ta sẽ mô tả một trường hợp đặc biệt của mã hoán vị. Giả sử m, n là các số nguyên dương. Hãy viết bản rõ theo thành từng hàng thành một hình chữ nhật m×n. Sau đó tạo ra bản mã bằng cách lấy các cột của hình chữ nhật này. Ví dụ, nếu m = 4, n = 3 thì ta sẽ mã hoá bản rõ "*cryptography*" bằng cách xây dựng hình chữ nhật :

cryp
togr
aphy

Bản mã sẽ là: 'CTAROPYGHPRY'

a) Hãy mô tả cách Bob giải mã một bản mã (với m, n đã biết)
b) Hãy giải mã bản mã sau: (nhận được theo phương pháp đã nêu):

MYAMRARUYIQTENCTORAHROYWDSOYEOUARRGDERNOGW

1.8. Có 8 phép đệ quy tuyến tính bậc 4 khác nhau trên Z_2 với $c_0 = 1$. Hãy xác định những phép đệ quy nào tạo được dòng khoá có chu kỳ 15 (với véc tơ khởi tạo khác 0).

1.9. Mục đích của bài tập này để chứng minh khẳng định ở phần 1.2.5 là : ma trận hệ số cấp m×m có nghịch đảo. Điều này tương đương với khẳng định rằng, các hàng ma trận này là các véc tơ độc lập tuyến tính trên Z_2.

Giả sử rằng phép đệ quy có dạng:

$$z_{m+i} = \sum_{j=0}^{m-1} c_j z_{i+j} \ mod 2$$

($z_1,. . .,z_m$) là véc tơ khởi tạo. Với i ≥ 1 ta xác định:

$$v_i = (z_i,. . .,z_{i+m-1})$$

Chú ý rằng, ma trận hệ số có các véc tơ $v_1,. . .,v_m$ là các hàng của nó. Bởi vậy, nhiệm vụ của ta là chứng tỏ rằng m véc tơ này là độc lập tuyến tính.

Hãy chứng minh hai khẳng định sau:

a) Với i ≥ 1 bất kì:

$$v_{m+i} = \sum_{j=0}^{m-1} c_j v_{i+j} \ mod 2$$

b)Chọn h là số nguyên nhỏ nhất sao cho tồn tại một tổ hợp tuyến tính không tầm thường của các véc tơ $v_1,. . .,v_h$ có tổng là véc tơ (0, . . . , 0) theo modulo 2. Khi đó:

$$v_h = \sum_{j=0}^{h-2} c_j v_{j+1} \ mod 2$$

(Các α_j không đồng nhất bằng 0). Để ý rằng, h ≤ m+1 vì m+1 là véc tơ bất kỳ trong không gian tuyến tính m chiều đều phụ thuộc tuyến tính .

c) Hãy chứng tỏ rằng dòng khoá phải thảo mãn phép đệ quy:

$$z_{h-1+i} = \sum_{j=0}^{h-2} \alpha_j z_{j+i} \ mod 2$$

với bất kì i ≥ 1.

d) Ta nhận thấy rằng, nếu $h \leq m$ thì dòng khoá thảo mãn phép đệ quy tuyến tính có bậc nhỏ hơn m. Điều này mâu thuẫn. Bởi vậy $h = m + 1$ và ma trận phải là khả nghịch.

1.10. Hãy giải mã bản mã sau (thu được từ mã khoá tự sinh) bằng phương pháp tìm khoá vét cạn.

MALVVMAFBHBUQPTSOXALTGVWWRG

1.11. Ta sẽ mô tả một hệ mã dòng là biến thể của mã Vigenère như sau. Với một từ khoá độ dài m cho trước ($k_1,. . .,k_m$), ta tạo dòng khoá theo quy tắc $z_i=k_i$ $(1 \leq i \leq m)$, $z_{i+m} = z_i+1 \bmod 26$ ($i \geq m+1$). Nói cách khác, mỗi lần dùng từ khoá ta sẽ thay mỗi kí tự bằng kí tự đứng sau nó theo modulo 26. Ví dụ, nếu SUMMER là từ khoá thì ta dùng SUMMER để mã hoá 6 kí tự đầu.,sau đó dùng TVNNFS để mã hoá 6 kí tự tiếp theo và cú tiếp tục như vậy.

Hãy mô tả cách có thể dùng khái niệm chỉ số trùng hợp như thế nào để trước hết là xác định độ dài từ khoá và sau đó là tìm từ khoá.

Hãy kiểm tra phương pháp của bạn bằng cách bằng cách phân tích bản mã sau:

IYMYSILONRFNCQXQJEDSHBUIBCJUZBOLFQYSCHATPEQGQ
JEJNGNXZWHHGUFSUKULJQACZKKJOAAHGKEMTAFGMKVRDO
PXNEHEKZNKFSKIFRQVHHOVXINPHMRTJPYWQGJWPUUKFP
OAWPMRKKQZWLQDYAZDRMLPBJKJOBWIWPSEPVVQMBCRYVC
RUZAAOUMBCHDAGDIEMSZFZHALIGKEMJJFPCIWKRMLMPIN
AYOFIREAOLDTHITDVRMSE

Bản rõ được lấy từ "The codebreakers" của D.Kahn, 1967.

CHƯƠNG 2

LÝ THUYẾT SHANNON

Năm 1949, Claude shannon đã công bố một bài báo có nhan đề " Lý thuyết thông tin trong các hệ mật" trên tạp chí " The Bell System Technical Journal". Bài báo đã có ảnh hưởng lớn đến việc nghiên cứu khoa học mật mã. Trong chương này ta sẽ thảo luận một vài ý tưởng trong lý thuyết của Shannan.

2.1 ĐỘ MẬT HOÀN THIỆN.

Có hai quan điểm cơ bản về độ an toàn của một hệ mật.

Độ an toàn tính toán:

Đo độ này liên quan đến những nỗ lực tính toán cần thiết để phá một hệ mật. Một hệ mật là an toàn về mặt tính toán nếu có một thuật toán tốt nhất để phá nó cần ít nhất N phép toán, N là số rất lớn nào đó. Vấn đề là ở chỗ, không có một hệ mật thực tế đã biết nào có thể được chứng tỏ là an toàn theo định nghĩa này. Trên thực tế, người ta gọi một hệ mật là "an toàn về mặt tính toán" nếu có một phương pháp tốt nhất phá hệ này nhưng yêu cầu thời gian lớn đến mức không chấp nhận được.(Điều này tất nhiên là rất khác với việc chứng minh về độ an toàn).

Một quan điểm chứng minh về độ an toàn tính toán là quy độ an toàn của một hệ mật về một bài toán đã được nghiên cứu kỹ và bài toán này được coi là khó. Ví dụ, ta có thể chứng minh một khẳng định có dạng " Một hệ mật đã cho là an toàn nếu không thể phân tích ra thừa số một số nguyên n cho trước". Các hệ mật loại này đôi khi gọi là " an toàn chứng minh được". Tuy nhiên cần phải hiểu rằng, quan điểm này chỉ cung cấp một chứng minh về độ an toàn có liên quan đế một bài toán khác chứ không phải là một chứng minh hoàn chỉnh về ọ an toàn. (Tình hình này cũng tương tự như việc chứng minh một bài toán là NP đầy đủ: Có thể chứng tỏ bài toán đã cho chí ít cũng khó như một bài toán NP đầy đủ khác , song không phải là một chứng minh hoàn chỉnh về độ khó tính toán của bài toán).

Độ an toàn không điều kiện.

Độ đo này liện quan đến độ an toàn của các hệ mật khi không có một hạn chế nào được đặt ra về khối lượng tính toán mà Oscar được phép thực

hiện. Một hệ mật được gọi là an toàn không điều kiện nếu nó không thể bị phá thậm chí với khả năng tính toán không hạn chế.

Khi thảo luận về độ an toàn của một mật, ta cũng phải chỉ ra kiểu tấn công đang được xem xét. Trong chương 1 đã cho thấy rằng, không một hệ mật nào trong các hệ mã dịch vòng, mã thay thế và mã Vigenère được coi là an toàn về mặt tính toán với phương pháp tấn công chỉ với bản mã (Với khối lượng bản mã thích hợp).

Điều này mà ta sẽ làm trong phần này là để phát triển lý thuyết về các hệ mật có độ an toàn không điều kiện với phương pháp tấn công chỉ với bản mã. Nhận thấy rằng, cả ba hệ mật nêu trên đều là các hệ mật an toàn vô điều kiện chỉ khi mỗi pkần tử của bản rõ được mã hoá bằng một khoá cho trước!.

Rõ ràng là độ an toàn không điều kiện của một hệ mật không thể được nghiên cứu theo quan điểm độ phức tạp tính toán vì thời gian tính toán cho phép không hạn chế. Ở đây lý thuyết xác suất là nền tảng thích hợp để nghiên cứu về độ an toàn không điều kiện. Tuy nhiên ta chỉ cần một số kiến thức sơ đẳng trong xác suất; các định nghĩa chính sẽ được nêu dưới đây.

Định nghĩa 2.1.

Giả sử X và Y là các biến ngẫu nhiên. Kí hiệu xác suất để X nhận giá trị x là p(x) và để Y nhận giá trị y là p(y). Xác suất đồng thời p(x,y) là xác suất để X nhận giá trị x và Y nhận giá trị y. Xác suất có điều kiện p(x / y) là xác suất để X nhận giá trị với điều kiện Y nhận giá trị y. Các biến ngẫu nhiên X và Y được gọi là độc lập nếu p(x,y) = p(x) p(y) với mọi giá trị có thể x của X và y của Y.

Quan hệ giữa xác suất đồng thời và xác suất có điều kiện được biểu thị theo công thức:

$$p(x,y) = p(x \mid y)\, p(y)$$

Đổi chỗ x và y ta có :

$$p(x,y) = p(y \mid x)\, p(x)$$

Từ hai biểu thức trên ta có thể rút ra kết quả sau:(được gọi là định lý Bayes)

Định lý 2.1: (Định lý Bayes).

Nếu p(y) > 0 thì:

$$p(x \mid y) = \frac{p(x)\, p(y \mid x)}{p(y)}$$

Hệ quả 2.2.

X và Y là các biến độc lập khi và chỉ khi:

$$p(x \mid y) = p(x) \text{ với mọi } x,y.$$

Trong phần này ta giả sử rằng, một khoá cụ thể chỉ dùng cho một bản mã. Giả sử có một phân bố xác suất trên không gian bản rõ $\mathcal{P}$. Kí hiệu xác suất tiên nghiệm để bản rõ xuất hiện là $p_{\mathcal{P}}(x)$. Cũng giả sử rằng, khóa K được chọn (bởi Alice và Bob) theo một phân bố xác suất xác định nào đó. (Thông thường khoá được chọn ngẫu nhiên, bởi vậy tất cả các khoá sẽ đồng khả năng, tuy nhiên đây không phải là điều bắt buộc). Kí hiệu xác suất để khóa K được chọn là $p_{\mathcal{K}}(K)$. Cần nhớ rằng khóa được chọn trước khi Alice biết bản rõ. Bởi vậy có thể giả định rằng khoá K và bản rõ x là các sự kiện độclập.

Hai phân bố xác suất trên $\mathcal{P}$ và $\mathcal{K}$ sẽ tạo ra một phân bố xác suất trên $\mathcal{C}$. Thật vậy, có thể dễ dàng tính được xác suất $p_{\mathcal{C}}(y)$ với y là bản mã được gửi đi. Với một khoá $K \in \mathcal{K}$, ta xác định:

$$C(K) = \{ e_K(x) : x \in \mathcal{P} \}$$

Ở đây C(K) biểu thị tập các bản mã có thể K là khóa. Khi đó với mỗi $y \in \mathcal{C}$, ta có :

$$p_{\mathcal{C}}(y) = \sum_{\{K:y \in C(K)\}} p_{\mathcal{K}}(K)\, p_{\mathcal{P}}(d_K(y))$$

Nhận thấy rằng, với bất kì $y \in \mathcal{C}$ và $x \in \mathcal{P}$, có thể tính được xác suất có điều kiện $p_{\mathcal{C}}(y \mid x)$.(Tức là xác suất để y là bản mã với điều kiện bản rõ là x):

$$p_{\mathcal{C}}(y \mid x) = \sum_{\{K:x = d_K(y)\}} p_{\mathcal{K}}(K)$$

Bây giờ ta có thể tính được xác suất có điều kiện $p_{\mathcal{P}}(x \mid y)$ (tức xác suất để x là bản rõ với điều kiện y là bản mã) bằng cách dùng định lý Bayes. Ta thu được công thức sau:

$$p_{\mathcal{P}}(x) = \sum_{\{K:x = d_K(y)\}} p_{\mathcal{K}}(K)$$

$$p_{\mathcal{P}}(y \mid x) = \frac{}{\sum_{\{k,U:y \in c(k)\}} p_{\mathcal{K}}(K)\, p_{\mathcal{P}}(d_K(y))}$$

Các phép tính này có thể thực hiện được nếu biết được các phân bố xác suất.

Sau đây sẽ trình bày một ví dụ đơn giản để minh hoạ việc tính toán các phân bố xác suất này.

Ví dụ 2.1.

Giả sử $\mathcal{P} = \{a,b\}$ với $p_{\mathcal{P}}(a) = 1/4$, $p_{\mathcal{P}}(b) = 3/4$. Cho $K = \{ K_1, K_2, K_3\}$ với $p_{\mathcal{K}}(K_1) = 1/2$, $p_{\mathcal{K}}(K_2) = p_{\mathcal{K}}(K_3) = 1/4$. Giả sử $\mathcal{C} = \{1,2,3,4\}$ và các hàm mã được xác định là $e_{K1}(a) = 1$, $e_{K2}(b) = 2$, $e_{K2}(a) = 2$, $e_{K2}(b) = 3$, $e_{K3}(a) = 3$, $e_{K3}(a) = 4$. Hệ mật này được biểu thị bằng ma trận mã hoá sau:

	a	b
K_1	1	2
K_2	2	3
K_3	2	4

Tính phân bố xác suất p_C ta có:

$$p_C(1) = 1/8$$
$$p_C(2) = 3/8 + 1/16 = 7/16$$
$$p_C(3) = 3/16 + 1/16 = 1/4$$
$$p_C(4) = 3/16$$

Bây giờ ta đã có thể các phân bố xác suất có điều kiện trên bản rõ với điều kiện đã biết bản mã. Ta có :

$p_{\mathcal{P}}(a \mid 1) = 1$ $p_{\mathcal{P}}(b \mid 1) = 0$ $p_{\mathcal{P}}(a \mid 2) = 1/7$ $p_{\mathcal{P}}(b \mid 2) = 6/7$
$p_{\mathcal{P}}(a \mid 3) = 1/4$ $p_{\mathcal{P}}(b \mid 3) = 3/4$ $p_{\mathcal{P}}(a \mid 4) = 0$ $p_{\mathcal{P}}(b \mid 4) = 1$

Bây giờ ta đã có đủ điều kiện để xác định khái niệm về độ mật hoàn thiện. Một cách không hình thức, độ mật hoàn thiện có nghiã là Oscar với bản mã trong tay không thể thu được thông tin gì về bản rõ. Ý tưởng này sẽ được làm chính xác bằng cách phát biểu nó theo các thuật ngữ của các phân bố xác suất định nghĩa ở trên như sau:

Định nghĩa 2.2.

Một hệ mật có độ mật hoàn thiện nếu $p_{\mathcal{P}}(x \mid y) = p_{\mathcal{P}}(x)$ với mọi $x \in \mathcal{P}$, $y \in \mathcal{C}$. Tức xác suất hậu nghiệm để bản rõ là x với điều kiện đả thu được bản mã y là đồng nhất với xác suất tiên nghiệm để bản rõ là x.

Trong ví dụ 2.1 chỉ có bản mã 3 mới thoả mãn tính chất độ mật hoàn thiện, các bản mã khác không có tính chất này.

Sau đây sẽ chứng tỏ rằng, MDV có độ mật hoàn thiện. Về mặt trực giác, điều này dường như quá hiển nhiên. Với mã dịch vòng, nếu đã biết một phần tử bất kỳ của bản mã $y \in Z_{26}$, thì một phần tử bất kỳ của bản rõ $x \in Z_{26}$ cũng có thể là bản mã đả giải của y tuỳ thuộc vào giá trị của khoá. Định lý

sau cho một khẳng định hình thức hoá và được chứng minh theo các phân bố xác suất.

Định lý 2.3.

Giả sử 26 khoá trong MDV có xác suất như nhau và bằng1/26 khi đó MDV sẽ có độ mật hoàn thiện với mọi phân bố xác suất của bản rõ.

Chứng minh: Ta có $\mathcal{P} = C = \mathcal{K} = Z_{26}$ và với $0 \le K \le 25$, quy tắc mã hoá e_K là $e_K(x) = x + K \bmod 26$ $(x \in 26)$. Trước tiên tính phân bố $\mathcal{P}_C$. Giả sử $y \in Z_{26}$, khi đó:

$$p_C(y) = \sum_{K \in Z_{26}} p_{\mathcal{K}}(K)\, p_{\mathcal{P}}(d_K(y))$$
$$= \sum_{K \in Z_{26}} 1/26\, p_{\mathcal{P}}(y - K)$$
$$= 1/26 \sum_{K \in Z_{26}} p_{\mathcal{P}}(y - K)$$

Xét thấy với y cố định, các giá trị y -K mod 26 sẽ tạo thành một hoán vị của Z_{26} và $p_{\mathcal{P}}$ là một phân bố xác suất. Bởi vậy ta có:

$$\sum_{K \in Z_{26}} p_{\mathcal{P}}(y - K) = \sum_{y \in Z_{26}} p_{\mathcal{P}}(y)$$

$$= 1$$

Do đó $p_C(y) = 1/26$

với bất kỳ $y \in Z_{26}$.

Tiếp theo ta có:

$$p_C(y|x) = p_{\mathcal{K}}(y - x \bmod 26)$$

$$= 1/26$$

Vơi mọi x,y vì với mỗi cặp x,y, khóa duy nhất K (khoá đảm bảo $e_K(x) = y$) là khoá $K = y\text{-}x \bmod 26$. Bây giờ sử dụng định lý Bayes, ta có thể dễ dàng tính:

$$p_{\mathcal{P}}(x|y) = \frac{p_{\mathcal{P}}(x)\, p_C(y|x)}{p_C(y)}$$

$$= \frac{p_{\mathcal{P}}(x) \cdot (1/26)}{(1/26)}$$

$$= p_{\mathcal{P}}(x)$$

Bởi vậy, MDV có độ mật hoàn thiện.

Như vậy, mã dịch vòng là hệ mật không phá được miễn là chỉ dùng một khoá ngẫu nhiên để mã hoá mỗi ký tự của bản rõ.

Sau đây sẽ ngiên cứu độ mật hoàn thiện trong trường hợp chung. Trước tiên thấy rằng,(sử dụng định lý Bayes) điều kiện để $p_{\mathcal{P}}(x \mid y) = p_{\mathcal{P}}(x)$ với mọi $x \in \mathcal{P}$, $y \in \mathcal{P}$ là tương đương với $p_C(y \mid x) = p_C(y)$ với mọi $x \in \mathcal{P}$, $y \in \mathcal{P}$.

Giả sử rằng $p_C(y) > 0$ với mọi $y \in C$ ($p_C(y) = 0$ thì bản mã sẽ không được dùng và có thể loại khỏi C). Cố định một giá trị nào đó $x \in \mathcal{P}$. Với mỗi $y \in C$ ta có $p_C(y \mid x) = p_C(y) > 0$. Bởi vậy, với mỗi $y \in C$ phải có ít nhất một khoá K sao cho $e_K(x) = y$. Điều này dẫn đến $|\mathcal{K}| \geq |C|$. Trong một hệ mật bất kỳ ta phải có $|C| \geq |\mathcal{P}|$ vì mỗi quy tắc mã hoá là một đơn ánh. Trong trường hợp giới hạn, $|\mathcal{K}| = |C| = |\mathcal{P}|$, ta có định lý sau (Theo Shannon).

Định lý 2.4

Giả sử $(\mathcal{P}, C, \mathcal{K}, \mathcal{E}, \mathcal{D})$ là một hệ mật , trong đó $|\mathcal{K}| = |C| = |\mathcal{P}|$. Khi đó, hệ mật có độ mật hoàn thiện khi và mỗi khi khoá K được dùng với xác suất như nhau bằng $1/|\mathcal{K}|$, và mỗi $x \in \mathcal{P}$,mỗi $y \in C$ có một khoá duy nhất K sao cho $e_K(x) = y$.

Chứng minh

Giả sử hệ mật đã cho có độ mật hoàn thiện. Như đã thấy ở trên, với mỗi $x \in \mathcal{P}$ và $y \in C$, phải có ít nhất một khoá K sao cho $e_K(x) = y$. Bởi vậy ta có bất đẳng thức:

$$|C| = |\{e_K(x) : K \in \mathcal{K}\}| \leq |\mathcal{K}|$$

Tuy nhiên, ta giả sử rằng $|C| = |\mathcal{K}|$. Bởi vậy ta phải có:

$$|\{e_K(x) : K \in \mathcal{K}\}| = |\mathcal{K}|$$

Tức là ở đây không tồn tại hai khoá K_1 và K_2 khác nhau để $e_{K1}(x) = e_{K2}(x) = y$. Như vậy ta đã chứng tỏ được rằng, với bất kỳ $x \in \mathcal{P}$ và $y \in C$ có đúng một khoá K để $e_K(x)=y$.

Ký hiệu n = | $\mathcal{K}$| . Giả sử $\mathcal{P}$ = { x_i: 1 ≤ i ≤ n } và cố định một giá trị y ∈$\mathcal{C}$. Ta có thể ký hiệu các khoá $K_1,K_2,\ldots,K_n$ sao cho $e_{Ki}(x_i) = y_i$, 1 ≤ i ≤ n. Sử dụng định lý Bayes ta có:

$$p_{\mathcal{P}}(x_i|y) = \frac{p_{\mathcal{C}}(y|x_i)\, p_{\mathcal{P}}(x_i)}{p_{\mathcal{C}}(y)}$$

$$= \frac{p_{\mathcal{K}}(K_1).\,(p_{\mathcal{P}}(x_i))}{p_{\mathcal{C}}(y)}$$

Xét điều kiện độ mật hoàn thiện $p_{\mathcal{P}}(x_i|y) = p_{\mathcal{P}}(x_i)$. Điều kiện này kéo theo $p_{\mathcal{K}}(K_i) = p_{\mathcal{C}}(y)$ với 1 ≤ i ≤ n. Tức là khoá được dùng với xác suất như nhau (chính bằng $p_{\mathcal{C}}(y)$). Tuy nhiên vì số khoá là | $\mathcal{K}$| nên ta có $p_{\mathcal{K}}(K) = 1/|\mathcal{K}|$ với mỗi K ∈$\mathcal{K}$.

Ngược lại, giả sử hai điều giả định đều thảo mãn. Khi đó dễ dàng thấy được hệ mật có độ mật hoàn thiện với mọi phân bố xác suất bất kỳ của bản rõ (tương tự như chướng minh định lý 2.3). Các chi tiết dành cho bạn đọc xem xét.

Mật mã khoá sử dụng một lần của Vernam (One-Time-Pad:OTP) là một ví dụ quen thuộc về hệ mật có độ mật hoàn thiện. Gillbert Verman lần đầu tiên mô tả hệ mật này vào năm 1917. Hệ OTP dùng để mã và giải mã tự động các bản tin điện báo. Điều thú vị là trong nhiều năm OTP được coi là một hệ mật không thể bị phá nhưng không thể chướng minh cho tới khi Shannon xây dựng được khái niệm về độ mật hoàn thiện hơn 30 năm sau đó.

Mô tả về hệ mật dùng một lần nêu trên hình 2.1.

Sử dụng định lý 2.4, dễ dàng thấy rằng OTP có độ mật hoàn thiện. Hệ thống này rất hấp dẫn do dễ thực hiện mã và giải mã.

Vernam đã đăng ký phát minh của mình với hy vọng rằng nó sẽ có ứng dụng thương mại rộng rãi. Đáng tiếc là có nhưỡng những nhược điểm quan trọng đối với các hệ mật an toàn không điều kiện, chẳng hạn như OTP. Điều kiện $|\mathcal{K}| \geq |\mathcal{P}|$ có nghĩa là lượng khóa (cần được thông báo một cách bí mật) cũng lớn như bản rõ. Ví dụ , trong trường hợp hệ OTP, ta cần n bit khoá để mã hoá n bit của bản rõ. Vấn đề này sẽ không quan trọng nếu có thể dùng cùng một khoá để mã hoá các bản tin khác nhau; tuy nhiên, độ an toàn của các hệ mật an toàn không điều kiện lại phụ thuộc vào một thực tế là mỗi

khoá chỉ được dùng cho một lần mã. Ví dụ OTP không thể đứng vững trước tấn công chỉ với bản rõ đã biết vì ta có thể tính được K băngf phép hoặc loại trừ xâu bít bất kỳ x và $e_K(x)$. Bởi vậy, cần phải tạo một khóa mới và thông báo nó trên một kênh bảo mật đối với mỗi bản tin trước khi gửi đi. Điều nàytạo ra khó khăn cho vấn đề quản lý khoá và gây hạn chế cho việc sử dụng rộng rãi OTP. Tuy nhiên OTP vẫn được áp dụng trong lĩnh vực quân sự và ngoại giao, ở những lĩnh vực này độ an toàn không điều kiện có tầm quan trọng rất lớn.

Hình 2.1. Hệ mật sử dụng khoá một lần (OTP)

Giả sử n ≥1 là số nguyên và $\mathcal{P} = \mathcal{C} = \mathcal{K} = (Z_2)^n$. Với K $(Z_2)^n$, ta xác định $e_K(x)$ là tổng véc tơ theo modulo 2 của K và x (hay tương đương với phép hoặc loại trừ của hai dãy bit tương ứng). Như vậy, nếu $x = (x_1,..., x_n)$ và $K = (K_1,..., K_n)$ thì:

$$e_K(x) = (x_1 + K_1,..., x_n + K_n) \bmod 2.$$

Phép mã hoá là đồng nhất với phép giải mã. Nếu $y = (y_1,..., y_n)$ thì:

$$d_K(y) = (y_1 + K_1,..., y_n + K_n) \bmod 2.$$

Lịch sử phát triển của mật mã học là quá trình cố gắng tạo các hệ mật có thể dùng một khoá để tạo một xâu bản mã tương đối dài (tức có thể dung một khoá để mã nhiều bản tin) nhưng chí ít vẫn còn dữ được độ an toàn tính toán. Chuẩn mã dữ liệu (DES) là một hệ mật thuộc loại này (ta sẽ nghiên cứu vấn đề này trong chương 2).

2.2. ENTROPI

Trong phần trước ta đã thảo luận về khái niệm độ mật hoàn thiện và đặt mối quan tâm vào một trường hợp đặc biệt, khi một khoá chỉ được dùng cho một lần mã. Bây giờ ta sẽ xét điều sẽ xẩy ra khi có nhiều bản rõ được mã bằng cùng một khoá và bằng cách nào mà thám mã có thể thực hiện có kết quả phép tấn công chỉ chỉ với bản mã trong thời gian đủ lớn.

Công cụ cơ bản trong nghiên cứu bài toán này là khái niệm entropi. Đây là khái niệm trong lý thuyết thông tin do Shannon đưu ra vào năm 1948. Có thể coi entropi là đại lượng đo thông tin hay còn gọi là độ bất định. Nó được tính như một hàm phân bố xác suất.

Giả sử ta có một biến ngẫu nhiên X nhận các giá trị trên một tập hữu hạn theo một phân bố xác suất p(X). Thông tin thu nhận được bởi một sự kiện xảy ra tuân theo một phân bố p(X) là gì?. Tương tự, nếu sự kiện còn chưa xảy ra thì cái gì là độ bất định và kết quả?. Đại lượng này được gọi là entropi của X và được kí hiệu là H(X).

Các ý tưởng này có vẻ như khá trìu tượng, bởi vậy ta sẽ xét một ví dụ cụ thể hơn. Giả sử biến ngẫu nhiên X biểu thị phép tung đồng xu. Phân bố xác suất là: p(mặt xấp) = p(mặt ngữa) = 1/2. Có thể nói rằng, thông tin (hay entropi) của phép tung đồng xu là một bit vì ta có thể mã hoá mặt xấp bằng 1 và mặt ngữa bằng 0. Tương tự entropi của n phép tung đồng tiền có thể mã hoá bằng một xâu bít có độ dài n.

Xét một ví dụ phức tạp hơn một chút. Giả sử ta có một biến ngẫu nhiên X có 3 giá trị có thể là x_1, x_2, x_3 với xác suất tương ứng bằng 1/2, 1/4, 1/4. Cách mã hiệu quả nhất của 3 biến cố này là mã hoá x_1 là 0, mã của x_2 là 10 và mã của x_3 là 11. Khi đó số bít trung bình trong phép mã hoá này là:

$$1/2 \times 1 + 1/4 \times 2 + 1/4 \times 2 = 3/2.$$

Các ví dụ trên cho thấy rằng, một biến cố xảy ra với xác suất 2^{-n} có thể mã hoá được bằng một xâu bít có độ dài n. Tổng quát hơn, có thể coi rằng, một biến cố xảy ra với xác suất p có thể mã hoá bằng một xâu bít có độ dài xấp xỉ $-\log_2 p$. Nếu cho trước phân bố xác suất tuỳ ý $p_1, p_2, \ldots, p_n$ của biến ngẫu nhiên X, khi đó độ đo thông tin là trọng số trung bình của các lượng $-\log_2 p_i$. Điều này dẫn tới định nghĩa hình thức hoá sau.

Định nghĩa 2.3

Giả sử X là một biến ngẫu nhiên lấy các giá trị trên một tập hữu hạn theo phân bố xác suất p(X). Khi đó entropy của phân bố xác suất này được định nghĩa là lượng:

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i$$

Nếu các giá trị có thể của X là $x_i, 1 \le i \le n$ thì ta có:

$$H(X) = -\sum_{i=1}^{n} p(X=x_i) \log_2 p(X= x_i)$$

Nhận xét

Nhận thấy rằng, $\log_2 p_i$ không xác định nếu $p_i = 0$. Bởi vậy đôi khi entropy được định nghĩa là tổng tương ứng trên tất cả các xác suất khác 0. Vì $\lim_{x\to 0} x\log_2 x = 0$ nên trên thực tế cũng không có trở ngại gì nếu cho $p_i = 0$ với giá trị i nào đó. Tuy nhiên ta sẽ tuân theo giả định là khi tính entropy của một phân bố xác suất p_i, tổng trên sẽ được lấy trên các chỉ số i sao cho $p_i \neq 0$. Ta cũng thấy rằng việc chọn cơ số của logarit là tuỳ ý; cơ số này không nhất thiết phải là 2. Một cơ số khác sẽ chỉ làm thay đổi giá trị của entropy đi một hằng số.

Chú ý rằng, nếu $p_i = 1/n$ với $1 \le i \le n$ thì $H(X) = \log_2 n$. Cũng dễ dàng thấy rằng $H(X) \ge 0$ và $H(X) = 0$ khi và chỉ khi $p_i = 1$ với một giá trị i nào đó và $p_j = 0$ với mọi $j \neq i$.

Xét entropy của các thành phần khác nhau của một hệ mật. Ta có thể coi khoá là một biến ngẫu nhiên K nhận các giá trị tuân theo phân bố xác suất p_K và bởi vậy có thể tính được H(K). Tương tự ta có thể tính các entropy H(P) và H(C) theo các phân bố xác suất tương ứng của bản mã và bản rõ.

Ví dụ 2.1: (tiếp)

Ta có:
$$\begin{aligned} H(P) &= -1/4\log_2 1/4 - 3/4\log_2 3/4 \\ &= -1/4(-2) - 3/4(\log_2 3 - 2) \\ &= 2 - 3/4\log_2 3 \\ &\approx 0{,}81 \end{aligned}$$

bằng các tính toán tương tự, ta có H(K) = 1,5 và H(C) ≈1,85.

2.2.1. Mã huffman và entropy

Trong phần này ta sẽ thảo luận sơ qua về quan hệ giữa entropy và mã Huffman. Vì các kết quả trong phần này không liên quan đến các ứng dụng trong mật mã của entropy nên ta có thể bỏ qua mà không làm mất tính liên tục. Tuy nhiên các hệ quả ở đây có thể dùng để nghiên cứu sâu hơn về khái niệm entropy.

Ở trên đã đưa ra entropy trong bối cảnh mã hoá các biến cố ngẫu nhiên xảy ra theo một phân bố xác suất đã định. Trước tiên ta chính xác hoá thêm những ý tưởng này. Cũng như trên, coi X là biến ngẫu nhiên nhận các giá trị trên một tập hữu hạn và p(X) là phân bố xác suất tương ứng.

Một phép mã hoá X là một ánh xạ bất kỳ:
$$f: X \to \{0,1\}^*$$

trong đó $\{0,1\}^*$ kí hiệu tập tất cả các xâu hữu hạn các số 0 và 1. Với một danh sách hữu hạn (hoặc một xâu) các biến cố $x_1, x_2, \ldots, x_n$, ta có thể mở rộng phép mã hoá f nhờ sử dụng định nghĩa sau:

$$f(x_1x_2...x_n) = f(x_1) || ... || f(x_n)$$

trong đó || kí hiệu phép ghép. Khi đó có thể coi f là ánh xạ:

$$f:X^* \rightarrow \{0,1\}^*$$

Bây giờ giả sử xâu $x_1...x_n$ được tạo từ một nguồn không nhớ sao cho mỗi x_i xảy ra đều tuân theo phân bố xác suất trên X. Điều đó nghĩa là xác suất của một xâu bất kì $x_1...x_n$ được tính bằng $p(x_1) \times ... \times p(x_n)$ (Để ý rằng xâu này không nhất thiết phải gồm các giá trị phân biệt vì nguồn là không nhớ). Ta có thể coi dãy n phép tung đồng xu là một ví dụ.

Bây giờ giả sử ta chuẩn bị dùng ánh xạ f để mã hoá các xâu. Điều quan trọng ở đây là giải mã được theo một cách duy nhất. Bởi vậy phép mã f nhất thiết phải là một đơn ánh.

Ví dụ 2.2.

Giả sử X = {a,b,c,d} , xét 3 phép mã hoá sau:

f(a) = 1	f(b) = 10	f(c) = 100	f(d) = 1000
g(a) = 0	g(b) = 10	g(c) = 110	g(d) = 111
h(a) = 0	h(b) = 01	h(c) = 10	h(d) = 11

Có thể thấy rằng, f và g là các phép mã đơn ánh, còn h không phải là một đơn ánh. Một phép mã hoá bất kỳ dùng f có thể được giải mã bằng cách bắt đầu ở điểm cuối và giải mã ngược trở lại: Mỗi lần gặp số một ta sẽ biết vị trí kết thúc của phần tử hiện thời.

Phép mã dùng g có thể được giải mã bằng cách bắt đầu ở điểm đầu và xử lý liên tiếp. Tại thời điểm bất kì mà ở đó có một dãy con là các kí tự mã của a ,b,c hoặc d thì có thể giải mã nó và có thể cắt ra khỏi dãy con. Ví dụ, với xâu10101110, ta sẽ giải mã 10 là b, tiếp theo 10 là b, rồi đến 111 là d và cuối cùng 0 là a. Bởi vậy xâu đã giải mã là bbda.

Để thấy rằng h không phải là một đơn ánh, chỉ cần xét ví dụ sau:

$$h(ac) = h(bc) = 010$$

Theo quan điểm dễ dàng giải mã, phép mã g tốt hơn f. Sở dĩ như vậy vì nếu dùng g thì việc giải mã có thể được làm liên tiếp từ đầu đến cuối và bởi vậy không cần phải có bộ nhớ. Tính chất cho phép giải mã liên tiếp đơn giản của g được gọi là tính chất tiền tố độclập (một phép mã g được gọi là có tiền

tố độc lập nếu không tồn tại 2 phần tử x,y $\in$ X và một xâu z $\in \{0,1\}^*$ sao cho g(x) = g(y) || z).

Thảo luận ở trên không liên hệ gì đến entropy. Tuy nhiên không có gì đáng ngạc nhiên khi entropy lại có liên quan đến tính hiệu quả của phép mã. Ta sẽ đo tính hiệu quả của phép mã f như đã làm ở trên: đó là độ dài trung bình trọng số (được kí hiệu là *l (f)*) của phép mã một phần tử của X. Bởi vậy ta có định nghĩa sau:

$$l(f) = \sum_{x \in X} p(x) \mid f(x) \mid$$

Trong đó |y| kí hiệu là độ dài của xâu y.

Bây giờ nhiệm vụ chủ yếu của ta là phải tìm một phép mã hoá đơn ánh sao cho tối thiểu hoá được *l(f)* . Thuật toán Huffman là một thuật toán nổi tiếng thực hiện được mục đích này. Hơn nữa, phép mã f tạo bởi thuật toán Huffman là một phép mã có tiền tố độc lập và

$$H(X) \leq l(f) \leq H(X) + 1$$

Như vậy, giá trị của entropy cho ta đánh giá khá chính xác về độ dài trung bình của một phép mã đơn ánh tối ưu.

Ta sẽ không chứng minh các kết quả đã nêu mà chỉ đưa ra một mô tả ngắn gọn hình thức hoá về thuật toán Huffman. Thuật toán Huffman bắt đầu với phân bố xác suất trên tập X và mã mỗi phần tử ban đầu là trống. Trong mỗi bước lặp, 2 phần tử có xác suất thấp nhất sẽ được kết hợp thành một phần tử có xác suất bằng tổng của hai xác suất này. Trong 2 phần tử, phần tử có xác suất nhỏ hơn sẽ được gán giá trị "0", phần tử có giá trị lớn hơn sẽ được gán giá trị "1". Khi chỉ còn lại một phần tử thì mã của x $\in$ X sẽ được cấu trúc bằng dãy các phần tử ngược từ phần tử cuối cùng tới phần tử ban đầu x.

Ta sẽ minh hoạ thuật toán này qua ví dụ sau.

Ví dụ 2.3.

Giả sử X = {a,b,c,d,e} có phân bố xác suất: p(a) = 0,05; p(b) = 0,10; p(c) = 0,12; p(d) = 0,13 và p(e) = 0,60. Thuật toán Huffman được thực hiện như trong bảng sau:

A	b	c	d	e

0,05	0,10	0,12	0,13	0,60
0	1			

0,15		0,12	0,13	0,60
		0	1	

0,15		0,25	0.60
0		1	

0,40			0,60
0			1

1,0

Điều này dẫn đến phép mã hoá sau:

x	*f(x)*
a	000
b	001
c	010
d	011
e	1

Bởi vậy độ dài trung bình của phép mã hoá là:

$$l(\mathrm{f}) = 0{,}05 \times 3 + 0{,}10 \times 3 + 0{,}12 \times 3 + 0{,}13 \times 3 + 0{,}60 \times 1 = 1{,}8$$

So sánh giá trị này với entropy:

$$\begin{aligned} \mathrm{H(X)} &= 0{,}2161 + 0{,}3322 + 0{,}3671 + 0{,}3842 + 0{,}4422 \\ &= 1{,}7402. \end{aligned}$$

2.3. CÁC TÍNH CHẤT CỦA ENTROPI

Trong phần này sẽ chứng minh một số kết quả quan trọng liên quan đến entropi. Trước tiên ta sẽ phát biểu bất đẳng thức Jensen. Đây là

một kết quả cơ bản và rất hữu ích. Bất đẳng thức Jensen có liên quan đến hàm lồi có định nghĩa như sau.

Định nghĩa 2.4.

Một hàm có giá trị thực f là lồi trên khoảng I nếu:

$$f\left(\frac{x+y}{2}\right) \geq \frac{f(x)+f(y)}{2}$$

với mọi x,y $\in$I. f là hàm lồi thực sự trên khoảng I nếu:

$$f\left(\frac{x+y}{2}\right) > \frac{f(x)+f(y)}{2}$$

với mọi x,y $\in I, x \neq y$.

Sau đây ta sẽ phát biểu mà không chứng minh bất đẳng thức Jensen.

Định lý 2.5.(Bất đẳng thức Jensen).

Giả sử h là một hàm lồi thực sự và liên tục trên khoảng l,

$$\sum_{i=1}^{n} a_i = 1$$

và $a_i > 0, 1 \leq i \leq n$. Khi đó:

$$\sum_{i=1}^{n} a_i f(x_i) \leq f\left(\sum_{i=1}^{n} a_i x_i\right)$$

trong đó $x_i \in I, 1 \leq i \leq n$. Ngoài ra dấu "=" chỉ xảy ra khi và chỉ khi $x_1 = \ldots = x_n$.

Bây giờ ta sẽ đưa ra một số kết quả về entropi. Trong định lý sau sẽ sử dụng khẳng định: hàm $\log_2 x$ là một hàm lồi thực sự trong khoảng $(0, \infty)$ (Điều này dễ dàng thấy được từ những tính toán sơ cấp vì đạo hàm cấp 2 của hàm logarith là âm trên khoảng $(0, \infty)$).

Định lý 2.6.

Giả sử X là biến ngẫu nhiên có phân bố xác suất $p_1, p_2, \ldots, p_n$, trong đó $p_i > 0, 1 \leq i \leq n$. Khi đó $H(X) \leq \log_2 n$. Dờu "=" chỉ xảy ra khi và chỉ khi $p_i = 1/n, 1 \leq i \leq n$.

Chứng minh:

Áp dụng bất đẳng thức Jensen, ta có:

$$H(X) = -\sum_{i=1}^{n} p_i \log_2 p_i = \sum_{i=1}^{n} p_i \log_2 (1/p_i)$$

$$\leq \log_2 \sum_{i=1}^{n} (p_i \times 1/p_i)$$

$$= \log_2 n$$

Ngoài ra, dấu "=" chỉ xảy ra khi và chỉ khi $p_i = 1/n$, $1 \leq i \leq n$.

Định lý 2.7.

$$H(X,Y) \leq H(X) + H(Y)$$

Đẳng thức (dấu "=") chỉ xảy ra khi và chỉ khi X và Y là các biến cố độc lập

Chứng minh.

Giả sử X nhận các giá trị x_i, $1 \leq i \leq m$; Y nhận các giá trị y_j, $1 \leq j \leq n$. Kí hiệu: $p_i = p(X = x_i)$, $1 \leq i \leq m$ và $q_j = p(Y = y_j)$, $1 \leq j \leq n$. Kí hiệu $r_{ij} = p(X = x_i, Y = y_j)$, $1 \leq i \leq m$, $1 \leq j \leq n$. (Đây là phân bố xác suất hợp).

Nhận thấy rằng

$$p_i = \sum_{j=1}^{n} r_{ij}$$

($1 \leq i \leq m$) và

$$q_j = \sum_{i=1}^{m} r_{ij}$$

($1 \leq j \leq n$). Ta có

$$H(X) + H(Y) = -(\sum_{i=1}^{m} p_i \log_2 p_i + \sum_{j=1}^{n} q_j \log_2 q_j)$$

$$= -(\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 p_i + \sum_{j=1}^{n}\sum_{i=1}^{m} r_{ij} \log_2 q_j)$$

$$= -\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 p_i q_j$$

Mặt khác

$$H(X,Y) = -\sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 r_{ij}$$

Kết hợp lại ta thu được kết quả sau:

$$H(X,Y) - H(X) - H(Y) = \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 (1/r_{ij}) + \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 p_i q_j$$

$$= \sum_{i=1}^{m}\sum_{j=1}^{n} r_{ij} \log_2 (p_i q_j / r_{ij})$$

$$= \log_2 \sum_{i=1}^{m}\sum_{j=1}^{n} p_i q_j$$

$$= \log_2 1$$

$$= 0$$

(ở đây đã áp dụng bất đẳng thức Jensen khi biết rằng các r_{jj} tạo nên một phân bố xác suất).

Khi đẳng thức xảy ra, có thể thấy rằng phải có một hằng số c sao cho $p_{jj} / r_{jj} = c$ với mọi i,j. Sử dụng đẳng thức sau:

$$\sum_{j=1}^{n}\sum_{i=1}^{m} r_{ij} = \sum_{j=1}^{n}\sum_{i=1}^{m} p_i q_j = 1$$

Điều này dẫn đến c=1. Bởi vậy đâửng thức (dấu "=") sẽ xảy ra khi và chỉ khi $r_{jj} = p_j q_j$, nghĩa là:

$$p(X = x_j, Y = y_j) = p(X = x_j) p(Y = y_j)$$

với $1 \le i \le m$, $1 \le j \le n$. Điều này có nghĩa là X và Y độc lập.

Tiếp theo ta sẽ đưa ra khái niệm entropi có điều kiện

Định nghĩa 2.5.

Giả sử X và Y là hai biến ngẫu nhiên. Khi đó với giá trị xác định bất kỳ y của Y, ta có một phân bố xác suất có điều kiện p(X/y). Rõ ràng là :

$$H(X \mid y) = -\sum_{x} p(x \mid y) \log_2 p(x \mid y)$$

Ta định nghĩa entropi có điều kiện H(X/Y) là trung bình trọng số (ứng với các xác suất p(y) của entropi H(X/y) trên mọi giá trị có thể y. H(X/y) được tính bằng:

$$H(X \mid Y) = -\sum_{y} \sum_{x} p(y) p(x \mid y) \log_2 p(x \mid y)$$

Entropi có điều kiện đo lượng thông tin trung bình về X do Y mang lại.

Sau đây là hai kết quả trực tiếp (Bạn đọc có thể tự chứng minh)

Định lý 2.8.

$$H(X,Y) = H(Y) + H(X/Y)$$

Hệ quả 2.9.

$$H(X/Y) \leq H(X)$$

Dấu bằng chỉ xảy ra khi và chỉ khi X và Y độc lập.

2.4. CÁC KHOÁ GIẢ VÀ KHOẢNG DUY NHẤT

Trong phần này chúng ta sẽ áp dụng các kết quả về entropi ở trên cho các hệ mật. Trước tiên sẽ chỉ ra một quan hệ cơ bản giữa các entropi của các thành phần trong hệ mật. Entropi có điều kiện H(K|C) được gọi là độ bất định về khoá. Nó cho ta biết về lượng thông tin về khoá thu được từ bản mã.

Định lý 2.10.

Giả sử(𝒫, 𝒞, 𝒦, ℰ, 𝒟) là một hệ mật. Khi đó:

$$H(K/C) = H(K) + H(P) - H(C)$$

Chứng minh:

Trước tiên ta thấy rằng H(K,P,C) = H(C | K,P) + H(K,P). Do $y = e_K(x)$ nên khoá và bản rõ sẽ xác định bản mã duy nhất. Điều này có nghĩa là H(C|K,C) = 0. Bởi vậy H(K,P,C) = H(K,P). Nhưng K và P độc lập nên H(K,P) = H(K) + H(P). Vì thế:

$$H(K,P,C) + H(K,P) = H(K) + H(P)$$

Tương tự vì khoá và bản mã xác định duy nhất bản rõ (tức $x = d_K(y)$) nên ta có H(P | K,C) = 0 và bởi vậy H(K,P,C) = H(K,P). Bây giờ ta sẽ tính như sau:

$$H(K \mid C) = H(K,C) - H(C)$$
$$= H(K,P,C) - H(C)$$

$$= H(K) + H(P) - H(C)$$

Đây là nội dung của định lý.

Ta sẽ quay lại ví dụ 2.1 để minh hoạ kết quả này.

Ví dụ 2.1 (tiếp)

Ta đã tính được $H(P) \approx 0,81$, $H(K) = 1,5$ và $H(C) \approx 1,85$. Theo định lý 2.10 ta có $H(K \mid C) \approx 1,5 + 0,85 - 1,85 \approx 0,46$. Có thể kiểm tra lại kết quả này bằng cách áp dụng định nghĩa về entropi có điều kiện như sau. Trước tiên cần phải tính các xác suất xuất $p(K_j \mid j)$, $1 \le i \le 3$, $1 \le j \le 4$. Để thực hiện điều này có thể áp dụng định lý Bayes và nhận được kết quả như sau:

$P(K_1 \mid 1) = 1$	$p(K_2 \mid 1) = 0$	$p(K_3 \mid 1) = 0$
$P(K_1 \mid 2) = 6/7$	$p(K_2 \mid 2) = 1/7$	$p(K_3 \mid 2) = 0$
$P(K_1 \mid 3) = 0$	$p(K_2 \mid 3) = 3/4$	$p(K_3 \mid 3) = 1/4$
$P(K_1 \mid 4) = 0$	$p(K_2 \mid 4) = 0$	$p(K_3 \mid 4) = 1$

Bây giờ ta tính:

$$H(K \mid C) = 1/8 \times 0 + 7/16 \times 0,59 + 1/4 \times 0,81 + 3/16 \times 0 = 0,46$$

Giá trị này bằng giá trị được tính theo định lý 2.10.

Giả sử $(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ là hệ mật đang được sử dụng. Một xâu của bản rõ $x_1x_2 \ldots x_n$ sẽ được mã hoá bằng một khoá để tạo ra bản mã $y_1y_2 \ldots y_n$. Nhớ lại rằng, mục đích cơ bản của thám mã là phải xác định được khoá. Ta xem xét các phương pháp tấn công chỉ với bản mã và coi Oscar có khả năng tính toán vô hạn. Ta cũng giả sử Oscar biết bản rõ là một văn bản theo ngôn ngữ tự nhiên (chẳng hạn văn bản tiếng Anh). Nói chung Oscar có khả năng rút ra một số khoá nhất định (các khoá có thể hay các khoá chấp nhận được) nhưng trong đó chỉ có một khoá đúng, các khoá có thể còn lại (các khoá không đúng) được gọi là các khoá giả.

Ví dụ, giả sử Oscar thu được một xâu bản mã WNAJW mã bằng phương pháp mã dịch vòng. Dễ dàng thấy rằng, chỉ có hai xâu bản rõ có ý nghĩa là *river* và *arena* tương ứng với các khoá F(= 5) và W(= 22). Trong hai khoá này chỉ có một khoá đúng, khoá còn lại là khoá giả. (Trên thực tế, việc tìm một bản mã của MDV có độ dài 5 và 2 bản giải mã có nghĩa không phải quá khó khăn, bạn đọc có thể tìm ra nhiều ví dụ khác). Mục đích của ta là phải tìm ra giới hạn cho số trung bình các khoá giả. Trước tiên, phải xác định giá trị này theo entropi (cho một kí tự) của một ngôn ngữ tự nhiên L (kí hiệu là H_L). H_L là lượng thông tin trung bình trên một kí tự trong một xâu có nghĩa của bản rõ. (Chú ý rằng, một xâu ngẫu nhiên các kí tự của bảng chữ

cái sẽ có entropi trên một kí tự bằng $\log_2 26 \approx 4{,}76$). Ta có thể lấy H(P) là xấp xỉ bậc nhất cho H_L. Trong trường hợp L là Anh ngữ, sử dụng phân bố xác suất trên bảng 1.1, ta tính được $H(P) \approx 4{,}19$.

Dĩ nhiên các kí tự liên tiếp trong một ngôn ngữ không độc lập với nhau và sự tương quan giữa các kí tự liên tiếp sẽ làm giảm entropi. Ví dụ, trong Anh ngữ, chữ Q luôn kéo theo sau là chữ U. Để làm xấp xỉ bậc hai, tính entropi của phân bố xác suất của tất cả các bộ đôi rồi chia cho 2. Một cách tông quát, ta định nghĩa P^n là biến ngẫu nhiên có phân bố xác suất của tất cả các bộ n của bản rõ. Ta sẽ sử dụng tất cả các định nghĩa sau:

Định nghĩa 2.6

Giả sử L là một ngôn ngữ tự nhiên. Entropi của L được xác định là lượng sau:

$$H_L = \lim_{n\to\infty} \frac{H(P^n)}{n}$$

Độ dư của L là: $R_L = l - (H_L / \log_2 |\mathcal{P}|)$

Nhận xét: H_L đo entropi trên mỗi kí tự của ngôn ngữ L. Một ngôn ngữ ngẫu nhiên sẽ có entropi là $\log_2 |\mathcal{P}|$. Bởi vậy đại lượng R_L đo phần "kí tự vượt trội" là phần dư.

Trong trường hợp Anh ngữ, dựa trên bảng chứa một số lớn các bộ đôi và các tần số, ta có thể tính được $H(P^2)$. Ước lượng theo cách này, ta tính được $H(P^2) \approx 3{,}90$. Cứ tiếp tục như vậy bằng cách lập bảng các bộ ba .v.v... ta thu được ước lượng cho H_L. Trên thực tế, bằng nhiều thực nghiệm khác nhau, ta có thể đi tới kết quả sau $1{,}0 \le H_L \le 1{,}5$. Tức là lượng thông tin trung bình trong tiếng Anh vào khoảng 1 bít tới 1,5 bít trên mỗi kí tự!.

Giả sử lấy 1,25 là giá trị ước lượng của giá trị của H_L. Khi đó độ dư vào khoảng 0,75. Tức là tiếng Anh có độ dư vào khoảng 75%! (Điều này không có nghĩa loại bỏ tuỳ ý 3 trên 4 kíb tự của một văn bản tiếng Anh mà vẫn có khả năng đọc được nó. Nó chỉ có nghĩa là tìm được một phép mã Huffman cho các bộ n với n đủ lớn, phép mã này sẽ nén văn bản tiếng Anh xuống còn 1/4 độ dài của bản gốc).

Với các phân bố xác suất đã cho trên $\mathcal{K}$ và $\mathcal{P}^n$. Có thể xác định phân bố xác suất trên $\mathcal{C}^n$ là tập các bộ n của bản mã. (Ta đã làm điều này trong trường

hợp n =1). Ta đã xác định P^n là biến ngẫu nhiên biểu diễn bộ n của bản rõ. Tương tự C^n là biến ngẫu nhiên biểu thị bộ n của bản mã.

Với $y \in C^n$, định nghĩa:

$$K(y) = \{ K \in \mathcal{K}: \exists x \in \mathcal{P}^n, p_{\mathcal{P}^n}(x) > 0, e_K(x) = y\}$$

nghĩa là K(y) là tập các khoá K sao cho y là bản mã của một xâu bản rõ độ dài n có nghĩa, tức là tập các khoá "có thể" với y là bản mã đã cho. Nếu y là dãy quan sát được của bản mã thì số khoá giả sẽ là $| K(y) | -1$ vì chỉ có một khoá là khoá đúng trong số các khoá có thể. Số trung bình các khoá giả (trên tất cả các xâu bản mã có thể độ dài n) được kí hiệu là s_n và nó được tính như sau:

$$\begin{aligned}\bar{s}_n &= \sum_{y \in C^n} p(y)(| K(y) | -1) \\ &= \sum_{y \in C^n} p(y) | K(y) | - \sum_{y \in C^n} p(y) \\ &= \sum_{y \in C^n} p(y) | K(y) | -1\end{aligned}$$

Từ định lý 2.10 ta có:

$$H(K| C^n) = H(K) + H(P^n) - H(C^n).$$

Có thể dùng ước lượng sau:

$$H(P^n) \approx nH_L = n(1 - R_L)\log_2 | \mathcal{P} |$$

với điều kiện n đủ lớn. Hiển nhiên là:

$$H(C^n) \leq n\log_2 | C |.$$

Khi đó nếu $| \mathcal{P} | = | C |$ thì:

$$H(K| C^n) \geq H(K) - nR_L \log_2 | \mathcal{P} | \qquad (2.1)$$

Tiếp theo xét quan hệ của lượng $H(K | C^n)$ với số khoá giả s_n. Ta có:

$$\begin{aligned}H(K | C^n) &= \sum_{y \in C^n} p(y) H(K | y) \\ &\leq \sum_{y \in C^n} p(y) \log_2 | K(y) | \\ &\leq \log_2 \sum_{y \in C^n} p(y) | K(y) | \\ &= \log_2 (\bar{s}_n + 1)\end{aligned}$$

ở đây ta áp dụng bất đẳng thức Jensen (định lý 2.5) với $f(x) = \log_2 x$. Bởi vậy ta có bất đẳng thức sau:

$$H(K | C^n) \leq \log_2 (\bar{s}_n + 1) \qquad (2.2)$$

Kết hợp hai bất đẳng thức (2.1) và (2.2), ta có :

$$\log_2 (\bar{s}_n + 1) \geq H(K) - nR_L \log_2 | P |$$

Trong trường hợp các khoá được chọn đồng xác suất (Khi đó H(K) có giá trị lớn nhất) ta có kết quả sau.

Định lý 2.11

Giả sử ($\mathcal{P}$, $\mathcal{C}$, $\mathcal{K}$, $\mathcal{E}$, $\mathcal{D}$) là một hệ mật trong đó | C | = | P | và các khoá được chọn đồng xác suất. Giả sử R_L là độ dư của ngôn ngữ gốc. Khi đó với một xâu bản mã độ dài n cho trước (n là số đủ lớn), số trung bình các khoá giả s_n thoả mãn bất đẳng thức như sau:

$$\overline{s_n} \geq \{| K | /(| P | nR_L)\} - 1$$

Lượng |K| / |P|nR_L-1 tiến tới 0 theo hàm mũ khi n tăng. Ước lượng này có thể không chính xác với các giá trị n nhỏ. Đó là do $H(P^n)$/ n không phải là một ước lượng tốt cho H_L nếu n nhỏ.

Ta đưa ra đây một khái niệm nữa

Định nghĩa 2.7.

Khoảng duy nhất của một hệ mật được định nghĩa là giá trị của n mà ứng với giá trị này, số khoá giả trung bình bằng 0 (kí hiệu giá trị này là n_0). Điều đó có nghĩa là n_0 là độ dài trung bình cần thiết của bản mã để thám mã có thể tính toán khoá một cách duy nhất với thời gian đủ lớn.

Nếu đặt s_n =0 trong định lý 2.11 và giải theo n ta sẽ nhận được ước lượng cho khoảng duy nhất:

$$n_0 \approx \log_2|\mathcal{K}| / R_L \log_2 |\mathcal{P}|$$

Ví dụ với MTT, ta có $|\mathcal{P}| = 26$ và $|\mathcal{K}| = 26!$. Nếu lấy R_L =0,75 thì ta nhận được ước lượng cho khoảng duy nhất bằng:

$$n_0 \approx 88{,}4/ (0{,}75 \times 4{,}7) \approx 25$$

Điều đó có nghĩa là thông thường nếu mã thám có được xâu bản mã với độ dài tối thiểu là 25, anh ta có thể nhận được bản giải mã duy nhất.

2.5. CÁC HỆ MẬT MÃ TÍCH

Một phát minh khác do Shannon đưa ra trong bài báo của mình năm 1949 là ý tưởng kết hợp các hệ mật bằng cách tạo tích của chúng. Ý tưởng này có tầm quan trọng to lớn trong việc thiết kế các hệ mật hiện nay (chẳng hạn chuẩn mã dữ liệu -DES).

Để đơn giản, trong phần này chỉ hạn chế xét các hệ mật trong đó $\mathcal{C}=\mathcal{P}$: các hệ mật loại này được gọi là tự đồng cấu. Giả sử $S_1= (\mathcal{P}, \mathcal{P}, \mathcal{K}_1, \mathcal{E}_1, \mathcal{D}_1)$ và $S_2= (\mathcal{P}, \mathcal{P}, \mathcal{K}_2, \mathcal{E}_2, \mathcal{D}_2)$ là hai hệ mật tự đồng cấu có cùng các không gian bản mã và rõ. Khi đó, tích của S_1 và S_2 (kí hiệu là $S_1 \times S_2$) được xác định là hệ mật sau:

$$(\mathcal{P}, \mathcal{P}, \mathcal{K}_1 \times \mathcal{K}_2, \mathcal{E}, \mathcal{D})$$

Khoá của hệ mật tích có dạng $K = (K_1,K_2)$ trong đó $K_1 \in \mathcal{K}_1$ và $K_2 \in \mathcal{K}_2$. Các quy tắc mã và giải mã của hệ mật tích được xác định như sau: Với mỗi K = (K1,K2), ta có một quy tắc mã E_K xác định theo công thức:

$$e_{(K_1,K_2)}(x) = e_{K_2}(e_{K_1}(x))$$

và quy tắc giải mã:

$$d_{(K_1,K_2)}(y) = d_{K_1}(d_{K_2}(y))$$

Nghĩa là trước tiên ta mã hoá x bằng e_{K1} rồi mã lại bản kết quả bằng e_{K2}. Quá trình giải mã tương tự nhưng thực hiện theo thứ tự ngược lại:

$$\begin{aligned} & d_{(K_1,K_2)}(e_{(K_1,K_2)}(x) = d_{(K_1,K_2)}(e_{K_2}(e_{K_1}(x))) \\ & = d_{K_1}(d_{K_2}(e_{K_2}(e_{K_1}(x))) \\ & = d_{K_1}(e_{K_1}(x))) \\ & = x. \end{aligned}$$

Ta biết rằng, các hệ mật đều có các phân bố xác suất ứng với các không gian khoá của chúng. Bởi vậy, cần phải xác định phân bố xác suất cho không gian khoá $\mathcal{K}$ của hệ mật tích. Hiển nhiên ta có thể viết:

$$p_{\mathcal{K}}(K_1,K_2)= p_{\mathcal{K}1}(K_1) \times p_{\mathcal{K}2}=(K_2)$$

Nói một cách khác, ta chọn K_1 có phân bố $p_{\mathcal{K}1}$ rồi chọn một cách độc lập K_2 có phân bố $p_{\mathcal{K}2}(K_2)$.

Sau đây là một ví dụ đơn giản để minh hoạ khái niệm hệ mật tích. Giả sử định nghĩa hệ mật mã nhân như trong hình 2.2 sau.

Hình 2.2. Mã nhân

Giả sử $\mathcal{P} = \mathcal{C} = Z_{26}$ và giả sử:

$$K = \{a \ Z_{26}: UCLN(a,26) = 1\}$$

Với $a \in \mathcal{K}$, ta xác định: $e_a(x) = ax \bmod 26$

và $d_a(y) = a^{-1}y \bmod 26$

$(x,y) \in Z_{26}$.

Cho M là một hệ mã nhân (Với các khoá được chọn đồng xác suất) và S là MDV (với các khoá chọn đồng xác suất). Khi đó dễ dàng thấy rằng M×S chính là hệ mã Affine (cùng với các khoá được chọn đồng xác suất). Tuy nhiên việc chướng tỏ S ×M cũng là hệ mã Affine khó hơn một chút (cũng với các khóa đồng xác suất).

Ta sẽ chứng minh các khẳng định này. Một khoá dịch vòng là phần tử $K \in Z_{26}$ và quy tắc giải mã tương ứng là $e_K(x) = x + K \bmod 26$. Còn khoá trong hệ mã nhân là phần tử $a \in Z_{26}$ sao cho UCLN(a,26) = 1. Quy tắc mã tương ứng là $e_a(x) = a \bmod 26$. Bởi vậy, một khoá trong mã tích M × S có dạng (a,K), trong đó

$$e_{(a,K)}(x) = a\,x + K \bmod 26$$

Đây chính là định nghĩa về khoá trong hệ mã Affine. Hơn nữa, xác suất của một khoá trong hệ mã Affine là:1/312 = 1/12 × 1/26. Đó là tích của xác suất tương ứng của các khoá a và K. Bởi vậy M ×S là hệ mã A ffine.

Bây giờ ta sẽ xét S ×M. Một khoá này trong hệ mã này có dạng (K ,a) trong đó:

$$e_{(K,a)}(x) = a(x+K) = a\,x + aK \bmod 26$$

Như vậy khoá (K,a) của mã tích S×M đồng nhất với khoá (a, aK) của hệ mã Affine. Vấn đề còn lại là phải chứng tỏ rằng mỗi khoá của mã Affine xuất hiện với cùng xác suất 1/312 như trong mã tích S×M. Nhận thấy rằng, aK = K_1 khi và chỉ khi $K = a^{-1}K_1$, (hãy nhớ lại rằng UCLN(a,26) =1, bởi vậy a có phần tử nghịch đảo). Nói cách khác, khoá (a, K_1) của hệ mã Affine tương đương với khoá ($a^{-1}K_1$,a) của mã tích S×M. Bởi vậy, ta có một song ánh giữa hai không gian khoá. Vì mỗi khoá là đồng xác suất nên có thể thấy rằng S×M thực sự là mã Affine.

Ta chứng minh rằng M ×S = S × M. Bởi vậy, hai hệ mật là giao hoán. Tuy nhiên không phải mọi cặp hệ mật đều giao hoán; có thể tìm ta được các cặp phản ví dụ, Mặt khác ta thấy rằng phép tích luôn kết hợp:

$$(S_1 \times S_2) \times S_3 = S_1 \times (S_2 \times S_3)$$

Nếu lấy tích của một hệ mật tự đồng cấu với chính nó thì ta thu được hệ mật S×S (kí hiệu là S^2). Nếu lấy tích n lần thì hệ mật kết quả là S^n. Ta gọi S^n là hệ mật lặp.

Một hệ mật S được gọi là luỹ đẳng nếu $S^2 = S$. Có nhiều hệ mật đã nghiên cứu trong chương 1 là hê mật luỹ đẳng. Chẳng hạn các hệ MDV, MTT, Affine, Hill, Vigenère và hoán vị đều là luỹ đẳng. Hiển nhiên là nếu

hệ mật S là luỹ đẳng thì không nên sử dụng hệ mâth tích S^2 vì nó yêu cầu lượng khoá cực lớn mà không có độ bảo mật cao hơn.

Nếu một hệ mật không phải là luỹ đẳng thì có khả năng làm tăng độ mật bằng cách lặp nhiều lần. Ý tưởng này đã được dùng trong chuẩn mã dữ liệu (DES). Trong DES dùng 16 phép lặp, tất nhiên hệ mật ban đầu phải là hệ mật không luỹ đẳng. Một phương pháp có thể xây dựng các hệ mật không luỹ đẳng đơn giản là lấy tích của hai hệ mật đơn giản khác nhau.

Nhận xét:

Có thể dễ dàng chứng tỏ rằng, nếu cả hai hệ mật S_1 và S_2 là luỹ đẳng và giao hoán thì S_1 và S_2 cũng là luỹ đẳng. Điều này rút ra từ các phép toán đại số sau:

$$\begin{aligned}(S_1 \times S_2) \times (S_1 \times S_2) &= S_1 \times (S_2 \times S_1) \times S_2 \\ &= S_1 \times (S_1 \times S_2) \times S_2 \\ &= (S_1 \times S_1) \times (S_2 \times S_2) \\ &= S_1 \times S_2.\end{aligned}$$

(Chú ý dùng tính chất kết hợp trong chứng minh trên)

Bởi vậy, nếu cả S_1 và S_2 đều là luỹ đẳng và ta muốn $S_1 \times S_2$ là không luỹ đẳng thì điều kiện cần là S_1 và S_2 không giao hoán.

Rất may mắn là nhiều hệ mật đơn giản thoả mãn điều kiện trên. Kỹ thuật thường được sử dụng trong thực tế là lấy tích các hệ mã kiểu thay thế và các hệ mã kiểu hoán vị. Trong chương sau ta sẽ xét một thể hiện cụ thể của kỹ thuật này.

2.5. CÁC CHÚ GIẢI.

Khái niệm độ mật hoàn thiện và việc sử dụng các kỹ thuật entropi trong các hệ mật lần đầu tiên do Shannon đưa ra trong [SH49]. Các hệ mật tích cũng được thảo luận trong bài báo này. Khái niệm entropi cũng do Shannon đưa ra trong [SH48]. Các sách nhập môn tốt về entropi, mã Huffman và các vấn đề có liên quan có trong các tài liệu của Welsh [WE88] và Goldie, Pinch [GP91]. Các kết quả trong phần 2.4 được lấy theo Beauchemin và Brassard [BB88], các tác giả này đã tổng quát hoá các kết quả ban đầu của Shannon.

BÀI TẬP

2.1. Cho n là một số nguyên dương. Một hình vuông Latin cấp n (L) là một bảng n × n các số nguyên 1, . . . , n sao cho mỗi một số trong n số nguyên này chỉ xuất hiện đúng một lần ở mỗi hàng và mỗi cột của L. Ví dụ hình vuông Latin cấp 3 có dạng:

1	2	3
3	1	2
2	3	1

Với một hình vuông Latin L bất kỳ cấp n, ta có thể xác định một hệ mã tương ứng. Giả sử P = C = K = { 1, . . ., n}. Với $1 \le i \le n$, quy tắc mã hoá e_i được xác định là $e_i(j) = L(i,j)$. Do đó mỗi hàng của L sẽ cho một quy tắc mã hoá).

Hãy chứng minh rằng, hệ mật hình vuông Latin này có độ mật hoàn thiện.

2.2. Hãy chứng tỏ rằng mã Affine có độ mật hoàn thiện

2.3. Giả sử một hệ mật đạt được độ mật hoàn thiện với phân bố xác suất p_0 nào đó của bản rõ. Hãy chứng tỏ rằng độ mật hoàn thiện vẫn còn dữ được đối với một phân bố xác suất bất kì của bản rõ.

2.4. Hãy chứng tỏ rằng nếu một hệ mật có độ mật hoàn thiên và $|\mathcal{K}| = |C| = |\mathcal{P}|$ thì mọi bản mã là đồng xác suất.

2.5. Giả sử X là tập có lực lượng n, trong đó $2^k \le n \le 2^{k+1}$ và p(x) =1/n với mọi $x \in X$.

a/ Hãy tìm một phép mã hoá có tiền tố độc lập của X (kí hiệu là f) sao cho $l(f) = k+2 - 2^{k+1}/n$

Chỉ dẫn: Hãy mã hoá 2^{k+1}-n các phần tử của X bằng các xâu có độ dài k và mã hoá các phần tử còn lại bằng các xâu có độ dài k+1.

b/ Hãy minh hoạ cấu trúc của bạn khi n = 6. Tính $l(f)$ và H(X) trong trường hợp này.

2.6. Giả sử X = {a,b,c,d,e} có phân bố xác suất như sau: p(a) = 0,32, p(b) = 0,23 p(c) = 0,20, p(d) = 0,15, p(e) = 0,10. Hãy dùng thuật toán Huffman để tìm phép mã hoá tối ưu có tiền tố độc lập của X. So sánh độ dài của phép mã này với H(X).

2.7. Hãy chứng tỏ rằng H(X,Y) = H(Y) +H(X|Y). Sau đó chứng minh bổ đề là $H(X|Y) \le H(X)$, đẳng thức chỉ xảy ra khi và chỉ khi X và Y độc lập.

2.8. Chứng minh rằng, một hệ mật có độ mật hoàn thiện khi và chỉ khi H(P|C) = H(P).

2.9. Chứng minh rằng trong một hệ mật bất kỳ H(K|C) ≥H(P C) (về mặt trực giác, kết quả này nói rằng với bản mã cho trước, độ bất định của thám mã về khoá ít nhất cũng lớn bằng độ bất định khi thám mã bản rõ).
2.10. Xét một hệ mật trông đó $\mathcal{P}$= {a,b,c}, $\mathcal{K}$= {K_1,K_2,K_3} và $\mathcal{C}$ = {1,2,3,4}. Giả sử ma trận mã hoá như sau:

	a	B	c
K_1	1	2	3
K_2	2	3	4
K_3	3	4	1

Giả sử các khoá được chọn đồng xác suất và phần bố xác suất của bản rõ là $p_{\mathcal{P}}$(a) = 1/2, $p_{\mathcal{P}}$(b) = 1/3, $p_{\mathcal{P}}$(c) = 1/6. Hãy tính H(P), H(C), H(K), H(K|C) và H(P|C).
2.11. Hãy tính H(K|C) và H(K|P,C) của hệ mã Affine.
2.12. Xét hệ mã Vigenère có độ dài từ khoá là m. Hãy chứng tỏ khoảng duy nhất là 1/R_L, trong đó R_L là độ dư của ngôn ngữ đang xét. (kết quả này được hiểu như sau: Nếu n_0 là số kí tự cần mã hoá thì độ dài của bản rõ là n_0/m vì mỗi phần tử của bản rõ gồm m kí tư. Bởi vậy, khoảng duy nhất 1/R_L ứng với một bản rõ gồm m/R_L kí tự).
2.13. Hãy chỉ ra rằng, khoảng duy nhất của hệ mã Hill (với ma trận mã hoá m×m) là nhỏ hơn m/R_L (hãy chú ý rằng số kí tự trong môt bản rõ có độ dài là m^2/R_L).
2.14. MTT trên không gian rõ (có kích thước n) sẽ có |$\mathcal{K}$| = n!. Công thức Stirling cho ước lượng sau đối với n:

$$n \approx \sqrt{2\pi m}(n/e)^n$$

a/ Dùng công thức Stirling, đưa ra một khoảng ước lượng cho khoảng duy nhất của MTT.

b/ Cho m ≥1 là một số nguyên. MTT bộ m là hệ mã thay thế trong đó các không gian rõ (và bản mã) chứa tất cả 26^m các bộ m. Hãy đánh giá khoảng duy nhất của MTT bộ m nếu R_L = 0,75.
2.15. Hãy chứng minh rằng MDV là luỹ đẳng.
2.16. Giả sử S_1 là MDV (với các khoá đồng xác suất) và S_2 là MDV trong đó các khoá được chọn theo một phân bố xác suất $p_{\mathcal{K}}$ nào đó (không đồng xác suất). Hãy chứng tỏ rằng S_1×S_2 = S_1.
2.17. Giả sử S_1 và S_2 là các hệ mã Vigenère có độ dài từ khoá tương ứng là m_1 và m_2 trong đó $m_1 \geq m_2$.

a/ Nếu m_1 | m_2 thì chỉ ra rằng $S_1 \times S_2 = S_1$.

b/ Ta thử tổng quát hoá kết quả trên bằng giả định rằng $S_2 \times S_1 = S_3$, S_3 là hệ mã Vigenère có độ dài từ khoá là BCNN(m_1,m_2) (BCNN - bội chung nhỏ nhất). Hãy chứng tỏ rằng giả định này là không đúng.

Chỉ dẫn: Nếu $m_1 \neq 0 \bmod m_2$ thì số các khoá trong hệ mã tích $S_1 \times S_2$ nhỏ hơn số khoá trong S_3.

CHƯƠNG 3

CHUẨN MÃ DỮ LIỆU

3.1. MỞ ĐẦU.

Ngày 15.5.1973. Uỷ ban tiêu chuẩn quốc gia Mỹ đã công bố một khuyến nghị cho các hệ mật trong Hồ sơ quản lý liên bang. Điều này cuối cùng đã dẫn đến sự phát triển của Chuẩn mã dữ liệu (DES) và nó đã trở thành một hệ mật được sử dụng rộng rãi nhất trên thế giới. DES được IBM phát triển và được xem như một cải biên cuả hệ mật LUCIPHER. Lần đầu tiên DES được công bố trong Hồ sơ Liên bang vào ngày 17.3.1975. Sau nhiều cuộc trành luận công khai, DES đã được chấp nhận chọn làm chuẩn cho các ứng dụng không được coi là mật vào 5.1.1977. Kể từ đó cứ 5 năm một lần, DES lại được Uỷ ban Tiêu chuẩn Quốc gia xem xét lại. Lần đổi mới gàn đây nhất của DES là vào tháng 1.1994 và tiếp tới sẽ là 1998. Người ta đoán rằng DES sẽ không còn là chuẩn sau 1998.

3.2. MÔ TẢ DES

Mô tả đầy đủ của DES được nêu trong Công bố số 46 về các chuẩn xử lý thông tin Liên bang (Mỹ) vào 15.1.1977. DES mã hoá một xâu bít x của bản rõ độ dài 64 bằng một khoá 54 bít. Bản mã nhậ được cũng là một xâu bít có độ dài 48. Trước hết ta mô tả ở mức cao của hệ thống.

Thuật toán tiến hành theo 3 giai đoạn:

1.Với bản rõ cho trước x, một xâu bít x_0 sẽ được xây dựng bằng cách hoán vị các bít của x theo phép hoán vị cố định ban đầu IP. Ta viết:x_0= IP(X) = L_0R_0, trong đó L_0 gồm 32 bít đầu và R_0 là 32 bít cuối.

2. Sau đó tính toán 16 lần lặp theo một hàm xác định. Ta sẽ tính L_iR_i, $1 \le i \le 16$ theo quy tắc sau:

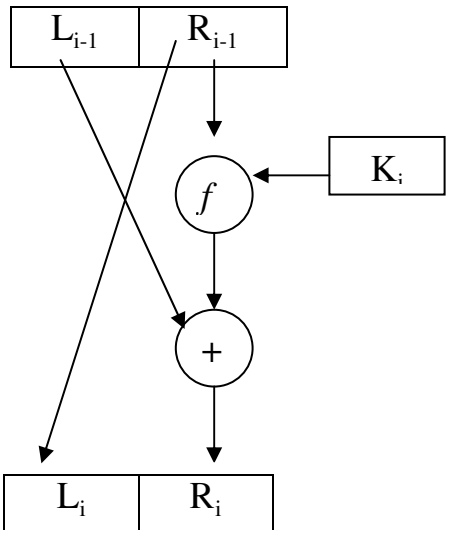
$$L_i = R_{i-1}$$
$$R_i = L_{i-1} \oplus f(R_{i-1},K_i)$$

trong đó $\oplus$ kí hiệu phép hoặc loại trừ của hai xâu bít (cộng theo modulo 2). f là một hàm mà ta sẽ mô tả ở sau, còn $K_1,K_2, \ldots ,K_{16}$ là các xâu bít độ dài 48 được tính như hàm của khoá K. (trên thực tế mỗi K_i là một phép chọn hoán

vị bít trong K). K_1, . . ., K_{16} sẽ tạo thành bảng khoá. Một vòng của phép mã hoá được mô tả trên hình 3.1.

3. Áp dụng phép hoán vị ngược IP^{-1} cho xâu bít $R_{16}L_{16}$, ta thu được bản mã y. Tức là $y=IP^{-1}(R_{16}L_{16})$. Hãy chú ý thứ tự đã đảo của L_{16} và R_{16}.

Hình 3.1. Một vong của DES



Hàm f có hai biến vào: biến thứ nhất A là xâu bít độ dài 32, biến thứ hai J là một xâu bít độ dài 48. Đầu ra của f là một xâu bít độ dài 32. Các bước sau được thực hiện:

1. Biến thứ nhất A được mở rộng thành một xâu bít độ dài 48 theo một hàm mở rộng cố định E. E(A) gồm 32 bít của A (được hoán vị theo cách cố định) với 16 bít xuất hiện hai lần.

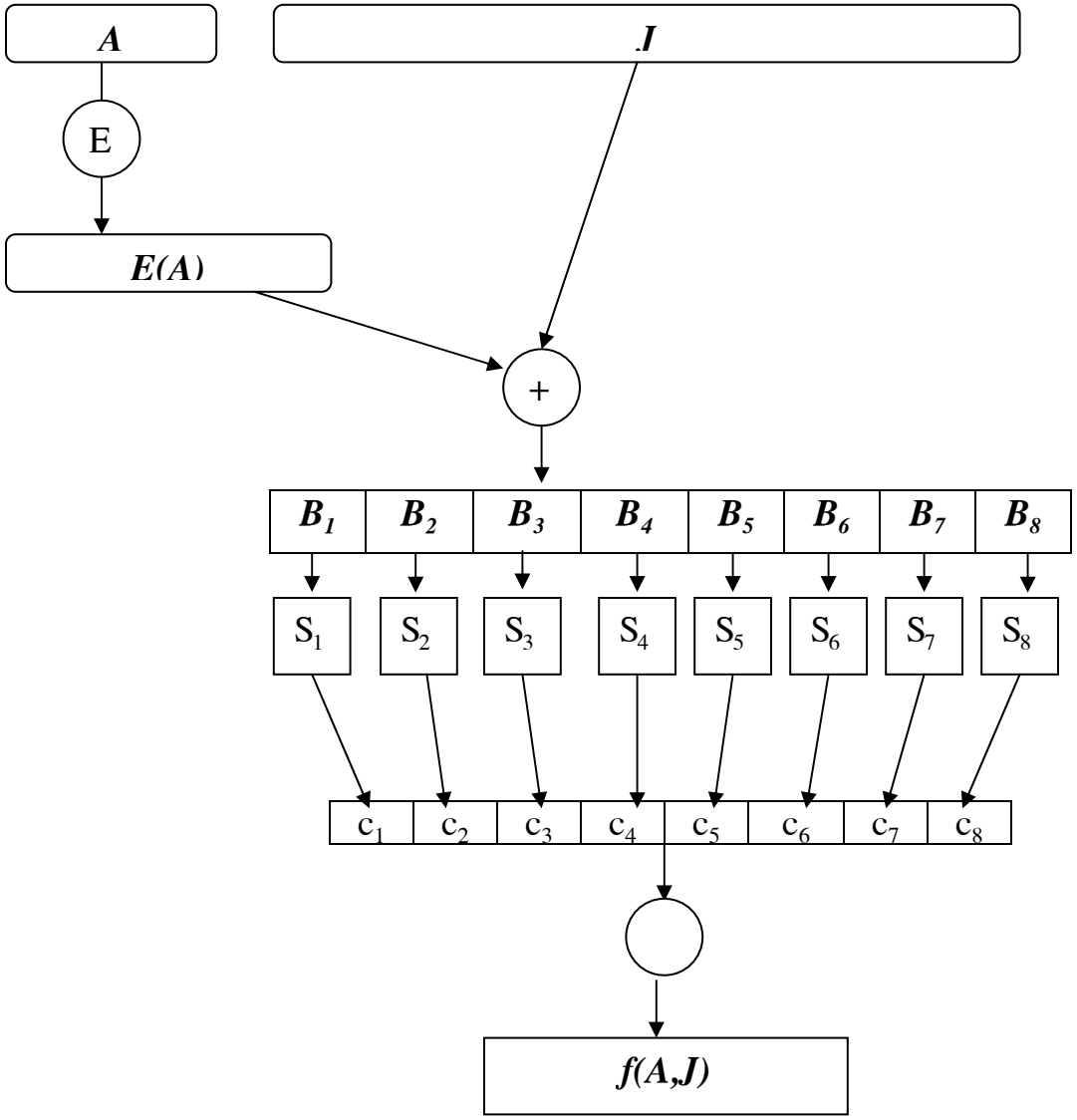
2. Tính $E(A) \oplus J$ và viết kết quả thành một chuỗi 8 xâu 6 bít = $B_1B_2B_3B_4B_5B_6B_7B_8$.

3.Bước tiếp theo dùng 8 bảng S_1, S_2, ... ,S_8 (được gọi là các hộp S). Với mỗi S_i là một bảng 4×16 cố định có các hàng là các số nguyên từ 0 đến 15. Với xâu bít có độ dài 6 (Kí hiệu $B_i = b_1b_2b_3b_4b_5b_6$), ta tính $S_j(B_j)$ như sau: Hai bít b_1b_6 xác định biểu diễn nhị phân của hàng r của S_j ($0 \le r \le 3$) và bốn bít ($b_2b_3b_4b_5$) xác định biểu diễn nhị phân của cột c của S_j ($0 \le c \le 15$). Khi đó $S_j(B_j)$ sẽ xác định phần tử $S_j(r,c)$; phần tử này viết dưới dạng nhị phân là một xâu bít có độ dài 4. (Bởi vậy, mỗi S_j có thể được coi là một hàm mã mà đầu vào là một xâu bít có độ dài 2 và một xâu bít có độ dài 4, còn đầu ra là một xâu bít có độ dài 4). Bằng cách tương tự tính các $C_j = S_j(B_j)$, $1 \le j \le 8$.

4. Xâu bít $C = C_1C_2... C_8$ có độ dài 32 được hoán vị theo phép hoán vị cố định P. Xâu kết quả là P(C) được xác định là f(A,J).

Hàm f được mô tả trong hình 3.2. Chủ yếu nó gômg một phép thế (sử dụng hộp S), tiếp sau đó là phép hoán vị P. 16 phép lặp của f sẽ tạo nên một hệ mật tích nêu như ở phần 2.5.

Hình 3.2. Hàm f của DES



Trong phần còn lại của mục này, ta sẽ mô tả hàm cụ thể được dùng trong DES. Phép hoán vị ban đầu IP như sau:

IP
58 50 42 34 26 18 10 2
60 52 44 36 28 20 12 4
62 54 46 38 31 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

Bảng này có nghĩa là bít thứ 58 của x là bít đầu tiên của IP(x); bít thứ 50 của x là bít thứ hai của IP(x), .v.v . . .

Phép hoán vi ngược IP^{-1} là:

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

Hàm mở rộng E được xác đinh theo bảng sau:

Bảng chọn E bít					
32	1	2	3	4	5
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

Tám hộp S là:

S_1															
14	4	13	1	2	15	11	8	3	10	3	12	5	9	1	7
1	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

S_1															
15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

S_3															
10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
13	6	4	9	8	5	3	0	11	1	2	12	5	10	14	7
1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

S_4															
7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S_5															
2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S_6															
12	1	10	15	9	2	6	8	0	13	3	4	14	7	15	11
10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
4	3	2	12	9	5	15	10	11	14	11	7	6	0	8	13

S_7															
4	11	12	14	15	0	8	13	3	12	9	7	5	10	6	1
13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S_8															
13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Và phép hoán vị P có dạng:

P		
16	7	20
29	12	28
1	15	23
5	18	31
32	27	3
19	13	30
22	11	4

Cuối cung ta cần mô tả việc tính toán bảng khoá từ khoá K. Trên thực tế, K là một xâu bít độ dài 64, trong đó 56 bít là khoá và 8 bít để kiểm tra tính chẵn lẻ nhằm phát hiện sai. Các bít ở các vị trí 8,16, . . ., 64 được xác định sao cho mỗi byte chứa một số lẻ các số "1". Bởi vậy một sai sót đơn lẻ có thể phát hiện được trong mỗi nhóm 8 bít. Các bít kiểm tra bị bỏ qua trong quá trình tính toán bảng khoá.

1. Với một khoá K 64 bít cho trước, ta loại bỏ các bít kiểm tra tính chẵn lẻ và hoán vị các bít còn lại của K theo phép hoán vị cố định PC-1. Ta viết:

$$PC\text{-}1(K) = C_0D_0$$

2. Với i thay đổi từ 1 đến 16:
3.

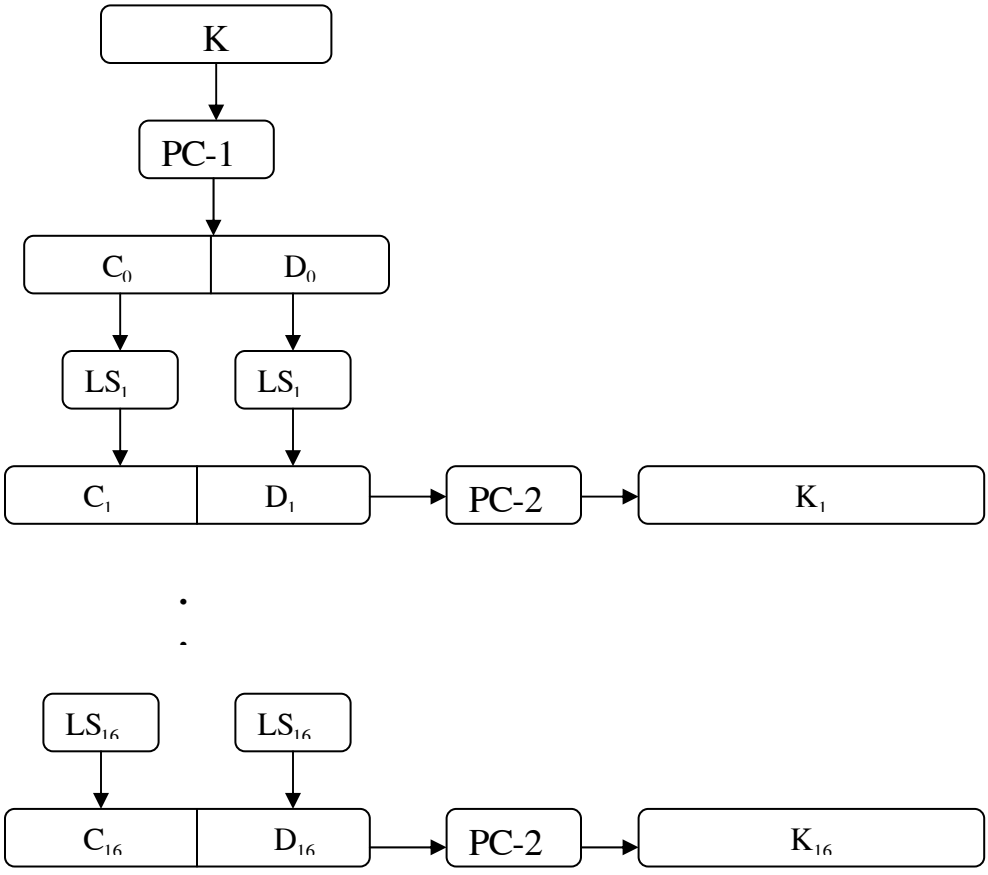
$$C_i = LS_i(C_{i-1})$$
$$D_i = LS_i(Di\text{-}1)$$

Việc tính bảng khoá được mô tả trên hình 3.3

Các hoán vị PC-1 và PC-2 được dùng trong bảng khoá là:

PC-1					
57	49	41	33	25	17
1	58	50	42	34	26
10	2	59	51	43	35
19	11	3	60	52	44
63	55	47	39	31	23
7	62	54	46	38	30
14	6	61	53	45	37
21	13	5	28	20	12

Hình 3.3 Tính bảng khoá DES.



PC-2					
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

Bây giờ ta sẽ đưa ra bảng khoá kết quả. Như đã nói ở trên, mỗi vòng sử dụng một khoá 48 bít gồm 48 bít nằm trong K. Các phần tử trong các bảng dưới đây biểu thị các bít trong K trong các vòng khoá khác nhau.

Vòng 1											
10	51	34	60	49	17	35	57	2	9	19	42
3	35	26	25	44	58	59	1	36	27	18	41
22	28	39	54	37	4	47	30	5	53	23	29
61	21	38	63	15	20	45	14	13	62	55	31

Vòng 2											
2	43	26	52	41	9	25	49	59	1	11	34
60	27	18	17	36	50	51	58	57	19	10	33
14	20	31	46	29	63	39	22	28	45	15	21
53	13	30	55	7	12	37	6	5	54	47	23

Vòng 3											
51	27	10	36	25	58	9	33	43	50	60	18
44	11	2	1	49	34	35	42	41	3	59	17
61	4	15	30	13	47	23	6	12	29	62	5
37	28	14	39	54	63	21	53	20	38	31	7

Vòng 4											
35	11	59	49	9	42	58	17	27	34	44	2
57	60	51	50	33	18	19	26	25	52	43	1
45	55	62	14	28	31	7	53	63	13	46	20
21	12	61	23	38	47	5	37	4	22	15	54

Vòng 5
19 60 43 33 58 26 42 1 11 18 57 51
41 44 35 34 17 2 3 10 9 36 27 50
29 39 46 61 12 15 54 37 47 28 30 4
.5 63 45 7 22 31 20 21 55 6 62 38

Vòng 6
3 44 27 17 42 10 26 50 60 2 41 35
25 57 19 18 1 51 52 59 58 49 11 34
13 23 30 45 63 62 38 21 31 12 14 55
20 47 29 54 6 15 4 5 39 53 46 22

Vòng 7
52 57 11 1 26 59 10 34 44 51 25 19
9 41 3 2 50 35 36 43 42 33 60 18
28 7 14 29 47 46 22 5 15 63 61 39
4 31 13 38 53 62 55 20 23 38 30 6

Vòng 8
36 41 60 50 10 43 59 18 57 35 9 3
58 25 5251 34 19 49 27 26 17 44 2
12 54 61 13 31 30 6 20 62 47 45 23
55 15 28 22 37 46 39 4 721 14 53

Vòng 9
57 33 52 42 2 35 51 10 49 27 1 60
50 17 44 43 26 11 41 19 18 9 36 59
4 46 53 5 23 22 61 12 54 39 37 15
47 7 20 14 29 38 31 63 62 13 6 45

Vòng 10
41 17 36 26 51 19 35 59 33 11 50 44
34 1 57 27 10 60 25 3 2 58 49 43
55 30 37 20 7 6 45 63 38 23 21 62
31 54 4 61 13 22 15 47 46 28 53 29

Vòng 11
25 1 49 10 35 3 19 43 17 60 34 57
18 50 41 11 59 44 9 52 51 42 33 27
39 14 21 4 54 53 29 47 22 7 5 46
15 38 55 45 28 6 62 31 30 12 37 13

Vòng 12
9 50 33 59 19 52 3 27 1 44 18 41
2 34 25 60 43 57 58 36 35 26 17 11
23 61 5 55 38 37 13 31 6 54 20 30
62 22 39 29 12 53 46 15 14 63 21 28

Vòng 13
58 34 17 43 3 36 52 11 50 57 2 25
51 18 9 44 27 41 42 49 19 10 1 60
7 45 20 39 22 21 28 15 53 38 4 14
46 6 23 13 63 37 30 62 61 47 5 12

Vòng 14
42 18 1 27 52 49 36 60 34 41 51 9
35 2 58 57 11 25 26 33 3 59 50 44
54 29 4 23 6 5 12 62 37 22 55 61
30 53 7 28 47 21 14 46 45 31 20 63

Vòng 15
26 2 50 11 36 33 49 44 18 25 35 58
19 51 42 41 60 9 10 17 52 43 34 57
38 13 55 7 53 20 63 46 21 6 39 45
14 37 54 12 31 5 61 30 29 15 4 47

Vòng 16
18 59 42 3 57 25 41 36 10 17 27 50
11 43 34 33 52 1 2 9 44 35 26 49
30 5 47 62 45 12 55 58 13 61 31 37
6 27 46 4 23 28 53 22 21 7 62 39

Phép giải mã được thực hiện nhờ dùng cùng thuật toán như phép mã nếu đầu vào là y nhưng dùng bảng khoá theo thứ tự ngược lại $K_{16},...K_1$. Đầu ra của thuật toán sẽ là bản rõ x.

3.2.1. Một ví dụ về DES.

Sau đây là một ví dụ về phép mã DES. Giả sử ta mã bản rõ (ở dạng mã hexa - hệ đếm 16):

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

Bằng cách dùng khoá

1 2 3 4 5 7 7 9 9 B B C D F F 1

Khoá ở dạng nhị phân (không chứa các bít kiểm tra) là:

0001001001101001010110111100100110110111101101111111000

Sử dụng IP, ta thu được L_0 và R_0 (ở dạng nhị phân) như sau:

L_0 = 11001100000000000110010011111111
$L_1 = R_0$ = 11110000101010101111000010101010

Sau đó thực hiện 16 vòng của phép mã như sau:

$E(R_0)$ = 011110100001010101010101011110100001010101010101
K_1 = 000110110000000101110111111111110001110000011100 10
$E(R_0) \oplus K_1$ = 011000010001011110111010100001100110010100100111
S-box outputs 01011100100000101011010110010111
$f(R_0,K_1)$ = 00100011010010101010100110111011
$L_2 = R_1$ = 11101111010010100110010101000100

$E(R_1)$ = 011101011110101001010100001100001010101000001001
K_2 = 011110011010111011011001110110111100100111100101
$E(R_1) \oplus K_2$ = 000011000100010010001101111010110110001111101100
S-box outputs 11111000110100000011101010101110
$f(R_1,K_2)$ = 00111100101010111000011110100011
$L_3 = R_2$ = 11001100000000001011011100001001

$E(R_2)$ = 11100101100000000000001010111010111010000101001 1
K_3 = 010101011111110010001010010000101100111110011001
$E(R_2) \oplus K_3$ = 101100000111110010001000111110000010011111001010
S-box outputs 00100111000100001110000101101111
$f(R_2,K_3)$ = 01001101000101100110111010110000
$L_4 = R_3$ = 10100010010111000000101111110100

$E(R_3)$ =010100000100001011111000000001010111111111010100
K_4 = 011100101010110111010110110110110011010100011101
$E(R_3) \oplus K_4$ = 001000101110111100101110110111100100101010110100
S-box outputs 00100001111011011001111100111010
$f(R_3,K_4)$ = 10111011001000110111011101001100
$L_5 = R_4$ = 01110111001000100000000001000101

$E(R_4)$ = 10111010111010010000010000000000000000001000001010
K_5 = 01111100111011000000011111101011010100111010100 0
$E(R_4) \oplus K_5$ = 11000110000001010000001111101011010100011010001 0
S-box outputs 01010000110010000011000111101011
$f(R_4,K_5)$ = 00101000000100111010110111000011
$L_6 = R_5$ = 10001010010011111010011000110111

$E(R_5)$ = 11000101010000100101111111010000110000011010111 1
K_6 = 01100011101001010011111001010000011110110010111 1
$E(R_5) \oplus K_6$ =10100110111001110110000011000000010111010100000000
S-box outputs 01000001111100110100110000111101
$f(R_5,K_6)$ = 10011110010001011100110100101100
$L_7 = R_6$ = 11101001011001111100110101101001

$E(R_6)$ = 11110101001010110000111111100101101010110101001 1
K_7 = 11101100100001001011011111111011000011000101111 00
$E(R_6) \oplus K_7$ = 00011001101011111011100000010011101100111110111 1
S- box outputs 00010000011101010100000010101101
$f(R_6,K_7)$ = 10001100000001010001110000100111
$L_8 = R_7$ = 00000110010010101011101000010000

$E(R_7)$ = 00000000110000100101010101011111010000001010000 0
K_8 = 11110111100010100011101011000001001110111111101 1
$E(R_7) \oplus K_8$ = 11110111010010000110111110011110011110110101101 1
S-box outputs 01101100000110000111110010101110
$f(R_7,K_8)$ = 00111100000011101000011011111001
$L_9 = R_8$ = 11010101011010010100101110010000

$E(R_8)$ = 01101010101010110101001010100101011111100101000 01
K_9 = 11100000110110111110101111101101111001111000000 1
$E(R_8) \oplus K_9$ = 10001010011100001011100101001000100110110010000 0
S-box outputs 00010001000011000101011101110111
$f(R_8,K_9)$ = 00100010001101100111110001101010
$L_{10} = R_9$ = 00100100011111001100011001111010

$E(R_9)$ = 00010000100000111111100101100000110000011111101 00
K_{10} = 10110001111100110100011110111010010000110010011 11
$E(R_9) \oplus K_{10}$ = 10100001011100001011111011011010100001011011101 1
S-box outputs 11011010000001000101001001110101
$f(R_9,K_{10})$ = 01100010101111001001110000100010
$L_{11} = R_{10}$ = 10110111110101011101011110110010

$E(R_{10})$ = 01011010111111101010101111101010111111011010010 1
K_{11} = 00100001010111111101001111011110110100111000011 0
$E(R_{10}) \oplus K_{11}$ = 01111011101000010111100000110100001011100010001 1
S-box outputs 01110011000001011101000100000001
$f(R_{10},K_{11})$ = 11100001000001001111101000000010
$L_{12} = R_{11}$ = 11000101011110000011110001111000

$E(R_{11})$ = 011000001010101111110000000111111000001111110001
K_{12} = 011101010111000111110101100101000110011111101001
$E(R_{11}) \oplus K_{12}$ = 000101011101101000000101100010111110010000011000
S-box outputs 01110011000001011101000100000001
$f(R_{11},K_{12})$ = 11000010011010001100111111101010
$L_{13} = R_{12}$ = 01110101101111010001100001011000

$E(R_{12})$ = 001110101011110111111010100011110000001011110000
K_{13} = 100101111100010111010001111111010101110100100001
$E(R_{12}) \oplus K_{13}$ = 101011010111100000101011011101011011100010110001
Sbox outputs 10011010110100011000101101001111
$f(R_{12},K_{13})$ = 11011101101110110010100100100010
$L_{14} = R_{13}$ = 00011000110000110001010101011010

$E(R_{13})$ = 000011110001011000000110100010101010101011110100
K_{13} = 010111110100001110110111111100101110011100111010
$E(R_{13}) \oplus K_{14}$ = 010100000101010110110001011110000100110111001110
S-box outputs 01100100011110011001101011110001
$f(R_{13},K_{14})$ = 10110111001100011000111001010101
$L_{15} = R_{14}$ = 11000010100011001001011000001101

$E(R_{14})$ = 11100000010101000101100101001010110000000101011
K_{15} = 10111111100100011000110100111101001111110000 1010
$E(R_{14}) \oplus K_{15}$ = 01011111110001011101010001110111111111111010100 01
S-box outputs 10110010111010001000110100111100
$f(R_{14},K_{15})$ = 01011011100000010010011101101110
R_{15} = 01000011010000100011001000110100

$E(R_{15})$ = 00100000011010100000010000011010010000011010 1000
K_{16} = 11001011001111011000101100001110000101111111010 5
$E(R_{15}) \oplus K_{16}$ = 11101011010101111000111100010100010101100101110 1
S-box outputs 10100111100000110010010000101001
$f(R_{15},K_{16})$ = 11001000110000000100111110011000
R_{16} = 00001010010011001101100110010101

Cuối cùng áp dụng IP^{-1} vào L_{16},R_{16} ta nhận được bản mã hexa là:

8 5 E 8 1 3 5 4 0 F 0 A B 4 0 5

3.3. TRANH LUẬN VỀ DES.

Khi DES được đề xuất như một chuẩn mật mã, đã có rất nhiều ý kiến phê phán. Một lý do phản đối DES có liên quan đến các hộp S. Mọi tính toán liên quan đến DES ngoại trừ các hộp S đều tuyến tính, tức việc tính phép hoặc loại trừ của hai đầu ra cũng giống như phép hoặc loại trừ của hai đầu vào rồi tính tóan đầu ra. Các hộp S - chứa đựng thành phần phi tuyến của hệ

mật là yếu tố quan trong nhất đối với độ mật của hệ thống(Ta đã thấy trong chương 1 là các hệ mật tuyến tính - chẳng hạn như Hill - có thể dễ dàng bị mã thám khi bị tấn công bằng bản rõ đã biết). Tuy nhiên tiêu chuẩn xây dựng các hộp S không được biết đầy đủ. Một số người đã gợi ý là các hộp S phải chứa các "cửa sập" được dấu kín, cho phép Cục An ninh Quốc gia Mỹ (NSA) giải mã được các thông báo nhưng vẫn giữ được mức độ an toàn của DES. Dĩ nhiên ta không thể bác bỏ được khẳng định này, tuy nhiên không có một chứng cớ nào được đưa ra để chứng tỏ rằng trong thực tế có các cửa sập như vậy.

Năm 1976 NSA đã khẳng định rằng, các tính chất sau của hộp S là tiêu chuẩn thiết kế:

P_0 Mỗi hàng trong mỗi hộp S là một hoán vị của các số nguyên 0, 1, . . . , 15.

P_1 Không một hộp S nào là một hàm Affine hoặc tuyến tính các đầu vào của nó.

P_2 Việc thay đổi một bít vào của S phải tạo nên sự thay đổi ít nhất là hai bít ra.

P_3 Đối với hộp S bất kì và với đầu vào x bất kì S(x) và S(x $\oplus$ 001100) phải khác nhau tối thiểu là hai bít (trong đó x là xâu bít độ dài 6).

Hai tính chất khác nhau sau đây của các hộp S có thể coi là được rút ra từ tiêu chuẩn thiết kế của NSA.

P_4 Với hộp S bất kì, đầu vào x bất kì và với e, f $\in\{0,1\}$: S(x) $\neq$S(x $\oplus$ 11ef00).

P5 Với hộp S bất kì , nếu cố định một bít vào và xem xét giá trị của một bít đầu ra cố định thì các mẫu vào để bít ra này bằng 0 sẽ xấp xỉ bằng số mẫu ra để bít đó bằng 1.(Chú ý rằng, nếu cố định giá trị bít vào thứ nhất hoặc bít vào thứ 6 thì có 16 mẫu vào làm cho một bít ra cụ thể bằng 0 và có 16 mẫu vào làm cho bít này bằng 1. Với các bít vào từ bít thứ hai đến bít thứ 5 thì điều này không còn đúng nữa. Tuy nhiên phân bố kết quả vẫn gần với phân bố đều. Chính xác hơn, với một hộp S bất kì, nếu ta cố định giá trị của một bít vào bất kì thì số mẫu vào làm cho một bít ra cố định nào đó có giá trị 0 (hoặc 1) luôn nằm trong khoảng từ 13 đến 19).

Người ta không biết rõ là liệu có còn một chuẩn thiết kế nào đầy đủ hơn được dùng trong việc xây dựng hộp S hay không.

Sự phản đối xác đáng nhất về DES chính là kích thước của không gian khoá: 2^{56} là quá nhỏ để đảm bảo an toàn thực sự. Nhiều thiết bi chuyên dụng đã được đề xuất nhằm phục vụ cho việc tấn công với bản rõ đã biết. Phép tấn công này chủ yếu thực hiện tìm khoá theo phương pháp vét cạn. Tức với bản rõ x 64 bít và bản mã y tương ứng, mỗi khoá đều có thể được kiểm tra cho tới khi tìm được một khoá K thảo mãn $e_K(x) = y$. Cần chú ý là có thể có nhiều hơn một khoá K như vậy).

Ngay từ năm 1977, Diffie và Hellman đã gợi ý rằng có thể xây dựng một chíp VLSI (mạch tích hợp mật độ lớn) có khả năng kiểm tra được 10^6khoá/giây. Một máy có thể tìm toàn bộ không gian khoá cỡ 10^6 trong khoảng 1 ngày. Họ ước tính chi phí để tạo một máy như vậy khoảng $2.10^7$$.

Trong cuộc hội thảo tại hội nghị CRYPTO'93, Michael Wiener đã đưa ra một thiết kế rất cụ thể về máy tìm khoá. Máy này xây dựng trên một chíp tìm khoá, có khả năng thực hiện đồng thời 16 phép mã và tốc độ tới 5×10^7 khoá/giây. Với công nghệ hiện nay, chi phí chế tạo khoảng 10,5$/chíp. Giá của một khung máy chứa 5760 chíp vào khoảng 100.000$ và như vậy nó có khả năng tìm ra một khoá của DES trong khoảng 1,5 ngày. Một thiết bị dùng 10 khung máy như vậy có giá chừng 10^6 $ sẽ giảm thời gian tìm kiếm khoá trng bình xuống còn 3,5 giờ.

3.4. DES TRONG THỰC TẾ.

Mặc dù việc mô tả DES khá dài dòng song người ta có thể thực hiện DES rất hữa hiệu bằng cả phần cứng lẫn phần mền. Các phép toán duy nhất cần được thực hiện là phép hoặc loại trừ các xâu bít. Hàm mở rộng E, các hộp S, các hoán vị IP và P và việc tính toán các giá tri K_1,.. . ,K_{16} đều có thể thực hiện được cùng lúc bằng tra bảng (trong phần mền) hoặc bằng cách nối cứng chúng thành một mạch.

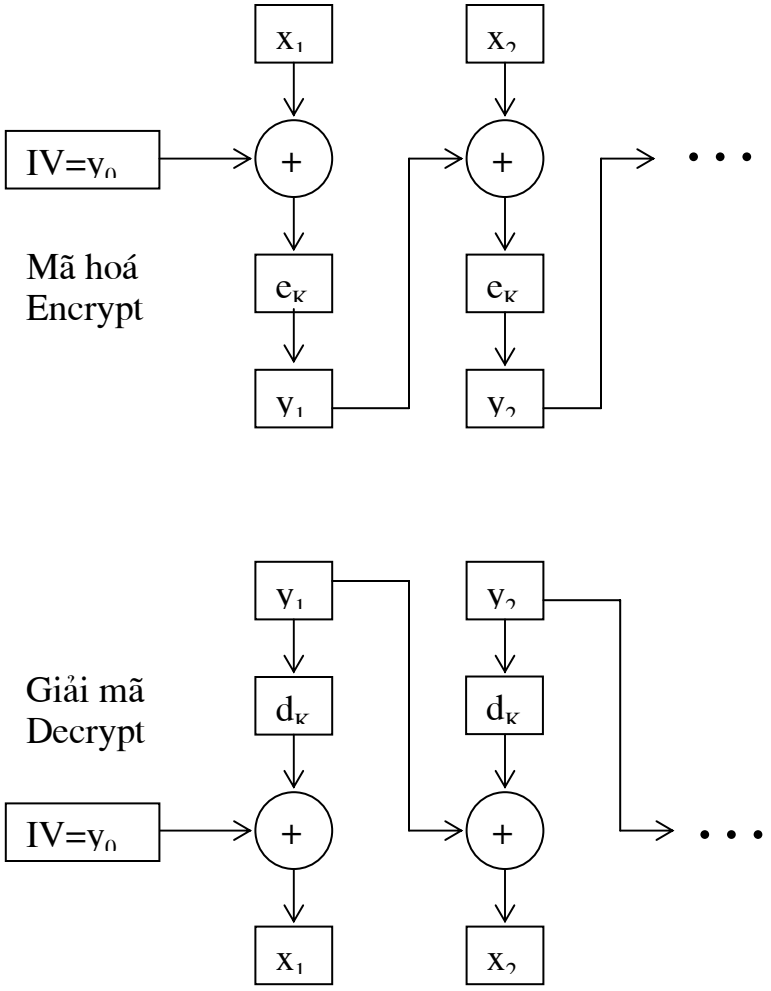
Các ứng dụng phần cứng hiện thời có thể đạt được tốc độ mã hoá cực nhanh. Công ty Digital Equipment đã thông báo tại hội nghị CRUPTO'92 rằng họ sẽ chế tạo một chíp có 50 ngàn tranzistor có thể mã hoá với tốc độ 1 Gbít/s bằng cách dùng nhịp có tốc độ 250MHz. Giá của chíp này vào khoảng 300$. Tới năm 1991 đã có 45 ứng dụng phần cứng và chương trình cơ sở của DES được Uỷ ban tiêu Chuẩn quốc gia Mỹ (NBS) chấp thuận.

Một ứng dụng quan trọng của DES là trong giao dịch ngân hàng Mỹ - (ABA) DES được dùng để mã hoá các số định danh cá nhân (PIN) và việc chuyển tài khoản bằng máy thủ quỹ tự động (ATM). DES cũng được Hệ thống chi trả giữa các nhà băng của Ngân hàng hối đoái (CHIPS) dùng để xác thực các giao dịch vào khoản trên $1,5\times10^{12}$ USA/tuần. DES còn được sử dụng rộng rãi trong các tổ chức chính phủ. Chẳng hạn như bộ năng lượng, Bộ Tư pháp và Hệ thống dự trữ liên bang.

3.4.1. Các chế độ hoạt động của DES.

Có 4 chế độ làm việc đã được phát triển cho DES: Chế độ chuyển mã điện tử (ECB), chế độ phản hồi mã (CFB), chế độ liên kết khối mã (CBC) và chế độ phản hồi đầu ra (OFB). Chế độ ECB tương ứng với cách dùng thông thường của mã khối: với một dãy các khối bản rõ cho trước $x_1,x_2,\ldots$ (mỗi khối có 64 bít), mỗi x_i sẽ được mã hoá bằng cùng một khoá K để tạo thành một chuỗi các khối bản mã y_1y_2 ... theo quy tắc $y_i = e_K(y_{i-1} \oplus x_i)$ $i \geq 1$. Việc sử dụng chế độ CBC được mô tả trên hình 3.4.

Hình 3.4. Chế độ CBC.

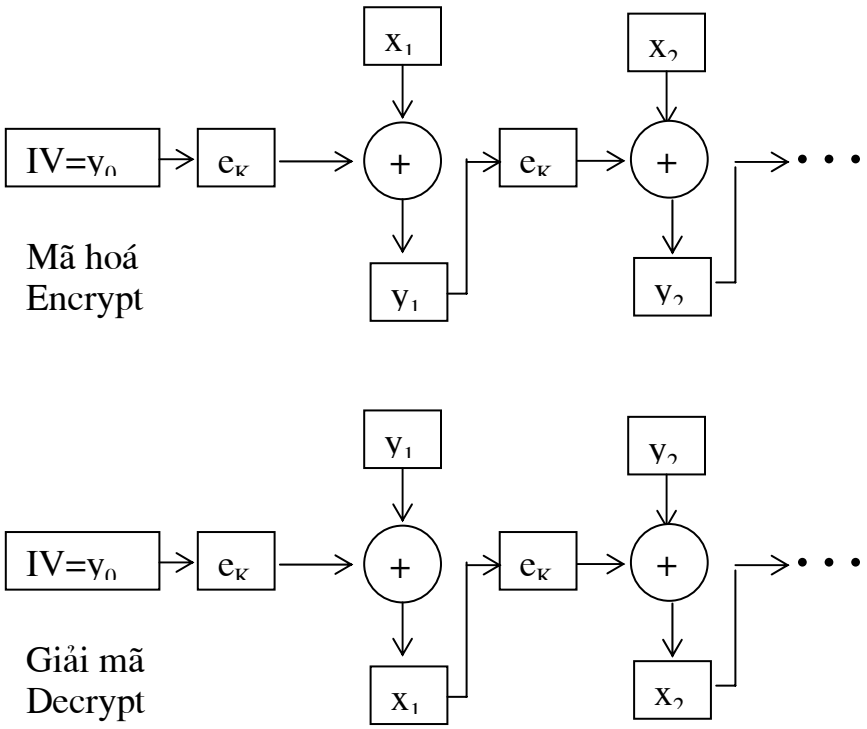


Trong các chế độ OFB và CFB dòng khoá được tạo ra sẽ được cộng mod 2 với bản rõ (tức là nó hoạt động như một hệ mã dòng, xem phần 1.1.7). OFB thực sự là một hệ mã dòng đồng bộ: dòng khoá được tạo bởi việc mã lặp véc tơ khởi tạo 64 bít (véc tơ IV). Ta xác định z_0 =IV và rồi tính dòng

khoá z_1z_2 . . . theo quy tắc $z_i = e_K(z_{i-1})$, i≥1. Dãy bản rõ x_1x_2 . . . sau đó sẽ được mã hoá bằng cách tính $y_i = x_i \oplus z_i$,i ≥1.

Trong chế độ CFB, ta bắt đầu với y_0 = IV (là một véc tơ khởi tạo 64 bít) và tạo phần tử z_i của dòng khoá bằng cách mã hoá khối bản mã trước đó. Tức $z_i = e_K(y_{i-1})$, i ≥1. Cũng như trong chế độ OFB: $y_i = x_i \oplus z_i$,i ≥1. Việc sử dụng CFB được mô tả trên hình 3.5 (chú ý rằng hàm mã DES e_K được dùng cho cả phép mã và phép giải mã ở các chế độ CFB và OFB).

Hình 3.5. Chế độ CFB



Cũng còn một số biến tấu của OFB và CFB được gọi là các chế độ phản hồi K bít (1 < K < 64). Ở đây ta đã mô tả các chế độ phản hồi 64 bít. Các chế độ phản hồi 1 bít và 8 bít thường được dùng trong thực tế cho phép mã hoá đồng thời 1 bit (hoặc byte) số liệu.

Bốn chế độ công tác có những ưu, nhược điểm khác nhau. Ở chế độ ECB và OFB, sự thay đổi của một khối bản rõ x_i 64 bít sẽ làm thay đổi khối bản mã y_i tương ứng, nhưng các khối bản mã khác không bị ảnh hưởng. Trong một số tình huống đây là một tính chất đáng mong muốn. Ví dụ, chế độ OFB thường được dùng để mã khi truyền vệ tinh.

Mặt khác ở các chế độ CBC và CFB, nếu một khối bản rõ x_i bị thay đổi thì y_i và tất cả các khối bản mã tiếp theo sẽ bi ảnh hưởng. Như vậy các chế độ CBC và CFB có thể được sử dụng rất hiệu quả cho mục đích xác thực. Đặc biệt hơn, các chế độ này có thể được dùng để tạo mã xác thực bản tin (MAC - message authentication code). MAC được gắn thêm vào các khối bản rõ để thuyết phục Bob tin rằng, dãy bản rõ đó thực sự là của Alice mà không bị Oscar giả mạo. Như vậy MAC đảm bảo tính toàn vẹn (hay tính xác thực) của một bản tin (nhưng tất nhiên là MAC không đảm bảo độ mật).

Ta sẽ mô tả cáchb sử dụng chế độ BCB để tạo ra một MAC. Ta bắt đầu bằng véc tơ khởi tạ IV chứa toàn số 0. Sau đó dùng chế đô CBC để tạo các khối bản mã $y_1,. . . ,y_n$ theo khoá K. Cuối cùng ta xác định MAC là y_n. Alice sẽ phát đi dãy các khối bản rõ $x_1,x_2,. . . ,x_n$ cùng với MAC. Khi Bob thu được $x_1. . .x_n$ anh ta sẽ khôi phục lại $y_1. . .y_n$ bằng khoá K bí mật và xác minh xem liệu y_n có giống với MAC mà mình đã thu được hay không.

Nhận thấy Oscar không thể tạo ra một MAC hợp lệ do anh ta không biết khoá K mà Alice và Bob đang dùng. Hơn nữa Oscar thu chặn được dãy khối bản rõ $x_1. . .x_n$ và thay đổi ít nhiều nội dung thì thì chắc chắn là Oscar không thể thay đổi MAC để được Bob chấp nhận.

Thông thường ta muốn kết hợp cả tính xác thực lẫn độ bảo mật. Điều đó có thể thực hiện như sau: Trước tiên Alice dùng khoá K_1 để tạo MAC cho $x_1. . . x_n$. Sau đó Alice xác định x_{n+1} là MAC rồi mã hoá dãy x1. . .x_{n+1} bằng khoá thứ hai K_2 để tạo ra bản mã $y_1. . .y_{n+1}$. Khi Bob thu được y1. . .y_{n+1} , trước tiên Bob sẽ giải mã (bằng K_2) và kiểm tra xem x_{n+1} có phải là MAC đối với dãy $x_1. . .x_n$ dùng K_1 hay không.

Ngược lại, Alice có thể dùng K_1 để mã hoá $x_1. . .x_n$ và tạo ra được $y_1...y_n$, sau đó dùng K_2 để tạo MAC y_{n+1} đối với dãy $y_1. . .y_n$. Bob sẽ dùng K_2 để xác minh MAC và dung K_1 để giải mã $y_1. . .y_n$.

3.5. PHÉP TỐI ƯU HOÁ THỜI GIAN - BỘ NHỚ.

Trong phần này sẽ mô tả phép tối ưu hoá thời gian - bô nhớ khá lý thú khi phá DES bằng tấn công bản rõ chọn lọc. Ta nhớ lại rằng, trong phép tấn công bản rõ chọn lọc, Oscar đã thu được cặp rõ - mã được tạo bởi khoá K (chưa biết). Bởi vậy, Oscar có x và y, trong đó $y = e_K(x)$ và anh ta muốn xác định được K.

Một đặc điểm của phép tối ưu hoá thời gian - bộ nhớ này là nó không phụ thuộc vào "cấu trúc" của DES trên mọi phương diện. Khía cạnh duy nhất của DES có quan hệ tới phép tấn công này là các bản rõ và các bản mã 64 bít trong khi các khoá có 56 bít.

Ta đã thảo luận về ý tưởng tìm khoá bằng phương pháp vét cạn: với một cặp rõ - mã cho trước, hãy thử tất cả 2^{56} khoá cụ thể. Điều này không yêu cầu bộ nhớ, nhưng trung bình phải thử 2^{55} khoá trước khi tìm được khoá đúng. Mặt khác, với một bản rõ x cho trước, Oscar có thể tính trước $y_K = e_K(x)$ đối với toàn bộ 2^{56} khoá K và xây dựng một bảng các cặp (y_K,K) được sắp xếp theo các tạo độ đầu của chúng. Sau đó khi Oscar thu được bản mã y (là kết quả của phép mã bản rõ x), anh ta phải nhìn vào giá trị y trong bảng và lập tức tìm được khoá K. Như vậy trong trường hợp này việc tìm được khoá K chỉ yêu câu một thời gian cố định nhưng ta phải có một bô nhớ có dung lượng lớn và cần thời gian tính toán trước lớn (chú ý là quan điểm này không có lợi thế về thời gian tính toán tổng cộng nếu chỉ cần tìm một khoá, bởi vì việc xây dựng bảng cũng mất nhiều thời gian như việc tìm khóa vét cạn. Phương pháp này chỉ có lợi khi cần tìm nhiều khoá trong một khoảng thời gian vì ta chỉ cấn dùng một bảng cho tất cả các trường hợp).

Phép tối ưu hoá thời gian - bộ nhớ sẽ có thời gian tính toán nhỏ hơn phép tìm kiếm vét cạn và có yêu cầu bộ nhớ nhỏ hơn việc lập bangr tra cứu. Thuật toán có thể mô tả theo hai tham số m và t là các số nguyên dương. Thuật toán cần một hàm rút gọn R để rút gọn một xâu bít có độ dài 64 thành một xâu bít có độ dài 56 (chẳng hạn R phải vứt bỏ 8 trong 64 bít). Giả sử x là một xâu bản rõ cố định 64 bít. Hãy xác định hàm $g(K_0) = R(e_{K_o}(x))$ với một xâu bít K_0 có độ dài 56. Chú ý rằng g là một hàm thực hiện ánh xạ 56 bít sab\ng 56 bít.

Trong giai đoạn tiền xử lý, Oscar chọn m xâu bít ngẫu nhiên có độ dài 56 được kí hiệu là X(i,0), $1 \le i \le m$. Oscar tính x(i,j) với $1 \le j \le t$ theo quan hệ truy toán sau: $X(i,j) = g(X(i,j-1))$, $1 \le i\ x \le m$, $1 \le j \le t$ như chỉ trên hình 3.6.

Hình 3.6. Tính X(i,j)

$$X(1,0) \xrightarrow{g} X(1,1) \xrightarrow{g} \ldots \xrightarrow{g} X(1,t)$$
$$X(2,0) \xrightarrow{g} X(2,1) \xrightarrow{g} \ldots \xrightarrow{g} X(2,t)$$
$$\vdots$$
$$X(m,0) \xrightarrow{g} X(m,1) \xrightarrow{g} \ldots \xrightarrow{g} X(m,t)$$

Sau đó Oscar xây dựng một bảng các cặp T = (X(i,t), X(i,0) được sắp xếp theo toạ độ đầu của chúng(tức là chỉ lưu giữ cột đầu và cột cuối của X).

Sau khi thu được bản mã y (là bản mã của bản rõ x đã chọn). Oscar cần phải xác định K và anh ta sẽ xác định được nếu K nằm trong t cột đầu của bảng X, tuy nhiên anh ta chỉ làm điều này bằng cách chỉ nhìn vào bảng T.

Giả sử rằng K = X(i,t-j) với j nào đó, $1 \le j \le t$ (tức giả sử rằng K nằm ở t cột đầu tiên của X). Khi đó rõ ràng là $g^j(K) = x(i,t)$, trong đó g^j kí hiệu hàm nhận được bằng cách lặp g một số lần bằng j. Bây giờ ta thấy rằng:

$$\begin{aligned} g^j(K) &= g^{j-1}(g(K)) \\ &= g^{j-1}(R(e_K(x))) \\ &= g^{j-1}(R(y)) \end{aligned}$$

Giả sử tính $y_j, 1 \le j \le t$, từ quan hệ truy toán

$$y_i = \begin{cases} R(y) & \text{nếu } j = 1 \\ g(y_{j-1}) & \text{nếu } 2 \le j \le t \end{cases}$$

Từ đó rút ra rằng $y_j = X(i,t-j)$ nếu K = X(i,t-j). Tuy nhiên cần chú ý rằng $y_j = X(i,t)$ chưa đủ để đảm bảo là K = X(i,t-j). Sở dĩ như vậy vì hàm rút gọn R không phải là một đơn ánh: miền xác định của R có lực lượng 2^{64} và giá trị của R có lực lượng 2^{56}, bởi vậy tính trung bình có $2^8 = 256$ nghịch ảnh của một xâu bít bất kì cho trước có độ dài 56. Bởi vây cần phải kiểm tra xem $y = e_{X(i,t-j)}(x)$ hay không để biết liệu X(i,t-j) có thực sự là khoá hay không. Ta không lưu trữ giá trị X(i,t-j) nhưng có thể dễ dàng tính lại nó từ X(i,0) bằng cách lặp t-j lần hàm g.

Oscar sẽ thực hiện theo thuật toán được mô tả trên hình 3.7.

Hình 3.7. Phép tối ưu hoá bộ nhớ - thời gian trong DES.

1. Tính $y_1 = R(y)$
2. **for** j = 1 to t **do**
3. **if** $y_j = X(i,t-j)$ với giá trị i nào đó **then**
4. Tính X(i,t-j) từ X(i,0) bằng cách lặp t-j lần hàm g.
5. **if** y = eX(i,t-j)(x) **then**
   đặt K = X(i,t-j) và **QUIT**
6. Tính $y_{j+1} = g(y_j)$

Bằng cách phân tích xác suất thành công của thuật toán, có thể chứng tỏ rằng nếu $mt^2 \approx N = 2^{56}$ thì xác suất để K = X(i,t-j) với i, j nào đó sẽ vào khoảng 0,8môi trường/N. Thừa số 0,8 tính theo điều kiện không phải tất cả cácX(i,t) đều phân biệt . Điều này gợi ý cho ta nên lấy $m \approx t \approx N^{1/3}$ và xây dựng khoảng $N^{1/3}$ bảng, mỗi bảng dùng một hàm rút gọn R khác nhau. Nếu thực hiện đươc điều này thì yêu cầu về bộ nhớ là $112\times N^{1/3}$ bít (vì ta cần lưu trữ $2\times N^{2/3}$ số nguyên, mỗi số có 56 bít). Thời gian tiền tính toán dễ dàng thấy là cỡ O(N).

Việc phân tích thời gian chạy của thuật toán có khó hơn hơn một chút: Trước hết ta thấy rằng, bước 3 có thể chạy trong một thời gian không đổi (sử dụng phép mã hash) hoặc trong trường hợp xấu nhất, bước 3 có thể chạy với thời gian O(logm) khi dùng phép tmf kiếm nhị phân. Nếu bước 3 không thoả mãn (tức là phép tìm kiếm không thành công) thì thời gian chạy là $O(N^{2/3})$. Các phân tích chi tiết hơn chứng tỏ rằng, ngay cả khi tính cả thời gian chạy của các bước 4 và5 thì thời gian chạy trung bình chỉ tăng một lương là hằng số.

3.6 THÁM MÃ VI SAI (DC).

Phương pháp DC do Biham và Shamir đưa ra là một phương pháp tấn công DES rất nổi tiếng. Đây là một phép tấn công với bản rõ chọn lọc. Mặc dù phương pháp này không cho một phương pháp thực tế để phá DES 16 vòng thông dụng, nhưng nó có thể thực hiện thành công trong việc phá DES có số vòng mã hoá ít hơn. Chẳng hạn DES 8 vòng có thể phá được trong vòng vài phút trên một máy tính cá nhân nhỏ.

Bây giờ ta sẽ mô tả những ý tưởng cơ bản dùng trong kỹ thuật này, ta có thể bỏ qua phép hoá vị ban đầu IP và phép hoán vị ngược của nó (không

ảnh hưởng tới việc phân tích mã). Như đã nói ở trên, ta chỉ xét hạn chế DES n vòng với $n \leq 16$. Bởi vậy, với các điều kiện trên, ta coi L_0R_0 là bản rõ và L_nR_n là bản mã trong DES n vòng (cần chú ý rằng ta không cần đảo L_nR_n).

Phương pháp DC xoay quanh việc so sánh kết quả phép hoặc - loại trừ của hai bản rõ với kết quả của phép hoặc - loại trừ của hai bản mã tương ứng. Đại thể ta sẽ xét hai bản rõ L_0R_0 và$L_0^*R_0^*$ với giá trị của phép hoặc - loại trừ $L_0'R_0' = L_0R_0 \oplus L_0^*R_0^*$. Trong phần này ta sẽ sử dụng ký hiệu (') để chỉ phép hoặc - loại trừ (XOR) của hai xâu bít.

Định nghĩa 3.1

Giả sử S_j là một hộp S ($1 \leq j \leq 8$). Xét một cặp đã sắp xếp của các xâu bít độ dài 6 (ký hiệu là B_j, B_j^*). Ta nói rằng XOR vào (của S_j) là $B_j \oplus B_j^*$ và XOR ra (của S_j) là $S_j(B_j) \oplus S_j(B_j^*)$.

Chú ý rằng XOR vào là một xâu bít có độ dài 6 và XOR ra là một xâu bít có độ dài 4.

Định nghĩa 3.2

Với bất kỳ $B_j' \oplus (Z_2)^6$, ta định nghĩa tập $\nabla(B_j')$ gồm các cặp được sắp xếp (B_j,B_j^*) có XOR vào là B_j'.

Dễ dàng thấy rằng một tập $\nabla(B_j')$ bất kỳ đều chứa $2^6 = 64$ cặp và

$$\nabla(B_j') = \{(B_j, B_j \oplus B_j') : B_j \in (Z_2)^6\}$$

Với mỗi cặp trong $\nabla(B_j')$ ta có thể tính XOR ra của S_j và lập bảng phân bố kết quả. Có 64 XOR ra phân bố trong $2^4 = 16$ giá trị có thể. Tính không đều của các phân bố này là cơ sở cho phép tấn công.

Ví dụ 3.1.

Giả sử xét hộp S đầu tiên S_1 và XOR vào 110100, khi đó:

$\nabla(110100) = \{(000000,110100), (000001,110100), \ldots ,(111111,110100)\}$

Với mỗi cặp được sắp trong tập$\nabla(110100)$ ta tính XOR ra của S_1. Ví dụ $S_1(000000) = E_{16} = 1110$ và $S_1(110100) = 9_{16} = 1001$, bởi vậy XOR đối với cặp (000000,110100) là 0111.

Nếu làm công việc này cho tất cả 64 cặp trong $\nabla(110100)$ thì ta sẽ thu được phân bố sau của các XOR ra:

0000	0001	0010	0011	0100	0101	0110	0111
0	8	16	6	2	0	0	12

1000	1001	1010	1011	1100	1101	1110	1111
6	0	0	0	0	8	0	6

Trong ví dụ 3.1 chỉ có 8 trong 16 XOR ra có thể xuất hiện trên thực tế. Ví dụ cụ thể này có phân bố rất không đều. Nói chung nếu ta cố định một hộp S là S_j và một XOR vào B_j' thì trung bình có khoảng 75-80% các XOR ra là có thể xuất hiện.

Để mô tả và đưa ra các phân bố này, ta cần phải có thêm mọt số khái niệm thích hợp. Sau đó là một số định nghĩa.

Định nghĩa 3.3

Với $1 \le j \le 8$ và với các xâu bít B_j' có độ dài 6 còn C_j' có độ dài 4, ta định nghĩa:

$$IN_j(B_j',C_j') = \{ Bj \in (Z_2)^6 : S_j(B_j) \oplus S_j(B_j \oplus B_j') = C_j'\}$$

và

$$N_j(B_j',C_j') = | IN_j(B_j',C_j') |.$$

$N_j(B_j',C_j')$ là số các cặp có XOR vào bằng B_j' và có XOR ra bằng C_j' với hộp S_j. Các cặp thực tế có các XOR vào xác định và tạo nên các XOR ra xác định có thể nhận được từ tập $IN_j(B_j',C_j')$. Ta thấy rằng, tập này có thể được phân thành $N_j(B_j',C_j')/2$ cặp, mỗi cặp có số XOR vào bằng B_j'.

Chú ý rằng phân bố được lập bảng ở trong ví dụ 3.1 chứa các giá trị $N_1(110100,C_1')$, $C_1' \in (Z_2)^4$. Các tập $IN_1(110100,C_1')$ được liệt kê trên hình 3.8.

Với mỗi hộp trong 8 hộp S có 64 XOR và có thể. Bởi vậy có thể tính được tất cả 512 phân bố và dễ dàng dùng máy tính để lập bảng các phân bố này.

Cần nhớ lại rằng, đầu vào của các hộp S ở vong thứ i là $B = E \oplus J$, trong đó $E = E(R_{i-1})$ là một hàm mở rộng của R_{i-1} và $J = K_i$ là các bít khoá của vòng thứ i. Bây giờ số XOR vào (cho tất cả 8 hộp) có thể được tính như sau:

$$B \oplus B^* = (E \oplus J) \oplus (E^* \oplus J)$$
$$= E \oplus E^*$$

Có thể thấy môt điều rất quan trọng là XOR vao không phụ thuộc vào các bít khoá J (tuy nhiên chắc chắn XOR ra sẽ phụ thuộc vào các bít khóa này.

Hình 3.8. Các xâu vào có thể với XOR vào là 110100.

Các XOR ra	Các xâu vào có thể
0000	
0001	000011,001111,011110,011111, 101010,101011,110111,111011
0010	000100,000101,001110,010001, 010010,010100,011010,011011, 100000,100101,010110,101110, 101111,110000,110001,111010
0011	000001,000010,010101,100001, 110101,110110
0100	010011,100111
0101	
0110	
0111	000000,001000,001101,010111 011000,011101,100011,101001 101100,110100,111001,111100
1000	001001,001100,011001,101101 111000,111101
1001	
1010	
1011	
1100	
1101	000110,010000,010110,011100 100010,100100,101000,110010
1110	
1111	000111,001010,001011,001100 111110,111111

Ta viết các E,B, và J là một dãy ghép kế tiếp 8 xâu 6 bít:

$$B = B_1\, B_2\, B_3\, B_4\, B_5\, B_6\, B_7\, B_8$$
$$E = E_1\, E_2\, E_3\, E_4\, E_5\, E_6\, E_7\, E_8$$
$$J = J_1\, J_2\, J_3\, J_4\, J_5\, J_6\, J_7\, J_8$$

và viết B^*, E^*,J^* theo cách tương tự. Nếu biết các giá trị E_j và E_j^* với j nào đó, $1 \le j \le 8$, và giá trị XOR ra (của S_j) là $C_j' = S_j(B_j) \oplus S_j(B_j^*)$. Khi đó chắc chắn rằng:

$$E_j \oplus J_j \in IN_j(E_j',C_j')$$

trong đó $E_j' = E_j \oplus E_j^*$.

Giả sử ta xác định tập $test_j$ như sau:

Định nghĩa 3.4.

Giả sử E_j và E_j^* là các xâu bít độ dài 6 và C_j' là xâu bít độ dài 4. Ta định nghĩa:

$$test_j(E_j , E_j^* , C_j') = \{B_j \oplus E_j : B_j \in IN_j(E_j',C_j')\}$$

trong đó $E_j' = E_j E_j^*$

Nghĩa là lấy XOR E_j với mỗi phần tử của tập $IN_j(E_j',C_j')$.

Kết quả sau đây là một hệ quả trực tiếp rút ra từ suy luận ở trên.

Định lý 3.1

Giả sử E_j và E_j^* là hai xâu vào của hộp S_j còn XOR ra của S_j là C_j. Kí hiệu $E_j' = E_j \oplus E_j^*$. Khi đó các bít khoá J_j sẽ nằm trong tập $test_j (E_j, E_j^* , C_j')$.

Nhận thấy rằng có đúng $N_j(E_j',C_j')$ xâu bít độ dài 6 trong tập $test_j(E_j,E_j^*,C_j')$; giá trị đúng của J_j phải là một trong các khả năng này.

Ví dụ 3.2.

Giả sử $E_1 = 000001$, $E_1^* = 110101$ và $C_1' = 1101$. Vì $N_1(110100,1101) = 8$ nên có đúng 8 xâu bít trong tập $test_1(000001,110101,1101)$. Từ hình 3.8 ta thấy rằng:

$IN_1(110100,1101) =$

{000110,010000,010110,011100,100010,100100,101000,110010}

Bởi vậy

$test_1(000001,110101,1101) =$

{000111,010001,010111,011101,100011,100101,101001,110011}.

Nếu ta có một bộ ba E_1,E_1^*,C_1' thứ hai như vậy thì có thể thu được tập $test_1$ thứ hai chứa các giá trị có thể chứa các bít khoá trong J_1. Giá trị đúng của J_1 phải nằm trong phần giao của hai tập này. Nếu ta có vài bộ ba như vậy thì có thể nhanh chóng xác định được các bít khoá trong J_1. Phương pháp đơn giản để làm điều này là tạo một dãy 64 bộ đếm biểu diễn 64 khả năng của 6 bít khoá trong J_1 . Bộ đếm sẽ đếm tăng mỗi khi các bít khoá tương ứng xuất

hiện trong tập $test_1$ với một bộ ba cụ thể. Với t bộ ba, ta tin rằng sẽ tìm được bộ đếm duy nhất có giá trị t tương ứng với giá trị đúng của các bít khoá trong J_1.

3.6.1. Tấn công DES 3 vòng

Bây giờ ta xét xem việc ứng dụng các ý tưởng của phần trước trong phép tấn công bản rõ chọn lọc lên một hệ DES 3 vòng. Ta bắt đầu bằng một cặp các bản rõ và bản mã tương ứng L_0R_0, $L_0^*R_0^*$,L_3R_3, và $L_3^*R_3^*$. Có thể biểu thị R_3 như sau:

$$\begin{aligned} R_3 &= L_2 \oplus f(R_2,K_3) \\ &= R_1 \oplus f(R_2,K_3) \\ &= L_0 \oplus f(R_0,K_1) \oplus f(R_2,K_3) \end{aligned}$$

Biểu diễn R_3^* theo cách tương tự như vậy

$$R_3' = L_0' \oplus f(R_0,K_1) \oplus f(R_0^*,K_1) \oplus f(R_2,K_3) \oplus f(R_2^*,K_3)$$

Bây giờ, giả sử ta đã chọn được các bản rõ sao cho $R_0 = R_0^*$, nghĩa là để

$$R_0' = 00\ldots0$$

Khi đó $f(R_0,K_1) = f(R_0^*,K_1)$ và như vậy:

$$R_3' = L_0' \oplus f(R_2,K_3) \oplus f(R_2^*,K_3).$$

Lúc này R_3' đã biết vì có thể tính được nó từ hai bản mã. L_0' cũng đã biết do có thể tính được nó từ hai bản rõ. Điều này có nghiã là ta có thể tính $f(R_2,K_3)\oplus f(R_2^*,K_3)$ từ phương trình:

$$f(R_2,K_3)\oplus f(R_2^*,K_3) = R_3' \oplus L_0'$$

Bây giờ ta có $f(R_2,K_3) = P(C)$ và $f(R_2^*,K_3) = P(C^*)$, trong đó C và C^* ký hiệu tương ứng 2 dãy ra của 8 hộp S (hãy nhớ lại rằng P là một phép hoán vị cố định công khai). Bởi vậy:

$$P(C) \oplus P(C^*) = R_3' \oplus L_0'$$

và do đó:

$$C' = C \oplus C^* = P^{-1}(R_3' \oplus L_0') \qquad (3.1)$$

Đây là XOR ra của 8 hộp S ở vòng thứ 3.

Bây giờ $R_2 = L_3$ và $R_2^* = L_3^*$ cũng đã biết (chúng là một phần của các bản mã). Bởi vậy, có thể tính

$$E = E(L_3) \qquad (3.2)$$

và

$$E^* = E(L_3^*) \qquad (3.3)$$

bằng cách dùng hàm mở rộng E được biết công khai. Đây là các mẫu vào các hột S ở vòng thứ 3. Như thế ta đã biết E và E^* và C ' của vòng thứ 3 và có thể thực hiện (như ở phần trước) để xây dựng các tệp $test_1$, . .., $test_8$ chứa các giá trị có thể của các bít trong J_1,. . .,J_8 .

Mô tả dạng giả mã của thuật toán này được cho ở hình 3.9.

Hình 3.9. Cách tấn công DC lên DES 3 vòng.

Đầu vào L_0R_0,$L_0^*R_0^*$, L_3R_3 và $L_3^*R_3^*$, trong đó $R_0 = R_0^*$
1. Tính $C' = P^{-1}(R_3' \oplus L_0')$
2. Tính $E = E(L_3)$ và $E^* = E(L_3^*)$
3. **For** j = 1 **to** 8 **do**
   Tính $test_j(E_j, E_j^*, C_j')$

Trong phương pháp tấn công này sẽ phải dùng một số bộ ba E, E^*,C ' như vậy, Ta phải thiết lập 8 dãy bộ đếm và nhờ vậy xác định được 48 bít trong khoá K_3 (khoá của vòng thứ 3). Sau đó tính 56 bít trong khóa theo cách tìm kiếm vét cạn trong $2^8 = 256$ khả năng cho 8 bít khoá còn lại.

Ta sẽ xem xét một ví dụ để minh hoạ.

Ví dụ 3.3.

Giả sử ta có 3 cặp các bản rõ và các bản mã, trong đó các bản rõ có các phép XOR xác định, chúng được mã hoá bằng cùng một khoá. Để cho gọn ta sẽ biểu thị dưới dạng mã Hexa:

Bản rõ	Bản mã
748502CD38451097 3874756438451097	03C70306D8AO9F10 78560A960E6D4CB
486911026ACDFF31 375BD31F6ACDFF31	45FA285BE5ADC730 134F7915AC253457
357418DA013FEC86 12549847013FEC86	D8A31B2F28BBC5CF 0F317AC2B23CB944

Từ cặp đầu tiên, tính các đầu vào của hộp S (cho vòng 3) theo các phương trình (3.2) và (3.3). Ta có:

E = 000000000111111000001110100000000110100000001100
E^* = 101111110000001010101100000001010100000001010010

XOR ra của các hộp S được tính theo phương trình (3.1) là:

C' = 10010110010111010101101101100111

Từ cặp thứ hai, ta tính được các đầu vào của các hộp S là:

E = 10100000101111111111010000010101000000101111010 6
E^* = 00000101111010011010001010111111010101100000010 0

và XOR ra của các hộp S là:

C' = 11010101011101011101101100101011

Tiếp theo, lập bảng các giá trị trong 8 dãy bộ đếm cho từng cặp. Minh hoạ thủ tục này với dãy bộ đếm cho J_1 theo cặp đầu tiên. Trong cặp này ta có: E' = 101111 và C' = 1001. Khi đó tập:

IN_1(101111,1001) = {000000,000111,101000,101111}

vì E_1 = 000000 nê ta có:

$J_1 \in test_1$(000000,101111,1001) = {000000,000111,101000,101111}

Bởi vậy ta sẽ tăng các giá trị 0,7,40 và 47 trong dãy bộ đếm cho J_1.

Bây giờ sẽ trình bày các bảng cuối cùng. Nếu coi một xâu bít độ dài 6 như biểu diễn nhị phân của một số nguyên nằm giữa 0 và 63 thì 64 giá trị tương ứng là 0,1,. . . ,63. Các mảng bộ đếm sẽ như sau:

J_1															
1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

J_2															
0	0	0	1	0	3	0	0	1	0	0	1	0	0	0	0
0	1	0	0	0	2	0	0	0	0	0	0	1	0	0	0
0	0	0	0	0	1	0	0	1	0	1	0	0	0	1	0
0	0	1	1	0	0	0	0	1	0	1	0	2	0	0	0

J_3															
1	0	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

J_4															
3	1	0	0	0	1	0	1	0	0	0	0	0	0	0	0
0	0	0	0	0	1	1	0	0	0	0	1	1	0	0	0
0	1	0	0	0	1	0	0	1	0	0	0	0	0	0	3
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	1

J_5															
0	0	0	0	0	0	1	0	0	0	1	0	0	0	0	0
0	0	0	0	2	0	0	0	3	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	2	0	0	0	0	0	0	1	0	0	0	0	2	0

J_6															
1	0	0	1	1	0	0	3	0	0	0	0	1	0	0	1
0	0	0	0	1	1	0	0	0	0	0	0	0	0	0	0
0	1	0	0	1	1	0	1	0	0	0	0	0	0	0	0
1	0	0	1	1	0	1	1	0	0	0	0	0	0	0	0

J_7															
0	0	2	1	0	3	0	0	0	1	0	0	0	0	0	0
0	1	0	0	0	0	0	0	0	0	0	1	0	0	0	1
0	0	2	0	0	0	2	0	0	0	0	1	2	1	1	0
0	0	0	0	0	0	0	0	0	0	1	0	0	0	1	1

J_8															
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
0	0	0	0	0	0	0	0	1	0	1	0	0	1	0	1
0	3	0	0	0	0	1	0	0	0	0	0	0	0	0	1

Trong số 8 mảng bộ đếm (trong 8 mangt ở trên) có duy nhất một bộ đếm có giá trị 3, các vị trí của các bộ đếm này sẽ được xác định các bít khoá trong J_1,.. ., J_8. Các vị trí này tương ứng là 47,5,19,0,24,7,7,49. Đổi các số nguyên sang dạng nhị phân ta nhận được J_1, . . .,J_8:

$J_1 = 101111$
$J_2 = 000101$
$J_3 = 010011$
$J_4 = 000000$
$J_5 = 011000$
$J_6 = 000111$
$J_7 = 000111$
$J_8 = 110001$

Bây giờ ta có thể xây dựng 48 bít của khoá bằng cách nhìn vào bảng khoá đối với vòng 3. Khi đó K có dạng:

0001101 0110001 01?01?0 1?00100
0101001 0000??0 111?11? ?100011

ở đây ta đã bỏ qua các bít kiểm tra chặn lẻ và"?" chỉ bít khoá chưa biết. Khóa đầy đủ (ở dạng hexa gồm cả bít kiểm tra chặn lẻ) là:

1A624C89520DEC46

3.6.2. Tấn công DES 6 vòng

Trong mục này ta sẽ mở rộng các ý tưởng ở trên cho phép tấn công xác suất đối DES 6 vòng. Ý tưởng ở đây là phải chọn cẩn thận một acặp bản rõ với một phép XOR chỉ ra trước rồi xác định các xác suất của một dãy xác định các XOR qua các vong mã. Bây giờ ta sẽ định nghiã một khái niệm quan trọng.

Định nghĩa 3.5.

Cho $n \geq 1$ là một số nguyên. Một đặc trưng n vòng là một danh sách có dạng: $L_0', R_0', L_1', R_1', p_1 \ldots L_n', R_n', p_n$

thảo mãn các tính chất sau:

1. $Li' = R_{i-1}'$ với $1 \le i \le n$.

2. Cho $1 \le i \le n$ và giả sử L_{i-1}, R_{i-1} và L_{i-1}^*, R_{i-1}^* được chọn sao cho $L_{i-1} \oplus L_{i-1}^* = L_{i-1}'$ và $R_{i-1} \oplus R_{i-1}^* = R_{i-1}'$. Giả sử L_i,R_i và L_i^*,R_i^* được tính bằng cách áp dụng một vòng mã hoá của DES; Khi đó xác suất để $L_i \oplus L_i^* = L_i'$ và $R_i \oplus R_i^* = R_i'$ đúng bằng p_i (chú ý rằng xác suất này được tính trên mọi bộ 48 $J = J_1 . . . J_8$ có thể).

Xác suất của đặc trưng này sẽ được xác định bằng tích:

$$p = p_1 \times p_2 \times . . . \times p_n$$

Nhận xét: Giả sử ta chọn L_0,R_0 và L_0^*,R_0^* sao cho $L_0 \oplus L_0^* = L_0'$ và $R_0 \oplus R_0^* = R_0'$. Áp dụng n vòng mã hoá của DES để thu được L_1,. . ., L_n và R_1,. . .,R_n. Khi đó không thể khẳng định rằng xác xuất để $L_i \oplus L_i^* = L_i'$ và $R_i \oplus R_i^* = R_i'$ với mọi i, ($1 \le i \le n$)là $p = p_1 \times . . . \times p_n$. Sở dĩ như vậy vì các bộ 48 trong bảng khoá $K_1 . . . K_n$ không độc lập với nhau (nếu n bộ 48 này được chọn ngẫu nhiên và độc lập với nhau thì khẳng định trên là đúng). Tuy vậy, ta vẫn hy vọng rằng, $p_1 \times .. . \times p_n$ là một ước lượng khá chính xác cho xác suất này.

Cũng cần phải thấy rằng, các xác suất p_i ở một đặc trưng sẽ xác định theo một cặp bản rõ tuỳ ý (nhưng cố định) cho phép XOR xác định trước. Tại đây 48 bít khoá cho một vòng mã DES sẽ thay đổi trên toàn bộ 2^{48} khả năng. Tuy nhiên thám mã lại đang cố gắng xác định một khoá cố định (nhưng chưa biết). Anh ta sẽ chọn ngẫu nhiên các bản rõ (sao cho chúng có các XOR xác định) với hy vọng rằng, các xác suất để các XOR trong n vòng mã phù hợp với các XOR được xác định trong đặc trưng phải khá gần với các p_1,. . . ,p_n tương ứng.

Ví dụ đơn giản trên hình 3.10 là một đặc trưng một vòng, nó là cơ sở cho phép tấn công lên DES 3 vòng (cũng như trước kia, ta dùng biểu diễn hexa). Hình 3.11 mô tả một đặc trưng một vòng khác.

Hình 3.10. Đặc trưng một vòng.

L_0' = bất kì	$R_0' = 00000000_{16}$	
$L_1' = 00000000_{16}$	$R_1' = L_0'$	p = 1

Hình 3.11. Một đặc trưng một vòng khác.

$L_0' = 00000000_{16}$	$R_0' = 60000000_{16}$
$L_1' = 60000000_{16}$	$R_1' = 00808200_{16}$

Ta sẽ xem xét kỹ hơn các đặc trưng trong hình 3.11. Khi $f(R_0,K_1)$ và $f(R_0^*,K_1)$ được tính, bước đầu tiên là phải mở rộng R_0 và R_0^* . Kết quả của phép XOR hai mở rộng này là:

001100. . .0

Bởi vậy XOR vào của S_1 là 001100 và các XOR vào của 7 hộp S khác đều là 000000. Các XOR của S_2 tới S_8 đều là 0000. XOR ra của S_1 sẽ là 1110 với xác suất bằng 14/64 (vì có thể tính được $N_1(001100,1110)= 14$). Như vậy ta được:

$$C' = 11100000000000000000000000000000$$

với xác suất bằng 14/64. Sử dụng P ta có :

$$P(C) \oplus P(C^*) = 00000000100000001000001000000000$$

dưới dạng hexa giá trị này là 00808200_{16}. Khi giá trị này được XOR với L_0' ta sẽ nhận được R_1' chỉ ra với xác suất 14/64. Dĩ nhiên ta luôn có $L_1' = R_0'$.

Việc tấn công DES 6 vòng sẽ dựa trên đặc trưng 3 vòng cho ở hình 3.12.

Hình 3.12. Một đặc trưng 3 vòng.

$L_0' = 40080000_{16}$	$R_0' = 04000000_{16}$	
$L_1' = 04000000_{16}$	$R_1' = 00000000_{16}$	$p = 1/4$
$L_2' = 00000000_{16}$	$R_2' = 04000000_{16}$	$p = 1$
$L_3' = 04000000_{16}$	$R_3' = 40080000_{16}$	$p = 1/4$

Trong tấn công 6 vòng ta sẽ bắt đầu với L_0R_0, $L_0^*R_0^*$, L_6R_6, $L_6^*R_6^*$, trong đó đã chọn các bản rõ để $L_0' = 40080000_{16}$ và $R_0' = 04000000_{16}$. Có thể biểu thị R_6 như sau:

$$\begin{aligned} R_6 &= L_5 \oplus f(R_5,K_6) \\ &= R_4 \oplus f(R_5,K_6) \\ &= L_3 \oplus f(R_3,K_4) \oplus f(R_5,K_6) \end{aligned}$$

R_6^* có thể biểu thị theo cách tương tự và bởi vậy:

$$R_6' = L_3' \oplus f(R_3,K_4) \oplus f(R_3^*,K_4) \oplus f(R_5,K_6) \oplus f(R_5^*,K_6) \qquad (3.4)$$

(hãy chú ý sự tương tự với phép tấn công 3 vòng)

R_6' đã biết. Từ đặc trưng này ta thấy rằng $L_3' = 04000000_{16}$ và $R_3' = 40080000_{16}$ với xác suất1/16. Nếu đây là trường hợp thực tế thì XOR vào của cá hộp S trong vòng 4 có thể tính được theo hàm mở rộng bằng:

$$001000000000000000010000...0$$

Các XOR vào của S_2,S_5,S_6,S_7 và S_8 đều là 000000 và bởi vậy ở vòng 4, các XOR ra của 5 hộp này đều là 0000. Điều đó có nghĩa là có thể tính các XOR ra của 5 hộp S này ở vòng 6 theo (3.4). Bởi vậy giả sử ta tính:

$$C_1'C_2'C_3'C_4'C_5'C_6'C_7'C_8' = P^{-1}(R_6' \oplus 04000000_{16})$$

trong đó mỗi C_i' là một xâu bít có độ dài 4. Khi đó, với xác suất 1/16 các xâu bít C_2',C_5', C_6', C_7' và C_8' tương ứng là các XOR ra của S_2,S_5,S_6,S_7 và S_8 ở vong 6. Các đầu ra của các hộp S ở vòng 6 có thể được tính là E_2, E_5, E_6, E_7, E_8 và E_2^*, E_5^*, E_6^*, E_7^*và E_8^* , trong đó :

$$E_1\ E_2\ E_3\ E_4\ E_5\ E_6\ E_7\ E_8 = E(R_5) = E(L_6)$$

và

$$E_1^*\ E_2^*\ E_3^*\ E_4^*\ E_5^*\ E_6^*\ E_7^*\ E_8^* = E(R_5^*) = E(L_6^*)$$

có thể được tính theo các bản mã như đã mô tả trên hình 3.13.

Hình 3.13. DC đối với DES 6 vòng.

Đầu vào $L_0R_0,L_0^*R_0^*,L_6R_6$,và $L_6^*R_6^*$ trong đó
$L_0' = 40080000_{16}$ và $R_0' = 04000000_{16}$

1. Tính $C' = P^{-1}(R_6' \oplus 40080000_{16})$
2. Tính $E = E(L_6)$ và $E^* = E(L_6^*)$
3. **For** $\in$ j {2, 5, 6, 7, 8} **do**
   Tính $test_j(E_j, E_j^*, C_j')$

Bây giờ ta muốn xác định 30 bít khoá trong J_2, J_5, J_6, J_7 và J_8 như cách đã làm trong tấn công 3 vòng. Vấn đề ở đây là XOR được giả định cho vòng 6 chỉ đúng với xác suất 1/16. Bởi vậy 15/16 thời gian ta chỉ thu được các bít ngẫu nhiên không phải là các bít khoá có thể. Bằng một cách nào đó ta phải có khả năng xác định được các khoá đúng bằng các số liệu đã cho (trong đó có 15/16 các số liệu sai). Điều này có vẻ không sáng sủa cho lắm, song rất may mắn là viễn cảnh của ta không tối tăm như vậy.

Định nghĩa 3.6.

Giả sử $L_0 \oplus L_0^* = L_0'$ và $R_0 \oplus R_0^* = R_0'$. Ta nói rằng cặp bản rõ L_0R_0 và $L_0^*R_0^*$ là cặp đúng ứng với một đặc trưng nếu $L_i \oplus L_i^* = L_i'$ và $R_i \oplus R_i^* = Ri'$ với mọi i, $1 \le i \le n$. Ngược lại, cặp này được xác định là cặp sai.

Hy vọng rằng khoảng 1/16 các cặp là đúng và các cặp còn lại là sai ứng với đặc trưng 3 vòng.

Chiến thuật của ta là tính E_j, E_j^* và C_j' (như đã mô tả ở trên) sau đó xác định các $test_j(E_j, E_j^*, C_j')$ vơi j = 2, 5, 6, 7, 8. Nếu bắt đầu bằng một cặp đúng thì các bít khoá đúng cho mỗi J_j sẽ nằm trong tập $test_j$. Nếu cặp này sai thì giá trị của C_j' sẽ không đúng và giả định rằng mỗi tập $test_j$ sẽ chủ yếu là ngẫu nhiên có thể coi là có lý:

Có thể nhận ra một cặp sai theo phương pháp sau: Nếu $| test_j | = 0$ với bất kì $j \in \{2, 5, 6, 7, 8\}$ thì chắc chắn là ta có một cặp sai. Bây giờ , với một cặp sai cho trước, có thể thấy rằng xác suất để $test_j = 0$ với giá trị j nhất định sẽ xấp xỉ bằng 1/5. Đây là một giả định hợp lý bởi vì $N_j(E_j',C_j') = | test_j |$ và như đã nói ở trên, xác suất để $N_j(E_j',C_j') = 0$ xấp xỉ bằng 1/5. Xác suất để tất cả 5 tập $test_j$ có lực lượng dương được ước lượng bằng $0,8^5 \approx 0,33$. Bởi vậy xác suất để ít nhất một tập $test_j$ có lực lượng bằng 0 sẽ vào khoảng 0,67. Như vậy ta hy vọng loại bỏ được 2/3 số cặp sai bằng cách quan sát đơn giản này (ta sẽ gọi là phép lọc). Tỷ lệ các cặp đúng còn lại sau phép lọc xấp xỉ bằng (1/16)/(1/3) = (3/16).

Ví dụ 3.4.

Giả sử ta có cặp mã - rõ sau:

Bản rõ	Bản mã
86FA1C2B1F51D3BE	1E23ED7F2F553971
C6F21C2B1B51D3BE	296DE2B687AC6340

Nhận thấy rằng $L_0' = 40080000_{16}$ và $R_0' = 04000000_{16}$. Các đầu vào và các đầu ra của các hộp S ở vòng 6 được tính như sau:

j	**E_j**	E_j^*	**C_j'**
2	111100	010010	1101
5	111101	111100	0001
6	011010	000101	0010
7	101111	010110	1100
8	111110	101100	1101

Khi đó tập $test_j$ (2, 5, 6, 7, 8) là:

j	$test_j$
2	14, 15, 26, 30, 32, 33, 48,52
5	
6	7, 24, 36, 41, 54, 59
7	
8	34, 35, 48, 59

Ta thấy tập $test_5$ và $test_7$ là các tập rỗng, bởi vậy cặp này là một cặp sai và sẽ bị loại bỏ bằng phép lọc.

Bây giờ, giả sử rằng ta có một cặp sao cho $| test_j | > 0$, với j = 2, 5, 6, 7, 8, để nó còn tồn tại lại sau phép lọc (tuy nhiên vẫn chưa biết cặp này đúng hay sai). Ta nói rằng xâu bít $J_2\ J_5\ J_6\ J_7\ J_8$ có độ dài 30 là xâu bít được gợi ý bởi cặp trên nếu $J_j \in test_j$ với j = 2,5,6,7,8. Số các xâu bít được gợi ý là:

Thông thường số các xâu bít được gợi ý có giá trị quá lớn (ví dụ: > 80000).

Giả sử ta đã lập bảng tất cả các xâu bít được gợi ý thu được từ N cặp (không bị loại bỏ bởi phép lọc). Với một cặp đúng, xâu bít đúng $J_2\ J_5\ J_6\ J_7\ J_8$ sẽ là một xâu bít được gợi ý. Xâu bít đúng này sẽ xuất hiện vào khoảng 3N/16 lần. Các xâu bít không đúng thường xuất hiện ít hơn nhiều do chúng cơ bản là xuất hiện một cách ngẫu nhiên có 2^{30} khả năng (một số rất lớn).

Việc lập bảng tất cả các xâu bít được ôụi ý sẽ rất cồng kềnh, bởi vậy ta sẽ dùng một thuật toán yêu cầu ít thời gian và không gian (bộ nhớ). Ta có thể mã tập $test_j$ bất kỳ bằng véc tơ T_j có độ dài 64, trong đó tạo độ thứ i của T_j được đặt về giá trị 1 (với $0 \le i \le 63$) nếu xâu bít độ dài 6 (là biểu diễn nhị phân của i) nằm trong tập $test_j$. Trong trường hợp ngược lại, toạ độ thứ i

được đặt về 0 (giống như cách biểu diễn mảng bộ đếm đã dùng trong phép tấn công 3 vòng).

Với mỗi cặp còn lại ta xây dượng các véc tơ này như đã mô tả ở trên về ký hiệu chúng là T_j^i , j = 2,5,6,7,8, $1 \le i \le N$. Với $I \subseteq \{1,. . . ,N\}$ ta nói rằng I là tập được phép nếu với mỗi $j \in \{2,5,6,7,8\}$ có ít nhất một toạ độ bằng $|I|$ trong véc tơ $\sum Tj$ (với $j \in I$).

Nếu cặp thứ i là một cặp đúng với mọi $i \in I$ thì I sẽ là tập được phép. Bởi vậy ta tin rằng sẽ có một tạp được phép với kích thước xấp xỉ 3N/16 chứa các bít khoá đúng và ngoài ra không có một tập nào khác. Có thể dễ dàng xây dựng các tập được phép I bằng một thuật toán đệ quy.

Ví dụ 3.5.

Một số chương trình máy tính đã được thực hiện để kiêmr tra phương pháp này. Trong đó đã tạo ra một mẫu ngẫu nhiên gồm 120 cặp bản rõ với các XOR xác định và các bản rõ này đã được mã hoá bằng cùng một khoá (ngẫu nhiên). Bảng 3.1 đưa ra 120 cặp các bản rõ và các bản mã tương ứng ở dạng mã hexa.

Khi tính các tập được phép ta thu được n_i tập được phép có lực lượng như sau:

i	**n_i**
2	111
3	180
4	231
5	255
6	210
7	120
8	45
9	10
10	1

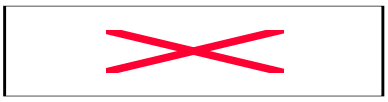
Tập được phép duy nhất có lực lượng 10 là:

{24,29,30,48,50,52,55,83,92,118}

Thực tế tập được tạo ra theo 10 cặp đúng. Chỉ có tập được phép này mới chứa các bít khoá đúng cho J_2, J_5, J_6, J_7, J_8. Chúng có giá trị như sau:

$J_2 = 011001$
$J_5 = 110000$
$J_6 = 001001$
$J_7 = 101010$
$J_8 = 100011$

Chú ý rằng tất cả các tập được phép có lực lượng tối thiểu là 6 không kể 3 tập được phép có lực lượng là 5 sinh ra từ các cặp đúng bởi vì



với $6 \le i \le 10$.

Phương pháp này sẽ cho ta biết 30 bít trong 56 bít khoá. Bằng một đặc trưng 3 vòng khác (nêu ở hình 3.14), ta có thể tính thêm 12 bít khoá nữa (các bít này nằm trong J_1 và J_4). Bây giờ chỉ còn lại 14 bít khoá chưa biết. Vì $2^{14} = 16384$ là một số quá nhỏ nên có thể dùng phép tìm kiếm vét cạn để xác định nốt chúng.

Hình 3.14.

$L_0' = 00200008_{16}$	$R_0' = 00000400_{16}$	
$L_1' = 00000400_{16}$	$R_1' = 00000000_{16}$	$p = 1/4$
$L_2' = 00000000_{16}$	$R_2' = 00000400_{16}$	$p = 1$
$L_3' = 00000400_{16}$	$R_3' = 00200008_{16}$	$p = 1/4$

Toàn bộ khoá (ở dạng hexa, kể cả các bít kiểm tra chẵn lẻ) sẽ là:

34E9F71A20756231

Như đã nói ở trên, 120 cặp được cho ở bảng 3.1. Trong cột thứ hai, dấu (*) kí hiệu cặp đúng, dấu (**) kí hiệu cặp sai nhận biết được và nó sẽ bị loại bỏ bởi phép lọc. Trong số 120 cặp, có 73 cặp được xác định là các cặp sai nhờ quá trình lọc, bởi vậy 47 cặp còn lại sẽ là các cặp đúng có thể.

3.6.3. Các ví dụ khác về DC.

Các kỹ thuật DC có thể được sử dụng để tấn công DES có trên 6 vòng. Với DES 8 vòng cần 2^{14} bản rõ chọn lọc, DES 10, 12, 14, 16 vòng có thể phá được với tương ứng là 2^{24}, 2^{31}, 2^{39} và 2^{47} bản rõ chọn lọc. Vào thời điểm hiện tại, tấn công DES có trên 10 vòng là không thực tế.

Một loại mã tích hoán vị - thay thế khác với DES cũng có thể dùng DC để phá (ở mức độ khác nhau). Trong các hệ này, có một số hệ mật hoán vị - thay thế đã được đưa ra trong những năm gần đây như FEAL,REDOC-II và LOKI.

Ghi chú (của người dịch): theo công bố của Micheal Wiener vào 1993, với 10^7 USD có thể xây dựng thiết bị chuyên dụng để phá DES trong khoảng 21 phút. Với 10^8 USD, các bản tin DES có thể bị phá trong khoảng 2 phút. Như vậy, DES không còn bí mật đối với NAS. Tuy nhiên cũng không cần một thiết bị chuyên dụng đắt tiền như vậy để phá DES. Các thông báo được mã hoá bằng DES có thể bị phá bằng các máy tính thông thường trên với điều kiẹen có vẻ siêu thực: có trên $2^{43} \times 2^6$ bít rõ - mã với một khoá 56 bít cố định, tuy nhiên bạn phải chờ lâu hơn.

Trong hội nghị CRYPTO'94 M.Matsui đã trình bày một kỹ thuật phá DES mới được gọi là " thám mã tuyến tính". Sử dụng 2^{43} (8.796.093.022.208) bản mã đã biết. Matsui có thể phá một khoá DES trong 50 ngày bằng một máy tính cá nhân.

3.7. Các chú giải và tài liệu dẫn.

Smid và Branstad đã có một bài báo hay về lịch sử của DES [SB 92]. Các công bố về Chuẩn xử lý thông tin liên bang (FIPS) liên quan đến DES gồm: mô tả DES [NBS 77]; ứng dụng và sử dụng DES [NBS 81]; các chế độ làm việc của DES [NBS 80]; xác thực bằng DES [NBS 85].

Một số tính chất của các hộp S được Brickell, Moore và Purtill [BMP87] nghiên cứu. Chíp giải mã DES được mô tả trong [EB 93]. Thiết bị tìm khoá của Wiener được mô tả ở CRYPTO' 93 [Wi 94]. Phép tối ưu hoá thời gian - bộ nhớ tổng quát hơn được trình bày bởi Fiat và Naor trong [FN 91]. Kỹ thuật DC được phát triển bởi Biham và Shamir [BS 91] (cũng có thể xem [BS93A] và sách của họ [BS 93] trong đó cũng thảo luận thám mã hệ mật khác). Cách trình bày về DC trong chương này phần lớn dựa trên [BS93]. Một phương pháp mã thám mới có thể dùng để tấn công DES và các hệ mật tương ứng khác là phương pháp thám mã tuyến tính của Matsui [MA 94], [MA 94A].

Các mô tả về hệ mật hoán vị - thay thế khác có thể tìm trong các tài liệu sau: LUCIFER [FE 73], FEAL [MI 91], REDOC-II [CW 91] và LOKI [BKPS 90].

Bảng 3.1. Mã thám DES 6 vòng.

CẶP	CẶP ĐÚNG	BẢN RÕ	BẢN MÃ
1	**	86FA1C2B1F51D3BE C6F21CC2B1B51D3BE	1E23ED7F2F551971 296DE2BG87AC6310
2	**	EDC439EC935E1ACD ADCC39EC975E1ACD	0F847EFE90466588 93E84839E374440B
3	**	9468A0BE00166155 D460A0BE04166155	3D6A906A6566D0BF 3BC3B23698379E1
4	**	D4FF2B18A5A8AACB 94F72B18A1A8AAC8	26B14738C2556BA4 15753FDE86575ABF

BÀI TẬP

3.1.Hãy chứng minh rằng phép giải mã DES có thể thực hiện bằng cách áp dụng thuật toán mã hoá DES cho bản rõ với bảng khoá đảo ngược.

3.2.Cho DES(x,K) là phép mã hoá DES của bản rõ x với khoá K. Giả sử y = DES(x,K) và y' = DES(c(x),c(K)) trong đó c(.) kí hiệu là phần bù theo các bít của biến. Hãy chứng minh rằng y' = c(y) (tức là nếu lấy phần bù của bản rõ và khoá thì bản mã kết quả cũng là phần bù của bản mã ban đầu). Chú ý rằng kết quả trên có thể chứng minh được chỉ bằng cách sử dụng mô tả "mức cao" của DES - cấu trúc thực tế của các hộp S và các thành phần khác của hệ thống không ảnh hưởng tới kết quả này.

3.3.Mã kép là một cách để làm mạnh thêm cho DES: với hai khóa K_1 và K_2 cho trước, ta xác định $y = e_{K2}(e_{K1}(x))$ (dĩ nhiên đây chính là tích của DES với chính nó. Nếu hàm mã hoá e_{K2} giống như hàm giải mã d_{K1} thì K_1 và K_2 được gọi là các khoá đối ngẫu (đây là trường hợp không mong muốn đối với phép mã kép vì bản mã kết quả lại trùng với bản rõ). Một khoá được gọi là tự đối ngẫu nếu nó đối ngẫu với chính nó.

a/ Hãy chứng minh rằng nếu C_0 gồm toàn các số 0 hoặc gồm toàn các số 1 và D_0 cũng vậy thì K là tự đối ngẫu.

b/ Hãy tự chứng minh rằng các khoá sau (cho ở dạng hexa) là tự đối ngẫu:

0101010101010101
FEEFEFEFEFEFEFE
1F1F1F1F0E0E0E0E
E0E0E0E0F1F1F1F1

c/ Hãy chứng tỏ rằng nếu C_0 = 0101. . . 01 hoặc 1010. . .10 (ở dạng nhị phân) thì XOR các xâu bít C_i và C_{17-i} là 111. . .11, vơi $1 \le i \le 16$ (khẳng định tương tự cũng đúng đối với D_i).

d/ Hãy chứng tỏ các cặp khoá sau là đối ngẫu:

E001E001F101F101	01E001E001F101F1
FE1FFE1FF0EFE0E	1FFE1FFE0EFE0EFE
E01FE01FFF10FF10	1FE01FE00EF10EF1

3.4.Có thể tạo một mã xác thực thông báo bằng chế độ CFB cũng như chế độ CBC. Cho dãy các khối bản rõ x_1. . .x_n , giả sư ta xác định véc tơ khởi đầu IV là x_1 . Sau đó mã hoá x_2. . .x_n bằng khoá K ở chế độ CFB để thu được

$y_1...y_{n-1}$ (chú ý rằng chỉ có n-1 khối bản mã). Cuối cùng xác định $e_K(y_{n-1})$ làm MAC. Hãy chứng minh rằng MAC này đồng nhất với MAC được tạo trong phần 3.4.1. dùng chế độ CBC.

3.5.Giả sử một dãy các khối bản rõ $x_1. . .x_n$ được mã hoá bằng DES, tạo ra các khối bản mã $y_1. . .y_2$. Giả sử rằng một khối bản mã (chẳng hạn y_i) bị phát sai (tức là có một số số 1 bị chuyển thành số 0 và ngược lại). Hãy chỉ ra rằng số các khối bản rõ bị giải mã không đúng bằng một nếu ta dùng các chế độ ECB và OFB để mã hoá; và bằng hai nếu dùng các chế độ CBC và CFB để mã hoá.

3.6.Bài tập này nhằm nghiên cứu một phép tối ưu hoá thời gian - bộ nhớ đơn giản đối với phép tấn công bản rõ chọn lọc. Giả sử có một hệ mật trong đó $\mathcal{P} = \mathcal{C} = \mathcal{K}$ và đạt được độ mật hoàn thiện. Khi đó $e_K(x) = e_{K1}(x)$ có nghĩa là $K = K_1$. Kí hiệu $P = Y = \{y_1,. . .,y_N\}$. Cho x là bản rõ cố định. Định nghĩa hàm g: $Y \rightarrow Y$ theo quy tắc $g(y) = e_y(x)$. Ta xác định một đồ thì có hướng G chứa tập đỉnh Y, trong đó tập cạnh chứa tất cả các cạnh có hướng có dạng $(y_i,g(y_i))$, $1 \le i \le N$.

a/ Hãy chứng minh rằng G gồm tất cả các chu trình có hướng không liên thông.

b/ Cho T là một tham số thời gian mong muốn. Giả sử ta có một tập các phần tử $Z = \{z_1,. . .,z_m\} \subseteq Y$ sao cho với mỗi phần tử $y_i \in Y$ nằm trong một chu trình có độ dài tối đa là T hoặc tồn tại một phần tử $z_j \neq y_i$ sao cho khoảng cách tử y_i tới z_j trong G tối đa là T. Hãy chứng tỏ rằng tồn tại một tập Z như vậy sao cho: $| Z | \le 2N/T$ và như vậy $| Z | = 0(N/T)$.

c/ Với mỗi $z_j \in Z$ ta xác định $g^{-T}(z_j)$ là phần tử y_i sao cho $g^T(y_i) = z_j$, trong đó g^T là một hàm gồm T phép lặp của g. Hãy xây dựng một bảng X gồm các cặp $(z_i,g^{-T}(z_j))$ được sắp xếp theo các toạ độ đầu của chúng.

Một mô tả giả mã của một thuật toán tìm K với $y = e_K(x)$ cho trước được trình bày ở hình 3.15. Hãy chứng tỏ thuật toán này tìm K trong tối đa là T bước (bởi vậy cỡ của phép tối ưu hoá thời gian - bộ nhớ là 0(N)).

Hình 3.15. Phép tối ưu hoá thời gian - bộ nhớ.

1. $Y_{start} = y$
2. Backup = **false**
3. **While** $g(y) \neq y_{start}$ **do**
4. **if** $y = z_j$ với mỗi j nào đó **and not** backup **then**
5. $y = g^{-T}(z_j)$
6. backup = **true**

else

7. $y = g(y)$
8. $K = y$

d/ Hãy mô tả thuật toán giải mã để xây dựng một tập Z mong muốn trong thời gian 0(NT) không dùng một mảng có kích thước N.

3.7. Hãy tính các xác suất của đặc trưng 3 vòng sau:

$L_0' = 00200008_{16}$ $R_0' = 00000400_{16}$
$L_1' = 00000400_{16}$ $R_1' = 00000000_{16}$ p = ?
$L_2' = 00000000_{16}$ $R_2' = 00000400_{16}$ p = ?
$L_3' = 00000400_{16}$ $R_3' = 00200008_{16}$ p = ?

3.8. Sau đây là một phép tấn công vi sai đối với DES 4 vòng sử dụng đặc trưng sau (đây là một trường hợp đặc biệt của đặc trưng được trình bày ở hình 3.10).

$L_0' = 20000000_{16}$ $R_0' = 00000000_{16}$
$L_1' = 00000000_{16}$ $R_1' = 20000000_{16}$ p = 1

a/ Giả sử rằng thuật toán sau (được nêu ở hình 3.16) được dùng để tính các tập $test_2$,. . .,$test_8$. Hãy chứng tỏ rằng $J_j \in test_j$ với $2 \leq j \leq 8$.

Hình 3.16. Tấn công DC lên DES 4 vong.

Vào : L_0R_0, $L_0^*R_0^*$, L_3R_3 và $L_3^*R_3^*$, trong đó
$L_0' = 10000000_{16}$ và $R_0' = 00000000_{16}$
1. Tính $C' = P^{-1}(R_4')$
2. Tính $E = E(L_4)$ và $E^* = E^*(L_4^*)$
3. **For** j =2 **to** 8 **do**
Tính testj(E_j,E_j^*,C_j')

b/ Với các cặp bản rõ - mã sau, hãy xác định các bít khoá trong $J_2,...,J_8$.

Bản rõ	Bản mã
18493AC485B8D9A0	E332151312A18B4F
38493AC485B8D9A0	87391C27E5282161
482765DDD7009123	B5DDD833D82D1D1
682765DDD7009123	81F4B92BD94B6FD8
ABCD098733731FF1	93A4B42F62EA59E4
8BCD098733731FF1	ABA494072BF411E5
13578642AAEDCB	FDEB526275FB9D94
33578642AAFFEDCB	CC8F72AAE685FDB1

c/ Hãy tính toàn bộ khoá (14 bít khoá còn lại cần phải xác định có thể tìm theo phương pháp tìm kiếm vét cạn).

CHƯƠNG 4
KIỂM TRA TÍNH NGUYÊN TỐ XÁC SUẤT

Để thiết lập hệ mật RSA, ta phải tạo ra các số nguyên tố ngẫu nhiên lớn (chẳng hạn có 80 chữ số). Trong thực tế, phương cách thực hiện điều này là: trước hết phải tạo ra các số ngẫu nhiên lớn, sau đó kiểm tra tính nguyên thuỷ của chúng bằng cách dùng thuật toán xác suất Monte- Carlo thời gian đa thức (chẳng hạn như thuật toán Miller- Rabin hoặc là thuật toán Solovay- Strasen). Cả hai thuật toán trên đều được trình bày trong phần này. Chúng là các thuật toán nhanh (tức là một số nguyên n được kiểm tra trong thời đa thức theo $\log_2 n$, là số các bít trong biểu diện nhị phân của n). Tuy nhiên, vẫn có khả năng là thuận toán cho rằng n là số nguyên tố trong khi thực tế n là hợp lệ số. Bởi vậy, bằng cách thay đổi thuật toán nhiều lần, có thể giảm xác suất sai số dưới một mức ngưỡng cho phép (sau này sẽ thảo luận kỹ hơn một chút về vấn đề này).

Một vấn đề quan trọng khác: là cần phải kiểm tra bao nhiêu số nguyên ngẫu nhiên (với kích thươc xác định)cho tới khi tìm được một số nguyên tố. Một kết quả nỗi tiếng trong lý thuyết số (được gọi là định lý số nguyên tố) phát biểu rằng: số các số nguyên tố không lớn hơn N xấp xỉ bằng N/ln N. Bởi vậy, nếu p được chọn ngẫu nhiên thì xác suất p là một số nguyên tố sẽ vào khoảng 1/ln p. Với một mođun 512 bít, ta có $1/\ln p \approx 1/77$. Điều này có nghĩa là tính trung bình, cư 177 số nguyên ngẫu nhiên p với kích thước tương ứng sẽ có một số là số nguyên tố. Dĩ nhiên, nếu chĩ hạn chế xét các số nguyên lẻ thì xác suất sẽ tăng gấp đôi tới khoảng 2/177). Bởi vậy trên thực tế, hoàn toàn có khả năng tạo được các nguyên tố đủ lớn và do đó về mặt thực thể ta có thể thiết lập được một hệ mật RSA. Sau đây sẽ tiếp tục xem xét điều này được thực hiên như thế nào.

Một bài toán quyết định là một bài toán toán trong đó một câu hỏi cần được trả lời "có" hoặc "không". Một thuật toán xác suất là một thuật toán bất kỳ có sử dụng các số ngẫu nhiên (ngược lại, thuật toán không sử dụng các số ngẫu nhiên sẽ được gọi là một thuật toán tất định). Các định nghĩa sau có liên quan tới các thuật toán xác suất cho các bài toán quyết định.

Định nghĩa 4.1

Thuật toán Monte Carlo định hướng "có" là một thuật toán xác suất cho một bài toán quyết định, trong đó câu trả lời "có" luôn luôn là đúng còn câu trả lời "không" có thể là sai. Thuật toán Monte Carlo định hướng "không" cũng được định nghĩa theo cách tương tự.

Chúng ta nói rằng, một thuật toán Monte Carlo định hướng "có" có xác suất sai bằng ε nếu với bất kỳ mốt trường hợp nào mà câu trả lời là "có" thì thuật toán có câu trả lời sai "không" với xác suất không lớn hơn ε (xác suất này được tính trên mọi phép chon ngẫu nhiên, có thể thực hiên bởi thuật toán với một câu vào đã cho).

Bài toán quyết định ở đây là bài toán hợp lệ số mô tả ở hình 4.5.
Cần chú ý rằng một thuật toán quyết định chỉ có câu trả lời "có" hoặc "không" đặc biệt trong bài toán hợp lệ số là ta không yêu cầu thuật toán tính tích thừa số khi n là hợp lệ số.

Trước tiên ta sẽ mô tả thuật toán Soloway- Strasson. Đây là một thuật toán Monte- Carlo định hướng "có" cho bài toán hợp số có
Trước tiên ta sẽ mô tả thuật toán Soloway- Strasson. Đây là một thuật toán Monte-Carlo định hướng "có" cho bài toán hợp số và xác xuất sai 1/2. Bởi vậy, nếu thuật toán trả lời "có" thì n là hợp số; ngược lại nếu n là hợp số thì thuật toán trả lời "có" với xác xuất tối thiểu 1/2.

Hình 4.5. Bài toán hợp số.

Đặc trưng của bài toán: một số nguyên dương $n \geq 2$ Câu hỏi: n có phải là hợp số không ?

Hình 4.6. Bài toán về các thặng dư bậc hai.

Đặc trưng của bài toán: cho p là một số nguyên tố lẻ và một số nguyên x sao cho $0 \leq x \leq p-1$ Câu hỏi: x có phải là thặng dư bậc hai phép modulo p ?

Mặc dù thuật toán Miller-Rabin (ta sẽ xét sau) nhanh hơn thuật toán Soloway-Strasson (S-S) nhưng ta sẽ xét thuật toán S-S trước vì nó dễ hiểu hơn về khái niệm, đồng thời lại liên quan tới một số vấn đề của lý thuyết số (mà ta sẽ còn dùng trong các chương trình sau). Ta sẽ xây dựng một số nền tảng sâu sắc hơn trong lý thuyết số trước khi mô tả thuật toán.

Định nghĩa 4.2.

Giả sử p là một số nguyên tố lẻ và x là một số nguyên, $1 \le x \le p-1$. x được gọi là thặng dư bậc hai theo modulo p nếu phương trình đồng dư $y^2 \equiv x$ (modulo p) có một nghiệm $y \in Z_p$ x được gọi là thặng dư không bậc hai theo modulo p nếu x ??? 0 (mod p) và x không phải là thặng dư bậc hai theo modulo p.

Ví dụ 4.6.

Các thặng dư bậc hai theo modulo 11 là 1,3,4,5 và 9. Cần để ý rằng, $(\pm1)^2=1$, $(\pm5)^2=3$, $(\pm2)^2=4$, $(\pm4)^2=5$, $(\pm3)^2=9$ (ở đây tất cả các phép số học đều thực hiện trong Z_{11}).

Bài toán quyết định thặng dư bậc hai được trình bày trên hình 4.6 sẽ được thấy một cách tươnngf minh như sau:

Trước hết, ta sẽ chứng minh một kết quả- *tiêu chuẩn Euler* – tạo nên thuật toán tất định theo thời gian đa thức cho bài toán về các thặng dư bậc hai.

Định lý 4.8. (Tiêu chuẩn Euler)

Giả sử p là một số nguyên tố, khi đó x là một thặng dư bậc hai theo modulo p khi và chỉ khi:

$$x^{(p-1)/2} \equiv 1 \pmod p$$

Chứng minh:

Trước hết giả sử rằng, $x \equiv y^2 \pmod p$. Theo hệ quả 4.6, nếu p là số nguyên tố thì $x^{p-1} \equiv 1 \pmod p$ với mọi $x \not\equiv 0 \pmod p$. Bởi vậy ta có :

$$\begin{aligned} x^{(p-1)/2} &\equiv (y^2)^{(p-1)/2} \pmod p \\ &\equiv y^{p-1} \pmod p \\ &\equiv 1 \pmod p \end{aligned}$$

Ngược lại, giả sử rằng $x^{(p-1)/2} \equiv 1 \pmod p$. Cho p là một phần tử nguyên thuỷ theo modulo p. Khi đó $x \equiv b^i \pmod p$ với giá trị i nào đó. Ta có

$$\begin{aligned} x^{(p-1)/2} &\equiv (b^i)^{(p-1)/2} \pmod p \\ &\equiv b^{i(p-1)/2} \pmod p \end{aligned}$$

Vì p có bậc bằng p-1 nên p-1 phải là ước của i(p-1)/2. Bởi vậy i là số chẵn và như vậy căn bậc hai của x là $\pm b^{i/2}$.□

Định lý 4.8 sẽ dẫn tới một thuật toán thời gian đa thức cho các thặng dư bậc hai nhờ sử dụng kỹ thuật "bình phương và nhân" cho phép lấy luỹ thừa theo modulo p. Độ phức tạp của thuật toán khoảng $O((\log p)^3)$.

Sau đây tiếp tục đưa ra một số định nghĩa từ lý thuyết số:

Định nghĩa 4.3.

Giả sử p là số nguyên tố lẻ. Với một số nguyên tố bất kỳ a ≥0, ta định nghĩa ký hiệu Legendre $\left(\frac{a}{p}\right)$ như sau:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 \text{ nếu } a \equiv 0 \pmod p \\ 1 \text{ nếu là thăng dư bậc hai theo modulo p} \\ -1 \text{ nếu là thăng dư không bậc hai theo modulo p} \end{cases}$$

Ta đã biết là $a^{(p-1)/2} \equiv 1 \pmod p$ khi và chỉ khi a là một thặng dư bậc hai theo modulo p. Nếu a là bội của p thì rõ ràng $a^{(p-1)/2} \equiv 0 \pmod p$. Cuối cùng, nếu a là một thặng dư không bậc hai theo modulo p thì $a^{(p-1)} \equiv -1 \pmod p$ vì $a^{p-1} \equiv 1 \pmod p$. Bởi vậy, ta có kết quả cho phép xây dựng một thuật toán hữu hiệu để đánh giá các ký hiệu Legendre như sau

Định Lý 4.9.

Giả sử p là một số nguyên tố. Khi đó $\left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod p$.

Sau đây là một định nghĩa tổng quát hoá cho ký hiệu Legendre.

Định nghĩa 4.4.

Giả sử n là một số nguyên dương lẻ và phân tích theo các luỹ thừa ngu $\left(\frac{a}{r}\right)$ của n là $p_1^{e1} \ldots\ldots p_K^{ek}$. Giả sử a ≥ 0 là một số nguyên. Ký hiệu Jacobi được định nghĩa như sau:

$$\left(\frac{a}{n}\right) = \prod_{i=1}^{K}\left(\frac{a}{p_i}\right)^{ei}$$

Ví dụ 4.7.

Xét ký hiệu Jacobi $\left(\frac{6278}{9975}\right)$ Phân tích luỹ thừa nguyên tố của 9975 là: $9975=3 \times 5^2 \times 7$ vậy ta có:

$$\left(\frac{6278}{9975}\right) = \left(\frac{6278}{3}\right)\left(\frac{6278}{5}\right)^2\left(\frac{6278}{7}\right)\left(\frac{6278}{19}\right)$$

$$= \left(\frac{2}{3}\right)\left(\frac{3}{5}\right)^2\left(\frac{6}{7}\right)\left(\frac{8}{19}\right)$$

$$=(-1)(-1)^2(-1)(-1)$$

$$= -1.$$

Giả sử n > 1 là một số lẻ. Nếu n là một số nguyên tố thì ≡

$a^{(n-1)/2} \pmod n$ với a bất kỳ. Mặt khác nếu n là một hợp số thì đồn $\left(\frac{a}{n}\right)$ thức trên có thể đúng hoặc không. Nếu phương trình đó vẫn đún a được gọi
là số giả nguyên tố Euler theo cơ số n. Ví dụ: 10 là số giả nguyên tố Euler

theo cơ số 91 vì $\left(\frac{10}{91}\right)$ = $-1 = 10^{45} \bmod 91$

Tuy nhiên có thể chứng tỏ rằng, với một hợp số lẻ n bất kỳ, sẽ cóp nhiều nhất một nửa các số nguyên a (sao cho $1 \le a \le n-1$) là các số giả nguyên tố Euler cơ số n (xem các bài tập). Điều đó chứng tỏ rằng, việc kiểm tra tính nguyên tố theo thuật toán Soloway-Strasson

được nêu ở hình 4.7 là thuật toán Monte-Carlo định hướng "có"với xác xuất sai tối đa là 1/2.
Đến đây vẫn chưa xác định rõ thuật toán ttrên có theo thời gian đa thức hay không.

Ta đã biết cách đánh giá $a^{(n-1)/2} \pmod n$ trong thời gian đa thức $O((\log n)^3)$, tuy nhiên cần phải làm thế nào để tính các ký hiệu Jacobi một cách có hiệu quả. Vì ký hiệu Jacobi được xác định theo các thừa số trong phân tích của n. Tuy nhiên nếu có thể phân tích được n thì ta đã biết nó có phải là số nguyên tố hay không, bởi vậy cách làm này sẽ dẫn tới một vòng luẩn quẩn.

Hình 4.7. Thuật toán kiểm tra tính nguyên tố Solova-Strassen với số nguyên lẻ n.

1. Chọn một số nguyên ngẫu nhiên a, $1 \le a \le n-1$

2. Nếu $\equiv a^{(n-1)/2} \pmod n$ thì
   Nếu k $\left(\frac{a}{n}\right)$ rả lời " n là số nguyên tố "
   rả lời " n là một hợp số "

Rất may là có thể đánh giá ký hiệu Jacobi mà không cần phải phân tích n nhờ sử dụng một số kết quả của lý thuyết số, trong đó kết quả quan trọng nhất là tính chất 4 (tổng quát hoá luật tương hỗ bậc hai).
Ta sẽ liệt kê mà không chứng minh các tính chất này.

1. Nếu n là một số nguyên tố lẻ và $m_1 \equiv m_2 \pmod n$ thì:

$$\left(\frac{m_1}{n}\right) \qquad \left(\frac{m_2}{n}\right)$$

2. Nếu n là một số nguyên lẻ thì

$$= \begin{cases} 1 & \text{nếu } n \equiv \pm 1 \pmod 8 \\ -1 & \text{nếu } n \equiv \pm 3 \pmod 8 \end{cases}$$

3. Nếu n là một số nguyên lẻ thì

$$\left(\frac{m_1 m_2}{n}\right)=\left(\frac{m_1}{n}\right)\left(\frac{m_2}{n}\right)$$

Đặc biệt nếu $m=2^k t$ với t là một số lẻ thì:

$$\left(\frac{m}{n}\right) = \left(\frac{2}{n}\right)^k \left(\frac{t}{n}\right)$$

4. Giả sử m và n là các số nguyên lẻ, khi đó:

$$\left(\frac{2}{n}\right) = \begin{cases} -\left(\frac{n}{m}\right) & \text{nếu } m \equiv n \equiv 3 \pmod 4 \\ \left(\frac{n}{m}\right) & \text{trong các trường hợp còn lại} \end{cases}$$

ví dụ

Để minh hoạ cho việc áp dụng các tính chất trên , ta sẽ đánh giá kí

hiệu Jacobi $\left(\frac{7411}{9283}\right)$ như trong bảng dưới đây. Cần chú ý là trong ví dụ

này, ta đã sử dụng liên tiếp các tính chất4, 1,3 ,và 2.

Nói chung, bằng cách áp dụng 4 tính chất trên, có thể tính toánkí

hiệu Jacobi $\left(\frac{m}{n}\right)$ thời gian đa thức. Các phép tính số học dùng ở đây

chỉ là rút gọn theo modulo và phân tích ra các luỹ thừa của thuật toán được biểu diễn dưới dạng nhị phân thì việc phân tích ra các luỹ thừa của hai số chính là việc xác định số các số 0 tiếp sau. Bởi vậy, độ phức tạp của thuật toán được xác định bởi số các phép rút gọn theo modulo cần tiến hành. Không khó khăn lắm có thể chứng tỏ rằng, cần thực hiện nhiều nhất là.

$$\left(\frac{7411}{9283}\right) = -\left(\frac{9283}{7411}\right) \quad \text{theo tính chất 4}$$

$$= -\left(\frac{1872}{7411}\right) \quad \text{theo tính chất 1}$$

$$= -\left(\frac{2}{7411}\right)^4\left(\frac{117}{7411}\right) \quad \text{theo tính chất 3}$$

$$= -\left(\frac{117}{7411}\right) \quad \text{theo tính chất 2}$$

$$= -\left(\frac{7411}{117}\right) \quad \text{theo tính chất 4}$$

$$= -\left(\frac{40}{177}\right) \quad \text{theo tính chất 1}$$

$$= -\left(\frac{2}{117}\right)^3\left(\frac{5}{117}\right) \quad \text{theo tính chất 3}$$

$$= \left(\frac{5}{117}\right) \quad \text{theo tính chất 2}$$

$$= \left(\frac{117}{5}\right) \quad \text{theo tính chất 4}$$

$$= \left(\frac{2}{5}\right) \quad \text{theo tính chất 1}$$

$$= -1 \quad \text{theo tính chất 2}$$

Cần phải thận trọng hơn khi sự dụng các tính toán xác suất. Ta sẽ định nghĩa các biến ngẫu nhiên sau:

a- Chỉ sự kiện " số nguyên lẻ n có kích thước đã định là một hợp số".
b- Chỉ sự kiện " thuật toán trả lời n là số nguyên tố m lần liên tiếp .

Điều chắc chắn là prob(b| a)2^{-m}. Tuy nhiên xác suất mà ta thực sự quan tâm là prob(a/b), xác suất này thường không giống như prob(b/a).

CHƯƠNG 5

CÁC HỆ MẬT KHOÁ CÔNG KHAI KHÁC

Trong chương này ta sẽ xem xét một số hệ mật khoá công khai khác. Hệ mật Elgamal dựa trên bài toán logarithm rời rạc là bài toán được dùng nhiều trong nhiều thủ tục mật mã. Bởi vậy ta sẽ dành nhiều thời gian để thảo luận về bài toán quan trọng này. ở các phần sau sẽ xem xét sơ lược một số hệ mật khoá công khai quan trọng khác bao gồm các hệ thoóng loại Elgamal dựa trên các trường hữu hạn và các đường cong elliptic, hệ mật xếp ba lô Merkle-Helman và hệ mật McElice.

5.1. HỆ MẬT ELGAMAL VÀ CÁC LOGARITHM RỜI RẠC.

Hệ mật Elgamal được xây dựng trên bài toán logảithm rời rạc . Chúng ta sẽ bắt đầu băng việc mô tả bài toán bài khi thiết lập môi trường hữu hạn Z_p, p là số nguyên tố (hình 5.1) (Nhớ lại rằng nhóm nhân Z_p^* là nhóm cyclic và phần tử sinh của Z_p^* được gọi là phần tử nguyên thuỷ).

Bài toán logarithm rời rạc trong Zp là đối tượng trong nhiều công trình nghiên cứu và được xem là bài toán khó nếu p được chọn cẩn thận. Cụ thể không có một thuật toán thời gian đa thức nào cho bài toán logarithm rời rạc. Để gây khó khăn cho các phương pháp tấn công đã biết p phải có ít nhất 150 chữ số và (p-1) phải có ít nhất một thừa số nguyên tố lớn. Lợi thế của bài toán logarithm rời rạc trong xây dượng hệ mật là khó tìm được các logarithm rời rạc ,song bài toán ngược lấy luỹ thừa lại có thể tính toán hiệu quả theo thuật toán "bình phương và nhân". Nói cách khác , luỹ thừa theo modulo p là hàm một chiều với các số nguyên tố p thích hợp.

Elgamal đã phát triển một hệ mật khoá công khai dựa trên bài toán logarithm rời rạc. Hệ thống này được trình bày trên hình 5.2.

Hệ mật này là một hệ không tất định vì bản mã phụ thuộc vào cả bản rõ x lẫn giá trị ngẫu nhiên k do Alice chọn. Bởi vậy, sẽ có nhiều bản mã được mã từ cùng bản rõ.

Hình 2.6 Bài toán logarithm rời rạc trong Zp

Đặc trương của bài toán: I = (p,α,β) trong đó p là số nguyên tố,
α ∈ Zp là phần tử nguyên thuỷ , β ∈ Zp^*
Mục tiêu:Hãy tìm một số nguyên duy nhất a, 0 ≤ a ≤ p-2 sao cho:

$$\alpha^a \equiv \beta \pmod p$$

Ta sẽ xác định số nguyên a bằng $\log_\alpha \beta$

Hình 2.7 Hệ mật khoá công khai Elgamal trong Zp^*

Cho p là số nguyên tố sao cho bài toán logarithm rời rạc trong Zp là khó giải. Cho α ∈ Zp^* là phần tử nguyên thuỷ.Giả sử $P = Zp^*$, $C = Zp^* \times Zp^*$. Ta định nghĩa:

$$K = \{(p, \alpha, a, \beta): \beta \equiv \alpha^a \pmod p\}$$

Các giá trị p, α,β được công khai, còn a giữ kín

Với K = (p, α,a,β) và một số ngẫu nhiên bí mật k ∈ Z_{p-1}, ta xác định:

$$e_k(x,k) = (y_1, y_2)$$

trong đó

$$y_1 = \alpha^k \bmod p$$

$$y_2 = x\beta^k \bmod p$$

với $y_1, y_2 \in Zp^*$ ta xác định:

$$d_k(y_1, y_2) = y_2 (y_1^a)^{-1} \bmod p$$

Sau đây sẽ nmô tả sơ lược cách làm việc của hệ mật Elgamal .Bản rõ x được "che dấu" bằng cách nhân nó với β^k để tạo y_2. Giá trị α^k cũng được gửi đi như một phần của bản mã. Bob -người biết số mũ bí mật a có thể tính được β^k từ α^k . Sau đó anh ta sẽ "tháo mặt nạ" bằng cách chia y_2 cho β^k để thu được x.

Ví dụ 5.1

Cho p = 2579, $\alpha = 2$, a = 765. Khi đó

$$\beta = 2^{765} \bmod 2579 = 949$$

Bây giờ ta giả sử Alice muốn gửi thông báo x = 1299 tới Bob. Giả sử số ngẫu nhiên k mà cô chọn là k = 853. Sau đó cô ta tính

$$y_1 = 2^{853} \bmod 2579$$
$$= 435$$
$$y2 = 1299 \times 949853 \bmod 2579$$
$$= 2396$$

Khi đó Bob thu được bản mã y = (435,2396), anh ta tính

$$x = 2396 \times (435^{765})^{-1} \bmod 2579$$
$$= 1299$$

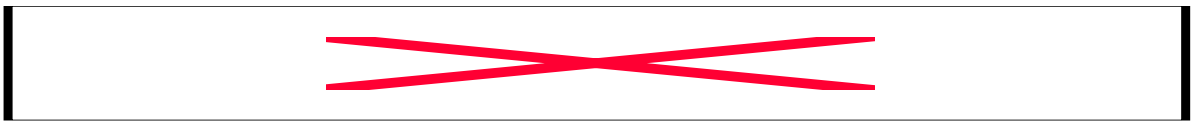
Đó chính là bản rõ mà Alice đã mã hoá.

5.1.1. Các thuật toán cho bài toán logarithm rời rạc.

Trong phần này ta xem rằng p là số nguyên tố, α là phần tử nguyên thuỷ theo modulo p. Ta thấy rằng p và α là các số cố định. Khi đó bài toán logarithm rời rạc có thể được phát biểu dưới dạng sau: tìm một số mũ a duy nhất, $0 \le a \le p-2$ sao cho $\alpha^a \equiv \beta \pmod p$, với $\beta \in Z_p^*$ cho trước.

Rõ ràng là bài toán logarithm rời rạc (DL) có thể giải bằng một phép tìm kiếm vét cạn với thời gian cỡ O(p) và không gian cỡ O(1) (bỏ qua các thừa số logarithm). Bằng cách tính toán tất cả các giá trị α^a có thể và sắp xếp các cặp có thứ tự (a, α^a mod p) có lưu ý đến các tạo độ thứ hai của chúng, ta có thể giải bài toán DL với thời gian cỡ O(1) bằng O(p) phép tính toán trước và O(p) bộ nhớ (vẫn bỏ qua các thừa số logarithm). Thuật toán không tầm thường đầu tiên mà chúng ta sẽ mô tả là thuật toán tối ưu hoá thời gian - bộ nhớ của Shanks.

Thuật toán Shanks



Hình 5.3. Thuật toán Shanks cho bài toán DL.

1. Tính $\alpha^{mj} \bmod p$, $0 \le j \le m-1$
2. Sắp xếp m cặp thứ tự ($j,\alpha^{mj} \bmod p$) có lưu ý tới các tạo độ thứ hai của các cặp này, ta sẽ thu được một danh sách L_1
3. Tính $\beta\alpha^{-i} \bmod p$, $0 \le i \le m-1$
4. Sắp xếp m cặp thứ tự (i, $\beta\alpha^{-i} \bmod p$) có lưu ý tới các toạ độ thứ hai của các cặp được sắp này, ta sẽ thu được một danh sách L_2
5. Tìm một cặp $(j,y) \in L_1$ và một cặp $(i,y) \in L_2$ (tức là một cặp có tạo độ thứ hai như nhau).
6. Xác định $\log_\alpha\beta = mj + i \bmod (p-1)$
7.

- Nếu cần, các bước 1 và 2 có thể tính toán trước (tuy nhiên, điều này không ảnh hưởng tới thời gian chạy tiệm cận)
- Tiếp theo cần để ý là nếu $(j,y) \in L_1$ và $(i,y) \in L_2$ thì

$$\alpha^{mj} = y = \beta\alpha^{-i}$$

Bởi vậy

$$\alpha^{mj+i} = \beta$$

như mong muốn. Ngược lại, đối với β bất kì ta có thể viết

$$\log_\alpha\beta = mj+i$$

trong đó $0 \le j,i \le m-1$. Vì thế phép tìm kiếm ở bước 5 chắc chắn thành công.

Có thể áp dụng thuật toán này chạy với thời gian O(m) và với bộ nhớ cỡ O(m) (bỏ qua các thừa số logarithm). Chú ý là bước 5 có thể thực hiện một cách (đồng thời) qua từng danh sách L_1 và L_2.

Sau đây là một ví dụ nhỏ để minh hoạ.

Ví dụ 5.2.

Giả sử p = 809 và ta phải tìm $\log_3 525$. Ta có $\alpha = 3$, $\beta = 525$ và $m = \lceil\sqrt{808}\rceil = 29$. Khi đó:

$$\alpha^{29} \bmod 809 = 99$$

Trước tiên tính các cặp được sắp $(j, 99^j \bmod 809)$ với $0 \le j \le 28$. Ta nhận được danh sách sau:

(0,1)	(1,99)	(2,93)	(3,308)	(4,559)
(5,329)	(6,211)	(7,664)	(8,207)	(9,268)
(10,644)	(11,654)	(12,26)	(13,147)	(14,800)
(15,727)	(16,781)	(17,464)	(18,314)	(19,275)
(20,582)	(21,496)	(22,564)	(23,15)	(24,676)
(25,586)	(26,575)	(27,295)	(28,81)	

Danh sách này sẽ được sắp xếp để tạo L_1.

Danh sách thứ hai chứa các cặp được sắp $(i, 525\times(3^i)^{-1} \bmod 809)$, với $0 \le i \le 28$. Danh sách này gồm:

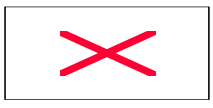
(0,525)	(1,175)	(2,328)	(3,379)	(4,396)
(5,132)	(6,44)	(7,554)	(8,724)	(9,511)
(10,440)	(11,686)	(12,768)	(13,256)	(14,,355)
(15,388)	(16,399)	(17,133)	(18,314)	(19,644)
(20,754)	(21,496)	(22,564)	(23,15)	(24,676)
(25,356)	(26,658)	(27,489)	(28,163)	

Sau khi sắp xếp danh sách này, ta có L_2 .
Bây giờ nếu xử lý đồng thời qua cả hai danh sách, ta sẽ tìm được (10,644) trong L_1 và (19,644) trong L_2. Bây giờ ta có thể tính

$$\log_3 525 = 29\times 10+19$$
$$= 309$$

Có thể kiểm tra thấy rằng quả thực $3^{309} \equiv 525 \pmod{809}$.

Thuật toán Pohlig - Hellman.
Thuật toán tiếp theo mà ta nghiên cứu là thuật toán Pohlig - Hellman. Giả sử



p_i là số nguyên tố đặc biệt. Giá trị $a = \log_\alpha \beta$ được xác định một cách duy nhất theo modulo p-1. Trước hết nhận xét rằng, nếu có thể tính a mod $p_i^{c_i}$ với

mỗi i, $1 \le i \le k$, thì có thể tính a mod (p-1) theo định lý phần dư China. Để thực hiện diều đó ta giả sử rằng q là số nguyên tố.

$$p-1 \equiv 0 \pmod{q^c}$$

$$p-1 \not\equiv 0 \pmod{q^{c+1}}$$

Ta sẽ chỉ ra cách tính giá trị

$$x = a \bmod q^c$$

$0 \le x \le q^c-1$. Ta có thể biểu diễn x theo cơ số q như sau:

trong đó $0 \le a_i \le q-1$ với $0 \le i \le c-1$. Cũng có thể biểu diễn như sau:

$$a = x + q^c s$$

với s là một số nguyên nào đó.

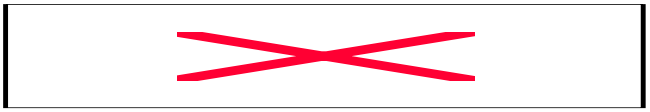
Bước đầu tiên của thuật toán tính a_0. Kết quả chính ở đây là:

$$\beta^{(p-1)/q} \equiv \alpha^{(p-1)a0/q} \pmod{p}$$

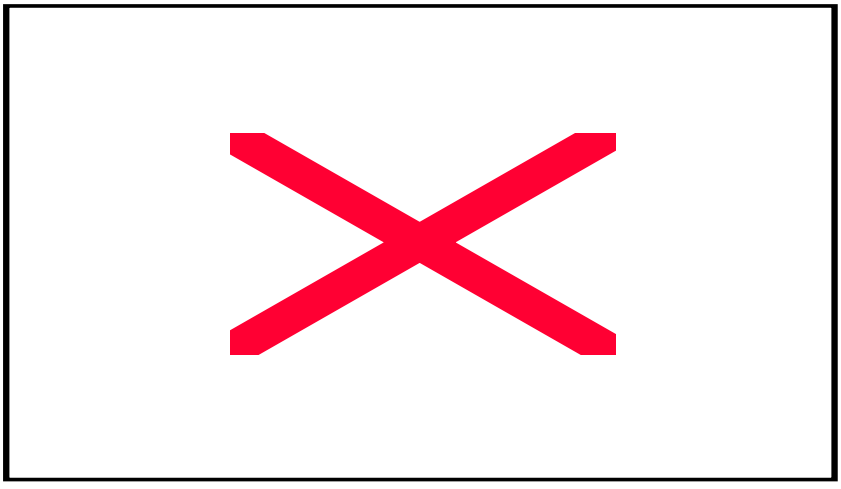
Để thấy rõ điều đó cần chú ý rằng:

Điều này đủ để cho thấy:

Kết quả này đúng khi và chỉ khi:



Tuy nhiên



Đó chính là điều cần chứng minh.

Do đó ta sẽ bắt đầu bằng việc tính $\beta^{(p-1)/q} \bmod p$. Nếu

$$\beta^{(p-1)/q} \equiv 1 \pmod p$$

thì $a_0=0$. Ngược lại chúng ta sẽ tính liên tiếp các giá trị:

$$\gamma = \alpha^{(p-1)/q} \bmod p, \gamma^2 \bmod p, \ldots,$$

cho tới $\gamma^i \equiv \beta^{(p-1)/q} \pmod p$.

với một giá trị i nào đó. Khi điều này xảy ra ta có $a_0 = i$.

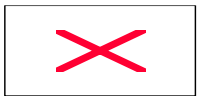
Bây giờ nếu c = 1 thì ta đã thực hiện xong. Ngược lại, nếu c > 1 thì phải tiếp tục xác định a_1. Để làm điều đó ta phải xác định

$$\beta_1 = \beta\, \alpha^{-a_0}$$

và kí hiệu

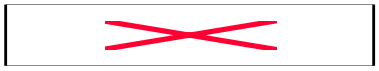
$$x_1 = \log_\alpha \beta_1 \bmod q^c$$

Dễ dàng thấy rằng



Vì thế dẫn đến

Như vậy ta sẽ tính $\beta_1^{(p-1)/q^2} \bmod p$ và rồi tìm i sao cho



Khi đó $a_1 = i$.

Nếu c =2 thì công việc kết thúc; nếu không, phải lặp lại công việc này c-2 lần nữa để tìm $a_2, \ldots, a_{c-1}$.

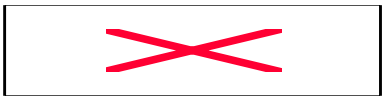
Hình 5.4 là mô tả giải mã của thuật toán Pohlig - Hellman. Trong thuật toán này, α là phần tử nguyên thuỷ theo modulo p, q là số nguyên tố .

$$p-1 \equiv 0 \pmod{q^c}$$

và

$$p-1 \not\equiv 0 \pmod{q^{c+1}}$$

Thuật toán tính các giá trị $a_0, \ldots, a_{c-1}$ trong đó
$\log_\alpha \beta \bmod q^c$



Hình 5.4. Thuật toán Pohlig - Hellman để tính $\log_\alpha \beta \bmod q^c$.

1. Tính $\gamma = \alpha^{(p-1)/q} \bmod p$ với $0 \le i \le q-1$
2. Đặt $j = 0$ và $\beta_j = \beta$
3. **While** $j \le c-1$ **do**
4. Tính $\delta = \beta_j^{(p-1)/q\, j+1} \bmod p$
5. Tìm i sao cho $\delta = \gamma_i$
6. $a_j = i$
7. $\beta_{j+1} = \beta_j\, \alpha^{-a_j q j} \bmod p$
8. $j = j + 1$

Chúng ta minh hoạ thuật toán Pohlig - Hellman (P - H) qua một ví dụ nhỏ.

Ví dụ 5.3

Giả sử p=29; khi đó

$$n = p-1 = 28 = 2^2.7^1$$

Giả sử $\alpha = 2$ và $\beta = 18$. Ta phải xác định $a = \log_2 18$. Trước tiên tính a mod 4 rồi tính a mod 7.

Ta sẽ bắt đầu bằng việc đặt q = 2, c = 2. Trước hết

$$\gamma_0 = 1$$

và

$$\gamma_1 = \alpha^{28/2} \bmod 29$$
$$= 2^{14} \bmod 29$$
$$= 28$$

Tiếp theo

$$\delta = \beta^{28/2} \bmod 29$$
$$= 18^{14} \bmod 29$$
$$= 28$$

Vì $a_0 = 1$. Tiếp theo ta tính:

$$\beta_1 = \beta_0 \alpha^{-1} \bmod 29$$
$$= 9$$

và

$$\beta_1^{28/4} \bmod 29 = 9^7 \bmod 29$$
$$= 28$$

Vì

$$\gamma_1 \equiv 28 \bmod 29$$

Ta có $a_1 = 1$. Bởi vậy $a \equiv 3 \pmod 4$.

Tiếp theo đặt $q = 7$ và $c = 1$, ta có

$$\beta^{28/7} \bmod 29 = 18^4 \bmod 29$$
$$= 25$$

và
$$\gamma_1 = \alpha^{28/7} \bmod 29$$
$$= 2^4 \bmod 29$$
$$= 16.$$

Sau đó tính:
$$\gamma_2 = 24$$
$$\gamma_3 = 7$$
$$\gamma_4 = 25$$

Bởi vậy $a_0 = 4$ và $a \equiv 4 \pmod 7$

Cuối cùng giải hệ phương trình

$$a \equiv 3 \pmod 4$$
$$a \equiv 4 \pmod 7$$

bằng định lý phần dư China, ta nhận được $a \equiv 11 \pmod{28}$. Điều này có nghĩa là đã tính được $\log_2 18$ trong Z_{29} là 11.

Phương pháp tính toán chỉ số.

Phương pháp tính chỉ số khá giống với nhiều thuật toán phân tích thừa số tốt nhất. Trong phần này sẽ xét tóm tắt về phương pháp. Phương pháp này chỉ dùng một *cơ sở nhân tử* là tập $\mathcal{B}$ chứa các số nguyên tố nhỏ. Giả sử $\mathcal{B} = \{p_1, p_2, \ldots, p_{\mathcal{B}}\}$. Bước đầu tiên (bước tiền xử lý) là tìm các logarithm của B số nguyên tố trong cơ sở nhân tử. Bước thứ hai là tính các logarithm rời rạc của phần tử β bằng cách dùng các hiểu biết về các log của các phần tử trong cơ sở.

Trong quá trình tiền xử lý, ta sẽ xây dựng C = B +10 đồng dư thức theo modulo p như sau:

$$\alpha^{x_j} \equiv p_1^{a_{1j}} p2^{a_{2j}} \ldots p_B^{a_{Bj}} \pmod p$$

$1 \le j \le C$. Cần để ý rằng, các đồng dư này có thể viết tương đương như sau:

$$x_j \equiv a_{1j}\log_\alpha p_1 + \ldots + a_{Bj}\log_\alpha p_B \pmod{p-1}$$

$1 \le j \le C$. C đồng dư thức được cho theo B giá trị $\log_\alpha p_i$ $(1 \le i \le B)$ chưa biết. Ta hy vọng rằng, có một nghiệm duy nhất theo modulo p-1. Nếu đúng như vậy thì có thể tính các logarithm của các phần tử theo cơ sở nhân tử.

Làm thế nào để tạo các đồng dư thức có dạng mong muốn?. Một phương pháp sơ đẳng là chọn một số ngẫu nhiên x, tính α^x mod p và xác định xem liệu α^x mod p có tất cả các thừa số của nó trong $\mathcal{B}$ hay không. (Ví dụ bằng cách chia thử).

Bây giờ giả sử rằng đã thực hiện xong bước tiên tính toán, ta sẽ tính giá trị mong muốn $\log_\alpha\beta$ bằng thuật toán xác suất kiểu Las Vegas. Chọn một số ngẫu nhiên s ($1 \le s \le p-2$) và tính :

$$\gamma = \beta\ \alpha^s \bmod p$$

Bây giờ thử phân tích γ theo cơ sở $\mathcal{B}$. Nếu làm được điều này thì ta tính được đồng dư thức dạng:

$$\beta\alpha^s = p_1^{c_1}p_2^{c_2}. . . p_B^{c_B} \pmod p$$

Điều đó tương đương với

$$\log_\alpha\beta + s \equiv c_1\log_\alpha p_1 + . . . + c_B\log_\alpha p_B \ (\bmod\ p-1)$$

Vì mọi giá trị đều đả biết trừ giá trị $\log_\alpha\beta$ nên có thể dễ dàng tìm được $\log_\alpha\beta$.

Sau đây là một ví dụ minh hoạ 2 bước của thuật toán.

Ví dụ 5.4.

Giả sử p =10007 và $\alpha = 5$ là một phần tử nguyên thuỷ được dùnglàm cơ sở của các logarithm theo modulo p. Giả sử lấy B = {2, 3, 5, 7} làm cơ sở. Hiển nhiên là $\log_5 5 = 1$ nên chỉ có 3 giá trị log của các phần tử trong cơ sở cần phải xác định. Để làm ví dụ, chọn một vài số mũ "may mắn" sau: 4063, 5136 và 985.

Với x = 4063, ta tính

$$5^{4063} \bmod 10007 = 2\times3\times7$$

ứng với đồng dư thức

$$\log_5 2 + \log_5 3 + \log_5 7 \equiv 4063 \ (\bmod\ 10006).$$

Tương tự, vì

$$5^{5136} \bmod 10007 = 54 = 2\times3^3$$

và

$$5^{9865} \bmod 10007 = 189 = 3^3\times7$$

ta tìm được hai đồng dư thức nữa:

$$\log_5 2 + 3\log_5 3 \equiv 5136 \ (\bmod\ 10006)$$
$$3\log_5 3 + \log_5 7 \equiv 9865 \ (\bmod\ 10006)$$

Bây giờ ta có 3 đồng dư thức theo 3 giá trị log chưa biết. Giải các phương trình đồng dư này, ta có $\log_5 2 = 6578$, $\log_5 3 = 6190$, $\log_5 7 = 1301$.

Bây giờ giả sử ta cần tìm $\log_5 9451$, ta chọn số mũ "ngẫu nhiên" s=7736 và tính:

$$9451 \times 5^{7736} \bmod 10007 = 8400$$

Vì $8400 = 2^4 3^1 5^2 7^1$ các thừa số trong $\mathcal{B}$ nên ta nhận được:

$$\begin{aligned}\log_5 9451 &= 4\log_5 2 + \log_5 3 + \log_5 5 + \log_5 7 - s \bmod 10006 \\ &= 4\times 6578 + 6190 + 2\times 1 + 1310 - 7736 \bmod 10006 \\ &= 6057.\end{aligned}$$

Kiểm tra lại ta thấy rằng $5^{6057} \equiv 9451 \pmod{10007}$.

Đã có nhiều nghiên cứu phân tích mò mẫm nhiều kiểu thuật toán khác nhau. Với giả thiết hợp lý, Thời gian chạy tiệm cận của giai đoạn tiền tính toán này cỡ

và thời gian để tính một giá trị logarithm rời rạc riêng là khoảng

Hình 5.5. Bít thứ i của logarithm rời rạc.

Bản chất của bài toán: I = (p, α, β, i) trong đó p là số nguyên tố , $\alpha \in Z_p^*$ là phần tử nguyên thuỷ, $\beta \in Z_p^*$ và i là một số nguyên sao cho $1 \le i \le \lfloor \log_2(p-1) \rfloor$.

*Mục tiêu:*Tính $L_i(\beta)$ là bít thấp nhất thứ i của $\log_\alpha \beta$. (với α và p cho trước)

5.1.2. Độ bảo mật tưng bít của các logarithm rời rạc.

Bây giờ ta xem xét vấn đề về thông tin bộ phận của các logarithm rời rạc và thử xem việc tính các bít riêng của các logarithm rời rạc là khó hay dễ. Cụ thể , xét bài toán trình bày trên hình 5.5. Bài toán này được gọi là bài toán về bít thứ i.

Trước tiên, ta sẽ chỉ ra rằng, bít thấp nhất của các logarithm rời rạc rất dễ tính toán. Nói cách khác, nếu i = 1 thì bài toán về bít thứ i có thể giải được một cách hiệu quả. Điều này rút ra từ tiêu chuẩn Euler liên quan đến thặng dư bình phương theo modulo p, với p là số nguyên tố .

Xét ánh xạ f: $Z_p^* \rightarrow Z_p^*$ được định nghĩa như sau:

$$f(x) = x^2 \bmod p$$

Nếu kí hiệu QR(p) là tập các thặng dư bình phương theo modulo p thì

$$QR(p) = \{ x^2 \bmod p : x \in Z_p^* \}$$

Trước tiên ta thấy rằng, f(x) = f(p-x). Tiếp theo xét thấy:

$$w^2 \equiv x^2 \bmod p$$

khi và chỉ khi $p \mid (w-x)(w+x)$

điều này sẽ xảy ra khi và chỉ khi $w \equiv \pm x \bmod p$. Từ đây rút ra:

$$| f^{-1}(y) | = 2$$

với mọi $y \in QR(p)$ và bởi vậy:

$$| QR(p) = (p-1)/2$$

Điều đó có nghĩa là có đúng một nữa các thặng dư trong Z_p^* là các thặng dư bình phương và một nữa không phải.

Bây giờ giả sử rằng, α là một phần tử nguyên thuỷ của Z_p^* . Khi đó $\alpha^a \in QR(p)$ nếu a chẵn. Vì (p-1)/2 phần tử $\alpha^0 \bmod p$, $\alpha^2 \bmod p$,. . .,$\alpha^{p-3} \bmod p$ đều là các phần tử khác nhau nên:

$$QR(p) = \{\alpha^{2i} \bmod p: 0 \le i \le (p-3)/2\}$$

Bởi vậy, β là thặng dư bình phương khi và chỉ khi $\log_\alpha \beta$ là chẵn, tức khi và chỉ khi $L_1(\beta) = 0$. Tuy nhiên theo tiêu chuẩn Euler β là thặng dư bình phương khi và chỉ khi

$$\beta^{(p-1)/2} \equiv 1 \pmod p$$

Như vậy, ta đã có công thức hữu hiệu sau để tính $L_1(\beta)$:

$$L_1(\beta)= \begin{cases} 0 & \text{nếu } \beta^{(p-1)/2} \equiv 1 (\bmod p) \\ 1 & \text{trong các trường hợp còn lại} \end{cases}$$

Bây giờ xét việc tính $L_i(\beta)$ với $i > 1$. Giả sử

$$p-1 = 2^s t$$

trong đó t là số lẻ. Khi đó có thể chỉ ra rằng, dễ dàng tính được $L_i(\beta)$ nếu $1 \le s$. Mặt khác, việc tính $L_{s+1}(\beta)$ chắc chắn là khó nếu dùng thuật toán giả định bất kì cho việc tính $L_{s+1}(\beta)$ để tính các logarithm rời rạc trong Z_p.

Ta sẽ chứng minh kết quả này trong trường hợp s = 1. Chính xác hơn, nếu $p \equiv 3 \pmod 4$là số nguyên tố thì ta sẽ chỉ ra cách sử dụng một thuật toán giả định bất kì tính $L_2(\beta)$ để giải bài toán logarithm rời rạc trong Z_p.

Nếu β là một thặng dư bình phương trong Z_p và $p \equiv 3 \pmod 4$ thì $\pm\beta^{(p+1)/2} \bmod p$ là hai giá trị căn bậc hai của modulo p. Một chú ý cũng quan trọng là với bất kì $\beta \neq 0$:

$$L_1(\beta) \neq L_1(p-\beta).$$

nếu $p \equiv 3 \pmod 4$. Ta sẽ thấy điều đó như sau. Giả sử

$$\alpha^a \equiv \beta \pmod p$$

thì

$$\alpha^{a+(p-1)/2} \equiv -\beta \pmod p$$

Vì $p \equiv 3 \pmod 4$ nên số nguyên $(p-1)/2$ là một số lẻ. Từ đây rút ra kết quả.

Bây giờ giả sử $\beta = \alpha^a$ với số mũ chẵn a (chưa biết) nào đó. Khi đó hoặc:

$$\beta^{(p+1)/4} \equiv \alpha^{a/2} \pmod p$$

hoặc

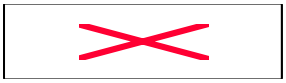
$$-\beta^{(p+1)/4} \equiv \alpha^{a/2} \pmod p$$

Ta có thể xác định giá trị nào trong hai giá trị có thể này là đúng nếu biết giá trị $L_2(\beta)$, vì

$$L_2(\beta) = L_1(\alpha^{a/2})$$

Điều này được khai thác trong thuật toán được mô tả trong hình 5.6.

Ở cuối thuật toán, các giá trị x_i là các bít biểu diễn nhị phân của $\log_\alpha\beta$, nghĩa là:



Dưới đây là một ví dụ nhỏ để minh hoạ.

Ví dụ 5.5.

Giả sử p =19, $\alpha = 2$ và $\beta = 6$. Vì trong ví dụ này, các giá trị quá nhỏ nên có thể lập bảng các giá trị của $L_1(\gamma)$ và $L_2(\gamma)$ với mọi mọi giá trị $\gamma \in Z_{19}^*$.(Nói chung L_1 có thể tính được một cách hiệu quả bằng tiêu chuẩn Euler, còn L_2 được tính theo thuật toán giả định). Các giá trị này được cho trên bảng 5.1. Thuật toán được tiến hành như trên hình 5.7.

Bởi vậy, $\log_2 6 = 1110_2 = 14$, ta có thể dễ dàng kiểm tra được giá trị này.

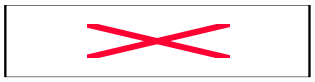
Hình 5.6. Tính các logarithm rời rạc trong Z_p với $p \equiv 3 \pmod 4$ khi biết trước thuật toán giả định $L_2(\beta)$.

1. $x_0 = L_1(\beta)$
2. $\beta = \beta/\alpha^{x_0} \bmod p$
3. i =1
4. **While** $\beta \neq 1$ **do**
5. $x_i = L_2(\beta)$
6. $\gamma = \beta^{(p+1)/4} \pmod p$
7. **if** $L_1(\gamma) = x_i$ **then**
8. $\beta = \gamma$
9. **else**
10. $\beta = p - \gamma$
11. $\beta = \beta/\alpha^{x_i} \bmod p$
12. i = i+1

Bảng 5.1. Các giá trị của L_1 và L_2 với p =19, $\alpha = 2$

γ	$L_1(\gamma)$	$L_2(\gamma)$	γ	$L_1(\gamma)$	$L_2(\gamma)$	γ	$L_1(\gamma)$	$L_2(\gamma)$
1	0	0	7	0	1	13	1	0
2	1	0	8	1	1	14	1	1
3	1	0	9	0	0	15	1	1
4	0	1	10	1	0	16	0	0
5	0	0	11	0	0	17	0	1
6	0	1	12	0	0	18	1	0

Có thể đưa ra một chứng minh hình thức cho tính đúng đắn của thuật toán bằng phương pháp quy nạp. Kí hiệu



Với $i \geq 0$, ta định nghĩa:

$$Y_i = \lfloor x/2^{i+1} \rfloor$$

Hình 5.7 Tính $\log_2 6$ trong Z_{19}

1. $x_0 = 0$
2. $\beta = 6$
3. $i = 1$
5. $x_1 = L_2(6) = 1$
6. $\gamma = 5$
7. $L_1(5) = 0 \neq x_1$
10. $\beta = 14$
11. $i = 2$
12. $i = 2$
5. $x_2 = L_2(7) = 1$
6. $\gamma = 11$
7. $L_1(11) = 0 \neq x_2$
10. $\beta = 8$
11. $\beta = 4$
12. $i = 3$
5. $x_3 = L_2(4) = 1$
6. $\gamma = 17$
7. $L_1(17) = 0 \neq x_3$
10. $\beta = 2$
11. $\beta = 1$
12. $i = 4$
4. DONE

Cũng vậy ta xác định β_0 là giá trị của β ở bước 2 trong thuật toán; và với $i \geq 1$, ta xác định β_i là giá trị của β ở bước 11 trong bước lặp thứ i của vòng **While**. Có thể chứng minh bằng phương pháp quy nạp rằng:

$$\beta i \equiv \alpha^{2Y_i} \pmod{p}$$

với mọi $i \geq 0$. Bây giờ để ý rằng: $2Y_i = Y_{i-1} - x_i$

điều này kéo theo

$$x_{i+1} = L_2(\beta_i) \text{ , } i \geq 0$$

Vì rằng $x_{i+1} = L_2(\beta)$ nên thuật toán là đúng. Các chi tiết dành cho độc giả xem xét.

5.2. TRƯỜNG HỮU HẠN VÀ CÁC HỆ THỐNG ĐƯƠNG CONG ELLIPTIC.

Chúng ta đã dành thời gian đáng kể để xét bài toán logarithm rời rạc (DL) vào việc phân tích số. Ta sẽ còn trở lại hai bài toán này trong các loại hệ mật và các giao thức mã khác nhau. Bài toán DL đã được nghiên cứu trong trương hữu hạn Z_p, tuy nhiên việc xét bài toán này theo các thiết lập khác nhau cũng rất có ích và là chủ đề của phần này.

Hệ mật Elgamal có thể được áp dụng trong một nhóm bất kì mà bài toán DL là khó giải. Ta đã dùng nhóm nhân Z_p^* tuy nhiên các nhóm khác cũng là những ứng cử viên thích hợp. Trước hết ta phát biểu bài toán DL trong một nhóm hữu hạn nói chung G (hữu hạn) và ở đó kí hiệu phép lấy nhóm là dấu "$\circ$". Dạng bài toán tổng quát hoá như vậy trình bài trên hình 5.8.

Dễ dàng xác định một hệ mật Elgamal trong nhóm con H theo cách tương tự đã mô tả trong Z_p^* và được trình bày trên hình 5.9. Chú ý rằng phép mã hoá yêu cầu dùng số nguyên k ngẫu nhiên sao cho $0 \leq k \leq |H| - 1$. Tuy nhiên, nếu Alice không biết cấp của nhóm con H thì cô ta có thể tạo một số nguyên k thoả mãn $0 \leq k \leq |G| - 1$, khi đó sẽ không có bất kì sự thay đổi nào trong quá trình mã và giải mã. Cũng cần chú ý là nhóm G không phải là nhóm Aben (Tuy H vẫn là nhóm Aben vì nó là nhóm cyclic).

Hình 5.8. Bài toán logarithm rời rạc trong (G,0)

Đặc trưng của bài toán: I = (G, α, β), trong đó G là một nhóm hữu hạn với phép lấy nhóm $\circ$, $\alpha \in G$ và $\beta \in H$, trong đó

$$H = \{ \alpha^i : i \geq 0 \}$$

là một nhóm con sinh bởi α.

Mục tiêu: Tìm một số nguyên duy nhất a sao cho $0 \leq a \leq |H| - 1$ và $\alpha^a = \beta$, với kí hiệu α^a có nghĩa là $\alpha \circ \ldots \circ \alpha$ (a lần)
Ta sẽ kí hiệu số nguyên a này bằng $\log_\alpha \beta$

Bây giờ ta sẽ trở lại bài toán DL tổng quát hoá . Nhóm con H được sinh bởi phần tử α tuỳ ý $\in G$ dĩ nhiên phải là nhóm con cyclic cấp |H|. Bởi vậy, dạng bất kì của bài toán theo một nghĩa nào đó đều tương đương với bài toán DL trong một nhóm cyclic. Tuy nhiên, độ khó của bài toán DL dường như phụ thuộc vào cách biểu diễn nhóm được dùng.

Xét một ví dụ về cách biểu diễn mà với nó, bài toán logarithm rời rạc rất dễ giải. Xét nhóm cộng cyclic Z_n và giả sử UCLN(α,n) = 1, bởi vậy α là phần tử sinh của Z_n. Vì phép toán trong nhóm là cộng theo modulo n nên phép lấy mũ sẽ là nhân với a theo modulo n. Vì thế trong cách xây dựng này, bài toán logarithm rời rạc sẽ là tìm số nguyên a sao cho.

$$\alpha a \equiv \beta \pmod n$$

Vì UCLN(α,n) = 1 nên α có phần tử nghịch đảo nhân theo modulo n và ta có thể dễ dàng tính α^{-1} mod n bằng thuật toán Euclide. Sau đó có thể giải để tìm a và nhận được

$$\log_\alpha \beta = \beta\, \alpha^{-1} \bmod n$$

Hình 5.9. Hệ mật khoá công khai Elgamal tổng quát

Giả sử G là một nhóm hữu hạn có phép lấy nhóm $\circ$. Giả sử $\alpha \in G$ là một phần tử sao cho bài toán DL trong H là khó; ở đây $H = \{\alpha_i, i \geq 0\}$ là một nhóm con sinh bởi α. Đặt $\mathcal{P} = G$, $\mathcal{C} = G \times G$ và định nghĩa:

$$\mathcal{K} = \{(G, \alpha, a, \beta) : \beta = \alpha^a\}$$

Các giá trị α, β công khai, còn a được giữ kín.

Với $K = (G, \alpha, a, \beta)$ và với một số ngẫu nhiên bí mật $k \in Z_{|H|}$ ta xác định:

$$e_K(x,k) = (y_1, y_2)$$

trong đó $y_1 = \alpha^k$

và $y_2 = (x \circ \beta^k)$

Với bản mã $y = (y_1, y_2)$ ta xác định:

$d_K(y) = y_2 \circ (y_1^a)^{-1}$

Ở phần trên ta đã nghiên cứu bài toán DL trong nhóm nhân Z_p^* với p là là số nguyên tố . Nhóm này là nhóm cyclic cấp p-1 và bởi vậy nó đẳng cấu với nhóm cộng Z_{p-1}. Theo thảo luận ở trên, ta đã biết cách tinh các logarithm rời rạc một cách hiệu quả trong nhóm cộng này. Điều đó gợi ý khả năng giải bài toán DL trong Z_p^* bằng cách quy nó về bài toán giải được dễ dàng trong Z_{p-1}.

Ta hãy xem xét điều này được thực hiện như thế nào?. Khi nói rằng, $(Z_p^*, \times)$ là đẳng cấu với $(Z_{p-1}, +)$ có nghĩa là có một song ánh :

$$\phi : Z_p^* \rightarrow Z_{p-1}$$

sao cho $\phi(xy \bmod p) = (\phi(x) + \phi(y)) \bmod (p-1)$

Điều đó kéo theo:

$$\phi(\alpha^a \bmod p) = a\,\phi(\alpha) \bmod (p-1)$$

Bởi vậy

$$\beta \equiv \alpha^a \bmod p \Leftrightarrow a\,\phi(\alpha) \equiv \phi(\beta) \pmod{p-1}$$

Do đó nếu tìm a theo mô tả ở trên, ta có:

$$\log_\alpha \beta = \phi(\beta)\,(\phi(\alpha))^{-1} \bmod (p-1)$$

Bây giờ, nếu có một phương pháp hữu hiệu để tính phép đẳng cấu ϕ thì ta sẽ có một thuật toán hữu hiệu để tính các logarithm rời rạc trong Z_p^*. Khó khăn ở đây là không có một phương pháp chung đã biết nào để tính hiệu quả phép đẳng cấu ϕ với số nguyên tố tuỳ ý. Ngay cả khi đã biết hai nhóm là

đẳng cấu thì vẫn không thể biết một thuật toán hiệu quả để mo tả tương minh phép đẳng cấu.

Phương pháp này có thể áp dụng cho bài toán DL trong một nhóm G tuỳ ý. Nếu có một phương pháp hiệu quả tính phép đẳng cấu giữa H và $Z_{|H|}$ thì bài toán DL trong G mô tả ở trên có thể giải được một cách hữu hiệu. Ngược lại, dễ dàng thấy rằng, một phương pháp tính các logarithm rời rạc có hiệu quả sẽ tạo ra phương pháp hiệu quả tính phép đẳng cấu giữa hai nhóm.

Thảo luận ở trên chỉ ra rằng, bài toán DL có thể dễ hoặc khó (xétbề ngoài) tuỳ thuộc vào biểu diễn của nhóm cyclic được dùng. Như vậy, sẽ tốt hơn nếu xem xét các nhóm khác với hy vọng tìm được các thiết lập khác nhau để bài toán DL có vẻ khó. Có hai lớp nhóm như vậy.

1. Nhóm nhân của trường Galois GF(p^n)
2. Nhóm của một đường cong elliptic xác định trên một trương hữu hạn.

Ta hãy xem xét hai lớp nhóm này ở phần sau.

5.1.2. Trường Galois

Ta đã biết rằng, nếu p là số nguyên tố thì Z_p sẽ là một trường. Tuy nhiên có nhiều trường hữu hạn khác không có dạng trên. Thực tế có các trường hữu hạn q phần tử nếu $q = p^n$, trong đó p là số nguyên tố , $n \geq 1$là số nguyên. Bây giờ ta sẽ mô tả ngắn gọn cách xây dựng một trường như vậy. Trước tiên ta sẽ đưa ra một vài định nghĩa.

Định nghĩa 5.1

Giả sử p là số nguyên tố. Gọi $Z_p[x]$ là tập tất cả các đa thức biến x. Bằng cách xây dựng phép cộng và nhân đa thức theo quy tắc thông thường (và rút gọn hệ số theo modulo p) ta sẽ tạo nên một vành.

Với $f(x), g(x) \in Z_p[x]$, ta nói rằng, f(x) chia hết cho g(x) (kí hiệu f(x) | g(x)) nếu tồn tại $q(x) \in Z_p[x]$ sao cho:

$$g(x) = q(x)f(x)$$

Với $f(x) \in Z_p[x]$, ta xác định bậc của f (kí hiệu là deg(f)) là số mũ cao nhất có trong các số hạng của f.

Giả sử $f(x), g(x), h(x) \in Z_p[x]$ và $deg(f) = n \geq 1$, ta định nghĩa:

$$g(x) \equiv h(x) \pmod{f(x)}$$

nếu $f(x) \mid (g(x) - h(x))$.

Chú ý sự tương tự giữa định nghĩa về đồng dư của các đa thức với định nghĩa về đồng dư của các số nguyên.

Bây giờ ta sẽ định nghĩa vành các đa thức theo modulo f(x). (ta kí hiệu vành này là $Z_p[x]/f(x)$). Việc xây dựng $Z_p[x]/f(x)$ từ $Z_p[x]$ dựa trên khái niệm về các đồng dư thức theo modulo f(x) và nó tương tự như việc xây dựng Z_m từ Z.

Giả sử deg(f) = n. Nếu chia g(x) cho f(x), ta thu được thương q(x) và phần dư r(x), trong đó:

$$g(x) = q(x)f(x) + r(x)$$

và $\deg(r) < n$.

Điều này có thể thực hiện theo cách chia các đa thức thông thường. Bởi vậy, một đa thứ bất kì trong $Z_p[x]$ đều đồng dư theo modulo f(x) với một đa thức duy nhất có bậc $\leq n-1$.

Bây giờ ta sẽ xác định các phần tử của $Z_p[x]/f(x)$ là p^n các đa thức trong $Z_p[x]$ có bậc nhiều nhất là n-1. Phép cộng và nhân trong $Z_p[x]/(f(x))$ được xác định như trong $Z_p[x]$, sau đó thực hiện rút gọn theo modulo f(x). Với phép toán này, $Z_p[x]/(f(x))$ sẽ tạo thành một vành.

Cần nhớ lại rằng, Z_m là một trường khi và chỉ khi m là số nguyên tố và các phần tử nghịch đảo nhân có thể tìm được qua thuật toán Euclide. Tình hình cũng tương tự xảy ra đối với $Z_p[x]/(f(x))$. Sự tương tự của các số nguyên tố với các đa thức bất khả quy được xác định như sau:

Định nghĩa 5.2

Đa thức $f(x) \in Z_p[x]$ được gọi là bất khả quy nếu không tồn tại các đa thức $f_1(x), f_2(x) \in Z_p[x]$ sao cho

$$f(x) = f_1(x)f_2(x).$$

trong đó $\deg(f_1) > 0$ và $\deg(f_2) > 0$.

Một thực tế rất quan trọng là $Z_p[x]/(f(x))$ là một trường khi và chỉ khi f(x) bất khả quy. Hơn nữa, các phần tử nghịch đảo nhân trong $Z_p[x]/(f(x))$ có thể tính được bằng cách dùng thuật toán Euclide mở rộng có biến đổi đôi chút.

Sau đây là một ví dụ minh hoạ cho vấn đề nêu ra.

Ví dụ 5.6

Xây dựng một trường 8 phần tử. Điều này có thể thực hiện bằng cách tìm một đa thức bất khả quy bậc 3 trong $Z_2[x]$. Ta chỉ cần xem xét các đa thức có thành phần hằng số bằng 1 vì một đa thức bất kì có thành phần hằng số bằng 0 sẽ chia hết cho x và bởi vậy nó là một đa thức bất khả quy . Có tất cả 4 đa thức như vậy.

$$f_1(x) = x^3 + 1$$
$$f_2(x) = x^3 + x + 1$$
$$f_3(x) = x^3 + x^2 + 1$$
$$f_4(x) = x^3 + x^2 + x + 1$$

Xét thấy $f_1(x)$ là khả quy vì:

$$x^3 +1 = (x+1)(x^2+x+1)$$

(cần để ý là tất cả các hệ số được rút gọn theo modulo 2). Tương tự, $f_4(x)$ cũng khả quy vì:

$$x^3+x^2+x+1 = (x+1)(x^2+1)$$

Tuy nhiên cả hai đa thức $f_2(x)$ va $f_3(x)$ lại đều là đa thức bất khả quy và có thể dùng hai đa thức này để xây dựng trường 8 phần tử .

Giả sử dùng $f_2(x)$ để xây dựng trường $Z_2[x]/(x^3+x+1)$. 8 phần tử của trường là 8 đa thức : 0, 1, x, x+1, x^2, x^2+1, x^2+x, x^2+x+1

Để tính tích của hai phần tử của trường, nhân hai đa thức với nhau và rút gọn theo modulo x^3+x+1 (tức chia cho (x^3+x+1) và tìm đa thức dư). Vì ta chia một đa thức bậc 3 nên đa thức dư có bậc nhiều nhất là 2 và vì thế nó là một phần tử của trường.

Ví dụ, ta hãy tính (x^2+1)(x^2+x+1) trong $Z_2[x]/(x^3+x+1)$. Trước hết tính tích trong $Z_2[x]$ là x^4+x^3+x+1. Khi chia cho x^3+x+1, ta nhận được biểu thức sau:

$$x^4+x^3+x+1 = (x+1)(x^3+x+1) +x^2+x$$

Bởi vậy, trong trường $Z_2[x]/(x^3+x+1)$ ta có :

$$(x^2+1)(x^2+x+1) = x^2+x$$

Dưới đây sẽ đưa ra bảng dầy đủ cho cá phần tử khác 0 của trường. Để đơn giản, ta viết đa thức : $a_2x^2+a_1x+a_0$ theo bộ ba được sắp $a_2a_1a_0$.

	001	010	011	100	101	110	111
001	001	010	011	100	101	110	111
010	010	100	110	011	001	111	101
011	011	110	101	111	100	001	010
100	100	011	111	110	010	101	001
101	101	001	100	010	111	011	110
110	110	111	001	101	011	010	100
111	111	101	010	001	110	100	011

Việc tính các phần tử nghịch đảo được tực hiện theo thuật toán Euclide mở rộng có biến đổi đôi chút.

Cuối cùng, ta thấy rằng nhóm nhân của các đa thức khác 0 trong trường là một nhóm cyclic cấp 7. Vì 7 là số nguyên tố nên suy ra mọi phần tử khác 0 của trường đều là phần tử sinh của nhóm này (tức là phần tử nguyên thuỷ).

Ví dụ, nếu tính các luỹ thừa của x, ta có:

$$x^1 = x$$
$$x^2 = x^2$$
$$x^3 = x+1$$
$$x^4 = x^2+1$$
$$x^5 = x^2+ x+1$$
$$x^6 = x^2+1$$
$$x^7 = 1$$

sẽ bao gồm tất cả các phần tử khác 0 của trường.

Vấn đề còn lại là sự tồn tại và tính duy nhất của các trường dạng này. Có thể chỉ ra rằng, có ít nhất một đa thức bất khả quy bậc bất kì $n \geq 1$ trong $Z_p[x]$. Bởi vậy, sẽ có một trường hữu hạn p_n phần tử đối với mọi nguyên tố p và mọi số nguyên $n \geq 1$. Thông thương có khá nhiều đa thức bất khả quy bậc n trong $Z_p[x]$. Tuy nhiên, những trường hữu hạn được xây dựng từ hai đa thức bất khả quy bất kì bậc n đều có thể chứng tỏ được chúng là đaửng cấu với nhau. Bởi vậy, chỉ có một trương hữu hạn duy nhất cấp p^n tuỳ ý (p - số nguyên tố, $n \geq 1$) là trường $GF(p^n)$. Trong trường hợp n = 1, trương GF(p) cũng chính là Z_p. Cuối cùng, có thể chỉ ra rằng, không tồn tại một trường hữu

hạn r phần tử trừ phi $r = p^n$ với p là số nguyên tố , n là số nguyên nào đó ($n \geq 1$).

Ta đã nhận thấy là nhóm nhân Z_p^* (p - số nguyên tố) là một nhóm cyclic cấp p-1. Thực tế, nhóm nhân của trường hữu hạn bất kì đều là nhóm cyclic: $GF(p^n)\backslash\{0\}$ là một nhóm cyclic cấp p^n-1. Nhóm này sẽ cho các ví dụ về các nhóm cyclic trong đó bài toán DL có thể được nghiên cứu.

Thực tế các trường hữu hạn $GF(2^n)$ đã được nghiên cứu khá kĩ. Cả hai thuật toán logarithm rời rạc Shanks và Pohlig-Hellman đều làm việc trên các trường $GF(2^n)$. Phương pháp tính toán chỉ số có thể sửa đổi để làm việc trên các trương này. Thời gian tiền tính toán của thuật toán tính toán chỉ số khoảng

$$O\left(e^{(1,405+O(1))n^{1/3}(\ln n)^{2/3}}\right)$$

còn thời gian để tìm một giá trị logarithm rời rạc riêng khoảng

$$O\left(e^{(1,098+O(1))n^{1/3}(\ln n)^{2/3}}\right)$$

Tuy nhiên, với các giá trị n lớn (n > 800), bài toán DL trong $GF(2^n)$ được coi là khó cỡ 2^n phải có ít nhất một thừa số nguyên tố "lớn" (để gây khó khăn cho cách tấn công Pohlig - Hellman).

5.2.2. Các đương cong Elliptic

Ta bắt đầu bằng việc định nghĩa khái niệm đường cong elliptic.

Định nghĩa 5.3

Cho p >3 là số nguyên tố. Đường cong elliptic $y^2 = x^3+ax+b$ trên Z_p là một tập các cặp $(x,y) \in Z_p \times Z_p$ thoả mãn đồng dư thức

$$y^2 \equiv x^3+ax+b \pmod p \qquad (5.1)$$

trong đó $a, b \in Z_p$ là các hằng số sao cho $4a3+27b2 \not\equiv 0 \pmod p$ cùng với một điểm đặc biệt O được gọi là điểm vô cực.

[Phương trình (5.1) có thể dùng để xác định một đường cong elliptic trên một trường bất kì $GF(p^n)$ với p - là số nguyên tố lớn hơn 3. Đường cong elliptic trên $GF(2^n)$ hoặc $GF(3^n)$ được xác định bằng một phương trình khác đôi chút)].

Đường cong elliptic E có thể tạo thành một nhóm Aben bằng cách xác định một phép toán thích hợp trên các điểm của nó. Phép toán này là phép cộng và được xác định như sau (ở đây mọi phép toán số học được thực hiện trên Z_p).

Giả sử

$$P = (x_1,y_1) \text{ và } Q = (x_2,y_2)$$

là các điểm trên E. Nếu $x_2=x_1$ và $y_2=-y_1$ thì P+Q = O; ngược lại $P+Q = (x_3,y_3)$ trong đó:

$$x_3 = \lambda^2-x_1-x_2$$
$$y_3 = \lambda(x_1-x_3)-y_1$$

và

$$\lambda = \begin{cases} (y_2-y_1)/(x_2-x_1) & \text{nếu } P \neq Q \\ (3x_1^2+a)/2y_1 & \text{nếu } P = Q \end{cases}$$

Cuối cùng ta xác định

$$P+O = O+P = P$$

đối với mọi $P \in E$. Với định nghĩa phép cộng như vậy, có thể chỉ ra rằng, E là một nhóm Aben với phần tử đơn vị O. (phần lớn các phép kiểm tra đều khá đơn giản song việc chứng minh tính kết hợp lại rất khó).

Cần để ý là các phần tử ngược (nghịch đảo) rất dễ tính toán. Phần tử nghịch đảo của (x,y) là (x,-y) với mọi $(x,y) \in E$ (ta kí hiệu phần tử này là -(x,y) do phép nhóm là phép cộng)

Xét ví dụ sau.

Ví dụ 5.7

Giả sử E là một đường cong elliptic $y^2 = x^3+x+6$ trên Z_{11}. Trước tiên ta xác định các điểm trên E. Để làm điều đó, xét mỗi giá trị có thể $x \in Z_{11}$, tính $x^3+x+6 \bmod 11$ và thử giải phương trình (5.1) đối với y. Với giá trị x cho trước, ta có thể kiểm tra xem liệu $z = x^3+x+6 \bmod 11$ có phải là một thặng dư bình phương hay không bằng cách áp dụng tiêu chuẩn Euler. Ta đã có một công thức tường minh để tính các căn bậc hai của các thặng dư bình phương theo modulo p với các số nguyên tố $p \equiv 3 \pmod 4$. Áp dụng công thức này, ta có các căn bậc hai của một thặng dư bình phương z là:

$$z^{(11+1)/4} \bmod 11 = z^3 \bmod 11$$

Kết quả của các phép tính này được nêu trên bảng 5.2

Như vậy, E có tất cả 13 điểm. Với một nhóm bất kì cấp nguyên tố đều là nhóm cyclic nên dẫn đến E đẳng cấu với Z_{13} và một điểm bất kì (không phải điểm vô cực) đều là phần tử sinh của nhóm E. Giả sử ta lấy phần tử sinh là $(2,7) = \alpha$. Khi đó ta có thể tính các "luỹ thừa" của α (chính là các bội của α vì phép nhóm là phép cộng). Để tính $2\alpha = (2,7) + (2,7)$, trước hết ta tính:

$\lambda = (3\times 2^2+1)(2\times 7)^{-1} \bmod 11$
$= 2\times 3^{-1} \bmod 11$
$= 2\times 4 \bmod 11$
$= 8$

Sau đó ta có: $x_3 = 8^2-2-2 \bmod 11$
$= 5$

và $y_3 = (8(2-5)-7) \bmod 11$
$= 2$

Bởi vậy $2\alpha = (5,2)$

Bảng 5.2 Các điểm trên đường cong elliptic $y^2 = x^3+x+6$ trên Z_{11}

x	x3+x+6 mod 11	Có trong QR(11)?	y
0	6	Không	
1	8	Không	
2	5	Có	4,7
3	3	Có	5,6
4	8	Không	
5	4	Có	2,9
6	8	Không	
7	4	Có	2,9
8	9	Có	3,8
9	7	Không	
10	4	Có	2,9

Bội tiếp theo là $3\alpha = 2\alpha+\alpha = (5,2) + (2,7)$. Ta lại bắt đầu bằng việc tính λ.

$\lambda = (7-2)(2-5)^{-1} \bmod 11$
$= 5\times 8^{-1} \bmod 11$
$= 5\times 7 \bmod 11$
$= 2$

Khi đó ta có $x_3 = 2^2-5-2 \bmod 11$
$= 8$

và $y_3 = 2(5-8) - 2 \bmod 11$
$= 3$

Bởi vậy $3\alpha = (8,3)$

Tiếp tục theo cách tương tự, có thể tính được các bội còn lại như sau:

$\alpha = (2,7)$ $2\alpha = (5,2)$ $3\alpha = (8,3)$
$4\alpha = (10,2)$ $5\alpha = (3,6)$ $6\alpha = (7,9)$
$7\alpha = (7,2)$ $8\alpha = (3,5)$ $9\alpha = (10,9)$
$10\alpha = (8,8)$ $11\alpha = (5,9)$ $12\alpha = (2,4)$

Do đó $\alpha = (2,7)$ thực sự là phần tử nguyên thuỷ.

Một đường cong elliptic xác định trên Z_p (p là số nguyên tố >3) sẽ có khoảng p điểm. Chính xác hơn, theo một định lý nổi tiếng của Hasse, số các điểm trên E (kí hiệu là #E) thảo mãn bất đẳng thức sau:

$$p + 1 - 2\sqrt{p} \le \#E \le p + 1 + 2\sqrt{p}$$

Việc tính toán chính xác giá trị của #E có khó hơn nhưng đã có một thuật toán hữu hiệu do Schoof đưa ra giúp tính toán dễ dàn hơn.(Nghĩa hữu hiệu ở đây được hiểu là thời gian chạy của thuật toán là thời gian đa thức theo log p. Thuật toán Schoof có thời gian chạy khoảng $O((\log p)^8)$ phép tính trên bít và có thể thực hiện đối với các số nguyên tố p có vài trăm chữ số).

Bây giờ giả sử có thể tính được #E. Vấn đề tiếp theo là phải tìm một nhóm con cyclic trong E sao cho bài toán DL trong nó là khó. Bởi vậy ta phải biết một vài điều về cấu trúc của nhóm E. Định lý sau đây cung cấp một thông tin đáng kể về cấu trúc nhóm của E.

Định lý 5.1

Cho E là một đường cong elliptic trên Z_p, p là số nguyên tố > 3. Khi đó, tồn tại các số nguyên n_1 và n_2 sao cho E là đẳng cấu với $Z_{n1} \times Z_{n2}$. Ngoài ra n_2 / n_1 và $n_2 / (p-1)$.

Bởi vậy nếu có thể tính được các số n_1 và n_2 thì ta sẽ biết rằng E có một nhóm con cyclic đẳng cấu với Z_{n1} và có thể dùng E để thiết lập một hệe mật Elgamal.

Chú ý là nếu $n_2 = 1$ thì E là một nhóm cyclic. Cũng vậy, nếu #E là một số nguyên tố hoặc là tích của các số nguyên tố khác nhau thì E là nhóm cyclic có chỉ số nhóm cyclic.

Các thuật toán Shanks và Pohlig - Hellman có thể áp dụng cho bài toán rời rạc trên đường cong Elliptic song tới nay vẫn chưa có một thuật toán

thích hợp cho phương pháp tính chỉ số đối với các đường cong Elliptic.Tuy nhiên, đã có một phương pháp khai thác đẳng cấu một cách tường minh giữa các đường cong Elliptic trong trường hữu hạn. Phương pháp này dẫn đến các thuật toán hữu hiệu đối với một số lớp các đường cong Elliptic. Kỹ thuật này do Menezes, Okamoto và Vanstone đưa ra và có thể áp dụng cho một số trường hợp riêng trong một lớp đặc biệt các đường cong Elliptic (được gọi là các đường cong siêu biến, chúng đã được kiến nghị sử dụng trong các hệ thống mật mã). Tuy nhiên, nếu tránh các đường cong siêu biến thì lại xuất hiện một đường cong Elliptic có một nhóm con cyclic cỡ 2^{160} , đường cong này sẽ cho phép thiết lập an toàn một hệ mật miễn là bậc của nhóm con phải là bội của ít nhất một thừa số nguyên tố lớn (nhằm bảo vệ hệ mật khỏi phương pháp tấn công của Pohlig - Hellman).

Xét một ví dụ về phép mã Elgamal sử dụng đường cong elliptic nêu trên ví dụ 5.7.

Ví dụ 5.8.

Giả sử $\alpha = (2,7)$ và số mũ mật của Bob là a = 7. Bởi vậy:

$$\beta = 7\alpha = (7,2)$$

Phép mã hoá thực hiện như sau

$$e_K(x,k) = (k(2,7),x+k(7,2))$$

trong đó $x \in E$ và $0 \le k \le 12$ còn phép giải mã thực hiện như sau:

$$d_K(y_1,y_2) = y_2-7y_1$$

Giả sử Alice muốn mã bản tin x = (10,9) (là một điểm trên E). Nếu cô chọn giá trị ngẫu nhiên k=3 thì cô tính

$$y1 = 3(2,7)$$
$$= (8,3)$$

và

$$y_2 = (10,9) + 3(7,2)$$
$$= (10,9) + (3,5)$$
$$= (10,2)$$

Bởi vậy, y = ((8,3),(10,2)). Bây giờ nếu Bob nhận được bản mã y thì anh ta giải mã như sau:

$$x = (10,2) - 7(8,3)$$
$$= (10,2) - (3,5)$$
$$= (10,2) + (3,6)$$
$$= (10,9)$$

Đây chính là bản rõ đúng.

Trên thực tế có một số khó khăn khi áp dụng hệ mật Elgamal trên đường cong Elliptic. Hệ thống này được áp dụng trong Z_p (hoặc trong $GF(p^n)$ với $n > 1$) sẽ có hệ số mở rộng bản tin là 2. Việc áp dụng đường cong

Elliptic sẽ có thừa số mở rộng khoảng 4 lần. Điều này là do có xấp xỉ p bản rõ, nhưng mỗi bản mã lại gồm bốn phần tử của trường. Một trở ngại là không gian bản rõ chứa các điểm trên đường cong E và không có phương pháp nào xác định tường minh các điểm trên E

Menezes và Vanstone đã tìm ra một phương án hiệu quả hơn. theo phương án này đường cong Elliptic dùng để "che dấu", còn các bản rõ và bản mã hợp lệ là các cặp được sắp tùy ý các phần tử khác không của trường(tức là chúng không đòi hỏi phải là các điểm trên E). Điều này sẽ tạo hệ số mở rộng bản tin là 2 giống như trong hệ mật Elgamal ban đầu. Hệ mật Menezes - Vanstone được mô tả trên hình 5.10.

Nếu trở lại đường cong $y^2 = x^3 + x + 6$ trên Z_{11} ta sẽ thấy rằng hệ mật Menezes - Vanstone có $10\times10 = 100$ bản rõ, trong khi đó hệ mật ban đầu chỉ có 13 bản rõ. Ta sẽ minh hoạ phép mã và giải mã trong hệ mật này bằng cách sử dụng đường cong trên.

Hình 3.6 Hệ mật trên đường cong Elliptic của Menezes - Vanstone

Giả sử E là một đường cong Elliptic trên **Z**p (p là số nguyên tố > 3) sao cho E chứa một nhóm con cyclic H, trong đó bài toán DL là bài toán khó.

Giả sử $\mathcal{P} = Z_p^* \times Z_p^*$, $\mathcal{C} = E \times Z_p^* \times Z_p^*$,ta định nghĩa:

$$\mathcal{K} = \{ (E,\alpha,a,\beta) : \beta = a\,\alpha \}$$

trong đó $\alpha \in E$. Các giá trị α và β được công khai, còn a được giữ kín.

Đối với $K = (E,\alpha,a,\beta)$, với số ngẫu nhiên bí mật $k \in Z_{|H|}$

và $x = (x_1,x_2) \in Z_p^* \times Z_p^*$, ta xác định:

$$e_K(x,k) = (y_0,y_1,y_2)$$

$$y_0 = k\,\alpha$$

$$(c_1,c_2) = k\,\beta$$

$$y_1 = c_1 x_1 \bmod p$$

và $$y_2 = c_2 y_2 \bmod p$$

Với bản mã $y = (y_0,y_1,y_2)$, ta định nghĩa

$$d_K(y) = (y_1 c_1^{-1} \bmod p,\ y_2 c_2^{-1} \bmod p)$$

trong đó $a\,y_0 = (c_1,c_2)$

Ví dụ 5.9

Cũng như ví dụ trước, giả sử $\alpha = (2,7)$ và số mũ mật của Bob là 7. Khi đó

$$\beta = 7\alpha = (7,2)$$

Giả sử Alice muốn mã hoá bản rõ sau:

$$x = (x_1,x_2) = (9,1)$$

(Cần chú ý là x không phải là một điểm trên E) và cô chọn giá trị ngẫu nhiên k = 6. Đầu tiên cô tính:

$$y_0 = k\alpha = 6(2,7) = (7,9)$$

và

$$k\beta = 6(7,2) = (8,3)$$

Như vậy, $c_1 = 8$ còn $c_2 = 3$.

Tiếp theo Alice tính:

$$y_1 = c_1x_1 \bmod p = 8\times9 \bmod 11 = 6$$

và

$$y_2 = c_2x_2 \bmod p = 3\times1 \bmod 11 = 3$$

Bản mã mà cô giửi cho Bob là:

$$y = (y_0,y_1,y_2) = ((7,9), 6, 3)$$

Khi Bob nhận được bản mã này, Trước tiên anh ta tính:

$$(c_1,c_2) = (a\ y_0) = 7(7,9) = (8,3)$$

và sau đó tính:

$$\begin{aligned} x &= (y_1c_1^{-1} \bmod p, y_2c_2^{-1} \bmod p) \\ &= ((6\times8^{-1} \bmod 11, 3\times3^{-1} \bmod 11) \\ &= (6\times7 \bmod 11, 3\times4 \bmod 11) \\ &= (9,1). \end{aligned}$$

Hình 5.11. Bài toán tổng các tập con

Đặc trưng của bài toán: I = (s_1, s_2, . . . ,s_n, T) trong đó s_1, . . ., s_n và T là các số nguyên dương. Các s_i được gọi là các cỡ, T được gọi là tổng đích.

Vấn đề: Liệu có một véc tơ nhị phân $x = (x_1, . . . , x_n)$ sao cho:

$$\sum_{i=0}^{n} x_i s_i = T\ ?$$

Đây chính là bản rõ đúng.

5.3. HỆ MẬT XÊP BA LÔ MERKLE - HELLMAN

CHƯƠNG 6

CÁC SƠ ĐỒ CHỮ KÍ SỐ

6.1 GIỚI THIỆU.

Trong chương này, chúng ta xem xét các sơ đồ chữ kí số (còn được gọi là chữ kí số). Chữ kí viết tay thông thường trên tàI liệu thường được dùng để xác người kí nó. Chữ kí được dùng hàng ngày chẳng hạn như trên một bức thư nhận tiền từ nhà băng, kí hợp đồng...

Sơ đồ chữ kí là phương pháp kí một bức điện lưu dưới dang điện từ. Chẳng hạn một bức điện có ký hiệu được truyền trên mạng máy tinh. Chương trình này nghiên cứu vài sơ đồ chữ kí. Ta sẽ thảo luận trên một vàI khác biệt cơ bản giửa các chữ kí thông thường và chữ kí số.

Đầu tiên là một vấn đề kí một tài liệu. Với chữ kí thông thường, nó là một phần vật lý của tài liệu. Tuy nhiên, một chữ kí số không gắn theo kiểu vật lý vào bức điện nên thuật toán được dùng phải "không nhìn thấy" theo cách nào đó trên bức điện.

Thứ hai là vấn đề về kiểm tra. Chữ kí thông thường được kiểm tra bằng cách so sánh nó với các chữ kí xác thực khác. ví dụ, ai đó kí một tấm séc để mua hàng, người bán phải so sánh chữ kí trên mảnh giấy với chữ kí nằm ở mặt sau của thẻ tín dụng để kiểm tra. Dĩ nhiên, đây không phải là phương pháp an toàn vì nó dể dàng giả mạo. Mắt khác, các chữ kí số có thể được kiểm tra nhờ dùng một thuật toán kiểm tra công khai. Như vậy, bất kỳ ai cũng có thể kiểm tra dược chữ kí số. Việc dùng một sơ đồ chữ kí an toàn có thể sẽ ngăn chặn dược khả năng giả mạo.

Sự khác biệt cơ bản khác giữa chữ kí số và chữ kí thông thường bản copy tài liệu được kí băng chữ kí số đồng nhất với bản gốc, còn copy tài liệu có chữ kí trên giấy thường có thể khác với bản gốc. Điều này có nghĩa là phải cẩn thận ngăn chăn một bức kí số khỏi bị dung lạI. Ví dụ, Bob kí một bức điện xác nhận Alice có khả năng làm điều đó một lần. Vì thế, bản thân bức điện cần chứa thông tin (chẳng hạn như ngày tháng) để ngăn nó khỏi bị dung lại.

Một sơ đồ chữ kí số thường chứa hai thành phần: thuật toán kí và thuật toán xác minh. Bob có thể kí điện x dùng thuật toán kí an toàn. Chữ kí sig(x) nhận được có thể kiểm tra bằng thuật toán xác minh công khai ver. Khi cho trước cặp (x,y), thuật toán xác minh có giá trị TRUE hay FALSE tuỳ thuộc vào chữ kí được thực như thế nào. Dưới đây là định nghĩa hình thức của chữ kí:

Định nghĩa 6.1

Một sơ đồ chữ kí số là bộ 5($\mathcal{P}$,$\mathcal{A}$, $\mathcal{K}$,$\mathcal{S}$,$\mathcal{V}$) thoả mãn các điều kiện dưới đây:

1. $\mathcal{P}$ là tập hữu hạn các bức điện có thể.
2. $\mathcal{A}$ là tập hữu hạn các chữ kí có thể.
3. $\mathcal{K}$ không gian khoá là tập hữu hạn các khoá có thể.
4. Với mỗi K thuộc $\mathcal{K}$ tồn tạI một thuật toán kí $sig_k \in \mathcal{S}$ và là một thuật toán xác minh $ver_k \in \mathcal{V}$. Mỗi $sig_k : \mathcal{P} \to \mathcal{A}$ và $ver_k: \mathcal{P}\times\mathcal{A} \to \{true,false\}$ là những hàm sao cho mỗi bức điện $x \in \mathcal{P}$ và mối chữ kí $y \in \mathcal{A}$ thoả mãn phương trình dưới đây.

$$ver_k \begin{cases} \text{True nếu } y=sig(x) \\ \\ \text{False nếu } y \# sig(x) \end{cases}$$

Với mỗi k thuộc $\mathcal{K}$ hàm sig_k và ver_k là các hàm thơì than đa thức. Ver_k sẽ là hàm công khai sig_k là mật. Không thể dể dàng tính toán để giả mạo chữ kí của Bob trên bức điện x. Nghĩa là x cho trước, chỉ có Bob mới có thể tính được y để ver_k = True. Một sơ đồ chữ kí không thể an toàn vô điều kiện vì Oscar có thể kiểm tra tất cả các chữ số y có thể có trên bức điện x nhờ dùng thuật toán ver công khai cho đến khi anh ta tìm thấy một chữ kí đúng. Vi thế, nếu có đủ thời gian. Oscar luôn luôn có thể giả mạo chữ kí của Bob. Như vậy, giống như trường hợp hệ thống mã khoá công khai, mục đích của chúng ta là tìm các sơ đồ chữ kí số an toan về mặt tính toán.

Xem thấy rằng, hệ thống mã khoá công khai RSA có thể dùng làm sơ đồ chữ kí số, Xem hình 6.1.

Như vậy, Bob kí bức điện x dùng qui tắc giải mã RSA là d_k,. Bob là người tạo ra chữ kí vì $d_k = sig_k$ là mật. Thuật toán xác minh dùng qui tắc mã RSA e_k. Bất kì ai củng có xác minh chữ kí vi e_k được công khai.

Chú ý rằng, ai đó có thể giả mạo chữ kí của Bob trên một bức điện " ngẩu nhiên" x bằng cách tìm $x=e_k(y)$ với y nào đó; khi đó $y= sig_k(x)$. Một pháp xung quanh vấn đề khó khăn này là yêu cầu bức điện chưa đủ phần dư để chữ kí giả mạo kiểu này không tương ứng với bức điện đây nghĩa là x trừ một xác suất rất bé. Có thể dùng các hàm hash trong việc kết nối với các sơ đồ chữ kí số sẽ loại trừ được phương pháp giả mạo này (các hàm hash được xét trong chương 7).

Hình 6.1 sơ đồ chữ kí RSA

Cho n= pq, p và q là các số nguyên tố. Cho $\mathcal{P} = \mathcal{A} = Z_n$ và định nghĩa $\mathcal{P}$= {(n,p,q,a,b):=n=pq,p và q là nguyên tố, $ab \equiv 1(\mathrm{mod}(\phi(n)))$ }. Các giá trị n và b là công khai, ta định nghĩa :

$sig_k(x) = x^a \bmod n$

và $ver_k(x,y) = \mathrm{true} \Leftrightarrow x \equiv y^b \pmod n$

$(x,y \in Z_n)$

Cuối cùng, ta xét tóm tắt các kết hợp chữ kí và mã khoá công khai. Giả sử rằng, Alice tính toán chư kí của ta y= $sig_{Alice}(x)$ và sau đó mã cả x và y bằng hàm mã khoá công khai e_{Bob} của Bob, khi đó cô ta nhận được $z = e_{Bob}(x,y)$. Bản mã z sẽ được truyền tới Bob. Khi Bob nhận được z, anh ta sẽ trước hết sẽ giải mã hàm d_{Bob} để nhận được (x,y). Sau đó anh ta dung hàm xác minh công khai của Alice để kiểm tra xem $ver_{Alice}(x,y)$ có bằng True hay không.

Song nếu đầu tiên Alice mã x rồi sau đó mới kí tên bản mã nhận được thì sao?. Khi đó cô tính :

$$y = sig_{Alice}(e_{Bob}(x)).$$

Alice sẽ truyền cặp (z,y) tới Bob. Bob sẽ giải mã z, nhận x và sau đó xác minh chữ kí y trên x nhờ dùng ver_{Alice}. Một vấn đề tiềm ẩn trong biện pháp này là nếu Oscar nhận được cặp (x,y) kiểu này, được ta có thay chữ kí y của Alice bằng chữ kí của mình.

$$y' = sig_{Oscar}(e_{Bob}(x)).$$

(chú ý rằng,Oscar có thể kí bản mã $e_{Bob}(x)$ ngay cả khi anh ta không biết bản rõ x). Khi đó nếu Oscar truyền(x, y) đến Bob thì chữ kí Oscar được Bob xác minh bằng ver_{Oscar} và Bob có thể suy ra rằng, bản rõ x xuất phát từ Oscar. Do khó khăn này, hầu hết người sử dụng được khuyến nghị nếu kí trước khi mã.

6.2 SƠ ĐỒ CHỮ KÍ ELGAMAL

Sau đây ta sẽ mô tả sơ đồ chữ kí Elgamal đã từng dưới thiệu trong bài báo năm 1985. Bản cả tiến của sơ đồ này đã được Viện Tiêu chuẩn và Công Nghệ Quốc Gia Mỹ (NIST) chấp nhận làm chữ kí số. Sơ đồ Elgamal (E.) được

thiết kế với mục đích dành riêng cho chữ kí số, khác sơ đồ RSA dùng cho cả hệ thống mã khoá công khai lẫn chữ kí số.

Sơ đồ E, là không tất định giống như hệ thống mã khoá công khai Elgamal. Điều này có nghĩa là có nhiều chữ kí hợp lệ trên bức điện cho trước bất kỳ. Thuật toán xác minh phải cố khải năng chấp nhận bất kì chữ kí hợp lệ khi xác thực. Sơ đồ E. được môt tả trên hình 6.2

Nếu chữ kí được thiết lập đúng khi xác minh sẽ thành công vì :

$$\beta^{\gamma}\gamma^{\delta} \equiv \alpha^{a\gamma}\alpha^{k\gamma} (\text{mod } p)$$
$$\equiv \alpha^{x} (\text{mod } p)$$

là ở đây ta dùng hệ thức :

$$a\gamma + k\delta \equiv x \ (\text{mod } p-1)$$

Hình 6.2 sơ đồ chữ kí số Elgamal.

Cho p là số nguyên tố sao cho bài toán log rời rạc trên Z_p là khó và giả sử $\alpha \in Z_n$ là phần tử nguyên thuỷ $\mathcal{P} = Z_p^*$, $\mathcal{A} = Z_p^* \times Z_{p-1}$ và định nghĩa :

$$\mathcal{K} = \{(p,\alpha,a,\beta) : \beta \equiv \alpha^a (\text{mod } p)\}.$$

Giá trị p,α,β là công khai, còn a là mật.

Với $\mathcal{K} = (p,\alpha,a,\beta)$ và một số ngẫu nhiên (mật) $k \in Z_{p-1}$. định nghĩa :

$$Sig_k(x,y) = (\gamma,\delta),$$

trong đó $\gamma = \alpha^k \bmod p$

và $\delta = (x-a)\, k^{-1} \bmod (p-1)$

Với $x,\gamma \in Z_p$ và $\delta \in Z_{p-1}$, ta định nghĩa :

$$Ver(x,\gamma,\delta) = \text{true} \Leftrightarrow \beta^{\gamma}\gamma^{\delta} \equiv \alpha^{x} (\text{mod } p).$$

Bob tính chữ kí bằng cách dùng cả gía trị mật a (là một phần của khoá) lẫn số ngẫu nhiên mật k (dùng để kí lên bức điện x). Việc xác minh có thực hiện duy nhất bằng thông báo tin công khai.

Chúng ta hãy xét một ví dụ nhỏ minh hoạ.

Ví dụ 6.1

Giả sử cho p = 467, α =2,a = 127; khi đó:

$\beta = \alpha^a \bmod p$
$= 2^{127} \bmod 467$
$= 132$

Nếu Bob muốn kí lên bức điện x = 100 và chọn số ngẫu nhiên k =213 (chú ý là UCLN(213,466) =1 và $213^{-1} \bmod 466 = 431$. Khi đó

$\gamma = 2^{213} \bmod 467 = 29$

và $\delta = (100-127 \times 29)\ 431 \bmod 466 = 51.$

Bất kỳ ai củng có thể xác minh chữ kí bằng các kiểm tra :

$$132^{29}\ 29^{51} \equiv 189 \pmod{467}$$

và

$$2^{100} \equiv 189 \pmod{467}$$

Vì thế chữ kí là hợp lệ.

Xét độ mật của sơ đồ chữ kí E. Giả sử, Oscar thử giả mạo chữ kí trên bức điện x cho trước không biết a. Nếu Oscar chọn γ và sau đó thử tìm giá trị δ tương ứng, anh ta phải tính logarithm rời rạc $\log_\gamma \alpha^x\beta^{-\gamma}$. Mặt khác, nếu đầu tiên ta chọn δ và sau đó thử tim γ và thử giải phương trình:

$$\beta^\gamma \gamma^\delta \equiv \alpha^x \pmod p.$$

để tìm γ. Đây là bài toán chưa có lời giải nào: Tuy nhiên, dường như nó chưa được gắn với đến bài toán đã nghiên cứu kĩ nào nên vẫn có khả năng có cách nào đó để tính δ và γ đồng thời để (δ, γ)là một chữ kí. Hiện thời không ai tìm được cách giải song củng ai không khẳng định được rằng nó không thể giải được.

Nếu Oscar chọn δ và γ và sau đó tự giải tìm x, anh ta sẽ phảI đối mặt với bài toán logarithm rời rạc, tức bài toán tính $\log_\alpha$??? Vì thế Oscar không thể kí một bức điện ngẫu nhiên bằng biện pháp này. Tuy nhiên, có một cách để Oscar có thể kí lên bức điện ngẫu nhiên bằng việc chọn γ, δ và x đồng thời: giả thiết i và j là các số nguyên $0 \le i \le p-2$, $0 \le j \le p-2$ và UCLN(j,p-2) = 1. Khi đó thực hiện các tính toán sau:

$\gamma = \alpha^i \beta^j \bmod p$
$\delta = -\gamma\, j^{-1} \bmod (p-1)$
$x = -\gamma\, i\, j^{-1} \bmod (p-1)$

trong đó j^{-1} được tính theo modulo (p-1) (ở đây đòi hỏi j nguyên tố cùng nhau với p-1).

Ta nói rằng (γ, δ)là chữ kí hợp lệ của x. Điều này được chứng minh qua việc kiểm tra xác minh :

????

Ta sẽ minh hoạ bằng một ví dụ :

Ví dụ 6.2.

Giống như ví dụ trước cho p = 467, α = 2, β =132. Giả sữ Oscar chọn i = 99,j = 179; khi đó j^{-1} mod (p-1) = 151. Anh ta tính toán như sau:

$\gamma = 2^{99}132^{197} \bmod 467 = 117$

$\delta = -117 \times 151 \bmod 466 = 51.$

$x = 99 \times 41 \bmod 466 = 331$

Khi đó (117, 41) là chữ kí hợp lệ trên bức điện 331 hư thế đã xác minh qua phép kiểm tra sau:

$132^{117}\ 117^{41} \equiv 303 \pmod{467}$

và $2^{331} \equiv 303 \pmod{467}$

Vì thế chữ kí là hợp lệ.

Sau đây là kiểu giả mạo thứ hai trong đó Oscar bắt đầu bằng bức điện được Bob kí trước đây. Giả sử (γ, δ) là chữ kí hợp lệ trên x. Khi đó Oscar có khả năng kí lên nhiều bức điện khác nhau. Giả sử i, j, h là các số nguyên, $0 \le h, i, j \le p-2$ và UCLN (h γ - j δ, p-1) = 1. Ta thực hiện tính toán sau:

$\lambda = \gamma^h \alpha^i \beta^j \bmod p$

$\mu = \delta\lambda(h\gamma - j\delta)^{-1} \bmod (p-1)$

$x' = \lambda(hx + i\delta)^{-1} \bmod (p-1),$

trong đó $(h\gamma - j\delta)^{-1}$ được tính theo modulo (p-1). Khi đó dễ dàng kiểm tra điệu kiện xác minh :

$\beta^{\lambda} \lambda^{\mu} \equiv \alpha^{x'} \pmod{p}$

vì thế (λ, μ)là chữ kí hợp lệ của x'.

Cả hai phương pháp trên đều tạo các chữ kí giả mạo hợp lệ song không xuất hiện khả năng đối phương giả mạo chữ kí trên bức điện có sự lựu chọn của chính họ mà không phải giải bài toán logarithm rời rạc, vì thế không có gì nguy hiểm về độ an toàn của sơ đồ chữ kí Elgamal.

Cuối cùng, ta sẽ nêu vài cách có thể phái được sơ đồ này nếu không áp dụng nó một cách cẩn thận (có một số ví dụ nữa về khiếm khuyết của giao thức, một số trong đó là xét trong chương 4). Trước hết, giá trị k ngẫu nhiên được dùng để tính chữ kí phải giữ kín không để lộ. vì nếu k bị lộ, khá đơn giản để tính :

$$A = (x\text{-}k\,\gamma)\delta^{-1} \bmod (p\text{-}1).$$

Dĩ nhiên, một khi a bị lộ thì hệ thống bị phá và Oscar có thể dễ dang giả mạo chữ kí.

Một kiểu dung sai sơ đồ nữa là dùng cùng giá trị k để kí hai bức điện khác nhau. điều này cùng tạo thuận lợi cho Oscar tinh a và phá hệ thống. Sau đây là cách thực hiện. Giả sử (γ, δ_1) là chữ kí trên x_1 và (γ, δ_2) là chữ kí trên x_2. Khi đó ta có:

$$\beta^{\gamma}\gamma^{\delta_1} \equiv \alpha^{x_1} \pmod p$$

và

$$\beta^{\gamma}\gamma^{\delta_2} \equiv \alpha^{x_2} (\text{mod} p).$$

Như vậy

$$\alpha^{x_1-x_2} \equiv \alpha^{\delta_1-\delta_2} \pmod p.$$

Nếu viết $\gamma = \alpha^k$, ta nhận được phương trình tìm k chưa biết sau.

$$\alpha^{x_1-x_2} \equiv \alpha^{k(\delta_1-\delta_2)} \pmod p$$

tương đương với phương trình

$$x_1 - x_2 \equiv k(\delta_1 - \delta_2) \pmod{p-1}.$$

Bây giờ giả sử d =UCLN$(\delta_1 - \delta_2, p-1)$. Vì d | (p-1) và d | $(\delta_1-\delta_2)$ nên suy ra d | (x_1-x_2). Ta định nghĩa:

$$x' = (x_1 - x_2)/d$$
$$\delta' = (\delta_1 - \delta_2)/d$$
$$p' = (p-1)/d$$

Khi đó đồngdư thức trở thành:

$$x' \equiv k\,\delta' \pmod{p'}$$

vì UCLN$(\delta', p') = 1$,nên có thể tính:

$$\varepsilon = (\delta')^{-1} \bmod p'$$

Khi đó giá trị k xác định theo modulo p' sẽ là:

$$k = x' \varepsilon \bmod p'$$

Phương trình này cho d giá trị có thể của k

$$k = x' \varepsilon + i\, p' \bmod p$$

với i nào đó, $0 \le i \le d-1$. Trong số d giá trị có có thế này, có thể xác định được một giá trị đúng duy nhất qua việc kiểm tra điều kiện

$$\gamma \equiv \alpha^k \pmod p$$

6.3 CHUẨN CHỮ KÍ SỐ.

Chuẩn chữ kí số(DSS) là phiên bản cải tiến của sơ đồ chữ kí Elgamal. Nó được công bố trong Hồ Sơ trong liên bang vào ngày 19/5/94 và được làm chuẩn vào 1/12/94 tuy đã được đề xuất từ 8/91. Trước hết ta sẽ nêu ra những thay đổi của nó so với sơ đồ Elgamal và sau đó sẽ mô tả cách thực hiện nó.

Trong nhiều tinh huống, thông báo có thể mã và giải mã chỉ một lần nên nó phù hợp cho việc dùng với hệ mật Bất kì (an toàn tại thời điểm được mã). Song trên thực tế, nhiều khi một bức điện được dùng làm một tài liệu đối chứng, chẳng hạn như bản hợp đồng hay một chúc thư và vì thế cần xác minh chữ kí sau nhiều năm kể từ lúc bức điện được kí. Bởi vậy, điều quan trọng là có phương án dự phòng liên quan đến sự an toàn của sơ đồ chữ kí khi đối mặt với hệ thống mã. Vì sơ đồ Elgamal không an toàn hơn bài toán logarithm rời rạc nên cần dung modulo p lớn. Chắc chắn p cần ít nhất là 512 bít và nhiều người nhất trí là p nên lấy p=1024 bít để có độ an toàn tốt.

Tuy nhiên, khi chỉ lấy modulo p =512 thì chữ kí sẽ có 1024 bít. Đối với nhiều ứng dụng dùng thẻ thông minh thì cần lại có chữ kí ngắn hơn. DSS cải tiến sơ đồ Elgamal theo hướng sao cho một bức điện 160 bít được kí bằng chữ kí 302 bít song lại p = 512 bít. Khi đó hệ thống làm việc trong nhóm con Z_n^* kích thước 2^{160}. Độ mật của hệ thống dựa trên sự an toàn của việc tìm các logarithm rời rạc trong nhóm con Z_n^*.

Sự thay đổi đầu tiên là thay dấu " - " bằng "+" trong định nghĩa δ, vì thế:

$$\delta = (x + \alpha\, \gamma)k^{-1} \bmod (p-1)$$

thay đổi kéo theo thay đổi điều kiện xác minh như sau:

$$\alpha^x \beta^\gamma \equiv \gamma^\delta \pmod p \quad (6.1)$$

Nếu UCLN $(x + \alpha\gamma, p-1) = 1$ thì $\delta^{-1} \bmod (p-1)$ tồn tại và ta có thể thay đổi điều kiện (6.1) như sau:

$$\alpha^{x\delta^{-1}} \beta^{\gamma\delta^{-1}} \equiv \gamma \pmod p \quad (6.2)$$

Đây là thay đổi chủ yếu trong DSS. Giả sử q là số nguyên tố 160 bít sao cho $q \mid (q-1)$ và α là căn bậc q của một modulo p. (Dễ dàng xây dựng một α như vậy: cho α_0 là phần tử nguyên thuỷ của Z_p và định nghĩa $\alpha = \alpha_0^{(p-1)/q} \bmod p$).

Khi đó β và γ cũng sẽ là căn bậc q của 1. vì thế các số mũ Bất kỳ của α, β và γ có thể rút gọn theo modulo q mà không ảnh hưởng đến điều kiện xác minh (6.2). Điều rắc rối ở đây là γ xuất hiện dưới dạng số mũ ở vế trái của (6.2) song không như vậy ở vế phải. Vì thế, nếu γ rút gọn theo modulo q thì cũng phải rút gọn toàn bộ vế trái của (6.2) theo modulo q để thực hiện phép kiểm tra. Nhận xét rằng, sơ đồ (6.1) sẽ không làm việc nếu thực hiện rút gọn theo modulo q trên (6.1). DSS được mô tả đầy đủ trong hinh 6.3.

Chú ý cần có $\delta \not\equiv 0 \pmod q$ vì giá trị $\delta^{-1} \bmod q$ cần thiết để xác minh chữ kí (điều này tương với yêu cầu UCLN$(\delta, p-1) = 1$ khi biến đổi (6.1) thành (6.2). Nếu Bob tính $\delta \equiv 0 \pmod q$ theo thuật toán chữ kí, anh ta sẽ loại đi và xây dựng chữ kí mới với số ngẫu nhiên k mới. Cần chỉ ra rằng, điều này có thể không gần vấn đề trên thực tế: xác xuất để $\delta \equiv 0 \pmod q$ chắc sẽ xảy ra cở 2^{-160} nên nó sẽ hầu như không bao giờ xảy ra.

Dưới đây là một ví dụ minh hoạ nhỏ

Hình 6.3. Chuẩn chữ kí số.

Giả sử p là số nguyên tố 512 bít sao cho bài toán logarithm rời rạc trong Z_p khong Giải được, cho p là số nguyên tố 160 bít là ước của (p-1). Giả thiết $\alpha \in Z_p$ là căn bậc q của 1modulo p: Cho $\mathcal{P} = Z_p$. $\mathcal{A} = Z_q \times Z_p$ và định nghĩa :

$$\mathcal{A} = \{(p,q,\alpha,a,\beta) : \beta \equiv \alpha^a \pmod p\}$$

các số p, q, α và β là công khai, có a mật.

Với K = (p,q,α ,a,β)và với một số ngẫu nhiên (mật) k ,$1 \le k \le q-1$, ta định nghĩa:

$$sig_k(x,k) = (\gamma, \delta)$$

trong đó $\gamma = (\alpha^k \bmod p) \bmod q$

và $\delta = (x + a\gamma)k^{-1} \bmod q$

Với $x \in Z_p$ và $\gamma, \delta \in Z_q$, qua trình xác minh sẽ hoàn toàn sau các tính toán :

$$e_1 = x\delta^{-1} \bmod q$$

$$e_2 = \gamma\delta^{-1} \bmod q$$

$$ver_k(x, \gamma, \delta) = true \Leftrightarrow (\alpha^{e_1}\beta^{e_2} \bmod p) \bmod q = \gamma$$

Ví dụ 6.3:

Giả sử q =101, p = 78 q+1 =7879.3 là phần tử nguyên thuỷ trong Z_{7879} nên ta có thể lấy: $\alpha = 3^{78} \bmod 7879 = 170$

Giả sử a =75, khi đó :

$$\beta = \alpha^a \bmod 7879 = 4576$$

Bây giờ giả sữ Bob muốn kí bức điện x = 1234 và anh ta chọn số ngẫu nhiên k =50, vì thế :

$$k^{-1} \bmod 101 = 99$$

khi đó $\gamma = (170^{30} \bmod 7879) \bmod 101$

$= 2518 \bmod 101$

$= 94$

và $\delta = (1234 + 75 \times 94) \bmod 101$

$= 96$

Chữ kí (94, 97) trên bức điện 1234 được xác minh bằng các tính toán sau:

$$\delta^{-1} = 97^{-1} \bmod 101 = 25$$

$$e_1 = 1234 \times 25 \bmod 101 = 45$$
$$e_2 = 94 \times 25 \bmod 101 = 27$$
$$(170^{45}\ 4567^{27} \bmod 7879) \bmod = 2518 \bmod 101 = 94$$

vì thế chữ kí hợp lệ.

Khi DSS được đề xuất năm 1991, đã có một vài chỉ trích đưa ra. Một ý kiến cho rằng, việc xử lý lựa chọn của NIST là không công khai. Tiêu chuẩn đã được Cục An ninh Quốc gia (NSA) phát triển mà không có sự tham gia của khôi công nghiệp Mỹ. Bất chấp những ưu thế của sơ đồ, nhiều người đã đóng chặt cửa không tiếp nhận.

Còn những chỉ trích về mặt kĩ thuật thì chủ yếu là về kích thước modulo p bị cố định = 512 bít. Nhiều người muốn kích thước này có thể thay đổi được nếu cần, có thể dùng kích cỡ lớn hơn. Đáp ứng những đòi hỏi này, NIST đã chọn tiêu chuẩn cho phép có nhiều cở modulo, nghĩa là cỡ modulo bất kì chia hết cho 64 trong phạm vi từ 512 đến 1024 bít.

Một phàn nàn khác về DSS là chữ kí được tạo ra nhanh hơn việc xác minh nó. Trong khi đó, nếu dùng RSA làm sơ đồ chữ kí với số mũ xác minh công khai nhỏ hơn (chẳng hạn = 3) thì có thể xác minh nhanh hơn nhiều so với việc lập chữ kí. Điều này dẫn đến hai vấn đề liên quan đến những ứng dụng của sơ đồ chữ kí:

1.Bức điện chỉ được kí một lần, song nhiều khi lại cần xác minh chữ kí nhiều lần trong nhiều năm. Điều này lại gợi ý nhu cầu có thuật toán xác minh nhanh hơn.

2.Những kiểu máy tính nào có thể dùng để kí và xác minh ?. Nhiều ứng dụng, chẳng hạn các thẻ thông minh có khả năng xử lý hạn chế lại liên lạc với máy tính mạnh hơn. Vi thế có nhu cầu nhưng thiết kế một sơ đồ để có thực hiện trên thẻ một vài tính toán. Tuy nhiên, có những tình huống cần hệ thống mình tạo chữ kí, trong những tình huống khác lại cần thẻ thông minh xác minh chữ kí. Vì thế có thể đưa ra giải pháp xác định ở đây.

Sự đáp ứng của NIST đối với yêu cầu về số lần tạo xác minh chữ kí thực ra không có vấn đề gì ngoài yêu cầu về tốc độ, miễn là cả hai thể thực hiện đủ nhanh.

6.4 CHỮ KÍ MỘT LẦN

Trong phần, này chúng ta mô tả cách thiết lập đơn giản một sơ đồ chữ kí một lần từ hàm một chiều. Thuật ngữ "một lần" có nghĩa là bức điện được kí chỉ một lần (dĩ nhiên chữ kí có thể xác minh nhiều lần tuỳ ý). Sơ đồ mô tả là sơ đồ chữ kí Lamport nêu hình 6.4.

Sơ đồ làm viêc như sau: Bức điện được kí là một bức điện nhị phân k bít. Một bít được kí riêng biệt nhau. Giá trị $z_{i,j}$ tương ứng với bít thứ i của bức điện có giá trị j (j =0,1). Mỗi $z_{i,j}$ là ảnh hưởng đến $y_{i,j}$ dưới tác động của hàm một chiều f. Bít thứ i của bức điện được kí nhờ là ảnh gốc(nghịch ảnh - priemage) $y_{i,j}$ của $z_{i,j}$ (tương ứng với bít thứ i của bức điện). Việc xác minh chỉ đơn giản là kiểm tra xem mỗi phần tử trong chữ kí có là ảnh gốc của phần tử

Hình 6.4. Sơ đồ chữ kí Lamport

Cho k là số nguyên dương và cho $\mathcal{P} = \{0,1\}^k$. Giả sử f:Y → Z là hàm một chiều và cho $\mathcal{A} = Y^k$. Cho $y_{i,j} \in Y$ được chọn ngẫu nhiên. $1 \le i \le k$, j =0,1 và giả sử $z_{i,j} = f(y_{i,j})$. Khoá K gồm 2k giá trị y và 2k giá trị z. Các giá trị của i giữ bí mật trong khi các giá trị của z công khai.

Với $K = (y_{i,j}, z_{i,j} : 1 \le i \le k, j = 0,1)$, ta định nghĩa :

$$sig_k(x_1 \ldots x_k) = (????\text{tự đánh vào})$$

và $ver_k(x_1 \ldots x_k, a_1 \ldots a_k) = true \Leftrightarrow f(a_i) = ????\text{tự đánh vào}$

khoá công khai thích hợp hay không.

Sau đây sẽ minh hoạ sơ đồ bằng việc xem xét một thực hiện dùng hàm mũ $f(x) = \alpha^x \bmod p$. α là một phần tử nguyên thuỷ modulo p.

Ví dụ 6.4

7879 là số nguyên tố và 3 là phần tử nguyên thuỷ thuộc Z_{7879}. Định nghĩa:

$$f(x) = 3^x \bmod 7879$$

Giả sử Bob muốn kí một bức điện có 3 bít. Anh ta chọn 6 số tự nhiên (mật)

$y_{1,0} = 5831$ $y_{2,1} = 2467$
$y_{1,1} = 735$ $y_{3,0} = 4285$
$y_{2,0} = 803$ $y_{3,1} = 6449$

Khi đó, anh ta tính các ảnh của y dưới hàm f

$z_{1,0} = 2009$ $z_{2,1} = 4721$
$z_{1,1} = 3810$ $z_{3,0} = 268$
$z_{2,0} = 4672$ $z_{3,1} = 5731$

Các ảnh của z này được công khai. Bây giờ giả sử Bob muốn ký bức điện

$x = (1, 1, 0)$

chữ kí trên x là:

$(y_{1,1}, y_{2,1}, y_{3,0}) = (735, 2467, 4285)$

Để xác minh chữ kí, chỉ cần tính toán như sau:

$3^{735} \bmod 7879 = 3810$
$3^{4675} \bmod 7879 = 4721$
$2^{4285} \bmod 7879 = 268$

Vì thế, chữ kí hợp lệ.

Oscar không thể giả mạo chữ kí vì anh ta không thể đảo được hàm một chiều f(x) để có các giá trị y mật. Tuy nhiên, sơ đồ được dùng để kí chỉ một bức điện. Bởi vì nếu cho trước chữ kí của 2 bức điện khác nhau. Oscar sẽ dễ dàng xây dựng chữ kí cho bức điện khác.

Ví dụ, giã sử các bức điện (0, 1, 1) và (1, 0, 1) đều được kí bằng cùng một sơ đồ. Bức điện (0, 1, 1) có chữ kí $(y_{1,0}, y_{2,1}, y_{3,1})$ còn bức điện (1,0,1) có chữ kí $(y_{1,1}, y_{2,0}, y_{3,1})$. Nếu cho trước 2 chữ kí này, Oscar có thể xây dựng các chữ kí của bức điện (1,1,1) là $(y_{1,1}, y_{2,1}, y_{3,1})$ và chữ kí cho bức điện (0,0,10 là $(y_{1,0}, y_{2,0}, y_{3,1})$.

Mặc dù sơ đồ này hoàn toàn tốt song nó không được sử dụng trong thực do kích thước chữ kí. Ví dụ, nếu ta dùng hàm số mũ modulo như trong ví dụ ở trên thì yêu cầu an toàn đòi hỏi p dài ít nhất 512 bít. Điều này, có nghĩa mỗi bít của bức điện chữ kí dùng 512 bít. Kết quả chữ kí dài hơn bức điện 512 lần.

Bây giờ xét một cải tiến của Bos và Chaum cho phép chữ kí ngăn hơn một chút song không giảm độ mật. Trong sơ đồ Lamport, lý do Oscar không thể giả mão chữ kí trên bức điện (thứ hai) khi biết chữ kí ở bức điện là: các

ảnh của y (tương ứng với một bức điện) không bao giờ là tập con của các ảnh của y (tương ứng với bức điện khác).

Giả sử ta có tập $\mathcal{B}$ gồm các tập con của B sao cho $B_1 \subseteq B_2$ chỉ khi $B_1 = B_2$ với mọi $B_1, B_2 \in \mathcal{B}$. Khi đó $\mathcal{B}$ được gọi là thoả mãn tính chất Sperner. Cho trước một tập B có lực lượng n chẵn, khi đó kích thước cực đại của tập $\mathcal{B}$ gồm các tập con B có tính chất Sperner là $\binom{2n}{n}$. Điều này dễ dàng nhận được bằng cách lấy tất cả các tập con n của B: rõ ràng không có tập con n nào nhận được trong tập con n khác

Bây giờ, giã sử ta muốn ki một bức điện k bít như trước đây, ta chọn n đủ lớn để.

$$2^k \leq \binom{2n}{n}$$

Cho | B | =n và giả sữ $\mathcal{B}$ chỉ tập các tập con n của B. Giả sử $\phi:\{0,1\}^k \rightarrow \mathcal{B}$ là đơn ánh trong công khai đã biết. Khi đó, có thể liên kết mỗi bức điện có thể với một con n trong $\mathcal{B}$. Ta sẽ có 2n giá trị của y, và 2n giá trị của z và mỗi bức điện được kí bằng n ảnh của y. Hình 6.5 mô tả đầy đủ sơ đồ Bos- chaum.

Hình 6.5 Sơ đồ chữ kí Bos - chaum.

Cho k là số nguyên dương và giả sử $\mathcal{P}=\{0,1\}^k$. Cho n là số nguyên sao cho $2^k \leq \binom{2n}{n}$ và B là tập có lực lượng n và cho

$$\phi: \{0,1\}^k \rightarrow \mathcal{B}$$

là một đơn ánh , trong đó $\mathcal{B}$ là tập tất cả các con n của B. Giả sử f: Y→Z là hàm một chiều và $\mathcal{A} = Z^n$. Cho ??????????????

Ưu điểm của sơ đồ Bos- chaum là các chữ kí ngăn hơn sơ đồ Lamport. Ví dụ, ta muốn ký một bức điện 6 bit (k = 6). Vì 2^6 =64 và $\binom{8}{2}$ =70 nên có thể lấy n =4 và bức điện 6 bit được kí bằng 4 giá trị của y so với 6 của sơ đồ Lamport. Như vậy khoá k sẽ ngắn hơn, nó gồm 8 giá trị của z so với 12 của sơ đồ Lamport.

Sơ đồ Bos-Chaum đòi hỏi hàm đơn ánh ϕ để kết hợp tập con n của tập 2n với mỗi x nhị phân bội k (x_1 …. x_k). Ta sẽ đưa ra một thuật toán đơn giản để thực hiện điều này (hinh 6.6). Ví dụ, áp dụng thuật toán này với x = (0,1,0,0,1,1) sẽ tạo ra.

$$\phi(x) = \{2,4,6,8\}$$

Nói chung, n trong sơ đồ Bos-Chaum lớn bao nhiêu so với k ?. Ta cần thoả mãn bất phương trình $2^k \le \binom{2n}{n}$ Nếu đánh giá hệ số của nhị thức

$$\binom{2n}{2} = (2n)!/(n!)^2$$

Hình 6.6 ***Tính ϕ trong sơ đồ Bos - chaum***

1. $X = \sum_{i-1}^{k} x_i 2^{i-2}$
2. $\phi(x) = 0$
3. $t = 2n$
4. $e = n$
5. While $t > 0$ do
6. $t = t - 1$
7. if $x > \binom{t}{e}$ then
8. $x = x - \binom{t}{e}$
9. $e = e - 1$
10. $\phi(x) = \phi(x) \cup \{t+1\}$

băng công thức Stirling $2^{2n} / \sqrt{\pi n}$ Sau vài phép biến đổi đơn giản, bất kỳ đẳng thức trở thành

$$k \le 2n - \log_2 (\pi n)/2$$

Một cách gần đúng, $n \approx k/2$. Như vậy, ta đã giảm được khoảng 50% kích thước chữ kí bằng sơ đồ Bos - chaum.

6.5 CÁC CHỮ KÍ KHÔNG CHỐI ĐƯỢC

Các chữ kí không chối được do Chaum và Antwerpen đưa ra từ năm 1989. Chúng có vài đặc điểm mới. Nguyên thuỷ nhất trong các chữ kí này là chữ kí không thể xác minh được nếu không hợp tác với người ký là Bob. Như vậy sẽ bảo được Bob trước khả năng các tài liệu được anh ta ký bị nhân đôi và phân phối bằng phương pháp điện tử mà không có sự đồng ý của anh ta. Việc xác minh được thực hiên bằng giao thức yêu cầu và đáp ứng (Challege and repotocol).

Song liệu có cần sự hợp tác của Bob để xác minh chữ kí (nhằm ngăn chặn Bob từ chối không nhận đã ký trước đó) không? Bob có thể truyền thống chữ kí hợp lệ là giả mạo và từ chối xác minh nó, hoặc thực hiện giao thức theo cách để chữ kí không thể được xác minh. Để ngăn chặn tình huống này xảy ra, sơ đồ chữ kí không chối được đã kết hợp giao thức từ chối (theo giao thức này, Bob có thể chứng minh chữ kí là giả mạo). Như vậy, Bob sẽ có khả năng chứng minh trước toà rằng chữ kí bị lừa dối trên thực tế là giả mạo. (Nếu anh ta không chấp nhận tham vào giao thức từ chối, điều này được xem như bằng chứng chứng tỏ chữ kí trên thực tế là thật).

Như vậy, sơ đồ chữ kí không chối được gồm 3 thành phần: thuật toán ký, giao thức xác minh và giao thức từ chối (disavowal). Đầu tiên ta sẽ đưa ra thuật toán ký và giao thức xác minh của sơ đồ chữ kí không từ chối được của chaum - VanAntwerpen trên hình 6.7.

Xét vai trò của p và q trong sơ đồ này. Sơ đồ tồn tại trong Z_p; tuy vậy cần có khả năng tính toán theo nhóm nhân con G của Z_p^* có bậc nguyên tố. Củ thể, ta có khả năng tính được các phần tử nghịch đảo Modulo |G| - là lý do giải thích tại sao |G| phải là số nguyên tố. Để tiện lợi, lấy p=2q+1, q là số

nguyên tố. Theo cách này, nhóm con G lớn đến mức có thể là điều đáng mong muốn vì cả bức điện lẫn chữ kí đều là phần tử thuộc G.

Trước hết, cần chứng minh rằng, Alice sẽ chấp nhận một chữ kí hợp lệ. Trong các tính toán sau đây, tất cả các số mũ được rút gọn theo modulo q. Đầu tiên, nhận xét:

$$d \equiv c^{\alpha^{-1}} \pmod{p}$$

Hình 6.7. Sơ đồ chữ kí không chấp nhận chaum - Van Antwerpen.

Cho p =2q +1 là số nguyên tố sao cho q là nguyên tố và bài toán logarithm rời rạc trong Z_p là không thể giải được. Giả sử $\alpha \in Z_p$ là phần tử bậc q. Cho $1 \le a \le q-1$ và được định nghĩa $\beta = \alpha^a \bmod p$. Giả sử G biểu nhóm con bội Z_p^* bậc q (G là gồm các thặng dư bình thường modulo p). Cho $\mathcal{P} = \mathcal{A} = G$ và định nghĩa :

$\mathcal{K} = \{p, \alpha, a, \beta): \beta \equiv \alpha^a \pmod{p}\}$

Các giá trị p, α và β công khai, còn a mật.

Với k = (p, α, a, β) và x $\in$G , định nghĩa :

$$y = sig_k(x) = x^a \bmod p$$

Với x,y $\in$ G, việc xác minh được thực hiện qua giao thức sau:

1. Alice chọn e_1, e_2 ngẫu nhiên, $e_1, e_2 \in Z_p^*$
2. Alice tính $c = y^{e_1}\beta^{e_2} \bmod p$ và gửi cho no đến Bob
3. Bob tính $d = c^{a \bmod q} \bmod p$ và gữi nó cho Alice
4. Alice chấp nhận y là chữ kí hợp lệ khi và chỉ khi

$$d \equiv x^{e_1}\alpha^{e_2} \pmod{p}$$

Vì:

$$\beta \equiv \alpha^a \pmod{p}$$

Ta có:

????Chua viết

Tương tự

$$y = x^a \pmod{p}$$

có nghĩa là:

?????????? chưa viết

Vì thế $d \equiv x^{e_1}\alpha^{e_2} \pmod p$
như mong muốn.

Dưới đây là một ví dụ nhỏ.

Ví dụ 6.5

Giả sử lấy p = 467, vì 2 là phần tử nguyên thuỷ nên 2^2 =4 là phần tử sinh của G, các thặng dư bình phương modulo 467. Vì thế ta có thể lấy α =4. Giả thiết a = 101, khi đó

$$\beta = \alpha^a \bmod 467 = 499$$

Bob sẽ ký bức điện x= 119 với chữ ký

$$y = 119^{101} \bmod 467 = 129$$

Bây giờ giả sử Alice muốn xác minh chữ ký y, cô ta chọn các số ngẫu nhiên chẳng hạn e_1=38, e_2=397. Cô tính c=13, ngay lúc đó Bob sẽ trả lời với d =9,Alice kiểm tra câu trả lời bằng việc xác minh xem:

$$119^{38}\, 4^{397} = 9 \pmod{467}$$

vì thế Alice chấp nhận chữ ký là hợp lệ.

Tiếp theo, ta chứng minh rằng, Bob không thể lừa Alice chấp nhận chữ ký giả mạo (Fradulart) như là chữ ký hợp lệ trừ một xác suất rất bé. Kết quả này không phụ thuộc vào bất kỳ giả thiết tính toán nào, điều đó có nghĩa độ an toàn là vô điều kiện.

Định lý 6.1

Nếu $y \not\equiv x^a \pmod p$ thì Alice sẽ nhận y như la một chữ ký hợp lệ trên x với xác xuất 1/q.

Chứng minh

Trước hết, nhânj xét rằng mỗi yêu cầu (challenge) c có thể tương ứng chính xác với q cặp được sắp (e_1,e_2) (đó là vì cả y lẫn β đều là các phần tử của nhóm nhân G có bậc nguyên tố q). Bây giờ, khi Bob nhận được yêu cầu c, anh ta không có cách nào để biết về q cặp được sắp (e_1,e_2) có thể mà Alice đã

dùng để xây dựng c. Ta nói rằng, nếu $y \not\equiv x^a \pmod p$ thì đáp dụng ứng (respond) $d \in G$ mà Bob có thể là sẽ chỉ phủ hợp chính xác một trong q cặp được (e_1,e_2).

Vì α sinh ra G, nên ta có thể viết một phần tữ bất kỳ thuộc G như một số mũ của α, trong đó số mũ được xác minh duy nhất theo modulo q. Vì thế có thể viết $c = \alpha^i, d = \alpha^j$, $x = \alpha^k$ và $y = \alpha^{\ell}$ với $i, j, k, \ell \in Z_p$ và mọi phép tính số học là theo modulo q. Xét 2 đồng dư thức sau:

$$c \equiv y^{e_1}\beta^{e_2} \pmod p$$
$$d \equiv x^{e_1}\alpha^{e_2} \pmod p$$

Hệ thống này tương đương hệ đồng thức sau:

$$i \equiv \ell e_1 + a e_2 \pmod q$$
$$j \equiv k e_1 + e_2 \pmod q$$

Bầy giờ giả thiết rằng:

$$y \not\equiv x^a \pmod p$$

Comment [NVH1]:

nên rút ra : $\ell \not\equiv a k \pmod q$

Vì thế, ma trận hệ số của các đồng dư thức theo modulo q này có định thức khác 0 và như vây tồi tại nghiệm duy nhất cho hệ thống thống. Nghiã là, mỗi $d \in G$ là một đáp ứng với một trong q cặp (e_1,e_2) được sắp có thể. Hệ thống quả là, xác suất để Bob đưa cho Alice một đáp ứng(trả lời) d cần được xác minh đúng bằng 1/q. Định lý được chứng minh.

Hình 6.8. Thủ tục từ chối**.**

1. Alice chọn e_1,e_2 một cách ngẫu nhiên, $e_1,e_2 \in Z_q^*$
2. Alice tính $c = y^{e_1}\beta^{e_2} \bmod p$ và gửi nó cho Bob.
3. Bob tính d = ?
4. Alice xác minh xem d có $\not\equiv x^{e_1}\alpha^{e_2} \pmod p$ không
5. Alice chọn f_1,f_2 ngẫu nhiên , $f_1,f_2 \in Z_q^*$
6. Alice tính $C = y^{f_1}\beta^{f_2} \bmod p$ và gửi cho Bob
7. Bob tính D = ????????
8. Alice xác minh xem D có $\equiv x^{f_1}\alpha^{f_2} \pmod p$ không
9. Alice kết luận rằng y là giả mạo khi và chỉ khi

$$(d\,\alpha^{-e_2})^{f_1} \equiv (D\,\alpha^{-f_2})^{e_1} \pmod p$$

Bây giờ quay trở lại giao thức từ chối. Giao thức này gồm hai 2 thực hiện giao thức xác minh và được nêu trong hình 6.8.

Các bước 1- 4 và 5- 8 gồm 2 lần thực hiện không thành công giao thức xác minh. Bước 9 bước "tính kiểm tra phù hợp" cho Alice xác định xem liệu có phải đang lập các câu trả lời của anh ta theo thứ tự chỉ ra hay không.

Dưới đây là ví dụ minh hoạ.

Ví dụ 6.6

Như trước đây, giả sử p = 467, α = 4, a = 101, và β = 449. Giả thiết bức điện x = 286 được ký y = 83 và Bob muốn thiết phục Alice rằng chữ ký không hợp lệ.

Giả sử Alice bắt đầu bằng việc chọn các giá trị ngẫu nhiên e_1=45, e_2=237. Alice tính c =305 và Bob trả lời d = 109. Sau đó Alice tính

$$286^{125}4^{237} \bmod 467 = 149$$

Vì 149 # 109 nên Alice thực hiện bước 5 của giao thức.

Bây giờ giả sử Alice chọn giá trị ngẫu nhiên f_1= 125, f_2= 9. Alice tính C = 270 và Bob trả lời với D =68. Alice tính

$$186^{125}4^{9} \bmod 467 = 25$$

Vì 25 # 68 nên Alice tiếp tục sang bước 9 của giao thức kiểm tra tính phù hợp. Bước kiểm tra này thành công vì:

$$(109 \times 4^{-9})^{125} \equiv 188 \pmod{467}$$

và

$$(68 \times 4^{-9})^{45} \equiv 188 \pmod{467}$$

Vì thế Alice tin rằng chữ ký không hợp lệ. □

Bây giờ, ta phải chứng minh hai vấn đề:

1. Bob có thể thuyết phục Alice rằng, chữ ký không hợp lệ là giả mạo.
2. Bob không thể là Alice tin rằng chữ ký không hợp lệ là giả mạo trừ một xác suất rất bé.

Định lý 6.2

Nếu $y \not\equiv x^a \pmod p$ và cả Alice lẫn Bob thực hiện theo giao thức từ chối thì

$$(d\,\alpha^{-e_2})^{f_1} \equiv (D\,\alpha^{-f_2})^{e_1} \pmod p$$

Chứng minh:

Dùng các yếu tố

$$d \equiv ???$$

$$c \equiv y^{e_1}\beta^{e_2} (\text{mod } p)$$

và

$$\beta \equiv \alpha^a \text{ (mod } p)$$

Ta có:

$$(d\ \alpha^{-e_2})^{f_1} \equiv ????$$

Tượng tự, dùng các yếu tố $D \equiv ????$

$$(D\ \alpha^{-f_2})^{e_1} \equiv y^{e_1\ f_2} \text{ (mod } p)$$

vì thế phép kiểm tra tính phù hợp trong bước 9 thành công. □

Bây giờ xét xác suất để Bob có thể thử từ chối một chữ ký hợp lệ. Trường hợp này không giả thiết Bob thực hiên theo thủ tục. Nghĩa là Bob có thể không xây dưng D Và d như trong giao thức. Vì thế trong định lý tiếp theo chỉ là giả thiết rằng, Bob có thể tạo ra các D và d thoả mãn điều kiện trong các bước 4,8và 9 của giao thức nêu trên hình 6.8.

Định lý 6.3

Giả sử $y \equiv x^a \text{ (mod } p)$ và Alice thực hiện theo giao thức từ chối. Nếu

$$d \not\equiv x^{e_1}\ \alpha^{e_2} (\text{mod } p)$$

và

$$D \equiv x^{f_1}\ \alpha^{f_2} (\text{mod } p)$$

thì xác suất để:

$$(d\alpha^{-e_2})^{f_1} \not\equiv (D\alpha^{-f_2})^{e_1} \text{ (mod } p) = 1\text{-}1/q$$

chứng minh: giả sử rằng, các đồng dư thức sau được thoả mãn

$$y \not\equiv \alpha^a \text{ (mod } p)$$

$$d \not\equiv x^{e_1}\ \alpha^{e_2} (\text{mod } p)$$

$$D \not\equiv x^{f_1}\ \alpha^{f_2} (\text{mod } p)$$

$$(d\alpha^{-e_2})^{f_1} \equiv (D\alpha^{-f_2})^{e_1} \text{ (mod } p)$$

ta sẽ nhận được mâu thuẩn như trình bay sau đây:

có thể viết lại bước 9- bước kiểm tra tính phù hợp như sau

$$D \equiv d_0^{f_1}\ \alpha^{f_2}$$

trong đó

$$d_0 = d^{1/e_1}\alpha^{-e_2/e_1} \text{mod } p$$

là giá trị chỉ phụ thuộc vào các bước 1- 4 trong giao thức.

Áp dụng định lý 6.1, ta kết luận được y là chữ ký hợp lệ đối với d_0 với xác suất 1- 1/q. Song ta đã giả thiết y là chữ ký hợp lệ đối với x, nghĩa là là ta có (với xác suất cao)

$$x^a \equiv d_0^{\ a} \text{ (mod } p)$$

có nghĩa là $x = d_0$

Tuy nhiên do

$$d \not\equiv x^{e_1} \alpha^{e_2} (\text{mod } p)$$

có nghĩa là

$$x \not\equiv d^{1/e_1} \alpha^{-e_2/e_1} (\text{mod } p)$$

Và từ chỗ

$$d_0 \equiv d^{1/e_1} \alpha^{-e_2/e_1} (\text{mod } p)$$

suy ra $x \# d_0 \Rightarrow$ ta nhận được mâu thuẩn.

Như vậy Bob có thể lừa dối Alice theo cách này với xác suất 1/q.

6.6 CÁC CHỮ KÝ FAIL- STOP

Sơ đồ chữ ký Fail- stop dùng để tăng độ mật trước khả năng một đối thủ mạnh có thể giả mạo chữ ký. Nếu Oscar khả năng giả mạo chữ ký của Bob thì Bob có khả năng chứng minh được (với xác suất cao) rằng chữ ký của Oscar là giả mạo.

Phần này sẽ mô tả một sơ đồ Fail- stop do Van Heyst va Pedersen đua ra năm 1992. Đầu là sơ đồ chữ ký 1 lần (chỉ một bức điện có thể ký bằng một cho trước chỉ 1 lần). Hệ thống gồm các thuật toán ký, thuật toán xác minh và thuật toán "chứng minh giả mạo". Hình 6.9 mô tả các thuật toán ký và xác minh của sơ đồ Fail- stop của Van Heyst va Pedersen.

Không khó khăn nhận thấy rằng, chữ ký do Bob tao ra sẽ thoả mãn điều kiện xác minh nên ta lại trở các kía cạnh an toàn toàn của sơ đồ này và các thức làm việc của tính chất Fail- Safe (tự động ngừng khi có sai số). Trước hết, ta thiết lập vài yếu tố quan trọng có liên quan đến các khoá của sơ đồ. Đầu tiên đưa ra một định nghĩa: Hai khoá (γ_1, γ_2, a_1, a_2, b_1, b_2) và (γ_1', γ_2', a_1', a_2', b_1', b_2') là tương đương nếu $\gamma_1 = \gamma_1$', $\gamma_2 = \gamma_2$'. Và dễ dàng nhận thấy tồi tại q^2 khoá trong lớp tương đương bất kỳ.

Sau đây là vài bổ đề.

Bổ đề 6.4

Giả sử K và K'là các khoá tương đương và giả thiết chữ ký $ver_K(x,y)$ = true(đúng). Khi đó chữ ký $ver_{K'}(x,y)$ = true.

Chứng minh

Giả sử $K =(\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ và $K'= (\gamma_1', \gamma_2', a_1', a_2', b_1', b_2')$ trong đó :

$$\gamma_1= \alpha^{a_1}\beta^{a_2} \bmod p =\alpha^{a'_1}\beta^{a'_2} \bmod p$$
$$\gamma_2= \alpha^{b_1}\beta^{b_2} \bmod p =\alpha^{b'_1}\beta^{b'_2} \bmod p$$

Giả sử x được bằng cách dùng K và tạo ra các chữ ký $y =(y_1, y_2)$ trong đó:

$$y_1= a_1+xb_1 \bmod q$$
$$y_2= a_2+xb_2 \bmod q$$

Hình 6.9 Sơ đồ chữ ký Fail- stop.

Cho p = 2q+1 là số nguyên tố sao q là nguyên tố và bài toán logarithm rời rạc trong Z_p là khó giải. cho $\alpha \in Z_p^*$ là phần tử bậc q. Giả sử $1 \le a_0 \le q-1$ và định nghĩa $\beta =\alpha^{a_0} \bmod p$. Các giá trị p, q, α, β và a_0 đều do người có thẩm quyền (được tin cậy) chọn. Các số p, q, α và β công khai và cố định còn a_0 được giữ bí mật.

Cho $\mathcal{P} =Z_p$ và $\mathcal{A} = Z_q \times Z_q$. khoá có dạng:

$$K =(\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$$

trong đó $a_1, a_2, b_1, b_2 \in Z_q$

$$\gamma_1= \alpha^{a_1}\beta^{a_2} \bmod p$$

còn $$\gamma_2= \alpha^{b_1}\beta^{b_2} \bmod p$$

Với $K =(\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$ và $x \in Z_p^*$, ta định nghĩa

$$sig_k(x) =(y_1, y_2)$$

trong đó

$$y_1= a_1+xb_1 \bmod q$$

còn $$y_2= a_2+xb_2 \bmod q$$

Với $y =(y_1, y_2) \in Z_q \times Z_q$ ta có:

Xác minh ver(x,y) = true $\Leftrightarrow \gamma_1\gamma_2^x \equiv \alpha^{y_1}\beta^{y_2} \pmod p$

Bây giờ giả sử ta xác minh y bằng cách dùng K'

$$\alpha^{y_1}\beta^{y_2} \equiv \alpha^{a'_1 + xb'_1}\beta^{a' + xb'_2} \pmod p$$
$$\equiv \alpha^{a'_1}\beta^{a'_2}(\alpha^{b'_1}\beta^{b'_2})^x \pmod p$$
$$\equiv \gamma_1\gamma_2^x \bmod p$$

Như vậy, y cũng sẽ được xác minh bằng K'.

Bổ đề 6.5

Giả sử K là khoá còn $y = sig_{K'}(x)$. Khi đó tồn tại đúng q khoá K' tương đương với K sao cho $y= sig_{K'}(x)$.

Chứng minh

Giả sử $\gamma_1$và $\gamma_2$là các thành phần công khai của K. Ta muốn xác định số bội 4 (a_1, a_2, b_1, b_2)sao cho các đồng dư thức sau đây được thoả mãn.

$$\gamma_1 \equiv \alpha^{a_1}\beta^{a_2} \pmod p$$
$$\gamma_2 \equiv \alpha^{b_1}\beta^{b_2} \pmod p$$
$$y_1 \equiv a_1 + xb_1 \pmod q$$
$$y_2 \equiv a_2 + xb_2 \pmod q.$$

Vì α sinh ra G nên tồn tại các số mũ duy nhất $c_1, c_2, a_0 \in Z_q$ sao cho

$$\gamma_1 \equiv \alpha^{c_1} \pmod p$$
$$\gamma_2 \equiv \alpha^{c_2} \pmod p$$

và $$\beta \equiv \alpha^{a_0} \pmod p$$

vì thế nó điều kiện cần và đủ để hệ các đồng dư thức sau đây được thoả mãn:

$$c_1 \equiv a_1 + a_0a_2 \pmod q$$
$$c_2 \equiv b_1 + a_0b_2 \pmod q$$
$$y_1 \equiv a_1 + xb_1 \pmod q$$
$$y_2 \equiv a_2 + xb_2 \pmod q$$

Hệ thống này có thể viết dưới dạng phương trình ma trận trong Z_q như sau:

X

$$\begin{pmatrix} 1 & a_0 & 0 & 0 \\ 0 & 0 & 1 & a_0 \\ 1 & 0 & x & 0 \\ 0 & 1 & 0 & x \end{pmatrix} \begin{pmatrix} a_1 \\ a_2 \\ b_1 \\ b_2 \end{pmatrix} = \begin{pmatrix} c_1 \\ c_2 \\ y_1 \\ y_2 \end{pmatrix}$$

có thể thấy ma trận hệ thống số của phương trình có hạng là 3(hạng của một ma trận là số cực đại của các hàng độc lập tuyến tính mà nó có). Rõ ràng, hạng ít nhất bằng 3 vì các hàng 1, 2 và 4 là độc lập tuyến tính trên Z_p. còn hạng nhiều nhất cũng bằng 3 vì:

$$r_1 + x\, r_2 - r_3 - a_0 r_4 = (0,0,0,0).$$

Với r_1 chỉ hàng thứ i của ma trận.

Hệ phương trình này có ít nhâts một nghiệm nhận được bằng cách dùng khoá K.Vì hàng của ma trận hệ số bằng 3 nên suy ra rằng chiều của không gian nghiệm là 4-3=1 và có chính xác q nghiệm.

Tương tự như vậy ta có thể chứng minh được kết qủa sau:

Bổ đề 6.6

Giả sử K là khoá $y=sig_K(x)$ còn $ver_K(x',y')$=true, trong đó $x' \# x$. Khi đó tồn tại ít nhất một khoá K' tương đương với K sao cho $y=sig_{K'}(x)$ và $y'= sig_{K'}(x')$

Ta hãy làm sáng tỏ hai bổ đề trên về độ mật của sơ đồ. Khi cho trước y là chữ kí hợp lệ của x, sẽ tồn tại q khoá có thể để x sẽ được kí bằng y. Song với bức điện bất kì $x' \neq x$, q khoá này sẽ tạo ra q khoá khác nhau trên x'. Điều đó dẫn đến định lí sau đây:

Định lí 6.7:

Nếu cho trước $sig_K(x)=y$ và $x' \neq x$. Oscar có thể tính $sig_K(x')$ với xác suất là 1/q.

Chú ý rằng, định lí này không phụ thuộc vào khả năng tính toán của Oscar: Mức an toàn qui định đạt được vì Oscar không thể nói về q khoá có thể mà Bob đang dùng. Như vậy độ an toàn ở đây là vô điều kiện.

Tiếp tục xem xét về khái niệm Fail- Stop. Khi cho trước chữ ký y trên bức điện x. Oscar không thể tính ra được chữ ký y' của Bob trên bức điện x' khác. Điều này cũng có thể hiểu rằng, Oscar có thể tính được chữ ký giả mạo

$y'' = sig_K(x')$(sẽ được chứng minh). Tuy nhiên, nếu đưa cho Bob một chữ ký giả mạo hợp lệ, thì anh ta có thể tạo ra "một bằng chứng về sự giả mạo " với xác suất 1-1/q. Bằng chứng về sự giả mạo là giá trị $a_0=\log_\alpha \beta$ (chỉ người có thẩm quyền trung tâm biết).

Giả sử Bob sở hữu cặp (x',y'') sao cho $ver(x',y'')$= true và $y''\neq sig_K(x')$.

Nghĩa là:

$$\gamma_1\gamma_2^{x'} \equiv \alpha^{y''_1}\beta^{y''_2}(\text{mod } p)$$

trong đó :$y''=(y''_1,y''_2)$. Bây giờ Bob có thể tính chữ ký của mình trên x' là $y' = (y'_1,y'_2)$. Khi đó :

$$\gamma_1\gamma_2^{x'} \equiv \alpha^{y'_1}\beta^{y'_2}(\text{mod } p)$$

vì thế $$\alpha^{y''_1}\beta^{y''_2} \equiv \alpha^{y'_1}\beta^{y'_2}(\text{mod } p)$$

Nếu viết $\beta = \alpha^{a_0} \text{mod } p$, ta có :

$$\alpha^{y''_1+ a_0 y''_2} \equiv \alpha^{y'_1+ a_0 y'_2}(\text{mod } p)$$

hay:

$$y''+a_0y''_2 \equiv y'_1+a_0y'(\text{mod } q)$$

hoặc:

$$y''_1- y_1' \equiv a_0(y'_2-y''_2)(\text{mod } q)$$

Xét thấy $y'_1 \not\equiv y''_2(\text{mod } q)$ vì y' là giả mạo. Vì thế $(y'_2-y''_2)^{-1} \text{mod } q$ tồn tại và

$$a_0 =\log_\alpha \beta = (y''_1-y'_1)\ (y'_2-y''_2)^{-1} \text{ mod } q$$

Dĩ nhiên, bằng việc chấp nhận bằng chứng về sự giả mạo như vậy,ta giả thiết Bob không thể tự tính được logarithm rời rạc $\log_\alpha \beta$. Đây là gải thiết về mặt tính toán.

Cuối cùng, chú ý rằng,sơ đồ chữ kí là một lần vì khóa k của Bob có thể tính dễ dàng nếu hai bức điện đều dùng K để ký. Dưới đây là ví dụ minh hoạ cách Bob tạo một bằng chứng về sự giả mạo.

Vi dụ 6.7

Cho p=4367=2.1733+1. Phần tử α =4 có bậc là 1733 trong Z_{3467}^*
Giả sử a_o =1567, ta có:

$$\beta = 4^{1567} \bmod 346=514$$

(Bob biết α và β song không biết a_0). Giả sử Bob tập khoá bằng cách dùng a_1 = 888, a_2 = 1042, b_1 = 786, b_2 = 999. Khi đó

$$\gamma_1=4^{888}514^{1024} \bmod 3476=3405$$

và $$\gamma_2=4^{786}514^{999} \bmod 3476=2281$$

Tiếp theo, giả sử Bob nhận được chữ kí giả mạo (822,55) trênê bức điện 3383. Đây là chữ ký hợp lệ vì thoả mãn điều kiện xác minh.

$$3405 \times 2281^{3384} \equiv 2282 \pmod{3476}$$

và $$4^{822}514^{55} \equiv 2282 \pmod{3476}$$

Mặt khác đây không phải là chữ kí đã được Bob xây dựng. Bob có thể tíng chữ kí của mình như sau:

(888+3383×786 mod 1733.1024+3383×999 mod 1733)=(1504.1291)

Sau đó anh ta tính tiếp log rời rạc bí mật

$$a_0 =(822-1504)(1291-55)^{-1} \bmod 1733 =1567.$$

Đây là bằng chứng về sự giả mạo.

6.7 CÁC CHÚ GIẢI VỀ TÀI LIỆU DẪN

Mitchell, Piper và Wild [MPW 92] đã đưa ra một tổng quan đầy đủ về các sơ đồ chữ kí. Bài này cũng có hai phương pháp giả mạo chữ kí của Elgamal mà ta đã đưa ra trong 6.2.

Sơ đồ chữ kí Elgamal đã được nêu trong [EL 85], tiêu chuẩn chữ kí số được công bố đầu tiên vào 8/1991 bởi NIST và được chấp nhận làm tiêu chuẩn vào 12/94 [NBS 94]. Một cuộc thoả luận dài về DSS và những cuộc tranh cãi xung quanh nó vào 7/1992 được đăng trên Communication of the ACM.

Sơ đồ Lamport được mô tả trong bài báo của Diffie_Hellman [DH 76] năm 1976. Bản cải tiến củaBob và Chaum được nêu trong [BC 93]. Sơ đồ chữ

kí không chối nêu trong mục 6.5 do Chaum và Van Antwerpen đưa ra trong [CVA 90]. Sơ đồ chữ kí Fail-Stop trong mục 6.6 là của Van Heyst và Pederson [VHP 93].

Một số ví dụ về các sơ đồ chữ kí "phá được "gồm các sơ đồ của ông Schorss- Sshamir[OSS 85](cũng bị phá bởi Estes EAKMM 86) và sơ đồ hoán vị Birational của Shamir [SH94] (bị Coppessnuth, Steru và Vandeney CSV 94). Cuối cùng ESIGN là sơ đồ chữ kí của Fujioka.Okamoto và Meyaguchi [FOM 91]. Một số phiên bản của sơ đồ này đã bị phá. Song một sửa đổi trong [FOM 91] lại không bị phá.

BÀI TẬP

6.1. Giả thiết Bob đang dùng sơ dồ Elgamal, anh ta kí hai bức điện x_1 và x_2 bằng chữ kí (γ, δ_1) và (γ, δ_2) tương ứng (giá trị này của γ giống nhau trong cả hai chữ kí). Cũng giả sử UCLN $(\gamma_1-\gamma_2, p-1)=1$.

a) Hãy cho biết cách tính k hiệu quả khi biết thông tin này
b) Hãy mô tả cách sơ đồ chữ kí có thể bị phá.
c) Giả sử p=31847, α =5, và β =25703. Tính k và a khi cho trước chữ kí (2397 2,31396) với bức điện x=8990 và chữ kí (23972, 20481 trên bứ c điện x=31415)

6.2. Giả sử I thực hiên sơ đò Elgamal với p=31847, α =5,và β =26379. Hãy viết phương trìng thực hiện công việc sau:
a) Xác minh chữ kí (20679,11082) trên bức điện x=20543
b) Xác định số mũ mật a bằng cách dùng thuật toán tối ưu hoá thời gian - bộ nhớ của Shark, sau dó xác định giá trị k ngẫu nhiên dùng trong việc kí lên bức điện x.

6.3. Giả sử Bob dùng sơ đố chữ kí Elgamal như trong ví dụ 6.1:p=467,α =2 β=132.Giả sử Bob kí lên bức điện x=100 bằng chữ kí (29,51).Hãy tính chữ kí giả mạo mà Oscar có thể lập bằng cách dùng h=100,i=45 và j =293.Hãy kiểm tra xem chữ ký vừa nhận được có thoả mãn điều kiện xác minh không.

6.4. Chứng minh rằng phương pháp giả mạo thứ hai trên sơ đồ Elgamal (mô tả trong mục 6.2) cũng tạo ra chữ kí thoả mãn điều kiện xác minh.

6.5. Sau đây là phương án của sơ đồ Elgamal :Khoá được xây dựng tương tự theo mghĩa như trước đây:Bob chọn $\alpha \in Z_p^*$ là phần tử nguyên thuỷ. a là số mũ mật ($0 \le a \le p-2$) sao cho UCLN (a,p-1) =1 và $\alpha^a \bmod p$. Khoá K = (α, a, β), ở đây α và β công khai còn a mật. Cho $x \in Z_p$ là bức điện được kí. Bob tính chữ kí sig (x)=(γ, δ), trong đó:

$$\gamma = \alpha^k \bmod p$$

còn $\delta = (x - k\gamma)^{a-1} \bmod (p-1)$.

Sự khác nhau duy nhất so với sơ đồ Elgamal ban đầu là ở cách tính δ. Hãy trả lời các câu hỏi sau liên quan đến sơ đồ cải tiến này:

a) Mô tả cách xác minh một chữ kí (γ, δ) trên bức điện x bằng cách dùng công khai khoá của Bob.
b) Mô tả ưu điểm về mặt tính toán của sơ đồ cải tiến.
c) So sánh tóm tắt độ an toàn của sơ đồ cải tiến và sơ đồ ban đầu.

6.6. Giả sử Bob dùng DSS với q = 101, p = 7879, α = 170, a = 75 còn β = 4567 như trong ví dụ 6.3. Xác định chữ kí của Bob trên bức điện x=5011, bằng cách dùng giá trị ngẫu nhiên k =49 và chỉ ra cách xác minh chữ kí nhận được.

6.7. Trong sơ đồ Lamport,giả sử rằng hai bức điện x và x' bội k (k-tuple) đều do Bob kí. Cho $\ell = d(x,x')$ là toạ độ trên đó x và x' khác nhau. Hãy chỉ ra cách Oscar có thể kí $2^\ell - 2$ bức điện mới.

6.8. Trong sơ đồ Bob-Chaum với k = 6, n = 4, giả sử rằng các bức điện x = (0, 1,0,0,1,1) và x' = (1,1,0,1,1) đều được kí. Xác định bức điện mới được Oscar kí khi biết chữ kí trên x và x'.

6.9. Trong sơ đồ Bob- Chaum, giả sử rằng hia bức điện x và x' là các bội k đều do Bob kí Cho $l = |\phi(x) \cup \phi(x')|$. Hãy chỉ ra cách Oscar có thể kí $\binom{\ell}{n} - 2$ bức điện mới.

6.10. Giả sử Bob đang dùng chữ kí không chối được của Chaum –Van Antwerpen như trong ví dụ 6.5. Nghĩa là p = 467, α = 4, a = 101, β = 449. Giả sử Bob được trình chữ kí y = 25 trên bức điện x =157 và anh ta muốn chứng minh rằng nó giả mạo. Giả sử số ngẫu nhiên của Alice là e_1 = 46, e_2 = 123, f_1 =198, f_2 =11 trong thủ tục từ chối. Hãy tính các yêu cầu c, d, của Alice và các câu trả lời C, D của Bob; chỉ ra rằng phép kiểm tra tính phù hợp của Alice sẽ thành công.

6.11. Chứng minh rằng, mỗi lớp tương đương các khoá trong sơ đồ chữ kí Fail-Stop của Pedersen-Van Hőyt chứa q^2 khoá.

6.12. Giả sử Bob đang dùng sơ đồ chữ kí Fail-Stop của Pedersen-Van Heyst với p = 3467, α =4, a_0=1567 và β =514 (dĩ nhiên Bob không biết giá trị a_0).

a) Dùng yếu tố a_0 =1567, xác định tất cả các khoá có thể :

$K = (\gamma_1, \gamma_2, a_1, a_2, b_1, b_2)$

sao cho $sig_K(42)$ =(1118,1449)

b) Gái sử $sig_K(42)$ =(1118,1449) và $sig_K(969)$ =(899,471). Không cần dùng điều kiện a_0 =1567. Hãy xác định K (điều này sẽ chứng tỏ sơ đồ là dùng một lần).

6.13. Gải sử Bob dùng sơ đồ Fail-Stop của Pedersen-Van Heyst vơi p =5087, α =25, β =1866. Giả sử K =(5065, 5067,144,874,1873,2345) và Bob tìm chữ kí (2219,458) được giả mạo trên bức điện 4785

a) Chứng minh rằng, chữ kí giả mạo này thoả mãn điều kiện xác minh nên nó là chữ kí hợp lệ.

b) Chỉ ra cách Bob tính " bằng chứng giả mạo a_0 khi cho trước chữ kí giả mạo này. "

CHƯƠNG 7

CÁC HÀM HASH

7.1 CÁC CHŨ KÍ VÀ HÀM HASH.

Bạn đọc có thể thấy rằng các sơ dồ chữ kí trong chương 6 chỉ cho phép kí các bức điện nhỏ.Ví dụ, khi dùng DSS, bức điện 160 bit sẽ được kí bằng chữ kí dài 320 bít. Trên thực tế ta cần các bức điện dài hơn nhiều. Chẳng hạn, một tài liệu về pháp luật có thể dài nhiều Megabyte.

Một cách đơn giản để gải bài toán này là chặt các bức điện dài thành nhiều đoạn 160 bit, sau đó kí lên các đoạn đó độc lập nhau. Điều này cũng

tương tự như mã một chuoĩ dài bản rõ bằng cách mã của mỗi kí tự bản rõ độc lập nhau bằng cùng một bản khoá. (Ví dụ: chế độ ECB trong DES).

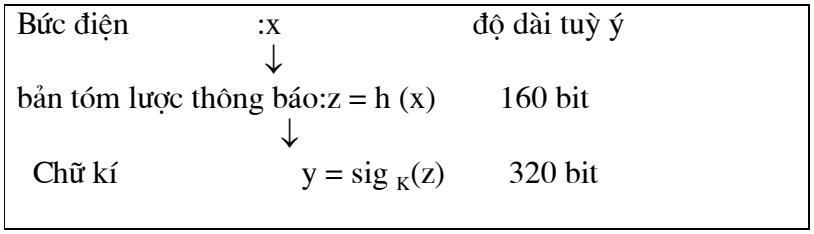
Biện pháp này có một số vấ đề trong việc tạo ra các chữ kí số. Trước hết, với một bức điện dài, ta kết thúc bằng một chữ kí rất lớn (dài gấp đôi bức điện gốc trong trường hợp DSS). Nhược điểm khác là các sơ đồ chữ kí "an toàn" lại chậm vì chúng dùng các pháp số học phức tạp như số mũ modulo. Tuy nhiên, vấn đề nghiêm trọng hơn với phép toán này là búc điện đã kí có thể bị sắp xếp lại các đoạn khác nhau,hoặc một số đoạn trong chúng có thể bị loại bỏ và bức điện nhận được vẫn phải xác minh được. Ta cần bảo vệ sự nguyên vẹn của toàn bộ bức điện và điều này không thể thực hiện được bằng cách kí độc lập từng mẩu nhỏ của chúng.

Giải pháp cho tất cả các vấn đề này là dùng hàm Hash mã khoá công khai nhanh. Hàm này lấy một bức điện có độ dài tuỳ ý và tạo ra một bản tóm lược thông báo có kích thước qui định (160 bit nếu dùng DSS).

Sau đó bản tóm lược thông báo sẽ được kí. Vơi DSS, việc dùng hàm Hash được biểu diễn trê hình 7.1.

Khi Bob muốn kí bức điện x, trước tiên anh ta xây dựng một bnr tóm lược thông báo $z = h(x)$ và sau đó tính $y = sig_K(z)$. Bob truyền cặp (x, y) trên kênh. Xét thấy có thể thực hiện xác minh (bởi ai đó) bằng cách trước hết khôi phục bản tóm lược thông báo $z = h(x)$ bằng hàm h công khai và sau đó kiểm tra xem $ver_k(x,y)$ có = true, hay không.

Hình 7.1.Kí một bản tóm lược thông báo



7.2. HÀM HASH KHÔNG VA CHẠM

Chúng ta cần chú ý rằng,việc dùng hàm hash h không làm giảm sự an toàn của sơ đồ chữ kí vì nó là bản tóm lược thông báo được chữ kí không phải là bức điện. Điều cần thiết đối với h là cần thoả mãn một số tính chất nào đó để tranh sự giả mạo.

Kiểu tấn công thông thường nhất là Oscar bắt đầu bằng một bức diện được kí hợp lệ (x, y), $y = sig_K(h(x))$,(Cặp (x, y) là bức điện bất kì được Bob kí trước đó). Sau đó anh ta tính $z = h(x)$ và thử tìm $x \neq x'$ sao cho $h(x') = h(x)$. Nếu Oscar làm được như vậy, (x', y) sẽ là bức điện kí hợp lệ, tức một bức điện giả mạo. Để tránh kiểu tấn công này, h cần thoả mãn tính không va chạm như sau:

Định nghĩa 7.1

Hàm hash h là hàm không va chạm yếu nếu khi cho trước một bức điện x, không thể tiến hành về mặt tính toán để tìm một bức điện $x \neq x'$ sao cho $h(x') = h(x)$.

Một tấn công kiểu khác như sau: Trước hết Oscar tìm hai bức điện $x \neq x'$ sao cho $h(x) = h(x')$. Sau đó Oscar đưa x cho Bob và thyết phục Bob kí bản tóm lược thông báo h(x) để nhận được y. Khi đố (x',y) là thông báo (bức điện) giả mạo hợp lệ.

Đây là lí do đưa ra một tính chất không va chạm khác.

Định nghĩa 7.2.

Hàm Hash h là không va chạm mạnh nếu không có khả năng tính toán để tìm ra bức điênk x và x' sao cho $x \neq x'$ và $h(x) = h(x')$.

Nhận xét rằng: không va chạm mạnh bao hàm va chạm yếu.

Còn đây là kiểu tấn công thứ 3: Như đã nói ở phần 6.2 việc giả mạo các chữ kí trên bản tóm lược thông báo z ngẫu nhiên thường xảy ra với sơ đồ chữ kí. Giả sử Oscar tính chữ kí trên bản tóm lược thông báo z ngẫu nhiên như vậy. Sau đó anh ta tìm x sao cho z= h(x). Nếu làm được như vậy thì (x,y) là bức điện giả mạo hợp lệ. Để tránh được tấn công này, h cần thoả mãn tính chất một chiều (như trong hệ mã khoá công khai và sơ đồ Lamport).

Định nghĩa 7.3.

Hàm Hash h là một chiều nếu khi cho trước một bản tóm lược thông báo z, không thể thực hiện về mặt tính toán để tìm bức điện x sao cho h(x) = z.

Bây giờ ta sẽ chứng minh rằng, tính chất không va chạm mạnh bao hàm tính một chiều bằng phản chứng. Đặc biệt ta sẽ chứng minh rằng, có thể dùng thuật toán đảo với hàm Hash như một chương trình con (giả định) trong thuật toán xác suất Las Vegas để tìm các va chạm.

Sự rút gọn này có thể thực hiện với một giả thiết yếu về kích thước tương đối của vùng và miền (domain and range) của hàm Hash. Ta cũng sẽ giả thiết tiếp là hàm Hash h: $X \to Z$, X,Z là các tập hữu hạn và $|X| \geq 2|Z|$. Đây là giả thiết hợp lí :Nếu xem một phần tử của X được mã như một xâu bít có độ dài $\log_2|X|$ và phần tử của Z được mã hoá như một xâu bít có độ dài $\log_2|X|$ thì bản tóm lược thông báo z = h(x) ít nhất cũng ngắn hơn bức điện x một bít (ta sẽ quan tâm đến tình huống vùng X là vô hạn vì khi đó có thể xem xét các bức điện dài tuỳ ý. Lập luận đó của ta cũng áp dụng cho tình huống này).

Tiếp tục giả thiết là ta có một thuật toán đảo đối với h, nghĩa là có một thuật toán A chấp nhận như đầu vào bản tóm lược thông báo $z \in Z$ và tìm một phần tử $A(z) \in X$ sao cho h(A(z)) = z.

Ta sẽ chứng minh định lí dưới đây:

Định lí 7.1:

Giả sử h: $X \to Z$ là hàm Hash, trong đó $|X|$ và $|Z|$ hữu hạn và $|X| \geq 2|Z|$. Cho A là thuật toán đảo đối với h. Khi đó tồn tại một thuật toán Las Vagas xác suất tìm được một va chạm đối với h với xác suất ít nhất là1/2.

Chứng minh :

Xét thuật toán B đưa ra trong hình 7.2. Rõ ràng B là một thuật toán xác suất kiểu Las Vegas vì nó hoạc tìm thấy một va chạm, hoặc cho câu trả lời không. Vấn đề còn lại là ta phải tịnh xac suất thành công, Với x bất kỳ thuộc X, định nghĩa $x \sim x_1$ nếu $h(x) = h(x_1)$. Dễ thấy rằng, ~ là quan hệ tương đương. Ta định nghĩa:

$$[x] = \{x_1 \in X: x \sim x_1\}$$

Mỗi lớp tương đương [x] chứa ảnh đảo của một phần tử thuộc Z nên số các lớp tương đương nhiều nhất là $|Z|$. Kí hiệu tập các lớp tương đương là C.

Bây giờ giả sử, x là phần tử $\in X$ được chọn trong bước 1. Với giá trị x này, sẽ có $|[x]|$ giá trị x_1 có thể cho phép trở lại bước 3. $|[x]|$-1 các giá trị x_1 này khác với x và như vậy bước 4 thành công. (Chú ý rằng thuật thoán A

không biết biểu diễn các lớp tương đương [x] đã chon trong bước 1). Như vậy, khi cho trước lựa chọn cụ thể $x \in X$, xác suất thành công là
$(|[x)|-1/|[x]|$.

Hình.7.2 Dùng thuật toán đảo A để tìm các va chạm cho hàm Hash

```
1.chọn một ssó ngẫu nhiên x ∈X
2.Tính z=h(x)
3.Tinh x1= A(Z)
4. if x1 ≠ x then
        x và x1 va chạm dưới h (thành công)
   else
        Quit (sai)
```

Xác suất thành công của thuật toán B bằng trung bình cộng tất cả các lựa chon x có thể:

$$P(\text{thành công}) = (1/|X|)\sum_{x\in X}(|[x]|-1)/|[x]|$$
$$= (1/|X|)\sum_{c\in C}\sum_{x\in C}(|c|-1)/|c|$$
$$= 1/|X|\sum_{c\in C}(|c|-1) = (1/|X|)\sum_{c\in C}|c| - \sum_{c\in C}1$$
$$>= (|X| - |Z|) / |X|$$
$$>= ((|X| - |Z|)/2) / |X| = ẵ$$

Như vậy, ta đã xây dựng thuật toán Las Vegas có xác suất thành công ít nhất bằng 1/2.

Vì thế, đó là điều kiện đủ để hàm Hash thoả mãn tính chất không va chạm mạnh vì nó bao hàm hai tính chất khác.Phần còn lại của chương này ta chỉ quan tâm đến các hàm Hash không va chạm mạnh.

7.3 TẤN CÔNG NGÀY SINH NHẬT(birthday)

Trong phần này, ta sẽ xác định điều kiện an toàn cần thít ch hàm Hash và điều kiện này chỉ phụ thuộc vào lực lượng của tập Z (tương đương về kích thước của bảng thông báo).Điều kiện cần thiết nà rút ra tư phương pháp tìm kiếm đơn giản ác va chạm mà người ta đã biết đến dưới cái tên tấn công ngày sinh nhật (birthday phương pháparradox), trong bài toán:một nhóm 23 người ngẫu nhiên, có ít nhất 2 người có ngày sinh trùng nhau với xác suất ít nhất là1/2.(Dĩ nhiên, đây chưa phải là nghịch lí,song đó là trực giác đối lập có thể

xảy ra). Còn lí do của thuật ngữ "tấn công ngày sinh nhật " sẽ rõ ràng khi ta tiếp tuch trình bày.

Như trước đây, ta hãy giả sử rằng :h:X→Z là hàm Hash, X,Z hữu hạn và $|X| >= 2|Z|$.Định nghĩa $|X| = m$ và $|Z| = n$.Không khó khăn nhận thấy rằng, có ít nhất n va chạm và vấn đề đặt ra là cách tìm chúng. Biện pháp đơn sơ nhất là chọn k phần tử ngẫu nhiên phân biệt $x_1,x_2…..x_k \in X$, tính $z_1 = h(x_1),1<= i <= k$ và sau đó xác định xem liệu có xảy ra va chạm nào không (bằng cách, chẳng hạn như sáp xếp lại các z_i).

Quá trình này tương tự với việc ném k quả bóng vào thùng và sau đó kiểm tra xem liệu có thùng nào chứa ít nhất hai quả hay không (k qủa bóng tương đương với k giá trị x_i ngẫu nhiên và n thùng tương ứng với n phần tử có thể trong Z).

Ta sẽ giới hạn dưới của xác suất tìm thấy một va chạm theo phương pháp này.Do chỉ quan tâm đến giới hạn dưới về xác suất va chạm nên ta sẽ giả sử rằng $|h^{-1}(z)| \approx m/n$ với mọi $z \in Z$. (đây là giả thiết hợp lí :Nếu các ảnh đảo không xấp xỉ bằng nhau thì xác suất tìm thấy một va chạm sẽ tăng lên).

Vì các ảnh đảo đều có kích thước bằng nhau và các $x_{i\ được}$ chọn một cách ngẫu nhiên nên các z_i nhận được có thể xem như các phần tử ngẫu nhiên của Z. Song việc tính toán xác suất để các phần tử ngẫu nhiên $z_1, z_{2,....} z_k \in Z$ là riêng biệt khá đơn giản.Xét các z_i theo thứ tự $z_1, …,z_k$. Phép chọn z_1 đầu tiên là tuỳ ý. Xác suất để $z_2 \neq z_1$ là 1-1/n; xác suất để $z_3 \neq z_1$ và z_2 là 1- 2/n. vv…

Vì thế ta ước lượng xác suất để không có va chạm nào là:

$(1-1/n)(1-2/n)… (1-(k-1/n)) = (1-1/n)$

Nếu x là số thực nhỏ thì $1- x \approx e^{-x}$. Ước lượng này nhận dược từ hai số hạng đầu tiên của cá chuỗi khai triển.

$$e^{-x} = 1 - x + x^2/2! - x^3/3! ...$$

Khi đó xác suất không có va chạm nào là :

$$\prod_{i=1}^{k-1}(1-\frac{i}{n}) \approx \prod_{i=1}^{k-1} e^{-1/n} = e^{-k(k-1)/n}$$

Vì thế ta ước lượng xác suất để có ít nhất một va chạm là

$$1-e^{-k(k-1)/n}$$

Nếu kí hiệu xác suất này là ε thì có thể giải phương trình đối với k (như một hàm của n và ε)

$$1-e^{-k(k-1)/n} \approx 1 -\varepsilon$$

$$-k(k-1)/n \approx \ln(1-\varepsilon)$$

$$k^2 - k \approx n \ln 1/(1-\varepsilon)$$

Nếu bỏ qua số hạng k thì :

$$k = \sqrt{n \ln \frac{1}{1-\varepsilon}}$$

Nếu lấy $\varepsilon = 0.5$ thì

$$k \approx 1.17\sqrt{n}$$

Điều này nói lên rằng, việc chặt (băm) trên $\sqrt{n}$ phần tử ngẫu nhiên của X sẽ tạo ra một va chạm với xác suấtt 50%. Chú ý rằng, cách chọn ε khác sẽ dẫn đến hệ số hằng số khác song k vẫn tỷ lên với $\sqrt{n}$.

Nếu X là tập người,Y là tập gồm 365 ngỳ trong năm (không nhuận tức tháng 2 có 29 ngày) còn h(x) là ngày sinh nhật của x, khi đó ta sẽ giả guyết bằng nhgịch lý ngày sinh nhật. Lấy n = 365, ta nhận được k ≈ 22,3. Vì vậy, như đã nêu ở trên, sẽ có ít nhất 2 người có ngày sinh nhật trùng nhau trong 23 người ngẫu nhiên với xác suất ít nhất bằng 1/2.

Tấn công ngày sonh nhật đặt giới hạn cho các kích thước các bản tóm lược thông báo. bản tóm lược thông báo 40 bit sẽ không an toàn vì có thể tìm thấy một va chạm với xác suất 1/2 trên 2^{20} (khoảng1.000.000)đoạn chặt ngẫu nhiên. Từ đây cho thấy rằng, kích thước tối thiểu chấp nhận được của bản tóm lược thông báo là 128 bit (tấn công ngày sinh nhật cần trên 2^{64} đoạn chặt trong trường hợp này). Đó chính là lý do chọn bản tóm lược thông báo dài 160 bit trong sơ đồ DSS.

Hình7.3. Hàm hash chaum-Van heyst-Plitzmann.

Giả sử p là số nguyên tố lớn và q =(p-1)/2 cũng là số nguyên tố. Cho α và β là hai phần tử nguyên thuỷ của Zp. Giá trị $\log_\alpha \beta$ không công khai và giả sử rằng không có khả năng tính toán được giá trị của nó.

Hàm Hash:

$$h: \{0,...,q-1\} \times \{0,...,q-1\} \rightarrow Zp \setminus \{0\}$$

được định nghĩa như sau:

$$h(x1,x2) = \alpha^{x_1} \beta^{x_2} \bmod p$$

7.3. hàm hash logarithm rời rạc

Trong phần này ta sẽ mô tả một hàm Hash do Chaum-Van Heyst và Pfĩtmann đưa ra. Hàm này an toàn do không thể tính được logarithm rời rạc. Hàm Hast này không đủ nhanh để dùng trong thực tế song nó đơn giản và cho một ví dụ tốt về một hàm Hash có thể an toàn dưới giả thuyết tính toán hợp lý nào số. Hàm Hash Caum-Van Heyst- Pfĩtmann được nêt trong hình 7.3. Sau đây sẽ chứng minh một định lý liên quan đến sự an toàn của hàm Hast này.

Định lý 7.2.

N*ếu cho trước một va chạm với hàm Hash Chaum-Van Heyst-Pfĩtmann h có thể tính được logarithm rời rạc* $\log_\alpha \beta$ *một cách có hiệu quả.*

Chứng minh

Giả sử cho trước va chạm

$$h(x_1,x_2) = h(x_3,x_4)$$

trong đó $(x_1,x_2) \neq (x_3,x_4)$. Như vậy ta có đồng dư thức sau:

$$\alpha^{x_1}\beta^{x_2} = \alpha^{x_3}\beta^{x_4}$$

hay

$$\alpha^{x_1}\beta^{x_2} \equiv \alpha^{x_3}\beta^{x_4} \pmod p$$

Ta kí hiệu

$$D = UCLN\ (x_4-x_2,p-1)$$

Vì p-1 =2q ,q là số nguyên tố nên $d \in \{1, 2, q, p-1\}$. Vì thế, ta có 4 xác suất với d sẽ xem xét lần lượt dwois đây.

Trước hết ,giả sử d =1 ,khi đó cho

$$y= (x_4-x_2)^{-1} \bmod (p-1)$$

ta có

$$\beta \equiv \beta^{(x_4-x_2)y} \pmod p$$
$$\equiv \alpha^{(x_1-x_2)y} \pmod p$$

Vì thế, có thể tính loarithm rời rạc $\log_\alpha\beta$ như sau:

$$\log_\alpha\beta = (x_1-x_3)\ (x_4-x_2)^{-1} \bmod (p-1)$$

Tiếp theo, giả sử d=2. Vì p-1 =2q, lẻ nên $UCLN(x_4-x_2,q) =1$. Giả sử:

$$y=(x_4-x_2)^{-1} \bmod q$$

xét thấy $(x_4-x_2)y = kq+1$
với số nguyên k nào đó. Vì thế ta có:

$$\beta^{(x_4-x_2)y} \equiv \beta^{kq+1} \pmod p$$
$$\equiv (-1)^k \beta \pmod p$$
$$\equiv \pm \beta \pmod p$$

Vì $\beta^q \equiv -1 \pmod p$
Nên

$$\alpha^{(x4-x2)y} \equiv \beta^{(x1-x3)} \pmod p$$
$$\equiv \pm \beta \pmod p$$

Từ đó suy ra rằng:

$$\log_\alpha \beta = (x_1-x_3)y \bmod (p-1)$$
$$\log_\alpha \beta = (x_1-x_3)y \bmod (p-1)$$

Ta có thể dễ dàng kiểm tra thấy một trong hai xác suất trên là đúng. Vì thế như trong trường hợp d =1, ta tính được $\log_\alpha \beta$.

Xác suất tiếp theo là d = q. Tuy nhiên

$$q-1 \geq x_1 \geq 0$$

và

$$q-1 \geq x_3 \geq 0$$

nên

$$(q-1) \geq x_4-x_2 \geq -(q-1)$$

do vậy UCLN$(x_4-x_2,p-1)$ không thể bằng q, nói cách khác trường hợp này không xảy ra.

Xác suất cuối cùng là d = p-1. Điều nàychỉ xảy ra khi x2 =x4. Song khi đó ta có

$$\alpha^{x_1}\beta^{x_2} \equiv \alpha^{x_3}\beta^{x_4} \pmod p$$

nên

$$\alpha^{x_1} \equiv \alpha^{x_3} \pmod p$$

và $x_1 = x_2$. Như vậy $(x_1,x_2) = (x_3,x_4) \Rightarrow$ mâu thuẫn. Như vậy trường hợp này cũng không thể có.

Vì ta đã xem xét tất cả các giá trị có thể đối với d nên có thể kết luận rằng ,hàm Hash h là không va chạm mạnh miễn là không thể tính được logarithm rời rạc $\log_\alpha \beta$ trong Z_p.
Ta sẽ minh hoạ lý thuyết nêu trên bằng một ví dụ.
Ví dụ 7.1

Giả sử p =12347 (vì thế q = 6173), $\alpha = 2$, $\beta = 8461$. Giả sử ta được đưa trước một va chạm

$$\alpha^{5692} \beta^{144} \equiv \alpha^{212} \beta^{4214} \pmod{12347}$$

Như vậy $x_1 = 5692$, $x_2 = 144$, $x_3 = 212$, $x_4 = 4214$. Xét thấy UCLN $(x_4 - x_2,p-1)$ =2 nên ta bắt đầu bằng việc tính

$$y = (x_4 - x_2)^{-1} \bmod q$$
$$= (4214 - 144)^{-1} \bmod 6173 = 4312$$

Tiếp theo tính

$$y = (x_1 - x_3) \bmod (p-1)$$
$$= (5692 - 212)\ 4312 \bmod 12346$$
$$= 11862$$

Xét thấy đó là trường hợp mà $\log_\alpha\beta \in \{y', y'+q \bmod (p-1)\}$. Vì

$$\alpha^y \bmod p = 2^{12346} = 9998$$

nên ta kết luận rằng:

$$\log_\alpha\beta = y' + q \bmod (p-1)$$
$$= 11862 + 6173 \bmod 12346$$
$$= 5689$$

như phép kiểm tra, ta có thể xác minh thấy rằng

$$2^{5689} = 8461 \pmod{12347}$$

Vì thế , ta các định được $\log_\alpha\beta$.

7.5.**các hàm hash mở rộng**

Cho đến lúc này, ta đã xét các hàm Hash trong vùng hữu hạn. Bây giờ ta nghiên xéu cách có thể mở rộng một hàm Hash không va chạm mạnh từ vùng hữu hạn sang vùng vô hạn. Điều này cho phép ký các bức điện có độ dài tuỳ ý. Gỉa sử h: $(Z_2)^m \rightarrow (Z_2)^t$ là một hàm hash không va chạm mạnh ,trong đó $m \geq t-1$. Ta sẽ dùng h đêu xây dựng hàm hash không va chạm mạnh h: $X \rightarrow (Z_2)^t$ trong đó

$$X = \bigcup_{i=m}^{\infty} (Z_2)^t$$

Trước tiên xét trường hợp $m \geq t+2$.

Ta sẽ xem các phần tử của X như các xây bit. |x| chỉ độ dàI của x (tức số các bit trong x) và x||y ký hiệu sự kết hợp các xây x và y. Giả sử $|x| = n > m$. Có thể biểu thị x như một chuỗi kết hợp.

$$X = x_1||x_2||...||x_k$$

Trong đó

$$|x_1| = |x_2| = ... = |x_{k-1}| = m- t-1$$

và $|x_k| = m- t- 1- d$

Hình 7.4. Mở rộng hàm hash h thành h* ($m \geq t+2$)

1. For i= 1 to k-1 do
   $y_i = x_i$
2. $y_k = x_k \| 0^d$
3. cho y_{k+1} là biểu diễn nhị phân của d
4. $g_i = h(0I+1\|y_1)$
5. for i=1 to k do
   $g_i+1 = h(g_i\|1\|y_i+1)$
6. $h^*(x) = g_k +1$

Trong đó m- t- 2 ≥ d ≥0. Vì thế ta có

$$k= \left\lfloor \frac{n}{m-t-1} \right\rfloor$$

Ta định nghĩa h*(x) theo thuật toán biểu kiễn trong hình 7.4.

Kí hiệu $y(x) = y1\|y2\|...\|y_{k-1}$

Nhận xét rằng y_k được lập từ x_k bằng cách chèn thêm d số 0 vào bên phảI để tất cả các khối y_i ($k \geq i \geq 1$)đều có chiều dàI m-t-1. Cũng như trong bước 3 y_{k+1} sẽ được đệm thêm về bên tráI các số 0 sao cho $|y_{k+1}| = m-t-1$.

Để băm nhỏ x ,trước hết ta xây dựng hàm y(x) và sau đó "chế biến" các khối $y_1...y_{k+1}$ theo một khuôn mẫu cụ thể. Điều quan trọng là $y(x) \neq y(x')$ khi $x \neq x$. Thực tế y_{k+1} được định nghĩa theo cách các phép ánh xạ $x \rightarrow y(x)$là một đơn ánh.

Định lý sau đây chứng minh rằng h* là an toàn khi h an toàn.

Định lý 7.3

Giả sử h: $(Z_2)^n \rightarrow (Z_2)$ là hàm hash không va chạm mạnhm≥ t+2. Khi đó hàm h: $\cup_{i=m}^{\infty} (Z_2)^t \rightarrow (Z_2)^t$ được xây dựng như trên hình 7.4 là hàm hash không*
và chạm mạnh.

Chứng minh:

Giả sử rằng ,ta có thể tìm được $x \neq x'$ sao cho $h^*(x) = h^*(x')$. Nếu cho trước một cặp như vậy, ta sẽ chỉ ra cách có thể tìm được một va chạm đối với

h trong thời gian đa thức. Vì h được giả thiết là không va chạm mạnh nên dẫn đến một mâu thuẫn như vậy h sẽ được chứng minh là không va chạm mạnh.

Kí hiệu $y(x)= y_1||..||y_{k+1}$

Và $y(x') = y_1'||...||y_{k+1}'$

ở đây x và x' được đệm thêm d và d' số 0 tương ứng trong bước 2. Kí hiệu tiếp các giá trị được tính trong các bước 4 và 5 là $g_1,g_2....,g_{k+1}$ và $g_1',....,g_{k+1}'$ tương ứng.

Chúng ta sẽ đồng nhất hai trường hợp tuỳ thuộc vào việc có hay không $|x| \equiv |x'| \pmod{m-t-1}$.

Trường hợp1: $|x| \neq |x'| \pmod{m-t-1}$

Tại đây $d \neq d'$ và $y_{k+1} \neq y'_{k+1}$. Ta có:

$$\begin{aligned} H(g_k||1||y_{k+1}) &= g_{k+1} \\ &= h^*(x) \\ &= h^*(x') \\ &= g'_{\iota+1} \\ &= h(g'_{\iota+1}||1||y'_{\iota+1)} \end{aligned}$$

là một va chạm đối với h vì $y_{k+1} \neq y'_{k+1.}$

Trường hợp2: $|x| \equiv |x'| \pmod{m-t-1}$

Ta chia trường hợp này thành hai trường hợp con:

Trường hợp 2a: $|x| = |x'|$.

Tạ đây ta có k= l và $y_{k+1} = y'_{k+1}$. Ta vắt đầu như trong trường hợp 1:

$$\begin{aligned} h(g_k||1||y_{k+1}) &= g_{k+1} \\ &= h^*(x) \\ &= h^*(x') \\ &= h(g'_k||1||y'_k+1) \end{aligned}$$

Nếu $g_k = g'_k$ thì ta tìm thấy một va chạm đối với h, vì thế giả sử $g_k = g'_k$ khi đó ta sẽ có:

$$\begin{aligned} h(g_{k-1}||1||y_k) &= g_k \\ &= g'_k \end{aligned}$$

$=h(0^{i+1}||y_1)$

Hoặc là tìm thấy một va chạm đối với h hoặc $g_{k-1} = g'_k - 1$ và $y_k = y'_k$. Giả sử không tìm thấy va chạm nào ,ta tiếp tục thực hiện ngược các bước cho đến khi cuối cùng nhận được :

$$h(0_{i+1}||y_1) = g_1$$
$$= g'_{i-k+1}$$
$$= g(g'_{i-k}||1||y'_{i-k+1}).$$

Nhưng bit thứ (t+1) của $0^{i+1}||y_1$ bằng 0 và bit thứ (t+1) của $g'_{i-k+1}||1||y'_{i-k+1}$ bằng 1. Vì thế ta tịm thấy một va chạm đối với h.

Vì đã xét hết các trường hợp có thể nên ta có kết luận mong muốn.

Cấu trúc của hình 7.4 chỉ được dùng khi m>= t+2. Bây giờ ta hãy xem xét tình huống trong đó m = t+1. Cần dùng một cấu trúc khác cho h. Như trước đây, giả sử |x|=n>m. Trước hết ta mã x theo cách đặc biệt. Cách này dùng hàm f có định nghĩa như sau:

f(0) = 0
f(1) = 01

Thuật toán để xây dựng h*(x)được miêu tả trong hình 7.5

Phép mã $x \rightarrow y = y(x)$ được định nghĩa trong vước 1 thoả mãn hai tính chất quan trọng sau:

1. nếu $x \neq x'$ thì $y(x) \neq y(x')$ (tức là $x \rightarrow y(x)$ là một đơn ánh)
2. Không tồn tạI hai chuỗi $x \neq x'$ và chuỗi z sao cho $y(x) = z||y(x')$. Nói cách khác không cho phép mã hoá nào là fpsstix của phép mã khác. ĐIều này dễ dàng thấy được do chuỗi y(x) bắt đầu bằng 11 và không tồn tạI hai số 1 liên tiếp trong phần còn lạI của chuỗi).

Hình 7.5 Mở rộng hàm hash h thành h* (m = t+1)

1. Giả sử $y = y_1y_2...y_k = 11||f(x_1)||....||f(x_n)$
2. $g_1 = h(0^1||y_1)$
3. for i=1 to k-1 do
   $g_{i+1} = h(g_i||y_{i+1})$
4. $h^*(x) = g_k$

Định lý 7.4

Giả sử h: $(Z_2)^n \rightarrow (Z_2)$ là hàm hash không va chạm mạnh. Khi đó hàm h: $\cup_{i=m}^{\infty} (Z_2)^t \rightarrow (Z_2)^t$ được xây dựng như trên hình 7.5 là hàm hash không va chạm mạnh.*

Chứng minh:

Giả sử rằng ta có thể tìm được x ≠x' sao cho h*(x)=h*(x'). Kí hiệu:

$$y(x) = y_1 y_2 \ldots . y_k$$

và $$y(x') = y'_1 y'_2 \ldots . y'_l$$

Ta xét hai trường hợp:

Trường hợp 1: k=l

Như trong định lý 7.3 hoặc ta tìm thấy một va chạm đỗi với h hoặc ta nhận được y = y' song đIều này lạI bao hàm x = x', dẫn đến mâu thuẫn.

Trường hợp2: k≠ l

Không mất tính tổng quát ,giả sử l>k . trường hợp này xử lý theo kiểu tương tự. Nếu giả thiết ta không tìm thấy va chạm nào đối với h ,ta có dãy các phương trình sau:

$$y_k = y'_l$$
$$y_{k-1} = y'_{l-1}$$
$$\ldots\ldots\ldots\ldots$$
$$y_1 = y'_{l-k+1}$$

Song đIều này mâu thuẫn với tính chất "không posfixx" nêu ở trên. Từ đây ta kết luận rằng h* là hạm không va chạm.

Ta sẽ tổng kết hoá hai xây dựng trong phần này và số các ứng dụng của h cần thiết để tính h* theo định lý sau:

Định lý 7.5

Giả sử h: $(Z_2)^n \rightarrow (Z_2)$ là hàm hash không va chạm mạnh,ở đây m>=t+1. Khi đó tồn tạI hàm không va chạm mạnh

$h^*: \cup_{i=m}^{\infty} (Z_2)^t \rightarrow (Z_2)^t$

Số lần h được tính trong ước lượng h nhiều nhất bằng :*

$$l + \left\lceil \frac{n}{m-t-1} \right\rceil \text{ nếu } m>=t+2$$

$2n+2$ *nếu* $m= t+2$

trong đó $|x|=n$.

7.6 các hàm hash dựa trên các hệ mật

Cho đến nay, các phương pháp đã mô tả để đưa đến nhứng hàm hash hầu như đều rất chậm đối với các ứng dụng thực tiễn. Một biện pháp khác là dùng các hệ thống mã hoá bí mật hiện có để xây dừng các hàm hash. Giả sử rằng (P,C,K,E,D) là một hệ thống mật mã an toàn về mặt tính toán. Để thuận tiện ta cũng giả thiết rằng $P = C = K = (Z_2)^n$.ở đâychọn n>=128 để xây ngăn chặn kiểu tấn công ngày sinh nhật. ĐIều này loạI trừ việc dùng DES (vì độ dài khoá của DES khác với độ dài bản rõ).

Giả sử cho trước một xâu bit:

x= x1||x2||....||xk

trong đó xi $\in$ $(Z_2)^n$, 1≤ i ≤ (nếu số bit trong x không phải là bội của n thì cần chèn thêm vào x theo cách nào đó. Chẳng hạn như cách làm trong nục 7.5. Để đơn giản ta sẽ bỏ qua đIểm này).

Ý tưởng cơ bản là bắt đầu bằng một "giá trị ban đầu" cố định g_0 =IV và sau đó ta xây dựng $g_1,...,g_k$ theo quy tắc thiết lập :

$$g_i = f(x_i, g_{i-1}).$$

ở đây f là hàm kết hợp toàn bộ các phép mã hoá của hệ mật được dùng. Cuối cùng ta định nghĩa bản tóm lược của thông báo $h(x) = g_k$.

Vài hàm hash kiểu này đã được đề xuất và nhiều loại trong chúng tỏ ra không an toàn (không phụ thuộc vào việc liệu hệ mật cơ bản có an toàn hay không). Tuy nhiên , có 4 phương án khác nhau có vẻ an toàn của sơ đồ này :

$$g_i = e_{g_{i-1}}(x_i) \oplus x_i$$

$g_i = e_{g_{i-1}}(x_i) \oplus x_I \oplus g_{i-1}$
$g_i = e_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_I$
$g_i = e_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_I \oplus g_{i-1}.$

7.7 **Hàm hash MD4.**

Hàm hash MD4 được Riverst đề xuất năm 1990 và một hiên bản mạnh là MD5 cũng được đưa ra năm 1991. Chuẩn hàm hash an toàn (hay SHS) phức tạp hơn song cũng dưa tên các phương pháp tương tự. Nó được công bố trong hồ sơ liên bang năm 1992 và được chấp nhận làm tiêu chuẩn vào ngày 11/5/1993. Tất cả các hàm hash trên đều rất nhanh nên trên thực tế chúng dùng để kí các bức điện dài.

Trong phần này sẽ mô tả chi tiết MD4 và thảo luận một số cảI tiến dùng trong MD5 và SHS.

Cho trước một xâu bit trước hết ta tạo một mạng:

$$M = M[0]\ M[1] \ldots M[N-1]\ .$$

trong đó M[i] là xâu bit có độ dàI 32 và $N \equiv 0 \bmod 16$. Ta sẽ gọi M[i] là *từ*. M được xây dựng từ x bằng thuật toán trong hình 7.6.

Hình 7.6 Xây dựng M trong MD4

1. d = 447-(|x| mod 512)
2. giả sử ℓ là kí hiệu biểu diễn nhị phân của |x| mod 2^{64}.|l| = 64
3. $M = x||1||0^d||\ell$

Trong việc xây dựng M, ta gắn số 1 ssơn lẻ vào x, sau đó sẽ gài thêm các số 0 đủ để độ dài trở nên đồng dư với 448 modulo 512.,cuối cùng nối thêm 64 bit chưa biểu diễn nhị phân về độ dàI (ban đầu) của x(được rút gọn theo móulo 2^{64} nếu cần). Xâu kết quả M có độ dàI chia hết cho 512. Vì thế khi chặt M thành các từ 32 bit , số từ nhận được là N-sẽ chia hết cho 16.

Bây giờ, tiếp tục xây dựng bản tóm lược thông báo 128 bit. Hình 7.7 đưa ra mô tả thuật toán ở mức cao. Bản tóm lược thông báo được xây dựng như sự kết nối 4 từ A,B,C và D mà ta sẽ gọi là các thanh ghi. Bốn thanh ghi được khởi động như trong bước 1. Tiếp theo ta xử lí bảng M 16 bit từ cùng lúc. Trong mỗi vòng lặp ở bước 2, đầu tiên lấy 16 từ "tiếp theo" của M và lưu

chúng trong bảng X (bước 3). Các giá trị của bốn thanh ghi dịch sau đó sẽ được lưu lại (bước 4). Sau đó ta sẽ thực hiện ba vòng "băm" (hash). Mỗi vaòng gồm một phép toán thực hiện trên một trong 16 từ trong X. 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 update (cập nhật) trong bước 8 bằng cách cộng ngược các giá trị lưu trước đó trong bước 4. Phép cộng này được xác định là cộng các số nguyên dương ,được rút gọn theo modelo 2^{32}.

Ba vòng trong MD4 là khác nhau (không giông như DES. 16 vòng đều như nhau). Trước hết ta sẽ mô tả vàI phép toán khác nhau trong ba vòng này. Trong phần sau,ta kí hiệu X và Y là các từ đầu vào và mỗi phép toán sẽ tạo ra một từ đầu ra. Dưới đây là phép toán được dùng:

X∧Y là phép "AND" theo bit giữa X và Y
X∨Y là phép "OR" theo bit giữa X và Y
X⊕Y là phép "XOR" theo bit giữa X và Y
¬X chỉ phần bù của X
X+Y là phép cộng theo modulo 2^{32}.
X<< s phép dịch vòng tráI X đI s vị trí (31>= s >=0).

Chú ý rằng, tất cả các phép toán trên đều tất nhanh và chỉ có phép số học duy nhất được dùng là phép cộng modulo 2^{32}. Nếu MD4 được ứng dụng thì cần tính đến kiến trúc cơ bản của máy tính mà nó chạy trên đó để thực hiện chính xác phép cộng. Giả sử $a_1a_2a_3a_4$ là 4 byte trong từ xem mỗi a_i,như một số nguyên trong dảI 0-255 được biểu diễn dưới dạng nhị phân. Trong kiến trúc kiểu endian lớn (chẳng hạn như trên trạm Sunsparc) từ này biểu diễn số nguyên.

$$a_1 2^{24} + a_2 2^{16} + a_3 2^8 + a_4$$

Trong kiến trúc kiểu endian nhỏ (chẳng hạn họ intel 80xxx). Từ này biểu diễn số nguyên:

$$a_4 2^{24} + + a_3\, 2^{16} + a_2\, 2^8 + a_1$$

MD4 giả thiết dùng kiến trúc kiểu endian nhỏ. ĐIều quan trọng là bản tóm lược thông báo độc lập với kiến trúc cơ bản. Vì thể nếu muốn chạy MD4 trên máy tính endian lớn cần thực hiện phép cộng X+Y như sau:

1. Trao đổi x_1 và x_4; x_2 và x_3; y_1 và y_4; y_2 và y_3
2. Tính Z = X+Y mod 2^{32}
3. Trao đổi z_1 và z_4 ; z_2 và z_3.

Hình 7.7 hàm hash MD4

1. A= 67452301 (hệ hexa)
   B = efcdab89 (hệ hexa)
   C = 98badcfe (hệ hexa)
   D = 10325476 (hệ hexa)
2. for i = 0 to N/16-1 do
3. for i = 1 to 15 do
   X[i] = M[16i+j]
4. AA = A
   BB = B
   CC = C
   DD = D
5. round1
6. round2
7. round3
8. A = A+AA
   B = B+ BB
   C = C + CC
   D = D + DD

Các vòng 1, 2 và 3 của MD4 dùng tương ứng ba hàm f, g, và h. Mỗi hàm này là một hàm boolean tính theo bit dùng 2 từ làm đầu vào và tạo ra một từ tại đầu ra. Chúng được xác định như sau:

$$f(X,Y,Z) = (X \wedge Y) \vee ((-X) \wedge Z)$$
$$g(X,Y,Z) = (X \wedge Y) \vee (X \wedge Z) \vee (Y \wedge Z)$$
$$h(X,Y,Z) = X \oplus Y \oplus Z$$

Các hình 7.8-7.10 sẽ mô tả đầy đủ các vòng 1,2 và 3 của MD4.

MD4 được thiết kế chạy rất nhanh và quả thực phần mềm chạy trên máy Sun SPARC có tốc độ 1.4 Mbyte/s. Mặt khác, khó có thể nói đIều gì cụ thể về độ mật của hàm hash, chẳng hạn như MD4 vì nó không dựa trên vàI toán khó đã nghiên cứu kĩ (ví dụ như phân tích nhân tử trên bàI toán logarithm rời rạc). Vì thế trong trường hợp Dé sự tin cậy vào độ an toàn của hệ thống chỉ có thể đạt được về thời gian và như vậy có thể hi vọng hệ thống vừa được nghiên cứu và không tìm thấy sự không an toàn nào.

Hình 7.8 : Vòng 1 của MD4 .(round 1)

```
1.   A = (A+ f(B,C,D) + X[0]) << 3
2.   D = (D + f(A,B,C) +X[1]) << 7
3.   C = (C + f(D,A,C) +X[2]) << 11
4.   B = (B + f(C,D,A) +X[3]) << 19
5.   A = (A + f(B,C,D) +X[4]) << 3
6.   D = (D + f(A,B,C) +X[5]) << 7
7.   C = (C + f(D,A,C) +X[6]) << 11
8.   B = (B + f(C,D,A) +X[7]) << 19
9.   A = (A + f(B,C,D) +X[8]) << 3
10.  D = (D + f(A,B,C) +X[9]) << 7
11.  C = (C + f(D,A,C) +X[10]) << 11
12.  B = (B + f(C,D,A) +X[11]) << 19
13.  A = (A + f(B,C,D) +X[12]) << 3
14.  D = (D + f(A,B,C) +X[13]) << 7
15.  C = (C + f(D,A,C) +X[14]) << 11
16.  B = (B + f(C,D,A) +X[15]) << 19
```

Mặc dù MD4 vẫn chưa bị phá song các phiên bản yếu cho phép bỏ qua hoặc vòng thứ nhất hoặc thứ ba dều có thể bị phá không khó khăn gì. nghĩa là dễ dàng tìn thấy các va chạm đối với các phiên bản chỉ có hai vòng. Phiên vản mạnh của MD5 là MD5 được công bố năm 1991. MD5 dùng vòng thay cho ba và chậm hơn 30% so với MD4 (khoảng 0.9 Mbyte/s trên cùng máy).

Chuẩn hàm hash an toàn phức tạp và chậm hơn. Ta sẽ không mô tả đầy đủ song sẽ chỉ ra một vàI cảI tiến trên nó.

1. SHS được thiết kế để chạy trên máy kiến trúc endian lớn hơn là trên máy endian nhỏ.
2. SHA tạo ra các bản tóm lược thông báo 5 thanh ghi (160 bit).
3. SHS xử lí 16 từ của bức điện cùng một lúc như MD4. Tuy nhiên, 16 từ trước tiên được ”mở rộng” thành 80 từ ,sau đó thực hiện một dãy 80 phép tính ,mỗi phép tính trên một từ.
   Hình 7.9 Vòng 2 củaMD4.

```
1.   A = (A +g(B,C,D) + X[0] + 5A827999) <<3
2.   D = (D +g(A,B,C) + X[4] + 5A827999) <<5
3.   C = (C +g(D,A,B) + X[8] + 5A827999) <<9
4.   B = (B +g(C,D,A) + X[12] + 5A827999) <<13
5.   A = (A +g(B,C,D) + X[1] + 5A827999) <<3
6.   D = (D +g(A,B,C) + X[1] + 5A827999) <<5
7.   C = (C +g(D,A,B) + X[5] + 5A827999) <<9
8.   B = (B +g(C,D,A) + X[13] + 5A827999) <<13
9.   A = (A +g(B,C,D) + X[2] + 5A827999) <<3
10.  D = (D +g(A,B,C) + X[6] + 5A827999) <<5
11.  C = (C +g(D,A,B) + X[10] + 5A827999) <<9
12.  B = (B +g(C,D,A) + X[14] + 5A827999) <<13
13.  A = (A +g(B,C,D) + X[3] + 5A827999) <<3
14.  D = (D +g(A,B,C) + X[7] + 5A827999) <<5
15.  C = (C +g(D,A,B) + X[11] + 5A827999) <<9
16.  B = (B +g(C,D,A) + X[15] + 5A827999) <<13
```

Dùng hàm mở rộng sau đây: Cho trước 16 từ X[0] ...X[15], ta tính thêm 64 từ nữa theo quan hệ đệ quy.

$X[j] = X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16], 79 \geq j \geq 16$ 7.1

Kết quả của phương trình (7.1) là mỗi một trong các từ X[16]...X[79] được thiết lập bằng cách cộng ⊕ với một tập con xác định nào đó của các từ X[0]...X[15].

Ví dụ: Ta có:

$X[16] = X[0] \oplus X[2] \oplus X[8] \oplus X[13]$

$X[17] = X[1] \oplus X[3] \oplus X[9] \oplus X[14]$

$X[18] = X[2] \oplus X[4] \oplus X[10] \oplus X[15]$

$X[19] = X[0] \oplus X[2] \oplus X[3] \oplus X[5] \oplus X[8] \oplus X[11] \oplus X[13]$

$X[79] = X[1] \oplus X[4] \oplus X[15] \oplus X[8] \oplus X[12] \oplus X[13]$.

Một đề xuất đòi hỏi sửa lại SHS liên quan đến hàm mở rộng trong đó đề nghị đặt lại phương trình 7.1 như sau:

$X[j] = X[j-3] \oplus X[j-8] \oplus X[j-14] \oplus X[j-16] <<1; 79 \geq j \geq 16$ (7.2)

Hình 7.10 : Vòng ba của MD4.

```
1.   A = (A + h(B,C,D) + X[0] + 6ED9EBA1) <<3
2.   D = (D + h(A,B,C) + X[8] + 6ED9EBA1) <<9
3.   C = (C + h(D,A,B) + X[4] + 6ED9EBA1) << 11
4.   B = (B + h(C,D,A) + X[12] + 6ED9EBA1) << 15
5.   A = (A + h(B,C,D) + X[2] + 6ED9EBA1) <<3
6.   D = (D + h(A,B,C) + X[10] + 6ED9EBA1) <<9
7.   C = (C + h(D,A,B) + X[6] + 6ED9EBA1) << 11
8.   B = (B + h(C,D,A) + X[14] + 6ED9EBA1) << 15
9.   A = (A + h(B,C,D) + X[1] + 6ED9EBA1) <<3
10.  D = (D + h(A,B,C) + X[9] + 6ED9EBA1) <<9
11.  C = (C + h(D,A,B) + X[13] + 6ED9EBA1) << 11
12.  B = (B + h(C,D,A) + X[13] + 6ED9EBA1) << 15
13.  A = (A + h(B,C,D) + X[3] + 6ED9EBA1) <<3
14.  D = (D + h(A,B,C) + X[11] + 6ED9EBA1) <<9
15.  C = (C + h(D,A,B) + X[7] + 6ED9EBA1) << 11
16.  B = (B + h(C,D,A) + X[15] + 6ED9EBA1) << 15
```

Như trước đây , toán tử ''<<1'' là phép dịch vòng trái một vị trí.

7.8 **nhãn thời gian (timestamping)**.

Một khó khăn trong sơ đồ chữ kí là thuật toán kí có thể bị tổn thương. Chẳng hạn, giả sử Oscar có khả năng xác định số mũ mật a của Bob trên bất kì bức điện nào mà anh ta muốn. Song còn vấn đề khác (có thể nghiêm trọng hơn)là : từ đây người ta sẽ đặt câu hỏi về tính xác thực của tất cả các bức điện mà Bob kí, kể cả những bức điện mà anh ta kí trước khi Oscar đánh cắp được thuật toán.

Từ đây lại có thể nảy sinh tình huống không mong muốn khác : giả sử Bob kí một bức điện và sau đó từ chối là đã không kí nó. Bob có thể công khai thuật toán kí của mình sau đó công bố rằng chữ kí của anh ta trên bức điện đang nói trên là giả mạo.

Lí do có các kiểu sự kiện này là do không có các nào các định bức điện được kí khi nào. Nhãn thời gian có thể cung cấp bằng chứng rằng, bức điện đã được kí vào thời điểm cụ thể nào đó. Khì đó nế thuật toán kí của Bob có nhược điểm (bị tổn thương) thì bất kì chữ kí nào anh ta kí trước đó sẽ không còn hợp lệ. ĐIều này giống với kiểu thực hiên các thẻ tín dụng: Nếu ai đó làm mất thẻ tín dụng và thông báo cho nhà băng đã phát hành thì thẻ mất hiệu lực. Song các cuộc mua bán thực hiện trước khi mất nó thì vẫn không bị ảnh hưởng.

Trong phần này sẽ mô tả một vàI phương pháp gắn nhãn thời gian. Trước hết,nhận xét rằng, Bob có thể tạo ra một nhãn thời gian có sức thuyết phục trên chữ kí của anh ta. Đầu tiên, Bob nhận được một thông tin "hiện thời " có sẵn công khai nào đó, thông tin này không thể dự đoán được trước khi nó xảy ra. Ví dụ thông tin chứa tất cả các lợi thế về môn bóng chày của các liên minh chính từ ngày trước đó, hay các giá trị của tất cả cổ phần đwocj lên danh sách trong sở giao dịch chứng khoán NewYork. Ta kí hiệu thông tin này bằng chữ *pub*.

Bây giờ giả sử Bob muốn dán nhãn thời gian trên chữ kí của mình trên bức điện x. Giả thiết rằng, h là hàm hash công khai biết trước. Bob sẽ thực hiện theo thuật toán trong hình 7.11.Sau đây là cách sơ đồ làm việc : sự có mặt của thông tin pub có nghĩa là bob không thể tạo ra được y trước ngày đang nói đến. Còn một thực tế là y công bố trong một tờ báo ra ngày tiếp theo chứng tỏ rằng bob đã không tính y sau ngày được nói đến. Vì thế chữ kí y của bob bị hạn chế trong thời hạn một ngày. Cũng nhận xét thấy rằng, bob không để lộ bức điện x trong sơ đồ này vì chỉ có x được công bố ... Nếu cần bob có thể chứng minh rằng x là bức điện mà anh ta đã kí và dán nhãn thời gian một cách đơn giản là làm lộ nó.

Cũng không khó khăn tạo ra tạo ra các nhãn thời gian nếu có một cơ quan dịch vụ dán nhãn đáng tin cậy. Bob có thể tính z = h(x) và y = sig_k (z) và sau đó gửi (z và x) đến cơ quan làm dịch vụ dán nhãn thời gian (TSS). TSS sau đó sẽ gắn ngày D và kí (đánh dấu)bộ ba (z,y,D). Công việc này sẽ hoàn hảo miễn là thuật toán kí của TSS an toàn và TSS không thể bị mua chuộc để lùi ngày dãn nhãn của thời gian. (chú ý rằng phương pháp này chỉ được thiết lập khi bob đã kí một bức điện trước một thời gian nào đó. Nếu bob muốn thiết lập cáI anh ta đã kí nó sau ngày nào đó ,anh ta có thể kết hợp thông tin công khai pub nào đó như phương pháp trước đó).

Hình 7.11 :Dán nhãn thời gian lên chữ kí trên bức điện x.

1. Bob tính z = h(x).
2. Bob tính z' = h(z ||pub).
3. Bob tính y = $sig_k(z')$.
4. Bob công bố (z,pub,y) trên tờ báo ra ngày hôm sau.

Nếu như không muốn tin vô điều kiện vào TSS thì có thể tăng độ an toàn lên bằng cách liên kết các thông báo đã dán nhãn thời gian. Trong sơ độ như vậy, bob sẽ gửi một bộ ba được xếp thứ tự (z,x,ID)(Bob) cho TSS. ở đây, z là bản tóm lược thông báo của bức điện x,y là chữ kí của bob trên z ,còn ID(Bob) là thông tin định danh của Bob. TSS sẽ dãn nhãn thời gian một chuỗi bộ ba có dạng này. Kí hiệu (z_n,y_n,ID_n) là bộ ba thứ tự n được TSS dán nhãn thời gian và cho t_n là kí hiệu thời gian lúc thực hiện yêu cầu thứ n.

Hình 7.12: Dán nhãn thời gian (z_n,y_n,ID_n).

1. TSS tính $L_n = (t_{n-1},ID_{n-1},Z_{n-1}y_{n-1},h(L_{n-1}))$
2. TSS tính $C_n = (n, t_n, z_n, ID_n, L_n)$
3. TSS tính $S_n = sig_{TSS}(h(C_n))$
4. TSS gửi (C_n, S_n, ID_n) cho ID_n.

TSS sẽ dán nhãn thời gian lên bộ ba thứ n bằng fthuật toán nêu trên hình 7.12. L_n là "thông tin liên kết để nối yêu cầu thứ n vào yêu cầu trước đó. (L_0 được chọn làm thông tin gia nào đó (được xác định trước đây)để quá trình được bắt đầu)".

Bây giờ nếu được yêu cầu (challenge). Bob có thể để lộ bức điện x_n của mình, và sau đó có thể xác minh y_n. Tiếp theo , các minh chữ kí s_n của TSS. Nếu muốn thì có thể đòi ID_{n-1} hoặc ID_{n+1} để tạo ra nhãn thời gian (C_{n-1}, S_{n-1}, ID_n) và (C_{n+1}, S_{n+1}, ID_{n+2}) tương ứng của chúng. Các chữ kí của TSS có thể được kiểm tra theo nhãn thời gian này. Dĩ nhiên , quá trình này có thể tiếp tục tới mức mong muốn, trước hay sau đó.

7.9.CÁC CHÚ Ý VỀ TÀI LIỆU DẪN

Hàm hash log rời rạc được mô tả trong mục 7.4 là của chaum , van heijst và pfitzmann [CvHP92]. Còn hàm hash có thể chứng mình đwocj là an toàn liễn là hợp số n không thể phân tích thành nhân tư là do gibson [Gib91] đưa ra (bài tập 7.4 có mô tả sơ đồ này).

Cơ sở cho việc m mở rộng hàm hash trong mục 7.5 là của Damgard [DA90] Merkle cũng đưa ra các phương pháp tương tự [ME90].

Các thông tin liên qua tới việc xây dựng các hàm hash dựa trên các hệ thông mã khoá bí mật. Bạn đọc có thể xem trong [PGV94] của Preneel, Govaerts và Vandewalle.

Thuật toán MD4 được đưa ra trong [Ri91] của Rivest, còn tiêu chuẩn hash an toàn được mô tả trong [NBS93]. Tấn công hai trong ba vòng MD4 là của Boer và Bossalaer [DBB92]. Các hàm hash gần đây kể cả N-hash là của [MOI90] và Snefru [ME90A].

Ngoài ra có thể tìm thấy tổng quan về kĩ thuật băm trong Preneel, Govaerts, Vandewalle [PGV93].

Bài tập

CHƯƠNG 7

CÁC HÀM HASH

7.1 CÁC CHỮ KÍ VÀ HÀM HASH.

Bạn đọc có thể thấy rằng các sơ dồ chữ kí trong chương 6 chỉ cho phép kí các bức điện nhỏ.Ví dụ, khi dùng DSS, bức điện 160 bit sẽ được kí bằng chữ kí dài 320 bít. Trên thực tế ta cần các bức điện dài hơn nhiều. Chẳng hạn, một tài liệu về pháp luật có thể dài nhiều Megabyte.

Một cách đơn giản để gải bài toán này là chặt các bức điện dài thành nhiều đoạn 160 bit, sau đó kí lên các đoạn đó độc lập nhau. Điều này cũng tương tự như mã một chuoĩ dài bản rõ bằng cách mã của mỗi kí tự bản rõ độc lập nhau bằng cùng một bản khoá. (Ví dụ: chế độ ECB trong DES).

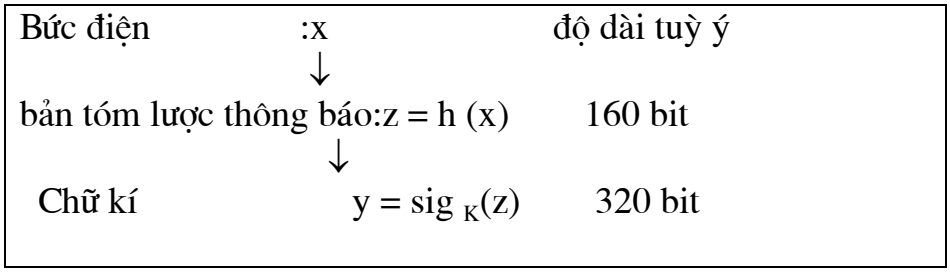
Biện pháp này có một số vấ đề trong việc tạo ra các chữ kí số. Trước hết, với một bức điện dài, ta kết thúc bằng một chữ kí rất lớn (dài gấp đôi bức điện gốc trong trường hợp DSS). Nhược điểm khác là các sơ đồ chữ kí "an toàn" lại chậm vì chúng dùng các pháp số học phức tạp như số mũ modulo. Tuy nhiên, vấn đề nghiêm trọng hơn với phép toán này là búc điện đã kí có thể bị sắp xếp lại các đoạn khác nhau,hoặc một số đoạn trong chúng có thể bị loại bỏ và bức điện nhận được vẫn phải xác minh được. Ta cần bảo vệ sự nguyên vẹn của toàn bộ bức điện và điều này không thể thực hiện được bằng cách kí độc lập từng mẩu nhỏ của chúng.

Giải pháp cho tất cả các vấn đề này là dùng hàm Hash mã khoá công khai nhanh. Hàm này lấy một bức điện có độ dài tuỳ ý và tạo ra một bản tóm lược thông báo có kích thước qui định (160 bit nếu dùng DSS).

Sau đó bản tóm lược thông báo sẽ được kí. Vơi DSS, việc dùng hàm Hash được biểu diễn trê hình 7.1.

Khi Bob muốn kí bức điện x, trước tiên anh ta xây dựng một bnr tóm lược thông báo $z = h(x)$ và sau đó tính $y = sig_K(z)$. Bob truyền cặp (x, y) trên kênh. Xét thấy có thể thực hiện xác minh (bởi ai đó) bằng cách trước hết khôi phục bản tóm lược thông báo $z = h(x)$ bằng hàm h công khai và sau đó kiểm tra xem $ver_k(x,y)$ có = true, hay không.

Hình 7.1.Kí một bản tóm lược thông báo



7.2. HÀM HASH KHÔNG VA CHẠM

Chúng ta cần chú ý rằng,việc dùng hàm hash h không làm giảm sự an toàn của sơ đồ chữ kí vì nó là bản tóm lược thông báo được chữ kí không phải là bức điện. Điều cần thiết đối với h là cần thoả mãn một số tinhs chất nào đó để tranh sự giả mạo.

Kiểu tấn công thông thường nhất là Oscar bắt đầu bằng một bức diện được kí hợp lệ (x, y), $y = sig_K(h(x))$,(Cặp (x, y) là bức điện bất kì được Bob kí trước đó). Sau đó anh ta tính $z = h(x)$ và thử tìm $x \neq x'$ sao cho $h(x') = h(x)$. Nếu Oscar làm được như vậy, (x', y) sẽ là bức điện kí hợp lệ, tức một bức điện giả mạo. Để tránh kiểu tấn công này, h cần thoả mãn tính không va chạm như sau:

Định nghĩa 7.1

Hàm hash h là hàm không va chạm yếu nếu khi cho trước một bức điện x, không thể tiến hành về mặt tính toán để tìm một bức điện $x \neq x'$ sao cho $h(x') = h(x)$.

Một tấn công kiểu khác như sau: Trước hết Oscar tìm hai bức điện $x \neq x'$ sao cho $h(x) = h(x')$. Sau đó Oscar đưa x cho Bob và thyết phục Bob kí bản tóm lược thông báo h(x) để nhận được y. Khi đố (x',y) là thông báo (bức điện) giả mạo hợp lệ.

Đây là lí do đưa ra một tính chất không va chạm khác.

Định nghĩa 7.2.

Hàm Hash h là không va chạm mạnh nếu không có khả năng tính toán để tìm ra bức điênk x và x' sao cho $x \neq x'$ và $h(x) = h(x')$.

Nhận xét rằng: không va chạm mạnh bao hàm va chạm yếu.

Còn đây là kiểu tấn công thứ 3: Như đã nói ở phần 6.2 việc giả mạo các chữ kí trên bản tóm lược thông báo z ngẫu nhiên thường xảy ra với sơ đồ chữ kí. Giả sử Oscar tính chữ kí trên bản tóm lược thông báo z ngẫu nhiên như vậy. Sau đó anh ta tìm x sao cho z= h(x). Nếu làm được như vậy thì (x,y) là bức điện giả mạo hợp lệ. Để tránh được tấn công này, h cần thoả mãn tính chất một chiều (như trong hệ mã khoá công khai và sơ đồ Lamport).

Định nghĩa 7.3.
Hàm Hash h là một chiều nếu khi cho trước một bản tóm lược thông báo z, không thể thực hiện về mặt tính toán để tìm bức điện x sao cho h(x) = z.

Bây giờ ta sẽ chứng minh rằng, tính chất không va chạm mạnh bao hàm tính một chiều bằng phản chứng. Đặc biệt ta sẽ chứng minh rằng, có thể dùng thuật toán đảo với hàm Hash như một chương trình con (giả định) trong thuật toán xác suất Las Vegas để tìm các va chạm.
Sự rút gọn này có thể thực hiện với một giả thiết yếu về kích thước tương đối của vùng và miền (domain and range) của hàm Hash. Ta cũng sẽ giả thiết tiếp là hàm Hash h: $X \to Z$, X,Z là các tập hữu hạn và $|X| \geq 2|Z|$. Đây là giả thiết hợp lí :Nếu xem một phần tử của X được mã như một xâu bít có độ dài $\log_2|X|$ và phần tử của Z được mã hoá như một xâu bít có độ dài $\log_2|X|$ thì bản tóm lược thông báo z = h(x) ít nhất cũng ngắn hơn bức điện x một bít (ta sẽ quan tâm đến tình huống vùng X là vô hạn vì khi đó có thể xem xét các bức điện dài tuỳ ý. Lập luận đó của ta cũng áp dụng cho tình huống này).

Tiếp tục giả thiết là ta có một thuật toán đảo đối với h, nghĩa là có một thuật toán A chấp nhận như đầu vào bản tóm lược thông báo $z \in Z$ và tìm một phần tử $A(z) \in X$ sao cho h(A(z)) = z.

Ta sẽ chứng minh định lí dưới đây:

Định lí 7.1:
Giả sử h: $X \to Z$ là hàm Hash, trong đó $|X|$ và $|Z|$ hữu hạn và $|X| \geq 2|Z|$. Cho A là thuật toán đảo đối với h. Khi đó tồn tại một thuật toán Las Vagas xác suất tìm được một va chạm đối với h với xác suất ít nhất là1/2.

Chứng minh :

Xét thuật toán B đưa ra trong hình 7.2. Rõ ràng B là một thuật toán xác suất kiểu Las Vegas vì nó hoặc tìm thấy một va chạm, hoặc cho câu trả lời không. Vấn đề còn lại là ta phải tịnh xac suất thành công, Với x bất kỳ thuộc X, định nghĩa $x \sim x_1$ nếu $h(x) = h(x_1)$. Dễ thấy rằng, ~ là quan hệ tương đương. Ta định nghĩa:

$$[x] = \{x_1 \in X: x \sim x_1\}$$

Mỗi lớp tương đương [x] chứa ảnh đảo của một phần tử thuộc Z nên số các lớp tương đương nhiều nhất là $|Z|$. Kí hiệu tập các lớp tương đương là C.

Bây giờ giả sử, x là phần tử $\in X$ được chọn trong bước 1. Với giá trị x này, sẽ có $|[x]|$ giá trị x_1 có thể cho phép trở lại bước 3. $|[x]|-1$ các giá trị x_1 này khác với x và như vậy bước 4 thành công. (Chú ý rằng thuật thoán A không biết biểu diễn các lớp tương đương [x] đã chon trong bước 1). Như vậy, khi cho trước lựa chọn cụ thể $x \in X$, xác suất thành công là $(|[x)|-1/|[x]|$.

Hình.7.2 Dùng thuật toán đảo A để tìm các va chạm cho hàm Hash

```
1.chọn một ssó ngẫu nhiên x ∈X
2.Tính z=h(x)
3.Tinh x1= A(Z)
4. if x1 ≠ x then
        x và x1 va chạm dưới h (thành công)
   else
        Quit (sai)
```

Xác suất thành công của thuật toán B bằng trung bình cộng tất cả các lựa chon x có thể:

$$\begin{aligned} P(\text{thành công}) &= (1/|X|)\sum_{x\in X}(|[x]|-1)/|[x]| \\ &= (1/|X|)\sum_{c\in C}\sum_{x\in C}(|c|-1)/|c| \\ &= 1/|X|\sum_{c\in C}(|c|-1) = (1/|X|)\sum_{c\in C}|c| - \sum_{c\in C}1 \\ &>= (|X| - |Z|) / |X| \\ &>= ((|X| - |Z|)/2) / |X| = \tilde{a} \end{aligned}$$

Như vậy, ta đã xây dựng thuật toán Las Vegas có xác suất thành công ít nhất bằng 1/2.

Vì thế, đó là điều kiện đủ để hàm Hash thoả mãn tính chất không va chạm mạnh vì nó bao hàm hai tính chất khác.Phần còn lại của chương này ta chỉ quan tâm đến các hàm Hash không va chạm mạnh.

7.3 TẤN CÔNG NGÀY SINH NHẬT(birthday)

Trong phần này, ta sẽ xác định điều kiện an toàn cần thít ch hàm Hash và điều kiện này chỉ phụ thuộc vào lực lượng của tập Z (tương đương về kích thước của bảng thông báo).Điều kiện cần thiết nà rút ra tư phương pháp tìm kiếm đơn giản ác va chạm mà người ta đã biết đến dưới cái tên tấn công ngày sinh nhật (birthday phương pháparradox), trong bài toán:một nhóm 23 người ngẫu nhiên, có ít nhất 2 người có ngày sinh trùng nhau với xác suất ít nhất là1/2.(Dĩ nhiên, đây chưa phải là nghịch lí,song đó là trực giác đối lập có thể xảy ra). Còn lí do của thuật ngữ "tấn công ngày sinh nhật " sẽ rõ ràng khi ta tiếp tuch trình bày.

Như trước đây, ta hãy giả sử rằng :h:X→Z là hàm Hash, X,Z hữu hạn và $|X| >= 2|Z|$.Định nghĩa $|X| = m$ và $|Z| = n$.Không khó khăn nhận thấy rằng, có ít nhất n va chạm và vấn đề đặt ra là cách tìm chúng. Biện pháp đơn sơ nhất là chọn k phần tử ngẫu nhiên phân biệt $x_1,x_2\ldots..x_k \in X$, tính $z_1 = h(x_1),1<= i <= k$ và sau đó xác định xem liệu có xảy ra va chạm nào không (bằng cách, chẳng hạn như sáp xếp lại các z_i).

Quá trình này tương tự với việc ném k quả bóng vào thùng và sau đó kiểm tra xem liệu có thùng nào chứa ít nhất hai quả hay không (k qủa bóng tương đương với k giá trị x_i ngẫu nhiên và n thùng tương ứng với n phần tử có thể trong Z).

Ta sẽ giới hạn dưới của xác suất tìm thấy một va chạm theo phương pháp này.Do chỉ quan tâm đến giới hạn dưới về xác suất va chạm nên ta sẽ giả sử rằng $|h^{-1}(z)| \approx m/n$ với mọi $z \in Z$. (đây là giả thiết hợp lí :Nếu các ảnh đảo không xấp xỉ bằng nhau thì xác suất tìm thấy một va chạm sẽ tăng lên).

Vì các ảnh đảo đều có kích thước bằng nhau và các $x_{i\ được}$ chọn một cách ngẫu nhiên nên các z_i nhận được có thể xem như các phần tử ngẫu nhiên của Z. Song việc tính toán xác suất để các phần tử ngẫu nhiên $z_1, z_{2,\ldots} z_k \in Z$ là riêng biệt khá đơn giản.Xét các z_i theo thứ tự $z_1, \ldots,z_k$. Phép chọn z_1 đầu tiên là tuỳ ý. Xác suất để $z_2 \neq z_1$ là 1-1/n; xác suất để $z_3 \neq z_1$ và z_2 là 1- 2/n. vv…

Vì thế ta ước lượng xác suất để không có va chạm nào là:

$(1-1/n)(1-2/n)\ldots (1-(k-1/n)) = (1-1/n)$

Nếu x là số thực nhỏ thì 1- x ≈ e^{-x}. Ước lượng này nhận dược từ hai số hạng đầu tiên của cá chuỗi khai triển.

$$e^{-x} = 1 - x + x^2/2! - x^3/3! \ ...$$

Khi đó xác suất không có va chạm nào là :

$$\prod_{i=1}^{k-1}(1-\frac{i}{n}) \approx \prod_{i=1}^{k-1} e^{-1/n} = e^{-k(k-1)/n}$$

Vì thế ta ước lượng xác suất để có ít nhất một va chạm là

$$1-e^{-k(k-1)/n}$$

Nếu kí hiệu xác suất này là ε thì có thể giải phương trình đối với k (như một hàm của n và ε)

$$1-e^{-k(k-1)/n} \approx 1 -\varepsilon$$

$$-k(k-1)/n \approx \ln(1-\varepsilon)$$

$$k^2 - k \approx n\ln 1/(1-\varepsilon)$$

Nếu bỏ qua số hạng k thì :

$$k= \sqrt{n ln\frac{1}{1-\varepsilon}}$$

Nếu lấy ε = 0.5 thì

$$k \approx 1.17\sqrt{n}$$

Điều này nói lên rằng, việc chặt (băm) trên $\sqrt{n}$ phần tử ngẫu nhiên của X sẽ tạo ra một va chạm với xác suấtt 50%. Chú ý rằng, cách chọn ε khác sẽ dẫn đến hệ số hằng số khác song k vẫn tỷ lên với $\sqrt{n}$.

Nếu X là tập người,Y là tập gồm 365 ngỳ trong năm (không nhuận tức tháng 2 có 29 ngày) còn h(x) là ngày sinh nhật của x, khi đó ta sẽ giả guyết bằng nhgịch lý ngày sinh nhật. Lấy n = 365, ta nhận được k ≈ 22,3. Vì vậy, như đã nêu ở trên, sẽ có ít nhất 2 người có ngày sinh nhật trùng nhau trong 23 người ngẫu nhiên với xác suất ít nhất bằng 1/2.

Tấn công ngày sonh nhật đặt giới hạn cho các kích thước các bản tóm lược thông báo. bản tóm lược thông báo 40 bit sẽ không an toàn vì có thể tìm thấy một va chạm với xác suất 1/2 trên 2^{20} (khoảng1.000.000)đoạn chặt ngẫu nhiên. Từ đây cho thấy rằng, kích thước tối thiểu chấp nhận được của bản tóm lược thông báo là 128 bit (tấn công ngày sinh nhật cần trên 2^{64} đoạn chặt trong trường hợp này). Đó chính là lý do chọn bản tóm lược thông báo dài 160 bit trong sơ đồ DSS.

Hình7.3. Hàm hash chaum-Van heyst-Plitzmann.

Giả sử p là số nguyên tố lớn và q =(p-1)/2 cũng là số nguyên tố. Cho α và β là hai phần tử nguyên thuỷ của Zp. Giá trị $\log_\alpha\beta$ không công khai và giả sử rằng không có khả năng tính toán được giá trị của nó.
Hàm Hash:

$h: \{0,...,q-1\}\times\{0,...,q-1\} \rightarrow Zp\setminus\{0\}$

được định nghĩa như sau:

$h(x1,x2) = \alpha^{x_1}\beta^{x_2} \bmod p$

7.3. hàm hash logarithm rời rạc

Trong phần này ta sẽ mô tả một hàm Hash do Chaum-Van Heyst và Pfĩtmann đưa ra. Hàm này an toàn do không thể tính được logarithm rời rạc. Hàm Hast này không đủ nhanh để dùng trong thực tế song nó đơn giản và cho một ví dụ tốt về một hàm Hash có thể an toàn dưới giả thuyết tính toán hợp lý nào số. Hàm Hash Caum-Van Heyst- Pfĩtmann được nêt trong hình 7.3. Sau đây sẽ chứng minh một định lý liên quan đến sự an toàn của hàm Hast này.

Định lý 7.2.

N*ếu cho trước một va chạm với hàm Hash Chaum-Van Heyst-Pfĩtmann h có thể tính được logarithm rời rạc $\log_\alpha\beta$ một cách có hiệu quả.*

Chứng minh

Giả sử cho trước va chạm

$h(x_1,x_2) = h(x_3,x_4)$

trong đó $(x_1,x_2) \neq (x_3,x_4)$. Như vậy ta có đồng dư thức sau:

$\alpha^{x_1}\beta^{x_2} = \alpha^{x_3}\beta^{x_4}$

hay

$\alpha^{x_1}\beta^{x_2} \equiv \alpha^{x_3}\beta^{x_4} \pmod p$

Ta kí hiệu

$D = UCLN\ (x_4-x_2,p-1)$

Vì p-1 =2q ,q là số nguyên tố nên d ∈ {1, 2, q, p-1}. Vì thế, ta có 4 xác suất với d sẽ xem xét lần lượt dwois đây.

Trước hết ,giả sử d =1 ,khi đó cho

$$y= (x_4-x_2)^{-1} \bmod (p-1)$$

ta có

$$\beta \equiv \beta^{(x_4-x_2)y} (\bmod p)$$
$$\equiv \alpha^{(x_1-x_2)y} (\bmod p)$$

Vì thế, có thể tính loarithm rời rạc $\log_\alpha \beta$ như sau:

$$\log_\alpha \beta = (x_1-x_3)\ (x_4-x_2)^{-1} \bmod (p-1)$$

Tiếp theo, giả sử d=2. Vì p-1 =2q, lẻ nên UCLN(x_4-x_2,q) =1. Giả sử:

$$y=(x_4-x_2)^{-1} \bmod q$$

xét thấy $(x_4-x_2)y = kq+1$

với số nguyên k nào đó. Vì thế ta có:

$$\beta^{(x_4-x_2)y} \equiv \beta^{kq+1} (\bmod p)$$
$$\equiv (-1)^k \beta (\bmod p)$$
$$\equiv \pm \beta (\bmod p)$$

Vì $\beta^q \equiv -1 (\bmod p)$

Nên

$$\alpha^{(x4-x2)y} \equiv \beta^{(x1-x3)} (\bmod p)$$
$$\equiv \pm \beta (\bmod p)$$

Từ đó suy ra rằng:

$$\log_\alpha \beta = (x_1-x_3)y \bmod (p-1)$$
$$\log_\alpha \beta = (x_1-x_3)y \bmod (p-1)$$

Ta có thể dễ dàng kiểm tra thấy một trong hai xác suất trên là đúng. Vì thế như trong trường hợp d =1, ta tính được $\log_\alpha \beta$.

Xác suất tiếp theo là d = q. Tuy nhiên

$$q-1 \geq x_1 \geq 0$$

và

$$q-1 \geq x_3 \geq 0$$

nên

$$(q-1) \geq x_4-x_2 \geq -(q-1)$$

do vậy UCLN(x_4-x_2,p-1) không thể bằng q, nói cách khác trường hợp này không xảy ra.

Xác suất cuối cùng là d = p-1. Điều nàychỉ xảy ra khi x2 =x4. Song khi đó ta có

$$\alpha^{x_1}\beta^{x_2} \equiv \alpha^{x_3}\beta^{x_4} (\bmod p)$$

nên $\alpha^{x_1} \equiv \alpha^{x_3} \pmod p$

và $x_1 = x_2$. Như vậy $(x_1,x_2) = (x_3,x_4) \Rightarrow$ mâu thuẫn. Như vậy trường hợp này cũng không thể có.

Vì ta đã xem xét tất cả các giá trị có thể đối với d nên có thể kết luận rằng ,hàm Hash h là không va chạm mạnh miễn là không thể tính được logarithm rời rạc $\log_\alpha\beta$ trong Z_p.

Ta sẽ minh hoạ lý thuyết nêu trên bằng một ví dụ.

Ví dụ 7.1

Giả sử p =12347 (vì thế q = 6173), $\alpha = 2$, $\beta = 8461$. Giả sử ta được đưa trước một va chạm

$$\alpha^{5692}\ \beta^{144} \equiv \alpha^{212}\ \beta^{4214} \pmod{12347}$$

Như vậy $x_1 = 5692$, $x_2 = 144$, $x_3 = 212$, $x_4 = 4214$. Xét thấy UCLN $(x_4 - x_2, p-1) = 2$ nên ta bắt đầu bằng việc tính

$$y = (x_4 - x_2)^{-1} \bmod q$$
$$= (4214 - 144)^{-1} \bmod 6173 = 4312$$

Tiếp theo tính

$$y = (x_1 - x_3) \bmod (p-1)$$
$$= (5692 - 212)\ 4312 \bmod 12346$$
$$= 11862$$

Xét thấy đó là trường hợp mà $\log_\alpha\beta \in \{y', y'+q \bmod (p-1)\}$. Vì

$$\alpha^y \bmod p = 2^{12346} = 9998$$

nên ta kết luận rằng:

$$\log_\alpha\beta = y' + q \bmod (p-1)$$
$$= 11862 + 6173 \bmod 12346$$
$$= 5689$$

như phép kiểm tra, ta có thể xác minh thấy rằng

$$2^{5689} = 8461 \pmod{12347}$$

Vì thế , ta các định được $\log_\alpha\beta$.

7.5.**các hàm hash mở rộng**

Cho đến lúc này, ta đã xét các hàm Hash trong vùng hữu hạn. Bây giờ ta nghiên xéu cách có thể mở rộng một hàm Hash không va chạm mạnh từ vùng hữu hạn sang vùng vô hạn. Điều này cho phép ký các bức điện có độ dài tuỳ ý. Gỉa sử h: $(Z_2)^m \rightarrow (Z_2)^t$ là một hàm hash không va chạm mạnh ,trong đó $m \geq t-1$. Ta sẽ dùng h đêu xây dựng hàm hash không va chạm mạnh h: $X \rightarrow (Z_2)^t$ trong đó

$$X = \bigcup_{i=m}^{\infty} (Z_2)^t$$

Trước tiên xét trường hợp m ≥ t+2.

Ta sẽ xem các phần tử của X như các xây bit. |x| chỉ độ dàI của x (tức số các bit trong x) và x||y ký hiệu sự kết hợp các xây x và y. Giả sử |x| = n > m. Có thể biểu thị x như một chuỗi kết hợp.

$$X = x_1||x_2||...||x_k$$

Trong đó

$$|x_1| = |x_2| = ... = |x_{k-1}| = m- t-1$$

và $|x_k| = m- t- 1- d$

Hình 7.4. Mở rộng hàm hash h thành h* (m ≥t+2)

1. For i= 1 to k-1 do
   $y_i = x_i$
2. $y_k = x_k ||0^d$
3. cho y_{k+1} là biểu diễn nhị phân của d
4. $g_i = h(0I+1||y_1)$
5. for i=1 to k do
   $g_i+1 = h(g_i||1||y_i+1)$
6. $h^*(x) = g_k +1$

Trong đó m- t- 2 ≥ d ≥0. Vì thế ta có

$$k= \left\lceil \frac{n}{m-t-1} \right\rceil$$

Ta định nghĩa h*(x) theo thuật toán biểu kiễn trong hình 7.4.

Kí hiệu $y(x) = y1||y2||...||y_{k-1}$

Nhận xét rằng y_k được lập từ x_k bằng cách chèn thêm d số 0 vào bên phảI để tất cả các khối y_i (k ≥ i ≥ 1)đều có chiều dàI m-t-1. Cũng như trong bước 3 y_{k+1} sẽ được đệm thêm về bên tráI các số 0 sao cho $|y_{k+1}| = m-t-1$.

Để băm nhỏ x ,trước hết ta xây dựng hàm y(x) và sau đó "chế biến" các khối $y_1...y_{k+1}$ theo một khuôn mẫu cụ thể. Điều quan trọng là y(x) ≠y(x') khi

x≠x. Thực tế y_{k+1} được định nghĩa theo cách các phép ánh xạ x → y(x)là một đơn ánh.

Định lý sau đây chứng minh rằng h* là an toàn khi h an toàn.

Định lý 7.3

Giả sử h: $(Z_2)^n \to (Z_2)$ là hàm hash không va chạm mạnhm≥ t+2. Khi đó hàm h: $\cup_{i=m}^{\infty}$ $(Z_2)^t \to (Z_2)^t$ được xây dựng như trên hình 7.4 là hàm hash không*
và chạm mạnh.

Chứng minh:

Giả sử rằng ,ta có thể tìm được x ≠x' sao cho h*(x) = h*(x'). Nếu cho trước một cặp như vậy, ta sẽ chỉ ra cách có thể tìm được một va chạm đối với h trong thời gian đa thức. Vì h được giả thiết là không va chạm mạnh nên dẫn đến một mâu thuẫn như vậy h sẽ được chứng minh là không va chạm mạnh.

Kí hiệu $y(x)= y_1||..||y_{k+1}$

Và $y(x') = y_1'||...||y_{k+1}'$

ở đây x và x' được đệm thêm d và d' số 0 tương ứng trong bước 2. Kí hiệu tiếp các giá trị được tính trong các bước 4 và 5 là $g_1,g_2....,g_{k+1}$ và $g_1',.....,g_{k+1}'$ tương ứng.

Chúng ta sẽ đồng nhất hai trường hợp tuỳ thuộc vào việc có hay không |x| ≡|x'| (mod m-t-1).

Trường hợp1: |x| ≠|x'| (mod m-t-1)

Tại đây d ≠d' và $y_{k+1} \neq y'_{k+1}$. Ta có:

$$\begin{aligned} H(g_k||1||y_{k+1}) &= g_{k+1} \\ &= h^*(x) \\ &= h^*(x') \\ &= g'_{t+1} \\ &= h(g'_{t+1}||1||y'_{t+1)} \end{aligned}$$

là một va chạm đối với h vì $y_{k+1} \neq y'_{k+1.}$

Trường hợp2: |x| ≡|x'| (mod m-t-1)

Ta chia trường hợp này thành hai trường hợp con:

Trường hợp 2a: |x| = |x'|.

Tạ đây ta có k= l và $y_{k+1} = y'_{k+1}$. Ta vắt đầu như trong trường hợp 1:

$$h(g_k||1||y_{k+1}) = g_{k+1}$$
$$= h^*(x)$$
$$= h^*(x')$$
$$= h(g'_k||1||y'_k+1)$$

Nếu $g_k = g'_k$ thì ta tìm thấy một va chạm đối với h, vì thế giả sử $g_k = g'_k$ khi đó ta sẽ có:

$$h(g_{k-1}||1||y_k) = g_k$$
$$=g'_k$$
$$=h(0^{i+1}||y_1)$$

Hoặc là tìm thấy một va chạm đối với h hoặc $g_{k-1} =g'_{k}-_1$ và $y_k = y'_k$. Giả sử không tìm thấy va chạm nào ,ta tiếp tục thực hiện ngược các bước cho đến khi cuối cùng nhận được :

$$h(0_{i+1}||y_1) = g_1$$
$$=g'_{i-k+1}$$
$$=g(g'_{i-k}||1||y'_{i-k+1}).$$

Nhưng bit thứ (t+1) của $0^{i+1}||y_1$ bằng 0 và bit thứ (t+1) của $g'_{i-k+1}||1||y'_{i-k+1}$ bằng 1. Vì thế ta tịm thấy một va chạm đối với h.

Vì đã xét hết các trường hợp có thể nên ta có kết luận mong muốn.

Cấu trúc của hình 7.4 chỉ được dùng khi m>= t+2. Bây giờ ta hãy xem xét tình huống trong đó m = t+1. Cần dùng một cấu trúc khác cho h. Như trước đây, giả sử |x|=n>m. Trước hết ta mã x theo cách đặc biệt. Cách này dùng hàm f có định nghĩa như sau:

f(0) = 0
f(1) = 01

Thuật toán để xây dựng h*(x)được miêu tả trong hình 7.5

Phép mã $x \rightarrow y = y(x)$ được định nghĩa trong vước 1 thoả mãn hai tính chất quan trọng sau:

1. nếu x ≠x' thì y(x)≠ y(x') (tức là x→ y(x) là một đơn ánh)
2. Không tồn tạI hai chuỗi x≠ x' và chuỗi z sao cho y(x)= z||y(x'). Nói cách khác không cho phép mã hoá nào là fpsstix của phép mã khác. ĐIều này dễ dàng thấy được do chuỗi y(x) bắt đầu bằng 11 và không tồn tạI hai số 1 liên tiếp trong phần còn lạI của chuỗi).

Hình 7.5 Mở rộng hàm hash h thành h* (m = t+1)

Định lý 7

1. Giả sử $y = y_1y_2...y_k = 11||f(x_1)||....||f(x_n)$
2. $g_1 = h(0^1||y_1)$
3. for i=1 to k-1 do
   $g_{i+1} = h(g_i||y_{i+1})$
4. $h^*(x) = g_k$

Giả sử h: $(Z_2)^n \rightarrow (Z_2)$ là hàm hash không va chạm mạnh. Khi đó hàm h: $\cup_{i=m}^{\infty}$ $(Z_2)^t \rightarrow (Z_2)^t$ được xây dựng như trên hình 7.5 là hàm hash không va chạm mạnh.*

Chứng minh:

Giả sử rằng ta có thể tìm được x ≠x' sao cho h*(x)=h*(x'). Kí hiệu:

$$y(x) = y_1y_2....y_k$$

và $$y(x') = y'_1y'_2....y'_l$$

Ta xét hai trường hợp:

Trường hợp 1: k=l

Như trong định lý 7.3 hoặc ta tìm thấy một va chạm đỗi với h hoặc ta nhận được y = y' song đIều này lạI bao hàm x = x', dẫn đến mâu thuẫn.

Trường hợp2: k≠ l

Không mất tính tổng quát ,giả sử l>k . trường hợp này xử lý theo kiểu tương tự. Nếu giả thiết ta không tìm thấy va chạm nào đối với h ,ta có dãy các phương trình sau:

$y_k = y'_l$
$y_{k-1} = y'_{l-1}$
.....
$y_1 = y'_{l-k+1}$

Song đIều này mâu thuẫn với tính chất "không posfixx" nêu ở trên. Từ đây ta kết luận rằng h* là hạm không va chạm.

Ta sẽ tổng kết hoá hai xây dựng trong phần này và số các ứng dụng của h cần thiết để tính h* theo định lý sau:

Định lý 7.5

Giả sử h: $(Z_2)^n \to (Z_2)$ là hàm hash không va chạm mạnh,ở đây m>=t+1. Khi đó tồn tạI hàm không va chạm mạnh

$$h^*: \cup_{i=m}^{\infty} (Z_2)^t \to (Z_2)^t$$

Số lần h được tính trong ước lượng h nhiều nhất bằng :*

$$l + \left\lceil \frac{n}{m-t-1} \right\rceil \text{ nếu } m>=t+2$$

$2n+2$ *nếu* $m= t+2$

trong đó $|x|=n$.

7.6 các hàm hash dựa trên các hệ mật

Cho đến nay, các phương pháp đã mô tả để đưa đến nhứng hàm hash hầu như đều rất chậm đối với các ứng dụng thực tiễn. Một biện pháp khác là dùng các hệ thống mã hoá bí mật hiện có để xây dừng các hàm hash. Giả sử rằng (P,C,K,E,D) là một hệ thống mật mã an toàn về mặt tính toán. Để thuận tiện ta cũng giả thiết rằng $P = C = K = (Z_2)^n$.ở đâychọn n>=128 để xây ngăn chặn kiểu tấn công ngày sinh nhật. ĐIều này loạI trừ việc dùng DES (vì độ dài khoá của DES khác với độ dài bản rõ).

Giả sử cho trước một xâu bit:

x= x1||x2||....||xk

trong đó xi $\in$ $(Z_2)^n$, 1≤ i ≤ (nếu số bit trong x không phải là bội của n thì cần chèn thêm vào x theo cách nào đó. Chẳng hạn như cách làm trong nục 7.5. Để đơn giản ta sẽ bỏ qua đIểm này).

Ý tưởng cơ bản là bắt đầu bằng một "giá trị ban đầu" cố định g_0 =IV và sau đó ta xây dựng g_1,...,g_k theo quy tắc thiết lập :

$$g_i = f(x_i, g_{i-1}).$$

ở đây f là hàm kết hợp toàn bộ các phép mã hoá của hệ mật được dùng. Cuối cùng ta định nghĩa bản tóm lược của thông báo h(x) =g_k.

Vài hàm hash kiểu này đã được đề xuất và nhiều loại trong chúng tỏ ra không an toàn (không phụ thuộc vào việc liệu hệ mật cơ bản có an toàn hay không). Tuy nhiên , có 4 phương án khác nhau có vẻ an toàn của sơ đồ này :

$$g_i = e_{g_{i-1}}(x_i) \oplus x_i$$
$$g_i = e_{g_{i-1}}(x_i) \oplus x_I \oplus g_{i-1}$$
$$g_i = e_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_I$$
$$g_i = e_{g_{i-1}}(x_i \oplus g_{i-1}) \oplus x_I \oplus g_{i-1.}$$

7.7 **Hàm hash MD4.**

Hàm hash MD4 được Riverst đề xuất năm 1990 và một hiên bản mạnh là MD5 cũng được đưa ra năm 1991. Chuẩn hàm hash an toàn (hay SHS) phức tạp hơn song cũng dưa tên các phương pháp tương tự. Nó được công bố trong hồ sơ liên bang năm 1992 và được chấp nhận làm tiêu chuẩn vào ngày 11/5/1993. Tất cả các hàm hash trên đều rất nhanh nên trên thực tế chúng dùng để kí các bức điện dài.

Trong phần này sẽ mô tả chi tiết MD4 và thảo luận một số cảI tiến dùng trong MD5 và SHS.

Cho trước một xâu bit trước hết ta tạo một mạng:

M = M[0] M[1]... M[N-1] .

trong đó M[i] là xâu bit có độ dàI 32 và N ≡ 0 mod 16. Ta sẽ gọi M[i] là *từ*. M được xây dựng từ x bằng thuật toán trong hình 7.6.

Hình 7.6 Xây dựng M trong MD4

1. d = 447-(|x| mod 512)
2. giả sử ℓ là kí hiệu biểu diễn nhị phân của |x| mod 2^{64}.|l| = 64
3. M = x||1||0^d||ℓ

Trong việc xây dựng M, ta gắn số 1 ssơn lẻ vào x, sau đó sẽ gài thêm các số 0 đủ để độ dài trở nên đồng dư với 448 modulo 512.,cuối cùng nối thêm 64 bit chưa biểu diễn nhị phân về độ dàI (ban đầu) của x(được rút gọn theo móulo 2^{64} nếu cần). Xâu kết quả M có độ dàI chia hết cho 512. Vì thế khi chặt M thành các từ 32 bit , số từ nhận được là N-sẽ chia hết cho 16.

Bây giờ, tiếp tục xây dựng bản tóm lược thông báo 128 bit. Hình 7.7 đưa ra mô tả thuật toán ở mức cao. Bản tóm lược thông báo được xây dựng như sự kết nối 4 từ A,B,C và D mà ta sẽ gọi là các thanh ghi. Bốn thanh ghi được khởi động như trong bước 1. Tiếp theo ta xử lí bảng M 16 bit từ cùng lúc. Trong mỗi vòng lặp ở bước 2, đầu tiên lấy 16 từ "tiếp theo" của M và lưu chúng trong bảng X (bước 3). Các giá trị của bốn thanh ghi dịch sau đó sẽ được lưu lại (bước 4). Sau đó ta sẽ thực hiện ba vòng "băm" (hash). Mỗi vaòng gồm một phép toán thực hiện trên một trong 16 từ trong X. 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 update (cập nhật) trong bước 8 bằng cách cộng ngược các giá trị lưu trước đó trong bước 4. Phép cộng này được xác định là cộng các số nguyên dương ,được rút gọn theo modelo 2^{32}.

Ba vòng trong MD4 là khác nhau (không giông như DES. 16 vòng đều như nhau). Trước hết ta sẽ mô tả vàI phép toán khác nhau trong ba vòng này. Trong phần sau,ta kí hiệu X và Y là các từ đầu vào và mỗi phép toán sẽ tạo ra một từ đầu ra. Dưới đây là phép toán được dùng:

X$\wedge$Y là phép "AND" theo bit giữa X và Y
X$\vee$Y là phép "OR" theo bit giữa X và Y
X$\oplus$Y là phép "XOR" theo bit giữa X và Y
$\neg$X chỉ phần bù của X
X+Y là phép cộng theo modulo 2^{32}.
X<< s phép dịch vòng tráI X đI s vị trí (31>= s >=0).

Chú ý rằng, tất cả các phép toán trên đều tất nhanh và chỉ có phép số học duy nhất được dùng là phép cộng modulo 2^{32}. Nếu MD4 được ứng dụng thì cần tính đến kiến trúc cơ bản của máy tính mà nó chạy trên đó để thực hiện

chính xác phép cộng. Giả sử $a_1a_2a_3a_4$ là 4 byte trong từ xem mỗi a_i,như một số nguyên trong dảI 0-255 được biểu diễn dưới dạng nhị phân. Trong kiến trúc kiểu endian lớn (chẳng hạn như trên trạm Sunsparc) từ này biểu diễn số nguyên.

$$a_1 2^{24} + a_2 2^{16} + a_3 2^8 + a_4$$

Trong kiến trúc kiểu endian nhỏ (chẳng hạn họ intel 80xxx). Từ này biểu diễn số nguyên:

$$a_4 2^{24} + + a_3\ 2^{16} + a_2\ 2^8 + a_1$$

MD4 giả thiết dùng kiến trúc kiểu endian nhỏ. ĐIều quan trọng là bản tóm lược thông báo độc lập với kiến trúc cơ bản. Vì thể nếu muốn chạy MD4 trên máy tính endian lớn cần thực hiện phép cộng X+Y như sau:

1. Trao đổi x_1 và x_4; x_2 và x_3; y_1 và y_4; y_2 và y_3
2. Tính $Z = X+Y \bmod 2^{32}$
3. Trao đổi z_1 và z_4 ; z_2 và z_3.

Hình 7.7 hàm hash MD4

1. A= 67452301 (hệ hexa)
   B = efcdab89 (hệ hexa)
   C = 98badcfe (hệ hexa)
   D = 10325476 (hệ hexa)
2. for i = 0 to N/16-1 do
3. for i = 1 to 15 do
   X[i] = M[16i+j]
4. AA = A
   BB = B
   CC = C
   DD = D
5. round1
6. round2
7. round3
8. A = A+AA
   B = B+ BB
   C = C + CC
   D = D + DD

Các vòng 1, 2 và 3 của MD4 dùng tương ứng ba hàm f, g, và h. Mỗi hàm này là một hàm boolean tính theo bit dùng 2 từ làm đầu vào và tạo ra một từ tại đẩu ra. Chúng được xác định như sau:

$f(X,Y,Z) = (X \wedge Y) \vee((-X) \wedge Z)$

$g(X,Y,Z) = (X \wedge Y) \vee(X \wedge Z) \vee(Y \wedge Z)$

$h(X,Y,Z) = X \oplus Y \oplus Z$

Các hình 7.8-7.10 sẽ mô tả đầy đủ các vòng 1,2 và 3 của MD4.

MD4 được thiết kế chạy rất nhanh và quả thực phần mềm chạy trên máy Sun SPARC có tốc độ 1.4 Mbyte/s. Mặt khác, khó có thể nói đIều gì cụ thể về độ mật của hàm hash, chẳng hạn như MD4 vì nó không dựa trên vàI toán khó đã nghiên cứu kĩ (ví dụ như phân tích nhân tử trên bàI toán logarithm rời rạc). Vì thế trong trường hợp Dé sự tin cậy vào độ an toàn của hệ thống chỉ có thể đạt được về thời gian và như vậy có thể hi vọng hệ thống vừa được nghiên cứu và không tìm thấy sự không an toàn nào.

Hình 7.8 : Vòng 1 của MD4 .(round 1)

```
1.   A = (A+ f(B,C,D) + X[0]) << 3
2.   D = (D + f(A,B,C) +X[1]) << 7
3.   C = (C + f(D,A,C) +X[2]) << 11
4.   B = (B + f(C,D,A) +X[3]) << 19
5.   A = (A + f(B,C,D) +X[4]) << 3
6.   D = (D + f(A,B,C) +X[5]) << 7
7.   C = (C + f(D,A,C) +X[6]) << 11
8.   B = (B + f(C,D,A) +X[7]) << 19
9.   A = (A + f(B,C,D) +X[8]) << 3
10.  D = (D + f(A,B,C) +X[9]) << 7
11.  C = (C + f(D,A,C) +X[10]) << 11
12.  B = (B + f(C,D,A) +X[11]) << 19
13.  A = (A + f(B,C,D) +X[12]) << 3
14.  D = (D + f(A,B,C) +X[13]) << 7
15.  C = (C + f(D,A,C) +X[14]) << 11
16.  B = (B + f(C,D,A) +X[15]) << 19
```

Mặc dù MD4 vẫn chưa bị phá song các phiên bản yếu cho phép bỏ qua hoặc vòng thứ nhất hoặc thứ ba dều có thể bị phá không khó khăn gì. nghĩa là dễ dàng tìn thấy các va chạm đối với các phiên bản chỉ có hai vòng. Phiên vản mạnh của MD5 là MD5 được công bố năm 1991. MD5 dùng vòng thay cho ba và chậm hơn 30% so với MD4 (khoảng 0.9 Mbyte/s trên cùng máy).

Chuẩn hàm hash an toàn phức tạp và chậm hơn. Ta sẽ không mô tả đầy đủ song sẽ chỉ ra một vàI cảI tiến trên nó.

1. SHS được thiết kế để chạy trên máy kiến trúc endian lớn hơn là trên máy endian nhỏ.
2. SHA tạo ra các bản tóm lược thông báo 5 thanh ghi (160 bit).
3. SHS xử lí 16 từ của bức đIện cùng một lúc như MD4. Tuy nhiên, 16 từ trước tiên được ”mở rộng” thành 80 từ ,sau đó thực hiện một dãy 80 phép tính ,mỗi phép tính trên một từ.
   Hình 7.9 Vòng 2 củaMD4.

```
1.   A = (A +g(B,C,D) + X[0] + 5A827999) <<3
2.   D = (D +g(A,B,C) + X[4] + 5A827999) <<5
3.   C = (C +g(D,A,B) + X[8] + 5A827999) <<9
4.   B = (B +g(C,D,A) + X[12] + 5A827999) <<13
5.   A = (A +g(B,C,D) + X[1] + 5A827999) <<3
6.   D = (D +g(A,B,C) + X[1] + 5A827999) <<5
7.   C = (C +g(D,A,B) + X[5] + 5A827999) <<9
8.   B = (B +g(C,D,A) + X[13] + 5A827999) <<13
9.   A = (A +g(B,C,D) + X[2] + 5A827999) <<3
10.  D = (D +g(A,B,C) + X[6] + 5A827999) <<5
11.  C = (C +g(D,A,B) + X[10] + 5A827999) <<9
12.  B = (B +g(C,D,A) + X[14] + 5A827999) <<13
13.  A = (A +g(B,C,D) + X[3] + 5A827999) <<3
14.  D = (D +g(A,B,C) + X[7] + 5A827999) <<5
15.  C = (C +g(D,A,B) + X[11] + 5A827999) <<9
16.  B = (B +g(C,D,A) + X[15] + 5A827999) <<13
```

Dùng hàm mở rộng sau đây: Cho trước 16 từ X[0] ...X[15], ta tính thêm 64 từ nữa theo quan hệ đệ quy.

X[j] = X[j-3]⊕ X[j-8]⊕ X[j-14]⊕ X[j-16], 79 ≥ j ≥ 16 7.1

Kết quả của phương trình (7.1) là mỗi một trong các từ X[16]...X[79] được thiết lập bằng cách cộng ⊕ với một tập con xác định nào đó của các từ X[0]...X[15].

Ví dụ: Ta có:

X[16] = X[0] ⊕X[2]⊕X[8]⊕X[13]
X[17] = X[1] ⊕X[3]⊕X[9]⊕X[14]
X[18] = X[2] ⊕X[4]⊕X[10]⊕X[15]
X[19] = X[0] ⊕X[2]⊕X[3]⊕X[5]⊕X[8]⊕X[11]⊕X[13]
X[79] = X[1] ⊕X[4]⊕X[15]⊕X[8]⊕X[12]⊕X[13].

Một đề xuất đòi hỏi sửa lại SHS liên quan đến hàm mở rộng trong đó đề nghị đặt lại phương trình 7.1 như sau:

X[j] = X[j-3] ⊕X[j-8]⊕X[j-14]⊕X[j-16] <<1; 79≥ j ≥ 16 (7.2)

Hình 7.10 : Vòng ba của MD4.

```
1.  A = (A + h(B,C,D) + X[0] + 6ED9EBA1) <<3
2.  D = (D + h(A,B,C) + X[8] + 6ED9EBA1) <<9
3.  C = (C + h(D,A,B) + X[4] + 6ED9EBA1) << 11
4.  B = (B + h(C,D,A) + X[12] + 6ED9EBA1) << 15
5.  A = (A + h(B,C,D) + X[2] + 6ED9EBA1) <<3
6.  D = (D + h(A,B,C) + X[10] + 6ED9EBA1) <<9
7.  C = (C + h(D,A,B) + X[6] + 6ED9EBA1) << 11
8.  B = (B + h(C,D,A) + X[14] + 6ED9EBA1) << 15
9.  A = (A + h(B,C,D) + X[1] + 6ED9EBA1) <<3
10. D = (D + h(A,B,C) + X[9] + 6ED9EBA1) <<9
11. C = (C + h(D,A,B) + X[13] + 6ED9EBA1) << 11
12. B = (B + h(C,D,A) + X[13] + 6ED9EBA1) << 15
13. A = (A + h(B,C,D) + X[3] + 6ED9EBA1) <<3
14. D = (D + h(A,B,C) + X[11] + 6ED9EBA1) <<9
15. C = (C + h(D,A,B) + X[7] + 6ED9EBA1) << 11
16. B = (B + h(C,D,A) + X[15] + 6ED9EBA1) << 15
```

Như trước đây , toán tử ''<<1'' là phép dịch vòng trái một vị trí.

7.8 **nhãn thời gian (timestamping)**.

Một khó khăn trong sơ đồ chữ kí là thuật toán kí có thể bị tổn thương. Chẳng hạn, giả sử Oscar có khả năng xác định số mũ mật a của Bob trên bất kì bức điện nào mà anh ta muốn. Song còn vấn đề khác (có thể nghiêm trọng hơn)là : từ đây người ta sẽ đặt câu hỏi về tính xác thực của tất cả các bức điện mà Bob kí, kể cả những bức điện mà anh ta kí trước khi Oscar đánh cắp được thuật toán.

Từ đây lại có thể nảy sinh tình huống không mong muốn khác : giả sử Bob kí một bức điện và sau đó từ chối là đã không kí nó. Bob có thể công khai thuật toán kí của mình sau đó công bố rằng chữ kí của anh ta trên bức điện đang nói trên là giả mạo.

Lí do có các kiểu sự kiện này là do không có các nào các định bức điện được kí khi nào. Nhãn thời gian có thể cung cấp bằng chứng rằng, bức điện đã được kí vào thời điểm cụ thể nào đó. Khì đó nế thuật toán kí của Bob có nhược điểm (bị tổn thương) thì bất kì chữ kí nào anh ta kí trước đó sẽ không còn hợp lệ. ĐIều này giống với kiểu thực hiên các thẻ tín dụng: Nếu ai đó làm mất thẻ tín dụng và thông báo cho nhà băng đã phát hành thì thẻ mất hiệu lực. Song các cuộc mua bán thực hiện trước khi mất nó thì vẫn không bị ảnh hưởng.

Trong phần này sẽ mô tả một vàI phương pháp gắn nhãn thời gian. Trước hết,nhận xét rằng, Bob có thể tạo ra một nhãn thời gian có sức thuyết phục trên chữ kí của anh ta. Đầu tiên, Bob nhận được một thông tin "hiện thời " có sẵn công khai nào đó, thông tin này không thể dự đoán được trước khi nó xảy ra. Ví dụ thông tin chứa tất cả các lợi thế về môn bóng chày của các liên minh chính từ ngày trước đó, hay các giá trị của tất cả cổ phần đwocj lên danh sách trong sở giao dịch chứng khoán NewYork. Ta kí hiệu thông tin này bằng chữ *pub*.

Bây giờ giả sử Bob muốn dán nhãn thời gian trên chữ kí của mình trên bức điện x. Giả thiết rằng, h là hàm hash công khai biết trước. Bob sẽ thực hiện theo thuật toán trong hình 7.11.Sau đây là cách sơ đồ làm việc : sự có mặt của thông tin pub có nghĩa là bob không thể tạo ra được y trước ngày đang nói đến. Còn một thực tế là y công bố trong một tờ báo ra ngày tiếp theo chứng tỏ

rằng bob đã không tính y sau ngày được nói đến. Vì thế chữ kí y của bob bị hạn chế trong thời hạn một ngày. Cũng nhận xét thấy rằng, bob không để lộ bức điện x trong sơ đồ này vì chỉ có x được công bố ... Nếu cần bob có thể chứng minh rằng x là bức điện mà anh ta đã kí và dán nhãn thời gian một cách đơn giản là làm lộ nó.

Cũng không khó khăn tạo ra tạo ra các nhãn thời gian nếu có một cơ quan dịch vụ dán nhãn đáng tin cậy. Bob có thể tính $z = h(x)$ và $y = sig_k(z)$ và sau đó gửi (z và x) đến cơ quan làm dịch vụ dán nhãn thời gian (TSS). TSS sau đó sẽ gắn ngày D và kí (đánh dấu)bộ ba (z,y,D). Công việc này sẽ hoàn hảo miễn là thuật toán kí của TSS an toàn và TSS không thể bị mua chuộc để lùi ngày dãn nhãn của thời gian. (chú ý rằng phương pháp này chỉ được thiết lập khi bob đã kí một bức điện trước một thời gian nào đó. Nếu bob muốn thiết lập cáI anh ta đã kí nó sau ngày nào đó ,anh ta có thể kết hợp thông tin công khai pub nào đó như phương pháp trước đó).

Hình 7.11 :Dán nhãn thời gian lên chữ kí trên bức điện x.

1. Bob tính $z = h(x)$.
2. Bob tính $z' = h(z \| pub)$.
3. Bob tính $y = sig_k(z')$.
4. Bob công bố (z,pub,y) trên tờ báo ra ngày hôm sau.

Nếu như không muốn tin vô điều kiện vào TSS thì có thể tăng độ an toàn lên bằng cách liên kết các thông báo đã dán nhãn thời gian. Trong sơ độ như vậy, bob sẽ gửi một bộ ba được xếp thứ tự (z,x,ID)(Bob) cho TSS. ở đây, z là bản tóm lược thông báo của bức điện x,y là chữ kí của bob trên z ,còn ID(Bob) là thông tin định danh của Bob. TSS sẽ dãn nhãn thời gian một chuỗi bộ ba có dạng này. Kí hiệu (z_n,y_n,ID_n) là bộ ba thứ tự n được TSS dán nhãn thời gian và cho t_n là kí hiệu thời gian lúc thực hiện yêu cầu thứ n.

Hình 7.12: Dán nhãn thời gian (z_n,y_n,ID_n).

1. TSS tính $L_n = (t_{n-1},ID_{n-1},Z_{n-1}y_{n-1},h(L_{n-1}))$
2. TSS tính $C_n = (n, t_n, z_n, ID_n, L_n)$
3. TSS tính $S_n = sig_{TSS}(h(C_n))$
4. TSS gửi (C_n, S_n, ID_n) cho ID_n.

TSS sẽ dán nhãn thời gian lên bộ ba thứ n bằng fthuật toán nêu trên hình 7.12. L_n là "thông tin liên kết để nối yêu cầu thứ n vào yêu cầu trước đó. (L_0 được chọn làm thông tin gia nào đó (được xác định trước đây)để quá trình được bắt đầu)".

Bây giờ nếu được yêu cầu (challenge). Bob có thể để lộ bức điện x_n của mình, và sau đó có thể xác minh y_n. Tiếp theo , các minh chữ kí s_n của TSS. Nếu muốn thì có thể đòi ID_{n-1} hoặc ID_{n+1} để tạo ra nhãn thời gian (C_{n-1}, S_{n-1}, ID_n) và (C_{n+1}, S_{n+1}, ID_{n+2}) tương ứng của chúng. Các chữ kí của TSS có thể được kiểm tra theo nhãn thời gian này. Dĩ nhiên , quá trình này có thể tiếp tục tới mức mong muốn, trước hay sau đó.

7.9.CÁC CHÚ Ý VỀ TÀI LIỆU DẪN

Hàm hash log rời rạc được mô tả trong mục 7.4 là của chaum , van heijst và pfitzmann [CvHP92]. Còn hàm hash có thể chứng mình đwocj là an toàn liễn là hợp số n không thể phân tích thành nhân tư là do gibson [Gib91] đưa ra (bài tập 7.4 có mô tả sơ đồ này).

Cơ sở cho việc m mở rộng hàm hash trong mục 7.5 là của Damgard [DA90] Merkle cũng đưa ra các phương pháp tương tự [ME90].

Các thông tin liên qua tới việc xây dựng các hàm hash dựa trên các hệ thông mã khoá bí mật. Bạn đọc có thể xem trong [PGV94] của Preneel, Govaerts và Vandewalle.

Thuật toán MD4 được đưa ra trong [Ri91] của Rivest, còn tiêu chuẩn hash an toàn được mô tả trong [NBS93]. Tấn công hai trong ba vòng MD4 là của Boer và Bossalaer [DBB92]. Các hàm hash gần đây kể cả N-hash là của [MOI90] và Snefru [ME90A].

Ngoài ra có thể tìm thấy tổng quan về kĩ thuật băm trong Preneel, Govaerts, Vandewalle [PGV93].

Bài tập

7.1. Giả sử h: $X \to Y$ là hàm hash. Với y bất kỳ $\in Y$, cho:

$h^{-1}(y) = \{ x: h(x) = y\}$

và ký hiệu $s_y = | h^{-1}(y)|$.

Định nghĩa N =

CHƯƠNG 8
PHÂN PHỐI VÀ THOẢ THUẬN VỀ KHOÁ

8.1 GIỚI THIỆU:

Chúng ta đã thấy rằng, hệ thống mã khoá công khai có ưu điểm hơn hệ thống mã khoá riêng ở chỗ không cần có kênh an toàn để trao đổi khoá mật. Tuy nhiên, đáng tiếc là hầu hết các hệ thống mã khoá công khai đều chậm hơn hệ mã khoá riêng, chẳng hạn như DES. Vì thế thực tế các hệ mã khoá riêng thường được dùng để mã các bức điện dài. Nhưng khi đó chúng ta lại trở về vấn đề trao đổi khoá mật.

Trong chương này, chúng ta sẽ thảo luận vài biện pháp thiết lập các khoá mật. Ta phân biệt giữa phân phối khoá và thoả thuận về khoá. Phân phối khoá được định nghĩa là cơ chế một nhóm chọn khoá mật và sau đó truyền nó đến các nhóm khác. Còn thoả thuận khoá là giao thức để hai nhóm (hoặc nhiều hơn) liên kết với nhau cùng thiết lập một khoá mật bằng cách liên lạc trên kênh công khai. Trong sơ đồ thoả thuận khoá, giá trị khoá được xác định như hàm của các đầu vào do cả hai nhóm cung cấp.

Giả sử, ta có một mạng không an toàn gồm n người sử dụng. Trong một số sơ đồ, ta có người uỷ quyền được tín nhiệm (TA) để đáp ứng những việc như xác minh danh tính của người sử dụng, chọn và gửi khoá đến người sử dụng ... Do mạng không an toàn nên cần được bảo vệ trước các đối phương. Đối phương (Oscar) có thể là người bị động, có nghĩa là hành động của anh ta chỉ hạn chế ở mức nghe trộm bức điện truyền trên kênh. Song mặt khác, anh ta có thể là người chủ động. Một đối phương chủ động có thể làm nhiều hành vi xấu chẳng hạn:

1. Thay đổi bức điện mà anh ta nhận thấy là đang được truyền trên mạng.
2. Cất bức điện để dùng lại sau này.
3. Cố gắng giả dạng làm những người sử dụng khác nhau trên mạng.

Mục tiêu của đối phương chủ động có thể là một trong những cái nêu sau đây:

1. Lừa U và V chấp nhận khoá "không hợp lê" như khoá hợp lệ (khoá không hợp lệ có thể là khoá cũ đã hết hạn sử dụng, hoặc khoá do đối phương chọn).
2. Làm U hoặc V tin rằng, họ có thể trao đổi khoá với người kia khi họ không có khoá.

Mục tiêu của phân phối khoá và giao thức thoả thuận khoá là, tại thời điểm kết thúc thủ tục, hai nhóm đều có cùng khoá K song không nhóm khác nào biết được (trừ khả năng TA). Chắc chắn, việc thiết kế giao thức có kiểu an toàn này khó khăn hơn nhiều trước đối phương chủ động.

Trước hết ta xem xét ý tưởng về sự phân phối khoá trước trong mục 8.2. Với mỗi cặp người sử dụng {U,V}, TA chọn một khoá ngẫu nhiên $K_{U,V}=K_{V,U}$ và truyền "ngoài dải" đến U và V trên kênh an toàn. (Nghĩa là, việc truyền khoá không xảy ra trên mạng do mạng không an toàn). Biện pháp này gọi là an toàn không điều kiện song nó đòi hỏi một kênh an toàn giữa TA và những người sử

dụng trên mạng. Tuy nhiên điều quan trọng hơn là mỗi người phải lưu n -1 khoá và TA cần truyền tổng cộng $\binom{n}{2}$khoá một cách an toàn (đôi khi bài toán này được gọi là bài toán n^2). Thậm chí với một số mạng tương đối nhỏ, giá để giải quyết vấn đề này là khá đắt và như vậy giải pháp hoàn toàn không thực tế.

Trong phần 8.2.1, chúng ta thảo luận một sơ đồ phân phối trước khoá an toàn không điều kiện khá thú vị do Blom đưa ra. Sơ đồ cho phép giảm lượng thông tin mật mà người sử dụng cần cất giữ trên mạng. Mục 8.2.2 cũng đưa ra một sơ đồ phân phối trước khoá an toàn về mặt tính toán dựa trên bài toán logarithm rời rạc.

Một biện pháp thực tế hơn là TA phân phối khoá trực tiếp. Trong sơ đò như vậy, TA làm việc như một người chủ khoá (key server). TA chia khoá mật K_U cho mỗi người sử dụng U trên mạng. Khi U muốn liên lạc với V, cô ta yêu cầu TA cung cấp khoá cho phiên làm việc (session key). TA tạo ra khoá session K và gửi nó dưới dạng mã hoá cho U và V để giải mã. Hệ thống mã Kerboros mô tả trong mục 8.3 là dựa trên biện pháp này.

Nếu như cảm thấy vấn đề phân phối khoá thông qua TA không thực tế hoặc không mong muốn thì biện pháp chung là dùng giao thức thoả thuận khoá. Trong giao thức thoả thuận khoá, U và V kết hợp chọn một khoá bằng cách liên lạc với nhau trên kênh công khai. ý tưởng đáng chú ý này do Martin và Diffie đưa ra độc lập với Merkle. ở đây mô tả vài giao thưc thoả thuận khoá phổ thông hơn. Giao thức đầu tiên của Diffie và Hellman được cải tiến để ứng phó với các đối phương tích cực được nêu trong phần 8.4.1. Hai giao thức đáng quan tâm nữa cũng được xem xét: sơ đồ MTI nên trong 8.4.2 và sơ đồ Girault nêu trong mục 8.4.3

8.2 Phân phối khoá trước

theo phương pháp cơ bản, TA tạo ra $\binom{n}{2}$ khoá và đưa mỗi khoa cho duy nhất một cặp người sử dụng trong mạng có n người sử dụng. Như đã nêu ở trên, ta cần một kênh an toàn giữa TA và mỗi người sử dụng để truyền đi các khoá này. Đây là một cải tiến quan trọng vì số kênh an toàn cần thiết giảm từ $\binom{n}{2}$ xuống còn n. Song nếu n lớn, giải pháp này cũng không thực tế cả về lượng thông tin cần truyền đi an toàn lẫn lượng thông tin mà mỗi người sử dụng phải cất giữ an toàn (nghĩa là các khoá mật của n-1 người sử dụng khác).

như vậy, điều cần quan tâm là cố gắng giảm được lượng thông tin cần truyền đi và cất giữ trong khi vẫn cho phép mỗi cặp người sử dụng U và V có khả năng tính toán khoá mật $K_{U,V}$. Một sơ đồ ưu việt hơn thoả mãn yêu cầu này là sơ đồ phân phối khoá trước của Blom.

8.2.1 Sơ đồ Blom.

Như trên, giả thiết rằng có một mạng gôm n người sử dụng. Để thuận tiện, giả sử rằng các khoá được chọn trên trường số hữu hạn Z_P, $p \geq n$ là số nguyên tố. Cho k là số nguyên, $1 < k < n - 2$. Giá trị k để hạn chế kích thước lớn nhất mà sơ đồ vẫn duy trì được mật độ. Trong sơ đồ Blom, TA sẽ truyền đi k +1 phần tử của Z_P cho mỗi người sử dụng trên kênh an toàn (so với n -1 trong sơ đồ phân phối trước cơ bản). Mỗi cặp người sử dụng U và V sẽ có khả năng tính khoá $K_{U,V} = K_{V,U}$ như trước đây. Điều kiện an toàn như sau: tập bất kì gồm nhiều nhất k người sử dụng không liên kết từ {U, V} phải không có khả năng xác định bất kì thông tin nào về $K_{U,V}$. (chú ý rằng, ta đang xét sự an toàn không điều kiện).

Trước hết, xét trường hợp đặc biệt của sơ đồ Blom khi k =1. ở đây TA sẽ truyền đi 2 phần tử của Z_P cho mỗi người sử dụng trên kênh an toàn và người sử dụng riêng W sẽ không thể xác định được bất kì thông tin nào về $K_{U,V}$ nếu $W \neq U,V$. Sơ đồ Blom được đưa ra trong hình 8.1. Ta sẽ minh hoạ sơ đồ Blom với k = 1 trong ví dụ sau:

Hình 8.1: Sơ đồ phân phối khoá của Blom (k =1)

1. Số nguyên tố p công khai, còn với mỗi người sử dụng U, phần tử $r_U \in Z_P$ là công khai. Phần tử r_U phải khác biệt.
2. Ta chọn 3 phần tử ngẫu nhiên a, b, c $\in Z_P$ (không cần khác biệt) và thiết lập đa thức

8.3.Kerboros

trong các phương pháp phân phối trước khoá xem xét trong các phần trước đó, mỗi cặp người sử dụng cần tính một khoá cố định. Nếu dùng cùng một khoá trong một thời gian dài sẽ dễ bị tổn thương, vì thế người ta thường thích dùng phương pháp trực tiếp trong đó khoá của phiên lamà việc mới chỉ được tạo ra mỗi khi hai người sử dụng muốn liên lạc với nhau (gọi là tính tươi mới của khoá).

Nếu dùng phân phối khoá trực tiếp thì người sử dụng mạng không cần phải lưu các khoá khi muốn liên lạc với những người sử dụng khác (Tuy nhiên mỗi người đều được chia sẻ khoá với TA). Khoá của phiên làm việc (khóa session) sẽ được truyền đi theo yêu cầu của TA. Đó là sự đáp ứng của TA để đảm bảo khoá tươi.

Korobos là hệ thống dịch vụ khóa phổ cập dựa trên mã khoá riêng. Trong phần này sẽ đưa ra một tổng quan về giao thức phát hành khoá session trong Korobos. Mỗi người sử dụng U sẽ chia sẻ khoá DES mật K_U cho TA. Trong phiên bản gần đây nhất của Korobos (version 5), mọi thông báo cần truyền được mã hoá theo chế độ xích khối (CBC) như mô tả trong 3.4.1

Như trong mục 8.2.2, ID(U) chỉ thông tin định danh công khai cho U. Khi có yêu cầu khoá session gửi đến TA, TA sẽ tạo ra một khoá session mới ngẫu nhiên K. Cũng vậy, TA sẽ ghi lại thời gian khi có yêu cầu T và chỉ ra thời gian (thời gian tồn tại) L để K có hiệu lực. Điều đó có nghĩa là khoá K chỉ có hiệu lực từ T đến T+L. Tất cả thông tin này đều được mã hoá và được truyênông dân đến U và V. Trước khi đi đến các chi tiết hơn nữa, ta sẽ đưa ra giao thức trong hình 8.4. thông tin được truyền đi trong giao thức được minh hoạ như sau:

Hình 8.4: Truyền khoá session trong Korobos.

1.

Ta sẽ giải thích điều sắp sửa xảy ra trong các bước của giao thức. Mặc dù không có chứng minh hình thức rằng Kerobos là an toàn trước đối thủ tích cực, song ít nhất ta cũng có thể đưa ra lí do nào đó về các đặc điểm của giao thức.

Như nêu ở trên, TA tạo ra K, T và L trong bước 2. Trong bước 3, thông tin này cùng với ID(V) được mã hoá bằng khoá K_U (được U và TA chia sẻ) để tạo lập m_1. Cả hai bức điện đã mã hoá này được gửi đến U.

U có thể dùng khoá của mình giải mã m_1, nhận được K, T và L. Cô sẽ xác minh xem thời gian hiện tại có nằm trong khoảng T đến T + L hay không. Cô cũng kiểm tra khoá session K được phát ra cho liên lạc giữa cô và V bằng cách xác minh thông tin ID(V) đã giải mã từ m_2.

Tiếp theo, U sẽ làm trễ thời gian m_2 và m_3 đến V. Cũng như vậy, U sẽ dùng khoá session K mới để mã T và ID(U) và gửi kết quả m_3 đến V.

Khi V nhận được m_3 và m_3 từ U, V giải mã m_2 thu được T, K, L và ID(U). Khi đó, anh ta sẽ dùng khoá session mới K để giải mã m_3 và xác minh xem T và

ID(U) nhận được từ m_2 và m_3 có như nhau không. Điều này đảm bảo cho V rằng khoá session được mã bằng m_2 cũng là khoá đã dùng để mã m_3. Khi đó V dùng K để mã T+1 và gửi kết quả m_4 trở về U.

Khi U nhận được m_4, cô dùng K giải mã nó và xác minh xem kết quả có bằng T+1 không. Công đoạn này đảm bảo cho U rằng khoá session K đã được truyền thành công đến V vì K đã được dùng để tạo ra m_4.

Điều quan trọng cần lưu ý là các chức năng khác nhau của các thông báo dùng trong giao thức, m_1 và m_2 dùng để bảo đảm an toàn trong việc truyền khoá session. Còn m_3 và m_4 dùng để khẳng định khoá, nghĩa là cho phép U và V có thể thuyết phục nhau rằng họ sở hữu cùng một khoá session K. Trong hầu hết các sơ đồ phân phối khoá, sự khẳng định khoá đượoc coi như một đặc tính. Thường thì nó được thực hiện tương tự kiểu Kerobos, U dùng K để mã ID(U) và T dùng để mã trong m_2. Tương tự, V dùng K để mã T+1.

Mục đích của thời gian hệ thống T và thời hạn L để ngăn đối phương tích cực khỏi "lưu" thông báo cũ nhằm tái truyền lại sau này (đây được gọi là tấn công kiểu chơi lại - relay attack). Phương pháp này hiệu quả vì các khoá không được chấp nhận là hợp lệ một khi chúng quá hạn.

Một trong hạn chế của Kerobos là mọi người sử dụng trong mạng đều phải có đồng hồ đồng bộ với nhau vì cần có thời gian hiện tại để xác định khoá session K cho trước là hợp lệ. Thực tế, rất khó có được sự đồng bộ hoàn hảo nên phải cho phép có khoảng thay đổi nào đó về thời gian.

Hình 8.5: Trao đổi khoá Diffie - Hellman

8.4 Trao đổi khoá Diffie - Hellman

Nếu ta không muốn dùng dịch vụ khoá trực tiếp thì buộc phải dùng giao thức thoả thuận khoá để trao đổi khoá mật. Trước hết, giao thức thoả thuận khoá nổi tiếng nhất là giao thức trao đổi khoá Diffie - Hellman. Giả sử rằng, p là số nguyên tố, α là phần tử nguyên thuỷ của Z_P và chúng đều là những tham số công khai. Giao thức trao đổi khoá Diffie - Hellman được đưa ra trong mục 8.5.

Cuối giao thức, U và V tính ra cùng một khoá:

Giao thức này cũng tương tự với sơ đồ phân phối khoá trước của Diffie - Hellman đã mô tả trước đây. Sự khác nhau ở chỗ các số mũ a_U, a_V của U và V đều được chọn lại mỗi lần thực hiện giao thức thay vì cố định. Cũng như vậy, trong giao thức này, cả U lẫn V đều được đảm bảo khoá tươi vì khoá session phụ thuộc vào cả hai số mũ ngẫu nhiên a_U và a_V.

8.4.1 Giao thức trạm tới trạm.

Trao đổi khoá Diffie - Hellman được đề xuất như sơ đồ sau:

(Sơ đồ)

Đáng tiếc là giao thức dễ bị tổn thương trước đối phương tích cực - những người sử dụng tấn công "kẻ xâm nhập vào giữa cuộc" (Intuder - in -middle - attack). Đó là tình tiết của vở "The Lucy show", trong đó nhân vật Vivian Vance đang dùng bữa tối với người bạn, còn Lucille Ball đang trốn dưới bàn. Vivian và người bạn của cô nắm tay nhau dưới bàn. Lucy cố tránh bị phát hiện đã nắm tay của cả hai người, còn hai người vẫn nghĩ rằng họ đang nắm tay nhau.

Cuộc tấn công kiểu "kẻ xâm nhập giữa cuộc" trên giao thức trao đổi khoá Diffie - Hellman cũng như vậy. W sẽ chặn bắt được các bức điện trao đổi giữa U và V và thay thế bằng các bức điện của anh ta như sơ đồ dưới đây:

(sơ đồ)

Tại thời điểm cuối của giao thức, U thiết lập thực sự khoá mật $\alpha^{a_U a_V'}$ cùng với W, còn V thiết lập khoá mật $\alpha^{a_U' a_V}$ với W. Khi U cố giải mã bức điện để gửi cho V, W cũng có khả năng giải mã nó song V không thể, (tương tự tình huống nắm tay nhau nếu V gửi bức điện cho U).

Rõ ràng, điều cơ bản đối với U và V là bảo đảm rằng, họ đang trao đổi khoá với nhau mà không có W. Trước khi trao đổi khoá, U và V có thể thực hiện những giao thưc tách bạch để thiết lập danh tính cho nhau, ví dụ, nhờ dùng một trong các sơ đồ định danh mô tả trong chương 9. Tuy nhiên, điều này có thể đưa đến việc không bảo vệ được trước tấn công kẻ xâm nhập giữa cuộc nếu W vẫn duy trì một cách đơn giản sự tấn công thụ động cho đến khi U và V đã chứng minh danh tính của họ cho nhau. Vì thế giao thức thoả thuận khoá tự nó cần xác thực được các danh tính của những người tham gia cùng lúc khoá được thiết lập. Giao thức như vậy được gọi là giao thức thoả thuận khoá đã xác thực.

Ta sẽ mô tả một giao thức thoả thuận khoá là cải tiến của sơ đồ trao đổi khoá Diffie - Hellman. Giao thức giả thiết số nguyên tố p và phần tử nguyên thuỷ α là công khai và nó dùng với các dấu xác nhận. Mỗi người sử dụng U sẽ có một sơ đồ chữ kí với thuận toán xác minh ver_U. TA cũng có sơ đồ chữ kí với thuật toán xác minh công khai ver_{TA}. Mỗi người sử dụng U có dấu xác nhận:

$C(U) = (ID(U), ver_U, sig_{TA}(ID(U), ver_U))$

Trong đó ID(U) là thông tin định danh cho U

Hình 8.6 Giao thức trạm tới trạm đơn giản.

Thoả thuận khoá đã xác thực do Diffie - Hellman, van Oorschot và Viener đưa ra được gọi là giao thức trạm đến trạm (viết tắt là STS). Giao thức đưa ra trên hình 8.6 đơn giản hơn một chút: nó có thể được dùng để có thể phù hợp với các giao thức của ISO 9798-3.

Thông tin được trao đổi trong sơ đồ STS đã đơn giản hoá (gồm cả các dấu xác nhận) được minh hoạ như sau:

(sơ đồ)
Ta hãy xem cách bảo vệ này trước tấn công kẻ xâm nhập giữa cuộc. Như trước đây, W sẽ chặn bắt α^{a_U} và thay nó bằng

8.4.2. Các giao thức thoả thuận khoá MTI

Matsumoto, Takashima và Imai đã xây dựng vài giao thức thoả thuận khoá đáng chú ý bằng cách biến đổi giao thức trao đổi khoá của Diffie - Hellman. Các giao thức này được gọi là MTI. Giao thức này không đòi hỏi U và V phải tính bất kì chữ kí nào. Chúng là các giao thức hai lần vì chỉ có hai lần truyền thông tin riêng biệt (một từ U đến V và một từ V đến U). Trái lại, giao thức STS được gọi là giao thức ba lần.

Hình 8.7: Giao thức thoả thuận khoá MTI.

Ta đã đưa ra một trong các giao thức MIT. Việc thiết lập chúng giống như giao thức phân phối khoá trước Diffie – Hellman. Giả thiết số nguyên tố p và phần tử nguyên thuỷ α là công khai. Mỗi người sử dụng U đều có chuỗi ID(U), số mũ mật a_U ($0 \le a_U \le p-2$) và giá trị công khai tương ứng:

TA có sơ đồ chữ kí với thuật toán xác minh (công khai) ver_{TA} và thuật toán kí mật sig_{TA}.

Mỗi người sử dụng U sẽ có dấu xác nhận:

$$C(U) = (ID(U), b_U, sig_{TA}(ID(U), b_U)).$$

Trong đó b_U được thiết lập như trên.

Giao thức thoả thuận khoá MTI được đưa ra trên hình 8.7. Cuối giao thức U và V đều tính cùng một khoá:

K =

Dưới đây là ví dụ minh hoạ giao thức này:

Ví dụ 8.3.

Giả sử p = 27803, α = 5. Giả sử U chọn a_U = 21131: sau đó cô ta tính:

$$b_U = 5^{21131} \bmod 27803 = 21420.$$

được đóng trên giấy xác nhận của cô. Cũng như vậy, V chọn a_V = 17555. Sau đó anh ta sẽ tính:

$$b_V = 5^{17555} \bmod 27803 = 17100.$$

được dặt trên giấy xác nhận của anh.

Bây giờ giả sử rằng U chọn r_U =169, sau đó cô gửi giá trị:

$$s_U = 5^{169} \bmod 27803 = 6268.$$

đến V. Lúc đó giả sử V chọn $r_V = 23456$, sau đó anh ta gửi giá trị:

$s_U = 5^{23456} \bmod 27803 = 26759$

đến U.

Bây giờ U tính khoá:

$K_{U,V} =$

$= 6268^{17555}\ 21420^{23456} \bmod 27803$

$= 21600.$

Như vậy, U và V đã tính cùng một khóa. …

Thông tin được truyền trong giao thức được miêu tả như sau:

(sơ đồ)

Hãy xét độ mật của sơ đồ. Không khó khăn nhận thấy rằng, độ mật của giao thức MTI trước tấn công thụ động đúng bằng bài toán Diffie – Hellman. Cũng như nhiều giao thức, việc chứng minh tính an toàn trước tấn công chủ động không phải đơn giản, chúng ta sẽ không thử chứng minh bất cứ điều gì về điều này và tự hạn chế đến một số đối số không hình thức.

Đây là một mối nguy hiểm có thể xem xét: Khi không dùng chữ kí trong suốt quá trình thực hiện giao thức, có thể xuất hiện tình huống không có sự bảo vệ nào trước tấn công xâm nhập vào điểm giữa. Quả thực, có khả năng W có thể chọn các giá trị mà U và V gửi cho nhau. Dưới đây mô tả một tình huống quan trọng có thể xuất hiện:

(sơ đồ)

Trong trường hợp này, U và V sẽ tính các khoá khác nhau: U tính

$K =$

Trong khi đó V tính:

$K =$

Tuy nhiên, W không thể tính toán ra khoá của U và V vì chúng đòi hỏi phải biết số mũ mật a_U và a_V tương ứng. Thậm chí ngay cả khi U và V tính ra các khoá khác nhau (mà dĩ nhiên là không dùng chúng) thì W cũng không thể tính được khoá nào trong chúng. Nói cách khác, cả U lẫn V đều được bảo đảm rằng, người sử dụng khác trên mạng chỉ có thể tính được khoá mà họ tính được. Tính chất này đôi khi được gọi là *xác thực khoá ẩn (implicit key authentication)*

8.4.3 Thoả thuận khoá dùng các khoá tự xác nhận

Trong phần này, ta mô tả một phương pháp thoả thuận khoá do chính Girault đưa ra không cần dấu xác nhận. Giá trị của khoá công khai và danh tính người sở hữu nó sẽ ngầm xác thực lẫn nhau.

Sơ đồ Girault kết hợp các tính chất của RSA và các logarithm rời rạc. Giả sử $n = pq$, $p = p_1+1$, $q = 2q_l+1$, còn p, q, $p_1$và q_1 đều là các số nguyên tố lớn. Nhóm nhân Z_n^* là đẳng cấu với $Z_p^* \times Z_q^*$. Bậc cực đại của phần tử bất kì trong Z_n^* bởi vậy là bội chung nhỏ nhất của p - 1 và q - 1, hoặc $2p_1q_1$. Cho α là phân tử có

bậc $2p_1q_1$. Khi đó nhóm cyclic của Z_n^* do α tạo ra là thiết lập thích hợp của bài toán logarithm rời rạc.

Trong sơ đồ Girault, chỉ TA biết được phân tích nhân tử của n. Các giá trị n và α là công khai, song p, q, p_1 và q_1 đều là mật. TA chọn số mũ mã công khai RSA, kí hiệu là e. Số mũ giải mã tương ứng bí mật là d (nhớ rằn $d = e^{-1} \bmod \phi(n)$).

Mỗi người sử dụng U có một chuỗi ID(U) như trong các sơ đồ trước đây. U nhận được khoá tự xác nhận công khai p_U từ TA như nêu trên hình 8.8. Nhận xét rằng, U cần TA giúp đỡ để tạo p_U. Cũng chú ý rằng:

$$b_U = p_U^e + ID(U) \bmod n$$

Hình 8.8: Nhận khoá tự xác nhận từ TA

1. U chọn số mũ mật a_U và tính:

   $b_U =$
2. U đưa a_U và b_U cho TA
3. TA tính:

   $p_U = (b_U - ID(U))^d \bmod n$
4. TA đưa p_U cho U

Có thể tính từ p_U và ID(U) bằng thông tin công khai có sẵn.

Giao thức thoả thuận khoá Girault được đưa ra trên hình 8.9. Thông tin truyền đi trong giao thức như sau:

$$U \quad \frac{ID(U),\ p_U,\ \alpha^{r_U} \bmod n}{ID(V),\ p_V,\ \alpha^{r_V} \bmod n} \quad V$$

Cuối giao thức, U và V tính khoá:

$$K = \alpha^{r_U a_V + r_V a_U} \bmod n$$

Dưới đây là một ví dụ về trao đổi khoá trong sơ đồ Girault.

Ví dụ 8.4:

Giả sử p =839, q = 863. Khi đó n = 724057 và $\phi(n) = 722356$. Phần tử α=5 có bậc $2p_1q_1 = \phi(n)/2$. Giả sử TA chọn d = 125777 làm số mũ giải mã RSA, khi đó e = 84453.

Giả sử U có ID(U) = 500021 và a_U = 111899. Khi đó b_U = 488889 và p_U =650704. Cũng giả thiết rằng V có ID(V) = 500022 và a_U = 123456. Khi đó b_V = 111692 và p_V = 683556.

Bây giờ U và V muốn trao đổi khoá. Giả sử U chọn r_U =56381, nghĩa là s_U=171007. Tiếp theo, giả sử V chọn r_V = 356935, nghĩa là s_V =320688.

Khi đó cả U lẫn V sẽ tính cùng một khoá K = 42869. …

Hình 8.9: Giao thức thoả thuận khoá của Girault

1. U chọn r_U ngẫu nhiên và tính

   $s_u =$
2. U gửi ID(U), p_U và s_U cho V.
3. V chọn r_V ngẫu nhiên và tính

$$s_V = \alpha^{r_V} \bmod n$$

4. V gửi ID(V), p_V và s_V cho U
5. U tính:

$$K = s_V^{a_U} (p_V^e + ID(V))^{r_U} \bmod n$$

Và V tính:

$$K = s_U^{a_V} (p_U^e + ID(U))^{r_V} \bmod n$$

Xét cách các khoá tự xác thực bảo vệ chống lại một kiểu tấn công. Vì các giá trị b_U, p_U và ID(U) không được TA kí nên không có cách nào để ai đó xác minh trực tiếp tính xác thực của chúng. Giả thiết thông tin này bị W - người muốn giả danh U - giả mạo (tức là không hợp tác với TA để tạo ra nó). Nếu W bắt đầu bằng ID(U) và giá trị giả b'_U. Khi đó không có cách nào để cô ta tính được số mũ a'_U tương ứng với b'_U nếu bài toán logarithm rời rạc khó giải. Không có a'_U, W không thể tính được khoá.

Tình huống tương tự nếu W hoạt động như kẻ xâm nhập giữa cuộc. W sẽ có thể ngăn được U và V tính ra khoá chung, song W không thể đồng thời thực hiện các tính toán của U và V. Như vậy, sơ đồ cho khả năng xác thực ngầm như giao thức MTI.

Bạn đọc có thể tự hỏi tại sao U được yêu cầu cung cấp các giá trị a_U cho TA. Quả thực, TA có thể tính p_U trực tiếp từ b_U mà không cần biết a_U song điều quan trọng ở đây là TA sẽ được thuyết phục rằng, U biết a_U trước khi TA tính p_U cho U.

Điểm này được minh hoạ bằng cách chỉ ra sơ đồ có thể bị tấn tông nếu TA phát bừa bãi các khoá công khai p_U cho những người sử dụng mà không kiểm tra trước hết xem họ có sở hữu các a_U tương ứng với các b_U của họ hay không. Giả sử W chọn một giá trị giả a'_U và tính giá trị tương ứng:

$$b'_U = \alpha^{a'_U} \bmod n$$

Đây là cách anh ta có thể xác định khoá công khai tương ứng

$$p'_U = (b'_U - ID(U))^d \bmod n$$

W sẽ tính:

$$p'_W = b'_W - ID(U) + ID(W)$$

và sau đó đưa b'_W và ID(W) cho TA. Giả sử TA phát ra khoá công khai

$$p'_W = (b'_W - ID(W))^d \pmod n$$

cho W. Nhờ dùng yếu tố:

$$b'_W - ID(W) \equiv b'_U - ID(U) \pmod n$$

có thể suy ra rằng: $p'_W = p'_U$.

Cuối cùng, giả sử U và V thực hiện giao thức còn W thay thế thông tin như sau:

$$\text{U} \quad \underset{\text{ID(V), } p_V,\ \alpha^{r_v} \bmod n}{\overset{\text{ID(U), } p_U,\ \alpha^{r_u} \bmod n}{\rightleftarrows}} \quad \text{V} \quad \underset{\text{ID(V), } p'_V,\ \alpha^{r_v} \bmod n}{\overset{\text{ID(U), } p'_U,\ \alpha^{r'_U} \bmod n}{\rightleftarrows}} \quad \text{W}$$

Xét thấy V sẽ tính khoá:

$$K' = \alpha^{r_U' a_V + r_V a_U'} \bmod n$$

trong khi U sẽ tính khoá

$$K = \alpha^{r_U a_V + r_V a_U} \bmod n$$

W có thể tính K' như sau:

$$K' = s_V^{a_U'} (p_V^e + ID(V))^{r_U'} \bmod n$$

Như vậy, W và V chia sẻ nhau một khoá, song V nghĩ anh ta đang chia khoá với U. Như vậy, W sẽ có thể giải mã được bức điện mà V gửi cho U.

8.5 Các chú ý và tài liệu tham khảo.

Blom đã đưa ra sơ đồ phân phối khoá của ông trong [BL85]. Các bài báo có tính chất tổng quát hoá cũng có trong một số bài báo khác của ông [BDSHKVY93] và của Beimel và Chor [BC94].

Diffie và Hellman đưa ra thuật toán trao đổi khoá của họ trong [DH76]. ý tưởng về trao đổi khoá cũng được Merkle đưa ra độc lập trong [ME78]. Những ý kiến về trao đổi khoá xác thực được lấy từ Diffie, Van Oorschot và Wiener [DVW92].

Phiên bản thứ 5 về Kerobos được mô tả trong [KN93]. Còn bài báo gần đây nhất về Kerobos xem trong [SC94] của Schiller.

Các giao thức của Matsumoto, Takashima và Imai có thể tìm thấy trong [MTI86]. Phân phối khoá tự xác nhận được giới thiệu trong Girault [GIR91]. Sơ đồ mà ông đưa ra thực sự là sơ đồ phân phối khoá trước: Bản cải tiến sơ đồ thoả thuận khoá dựa trên [RV94].

Hai tổng quan gần đây về phân phối khoá và thoả thuận khoá là của Rueppel và Van Oorschot [RV94] và Van Tilburg [VT93].

Bài tập

8.1 Giả sử sơ đồ Blom với k =1 được thực hiện cho tập 4 người sử dụng, U, V, W và X. Giả thiết p = 7873, r_U = 2365, r_V =6648, r_W = 1837 còn r_X = 2186. Các đa thức mật g như sau:

$g_U(x) = 6018 + 6351x$

$g_V(x) = 3749 + 7121x$

$g_W(x) = 7601 + 7802x$

$g_X(x) = 635 + 6828x$

a/ Tính khoá cho mỗi cặp người sử dụng, xác minh rằng mỗi cặp nhận được một khoá chung (nghĩa là $K_{U,V} = K_{V,U}$ v.v...)

b/ Chỉ ra cách W và X cùng nhau tính khoá $K_{V,U}$

8.2 Giả thiết sơ đồ Blom với k = 2 được thực hiện cho tập 5 người sử dụng U, V, W, X và Y. Giả thiết p = 97, r_U = 14, r_V = 38, r_W = 92, r_X =69 còn r_Y = 70. Các đa thức mật g như sau:

$g_U(x) = 15 + 15x + 2x^2$

$g_V(x) = 95 + 77x + 83x^2$
$g_W(x) = 88 + 32x + 18x^2$
$g_X(x) = 62 + 91x + 59x^2$
$g_Y(x) = 10 + 82x + 52x^2$

a/ Chỉ ra cách U và V tính khoá $K_{U,V} = K_{V,U}$
b/ Chỉ ra cách W, X và Y cùng nhau tính khoá $K_{U,V}$

Hình 8.10: Bài toán MTI

Bài toán: I =(p, α, β, γ, δ, ε) trong đó p là số nguyên tố, $\alpha \in Z^*_P$ là phần tử nguyên thuỷ còn β, γ, δ, ε $\in Z^*_P$
Mục tiêu: Tính $\beta^{\log_\alpha \gamma} \delta^{\log_\alpha \varepsilon} \bmod p$

8.3. Giả thiết U và V tiến hành trao đổi khoá theo sơ đồ Diffie - Hellman với p = 27001 và α = 101. Giả sử U chọn a_U = 21768 và V chọn a_V = 9898. Hãy chỉ ra các tính toán mà U và V thực hiện và xác định khoá mà họ tính được.

8.4. Giả thiết U và V tiến hành giao thức MTI với p = 30113, α = 52. Giả sử U có a_U = 12385. Hãy chỉ ra các tính toán mà cả U và V thực hiện và xác định khoá mà họ tính được.

8.5. Nếu đối phương thụ động cố gắng tính K do U và V xây dựng bằng giao thưc MTI (hình 8.10), khi đó anh ta phải đối mặt với bài toán MTI. Chứng minh rằng thuật toán bất kì giải được bài toán MTI thì cũng có thể giải được bài toán Diffie - Hellman và ngược lại.

8.6. Xét sơ đồ định danh Girault trong đó p = 167, q = 179 và vì thế n = 29893. Giả sử α = 2 và e = 11101.

a/ Tính d.

b/ Cho trước ID(U) = 10021 và a_U = 9843, tính b_U và p_U. Cho trước ID(V) = 10022 và a_V = 7692, hãy tính b_V và p_V

c/ Chỉ ta cách có thể tính b_U từ p_U và ID(V) bằng cách dùng số mũ công khai e. Tương tự, chỉ ra cách tính b_V từ p_V và ID(V).

d/ Giả sử U chọn ra r_U = 15556 và V chọn ra r_V = 6420. Hãy tính s_U và s_V và chỉ ra cách U và V tính khoá chung của họ.

CHƯƠNG 9
CÁC SƠ ĐỒ ĐỊNH DANH

9.1 GIỚI THIỆU.

Các kỹ thuật mật mã cho phép nhiều bài toán dường như không thể giải được thành có thể giải được. Một bài toán như vậy là bài toán xây dựng các sơ đồ định danh mật. Trong nhiều trường hợp cần thiết phải "chứng minh" bằng phương tiện điện tử danh tính của ai đó. Dưới đây là một số trường hợp điển hình:

1. Để rút tiền từ một máy thủ quỹ tự động (ATM), ta dùng thẻ cùng với số định danh cá nhân (PIN) có 4 chữ số.
2. Để trả tiền cho các cuộc mua bán trên điện thoại dùng thẻ tín dụng, tất cả đều cần số thẻ tín dụng (và thời hạn dùng thẻ)
3. Để trả tiền cho các cú gọi điện thoại đường dài (dùng thẻ gọi) chỉ cần số điện thoại và PIN 4 chữ số.
4. Để vào mạng máy tính, cần tên hợp lệ của người sử dụng và mật khẩu tương ứng.

Thực tế, các kiểu sơ đồ này thường không được thực hiện theo cách an toàn. Trong các giao thức thực hiện trên điện thoại, bất kì kẻ nghe trộm nào cũng có thể dùng thông tin định danh cho mục đích riêng của mình. Những người này cũng có thể là người nhận thông tin. Các mưu đồ xấu trên thẻ tín dụng đều hoạt động theo cách này. Thẻ ATM an toàn hơn một chút song vẫn còn những điểm yếu. Ví dụ, ai đó điều khiển đường dây liên lạc có thể nhận được tất cả các thông tin được mã hoá trên dải từ tính của thẻ cũng như thông tin về PIN. Điều này cho phép một kẻ mạo danh tiếp cận vào tài khoản nhà băng. Cuối cùng, việc chui vào mạng máy tính từ xa cũng là vấn đề nghiêm trọng do các ID và mật khẩu của người sử dụng được truyền trên mạng ở dạng không mã. Như vậy, họ là những vùng dễ bị tổn thương đối với những người điều khiển mạng máy tính.

Mục đích của sơ đồ định danh là: ai đó "nghe" như Alice tư xưng danh với Bob không thể tự bịa đặt mình là Alice. Ngoài ra, chúng ta sẽ cố gắng giảm xác suất để chính Bob có thể thử mạo nhận là Alice sau khi cô ta tự xưng danh với anh ta. Nói cách khác, Alice muốn có khả năng chứng minh danh tính của mình bằng phương tiện điện tử mà không cần đưa ra chút thông tin nào hết về danh tính của mình.

Một vài sơ đồ định danh như vậy đã được nêu ra. Một mục đích thực tế là tìm một sơ đồ đủ đơn giản để có thể thực hiện được trên thẻ thông minh, đặc biệt là thẻ tín dụng gắn thêm một chíp có khả năng thực hiện các tính toán số học. Vì thế, thẻ đòi hỏi cả khối lượng tính toán lẫn bộ nhớ nhỏ đến mức có thể. Thẻ như vậy an toàn hơn các thẻ ATM hiện tại. Tuy nhiên, điều quan trọng cần chú ý là sự an toàn "đặc biệt" liên quan đến người điều khiển

đường dây thông tin. Vì nó là thẻ để chứng minh danh tính nên không cần bảo vệ chống mất thẻ. Song nó vẫn cần thiết có PIN để biết ai là chủ nhân thực sự của thẻ.

Trong các phần sau sẽ mô tả một số sơ đồ định danh thông dụng nhất. Tuy nhiên, trước hết hãy xét một sơ đồ rất đơn giản dựa trên hệ thống mã khoá riêng bất kì, chẳng hạn như DES. Giao thức mô tả trên hình 9.1 được gọi là giao thức "yêu cầu và trả lời", trong đó giả thiết rằng, Alice đang tự xưng danh với Bob cô và Bob chia nhau một khoá mật chung K, khoá này chỉ là hàm mã e_K.

Hình 9.1: Giao thức Yêu cầu và đáp ứng:

1. Bob chọn một yêu cầu x- là một chuỗi ngẫu nhiên 64 bit. Bob gửi x cho Alice
2. Alice tính $y = e_K(x)$
   gửi nó cho Bob.
3. Bob tính:
   $y' = e_K(x)$
   và xác minh $y' = y$.

Ta sẽ minh hoạ giao thức này bằng ví dụ nhỏ dưới dây.

Ví dụ 9.1

Giả sử Alice và Bob dùng hàm mã làm luỹ thừa tính modulo:

$$e_K(x) = x^{102379} \bmod 167653.$$

Giả sử yêu cầu của Bob x = 77835. Khi đó Alice sẽ trả lời với y = 100369.

Mọi sơ đồ định danh thực sự đều là các giao thức "Yêu cầu và đáp ứng" song các sơ đồ hiệu quả nhất lại không yêu cầu các khoá chia sẻ (dùng chung). ý tưởng này sẽ được tiếp tục trong phần còn lại của chương này.

9.2 Sơ đồ định danh Schnorr.

Ta bắt đầu bằng việc mô tả sơ đồ định danh Schnorr - là một trong những sơ đồ định danh thực tiễn và đáng chú ý nhất. Sơ đồ này đòi hỏi một người được uỷ quyền có tín nhiệm mà ta ký hiệu là TA. Ta sẽ chọn các tham số cho sơ đồ như sau:

1. p là số nguyên tố lớn (tức $p \geq 2^{512}$) sao cho bài toán logarithm rời rạc trong Zp là không giải được.
2. q là ước nguyên tố lớn của p-1 (tức $q \geq 2^{140}$).
3. $\alpha \in Z_p^*$ có bậc q (có thể tính α như (p-1) ??) đều được công khai.

TA sẽ đóng một dấu xác nhận cho Alice. Khi Alice muốn nhận được một dấu xác thực từ TA, cô phải tiến hành các bước như trên hình 9.2. Vào

thời điểm cuối, khi Alice muốn chứng minh danh tính của cô trước Bob, cô thực hiện giao thức như trên hình 9.3.

Như đã nêu ở trên, t là một tham số mật. Mục đích của nó là ngăn kẻ mạo danh - chẳng hạn Olga - khỏi phỏng đoán yêu cầu r của Bob. Ví dụ, nếu Olga đoán đúng giá trị r, cô ta có thể chọn giá trị bất kỳ cho y và tính

$$\gamma = \alpha^y v^\gamma \bmod p$$

Cô sẽ đưa cho Bob γ như trong bước 1 và sau đó khi nhận được yêu cầu r, cô sẽ cung cấp giá trị y đã chọn sẵn. Khi đó γ sẽ được Bob xác minh như trong bước 6.

Hình 9.2 Cấp dấu xác nhận cho Alice.

1. TA thiết lập danh tính của Alice bằng cách lập giấy chứng minh thông thường chẳng hạn như xác nhận ngày sinh, hộ chiếu ... Sau đó TA thiết lập một chuỗi ID (Alice) chứa các thông tin định danh của cô ta.
2. Alice bí mật chọn một số mũ ngẫu nhiên a, $0 \le a \le q-1$. Alice tính:
   $v = \alpha^{-a} \bmod p$
   và gửi v cho TA
3. TA tạo ra một chữ kí:
   $s = sig_{TA}(I,v)$.
   Dấu xác nhận
   $C(Alice) = (ID(Alice),v,s)$
   và đưa cho Alice

Xác suất để Olga phỏng đoán đúng r là 2^{-t} nếu r được Bob chọn ngẫu nhiên. Như vậy, t = 40 là giá trị hợp lý với hầu hết các ứng dụng, (tuy nhiên, chú ý rằng, Bob sẽ chọn r ngẫu nhiên mỗi lần Alice xưng danh với anh ta. Nếu Bob luôn dùng cùng một r thì Olga có thể mạo danh Alice bằng phương pháp mô tả ở trên).

Có hai vấn đề nảy sinh trong giao thức xác minh. Trước hết, chữ kí s chứng minh tính hợp lệ của dấu xác nhân của Alice. Như vậy, Bob xác minh chữ ký của TA trên dấu xác nhận của Alice để thuyết phục chính bản thân mình rằng dấu xác nhận là xác thực. Đây là xác nhận tương tự như cách đã dùng ở chương 8.

Vấn đề thứ hai của giao thức liên quan đến mã số mật a. Giá trị a có chức năng tương tự như PIN để thuyết phục Bob rằng, người thực hiện giao thức định danh quả thực là Alice. Tuy nhiên có một khác nhau quan trọng so với PIN là: trong giao thức định danh, a không bị lộ. Thay vào đó, Alice (hay chính xác hơn là thẻ thông minh của cô) chứng minh rằng, cô (thẻ) biết giá trị a trong bước 5 bằng cách tính y trong khi trả lời đòi hỏi r do Bob đưa ra. Vì a không bị lộ nên kĩ thuật này gọi là chứng minh không tiết lộ thông tin.

Hình 9.3. sơ đồ định danh Schnorr

1. Alice chọn một số ngẫu nhiên k, $0 \le k \le q-1$ và tính:
   $\gamma = \alpha^k \bmod p$.

2. Alice gửi dấu xác nhận của mình cho C(Alice) = (ID(Alice),v,s) và γ cho Bob.
3. Bob xác minh chữ kí của TA bằng cách kiểm tra xem có thoả mãn
   *ver(*ID(Alice),v,s) = true hay không.
4. Bob chọn một số ngẫu nhiên r, $1 \le r \le 2^t$ và đưa nó cho Alice.
5. Alice tính:
   $y = k + ar \bmod q$
   và đưa y cho Bob.
6. Bob xác minh xem có thoả mãn đồng dư thức sau không
   $\gamma \equiv \alpha^y v^r \pmod p$.

Các đồng dư sau đây chứng minh rằng Alice có khả năng chứng minh danh tính của cô cho Bob:

$$\alpha^y v^r \equiv \alpha^{k+ar} v^r \pmod p$$
$$\equiv \alpha^{k+ar} v^{ar} \pmod p$$
$$\equiv \alpha^k \pmod p$$
$$\equiv \gamma \pmod p$$

Như vậy sẽ chấp nhận bằng chứng về danh tính của Alice và giao thức được gọi là có tính đầy đủ.

Dưới đây là một ví dụ nhỏ minh hoạ khía cạnh "thách thức và đáp ứng" của giao thức.

Ví dụ 9.2

Giả sử p=88667, q = 1031, t=10. Phần tử α = 70322 có bậc q thuộc Z_p^*. Giả sử số mã mật của Alice a = 755. Khi đó:

$$v = \alpha^{-a} \pmod p$$
$$= 70322^{1031-755} \bmod 88667$$
$$= 13136$$

Giả sử Alice chọn k = 543, sau đó cô tính:

$$\gamma = \alpha^k \bmod p$$
$$= 70322^{543} \bmod 88667$$
$$= 84109$$

và gửi γ cho Bob. Giả thiết Bob đưa ra yêu cầu r = 1000. Khi đó Alice tính:

$$y = k + ar \bmod q$$
$$= 543 + 755 \times 1000 \bmod 1031$$
$$= 851$$

và gửi y cho Bob. Sau đó Bob xác minh xem

$$84109 \equiv 70322^{851} 13136^{1000} \pmod{88667}$$

Nếu đúng, Bob sẽ tin rằng anh ta đang liên lạc với Alice.

Tiếp theo ta hãy xem xét cách ai đó có thể mạo danh Alice. Olga - kẻ đang cố mạo danh Alice bằng cách làm giả dấu xác nhận:

C'(Alice) = (ID(Alice), v', s'),

trong đó v'≠v. Song s' được giả thiết là chữ kí của (ID(Alice), v', s') và nó được Bob xác minh trong bước 3 của giao thức. Nếu sơ đồ chữ kí của TA là an toàn, Olga sẽ không thể làm giả chữ kí s' (mà sau này sẽ bị Bob xác minh).

Biện pháp khác sẽ cho Olga dùng dấu xác nhận đúng của Alice C(Alice) = (ID(Alice), v, s) (nhớ lại rằng, các dấu xác nhận không mật và thông tin trên dấu xác nhận bị lộ mỗi lần thực hiện giao thức định danh). Tuy nhiên Olga sẽ không thể mạo danh Alice trừ phi cô ta cũng biết giá trị a. Đó là vì "yêu cầu" r trong bước 4. ở bước 5, Olga sẽ phải tính y mà y là hàm của a. Việc tính a từ v bao hàm việc giải bài toán logarithm rời rạc là bài toán mà ta đã giả thiết là không thể giải được.

Có thể chứng minh một định lí chính xác hơn về tính an toàn của giao thức như sau:

Định lí 9.1.

Giả sử Olga biết giá trị γ nhờ đó cô có xác suất $\varepsilon \geq 1/2^{t-1}$ để giả mạo Alice thành công trong giao thức xác minh. Khi đó Olga có thể tính a trong thời gian đa thức.

Chứng minh

Với một phần ε trên 2^t yêu cầu r, Olga có thể tính giá trị y (sẽ được Bob chấp nhận trong bước 6). Vì $\varepsilon \geq 1/2^{t-1}$ nên ta có $2^t/\varepsilon \geq 2$ và bởi vậy, Olga có thể tính được các giá trị y_1, y_2, r_1 và r_2 sao cho

$$y_1 \neq y_2$$

và $\gamma \equiv \alpha^{y_1} v^{\hat{I}_1} \equiv \alpha^{y_2} v^{\hat{I}_2} \pmod p$

hay $\alpha^{y_1 - y_2} \equiv v^{r_1 - r_2} \pmod p$

Vì $v = \alpha^{-a}$ nên ta có:

$$y_1 - y_2 \equiv a(r_1 - r_2) \pmod q$$

Xét thấy $0 < | r_1 - r_2 | < 2^t$ và $q > 2^t$ là nguyên tố. Vì UCLN$(r_1 - r_2, q) = 1$ và Olga có thể tính:

$$a = (y_1 - y_2)(r_1 - r_2)^{-1} \bmod q$$

như mong muốn…

Định lý trên chứng minh rằng, bất kỳ ai có cơ hội (không phải không đáng kể) thực hiện thành công giao thức định danh đều phải biết (hoặc có thể tính trong thời gian đa thức) số mũ mật a của Alice. Tính chất này thường được gọi là tính đúng đắn (sound). Dưới đây là ví dụ minh hoạ:

Ví dụ 9.3

Giả sử ta cũng có các tham số như trong ví dụ 9.2: p = 88667, q = 1031, t= 10, α = 70322, a = 755 và v = 13136. Giả sử Olga nghiên cứu thấy rằng:

$\alpha^{851}v^{1000} \equiv \alpha^{454}v^{19} (\text{mod } p)$.

khi đó có thể tính:

$a = (851 - 454)(1000 - 19)^{-1} \text{ mod } 1031 = 755$

và như vậy sẽ khám phá ra số mũ mật của Alice. …

Chúng ta đã chứng minh rằng, giao thức có tính đúng đắn và đầy đủ. Song tính đúng đắn và đầy đủ chưa đủ để bảo đảm rằng giao thức là an toàn. Chẳng hạn, nếu Alice để lộ số mũ mật a của mình khi chứng minh danh tính của cô với Olga thì giao thức vẫn còn đúng đắn và đầy đủ. Tuy nhiên nó sẽ hoàn toàn không an toàn vì sau đó Olga có thể mạo danh Alice.

Điều này thúc đẩy động cơ xem xét thông tin mật đã cho người xác minh - người cũng tham gia trong giao thức - biết (trong giao thức này, thông mật là a). Hy vọng là không có thông tin nào về a có thể bị gia tăng bởi Olga khi Alice chứng minh danh tính của mình cho cô ta, để sau đó Olga có thể giả dạng như Alice.

Nói chung, có thể hình dung tình huống khi Alice chứng minh danh tính của mình với Olga trong một số tình huống khác nhau. Có lẽ Olga không chọn các yêu cầu của cô (tức các giá trị r) theo kiểu ngẫu nhiên. Sau vài lần thực hiện giao thức, Olga sẽ cố gắng xác định giá trị a để sau đó có thể mạo danh Alice. Nếu Olga không thể xác định được chút thông tin nào về a qua tham gia với số lần đa thức thực hiện giao thức và sau đó thực hiện một lượng tính toán đa thức thì giao thức có thể được gọi là an toàn.

Hiện tại vẫn chưa chứng minh được rằng giao thưc Schnorr là an toàn, song trong phần tiếp sau, ta sẽ đưa ra một cải tiến về sơ đồ này (do Okmoto đưa ra) mà có thể chứng minh được nó là an toàn khi cho trước giả thuyết tính toán nào đó.

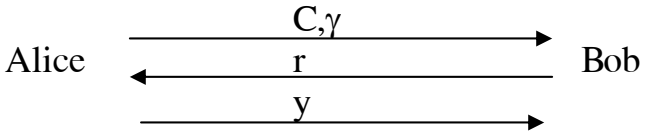
Sơ đồ Schnorr đã được thiết kế với tốc độ nhanh và hiệu quả theo quan điểm cả về tính toán lẫn lượng thông tin cần thiết để trao đổi trong giao thức. Nó cũng được thiết kế nhằm tối thiểu hoá lượng tính toán mà Alice phải thực hiện. Đây là những đặc tính tốt vì trong thực tế, các tính toán của Alice sẽ phải tính trên các thẻ thông minh có khả năng tính toán thấp trong khi các tính toán của Bob lại trên các máy lớn.

Vì mục đích thảo luận, ta hãy giả sử rằng, ID(Alice) là chuỗi 512 bit, v cũng gồm 512 bit, còn s bằng 320 bit nến DSS được dùng như sơ đồ chữ kí. Kích thước tổng cộng của dấu xác nhận C(Alice) (cần được lưu trên thẻ của Alice) là 1444 bit.

Xét các tính toán của Alice: Bước 1 cần lấy mũ theo modulo, bước 5 so sánh một phép công modulo và một phép nhân modulo. Đó là phép luỹ

thừa modulo mạnh về tính toán song có thể tính toán gián tiếp nếu muốn. Còn các tính toán trực tiếp được Alice thực hiện bình thường.

Việc tính số bit cần thiết trong quá trình liên lạc để thực hiện giao thức cũng khá đơn giản. Có thể mô tả thông tin được liên lạc ở dạng đồ hình như sau



Alice đưa cho Bob 1444 + 512 = 1956 bit thông tin trong bước 2: Bob đưa cho Alice 40 bit trong bước 4 và Alice đưa cho Bob 160 bit trong bước 6. Như vậy các yêu cầu về liên lạc rất mức độ.

9.3 Sơ đồ định danh của Okamoto.

Trong phần này ta sẽ đưa ra một biến thể của sơ đồ Schnorr do Okamoto đưa ra. Sơ đồ cải tiến này Zp không giải được.

Để thiết lập sơ đồ, TA chọn p và q như trong sơ đồ Schnorr. TA cũng chọn hai phần tử α_1 và $\alpha_2 \in Z_p^*$ đều có bậc q. Giá trị $c = \log_{\alpha 1}\alpha_2$ được giữ bí mật kể cả đối với Alice. Ta sẽ giả thiết rằng, không ai có thể giải được (thậm chí Alice và Olga liên minh với nhau) để tính ra giá trị c. Như trước đây, TA chọn sơ đồ chữ kí số và hàm hash. Dấu xác nhận mà TA đã phát cho Alice được xây dựng như mô tả ở hình 9.4.

Dưới đây là một ví dụ về sơ đồ Okamoto.

Ví dụ 9.4.

Cũng như ví dụ trước, ta lấy p = 88667, q = 1031, t = 10. Cho α_1 = 58902 và cho α_2 = 73611 (cả α_1 lẫn α_2 đều có bậc q trong Z_p^*). Giả sử a_1=846, a_2 = 515, khi đó v = 13078.

Giả sử Alice chọn k_1 = 899, k_2 = 16, khi đó γ = 14573. Nếu Bob đưa ra yêu cầu r = 489 thì Alice sẽ trả lời y_1 = 131 và y_2 = 287. Bob sẽ xác minh thấy:

$58902^{131}78611^{287}1378^{489} \equiv 14574 \pmod{88667}$.

Vì thế Bob chấp nhận bằng chứng của Alice về danh tính của cô. …

Việc chứng minh giao thức là đầy đủ không khó (tức là Bob sẽ chấp nhận bằng chứng về danh tính của cô). Sự khác nhau giữa sơ đồ của Okamoto và Schnorr là ở chỗ, ta có thể chứng minh rằng sơ đồ Okamota an toàn miễn là bài toán logarithm rời rác không giải được.

Hình 9.4: Đóng dấu xác nhận cho Alice.

1. TA thiết lập danh tính của Alice và phát chuỗi định danh ID(Alice).
2. Alice chọn bí mật hai số mũ ngẫu nhiên a_1,a_2 trong đó $0 \le a_1,a_2 \le q - 1$ Alice tính:

$$v = \alpha_1^{-a_1}\alpha_1^{-a_2} \bmod p$$

và đưa cho TA.

3. TA tạo chữ kí

$$s = sig_{TA}(I,v).$$

và đưa dấu xác nhận

$$C(Alice) = (ID(Alice),v,s)$$

cho Alice

Phép chứng minh về tính an toàn rất tinh tế. Đây là ý kiến chung: Như trước đây, Alice tự định danh với Olga trong nhiều thời gian đa thức thông qua thực hiện giao thức. Khi đó ta giả thiết rằng Olga có khả năng nghiên cứu một số thông tin về các giá trị a_1,a_2. Nếu như vậy thì Olga và Alice kết hợp với nhau có khả năng tính được logarithm rời rạc c trong thời gian đa thức. Điều này mâu thuẫn với giả định ở trên và chứng tỏ rằng Olga chắc không thể nhận được chút thông tin nào về các số mũ của Alice thông qua việc tham gia vào giao thức.

Phần đầu tiên của giao thức này tương tự với chứng minh tính đầy đủ trong sơ đồ Schnorr.

Định lý 9.2.

Giả sử Olga biết a giá trị γ mà nhờ nó cô có xác suất thành công $\varepsilon \geq 1/2^{t-1}$ khi đánh giá Alice trong giao thức xác minh. Khi đó, Olga có thể tính các giá trị b_1,b_2 trong thời gian đa thức sao cho

$$v \equiv \alpha_1^{-b_1}\alpha_1^{-b_2} \bmod p .$$

Chứng minh:

Với phần ε trên 2^t yêu cầu có thể r, Olga có thể tính các giá trị y_1, y_2, z_1, z_2, r và s với $r \neq s$ và:

$$\gamma \equiv \alpha_1^{y_1}\alpha_2^{y_2} v^r \equiv \alpha_1^{z_1}\alpha_2^{Z_2} v^8 (\text{mod } p).$$

Ta định nghĩa: $b_1 = (y_1 - z_1)(r - s)^{-t} \bmod q$

và $b_1 = (y_2 - z_2)(r - s)^{-t} \bmod q$

Khi đó dễ dàng kiểm tra thấy rằng:

$$v \equiv \alpha_1^{-b_1}\alpha_2^{-b_2} (\text{mod } p)$$

như mong muốn….

Hình 9.5. Sơ đồ định danh Okamoto.

1. Alice chọn các số ngẫu nhiên k_1, k_2, $0 \le k_1, k_2 \le q-1$ và tính:
$$\gamma = \alpha_1^{k_1}\alpha_2^{k_2} \pmod p.$$
2. Alice gửi dấu xác nhận của cô C(Alice) = (ID(Alice),v,s) và γ cho Bob.
3. Bob xác minh chữ kí của TA bằng cách kiểm tra xem có thoả mãn đồng nhất thức:
$$ver_{TA}(ID(Alice),v,s) = true$$
4. Bob chọn số ngẫu nhiên r, $1 \le r \le 2^t$ và đưa nó cho Alice.
5. Alice tính:
$$y_1 = k_1 + a_1 r \bmod q$$
và $$y_2 = k_2 + a_2 r \bmod q$$
và đưa y_1, y_2 cho Bob.
6. Bob xác minh xem:
$$\gamma \equiv \alpha_1^{y_1}\alpha_2^{y_2} v^r \pmod p$$ hay không.

Bây giờ ta tiếp tục chỉ ra cách Alice và Olga cùng tính giá trị c.

Định lý 9.3.

Giả sử Olga biết giá trị γ (mà với nó cô có xác suất giả danh Alice thành công là $\varepsilon \ge 1/2^{t-1}$) trong giao thức xác minh. Khi đó, Alice và Olga có thể cùng nhau tính $\log_{\alpha_1}\alpha_2$ *trong thời gian đa thức với xác suất 1-1/q.*

Chứng minh

Theo định lý 9.2, Olga có khả năng xác định các giá trị b_1 và b_2 sao cho
$$v \equiv \alpha_1^{b_1}\alpha_2^{b_2} \pmod p$$
Giả thiết rằng Alice để lộ các giá trị a_1 và a_2 cho Olga biết. Dĩ nhiên:
$$v \equiv \alpha_1^{a_1}\alpha_2^{a_2} \pmod p$$
vì thế $$\alpha_1^{a_1-b_1} \equiv \alpha_2^{b_2-a_2} \pmod p$$
giả sử rằng (a1,a2) ≠ (b1,b2), khi đó $(a1-b1)^{-1}$ tồn tại và logarithm rời rạc:
$$c = \log_{\alpha_1}\alpha_2 = (a_1-b_1)(b_2-a_2)^{-1} \bmod q$$
có thể tính được trong thời gian đa thức.

Phần còn lại là xem xét xác suất để (a1,a2) = (b1,b2). Nếu xảy ra điều này thì giá trị c không thể tính theo mô tả ở trên. Tuy nhiên, ta sẽ chỉ ra rằng (a1,a2) = (b1,b2) sẽ chỉ xảy ra với xác suất rất bé 1/q, vì thế giao thức nhờ đó Alice và Olga tính được c sẽ hầu như chắc chắn thành công.

Định nghĩa:
$$A = \{ (a_1', a_2') \in Z_p \times Z_q : \alpha_1^{-a_1'}\alpha_2^{-a_2'} \equiv \alpha_1^{-a_1'}\alpha_2^{-a_2'} \pmod p \}$$
Nghĩa là A gồm tất cả các cặp được sắp có thể và chúng có thể là các số mũ mật của Alice. Xét thấy rằng:
$$A = \{a_1 - c\theta, a_2 + \theta : \theta \in Z_P\},$$

Trong đó c = $\log_{\alpha_1} \alpha_2$. Như vậy A chứa q cặp được sắp.

Cặp được sắp (b1,b2) do Olga tính chắc chắn ở trong tập A. Ta sẽ chỉ ra rằng, giá trị của cặp (b1,b2) độc lập với cặp (a1,a2) chứa các số mũ mật của Alice. Vì (a1,a2) được Alice chọn đầu tiên một cách ngẫu nhiên nên xác suất để (a1,a2) = (b1,b2) là 1/q.

Như vậy, (b1,b2) là "độc lập" với (a1,a2). Cặp (a1,a2) của Alice là một trong q cặp được sắp có thể trong A và không có thông tin nào về nó (là cặp "đúng") đã bị Alice để lộ cho Olga biết khi cô xưng danh với Olga. (Một cách hình thức, Olga biết một cặp trong A chứa số mũ của Alice song cô ta không biết nó là cặp nào).

Ta hãy xét thông tin được trao đổi trong giao thức định danh. Về cơ bản, trong mỗi lần thực hiện giao thức, Alice chọn γ, Olga chọn r và Alice để lộ y_1 và y_2 sao cho:

$$\gamma = \alpha_1^{y_1} \alpha_1^{y_2} v^r \pmod p.$$

Ta nhớ lại rằng, Alice tính:

$$y_1 = k_1 + a_1 r \bmod q$$

và $$y_2 = k_2 + a_2 r \bmod q$$

trong đó

$$\gamma = \alpha_1^{k_1} \alpha_1^{k_2} \bmod q$$

Chú ý rằng k_1 và k_2 không bị lộ (mà a_1 và a_2 cũng không).

Bốn phần tử cụ thể (γ,r,y_1,y_2) được tạo ra trong thực hiện giao thức tuỳ thuộc vào cặp (a_1,a_2) của Alice vì y_1 và y_2 được định nghĩa theo a_1 và a_2. Tuy nhiên ta sẽ chỉ ra rằng, mỗi bộ bốn như vậy có thể được tạo ra như nhau từ cặp được sắp bất kì khác $(a'_1, a'_2) \in A$. Để thấy rõ, giả thiết $(a'_1, a'_2) \in A$, tức là $a'_1 = a_1 - c\theta$ và $a'_2 = a_2 + \theta$, trong đó $0 \le \theta \le q - 1$.

Có thể biểu diễn y_1 và y_2 như sau:

$$\begin{aligned} y_1 &= k_1 + a_1 r \\ &= k_1 + (a'_1 + c\theta) r \\ &= (k_1 + rc\theta) + a'_1 r \end{aligned}$$

và

$$\begin{aligned} y_2 &= k_2 + a_2 r \\ &= k_2 + (a'_2 - \theta) r \\ &= (k_2 - r\theta) + a'_2 r \end{aligned}$$

Trong đó tất cả các phép tính số học đều thực hiện trong Z_p. Nghĩa là bộ bốn (γ,r,y_1,y_2) cũng phù hợp với cặp được sắp (a'_1, a'_2) bằng việc dùng các phép chọn ngẫu nhiên $k'_1 = k_1 + rc\theta$ và $k'_2 = r\theta$ để tạo ra γ. Cần chú ý rằng, các giá trị k_1 và k_2 không bị Alice làm lộ nên bộ (γ, r, y_1, y_2) không cho biết thông tin gì về cặp nào trong A được Alice dùng làm số mũ mật của cô. Đây là điều phải chứng minh.…

Việc chứng minh tính an toàn này khá tinh vi và tối ưu. Chắc nó sẽ hữu dụng để lắp mới các đặc điểm của giao thức, dẫn tới bằng chứng về sự an toàn. Như vậy, Alice chọn 2 số mũ mật cao hơn là chọn một. Có tổng cộng q cặp trong A tương đương với cặp (a_1,a_2) của Alice. Điều này dẫn đến mâu thuẫn cơ bản là, việc hiểu biết hai cặp khác nhau trong A sẽ cho một phương pháp hiệu quả tính toán logarithm rời rạc c. Alice dĩ nhiên chỉ biết một cặp trong A; nếu ta chứng minh rằng Olga có thể giả danh Alice thì Olga có thể tính một cặp trong A khác với cặp của Alice (với xác suất cao). Như vậy Alice và Olga có thể cùng nhau tìm hai cặp trong A và tính c - cho mâu thuẫn như mong muốn.
Dưới đây là một ví dụ nhỏ minh hoạ việc Alice và Olga tính toán $\log_{\alpha_1} \alpha_2$:

Ví dụ 9.5.

Giống như trong ví dụ 9.4, ta lấy p =88667, q = 1031, t = 10 và giả sử v = 13078.
Giả thiết Olga đã xác định được rằng:

$$\alpha_1^{131}\alpha_2^{287}v^{489} \equiv \alpha_1^{890}\alpha_2^{303}v^{199} \pmod p$$

Khi đó cô tính:

$$b_1 = (131 - 890)(489 - 199)^{-1} \bmod 1031 = 456$$

và

$$b_2 = (287 - 303)(489 - 199)^{-1} \bmod 1031 = 519$$

Dùng các giá trị a_1 và a_2 do Alice đưa cho, giá trị c tính như sau:

$$c = (846 - 456)(519 - 515)^{-1} \bmod 1031 = 613$$

giá trị thực tế này là $\log_{\alpha_1} \alpha_2$ mà có thể xác minh bằng cách tính:

$$58902^{613} \bmod 88667 = 73611.$$

Cuối cùng, cần nhấn mạnh rằng, mặc dù không có chứng minh đã biết nào chứng tỏ sơ đồ Schnorr an toàn (thậm chí giả thiết rằng, bài toán logarithm rời rạc không giải được) song ta cũng không biết bất kì nhược điểm nào của sơ đồ này. Thực sự sơ đồ Schnorr được ưa thích hơn sơ đồ Okamoto do nó nhanh hơn.

9.4 Sơ đồ định danh Guillou - quisquater.

Trong phần này sẽ mô tả một sơ đồ định danh khác do Guillou và Quisquater đưa ra dựa trên RSA.

Việc thiết lập sơ đồ như sau: TA chọn 2 số nguyên tố p và q và lập tích n =pq. Giá trị của p và q được giữ bí mật trong khi n công khai. Giống như trước đây, p và q nên chọn đủ lớn để việc phân tích n không thể thực hiện được. Cũng như vậy, TA chọn số nguyên tố đủ lớn b giữ chức năng tham số mật như số mũ mật trong RSA. Giả thiết b là số nguyên tố dài 40 bít. Cuối cùng TA chọn sơ đồ chữ kí và hàm hash.

Hình 9.6: Phát dấu xác nhận cho Alice

1. TA thiết lập định danh cho Alice và phát chuỗi định danh ID(Alice).

2. Alice chọn bí mật một số nguyên u, trong đó $0 \le u \le n-1$. Alice tính:
$$v = (u^{-1})^b \bmod n$$
và đưa u cho TA
3. TA tạo ra chữ kí:
$$s = sig_{TA}(I,v)$$
Dấu xác nhận:
$$C(Alice) = (ID(Alice), v, s)$$
và đưa cho Alice

Dấu xác nhận do TA phát cho Alice được xây dựng như mô tả trong hình 9.6. Khi Alice muốn chứng minh danh tính của cô cho Bob, cô thực hiện giao thức hình 9.7. Ta sẽ chứng minh rằng, sơ đồ Guillou - Quisquater là đúng đắn và đầy đủ. Tuy nhiên, sơ đồ không được chứng minh là an toàn (mặc dù giả thiết hệ thống mã RSA là an toàn).

Ví dụ 9.6:

Giả sử TA chọn p = 467, q = 479, vì thế n = 223693. Giả sử b = 503 và số nguyên mật của Alice u = 101576. Khi đó cô tính:
$$\begin{aligned} v &= (u^{-1})^b \bmod n \\ &= (101576^{-1})^{503} \bmod 223693 \\ &= 24412. \end{aligned}$$

Hình 9.7: Sơ đồ định danh Guillou - Quisquater.

1. Alice chọn số ngẫu nhiên k, trong đó $0 \le k \le n-1$ và tính:
$$\gamma = k^b \bmod n$$
2. Alice đưa cho Bob dấu xác nhận của cô C(Alice) = (ID(Alice), v, s) và γ.
3. Bob xác minh chữ kí của TA bằng cách kiểm tra xem có thoả mãn hay không đồng dư thức:
$$ver(ID(Alice), v, s) = true.$$
4. Bob chọn số ngẫu nhiên r, $0 \le r \le b-1$ và đưa nó cho Alice.
5. Alice tính:
$$y = k\, u^r \bmod n$$
và đưa y cho Bob
6. Bob xác minh rằng
$$\gamma \equiv v^r y^b \pmod n$$

Giả sử Bob trả lời bằng yêu cầu r = 375. Khi đó Alice sẽ tính
$$\begin{aligned} y &= k u^r \bmod n \\ &= 187485 \times 101576^{375} \bmod 223693 \\ &= 93725 \end{aligned}$$
và đưa nó cho Bob. Bob xác minh thấy:
$$24412 \equiv 89888^{375} 93725^{503} \pmod{223693}$$
vì thế Bob chấp nhận bằng chứng về danh tính của Alice. …

Giống như trường hợp tổng quát, việc chứng minh tính đầy đủ rất đơn giản:

$v^r y^b \equiv (u^{-b})^r (ku^r)^b \pmod n$

$\equiv u^{-br} k^b u^{br} \pmod n$

$\equiv k^b \pmod n$

$\equiv \gamma \pmod n$

Bây giờ ta xét đến tính đúng đắn. Ta sẽ chứng minh sơ đồ là đúng đắn miễn là không dễ dàng tính được u từ v. Vì v được lập từ u bằng phép mã RSA nên đây là giả thiết có vẻ hợp lý.

Định lí 9.4

Giảsử Olga biết giá trị γ nhờ nó cô có xác suất thành công trong việc giả danh Alice là $\varepsilon > 1/b$ trong giao thức xác minh. Khi đó Olga có thể tính u trong thời gian đa thức.

Chứng minh

Với γ nào đó, Olga có thể tính giá trị y_1, y_2, r_1, r_2 với $r_1 \neq r_2$ sao cho:

$$\gamma \equiv v^{r_1} y^b \equiv v^{r_2} y_2^b \pmod n$$

không mất tính tổng quát, giả sử rằng $r_1 > r_2$. Khi đó ta có:

$$v^{r_1 - r_2} \equiv (y_2 / y_1)^b \pmod n$$

vì 0 < r1- r2 <b và b là số nguyên tố nên $t = (r_1 - r_2)^{-1} \bmod b$ tồn tại và Olga có thể tính nó trong thời gian đa thức bằng thuật toán Euclidean. Vì thế ta có:

$$v^{(r_1 - r_2)t} \equiv (y_2 / y_1)^{bt} \pmod n.$$

xét thấy, $(r_1 - r_2)t = lb + 1$

với số nguyên dương *l* nào đó, vì thế:

$$v^{lb+1} \equiv (y_2/y_1)^{bt} \pmod n$$

hay tương đương,

$$v \equiv (y_2/y_1)^{bt} (v^{-1})^{lb} \pmod n.$$

Nâng cả hai vế lên luỹ thừa $b^{-1} \bmod \phi(n)$ ta có:

$$u^{-1} \equiv (y_2/y_1)^t (v^{-1})^l \pmod n.$$

cuối cùng tính modulo đảo của cả hai vế của đồng dư thức này, ta nhận được công thức sau cho u:

$$u = (y_2/y_1)^t v^l \bmod n$$

Olga có thể dùng công thức này để tính u trong thời gian đa thức. …

Ví dụ 9.7

Giống như ví dụ trước, giả sử rằng n = 223963, b = 503, u = 101576 và v = 89888. Giả thiết Olga nghiên cứu thấy rằng:

$v^{401} 103386^b \equiv v^{375} 93725^b \pmod n$

Trước tiên cô tính:

$t = (r_1 - r_2)^{-1} \bmod b$

$= (401 - 375)^{-1} \bmod 503$

$= 445$

Tiếp theo cô tính:

$l = ((r_1 - r_2)t - 1)/b$

$= ((401 - 375)445 - 1)/503 = 23$

Cuối cùng cô có thể nhận được giá trị u mật như sau:

$u = (y_1/y_2)^t v^l \bmod n$

$= (103386/93725)^{445} 89888^{23} \bmod 233693$

$= 101576$

và như vậy, số mũ mật của Alice đã bị lộ. …

9.4.1Sơ đồ định danh dựa trên tính đồng nhất.

Sơ đồ định danh Guillou - Quisquater có thể chuyển thành sơ đồ định danh dựa trên tính đồng nhất. Điều này có nghĩa là không cần các dấu xác nhận. Thay vào đó, TA tính giá trị của u như một hàm của chuỗi ID của Alice bằng cách dùng một hàm hash công khai h trong phạm vi Z_n như chỉ ra trên hình 9.8. Giao thức định danh lúc này làm việc như mô tả trong hình 9.9. Giá trị v được tính từ chuỗi ID của Alice thông qua hàm hash công khai. Để tiến hành giao thức định danh, Alice cần biết giá trị u, giá trị này chỉ TA là có thể tính được (giả thiết hệ thống mã khoá công khai RSA là an toàn). Nếu Olga cố tự xưng mình là Alice cô sẽ không thành công nếu không biết u.

Hình 9.8: Phát giá trị u cho Alice

1. Thiết lập danh tính của Alice và phát chuỗi định danh ID(Alice):
2. TA tính

   $u = (h(ID(Alice)^{-1})^a \bmod n$

   và đưa u cho Alice

Hình 9.9: Sơ đồ định danh dựa trên tính đồng nhất Guillou - Quisquater.

1. Alice chọn một số ngẫu nhiên k, $0 \le k \le n - 1$ và tính:

   $\gamma = k^b \bmod n$
2. Alice đưa ID(Alice) và γ cho Bob
3. Bob tính:

   $v = h(ID(Alice))$
4. Bob chọn số ngẫu nhiên r, $0 \le r \le b-1$ và đưa nó cho Alice.
5. Alice tính:

   $y = ku^r \bmod n$

   và đưa y cho Bob
6. Bob xác minh xem có thoả mãn hay không điều kiện dưới đây:

   $\gamma = v^r y^b (\bmod n)$

9.5 Chuyến sơ đồ định danh thành sơ đồ chữ kí.

Có một phương pháp chuẩn để chuyển sơ đồ định danh thành sơ đồ chữ kí. ý tưởng cơ bản là thay thế người xác minh (Bob) bằng hàm hash công khai h. Trong sơ đồ chữ kí thực hiện theo phương pháp này, thông báo không

bị chặt ra (băm) trước khi được kí: Quá trình băm được tích hợp thành thuật toán kí.

Sau đây sẽ minh hoạ biện pháp này bằng việc chuyển sơ đồ Schnorr thành sơ đồ chữ kí (hình 9.10). Thực tế, có khả năng đưa hàm hash h vào SHS và làm giảm được modulo q. Do SHS tạo ra xâu bit có độ dài 160 và q là số nguyên tố 160 bit, nên việc giảm được modulo q chỉ cần thiết khi bản tóm lược của thông báo do SHS tạo ra vượt quá q. Thậm chí trong trường hợp này, chỉ cần trừ q khỏi kết quả.

Trong quá trình chuyển từ sơ đồ định danh thành sơ đồ chữ kí, ta đã thay yêu cầu 40 bit bằng bản tóm lược thông báo 160 bit, 40 bit là đủ đối với một yêu cầu (challenge) vì kẻ mạo danh cần có khả năng phỏng đoán yêu cầu để tính trước câu trả lời (mà sẽ được chấp nhận). Song trong phạm vi của sơ đồ chữ kí, ta cần cac bản tóm lược thông báo có kích thước lớn hơn nhiều để ngăn chặc sự tấn công thông qua việc tìm kiếm các va chạm trong hàm hash.

Hình 9.10: Sơ đồ chữ kí Schnorr.

Cho p là số nguyên tố 512 bít sao cho bài toán logarithm rời rạc trong Z_P là không giải được; cho q là số nguyên tố 160 bít chia hết cho p-1. Giả sử $\alpha \in Z_p^*$ là căn bậc q của 1modulo p. Cho h là hàm hash trong phạm vi Z_p^*. Định nghĩa $P = Z_p^*$.$A = Z_p^* \times Z_P$ và định nghĩa:

$K = \{(p, q, \alpha, a, v) : v \equiv \alpha^{-a} (\text{mod } p)\}$

Các giá trị p, q, α và v là công khai còn a mật.

Với K = (p, q, α, a, v) và với số ngẫu nhiên k mật $\in Z_p^*$, ta định nghĩa:

$sig_K(x,k) = (\gamma, y)$

trong đó

$\gamma = \alpha^k \bmod p$

và

$y = k + ah(x,\gamma) \bmod q.$

với $x, \gamma \in Z_p^*$ và $y \in Z_P$, định nghĩa

$ver(x, \gamma, y) = \text{true} \Leftrightarrow \gamma \equiv \alpha^y v^{h(x,y)} (\text{mod } p)$

9.6 Các chú giải và tài liệu tham khảo

Sơ đồ định danh Schnorr nêu trong tài liệu [Sc91], sơ đồ Okamoto được đưa ra trong [OK93] còn sơ đồ Guillou - quisquater có thể tìm thấy trong [GQ88]. Một sơ đồ khác chứng minh sự an toàn dưới giả thiết tính toán hợp lý là của Brickell và McCurley [BM92].

Các sơ đồ định danh phổ biến khác chứa đựng trong sơ đồ Fiege - Fiat - Shamir [FFS88] và sơ đồ nhân hoán vị [SH90]. Sơ đồ Fiege - Fiat - Shamir được chứng minh là an toàn nhờ dùng kĩ thuật tri thức không.

Phương pháp xây dựng sơ đồ chữ kí từ các sơ đồ định danh là do Fiat và Shamir đưa ra [FS87]. Chúng cũng được mô tả và phiên bản dựa trên tính đồng nhất của sơ đồ định danh của chính họ.

Các tổng quan về các sơ đồ định danh này đã được công bố trong công trình của Burmester, Desmedt và Beth [BDB92] và công trình của Waleffe và Quisquater [DWQ93].

Các bài tập

9.1. Xét sơ đồ định danh sau đây: Alice sở hữu khoá mật n = pq, p và q là những số nguyên tố và $p \equiv q \equiv 3 \pmod 4$. Các giá trị n và ID(Alice) đều do TA kí như thường lệ và lưu trên dấu xác nhận của Alice. Khi Alice muốn tự xưng danh với Bob, Bob sẽ đưa cho Alice một thặng dư bình phương theo modulo n gọi là x. Sau đó Alice sẽ tính căn bình phương y của x và đưa nó cho Bob. Khi đó Bob xác minh xem y^2 có $\equiv x \pmod n$ hay không. Hãy giải thích tại sao sơ đồ này không an toàn.

9.2. Giả sử Alice đang dùng sơ đồ Schnorr với q = 1201, p =122503, t = 10 còn α= 11538.

a/ Hãy kiểm tra xem α có bậc q trên Z_p^* không.

b/ Giả thiết số mũ mật của Alice là α = 357, hãy tính v.

c/ Giả sử k = 868, hãy tính γ.

d/ Giả sử Bob đưa ra yêu cầu r = 501, hãy tính câu trả lời y của Alice.

e/ Thực hiện các tính toán của Bob khi xác minh y

9.3. Giả thiết, Alice dùng sơ đồ Schnorr với p, q, t như trong bài tập 9.2. v=51131 và giả sử Olga có thể nghiên cứu thấy rằng:

$$\alpha^3 v^{148} \equiv \alpha^{151} v^{1077} \pmod p$$

Hãy chỉ ra cách Olga có thể tính số mũ mật a của Alice.

9.4. Giả sử Alice đang dùng sơ đồ Okamoto với q = 1201, p = 122503, t= 10, α_1=60497 và α_2 = 17163

a/ Giả sử các số mũ mật của Alice a_1=432, a_2 = 423, hãy tính v.

b/ Giả sử k_1 = 389, k_2 = 191, tính γ

c/ Giả thiết Bob đưa ra yêu cầu r = 21. Hãy tính câu trả lời y_1 và y_2 của Alice

d/ Thực hiện các tính toan của Bob để xác minh $y_1$và y_2.

9.5/ Cũng giả thiết rằng Alice dùng sơ đồ Okamoto với p, q, t, $\alpha_1$và α_2 như trong bài tập 9.4, và v = 119504.

a/ Hãy kiểm tra xem phương trình sau có thoả mãn không:

$$\alpha_1^{70} \alpha_2^{1033} v^{877} = \alpha_1^{248} \alpha_2^{883} v^{992} \pmod p$$

b/ Dùng thông tin trên để tính b_1 và b_2 sao cho:

$$\alpha_1^{-b_1}\alpha_2^{-b_2} \equiv v(\text{mod } p)\,.$$

c/ Giả sử rằng Alice để lộ α_1=484 và α_2 =935. Hãy chỉ ra cách Alice và Olga cùng nhau tính $\log_{\alpha_1}\alpha_2$.

9.6. Giả sử rằng, Alice đang dùng sơ đồ Quisquater với p = 503, q = 379 và b= 509.

a/ Giả sử giá trị u mật của Alice = 155863 tính v.

b/ Giả sử k = 123845, hãy tính γ.

c/ Giả thiết Bob đưa ra yêu cầu r = 487. Hãy tính câu trả lời y của Alice

d/ Thực hiện các tính toán của Bob để xác minh y

9.7. Giả sử Alice đang dùng sơ đồ Quisquater với n = 199543, b = 523 và v=146152. Giả thiết Olga đã khám phá ra rằng

$v^{456}101360^b = v^{257}36056^b(\text{mod } n)$

Hãy nêu cách Olga có thể tính u.

CHƯƠNG 10

CÁC MÃ XÁC THỰC

10.1 MỞ ĐẦU

Ta đã dành nhiều thời gian để nghiên cứu các hệ mật được dùng để đảm bảo độ mật .Mã xác thực sẽ cung cấp phương pháp bảo đảm tình toàn vẹn của bản tin,mghĩa là bản tin phải không bị can thiệp một cách bất hợp pháp và nó thực sự được gửi đi từ mày phát.

Mục đích của chương này là phải có được khả năng xá thực ngay cả khi có một đối phương tích cực-Oscar là người có thể quan sát các bản tin trong kênh.Mục đích này có thể đạt được bằng cách thiết lập một ‘’khoa riêng’’K bằng cách để Alice và Bob chungchung một khoá bí mật trước hki mỗi bản tin được gửi đi.

Trong chương này ta sẽ nghiên cứu đảm bảo xacs thực chứ không phải các mã đảm bảo độ mật.Trong mã này,khoá sẽ được dùng dể tính một mã xác thực cho phép Bob kiểm tra được tính xác thực của thông báo mà anh ta nhận được.Một ứng dụng khác của mã xác thực là để kiểm tra xem các số liệu trong một file lớn có bị can thiệp vào một cách hợp pháp hay không.Nhãn xác thực sẽ được lưu cùng với số liệu:KHOá ĐƯẻc dùng để tạo và kiểm tra dấu xác thực được lưu một cách tách bạch trong một’’vùng’’an toàn.

Ta cũng sẽ chỉ ra rằng,về nhiều khía cạnh mã xác thực cũng tương tự như một sơ đồ chữ kí hoặc tương tự như một maw xác thực thông báo(MAC).Sự khác biệt chính là sự an toàn của một maw xác thực là không điều kiện biên,trong khi đó các sơ đồ chữ kí và MAC lại được nghiên cứu theo quan điểm độ an toàn tính toán.Cũng vậy,khi một maw xác thực (hoặc MAC) được dùng,một bản tin chỉ có thể được kiểm tra bởi người nhận hợp pháp.Trong khi đó baats cứ mỗi ai cũng có thể xác minh được chữ kí bằng cách dùng một thuật toán xác minh công khai.

Bây giờ ta sẽ đưa ra một định nghia hình thức cho thuật ngữ được sử dụng khi nghiên cứu các mã xác thực.

Định nghĩa 10.1

Một mã xác thực là một bộ 4(S,R,K,C)thoả mãn các điều kiện sau :

1. *S là tập hữu hạn các trạng thái nguồn có thể*

2. *A là tập hợp các nhãn xác thực có thể*
3. *K là một tập hữu hạn các khoá có thể (không gian khoá)*
4. *Với mỗi $k \in K$ có một quy tắc xác thực $e_k : S \rightarrow R$*

Tập bản tin được xác định bằng $M = S \rightarrow R$

Nhận xét:

Chú ý một trạng thái nguồn tương đương với một bản rõ.Một bản tin gồm một bản rõ với một nhãn xác thực kèm theo,một cách chính xác hơn có thể coi đó là là một bản tin đã được xác nhận.Một quy tắc xác thực không nhất thiết phải là hàm đơn ánh.

Đểê phát một thông báo (đã được kí).Alice và Bob phải tuân theo giao thức sau.Trước tiên họ phải chộn một khoá ngẫu nhiên $K \in K$.Điều này được thuwc hiện một cách bí mật như trong hệ mật khoá bi mật.Sau đó giả sử rằng Alice muốn gửi một trạng thái nguồn $s \in S$ cho Bob trong một kênh không an toàn>Alice sẽ tính $a = e_k(s)$ và gửi bản tin (s,a)cho Bob.Khi nhận được (s,a) Bob tính $a' = e_K(s)$.Nếu a=a' thì Bob chấp nhận bản tin là xác thực,ngược lại Bob sẽ loại bỏ nó.

Ta sẽ nghiên cứu hai kiểu tấn công khác nhau mà Oscar có thể tiến hành.Trong cả hai loại này,Oscar sẽ là''kẻ xâm nhập vào giưa cuộc''.Các phép tấn công này được mô tả như sau:

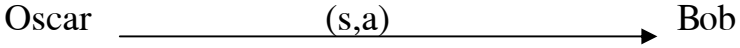
Giả mạo

Oscar đưa ra một bản tin (s,a) vào kênh và hi vọng nó sẽ được chấp nhận .Phương pháp này được mô tả trong hình 10.1.

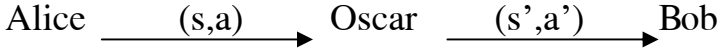
Thay thế

Oscar quan sát một bản tin trong (s,a)kênh ,sau đó anh ta biến đổi nó thành(s',a'),trong đó s'=s và hi vọng được Bob chấp nhận như một bản tin xác thực .Bởi vậy anh ta tin sẽ lái được Bob đi tới trạng thái nguồn mới này.Phương pháp này được mô tả như hình 10.2.

Hình 10.1. Việc giả mạo bởi Oscar



Hình 10.2 . Phép thay thế của Oscar.



Gắn với mỗi phong pháp này là một xác xuất lừa bịp,là xác suất để Oscar thành công trong việc lừa Bob nếu anh ta (Oscar) tuân thủ một chiến lược tối ưu .Các xác suất này được kí hiệu là Pd_0(trường hợp giả mạo)và Pd_1(trường hợp thay thế) .Để tình Pd_0 và Pd_1 ta cần phải xác định các phân bố xác suất trên *S* và*K*.Các xác suất này được kí hiệu tương ứng là p_s và p_k .

Giả sử rằng Oscar đã biết mã xác thực và hai phân bố xác suất này.Chỉ có một thông tin mà Alice và Bob có nhưng mà Oscar không được biết là giá trị của khoá K .Điều này tương tự với cách mà chúng ta đã nghiên cứu độ an toàn không điều kiện của các hệ mật khoá bí mật.

10.2.TÍNH XÁC SUẤT LỪA BỊP

Trong phần này sẽ xét đến việc tính các xác suất lừa bịp.Ta bắt đầu về một mã xác thực.

Ví dụ 10.1

Giả sử $K=R=Z$

và $K=Z_3xZ_3$

Với mỗi $(i,j)\in K$ và mỗi $s\in S$ ta xác định

$$e_k(s) = i.s+j \bmod 3$$

Để thuận tiện cho việc nghiên cứu ta dùng ma trận xác thực (ma trận này tạo bằng tất cả các giá trị $e_k(s)$).Với mỗi khoá $K\in K$ và với mỗi $s\in S$ ta đặt nhãn xác thực $e_k(s)$ vào hàng K và cột s của một ma trận M kích thước *K* x*S* .Mảng M được mô tả trên hình 10.3.

Hình 10.3.Ma trận xác thực

Khoá	0	1	2
(0,0)	0	0	0
(0,1)	1	1	1
(0,2)	2	2	2
(1,0)	0	1	2
(1,1)	1	2	0
(1,2)	2	0	1
(2,0)	0	1	2
(2,1)	1	0	2
(2,2)	2	1	0

Giả sử rằng khoá được chọn một cách ngẫu nhiên,tức là $p_k(K)=1/9$ đối với mọi $K\in K$. Ta không phải xác định phân bố xác suất p_S vì trong thí dụ này nó khong có ý nghĩa gì.

Trước tiên xét cách tấn công giả mạo,Oscar sẽ chọn ra một trạng thái nguồn s và cố gắng phỏng ddoand\s một nhãn xác thực ''đúng''.Kí hiệu K_0 là khoá đang sử dụng (mà Oscar không biết).ócả sẽ thành công trong việc đánh lừa Bob nếu anh ta phỏng đoán $a_0=e_{K0}(s)$.Tuy nhiên với bất kì $s\in S$ và $a\in R$ dễ dàng thấy rằng ,chỉ có đúng 3(chứ không phải là 9)quy tắc xác thực $K\in K$ sao cho $e_k(s)$ =a.(Nói cách khác mỗi kí hiệu chỉ xuất hiện 3 lần trong mỗi cột của ma trận xác thực).Bởi vậy dẫn tới $Pd_0=1/3$.

Phân tích phép thay thế có phức tạp hơn một chút.Giả sử Oscar đã quan sát được trên kênh 1 bản tin (0.0).Nhờ đó anh ta đã biết một thông tin nào đó về khoá:anh ta biết rằng :

$$K_0\in\{(0,0),(1,0),(2,0)\}$$

Bây giờ ,giả sử Oscar thay bản tin (0,0) bằng bản tin (1,1).Khi đó anh ta sẽ lừa bịp thành công khi và chỉ khi $K_0=(1,1)$,xác suất để K_0 là khoá bằng 1/3 vì khoá nằm trong tập {(0,0),(1,0),(2,0)}.

Có thể thực hiện một phân tích tương tự đối với bất kì một phép thay thế nào mà Oscar tiến hành.Nói chung nếu Oscar quan sát một bản tin (s,a) và thấy nó bằng một bản tin bất kì (s',a') trong đó s'=s thì anh ta sẽ đánh lừa được Bob với xác suất 1/3.Ta có thể thấy rõ điều này như sau .Việc quan sát được (s,a) sẽ hạn chế khóa và một trong ba khả năng.Trong khi đó với một phép chọn (s',a') chỉ có một khoá chứ không phải ba khoá có thể)theo quy tắc a là nhãn xác thực của s'.

Bây giờ ta sẽ thảo luận cách tính toán tổng quát cho các xác suất lừa bịp.Trước tiên ta hãy xát Pd_0.Cũng như trên K_0 là khoá được chọn bởi Alice và Bob.Với $s\in S$ và $a\in R$ ta xác định payoff(s,a)là xác suất để Bob chấp nhận bản tin (s,a) là bản tin xác thực .Dễ dàng thấy rằng :

$$\text{Payoff}(s,a) = \text{prob}(a=e_K(s))$$
$$=\sum_{K\in K\ (ek(s)=a)} p_K(K)$$

Nghĩa là payoff(s,a) được tính bằng cách chọn các hàng của ma trận xác thực có phần tử a nằm trong cột s và lấy tổng xác suất của các khoá K tương ứng.

Để cơ hội thành công là lớn nhất.Oscar phỉa chọn (s,a) sao cho payoff(s,a) là cực đại .Bởi vậy:

$$Pd_0 =\max\{\text{payoff}(s,a): s\in S.a\in R\} \qquad (10.1)$$

Chú ý rằng Pd_0 không phụ thuộc vào phân bố xác suất p_S

Việc tính Pd_1 có khó hơn một chút và nó có thể phụ thuộc vào p_S.Trước tiên ta sẽ xét bài toán sau:Giả sử Oscar quan sát được thông báo (s,a) trong kênh.Oscar sẽ thay (s,a) bằng một bản tin (s',a') nào đó ,trong đó $s' \neq s$.Khi đó,với $s,s' \in S$,$s \neq s'$ và $a,a' \in R$ ta định nghĩa payoff(s',a';s,a) là xác suất để phép thay thế (s,a) bằng (s',a') thành công(để đánh lừa Bob) .Khi đó có thể tính như sau :

$$Payoff(s',a';s,a) = prob(a' = e_{Ko}(s') \mid a = e_{Ko}(s))$$

$$= \frac{prob(a' = e_K(s') \Lambda a = e_K(s))}{prob(a = e_K(s))}$$

Tử số của phân số này được tính bằng cách chọn các hàng của ma trận xác thực có giá trị a trong cột s và giá trị a' trong cột s'và lấy tổng các xác suất của các khoá tương ứng.Vì Oscar muốn tăng cực đại cơ hội đánh lừa Bob nên anh ta tính :

$$P_S = \max\{payoff(s',a';s,a); s' \in S, s \neq s', a \in R\}$$

Đại lượng p,kí hiệu để Oscar đánh lừa Bob bằng một phép thay thế khi đã quan sát được bản tin (s,a) trên kênh.

Bây giờ phải làm thế nào để tính để tinhs xác suất lừa bịp Pd_1?Rõ ràng là ở đây ta ta phải tính trung bình các giá trị của lượng p_S theo các xác suất $p_M(s,a)$ quan sát các bản tin trên kênh.Nghĩa là Pd_1 được tính bằng :

$$Pd_1 = \sum_{(S,a) \in M} p_M(s,a).p_M \qquad (10.2)$$

Phân bố xác suất p_M như sau:

$$P_M(s,a) = p_s(s) x\, p_K(a \mid s)$$

$$= p_S(s) x \sum_{(K \in K;\, ek(s)=a)} p_K(K)$$

$$= p_S(s) x payoff(s,a)$$

Trong ví dụ 10.1:

$$Payoff(s,a) = 1/3$$

Với $\forall s',a',s,a, s \neq s'$ .Bởi vậy $Pd_1 = 1/3$ đối với mọi phân tố xác suất p_S (nói chung Pd_1 phụ thuộc vào p_S).

Trong ví dụ sau đây sẽ xét việc tính Pd_0 và Pd_1 .

Ví dụ 10.2:

Xét ma trận trên hình 10.4Giả sử các phân bố xác suất trên *S và K* là:

$$P_S(i) = 1/4$$

$1 \leq i \leq 4$ và

$p_K(1) = 1/2$; $p_K(2) = p_K(3) = 1/4$

Hình 10.4 Ma trận xác thực

Khoa	1	2	3	4
1	1	1	1	2
2	2	2	1	2
3	1	2	2	1

Các giá trị payoff(s,a) như sau :

Payoff(1,1) =3/4 Payoff(1,1) =1/4
Payoff(2,1) =1/2 Payoff(2,2) =1/2
Payoff(3,1) =3/4 Payoff(3,2) =1/4
Payoff(4,1) =1/4 Payoff(4,2) =3/4

Bởi vậy Pd_0=3/4 .Chiến lược đánh lừa tối ưu của Oscar là đưa một thông báo bất kì trong số các thông báo (1,1),(3,1) hoặc (4,2) vào kênh.

Bây giờ ta sẽ chuyển sang tính Pd_1.Trước hết ta đưa các giá trị khác nhau của payoff(s',a';s,a).

	(1,1)	(1,2)	(2,1)	(2,2)	(3,1)	(3,2)	(4,1)	(4,2)
(1,1) (1,2)			2/3 0	1/3 1	2/3 1	1/3 0	1/3 1	2/3 0
(2,1) (2,2)	1 1/2	0 1/2			0 1/2	1 1/2	0 1/2	1 1/2
(3,1) (3,2)	2/3 1	1/3 0	2/3 0	1/3 1			0 1	1 0
(4,1) (4,2)	1 2/3	0 1/3	0 2/3	1 1/3	0 1	1 0		

Như vậy ta có $p_{1.1}$=2/3,$p_{2.2}$=1/2,$p_{3.3}$=1 với mọi giá trị s,a khác .Khi đó việc đánh giá Pd_1 sẽ trở nên rất đơn giản:Pd_1=7/8.Chiến lược thay thế tối ưu của Oscar là:

(1,1) → (2,1)
(1,2) → (2,2)
(2,1) → (1,1)
(2,2) → (1,1)
(3,1) → (4,2)
(3,2) → (1,1)
(4,1) → (1,1)
(4,2) → (3,1)

Chiến lược này thực sự dẫn đến $Pd_1=7/8$
Việc tính toán Pd_1 trong ví dụ 10.2 dễ hiểu nhưng khá dài dòng .Trên thực tế có thể đơn giản hóa việc tính Pd_2 dựa trên nhận xét là ta đã thực hiện việc chia cho đại lượng payoff(s,a) khi tính $P_{s,a}$ và sau đó Lại nhân với payoff(s,a) khi tính Pd_1 .Dĩ nhiên là hai phép tính này loại bỏ nhau.Giả sử định nghĩa :

$q_{s,a}=\max\{\sum_{\{K\in K:ek(s)=a,ek(s')=a'_1\}} p_K(K) : s'\in S, s'\neq s, a'\in A\}$
Với mọi s,a. Khi đó có công thức đơn giản hơn sau:

10.3.CÁC GIỚI HẠN TỔ HỢP

Ta đã thấy ràng độ an toàn của một mã xác định được đo bằng
Các xác xuất lừa bịp . Bởi vậy cần xây dựng các mã sao cho các xác Xuất này nhỏ tới mức có thể .Tuy nhiên những khía canh khác cũng Rất qoan trọng .Ta xem xét một số vấn đề cấn qoan tâm trong mã xác thực .
1.Các xác xuất lừa bịp Pd_0 và Pd_1 phải đủ nhỏ để đạt được mức an toàn mong muốn .
2.số các trạng thái nguồn phải đủ lớn để có thể truyền các thông tin cần thiết bằng cách gán một nhãn xác thực vào một trạng thái nguồn .
3. Kích thước của không gian khóa phải được tối thiểu hóa và các giá trị của khóa phải truyền qua một kênh an toàn (Cần chú ý rằng phải thay đổi khóa sau mỗi lần truyền tin giống như khi dùng OTP).
Trong phần này sẽ xác địinh giới hạn dưới đối với các xác suất lừa bịp và chúng được tính theo các tham số của mã.Hãy nhớ lại rằng ta đã định nghĩa mã xác thực là một bộ bốn (S,R,K,E).Trong phần này ta sẽ ký hiệu $|R|=l$
Giả sử cố định một trạng thái nguồn $s\in S$.Khi đó có thể tính :

$$\sum_{a\in R} payoff(s,a)=\sum_{a\in R}\sum_{(K\in K:ek(s)=a\}} p_K(K)$$
$$=\sum_{K\in K} p_K(K)$$
$$=1$$
Bởi vậy với mỗi $s\in S$,tồn tại một nhãn xác thực a(s) sao cho :
$Payoff(s,a(s))\geq 1/l$.
Dễ dàng rút ra định lý sau:

Đinh lý 10.1

Giả sử (S,R,K,E) là một mã xác thực .Khi đó $Pd_0 \geq 1/l$ *trong đó* $l = |R|$*.Ngoài ra* $Pd_0 = 1/l$ *khi và chỉ khi :*

$$\sum_{\{K \in K : ek(s)=a\}} p(K) = 1/l \qquad (10.4)$$

với mỗi $s \in S, a \in R$.

Baauy giờ ta sẽ chuyển sang phương pháp thay thế .Giả sử cố định s,a và $s', s \neq s'$.Ta có:

$$\sum_{a' \in R} payoff\ (s', a'; s, a) = \sum_{a' \in R} \frac{\sum_{\{K \in K : ek(s)=a, ek(s')=a'\}} p_K(K)}{\sum_{\{K \in K : ek(s)=a\}} p_K(K)}$$

$$= \frac{\sum_{\{K \in K : ek(s)=a\}} p_K(K)}{\sum_{\{K \in K : ek(s)=a\}} p_K(K)} = 1$$

Như vậy tồn tại một nhãn thực $a'(s', s, a)$ sao cho :

Payoff$(s', a'(s', s, a) : s, a) \geq 1/l$

Định lý sau sẽ rút ra kết quả :

Định lý10.2

Giả sử (S,R,K,E) là một mã xác thực .Khi đó $Pd_1 >= 1/l$ *trong đó* $L = |R|$*.Ngoài ra* $Pd_1 \geq 1/l$ *khi và chỉ khi :*

$$\frac{\sum_{\{K \in K : ek(s)=a, ek(s')=a'\}} p_K(K)}{\sum_{\{K \in K : ek(s)=a\}} p_K(K)} = 1/l$$

Với mỗi $s, s' \in S, s = s', a, a' \in R$

Chứng minh

Ta có : $Pd_1 = \sum_{(s,a) \in M} p_M(s,a).p_{s,a} \geq \sum_{(s,a) \in M} p_M(s,a)/l = 1/l$

Ngoài ra dấu bằng chỉ tồn tại khi và chỉ khi $p_{s,a} = 1/l$ với mỗi (s,a) .Tuy nhiên điều kiện này lại tương đương với điều kiện :

Payoff$(s', a'; s, a) = 1/l$ với mọi (s,a).

Định lý 10.3

Giả sử (S,R,K,E) là một mã xác thực trong đó $l = |R|$*.Khi đó*$Pd_0 = Pd_1 = 1/l$ *khi và chỉ khi :*

$$\sum_{\{K \in K : ek(s)=a, ek(s')=a'\}} p_K(K) = 1/l^2 \qquad (10.6)$$

Với mọi $s,s' \in S, a,a' \in R, s \neq s'$
Chứng minh
Các phương trình (10.4)và (10.5) boa hàm phương trình (10.6).Ngược lại , phương trình (10.6) kéo theo các phương trình (10.4) và(10.5).
Nừu các khóa là đồng khả năng thì ta nhận được hệ quả sau:
Hệ quả 10.4:
Giả sử (S,R,K,e) là một mã xác thực ,trong đó l=$|R|$ và các khoá chọn đồng xác suất.Khi đó $Pd_0=Pd_1=1/l$ khi và chi khi :

$$|\{K \in K : e_K(s)=a, e_K(s')=a'\}| = |K|/l^2 \qquad (10.7)$$

Với mọi $s,s' \in S, s' \neq s, a,a' \in R$.

10.3.1.Các mạng trực giao

Trong phần này ta xét các mối liên quan giưa các mã xác thực và các cấu trúc tổ hợp được gọi là các mảng trực giao.Trước tiên ta sẽ đưa ra các định nghĩa:

Định nghĩa 10.2:
Một mạng trực giao 0A(n,k,λ)là một mảng kích thước $\lambda n^2 xk$ *chứa n kí hiệu sao cho trong hai cột bất kì của mảng mỗi cặp trong* n^2 *cặp kí hiệu chỉ xuất hiện trong đúng* λ *hàng.*
Các mạng trực giao là các cấu trúc đã được nghiên cứu kĩ trong lí thuyets thiết kế tổ hợp và tương đương với các cấu trúc khác như các hình vuông Latinh trực giao hỏi các lưới ...
Trong hình 10.5 ta đưa ra một mảng trực giao 0A(3.3.1) nhận được từ ma trận xác thực ở hình 10.3.

Hình 10.5. 0A(3.3.1)

$$\begin{bmatrix} 0 & 0 & 0 \\ 1 & 1 & 1 \\ 2 & 2 & 2 \\ 0 & 1 & 2 \\ 1 & 2 & 0 \\ 2 & 0 & 1 \\ 0 & 2 & 1 \\ 1 & 0 & 2 \\ 2 & 1 & 0 \end{bmatrix}$$

Có thể dùng một mảng trực giao bất kì 0A(n,k,λ) để xây dựng một mã xác thực có $Pd_0=Pd_1=1/n$ như được nêu trong định lí sau:

Định lí 10.5.
Giả sử có một mảng trực giao 0A(n,k,λ).Khi đó cùng tồn tại một mã xác thực (S,A,K,E).trong đó $|S|=k, |R|=n, |K|=\lambda n^2$ *và* $Pd_0=Pd_1=1/n$.
Chứng minh:
Hãy dùng mỗi hàng của mảng trực giao làm một quy tắc xác thực với xác suất như nhau bằng $1/(\lambda n^2)$.Mối liên hệ tương ứng giưa mảng trực giao và mã xác thực được cho ở bảng dưới đây.Vì phương trình (10.7) được thoả mãn nên ta có thể áp dụng hệ quả 10.4 để thu được một mã xác thực có các tính chất đã nêu.

Mảng trực giao	Mã xác thực
Hàng	Quy tắc xác thực
Cột	Trạng thái nghuồn
Kí hiệu	Nhãn xác thực

10.3.2.Phương pháp xây dựng và các giới hạn đối với các 0A

Giả sử ta xây dựng một mã xác thực từ một 0A(n,k,λ).Tham số n sẽ xác định số các nhãn (tức là độ an toàn của mã).Tham số k xác định số các trạng thái nguồn mà mã có thể thích ứng.Tham số λ chỉ quan hệ tới số khoá (là λn^2).Dĩ nhiên trường hợp λ=1là trường hợp mong muốn nhất tuy nhiên ta sẽ thấy rằng đôi khi cần phải dùng các mảng trực giao có λ lớn hơn.Giả sử ta muốn xây dựng một mã xác thực ới tập nguồn xác định S và có một mức an toàn ε xác định (tức là để $Pd_0<\varepsilon$ và $Pd_1<\varepsilon$).Khi đó mảng trực giao thích hợp phải thoả mãn các điều kiện sau:

1. $n\geq 1/\varepsilon$
2. $k\geq |S|$.(Xét thấy có thể loại một hoặc một số cột khỏi mảng trực giao và mảng kết quả vẫn còn là một mảng trực giao,bởi vậy không đòi hỏi $k=|S|$).
3. λ được tối thiểu hoá ,tuỳ thuộc vào các điếu kiện trên được thoả mãn

Trước tiên xét các mảng trực giao có λ=1 .Với một giá trị n cho trước ,ta cần làm cực đại hoá số cột,sau đây là một số điều kiện cần để tồn tại .

Định lí 10.6.

Giả sử tồn tại một 0A(n,k,λ) .Khi đó k≥ n+1

Chứng minh:

Cho A là một 0A(n,k,l) trên tập kí hiệu X={0,1...n-1}.Giả sử π là một phép hoán vị của X và ta hoán vị các kí hiệu trong một cột bất kì của A theo phép giao hoán π.Kết quả là ta lại có một 0A(n,k,l).Bởi vậy bằng cách áp dụng liên tiếp các phép vị kiểu này ,có thể xem (mà không làm mất tính tổng quát) rằng hàng đầu tiên cuả A là (00...0).

Tiếp theo ta sẽ chỉ ra rằng mỗi kí hiệu chỉ xuất hiện đùng n lần trong mỗi cột của A.Hãy chọn hai cột (chẳng hạn c và c')và cho X là một kí hiệu bất kì .Khi đó với mỗi kí hiệu x' tồn tại một hàng duy nhất của A trong đó x ở cột c và x' ở cột c'.Cho x' thay đổi trên X ta thấy rằng x xuất hiện đúng n lần trong cột c.

Vì hàng thứ nhất là (00...0) nên ta đã vét cạn các khả năng xuất hiện của các cặp được sắp (0.0).Bởi vậy không có một hàng nào khác có nhiều hơn một kí hiệu o.Bây giờ ta sẽ đếm số các hàng chứa ít nhất một kí hiệu 0.Tổng số là 1+k(n-1).Tuy nhiên tổng này không thể lớn hơn tổng số các hàng trong A (bằng n^2).Bởi vậy $1+k(n-1)\leq n^2$ hay $k\leq n+1$ như mong muốn .

Bây giờ ta sẽ đưa ra một cấu trúc cho mảng trực giao có λ=1 ,trong đó k=n .Trong thực tế đây chính là cấu trúc đã dùng để thu được mảng trực giao nêu ở hình 10.5.

Định lí 10.7

Giả sử p là một số nguyên tố.Khi đó tồn tại một mảng trực giao 0A(p.p.1).

Chứng minh:

Mảng này sẽ là một cấp $p^2\times p$,trong đó các hàng được lập chỉ số trong $Z_P x Z_P$ và các cột được lập chỉ số trong Z_P .Phần tử ở hàng (i,j) và cột x được tính bằng i.x+j mod p.

Giả sử chọn hai cột x và y,x≠y,và hai kí hiệu a,b.Ta cần tìm một hàng duy nhất (i,j) sao cho a nằm trong cột x và y nằm trong cột y của hàng (i,j).Vì thế cần giải hai phương trình:

a=i.x+j

b=i.y+j

theo các ẩn i và j (trong đó tất cả các phép tính số học được thực hiện trong trường Z).Nhưng hệ này có nghiệm duy nhất:

$$i=(a-b)(x-y)^4 \bmod p$$
$$j=a-y.x \bmod p$$

Bởi vậy ta có một mảng trực giao.

Nhận xét rằng một 0A(n,n,1) bất kì có thể mở rộng thêm một cột để tạo thành 0A(n,n+1,1)(xem các bài tập).Vì thế dùng định lí 10.7 có thể nhận được vô hạn các 0A đạt được giới hạn của định lí 10.6 với dấu bằng.

Định lí 10.6 cho biết rằng $\lambda>1$ nếu $k>n+1$.Ta sẽ chứng minh một kết quả tổng quát hơn khi đặt giới hạn dưới của λ như một hàm của n và k.Tuy nhiên,trước tiên cần đưa ra một bất đẳng thức quan trọng sẽ dùng trong chứng minh.

Bổ đề 10.8.

Giả sử $b_1....b_m$ là các số thực.Khi đó:

$$m\sum_{m=1}^{n} b_1^2 \geq (\sum_{m=1}^{n} b_1)^2$$

Chứng minh

áp dụng bất đẳng thức Jensen(Định lí 2.5) với $f(x)=-x^2$ và $a_1=1/m.1\leq i\leq m$.Hàm f là liên tục là và lõm.Vì thế ta nhận được :

$$\sum_{i=1}^{m} \frac{b_1^2}{m} \leq \left(\sum_{i=1}^{m} \frac{b_1}{m}\right)^2$$

Từ đây dễ dàng rút ra kết quả mong muốn.

Định lí 10.9.

Giả sử tồn tại một 0A(n,k,λ).Khi đó

$$\lambda \geq \frac{k(n-1)+1}{n^2}$$

X

Chứng minh

Cho A là một 0A(n,k,λ) trên tập kí hiệu X={0,1.....n-1},trong đó hàng đầu tiên của A là (0,0....0)(giả thiết này không làm mất tính tổng quát như đã thấy trong định lí 10.6).

Kí hiệu các tập hàng của Alà R và r_1 là hàng đầu tiên,cho R_1=R\{r1}.Với một hàng bất r của A,kí hiệu x_r chỉ số lần xuất hiện của 0 trong hàng r.Có thể dễ dàng tình được tổng số lần xuaat hiện của 0 trong R_1.Vì mỗi kí hiệu phải xuất hiện đúng λn lần trong mỗi cột của Anên ta có:

$$\sum_{r\in R_1} x_r = k(\lambda.n-1)$$

Bây giờ số lần xuất hiện cặp được sắp (0,0) ở các hàng trong R_1 là:

$$\sum_{r\in R_1} x_r(x_r-1) = \sum_{r\in R_2} x_r^2 - \sum_{r\in R_1} x_r$$

$$= \sum_{r\in R_2} x_r^2 - k(\lambda.n-1)$$

áp dụng bổ đề (10.8) ta có:

$$\sum_{r\in R_1} x_r^2 \geq \frac{(k(\lambda.n-1))^2}{\lambda.n^2-1}$$

và bởi vậy :

$$\sum_{r\in R_1} x_r(x_r-1) \geq \frac{(k(\lambda.n-1))}{\lambda..n^2-1} - k(\lambda.n-1)$$

Mặt khác,trong một cặp cột cho trước bất kì,cặp được sắp (0,0) xuất hiện trong đúng λ hàng .Vì có k(k-1)cặp các cột được sắp nên dẫn đến số lần xuất hiện của cặp được sắp (0,0) trong các hàng của R đúng bằng (λ-1)k(k-1).Bởi vậy ta có:

$$(\lambda\text{-}1)k(k\text{-}1) \geq \frac{(k(\lambda.n-1))^2}{\lambda.n^2-1} - k(\lambda.n-1)$$

và do đó :

$$((\lambda\text{-}1)k(k\text{-}1)+k(\lambda n\text{-}1)(\lambda n^2\text{-}1) \geq (k(\lambda n\text{-}1))^2$$

Khai triển ta có:

$$\lambda^2kn^2\text{-}\lambda k.n^2\text{-}\lambda^2n^2+\lambda^2n^3\text{-}\lambda k+k+\lambda\text{-}\lambda n \geq \lambda^2kn^2\text{-}2\lambda kn+k$$

hay:

$$\text{-}\lambda^2n^2+\lambda^2n^3 \geq \lambda kn^2+\lambda k\text{-}\lambda+\lambda n\text{-}2\lambda kn$$

hoặc $\lambda^2(n^3\text{-}n^2) \geq \lambda(k(n\text{-}1)^2+n\text{-}1)$

Cuối cùng,chia hai vế cho λ(n-1) ta có :

$$\lambda n^2 \geq k(n\text{-}1)+1$$

Đây chính là giới hạn cần tìm.

Kết quả sau thiết lập sự tồn tại của một lớp vô hạn các mảng trực giao đạt được giới hạn nêu trên với đấu ‘’=’’.

Định lí 10.10.

Giả sử p là một số nguyên tố và d≥2 là một số nguyên.Khi đó tồn tại một mảng trực giao 0A(p.(p^d-1)/(p-1).p^{d-2}

Chứng minh:

Kí hiệu $(Z_P)^d$ là không gian véc tơ chứa tất cả bộ d trên Z_P.Ta sẽ xây dựng A (là một $0A(p,(p^d-1)/(p-1),p^{d-2})$ trong đó các hàng và các cột được lập chỉ số theo các véc tơ trong $(Z_P)^d$.Các phần tử của A sẽ là các phần tử của Z_P.Tập hợp các hàng được xác định là $R=(Z_p)^d$):tập các cột là :

$$C = \{(c_1...c_d)\in(Z_p)^d: \exists j,0\le j\le d-1 ,c_1=...=c_j=0,c_{j+1}=1\}$$

R chứa tất cả các véc tơ trong $(Z_P)^d$,bởi vậy $|R|=p^d$.C chứa tất cả các véc tơ khác không có toạ độ khác 0 đầu tiên bằng 1.Nhận thấy rằng:

$$|C|=\frac{p^c-1}{p-1}$$

và không có hai véc tơ nào trong C là các bội vô hướng của nhau.

Bây giờ vuới mỗi véc tơ $r' \in R$ và mỗi $c'\in C$ ta định nghĩa:

$$A(r'.c')=r'.c'$$

Trong đó "."kí hiệu tích trong hai véc tơ (được rút gọn theo mod p).

Ta sẽ chứng minh A là mảng trực giao mong muốn.Cho $b',c'\in C$ là hai cột khác nhau và cho $x,y\in Z_P$.Ta sẽ tính số hàng r' để $A(r',b')=x$ và $A(r',b')=y$.Kí hiệu $r'=(r_1,r_2....r_d).b'=(b_1,b_2....b_d)$ và $c'=(c_1,c_2....c_d)$.Hai phương trình $r'.b'=x$ và $r'.c'=y$ có thể được viết thành hai phương trình tuyến tính trong Z_P

$$b_1.r_1+...+b_d.r_d=x$$
$$c_1.r_1+...+c_d.r_d=y.$$

Đây là hai phương trình tuyến tính với d ẩn $r_1...r_d$.Vì các bội b'và c' không phải là các bội vô hướng của nhau nên hai phương trình trên là độc lập tuyến tính.Bởi vậy hệ này có không gian nghiệm (d-2) chiều.Nghĩa là số các nghiệm (số các hàng trong đó x nằm ở cột b' và y ở cột c')bằng p^{d-2} theo mong muốn.

Ta sẽ làm một ví dụ nhỏ minh hoạ cách xây dựng này:

Ví dụ 10.3

Giả sử lấy p=2,d=3,khi đó ta sẽ xây dựng một 0A(2,7,2).Ta có :

R={000,001,010,011,100,101,110,111}

và C={001,010,011,100,101,110,111}

Ta nhận được kết quả là mảng trực giao như trên hình 10.6

Hình 10.6.Một 0A(2,7,2).
$$\begin{bmatrix} 0&0&0&0&0&0&0\\ 1&0&1&0&1&0&1\\ 0&1&1&0&0&1&1\\ 1&1&0&0&1&1&0\\ 0&0&0&1&1&1&1\\ 1&0&1&1&0&1&0\\ 0&1&1&1&1&0&0\\ 1&1&0&1&0&0&1 \end{bmatrix}$$

10.3.3Đặc trưng của mã xác thực .

Cho tới giờ ta đã nghiên cứu các mã xác thực nhận được từ các mảng trực giao.Ta cũng đã xem xét các điều kiện tồn tại cần thiết về việc xây dựng các mảng trực giao .Vấn đề ở đây là liệu có các phương pháp khác tốt hơn các mảng trực giao không?Tuy nhiên hai định lí đặc trưng sẽ cho biết rằng nếu chỉ giới hạn mối quan tâm tới các mã xác thực có xác suất lừa bịp nhỏ tới mức co thể thì vấn đề trên không cần phải đặt ra nữa.

Trước tiên ta sẽ chứng minh một định lí đảo một phần của định lí 10.5.

Định lí 10.11.

Giả sử (S,A,K,E)là một mã xác thực trong đó $|R|=n$ *và* $Pd_0=Pd_1=1/n$.*Khi đó* $|K|\geq n^2$.*Hơn nữa* $|K|=n^2$ *khi và chỉ khi có một mảng trực giao 0A(n.k.l) trong đó* $|S|=k$ *và* $p_K(K)=1/n^2$ *với mọi khoá* $K\in K$.

Chứng minh:

Cố định hai trạng thái nguồn tuỳ ý s và s' ,s=s' và xét phương trình (10.6).Với mỗi cặp được sắp (a,a') của các nhãn xác thực ta xác định :

$$K_{a,a'}=\{K\in K : e_K(s)=a, e_K(s')=a'\}.$$

Khi đó $|K|>0$ với mọi cặp (a,a').Cũng thấy rằng các tập $K_{a,a'}$ này rời nhau (có n^2 tập).Bởi vậy $K\geq n^2$.

Bây giờ giả sử rằng $|K|=n^2$ .Khi đó trị $|K_{a,a'}|=1$,với mọi cặp (a,a') và từ phương trình (10.6) ,cho ta thấy rằng $p_K(K)=1/n^2$ với mọi khoá $K\in K$.

Vấn đề còn lại là phải chứng tỏ ma trận xác thực sẽ tạo nên ma trận trực giao 0A(n,k,l) .Xét các cột lấy chỉ số theo các trạng thái nguồn s và s'.Vì $|K_{a,a'}|=1$ với mọi (a,a') nên mỗi cặp được sắp xuất hiện dúng một lần trong hai cột này.Vì s,s' là tuỳ ý nên mỗi cặp được sắp xuất hiện đúng một lần trong hai cột bất kì.

Đặc trưng sau đây có khó hơn một chút chúng ta chỉ phát biểu mà không chứng minh .

Định lí 10.2

Giả sử (S,A,K,E) là một mã xác thực ,trong đó $|A|=n$ *và* $Pd_0=Pd_1=1/n$.*Khi đó* $|K|\geq k(n-1)+1$.*Hơn nữa* $|K|=k(n-1)+1$ *khi và chỉ khi có một mảng trực giao* $0A(n,k,\lambda)$,*ở đây* $|S|=k,\lambda=(k(n-1)+1)/n^2$ và $p_K(K)=1/(k(n-1)+1)$ *với mọi khoá* $K\in K$.

Nhận xét.Chú ý rằng định lí 10.10 tạo ra một lớp vô hạn các mảng trực giao đạt được giới hạn ở định lí 10.12 với dấu "=".

10.4.CÁC GIỚI HẠN ENTROPY

Trong phần này chúng ta dùng kĩ thuật entropy để nhận được các giới hạn về các xác suất lừa bịp .Trước tiên ta sẽ xét các giới hạn đối với Pd_0.

Định lí 10.13

Giả sử (S,R.K,E) là một mã xác thực .Khi đó

$$LogPd_0\geq H(K|M)-H(K)$$

Chứng minh:

Từ phương trình (10.1) ta có :

$$Pd_0\geq \max\{payoff(s,a):s\in S,a\in R\}$$

Vì giá trị cực của payoff(s,a) phải lớn hơn trung bình các trọng số của chúng nên ta nhận được:

$$Pd_0\geq\sum_{s\in S,a\in R}p_M(s,a)payoff(s,a)$$

Như vậy thoe bất đẳng thức Jensen(định lí (2.5) ta có :

$$LogPd_0\geq\log\sum_{s\in S,a\in R}p_M(s,a)payoff(s,a)$$
$$\geq\sum_{s\in S,a\in R}p_M(s,a)\log payoff(s,a)$$

Theo phần 10.2:

$$P_M(s,a)=p_s(s)x\ payoff(s,a)$$

Ta thấy rằng:

$$Log\ Pd_0\geq\sum_{s\in S,a\in R}p_s(s)payoff(s,a)\ \log payoff(s,a)$$

Bây giờ ta thấy rằng $payoff(s,a)=p_R(a|s)$(tức là xác suất để a là nhãn xác thực với điều kiện s là trạng thái nguồn).Bởi vậy:

$$LogPd_0\geq\sum_{s\in S,a\in R}p_s(s).p_R(a|s)\ \log p_R(a|s)\ =-H(A|S)$$

Theo định nghĩa của entropy có điều kiện .Ta sẽ hoàn chỉnh chứng minh định lí bằng cách chỉ ra rằng: $-H(A|S)=H(K|M)-H(K)$.Điều kiện này được rút ra từ các đồng nhất thức cơ bản của entropy.Một mặt ta có :

$$H(K,A,S)=H(A|K,S)+H(A|S)+H(S)$$

Mặt khác ta tính:

$$H(K,A,S)=H(A|K,S)+H(K,S)=H(S)+H(K)$$

ậ đây ta có sử dụng điều kiện $H(A|K,S)=0$ vì khoá và trạng thái nguồn sẽ xác định nhãn xác thực một cách duy nhất .Ta cũng dùng đẳng thức $H(A|S)=H(K)+H(S)$ vì nguồn và khoá là các biến cố độc lập.

So sánh hai biểu thức biểu thị H(K,S,A) ta có:

$$-H(A,S)=H(K|A,S)-H(K)$$

Tuy nhiên thông báo m=(s,a) được xác định gồm một trạng thái nguồn và một trạng thái nhãn xác thực(nghĩa là $M=SxA$).Bởi vậy:

$$H(K|A,S)=H(K|M)$$

Định lí được chứng minh.

Sau đây ta sẽ chỉ đưa ra mà không chứng minh giới hạn tương tự cho Pd_1.

Định lí 10.4

Giả sử rằng (S,A,K,E) là một mã xác thực .Khi đó

$$Log Pd_1 \geq H(K/M^2)-H(K/M)$$

Cần phải xác định giới hạn entropy theo biến ngẫu nhiên M^2.Giả sử ta xác thực hai trạng thái nguồn khác nhau dùng cùng một khoá K.Theo cách này ta nhận được một cặp được sắp các banr tin $(m_1,m_2)\in MxM$.Để xác định phân bố xác suất trên MxM,cần phải xác định xác suất trên SxS với điều kiện $p_{sxs}(s,s)=0$ với mọi $s\in S$(nghĩa là không cho phép lặp lại trạng thái nguồn).Các phân bố xác suất trên K và SxS sẽ dẫn đến phân bố xác suất trên MxM tương tự như phân bố xác suất trên K và S sẽ tạo nên một phân bố xác suất trên M Để minh hoạ cho hai giới hạn trên ,xét cấu trúc mảng trực giao cơ bản và chỉ ra rằng cả hai giới hạn trong định lí 10.13 và 10.14 đều đạt được với dấu bằng.Trước hết ta dễ thấy rằng:

$$H(K)=\log \lambda n^2$$

Vì mỗi một trong λn^2 quy tắc xác thực đều được chọn đồng xác suất.Tiếp theo ta sẽ quay lại việc tính toán $H(K|M)$.Nừu đã quan sát được một bản tin m=(s,a) nào đó thì điều này sẽ giới hạn các khóa sẽ nằm trong tập con có lực lượng λn.Mỗi khoá trong λn khóa này sẽ có

tập con như nhau .Vì thế $H(K|m)=\log\lambda n$ với bản tin n bất kì .Khi đó ta có :

$$H(K|M)=\sum_{m\in M}p_M(m)H(K|m)$$
$$=\sum_{\in M}p_M(m)\log\lambda n$$
$$=\log\lambda n$$

Như vậy ta có:

$$H(K|M)-H(K)=\log\lambda n-\log\lambda n^2=-\log n=\log Pd_0$$

Như vậy giới hạn thoả mãn với dấu "=".

Nừu ta quan sát được hai bản tin (được tạo ra theo cùng một khoá và các trạng thái nguồn khác nhau)thì số các khoá có thể giảm xuống còn λ.Lập luận tương tự như trên ta thấy rằng $H(K|M^2)=\log\lambda$.Khi đó:

$$H(K|M)-H(K)=\log\lambda-\log\lambda n$$
$$=-\log n=-Pd_1$$

Như vậy giới hạn này được thoả mãn với dấu "=".

10.5.CÁC CHÚ GIẢI VÀ TÀI LIỆU DẪN

Các mã xác thực được phát minh vào năm 1974 bởi Gilbert.Mac-Williams và Sloane [GMS 74ư.Nhiếu phần lí thuyết về các mã xác thực đã được Simones phát triển,ông đã chứng minh nhiều kết quả cơ bản trong lĩnh vực này.Hai bài tổng quan hữa ích của Simones là [Si92] và [Si88].Massey cũng trình bày một tổng quan khá hay khác trong [Ma86].Các mối liên hệ giữa các mảng trực giao và các mã xác thực đã là mối quan tâm của nhiều nhà nghiên cứu..Cách trình bày ở đây dựa vào ba bài báo của Stinson[St 88],[St 90]và [St 92].Các mảng trực giao đã được nghiên cứu trong hơn 45 năm bởi các nhà nghiên cứu trong lĩnh vực thống kê và trong lí thuyết thiết kế tổ hợp.Ví dụ,giới hạn trong định lí 10.9 lần đầu tiên được chứng minh bởi Placket và Berman vào 1945 trong [PB 45].Nhiều kết quả thú vị về các mảng trực giao có thể tìm được trong nhiều giáo trình khác nhau về lí thuyết thiết kế tổ hợp(chẳng hạn như trong [BJL 8] của Beth,Jungickel và Lenz).

Cuối cùng việc sử dụng kĩ thuật entropy trong việc nghiên cứu các mã xác thực do Simone đưa ra .Giới hạn của định lí 10.13 đã được Simone chứng minh trước tiên trong [Si 85];một cánh chứng minh của định lí 10.14 có thể tìm được trong [Wa 90] của Walker.

BÀI TẬP

10.1.Hãy tính Pd_0 và Pd_1 của mã xác thực được biểu thị trong ma trận sau :

Khoá	1	2	3	4
1	1	1	2	3
2	1	2	3	1
3	2	1	3	1
4	2	3	1	2
5	3	2	1	3
6	3	3	2	1

Các phân bố xác suất trên S và K như sau:

$$P_s(1)=p_s(4)=1/6\ ,p_s(2)=p_s(3)=1/3$$

$$p_K(1)=p_K(6)=1/4,\ p_K(2)=p_K(3)=p_K(4)=p_K(5)=1/8.$$

Nêu các chiến lược thay thế và giả mạo tối ưu .

10.2.Ta đã biết cấu trúc đối với một mảng trực giao 0A(p,p,1)khi p là số nguyên tố.Hãy chứng tỏ rằng luôn có thể mở rộng 0A(p,p,1)thêm một cột nữa để tạo thành 0A(p,p+1,1).Hãy minh hạo cấu trúc của bạn trong trường hợp p=5.

10.3.Giả sử A là một cấu trúc $0A(n_1,k,\lambda_1)$ trên tập kí hiệu $\{1,...,n_1\}$ và giả sử B là một $0A(n_2,k,\lambda_2)$ trên tập kí hiệu $\{1,...,n_2\}$Ta xây dựng C là một $0A(n_1,n_2,k,\lambda_1\lambda_2)$ trên tập kí hiệu $\{1...n_1\}x\{1...n_2\}$ như sau :với mỗi hàng $r_1=(x_1...x_k)$ của A và với mỗi hàng $s_1=\{y_1...y_k\}$ của B ta xác định một hàng t_1 của C là:

$$t_1=((x_1,y_1),...,(x_k,y_k)).$$

Hãy chứng manh rằng C thực sự là một $0A(n_1n_2,k,\lambda_1\lambda_2)$.

10.4.Hãy xây dựng một mảng trực giao 0A(3,13,3).

10.5Hãy viết một chương trình máy tính để tính H(K),H(K|M) và $H(K|M^2)$cho mã xác thực ở bài toán 10.1Phân bố xác suất trên cavcs dãy của hai nguồn là :

$$p_{S^2}(1.2) = p_{S^2}(1.3) = p_{S^2}(1.4) = 1/18$$

$$p_{S^2}(2.1) = p_{S^2}(2.3) = p_{S^2}(2.4) = 1/9$$

$$p_{S^2}(3.1) = p_{S^2}(3.2) = p_{S^2}(3.4) = 1/9$$

$$p_{S^2}(4.1) = p_{S^2}(4.2) = p_{S^2}(4.3) = 1/18$$

Hãy so sánh giới hạn entropy của Pd_0 và Pd_1 với các giá trị mà bạn tính được trong bài tập 10.1.

Chỉ dẫn:Để tính $p_K(k|m)$ hãy dùng công thức Bayes:

$$p_K(k \mid m) = \frac{p_M(m \mid k)\, p_K(k)}{p_M(m)}$$

Ta đã biết cách tính $p_M(m)$.Để tính $p_M(m \mid k)$ hãy viết m=(s,a) và nhận xét thấy rằng :$p_M(m \mid k)=p_S(s)$ nếu $e_K(s)=a$ và $p_M(m \mid k)=0$ trong trường hợp ngược lại .