

## CHƯƠNG 3. MẬT MÃ KHOÁ CÔNG KHAI

Trước khi tìm hiểu hệ mật khoá công khai chúng ta ôn tập lại một số kiến thức về lý thuyết số.

### 3.1. SỐ HỌC MODULO

#### 3.1.1. Số nguyên

Tập các số nguyên:

$$\{\dots, -3, -2, -1, 0, 1, 2, 3, \dots\} = \mathbf{Z} \quad (3.1)$$

#### **Định nghĩa 3.1:**

Cho  $a, b \in \mathbf{Z}$ ,  $a$  là ước của  $b$  nếu  $\exists c \in \mathbf{Z} : b = ac$ . Ký hiệu  $a \mid b$

#### **Các tính chất chia hết**

$\forall a, b, c \in \mathbf{Z}$  ta có:

- (i)  $a \mid a$ .
- (ii) Nếu  $a \mid b$  và  $b \mid c$  thì  $a \mid c$
- (iii) Nếu  $a \mid b$  và  $a \mid c$  thì  $a \mid (bx + cy)$  với  $\forall x, y \in \mathbf{Z}$
- (iv) Nếu  $a \mid b$  và  $b \mid a$  thì  $a = \pm b$

#### **Định nghĩa 3.2:**

Thuật toán chia đối với các số nguyên:

Nếu  $a$  và  $b$  là các số nguyên với  $b \geq 1$  thì  $a = qb + r$ ,  $0 \leq r < b$

$q$  và  $r$  là duy nhất.

Phần dư của phép chia  $a$  và  $b$  được ký hiệu  $a \bmod b = r$

Thương của phép chia  $a$  và  $b$  được ký hiệu  $a \div b = q$

$$\text{Ta có } a \div b = \left\lfloor \frac{a}{b} \right\rfloor, \quad a \bmod b = a - b \left\lfloor \frac{a}{b} \right\rfloor$$

**Ví dụ 3.1:**  $a = 73, b = 17 \Rightarrow 73 \div 17 = 4, \quad 73 \bmod 17 = 5$

#### **Định nghĩa 3.3:** Ước chung.

$c$  là ước chung của  $a$  và  $b$  nếu  $c \mid a$  &  $c \mid b$

#### **Định nghĩa 3.4:** Ước chung lớn nhất (ƯCLN)

Số nguyên dương  $d$  là ƯCLN của các số nguyên  $a$  và  $b$  (Ký hiệu  $d = (a, b)$ ) nếu:

(i)  $d$  là ước chung của  $a$  và  $b$ .

(ii) Nếu có  $c | a$  và  $c | b$  thì  $c | d$ .

Như vậy  $(a, b)$  là số nguyên dương lớn nhất ước của cả  $a$  và  $b$  không kể  $(0, 0) = 0$ .

**Ví dụ 3.2:**

Các ước chung của 12 và 18 là  $\{\pm 1, \pm 2, \pm 3, \pm 6\} \rightarrow (12, 18) = 6$

**Định nghĩa 3.5:** Bội chung nhỏ nhất (BCNN)

Số nguyên dương  $d$  là bội chung nhỏ nhất (BCNN) của các số nguyên  $a$  và  $b$  (Ký hiệu  $d = BCNN(a, b)$ ) nếu:

(i)  $a | d, b | d$ .

(ii) Nếu có  $a | c, b | c$  thì  $d | c$ .

Như vậy  $d$  là số nguyên dương nhỏ nhất là bội của cả  $a$  và  $b$ .

**Tính chất**

$$BCNN(a, b) = \frac{a \cdot b}{(a, b)} \quad (3.2)$$

$$\text{Ví dụ 3.3: } (12, 18) = 6 \Rightarrow BCNN(12, 18) = \frac{12 \cdot 18}{6} = 36$$

**Định nghĩa 3.6:**

Hai số nguyên dương  $a$  và  $b$  được gọi là nguyên tố cùng nhau nếu:  $(a, b) = 1$

**Định nghĩa 3.7:**

Số nguyên  $p \geq 2$  được gọi là số nguyên tố nếu các ước dương của nó chỉ là 1 và  $p$ . Ngược lại  $p$  được gọi là hợp số.

**Định lý 3.1: (Định lý cơ bản của số học)**

Với mỗi số nguyên  $n \geq 2$  ta luôn phân tích được dưới dạng tích của lũy thừa của các số nguyên tố.

$$n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k} \quad (3.3)$$

Trong đó  $p_i$  là các số nguyên tố khác nhau và  $e_i$  là các số nguyên dương. Hơn nữa phân tích trên là duy nhất.

**Định nghĩa 3.8:**

Với  $n \geq 2$ , hàm  $\varphi(n)$  được xác định là số các số nguyên trong khoảng  $[1, n]$  nguyên tố cùng nhau với  $n$ .

**Các tính chất của hàm  $\varphi(n)$**

Nếu  $p$  là số nguyên tố thì  $\varphi(p) = p - 1$ .

Nếu  $(m, n) = 1$  thì  $\varphi(m \cdot n) = \varphi(m) \cdot \varphi(n)$ .

Nếu  $n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$  là phân tích ra thừa số nguyên tố của  $n$  thì:

$$\varphi(n) = n \left(1 - \frac{1}{p_1}\right) \left(1 - \frac{1}{p_2}\right) \dots \left(1 - \frac{1}{p_k}\right) \quad (3.4)$$

**Định lý 3.2:** Với  $\forall n \geq 5$ :

$$\varphi(n) > \frac{n}{6 \ln \ln n} \quad (3.5)$$

### 3.1.2. Các thuật toán trong $Z$

Cho  $a$  và  $b$  là các số nguyên không âm và nhỏ hơn hoặc bằng  $a \times b = b \times a$ . Cần chú ý rằng số các bit trong biểu diễn nhị phân của  $n$  là  $\lceil \lg n \rceil + 1$  và số này xấp xỉ bằng  $\lg n$ . Số các phép toán bit đối với bốn phép toán cơ bản trên các số là cộng, trừ, nhân và chia sử dụng các thuật toán kinh điển được tóm lược trên bảng sau. Các kỹ thuật tinh tế hơn đối với các phép toán nhân và chia sẽ có độ phức tạp nhỏ hơn.

Bảng 3.1. Độ phức tạp bit của các phép toán cơ bản trong  $Z$

Phép toán	Độ phức tạp bit
Cộng $a + b$	$O(\lg a + \lg b) = O(\lg n)$
Trừ $a - b$	$O(\lg a + \lg b) = O(\lg n)$
Nhân $a \cdot b$	$O((\lg a) \cdot (\lg b)) = O((\lg n)^2)$
Chia $a = qb + r$	$O((\lg a) \cdot (\lg b)) = O((\lg n)^2)$

ƯCLN của 2 số nguyên  $a$  và  $b$  có thể được tính theo định lý sau:

**Định lý 3.3:**

Nếu  $a = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$ ,  $b = p_1^{f_1} p_2^{f_2} \dots p_k^{f_k}$  trong đó  $e_i \geq 0$ ,  $f_i \geq 0$

thì  $UCLN(a, b) = p_1^{\min(e_1, f_1)} p_2^{\min(e_2, f_2)} \dots p_k^{\min(e_k, f_k)}$

và  $BCNN(a, b) = p_1^{\max(e_1, f_1)} p_2^{\max(e_2, f_2)} \dots p_k^{\max(e_k, f_k)}$

**Ví dụ 3.4:** Cho  $a = 4864 = 2^8 \cdot 19$ ,  $b = 3458 = 2 \cdot 7 \cdot 13 \cdot 19$ . Khi đó

$$UCLN(a, b) = (4864, 3458) = 2 \cdot 19 = 38$$

$$BCNN(a, b) = (4864, 3458) = 2^8 \cdot 7 \cdot 13 \cdot 19 = 442624$$

**Định lý 3.4:** Nếu  $a$  và  $b$  là các số nguyên dương với  $a > b$  thì:

$$UCLN(a, b) = UCLN(b, a \bmod b) \quad (3.6)$$

Thuật toán Euclide sau sẽ cho ta cách tính UCLN rất hiệu quả mà không cần phải phân tích ra thừa số nguyên tố.

### Thuật toán Euclide

Tính UCLN của 2 số nguyên

VÀO : Hai số nguyên không âm  $a$  và  $b$  với  $a > b$

RA : UCLN của  $a$  và  $b$ .

(1) While  $b \neq 0$  do

$$r \leftarrow a \bmod b, a \leftarrow b, b \leftarrow r$$

(2) Return  $(a)$ .

**Định lý 3.5:** Thuật toán trên có thời gian chạy chừng  $O((\lg n)^2)$  các phép toán bit.

**Ví dụ 3.5:** Sau đây là các bước chia của thuật toán trên khi tính:

$$(4864, 3458) = 38$$

$$4864 = 1 \cdot 3458 + 1406$$

$$3458 = 2 \cdot 1406 + 646$$

$$1406 = 2 \cdot 646 + 76$$

$$646 = 5 \cdot 76 + 38$$

$$76 = 2 \cdot 38 + 0$$

Thuật toán trên có thể được mở rộng để không những chỉ tính được UCLN của 2 số nguyên  $a$  và  $b$  mà còn tính được các số nguyên  $x$  và  $y$  thoả mãn  $ax + by = d$ .

### Thuật toán Euclide mở rộng

VÀO : Hai số nguyên không âm  $a$  và  $b$  với  $a \geq b$

RA :  $d = UCLN(a, b)$  và các số nguyên  $x$  và  $y$  thoả mãn  $ax + by = d$ .

Nếu  $b = 0$  thì đặt  $d \leftarrow a$ ,  $x \leftarrow 1$ ,  $y \leftarrow 0$  và return  $(d, x, y)$

(1) Đặt  $x_2 \leftarrow 1$ ,  $x_1 \leftarrow 0$ ,  $y_2 \leftarrow 0$ ,  $y_1 \leftarrow 1$

(2) While  $b > 0$  do

$$2.1 \ q \leftarrow \lfloor a/b \rfloor, r \leftarrow a - qb, x \leftarrow x_2 - qx_1, y \leftarrow y_2 - qy_1$$

$$2.2 \ a \leftarrow b, b \leftarrow r, x_2 \leftarrow x_1, x_1 \leftarrow x, y_2 \leftarrow y_1, y_1 \leftarrow y$$

(3) Đặt  $d \leftarrow a$ ,  $x \leftarrow x_2$ ,  $y \leftarrow y_2$  và return  $(d, x, y)$

**Định lý 3.6:**

Thuật toán trên có thời gian chạy cỡ  $O((\lg n)^2)$  các phép toán bit.

**Ví dụ 3.6:** Bảng 3.2 sau chỉ ra các bước của thuật toán trên với các giá trị vào  $a = 4864$  và  $b = 3458$

Bảng 3.2. Thuật toán Euclide mở rộng với các đầu vào  $a = 4864$  và  $b = 3458$

Q	r	x	y	a	b	$x_1$	$x_2$	$y_2$	$y_1$
–	–	–	–	4864	3458	1	0	0	1
1	1406	1	–1	3458	1406	0	1	1	–1
2	646	–2	3	1406	646	1	–2	–1	3
2	114	5	–7	646	114	–2	5	3	7
5	76	–27	38	114	76	5	–27	–7	38
1	38	32	–45	76	38	–27	32	38	–45
2	0	–91	128	38	0	32	–91	–45	128

Bởi vậy ta có:

$$UCLN(4864, 3458) = 38 \text{ và } (4864)(32) + (3458)(-45) = 38$$

### 3.1.3. Các số nguyên modulo n

**Định nghĩa 3.9:**

Nếu  $a$  và  $b$  là các số nguyên thì  $a$  được gọi là đồng dư với  $b$  theo modulo (ký hiệu là  $a \equiv b \pmod{n}$ ) nếu  $n \mid (a - b)$ .

Số nguyên  $n$  được gọi là modulo đồng dư.

**Ví dụ 3.7:**  $24 \equiv 9 \pmod{5}$  vì  $24 - 9 = 3 \cdot 5$

$$-11 \equiv 17 \pmod{7} \text{ vì } -11 - 17 = -4 \cdot 7$$

**Các tính chất**

Đối với  $a, a_1, b, b_1, c \in \mathbb{Z}$  ta có:

(1)  $a \equiv b \pmod{n}$  nếu và chỉ nếu  $a$  và  $b$  cũng có phần dư khi chia cho  $n$ .

(2) Tính phản xạ :  $a \equiv a \pmod{n}$ .

(3) Tính đối xứng : Nếu  $a \equiv b \pmod{n}$  thì  $b \equiv a \pmod{n}$

Tính bắc cầu : Nếu  $a \equiv b \pmod{n}$  và  $b \equiv c \pmod{n}$  thì  $a \equiv c \pmod{n}$

(4) Nếu  $a \equiv a_1 \pmod{n}$  và  $b \equiv b_1 \pmod{n}$  thì

$$a + b \equiv a_1 + b_1 \pmod{n} \text{ và } ab \equiv a_1 b_1 \pmod{n}$$

Lớp tương đương của một số nguyên  $a$  là tập các số nguyên đồng dư với  $a$  modulo  $n$ . Từ các tính chất (2), (3) và (5) ở trên ta có thể thấy rằng đối với  $n$  cố định, quan hệ đồng dư theo modulo  $n$  sẽ phân hoạch  $\mathbf{Z}$  thành các lớp tương đương.

Nếu  $a = qn + r$  với  $0 \leq r \leq n$  thì  $a \equiv r \pmod{n}$ .

Bởi vậy mỗi số nguyên  $a$  là đồng dư theo modulo  $n$  với một số nguyên duy nhất nằm trong khoảng từ 0 tới  $n - 1$ , số này được gọi là thặng dư tối thiểu của  $a \bmod n$ . Như vậy  $a$  và  $r$  có thể được dùng để biểu thị cho lớp tương đương này.

**Định nghĩa 3.10:**

Các số nguyên modulo  $n$  (ký hiệu  $\mathbf{Z}_n$ ) là tập (các lớp tương đương) của các số nguyên  $\{0, 1, 2, \dots, n - 1\}$ . Các phép cộng, trừ, nhân trong  $\mathbf{Z}_n$  được thực hiện theo modulo  $n$ .

**Ví dụ 3.8:**  $\mathbf{Z}_{25} = \{0, 1, \dots, 24\}$ . Trong  $\mathbf{Z}_{25}$  ta có:

$$13 + 16 = 4 \text{ vì } 13 + 16 = 29 \equiv 4 \pmod{25}$$

Tương tự  $13 \cdot 16 = 8$  trong  $\mathbf{Z}_{25}$ .

**Định nghĩa 3.11:** (Phần tử nghịch đảo)

Cho  $a \in \mathbf{Z}_n$ , phần tử nghịch đảo (ngược theo phép nhân) của  $a \bmod n$  là một số nguyên  $x \in \mathbf{Z}_n$  sao cho:  $ax \equiv 1 \pmod{n}$

Nếu  $x$  tồn tại thì nó là duy nhất,  $a$  được gọi là khả nghịch. Phần tử nghịch đảo của  $a$  được ký hiệu là  $a^{-1}$ .

**Định nghĩa 3.12:**

Phép chia của  $a$  cho  $b \bmod n$  là tích của  $a$  và  $b^{-1} \bmod n$  tích này được xác định nếu  $b$  là phần tử khả nghịch.

**Định lý 3.7:**

Cho  $a \in \mathbf{Z}_n$ , khi đó  $a$  là khả nghịch nếu và chỉ nếu:  $(a, n) = 1$

**Ví dụ 3.9:** Các phần tử khả nghịch trong  $\mathbf{Z}_9$  là 1, 2, 4, 5, 7 và 8. Chẳng hạn  $4^{-1} = 7$  vì  $4 \cdot 7 \equiv 1 \pmod{9}$ .

**Định lý 3.8:**

Cho  $d = (a, n)$ , phương trình đồng dư  $ax \equiv b \pmod{n}$  có nghiệm  $x$  nếu và chỉ nếu  $d \mid b$ , trong trường hợp này có đúng  $d$  nghiệm nằm giữa 0 và  $n - 1$ , những nghiệm này là tất cả các đồng dư theo modulo  $n \mid d$ .

**Định lý 3.9: (Phần dư China)**

Nếu các số nguyên  $n_1, n_2, \dots, n_k$  là nguyên tố cùng nhau từng đôi một thì hệ các phương trình đồng dư:

$$\begin{aligned} x &\equiv a_1 \pmod{n_1} \\ x &\equiv a_2 \pmod{n_2} \\ &\dots\dots\dots \\ x &\equiv a_k \pmod{n_k} \end{aligned}$$

sẽ có nghiệm duy nhất theo modulo  $n$  ( $n = n_1 \cdot n_2 \dots n_k$ )

**Thuật toán Gauss**

Nghiệm  $x$  của hệ phương trình đồng dư trong định lý phần dư China có thể được tính bằng:

$$x = \sum_{i=1}^k a_i N_i M_i \pmod{n} \quad (3.7)$$

Trong đó  $N_i = n / n_i$  và  $M_i = N_i^{-1} \pmod{n_i}$

Các tính toán này có thể được thực hiện trong  $O((\lg n)^2)$  các phép toán trên bit.

**Ví dụ 3.10:** Cặp phương trình đồng dư  $x \equiv 3 \pmod{7}, x \equiv 7 \pmod{13}$  có nghiệm duy nhất  $x \equiv 59 \pmod{91}$

**Định lý 3.10:**

Nếu  $(n_1, n_2) = 1$  thì cặp phương trình đồng dư.

$$x \equiv a \pmod{n_1}, x \equiv a \pmod{n_2}$$

có một nghiệm duy nhất  $x \equiv a \pmod{n_1, n_2}$

**Định nghĩa 3.13:**

Nhóm nhân của  $\mathbf{Z}_n$  là  $\mathbf{Z}_n^* = \{a \in \mathbf{Z}_n \mid (a, n) = 1\}$

Đặc biệt, nếu  $n$  là số nguyên tố thì  $\mathbf{Z}_n^* = \{a \mid 1 \leq a \leq n-1\}$

**Định nghĩa 3.14:**

Cấp của  $\mathbf{Z}_n^*$  là số các phần tử trong  $\mathbf{Z}_n^*$  (ký hiệu  $|\mathbf{Z}_n^*|$ )

Theo định nghĩa của hàm Phi-Euler ta thấy:

$$|\mathbf{Z}_n^*| = \varphi(n) \quad (3.8)$$

Chú ý: nếu  $a \in \mathbf{Z}_n^*$  và  $b \in \mathbf{Z}_n^*$  thì  $a, b \in \mathbf{Z}_n^*$  và bởi vậy  $\mathbf{Z}_n^*$  là đóng đối với phép nhân.

**Định lý 3.11:**

Cho  $p$  là một số nguyên tố:

- (1) *Định lý Euler:* Nếu  $a \in \mathbf{Z}_n^*$  thì  $a^{\varphi(n)} \equiv 1 \pmod{n}$ .
- (2) Nếu  $n$  là tích của các số nguyên khác nhau và nếu  $r \equiv s \pmod{\varphi(n)}$  thì  $a^r \equiv a^s \pmod{n}$  đối với mọi số nguyên  $a$ . Nói một cách khác khi làm việc với modulo  $n$  thì các số mũ có thể được rút gọn theo modulo  $\varphi(n)$ .

**Định lý 3.12:**

Cho  $p$  là một số nguyên tố:

- (1) *Định lý Ferma:* Nếu  $(a, p) = 1$  thì  $a^{p-1} \equiv 1 \pmod{p}$ .
- (2) Nếu  $r \equiv s \pmod{p-1}$  thì  $a^r \equiv a^s \pmod{p}$  đối với mọi số nguyên  $a$ . Nói một cách khác khi làm việc với modulo của một số nguyên tố  $p$  thì các lũy thừa có thể được rút gọn theo modulo  $p-1$ .
- (3) Đặc biệt  $a^p \equiv a \pmod{p}$  với mọi số nguyên  $a$ .

**Định nghĩa 3.15:**

Cho  $a \in \mathbf{Z}_n^*$ . Cấp của  $a$  (ký hiệu là  $\text{ord}(a)$ ) là số nguyên dương nhỏ nhất  $t$  sao cho  $a^t \equiv 1 \pmod{n}$ .

**Định nghĩa 3.16:**

Cho  $a \in \mathbf{Z}_n^*$ ,  $\text{ord}(a) = t$  và  $a^s \equiv 1 \pmod{n}$  khi đó  $t$  là ước của  $s$ .

Đặc biệt  $t \mid \varphi(n)$ .

**Ví dụ 3.11:** Cho  $n = 21$ , khi đó  $\mathbf{Z}_{21}^* = \{1, 2, 4, 5, 8, 10, 11, 13, 16, 17, 19, 20\}$

Chú ý rằng  $\varphi(21) = \varphi(7) \cdot \varphi(3) = 12 = |\mathbf{Z}_{21}^*|$ . Cấp của các phần tử trong  $\mathbf{Z}_{21}^*$  được nêu trong bảng sau:

Bảng 3.3. Cấp của các phần tử trong  $\mathbf{Z}_{21}^*$

$a \in \mathbf{Z}_{21}^*$	1	2	4	5	8	10	11	13	16	17	19	20
$\text{ord}(a)$	1	6	3	6	2	6	6	2	3	6	6	2



**Định nghĩa 3.17:**

Cho  $\alpha \in \mathbf{Z}_n^*$ . Nếu cấp của  $\alpha$  là  $\varphi(n)$  thì  $\alpha$  được gọi là phần tử sinh hay phần tử nguyên thủy của  $\mathbf{Z}_n^*$ . Nếu  $\mathbf{Z}_n^*$  có một phần tử sinh thì  $\mathbf{Z}_n^*$  được gọi là cyclic.

**Các tính chất của các phần tử sinh của  $\mathbf{Z}_n^*$**

(1)  $\mathbf{Z}_n^*$  có phần tử sinh nếu và chỉ nếu  $n = 2, 4, p^k$  hoặc là  $2p^k$ , trong đó  $p$  là một số nguyên tố lẻ và  $k \geq 1$ . Đặc biệt, nếu  $p$  là một số nguyên tố thì  $\mathbf{Z}_n^*$  có phần tử sinh.

(2) Nếu  $\alpha$  là một phần tử sinh của  $\mathbf{Z}_n^*$  thì:

$$\mathbf{Z}_n^* = \{\alpha^i \bmod n \mid 0 \leq i \leq \Phi(n) - 1\} \quad (3.9)$$

(3) Giả sử rằng  $\alpha$  là một phần tử sinh của  $\mathbf{Z}_n^*$  khi đó  $b = \alpha^i \bmod n$  cũng là một phần tử sinh của  $\mathbf{Z}_n^*$  nếu và chỉ nếu  $(i, \varphi(n)) = 1$ . Từ đó ta rút ra rằng nếu  $\mathbf{Z}_n^*$  là cyclic thì số các phần tử sinh là  $\varphi(\varphi(n))$ .

(4)  $\alpha \in \mathbf{Z}_n^*$  là một phần tử sinh của  $\mathbf{Z}_n^*$  nếu và chỉ nếu  $\alpha^{\varphi(n)/p} \not\equiv 1 \pmod{n}$  đối với mỗi nguyên tố  $p$  của  $\varphi(n)$

Ví dụ 1.12:  $\mathbf{Z}_{21}^*$  không là cyclic vì nó không chứa một phần tử có cấp  $\Phi(21) = 12$  (Chú ý rằng 21 không thoả mãn điều kiện (1) ở trên).

$\mathbf{Z}_{25}^*$  là cyclic và có một phần tử sinh  $\alpha = 2$

**Định nghĩa 3.18:**

Cho  $a \in \mathbf{Z}_n^*$ ,  $a$  được gọi là thặng dư bậc hai modulo  $n$  (hay bình phương của modulo  $n$ ) nếu tồn tại  $x \in \mathbf{Z}_n^*$  sao cho  $x^2 \equiv a \pmod{n}$ . Nếu không tồn tại  $x$  như vậy thì  $a$  được gọi là thặng dư không bậc hai modulo  $n$ . Tập tất cả các thặng dư bậc hai modulo  $n$  được ký hiệu là  $Q_n$ , còn tập tất cả các thặng dư không bậc hai được ký hiệu là  $\overline{Q_n}$ . Cần chú ý rằng theo định nghĩa  $0 \notin \mathbf{Z}_n^*$ . Bởi vậy  $0 \notin Q_n$  và  $0 \notin \overline{Q_n}$ .

**Định lý 3.13:**

Cho  $p$  là một số nguyên tố và  $\alpha$  là một phần tử sinh của  $\mathbf{Z}_p^*$ . Khi đó  $a \in \mathbf{Z}_p^*$  là một thặng dư bậc hai modulo  $p$  nếu và chỉ nếu  $a = \alpha^i \bmod p$ , trong đó  $i$  là một số nguyên chẵn. Từ đó rút ra:

$$|Q_p| = \frac{(p-1)}{2} \text{ và } |\overline{Q_p}| = \frac{(p-1)}{2} \quad (3.10)$$

tức là một nửa số phần tử trong  $\mathbf{Z}_p^*$  là các thặng dư bậc hai và nửa còn lại thặng dư không bậc hai.

**Ví dụ 3.12:**  $\alpha = 6$  là một phần tử sinh của  $\mathbf{Z}_{13}^*$ . Các lũy thừa của  $\alpha$  được liệt kê ở bảng 3.4. sau đây:

Bảng 3.4.

$i$	1	2	3	4	5	6	7	8	9	10	11	12
$\alpha^i \bmod 13$	6	10	8	9	2	12	7	3	5	4	11	1

$$\text{Bởi vậy } Q_{13} = \{1, 3, 4, 9, 10, 12\}, \quad \bar{Q}_{13} = \{2, 5, 6, 7, 8, 11\}$$

**Định lý 3.14:**

Cho  $n$  là tích của hai số nguyên tố lẻ khác nhau  $q$  và  $p$ ,  $n = pq$ , khi đó  $a \in \mathbf{Z}_n^*$  là một thặng dư bậc hai modulo  $n$  nếu và chỉ nếu  $a \in Q_p$  và  $a \in Q_q$ . Điều đó dẫn tới:

$$|Q_n| = |Q_q| \cdot |Q_p| = \frac{(p-1)(q-1)}{4}$$

$$\text{và} \quad |\bar{Q}_n| = \frac{3(p-1)(q-1)}{4}$$

**Ví dụ 3.13:** Cho  $n = 21$ . Khi đó :

$$Q_{21} = \{1, 4, 16\}, \quad \bar{Q}_{21} = \{2, 5, 8, 10, 11, 13, 17, 19, 20\}$$

**Định nghĩa 3.19:**

Cho  $a \in Q_n$ , nếu  $x \in \mathbf{Z}_n^*$  thỏa mãn  $x^2 \equiv a \pmod{n}$  thì  $x$  được gọi là căn bậc hai của  $a \bmod n$ .

**Định lý 3.15:** (Số các căn bậc hai).

Nếu  $p$  là một số nguyên tố lẻ và  $a \in Q_n$  thì  $a$  được gọi là căn bậc hai theo modulo  $p$ .

Tổng quát hơn, cho  $n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k}$ , trong đó  $p_i$  là các số nguyên tố lẻ phân biệt và  $e_i \geq 1$ . Nếu  $a \in Q_n$  thì có đúng  $2^k$  căn bậc hai khác nhau theo modulo  $n$ .

**Ví dụ 3.14:**

- Các căn bậc 2 của  $12 \bmod 37$  là 7 và 30.
- Các căn bậc 2 của  $121 \bmod 315$  là 11, 74, 101, 151, 164, 214, 241 và 304.

### 3.1.4. Các thuật toán trong $Z_n$

Cho  $n$  là một số nguyên dương. Các phần tử của  $Z_n$  sẽ được biểu thị bởi các số nguyên  $Q_{21} = \{0, 1, 2, \dots, n-1\}$ .

Ta thấy rằng, nếu  $a, b \in Z_n$  thì

$$(a+b) \bmod n = \begin{cases} a+b & a+b < n \\ a+b-n & a+b \geq n \end{cases} \quad (3.11)$$

Bởi vậy phép cộng (và trừ) theo modulo có thể thực hiện được mà không cần phép chia dài. Phép nhân modulo của  $a$  và  $b$  có thể được thực hiện bằng cách nhân các số nguyên thông thường rồi lấy phần dư của kết quả sau khi chia cho  $n$ . Các phần tử nghịch đảo trong  $Z_n$  có thể được tính bằng cách dùng thuật toán Euclide mở rộng được mô tả dưới đây.

#### 3.1.4.1. Thuật toán (Tính các nghịch đảo trong $Z_n$ )

VÀO:  $a \in Z_n$

RA :  $a^{-1} \bmod n$  (nếu tồn tại).

(1) Dùng thuật toán Euclide mở rộng để tìm các số nguyên  $x$  và  $y$  sao cho  $ax + ny = d$  trong đó  $d = (a, n)$ .

(2) Nếu  $d > 1$  thì  $a^{-1} \bmod n$  không tồn tại.

Ngược lại *return* ( $x$ ).

Phép lũy thừa theo modulo có thể được thực hiện có hiệu quả bằng thuật toán nhân và bình phương có lặp. Đây là một thuật toán rất quan trọng trong nhiều thủ tục mật mã. Cho biểu diễn nhị phân của  $k$  là:

$$\sum_{i=0}^t k_i 2^i \text{ trong đó mỗi } k_i \in \{0, 1\} \text{ khi đó}$$

$$a^k = \prod_{i=0}^t a^{k_i 2^i} = (a^{2^0})^{k_0} (a^{2^1})^{k_1} \dots (a^{2^t})^{k_t} \quad (3.12)$$

#### 3.1.4.2. Thuật toán nhân và bình phương có lặp để lấy lũy thừa trong $Z_n$

VÀO:  $a \in Z_n$  và số nguyên  $k$ , ( $0 \leq k < n$ ) có biểu diễn nhị phân:

$$k = \sum_{i=0}^t k_i 2^i$$

RA :  $a^k \bmod n$

(1) Đặt  $b \leftarrow 1$ . Nếu  $k = 0$  thì *return* ( $b$ )

- (2) Đặt  $A \leftarrow a$ .
- (3) Nếu  $k_0 = 1$  thì đặt  $b \leftarrow a$ .
- (4) For  $i$  from 1 to  $t$  do
  - (4.1). Đặt  $A \leftarrow A^2 \bmod n$ .
  - (4.2). Nếu  $k_i = 1$  thì đặt  $b \leftarrow A b \bmod n$
- (5) Return (b)

**Ví dụ 3.15:** Bảng 3.5 sau chỉ ra các bước tính toán  $5^{596} \bmod 1234 = 1013$

Bảng 3.5. Tính  $5^{596} \bmod 1234$

i	0	1	2	3	4	5	6	7	8	9
$k_i$	0	0	1	0	1	0	1	0	0	1
A	5	25	625	681	1011	369	421	779	947	925
b	1	1	625	625	67	67	1059	1059	1059	1013

Số các phép toán bit đối với phép toán cơ bản trong  $\mathbf{Z}_n$  được tóm lược trong bảng 3.6 dưới đây.

Bảng 3.6. Độ phức tạp bit của các phép toán cơ bản trong  $\mathbf{Z}_n$

Phép toán		Độ phức tạp bit
Cộng modulo	$a + b$	$O(\lg n)$
Trừ modulo	$a - b$	$O(\lg n)$
Nhân modulo	$a \cdot b$	$O((\lg n)^2)$
Nghịch đảo modulo	$a^{-1} \bmod n$	$O((\lg n)^2)$
Luỹ thừa modulo	$a^k \bmod n, k < n$	$O((\lg n)^3)$

### 3.1.5. Các ký hiệu Legendre và Jacobi

Ký hiệu Legendre là một công cụ hữu ích để xem xét liệu một số nguyên  $a$  có là một thặng dư bậc hai theo modulo của một số nguyên tố  $p$  hay không.

### 3.1.5.1. Ký hiệu Legendre

**Định nghĩa 3.20:**

Cho  $p$  là một số nguyên tố lẻ và  $a$  là một số nguyên. Ký hiệu Legendre  $\left(\frac{a}{p}\right)$  được xác định như sau:

$$\left(\frac{a}{p}\right) = \begin{cases} 0 & p \mid a \\ 1 & a \in Q_p \\ -1 & a \in \overline{Q}_p \end{cases} \quad (3.13)$$

### 3.1.5.2. Các tính chất của ký hiệu Legendre

Cho  $p$  là một số nguyên tố lẻ và  $a, b \in \mathbf{Z}$ . Khi đó ký hiệu Legendre có các tính chất sau:

$$(1) \quad \left(\frac{a}{p}\right) \equiv a^{(p-1)/2} \pmod{p}. \quad \text{Đặc biệt } \left(\frac{1}{p}\right) = 1 \quad \text{và} \quad \left(-\frac{1}{p}\right) = (-1)^{(p-1)/2} \quad \text{Bởi vậy}$$

$$-1 \in Q_p \text{ nếu } p \equiv 1 \pmod{4} \text{ và } -1 \in \overline{Q}_p \text{ nếu } p \equiv 3 \pmod{4}$$

$$(2) \quad \left(\frac{ab}{p}\right) \equiv \left(\frac{a}{p}\right) \cdot \left(\frac{b}{p}\right). \quad \text{Bởi vậy nếu } a \in Z_p^* \text{ thì } \left(\frac{a^2}{p}\right) = 1.$$

$$(3) \quad \text{Nếu } a \equiv b \pmod{p} \text{ thì } \left(\frac{a}{p}\right) = \left(\frac{b}{p}\right).$$

$$(4) \quad \left(\frac{2}{p}\right) = (-1)^{(p^2-1)/8}. \quad \text{Bởi vậy } \left(\frac{2}{p}\right) = 1 \text{ nếu } p \equiv 1 \text{ hoặc } 7 \pmod{8} \text{ và } \left(\frac{2}{p}\right) = -1 \text{ nếu}$$

$$p \equiv 3 \text{ hoặc } 5 \pmod{8}.$$

(5) Luật thuận nghịch bậc 2:

Giả sử  $p$  là một số nguyên tố lẻ khác với  $q$ , khi đó:

$$\left(\frac{p}{q}\right) = \left(\frac{q}{p}\right) (-1)^{(p-1)(q-1)/4}$$

Nói một cách khác  $\left(\frac{p}{q}\right) = \left(\frac{q}{p}\right)$  trừ phi cả  $p$  và  $q$  là đồng dư với  $3 \pmod{4}$ , trong

$$\text{trường hợp này } \left(\frac{p}{q}\right) = -\left(\frac{q}{p}\right).$$

Dấu hiệu Jacobi là tổng quát hoá của ký hiệu Legendre đối với các số nguyên lẻ  $n$  không nhất thiết là một số nguyên tố.

### 3.1.5.3. Ký hiệu Jacobi

#### Định nghĩa 3.21:

Cho  $n \geq 3$  là các số nguyên tố có phân tích  $n = p_1^{e_1} \cdot p_2^{e_2} \dots p_k^{e_k}$ . Khi đó ký hiệu Jacobi  $\left(\frac{a}{n}\right)$  được định nghĩa là

$$\left(\frac{a}{n}\right) = \left(\frac{a}{p_1}\right)^{e_1} \left(\frac{a}{p_2}\right)^{e_2} \dots \left(\frac{a}{p_k}\right)^{e_k} \quad (3.14)$$

Ta thấy rằng nếu  $n$  là một số nguyên tố thì ký hiệu Jacobi chính là ký hiệu Legendre.

#### 3.1.5.4. Các tính chất của ký hiệu Jacobi

Cho  $n \geq 3$  là các số nguyên tố  $a, b \in \mathbf{Z}$ . Khi đó ký hiệu Jacobi có các tính chất sau:

$$(1) \left(\frac{a}{n}\right) = 0, 1 \text{ hoặc } -1. \text{ Hơn nữa } \left(\frac{a}{n}\right) = 0 \text{ nếu và chỉ nếu } UCLN(a, n) \neq 1.$$

$$(2) \left(\frac{ab}{n}\right) \equiv \left(\frac{a}{n}\right) \cdot \left(\frac{b}{n}\right). \text{ Bởi vậy } a \in \mathbf{Z}_n^* \text{ thì } \left(\frac{a^2}{n}\right) = 1$$

$$(3) \left(\frac{a}{m \cdot n}\right) \equiv \left(\frac{a}{m}\right) \cdot \left(\frac{a}{n}\right).$$

$$(4) \text{ Nếu } a \equiv b \pmod{n} \text{ thì } \left(\frac{a}{n}\right) = \left(\frac{b}{n}\right).$$

$$(5) \left(\frac{1}{n}\right) = 1$$

$$(6) \left(-\frac{1}{n}\right) = (-1)^{(n-1)/2}. \text{ Bởi vậy: } \left(-\frac{1}{n}\right) = 1 \text{ nếu } n \equiv 1 \pmod{4}$$

$$\left(-\frac{1}{n}\right) = -1 \text{ nếu } n \equiv 3 \pmod{4}$$

$$(7) \left(\frac{2}{n}\right) = (-1)^{(n^2-1)/8}. \text{ Bởi vậy: } \left(\frac{2}{n}\right) = 1 \text{ nếu } n \equiv 1 \text{ hoặc } 7 \pmod{8}$$

$$\left(\frac{2}{n}\right) = -1 \text{ nếu } n \equiv 3 \text{ hoặc } 5 \pmod{8}$$

$$(8) \left(\frac{m}{n}\right) = \left(\frac{n}{m}\right) (-1)^{(m-1)(n-1)/4}$$

Nói một cách khác  $\left(\frac{m}{n}\right) = \left(\frac{n}{m}\right)$  trừ phi cả hai số  $m$  và  $n$  đều đồng dư với  $3 \pmod{4}$ , trong trường hợp này  $\left(\frac{m}{n}\right) = -\left(\frac{n}{m}\right)$ .

Từ các tính chất của ký hiệu Jacobi ta thấy rằng  $n$  lẻ và  $a = 2^e a_1$  trong đó  $a_1$  là một số lẻ thì:

$$\left(\frac{a}{n}\right) = \left(\frac{2^e}{n}\right) \left(\frac{a_1}{n}\right) = \left(\frac{2}{n}\right)^e \left(\frac{n \bmod a_1}{a_1}\right) (-1)^{(a_1-1)(n-1)/4}$$

Từ công thức này ta có thể xây dựng thuật toán đệ quy sau để tính  $\left(\frac{a}{n}\right)$  mà không cần phải phân tích  $n$  ra các thừa số nguyên tố.

### 3.1.5.5. Thuật toán tính toán ký hiệu Jacobi (và ký hiệu Legendre)

JACOBI  $(a, n)$

VÀO: Số nguyên lẻ  $n \geq 3$  số nguyên  $a$ ,  $(0 \leq a \leq n)$

RA : Ký hiệu Jacobi  $\left(\frac{a}{n}\right)$  (Sẽ là ký hiệu Legendre khi  $n$  là số nguyên tố)

- (1) Nếu  $a = 0$  thì *return* (0)
- (2) Nếu  $a = 1$  thì *return* (1)
- (3) Viết  $a = 2^e a_1$ , trong đó  $a_1$  là một số lẻ
- (4) Nếu  $e$  chẵn thì đặt  $s \leftarrow 1$ . Ngược lại hãy đặt  $s \leftarrow -1$  nếu  $n \equiv 1$  hoặc  $7 \pmod{8}$
- (5) Nếu  $n \equiv 3 \pmod{4}$  và  $a_1 \equiv 3 \pmod{4}$  thì đặt  $s \leftarrow -s$
- (6) Đặt  $r_1 \leftarrow n \bmod a_1$
- (7) Return  $(s \text{ JACOBI}(n_1, a_1))$

Thuật toán trên có thời gian chạy chừng  $O((\lg n)^2)$  các phép toán bit.

### 3.1.5.6. Nhận xét (tìm các thặng dư bậc hai theo modulo của số nguyên tố $p$ )

Cho  $p$  là một số nguyên tố lẻ. Mặc dù đã biết rằng một nửa các phần tử trong  $\mathbf{Z}_p^*$  là các thặng dư không bậc hai theo modulo  $p$  nhưng không có một thuật toán xác định theo thời gian đa thức nào được biết để tìm.

Một thuật toán ngẫu nhiên tìm một thặng dư không bậc hai là chọn ngẫu nhiên các số nguyên  $a \in \mathbf{Z}_p^*$  cho tới khi số đó thoả mãn  $\left(\frac{a}{p}\right) = -1$ . Phép lặp đối với số được chọn trước khi tìm được một thặng dư bậc hai là 2 và bởi vậy thuật toán được thực hiện theo thời gian đa thức.

### 3.1.5.7. Ví dụ tính toán ký hiệu Jacobi

Cho  $a = 158$  và  $n = 235$ . Thuật toán trên tính  $\left(\frac{158}{235}\right)$  như sau:

$$\begin{aligned}\left(\frac{158}{235}\right) &= \left(\frac{2}{235}\right)\left(\frac{79}{235}\right) = (-1)\left(\frac{235}{79}\right)(-1)^{78 \cdot 234/4} = \left(\frac{77}{79}\right) \\ &= \left(\frac{77}{79}\right)(-1)^{76 \cdot 78/4} = \left(\frac{2}{77}\right) = -1\end{aligned}$$

Khác với ký hiệu Legendre, ký hiệu Jacobi  $\left(\frac{a}{n}\right)$  không cho biết liệu  $a$  có phải là một thặng dư bậc 2 theo modulo  $n$  hay không. Sự thực là nếu  $a \in Q_n$  thì  $\left(\frac{a}{n}\right) = 1$  Tuy nhiên  $\left(\frac{a}{n}\right) = 1$  thì không có nghĩa là  $a \in Q_n$ .

### 3.1.5.8. Ví dụ (Các thặng dư bậc 2 và không bậc 2)

Bảng 3.7. Các ký hiệu Jacobi của các phần tử trong  $\mathbf{Z}_{21}^*$

$a \in \mathbf{Z}_{21}^*$	1	2	4	5	8	10	11	13	16	17	19	20
$a^2 \bmod n$	1	4	16	4	1	16	16	1	4	16	4	1
$\left(\frac{a}{3}\right)$	1	-1	1	-1	-1	1	-1	1	1	-1	1	-1
$\left(\frac{a}{7}\right)$	1	1	1	-1	1	-1	1	-1	1	-1	-1	-1
$\left(\frac{a}{21}\right)$	1	-1	1	1	-1	-1	-1	-1	1	1	-1	-1

Bảng 1.6 liệt kê các phần tử trong  $\mathbf{Z}_{21}^*$  và các ký hiệu Jacobi của chúng. Từ ví dụ trong phần c ta có  $Q_{21} = \{1, 4, 16\}$ . Ta thấy rằng  $\left(\frac{5}{21}\right) = 1$  nhưng  $5 \notin Q_{21}$ .



**Định nghĩa 3.22:**

Cho  $n \geq 3$  là các số nguyên tố lẻ và cho  $J_n = \left\{ a \in Z_n^* \mid \left( \frac{a}{n} \right) = 1 \right\}$  tập các thặng dư giả bậc 3 theo modulo  $n$  (Ký hiệu  $\hat{Q}_n$ ) được định nghĩa là tập  $J_n - Q_n$ .

**Định lý 3.16:**

Cho  $n = pq$  là tích của hai số nguyên tố lẻ khác nhau. Khi đó:

$$|Q_n| = |\tilde{Q}_n| = (p-1)(q-1)/4 \quad (3.15)$$

tức là một nửa các phần tử trong  $J_n$  là các thặng dư giả bậc hai.

### 3.1.6. Các số nguyên Blum

**Định nghĩa 3.23:**

Số nguyên Blum là một hợp số có dạng  $n = pq$ , trong đó  $p$  và  $q$  là các số nguyên tố khác nhau và thoả mãn:

$$\begin{aligned} p &\equiv 3 \pmod{4} \\ q &\equiv 3 \pmod{4} \end{aligned} \quad (3.16)$$

**Định lý 3.17:**

Cho  $n = pq$  là một số nguyên Blum và cho  $a \in Q_n$ . Khi đó  $a$  có đúng 4 căn bậc hai modulon và chỉ có một số nằm trong  $Q_n$ .

**Định nghĩa 3.24:**

Cho  $n$  là một số nguyên Blum và cho  $a \in Q_n$ . Căn bậc hai duy nhất của  $a$  nằm trong  $Q_n$  được gọi là căn bậc hai chính  $a \pmod{n}$ .

### 3.1.7. Ví dụ (Số nguyên Blum)

Đối với số nguyên Blum  $n = 21$ . Ta có  $J_n = \{1, 4, 5, 16, 17, 20\}$  và  $\tilde{Q}_n = \{5, 17, 20\}$ . Bốn căn bậc 2 của  $a = 4$  là 2, 5, 16 và 19, trong đó chỉ có 16 là cũng nằm trong  $Q_n$ . Bởi vậy 16 là căn bậc 2 chính của  $4 \pmod{21}$ .

**Định lý 3.18:**

Nếu  $n = pq$  là một số nguyên Blum thì ánh xạ.

$f: Q_n \rightarrow Q_n$  được xác định bởi  $f(x) = x^2 \pmod{n}$  là một phép hoán vị.

Ánh xạ ngược của  $f$  là:  $f^{-1}(x) = x^{((p-1)(q-1)+4/8)} \pmod{n}$ .

### 3.2. GIỚI THIỆU VỀ MẬT MÃ KHOÁ CÔNG KHAI

Trong mô hình mật mã cổ điển trước đây mà hiện nay đang được nghiên cứu Alice (người gửi) và Bob (người nhận) chọn một cách bí mật khoá  $K$ . Sau đó dùng  $K$  để tạo luật mã hoá  $e_k$  và luật giải mã  $d_k$ . Trong hệ mật này  $d_k$  hoặc giống  $e_k$  hoặc dễ dàng nhận được từ nó (ví dụ trong hệ DES quá trình giải mã hoàn toàn tương tự như quá trình mã nhưng thủ tục khoá ngược lại). Các hệ mật thuộc loại này được gọi là hệ khoá bí mật, nếu để lộ  $e_k$  thì làm cho hệ thống mất an toàn.

Nhược điểm của hệ mật này là nó yêu cầu phải có thông tin trước về khoá  $K$  giữa Alice và Bob qua một kênh an toàn trước khi gửi một bản mã bất kỳ. Trên thực tế điều này rất khó đảm bảo. Chẳng hạn khi Alice và Bob ở cách xa nhau và họ chỉ có thể liên lạc với nhau bằng thư tín điện tử (E.mail). Trong tình huống đó Alice và Bob không thể tạo một kênh bảo mật với giá phải chăng.

Ý tưởng xây dựng một hệ mật khoá công khai (hay dùng chung) là tìm một hệ mật không có khả năng tính toán để xác định  $d_k$  khi biết  $e_k$ . Nếu thực hiện được như vậy thì quy tắc mã  $e_k$  có thể được công khai bằng cách công bố nó trong một danh bạ (bởi vậy nên có thuật ngữ *hệ mật khoá công khai*). Ưu điểm của hệ mật khoá công khai là ở chỗ Alice (hoặc bất kỳ ai) có thể gửi một bản tin đã mã cho Bob (mà không cần thông tin trước về khoá mật) bằng cách dùng mật mã công khai  $e_k$ . Người nhận A sẽ là người duy nhất có thể giải được bản mã này bằng sử dụng luật giải bí mật  $d_k$  của mình.

Có thể hình dung hệ mật này tương tự như sau. Alice đặt một vật vào một hộp kim loại và rồi khoá nó lại bằng một khoá số do Bob để lại. Chỉ có Bob là người duy nhất có thể mở được hộp vì chỉ có anh ta mới biết tổ hợp mã của khoá số của mình.

Ý tưởng về một hệ mật khoá công khai được Diffie và Hellman đưa ra vào năm 1976. Còn việc hiện thực hoá nó thì do Rivesrt, Shamir và Adleman đưa ra lần đầu tiên vào năm 1977, họ đã tạo nên hệ mật nổi tiếng RSA (sẽ được nghiên cứu trong chương này).

Một chú ý quan trọng là một hệ mật khoá công khai không bao giờ có thể đảm bảo được độ mật tuyệt đối (an toàn vô điều kiện). Sở dĩ như vậy vì đối phương khi nghiên cứu một bản mã, anh ta có thể *mã lần lượt các bản tin rõ* bằng luật mã hoá công khai  $e_k$  cho tới khi anh ta tìm được bản rõ duy nhất  $M$  đảm bảo  $C = e_k(M)$ . Bản rõ này chính là kết quả giải mã. Bởi vậy, ta chỉ nghiên cứu độ mật về mặt tính toán của các hệ mật này.

Một khái niệm có ích khi nghiên cứu hệ mật khoá công khai là khái niệm về *hàm cửa sập một chiều*. Ta sẽ định nghĩa khái niệm này một cách không hình thức.

Hàm mã khoá công khai  $e_k$  của Bob phải là một hàm dễ tính toán. Song việc tìm hàm ngược (hàm giải mã) rất khó khăn (đối với bất kỳ ai không phải là Bob). Đặc tính dễ tính toán hàm ngược thường được gọi là đặc tính một chiều. Bởi vậy điều kiện cần thiết là  $e_k$  phải là hàm một chiều (tính thuận đơn giản, nhưng tính ngược rất phức tạp).

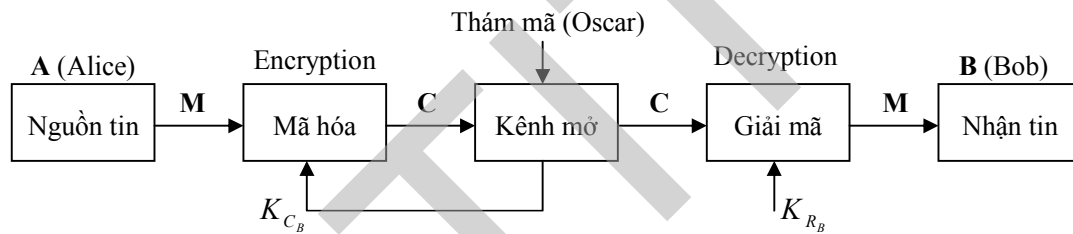
Các hàm một chiều đóng vai trò quan trọng trong mật mã học, chúng rất quan trọng trong các hệ mật khoá công khai và trong nhiều lĩnh vực khác. Đáng tiếc là mặc dù có rất

nhiều hàm được coi là hàm một chiều nhưng cho đến nay vẫn không tồn tại một hàm nào có thể chứng minh được là hàm một chiều.

Sau đây là một ví dụ về một hàm được coi là hàm một chiều. Giả sử  $n$  là tích của hai số nguyên tố lớn  $p$  và  $q$ , giả sử  $b$  là một số nguyên dương. Khi đó ta xác định ánh xạ  $f: \mathbb{Z}_n \rightarrow \mathbb{Z}_n$  là  $f(x) = x^b \bmod n$  (với  $b$  và  $n$  đã được chọn thích hợp thì đây chính là hàm mã RSA, sau này ta sẽ nói nhiều hơn về nó).

Để xây dựng một hệ mật khoá công khai thì việc tìm được một hàm một chiều vẫn chưa đủ. Ta không muốn  $e_k$  là hàm một chiều đối với Bob vì anh ta phải có khả năng giải mã các bản tin nhận được một cách hiệu quả. Điều cần thiết là Bob phải có một cửa sập chứa thông tin bí mật cho phép dễ dàng tìm hàm của  $e_k$ . Như vậy Bob có thể giải mã một cách hữu hiệu vì anh ta có một hiểu biết tuyệt mật nào đó về  $K$ . Bởi vậy một hàm được gọi là cửa sập một chiều nếu nó là một hàm một chiều và nó trở nên dễ tính ngược nếu biết một cửa sập nhất định.

### 3.3. SƠ ĐỒ CHỨC NĂNG CỦA HỆ MẬT KHÓA CÔNG KHAI



Hình 3.1. Sơ đồ chức năng hệ mật khóa công khai

Sơ đồ truyền tin bí mật từ A đến B sử dụng mật mã khóa công khai được mô tả trong Hình 3.1, trong đó:

- +  $M$  : bản tin rõ
- +  $C$  : Bản mã
- +  $K_{C_B}$  là khóa công khai của B (khóa mã hóa) được lấy trên kênh mở.
- +  $K_{R_B}$  là khóa bí mật của B (Khóa giải mã).

Hàm *mã hóa* là một ánh xạ 1:1:

$$C = E(M, K_{C_B}) \quad (3.17)$$

Và hàm *giải mã*:

$$M = E^{-1}(C, K_{R_B}) \quad (3.18)$$

Theo sơ đồ của hệ mật khóa công khai ta thấy các ưu điểm là:

- Không cần hai khóa bí mật.
- Không cần kênh an toàn riêng

- Biết khóa mã hóa trên kênh mở nhưng rất khó giải mã, tức là biết  $K_{C_B}$  nhưng rất khó suy ra được  $K_{R_B}$  (độ khó ở đây chính là độ phức tạp tính toán hoặc tài nguyên và thời gian tính toán)

Các nghiên cứu trên thế giới từ năm 1976 cho đến nay đã đưa ra được 5 bài toán một chiều như sau:

- Bài toán logarit rời rạc
- Bài toán phân tích thừa số (gắn với hệ mật nổi tiếng RSA)
- Bài toán xếp ba lô
- Bài toán mã sửa sai
- Bài toán xây dựng hệ mật trên đường cong elliptic.

Trong phần tiếp theo dưới sẽ lần lượt tìm hiểu từng bài toán một chiều và ứng dụng trong các hệ mật liên quan.

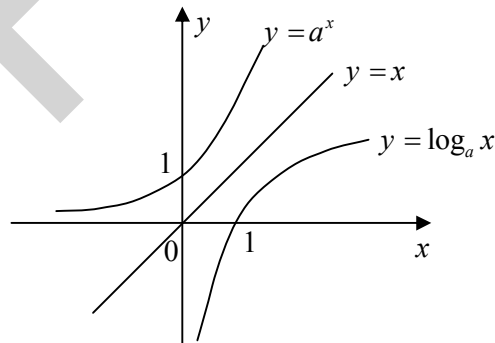
### 3.4. BÀI TOÁN LOGARIT RỜI RẠC VÀ CÁC HỆ MẬT LIÊN QUAN

#### 3.4.1. Bài toán logarit rời rạc

##### 3.4.1.1. Bài toán logarit trên trường số thực $R$

+ *Bài toán thuận*: Hàm số  $y = a^x$  với  $a, x \in R$ , việc tính toán hàm mũ này có thể được thực hiện dễ dàng bằng thuật toán nhân và bình phương.

+ *Bài toán ngược*: như ta đã biết phép tính ngược của hàm mũ chính là hàm logarit  $y = \log_a x$ , việc tính toán hàm ngược logarit này sẽ khó khăn hơn nhiều so với hàm thuận. Tuy nhiên, cả hai phép hàm mũ và logarit đều là các hàm đồng biến cho nên có thể xác định giá trị tương đối của hàm logarit được (như Hình 3.2).



Hình 3.2. Đồ thị hàm  $y = a^x$  và  $y = \log_a x$

Một số tính chất của hàm logarit.

$$+ y = \log_a bc = \log_a b + \log_a c$$

$$+ y = \log_a \frac{b}{c} = \log_a b - \log_a c$$

$$+ \log_a 1 = 0$$

$$+ y = \log_a x^{-1} = -\log_a x$$

### 3.4.1.2. Bài toán logarit trên trường hữu hạn

Xét vành  $\mathbf{Z}_p$  với  $p$  là số nguyên tố. Theo định lý 2.7 nếu  $p$  là nguyên tố thì  $\mathcal{C}_p$  là một trường ( $\mathcal{C}_p = \text{GF}(2)$ ).

Tập tất cả các phần tử khác không của trường sẽ tạo nên một nhóm nhân xyclic  $\mathcal{C}_p^*$ .

$$\mathcal{C}_p^* = \mathcal{C}_p / \{0\} = \{1, 2, \dots, p-1\}$$

+ **Bài toán thuận:**  $y = a^x \bmod p, (a, x \in \mathcal{C}_p^*)$

**Ví dụ 3.16:** Xét  $p = 19, a = 2$  ta có các giá trị  $y = a^x$  như trong Bảng 3.8

Bảng 3.8. Các giá trị của  $y = 2^x \bmod 19$  trên  $\mathcal{C}_{19}^*$

$x$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$2^x$	2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1

Chú ý:

- + Nếu  $a$  là một phần tử nguyên thủy thì  $a^x$  sẽ đi qua tất cả các phần tử của nhóm.
- + Nếu  $a$  là phần tử nguyên thủy thì  $a^i$  cũng là nguyên thủy với  $(i, p-1) = 1$  ( $p$  là số nguyên tố).

Trong ví dụ 3.16 các giá trị của  $i$  thỏa mãn  $(i, 18) = 1$  là  $i = (1, 5, 7, 11, 13, 17)$ . Số lượng các giá trị của  $i$  bằng giá trị hàm  $\varphi(p-1)$ .

$$N_i = \varphi(p-1) = \varphi(18) = 6$$

Cách tính hàm Phi-Euler  $\varphi$  như trình bày tại mục 1.1.12.

Như vậy trong nhóm  $\mathcal{C}_{19}^*$  có 6 phần tử nguyên thủy:

$$2 = 2^1; \quad 13 = 2^5; \quad 14 = 2^7; \quad 15 = 2^{11}; \quad 3 = 2^{13}; \quad 10 = 2^{17} \quad (3.19)$$

Các phần tử nguyên thủy này tạo thành các cặp nghịch đảo như sau:

$$(2, 10) \leftrightarrow 2 = 10^{-1}$$

$$(13, 3) \leftrightarrow 13 = 3^{-1}$$

$$(14, 15) \leftrightarrow 14 = 15^{-1}$$

+ **Bài toán ngược:**  $y = \log_a x, (a, x \in \mathcal{C}_p^*)$

Từ bảng 3.8 ta tính được hàm ngược  $\log_2 x$  như trong bảng 3.9.

Bảng 3.9. Giá trị  $\log_2 x \pmod{19}$  trên  $\mathbb{Z}_{19}^*$

$x$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$2^x$	2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
$\log_2 x$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9

Chú ý: vì  $2^{18} = 1$  nên  $\log_2 1 = 18$ .

Một số tính chất của hàm logarit rời rạc.

$$+ \quad y = \log_a bc = (\log_a b + \log_a c) \pmod{p-1}$$

$$+ \quad y = \log_a \frac{b}{c} = (\log_a b - \log_a c) \pmod{p-1}$$

$$+ \quad \log_{a^{-1}} x = -\log_a x = p-1 - \log_a x$$

$$+ \quad \log_a 1 = 0 = p-1 \text{ (coi } 0 = p-1 \text{)}$$

**Nhận xét:** Từ hai Bảng 3.8 và Bảng 3.9 ta thấy hai hàm thuận và ngược đều không phải hàm đồng biến, khi biết bài toán thuận thì mới tìm được bài toán ngược. Do đó việc giải bài toán ngược giống bài toán vét cạn, phải thử lần lượt các trường hợp.

Việc xác định logarit của một phần tử bất kỳ trong trường là bài toán khó giải.

#### + Bài toán logarit rời rạc

Cho  $\mathbb{Z}_p^*$  với  $p$  là số nguyên tố,  $\alpha$  là một phần tử nguyên thủy  $\alpha \in \mathbb{Z}_p^*$ .

Yêu cầu tìm  $y = \log_\alpha x$  với  $\alpha, x \in \mathbb{Z}_p^*$ .

Nhận xét:  $\forall x \in \mathbb{Z}_p^*$  thì:

- Bài toán có nghiệm khi  $\alpha$  là phần tử nguyên thủy.
- Bài toán có thể không có nghiệm khi  $\alpha$  bất kỳ.

**Ví dụ 3.17:** Với trường hợp  $p = 19$  ta đã tính được 6 phần tử nguyên thủy như trong (3.19) ta sẽ đi tính bài toán logarit rời rạc với cơ số là 6 phần tử nguyên thủy này.

Tuy nhiên ta có thể áp dụng tính chất của hàm logarit rời rạc:

$$\log_{a^{-1}} x = -\log_a x = p-1 - \log_a x, \text{ hay } \log_{a^{-1}} x + \log_a x = p-1$$

để tính logarit với cơ số là các cặp số nghịch đảo.

Tức là  $(2, 10)$  là cặp số nghịch đảo, khi đó  $\log_{10} x = p-1 - \log_2 x = 18 - \log_2 x$ . Tương tự  $(13, 3)$  và  $(14, 15)$  là các cặp nghịch đảo nên  $\log_3 x = 18 - \log_{13} x$  và  $\log_{15} x = 18 - \log_{14} x$ .

Với quy tắc như thế có thể tính được các giá trị logarit như trong Bảng 3.10.

Bảng 3.10. Bài toán logarit rời rạc trên  $\phi_{19}^*$

$x$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18
$2^x$	2	4	8	16	13	7	14	9	18	17	15	11	3	6	12	5	10	1
$\log_2 x$	18	1	13	2	16	14	6	3	8	17	12	15	5	7	11	4	10	9
$\log_{10} x$	18	17	5	16	2	4	12	15	10	1	6	3	13	11	7	14	8	9
$13^x$	13	17	12	4	14	11	10	16	18	6	2	7	15	5	8	9	3	1
$\log_{13} x$	18	11	17	4	14	10	12	15	16	7	6	3	1	5	13	8	2	9
$\log_3 x$	18	7	1	14	4	8	6	3	2	11	12	15	17	13	5	10	16	9
$14^x$	14	6	8	17	10	7	3	4	18	5	13	11	2	9	12	16	15	1
$\log_{14} x$	18	13	7	8	10	2	6	3	14	5	12	15	11	1	17	16	4	9
$\log_{15} x$	18	5	11	10	8	16	12	15	4	13	6	3	7	17	1	2	14	9

Có thể tính  $13^x$  thông qua  $2^x$ , ta thấy rằng  $13 = 2^5$  do đó  $13^x = 2^{5x}$ , tương tự như thế  $14^x = 2^{7x}$ .

### 3.4.2. Một số hệ mật xây dựng trên bài toán logarit rời rạc

#### 3.4.2.1. Trao đổi và thỏa thuận khóa Diffie-Hellman

Giả sử A và B muốn liên lạc sử dụng hệ mật khoá bí mật. Để thỏa thuận một khóa K chung cho cả hai bên qua một kênh không an toàn mà không ai khác có thể biết được, A và B có thể dùng thủ tục thỏa thuận khóa Diffie-Hellman sau:

Chọn trước một số nguyên tố  $p$  thích hợp và một phần tử nguyên thủy  $\alpha \in \phi_p^*$ .

Cặp số  $(p, \alpha)$  tạo thành khóa công khai.

Quá trình trao đổi khóa được thực hiện như sau:

A	B
+ A chọn một số nguyên $x$ bí mật thỏa mãn $1 < x < p-1$ và tính:	+ B chọn một số nguyên $y$ bí mật thỏa mãn $1 < y < p-1$ và tính:
<div style="display: flex; align-items: center; justify-content: center;"> <div style="border: 1px solid black; padding: 5px; margin-right: 10px;"><math>\alpha^x \bmod p</math></div> <div style="text-align: center;"> <math>\xrightarrow{\text{Gửi cho B}}</math> </div> </div>	<div style="display: flex; align-items: center; justify-content: center;"> <div style="text-align: center;"> <math>\xleftarrow{\text{Gửi cho A}}</math> </div> <div style="border: 1px solid black; padding: 5px; margin-left: 10px;"><math>\alpha^y \bmod p</math></div> </div>
Sau đó A gửi giá trị này cho B.	Sau đó B gửi giá trị này cho A.
+ A nhận $\alpha^y \bmod p$ và tính ra khóa dùng chung:	+ B nhận $\alpha^x \bmod p$ và tính ra khóa dùng chung:

$$K = (a^y)^x \bmod p = a^{xy} \bmod p$$

$$K = (a^x)^y \bmod p = a^{xy} \bmod p$$

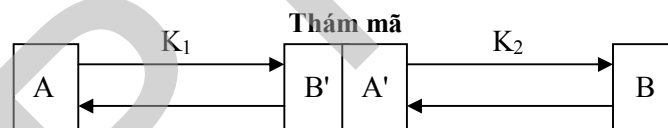
**Ví dụ 3.18:** Giả sử A và B chọn  $p = 19$  và  $a = 2$

A	B
+ A chọn một số nguyên $x = 3$ và tính: $2^3 \bmod 19 = 8$ , sau đó A gửi 8 cho B	+ B chọn một số nguyên $y = 5$ và tính: $2^5 \bmod 19 = 13$ , B gửi giá trị 13 cho A.
+ A nhận 13 và tính ra khóa dùng chung: $K = 13^3 \bmod 19 = 12$	+ B nhận 8 và tính ra khóa dùng chung: $K = 8^5 \bmod 19 = 12$

*Nhận xét:*

+ Thăm mã biết  $(a, p)$  nhưng không biết  $x, y$ , nếu muốn biết khóa dùng chung  $K = a^{xy} \bmod p$  thì thám mã phải giải bài toán logarit (bài toán ngược) để tìm  $x$  và  $y$ .

+ Tuy nhiên việc thỏa thuận theo phương thức này sẽ chịu phép tấn công "kẻ đứng giữa" (Man in the middle). Thăm mã đứng giữa sẽ giả mạo B để thỏa thuận khóa  $K_1$  dùng chung với A, đồng thời thám mã giả danh A để thỏa thuận khóa  $K_2$  dùng chung với B. Thăm mã liên lạc với A bằng khóa  $K_1$ , giải mã để lấy cắp thông tin của A, sau đó lại mã hóa thông tin của A bằng khóa  $K_2$  để liên lạc với B (và tương tự như thế theo chiều từ B đến A). Hai bên A và B vẫn nhận đúng thông tin tưởng là liên lạc đúng với nhau, nhưng thực tế là liên lạc với thám mã. Đây là điểm yếu của phương pháp thỏa thuận khóa kiểu này, để khắc phục người ta sử dụng các phương pháp xác thực (sẽ được trình bày sau).



Hình 3.3. Phép tấn công kẻ đứng giữa

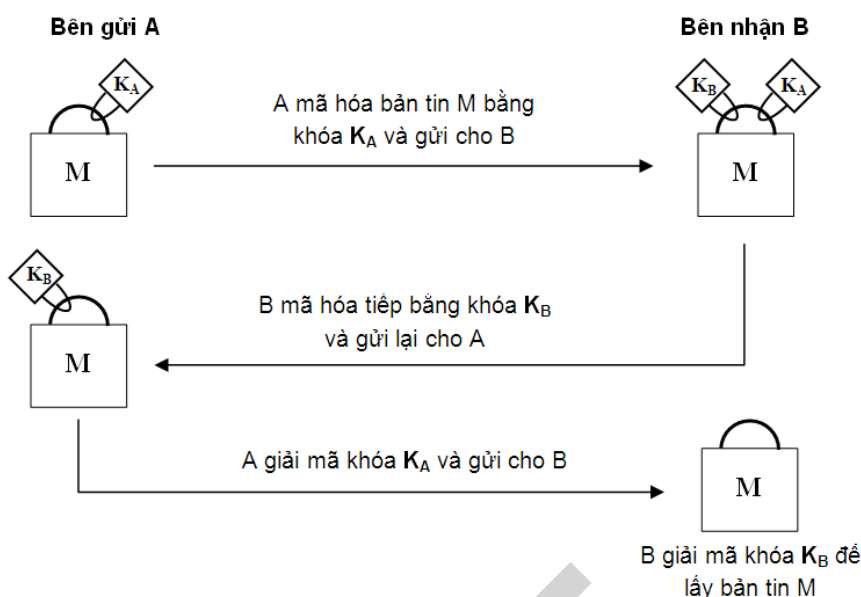
### 3.4.2.2. Hệ mật Omura-Massey

#### a) Sơ đồ hệ mật Omura – Massey

Ý tưởng của hệ mật Omura – Massey được mô tả trong Hình 3.4.

Hai bên liên lạc A và B sử dụng hai khóa bí mật khác nhau là  $K_A$  và  $K_B$ , đầu tiên bản tin M được A mã hóa bằng khóa bí mật của A là  $K_A$  và gửi bản mã cho B, tất nhiên trên đường truyền thám mã không thể có được bản tin M vì không có khóa  $K_A$ . Bên B nhận được bản mã lại thực hiện mã hóa một lần nữa bằng khóa  $K_B$  và gửi lại bản mã mới cho A. Khi A nhận lại bản mã thì tiến hành giải mã bằng khóa  $K_A$ , lúc này bản tin M chỉ được mã hóa bằng khóa  $K_B$ . A gửi bản mã này cho B, bên B nhận được và tiến hành giải mã bằng khóa  $K_B$  để lấy lại bản tin M.





Hình 3.4. Lưu đồ hệ mật Omura - Massey

Hệ mật này về cơ bản rất an toàn, tuy nhiên nó có một nhược điểm là bản mã truyền giữa A và B phải được thực hiện 3 lần, tức là tốc độ mã thấp hay dung lượng thông tin cần truyền sẽ tăng lên.

#### b) Hệ mật Omura – Massey xây dựng trên bài toán logarit rời rạc

Hai bên liên lạc A và B chọn trước  $\phi_p$ , với  $p$  là số nguyên tố.

**Tạo khóa:** Hai bên liên lạc A và B tạo cho mình khóa bí mật như sau:

A chọn  $m, n$  ngẫu nhiên thỏa mãn  $mn \equiv 1 \pmod{p-1}$ . Khóa bí mật của A là  $m, n$

B chọn  $u, v$  ngẫu nhiên thỏa mãn  $uv \equiv 1 \pmod{p-1} \rightarrow$  Khóa bí mật của B là  $u, v$

Giả sử A cần gửi một bản tin  $M$  cho B, quá trình truyền tin bảo mật sử dụng hệ mật Omura – Massey được mô tả như Hình 3.5:

$A_{(m,n)}$	$B_{(u,v)}$
+ A tính $M^m \pmod{p}$ và gửi cho B	+ B nhận $M^n \pmod{p}$ và tính $(M^m)^u \pmod{p}$
+ A nhận $(M^m)^u \pmod{p}$ và tính: $(M^{mu})^n \pmod{p} = M^u \pmod{p}$ và gửi cho B	B gửi giá trị này cho A
	+ B nhận $M^u \pmod{p}$ và giải mã ra bản tin M: $(M^u)^v \pmod{p} = M$

Hình 3.5. Mô tả hệ mật Omura – Massey sử dụng bài toán logarit rời rạc.

## CHƯƠNG 4. HÀM BĂM, XÁC THỰC VÀ CHỮ KÝ SỐ

### 4.1. CÁC HÀM BĂM VÀ TÍNH TOÀN VỆ CỦA DỮ LIỆU

#### 4.1.1. Khái niệm về hàm băm

Các hàm băm đóng vai trò cơ bản trong mật mã hiện đại. Hàm băm sẽ tạo ra một đầu ra có độ dài cố định từ bản tin đầu vào. Đầu ra này được định nghĩa là *mã băm* (kết quả băm, giá trị băm).

Nói một cách chính xác hơn, hàm băm  $h$  sẽ tạo ra ánh xạ các xâu bit có độ dài hữu hạn tùy ý thành các xâu bit có độ dài  $n$  cố định.

Xét xâu bit  $x \in (Z_2)^n$  và một ánh xạ:

$$h : x \rightarrow y \in (Z_2)^m \quad (4.1)$$

Trong đó:  $n > m$  nhận giá trị bất kỳ, còn  $m = \text{const}$

Hàm  $y = h(x)$  được gọi là hàm băm (hay mã băm, tóm lược thông báo)

Hàm băm  $h$  là một ánh xạ đầu ra có độ dài  $m$  cố định  $h : D \rightarrow R$  và  $|D| > |R|$  điều này có nghĩa là không thể tránh khỏi các va chạm (tức là cùng một giá trị đầu ra có thể có nhiều bộ giá trị vào khác nhau). Nếu hàm  $h$  là ngẫu nhiên theo nghĩa tất cả các đầu ra là đồng xác suất thì có chừng  $2^{n-m}$  các đầu vào ánh xạ tới mỗi đầu ra ( $n$  : số bit đầu vào,  $m$  : số bit đầu ra,  $n > m$ ) và 2 đầu vào được chọn ngẫu nhiên sẽ có cùng đầu ra với xác suất  $2^{-m}$  (không phụ thuộc vào  $n$ ).

Ý tưởng cơ bản của việc sử dụng các hàm băm trong mật mã là sử dụng chúng như một ảnh biểu diễn rút gọn (đôi khi còn được gọi là vết, dấu tay số hay tóm lược thông báo) của một xâu vào và có thể được dùng như thể nó chính là xâu vào đó.

Các hàm băm được dùng cho các sơ đồ chữ ký số kết hợp với việc đảm bảo tính toàn vẹn của dữ liệu, khi đó bản tin trước hết được băm và rồi giá trị băm (được xem như đại diện cho bản tin) sẽ được ký thay cho vị trí bản tin gốc.

Một lớp các hàm băm được gọi là *các mã xác thực thông báo* (MAC - Message Authentication Codes) sẽ cho phép xác thực thông báo bằng kỹ thuật đối xứng (mật mã cổ điển).

Các thuật toán MAC sử dụng 2 đầu vào (bao gồm bản tin và một khoá bí mật) để tạo ra một đầu ra có kích cỡ cố định ( $n$  bit) với ý đồ đảm bảo rằng nếu không biết khoá thì việc tạo ra cùng một đầu ra là không khả thi. MAC có thể được dùng để đảm bảo tính toàn vẹn của dữ liệu, xác thực tính nguyên bản của số liệu cũng như định danh trong sơ đồ mật mã cổ điển.

Một ứng dụng điển hình của hàm băm (không dùng khoá) để đảm bảo tính toàn vẹn của dữ liệu có thể được mô tả như sau:

Giá trị băm tương ứng với một bản tin riêng  $x$  sẽ được tính ở thời điểm  $T_1$ . Tính toàn vẹn của giá trị băm này (chứ không phải là bản thân bản tin) sẽ được bảo vệ theo một cách

nào đó. Ở thời điểm tiếp theo sau  $T_2$  phép kiểm tra sau sẽ được tiến hành để xác định xem liệu thông báo có bị sửa đổi hay không, tức là xem liệu bản tin  $x'$  có giống bản tin gốc hay không. Giá trị băm của  $x'$  sẽ được tính toán và so sánh với giá trị băm đã được bảo vệ, nếu chúng bằng nhau thì bên thu sẽ chấp nhận rằng  $x$  và  $x'$  là như nhau và như vậy có nghĩa là bản tin đã không bị sửa đổi. Như vậy vấn đề đảm bảo tính vẹn toàn của một bản tin lớn sẽ được gui về đảm bảo cho một giá trị băm có kích cỡ cố định (và nhỏ).

Ứng dụng trên thường được gọi là *mã phát hiện sự sửa đổi (MDC - Manipulation Detection Codes)*.

#### 4.1.2. Các định nghĩa, tính chất cơ bản và phân loại hàm băm

##### 4.1.2.1. Định nghĩa hàm băm

**Định nghĩa 4.1:** Hàm băm là một ánh xạ  $h(x)$  thỏa mãn hai tính chất:

- Tính chất nén:*  $h$  sẽ ánh xạ một đầu vào  $x$  có độ dài bit hữu hạn tùy ý tới một đầu ra  $h(x)$  có độ dài bit  $m$  hữu hạn.
- Tính chất dễ dàng tính toán:* Với  $h$  cho trước và một đầu vào  $x$ , có thể dễ dàng tính được  $h(x)$ .

##### 4.1.2.2. Một số tính chất của các hàm băm không có khoá

Giả sử  $h$  là một hàm băm không có khoá,  $x$  và  $x'$  là các đầu vào và  $y$  và  $y'$  là các đầu ra tương ứng. Ngoài hai tính chất cơ bản trên hàm băm mật mã còn có 3 tính chất sau:

- Tính khó tính toán nghịch ảnh:*

Đối với hầu hết các đầu ra được xác định trước, không có khả năng tính toán để tìm một đầu vào bất kỳ mà khi băm sẽ cho ra đầu ra tương ứng (Tức là tìm một nghịch ảnh  $x$  sao cho  $h(x) = y$  với  $y$  cho trước và không biến đầu vào tương ứng).

- Khó tìm nghịch ảnh thứ hai:*

Không có khả năng tính toán để tìm một đầu vào đã cho trước: Tức là với  $x$  cho trước phải tìm  $x' \neq x$  sao cho  $h(x) = h(x')$

- Tính khó va chạm.* Không có khả năng về tính toán để tìm hai đầu vào khác nhau bất kỳ  $x' \neq x$  để  $h(x) = h(x')$ .

##### 4.1.2.3. Định nghĩa hàm băm một chiều (OWHF - oneway hash function)

**Định nghĩa 4.2:** Định nghĩa hàm băm một chiều (OWHF - oneway hash function) là một hàm băm (ngoài hai tính chất cơ bản) có tính chất bổ sung là:

- Khó tìm nghịch ảnh
- Khó tìm nghịch ảnh thứ hai.

##### 4.1.2.4. Định nghĩa hàm băm khó va chạm (CRHF: Collision resistant HF)

**Định nghĩa 4.3:** Định nghĩa hàm băm khó va chạm (CRHF: Collision resistant HF) là một hàm băm (ngoài hai tính chất cơ bản) có tính chất bổ sung là:

- Khó tìm nghịch ảnh thứ hai
- Khó và chậm

#### 4.1.2.5. Chú ý về các thuật ngữ

Khó tìm nghịch ảnh  $\equiv$  Một chiều

Khó tìm nghịch ảnh thứ hai  $\equiv$  Khó và chậm yếu.

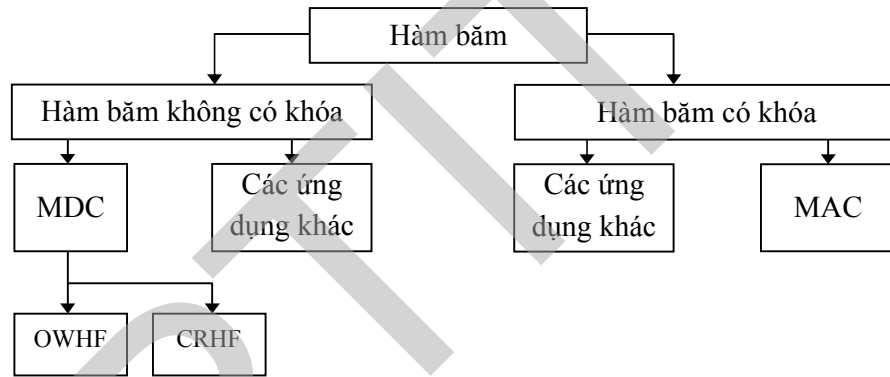
Khó và chậm  $\equiv$  Khó và chậm mạnh

OWHF  $\equiv$  Hàm băm một chiều yếu.

CRHF  $\equiv$  Hàm băm một chiều mạnh.

Ví dụ:  $r$  bit kiểm tra của một mã xyclic  $(n, k)$  với  $k > r$  có thể coi là một hàm băm thoả mãn hai tính chất cơ bản (dễ tính toán và nén). Tuy nhiên nó không thoả mãn tính chất khó tìm nghịch ảnh thứ hai.

#### 4.1.2.6. Phân loại các hàm băm mật mã và ứng dụng



Hình 4.1. Phân loại hàm băm

MDC: Manipulation Detection Code – Mã phát hiện sửa đổi

MAC: Message Authentication Code – Mã xác thực thông báo.

#### 4.1.3. Các hàm băm không có khóa (MDC)

*Các hàm băm không khóa dựa trên mật mã khối*

##### **Định nghĩa 4.4:**

Mật mã khối  $(n, r)$  là một mã khối xác định một hàm khả nghịch từ các bản rõ  $n$  bit sang các bản mã  $n$  bit bằng cách sử dụng một khoá  $r$  bit. Nếu  $E$  là một phép mã hoá như vậy thì  $E_k(x)$  ký hiệu cho phép mã hoá  $x$  bằng khoá  $k$ .

##### **Định nghĩa 4.5:**

Cho  $h$  là một hàm băm có lặp  $h$  được xây dựng từ một mật mã khối với hàm nén  $f$  thực hiện  $s$  phép mã hoá khối để xử lý từng khối bản tin  $n$  bit. Khi đó tốc độ của  $h$  là  $1/s$ .

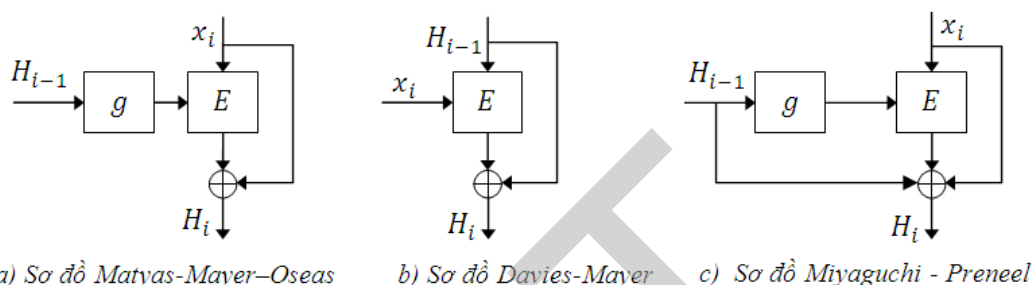
### 4.1.3.1. MDC độ dài đơn

Ba sơ đồ trong Hình 4.2 dưới đây có liên quan chặt chẽ với các hàm băm độ dài đơn, xây dựng trên các mật mã khối. Các sơ đồ này có sử dụng các thành phần được xác định trước như sau:

Một mật mã khối  $n$  bit khởi sinh  $E_k$  được tham số hoá bằng một khoá đối xứng  $k$ .

Một hàm  $g$  ánh xạ  $n$  bit vào thành khoá  $k$  sử dụng cho  $E$  (Nếu các khoá cho  $E$  cũng có độ dài  $n$  thì  $g$  có thể là hàm đồng nhất).

Một giá trị ban đầu cố định IV (Init Vector) thích hợp để dùng với  $E$ .



Hình 4.2. Một số phương pháp xây dựng hàm băm đơn

*Thuật toán băm Matyas - Mayer - Oseas (M-M-O)*

VÀO: Xâu bit  $x$

RA : Mã băm  $n$  bit của  $x$

Đầu vào  $x$  được phân chia thành các khối  $n$  bit và được đệm nếu cần thiết nhằm tạo khối cuối cùng hoàn chỉnh. Ta được  $t$  khối  $n$  bit:  $x = (x_1, x_2, \dots, x_t)$ . Phải xác định trước một giá trị ban đầu  $n$  bit (ký hiệu IV).

Đầu ra là  $H_t$  được xác định như sau:

$$\begin{cases} H_0 = IV \\ H_i = E_{g(H_{i-1})}(x_i) \oplus x_i, \quad 1 \leq i \leq t \end{cases}$$

*Thuật toán băm Davies - Mayer (D-M)*

VÀO: Xâu bit  $x$

RA : Mã băm  $n$  bit của  $x$

Đầu vào  $x$  được phân thành các khối  $k$  bit ( $k$  là kích thước khoá) và được đệm nếu cần thiết để tạo khối cuối cùng hoàn chỉnh. Biểu thị thông báo đã đệm thành  $t$  khối  $k$  bit:  $x = (x_1, x_2, \dots, x_t)$ . Xác định trước một giá trị ban đầu  $n$  bit (ký hiệu IV).

Đầu ra là  $H_t$  được xác định như sau:

$$\begin{cases} H_0 = IV \\ H_i = E_{x_i}(H_i) \oplus H_{i-1}, \quad 1 \leq i \leq t \end{cases}$$

*Thuật toán băm Miyaguchi - Preneel*

Sơ đồ này tương tự như M-M-O ngoại trừ  $H_{i-1}$  (đầu ra ở giai đoạn trước) được cộng modulo 2 với tín hiệu ra ở giai đoạn hiện thời. Như vậy:

$$\begin{cases} H_0 = IV \\ H_i = E_{g(H_{i-1})}(x_i) \oplus x_i \oplus H_{i-1}, 1 \leq i \leq t \end{cases}$$

*Nhận xét:* Sơ đồ D-M có thể coi là sơ đồ đối ngẫu với sơ đồ M - M - O theo nghĩa  $x_i$  và  $H_{i-1}$  đổi lẫn vai trò.

**4.1.3.2. MDC độ dài kép: MDC-2 và MDC-4**

MDC-2 và MDC-4 là các mã phát hiện sự sửa đổi yêu cầu tương ứng là 2 và 4 phép toán mã hoá khối trên mỗi khối đầu vào hàm băm. Chúng sử dụng 2 hoặc 4 phép lặp của sơ đồ M - M - O để tạo ra hàm băm có độ dài kép. Khi dùng DES chúng sẽ tạo ra mã băm 128 bit. Tuy nhiên trong cấu trúc tổng quát có thể dùng các hệ mật mã khối khác MDC-2 và MDC4 sử dụng các thành phần xác định như sau:

DES được dùng làm mật mã khối  $E_k$  có đầu vào/ ra 64 bit và được tham số hoá bằng khoá  $k$  56 bit.

Hai hàm  $g$  và  $\tilde{g}$  ánh xạ các giá trị 64 bit  $U$  thành các khoá DES 56 bit như sau:

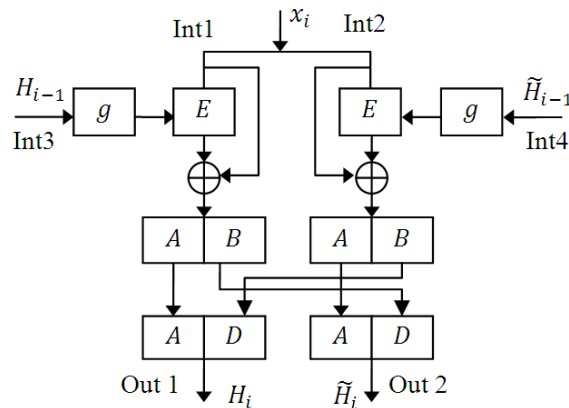
Cho  $U = u_1u_2...u_{64}$ , xoá mọi bit thứ 8 bắt đầu từ  $u_8$  và đặt các bit thứ 2 và thứ 3 về "10" đối với  $g$  và "01" đối với  $\tilde{g}$ .

$$g(U) = u_110u_4u_5u_6u_7u_9u_{10}...u_{63}$$

$$\tilde{g}(U) = u_101u_4u_5u_6u_7u_9u_{10}...u_{63}$$

Đồng thời điều này cũng phải đảm bảo rằng chúng không phải là các khoá DES yếu hoặc nửa yếu vì các khoá loại này có bit thứ hai bằng bit thứ ba. Đồng thời điều này cũng đảm bảo yêu cầu bảo mật là  $g(IV) \neq \tilde{g}(IV)$ .

Thuật toán MDC -2 có thể được mô tả theo sơ đồ sau:



Hình 4.3. Sơ đồ hàm băm MDC – 2

*Thuật toán MDC - 2*

VÀO: Xâu bit  $x$  có độ dài  $r = 64t$  với  $t \geq 2$ .

RA : Mã băm 128 bit của  $x$

Phân  $x$  thành các khối 64 bit  $x_i : (x_1, x_2, \dots, x_t)$ .

Chọn các hằng số không bí mật IV và  $\tilde{IV}$  từ một tập các giá trị khuyến nghị đã được mô tả trước. Tập ngầm định các giá trị cho trước này là (ở dạng HEXA)

$$IV = 0x5252525252525252$$

$$\tilde{IV} = 0x2525252525252525$$

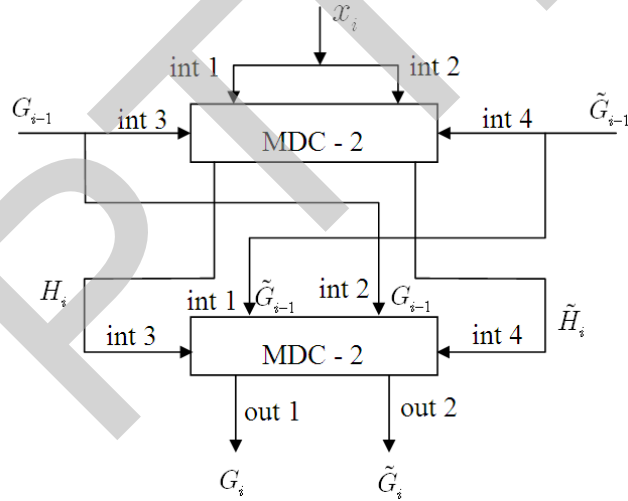
Ký hiệu  $\parallel$  là phép ghép và  $\dots$  là các nửa 32 bit phải và trái của  $C_i$

Đầu ra  $h(x) = H_t \parallel \tilde{H}_t$  được xác định như sau: (với  $1 \leq i \leq t$ )

$$H_0 = IV, k_i = g(H_{i-1}), C_i = E_{k_i}(x_i) \oplus x_i, H_i = C_i^L \parallel \tilde{C}_i^R$$

$$\tilde{H}_0 = \tilde{IV}, \tilde{k}_i = \tilde{g}(\tilde{H}_{i-1}), \tilde{C}_i = E_{\tilde{k}_i}(x_i) \oplus x_i, \tilde{H}_i = \tilde{C}_i^L \parallel C_i^R$$

Thuật toán MDC - 4 có thể được mô tả theo sơ đồ sau:



Hình 4.4. Sơ đồ hàm băm MDC – 4

#### 4.1.4. Các hàm băm có khoá (MAC)

**Định nghĩa 4.6:** Định nghĩa thuật toán mã xác thực thông báo (MAC).

Thuật toán MAC là một họ các hàm  $h_k$  (được tham số hoá bằng một khoá bí mật  $k$ ) có các tính chất sau:

a) *Dễ dàng tính toán:* Với  $h_k$  đã biết và giá trị  $k$  cho trước và một đầu vào  $x$ ,  $h_k(x)$  có thể được tính dễ dàng ( $h_k(x)$  được gọi là giá trị MAC hay MAC).

b) *Nén:*  $h_k$  ánh xạ một đầu vào  $x$  có độ dài bit hữu hạn tùy tới một đầu ra  $h_k(x)$  có độ dài bit  $n$  cố định.

c) *Khó tính toán*: Với các cặp giá trị  $(x_i, h_k(x_i))$  không có khả năng tính một cặp  $(x_i, h_k(x_i))$  với  $x \neq x_i$  (kể cả có khả năng  $h_k(x) = h_k(x_i)$  với một  $i$  nào đó).

Nếu tính chất  $c$  không thoả mãn thì thuật toán được coi là giả mạo MAC.

Các hàm băm có khoá được sử dụng để xác thực thông báo và thường được gọi là *các thuật toán tạo mã xác thực thông báo (MAC)*.

MAC dựa trên các mật mã khối.

*Thuật toán*

VÀO: Dữ liệu  $x$ , mật mã khối  $E$ , khoá MAC bí mật  $k$  của  $E$ .

RA:  $n$  bit MAC trên  $x$  ( $n$  là độ dài khối của  $E$ )

1) Độn và chia khối: Độn thêm các bit vào  $x$  nếu cần. Chia dữ liệu đã độn thành từng khối  $n$  bit:  $x_1, x_2, \dots, x_t$ .

2) Xử lý theo chế độ CBC.

Ký hiệu  $E_k$  là phép mã hoá  $E$  với khoá  $k$ .

Tính khối  $H_i$  như sau:

$$H_1 \leftarrow E_k(x_1)$$

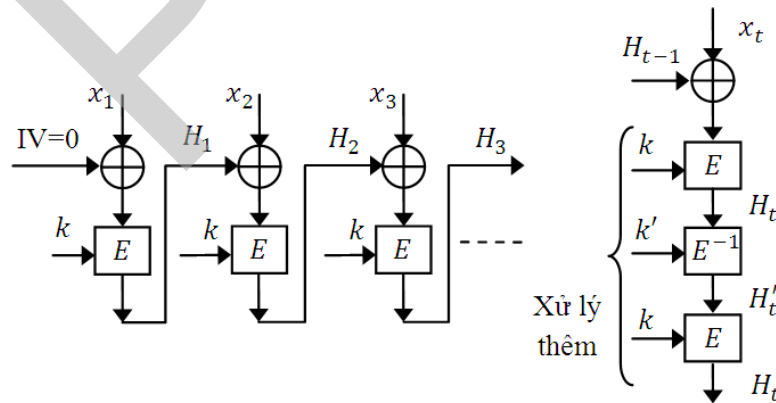
$$H_i \leftarrow K_k(H_{i-1} \oplus x_i) \quad 2 \leq i \leq t$$

3) Xử lý thêm để tăng sức mạnh của MAC

Dùng một khoá bí mật thứ hai  $k' \neq k$ . Tính

$$H'_t \leftarrow E_{k'}^{-1}(H_t), \quad H_t = E_k(H'_t)$$

4) Kết thúc: MAC là khối  $n$  bit  $H_t$



Hình 4.5. Thuật toán MAC dùng CBC

#### 4.1.5. Tính toàn vẹn của dữ liệu và xác thực thông báo

**Định nghĩa 4.7:**

Tính toàn vẹn của dữ liệu là tính chất đảm bảo dữ liệu không bị sửa đổi một cách bất hợp pháp kể từ khi dữ liệu được tạo ra, được phát hoặc được lưu giữ bởi một nguồn được xác định.



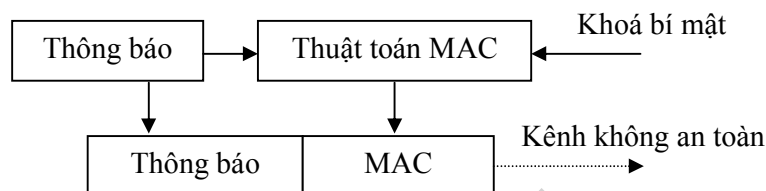
**Định nghĩa 4.8:**

Xác thực tính nguyên bản của dữ liệu là một kiểu xác thực đảm bảo một bên liên lạc được chứng thực là nguồn thực sự tạo ra dữ liệu đó ở một thời điểm nào đó trong quá khứ.

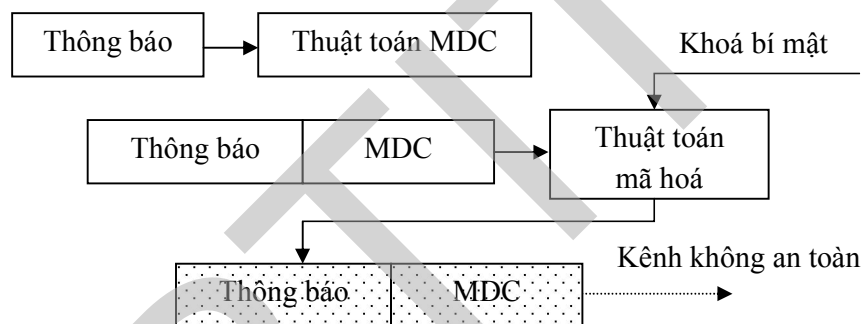
Xác thực thông báo là một thuật ngữ được dùng tương đương với xác thực nguyên gốc của dữ liệu.

Có ba phương pháp cung cấp tính toàn vẹn của dữ liệu bằng cách dùng các hàm băm.

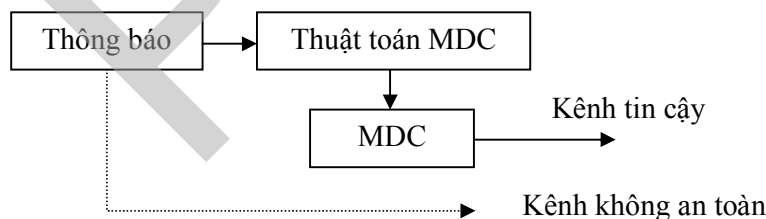
a) Chỉ dùng MAC:



b) Dùng MDC và mã hoá:



c) Sử dụng MDC và kênh tin cậy



**Các phương pháp đảm bảo xác thực tính nguyên vẹn của dữ liệu.**

- Dùng MAC.
- Dùng các sơ đồ chữ ký số.
- Gắn (trước khi mã hoá) một giá trị xác thực bí mật vào văn bản được mã.

## 4.2. CHỮ KÝ SỐ

Trong quá trình làm việc với các văn bản bằng giấy chúng ta sử dụng chữ ký tươi, các yêu cầu cơ bản quan trọng của chữ ký tươi bao gồm:

- + Ngắn gọn (ngắn hơn văn bản cần ký)

+ Là đại diện duy nhất cho người ký.

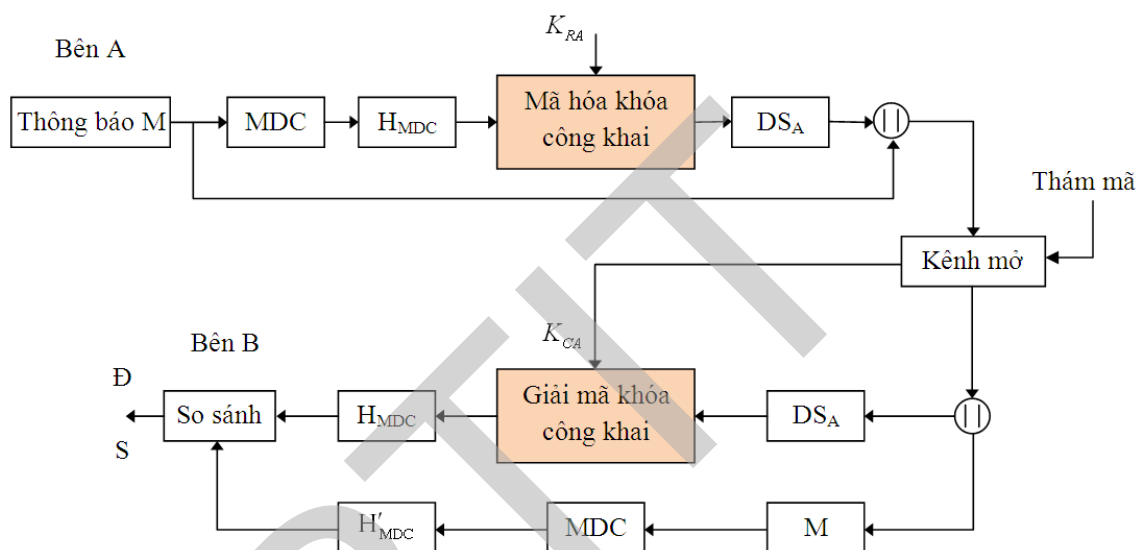
+ Khó bắt chước.

Khi làm việc qua mạng, do không gặp trực tiếp nên phải dùng chữ ký điện tử và nó cũng thỏa mãn các yêu cầu như chữ ký tươi. Tuy nhiên chữ ký số gắn với từng văn bản.

Nếu dùng hàm băm không khóa để làm chữ ký thì ai cũng làm được, còn dùng hàm băm có khóa thì phải trao đổi khóa, có thể giả mạo được.

#### 4.2.1. Sơ đồ chữ ký số

Để thực hiện được chữ ký số, người ta xây dựng trên cơ sở kết hợp mã hoá khoá công khai với hàm băm như được mô tả trong Hình 4.6.



Hình 4.6. Sơ đồ chữ ký số

Trong sơ đồ Hình 4.6, quá trình tạo chữ ký số của A như sau:

Bên A cần tạo chữ ký số cho thông báo M (văn bản cần ký), đầu tiên A băm M bằng hàm băm không khóa (MDC) được mã băm đầu ra  $H_{MDC}$ , mã băm này được mã hóa bằng một hệ mật khóa công khai bằng *khóa bí mật*  $K_{RA}$  của A, kết quả được chữ ký số của A là  $DS_A$ . Cuối cùng chữ ký số này được ghép với thông báo M và gửi đến cho B.

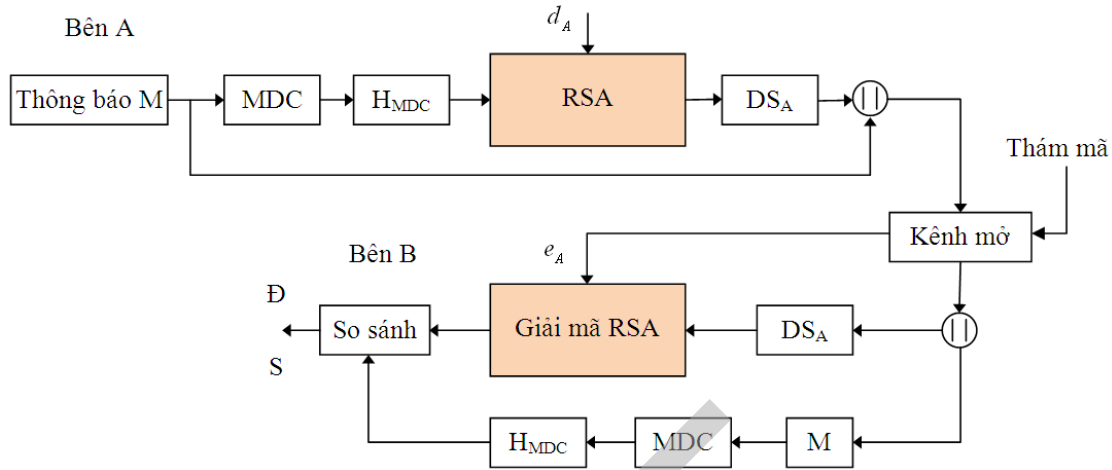
Bên nhận B tiến hành kiểm tra chữ ký số:

- Tách chữ ký số  $DS_A$  khỏi thông báo M.
- Dùng *khóa công khai của A* là  $K_{CA}$  để giải mã  $DS_A$  thu được mã băm  $H_{MDC}$  của thông báo M của A khi chưa truyền qua kênh mở.
- Tiến hành băm thông báo M theo đúng thuật toán của A để có được mã băm  $H'_{MDC}$  của thông báo M đã truyền qua kênh mở.
- So sánh hai mã băm  $H_{MDC}$  và  $H'_{MDC}$  để xác thực thông báo M.

Trong sơ đồ chữ ký số ta nhận thấy rằng việc sử dụng hàm băm là để xác thực nội dung thông báo M, còn hệ mật khóa công khai dùng để xác thực chủ thể nội dung (Bên A)

#### 4.2.2. Sơ đồ chữ ký số RSA.

Có thể coi bài toán xác thực là bài toán "đổi ngẫu" với bài toán bảo mật. Vì vậy, sử dụng ngược thuật toán RSA ta có thể có được một sơ đồ chữ ký số RSA như sau:



Hình 4.7. Sơ đồ chữ ký số RSA không bảo mật

Tạo chữ ký số:

- + Băm thông báo M:  $H_{MDC} = h(M)$
- + Mã hóa mã băm bằng khóa bí mật  $d_A$  để tạo chữ ký số:  $DS_A = [h(M)]^{d_A} \bmod n_A$
- + Ghép chữ ký số với thông báo:  $M \parallel DS_A$

Kiểm tra chữ ký số:

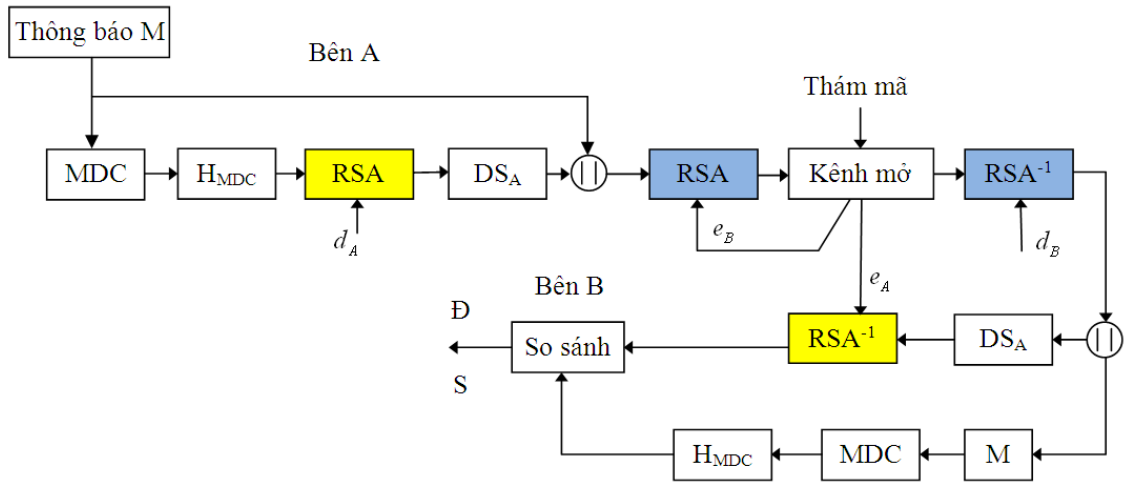
- + Tách  $DS_A$  khỏi thông báo M
- + Giải mã  $DS_A$  bằng khóa công khai của A ( $e_A$ ) để tìm mã băm:

$$H_{MDC} = h(M) = \left( h(M)^{d_A} \right)^{e_A} \bmod n_A$$

- + Băm thông báo nhận được:  $H'_{MDC} = h(M)$
- + So sánh hai mã băm  $H_{MDC}$  và  $H'_{MDC}$  để kiểm tra chữ ký số.

Sơ đồ chữ ký số trong Hình 4.7 là sơ đồ không bảo mật, khi cần bảo mật cả thông báo M người ta sử dụng sơ đồ Hình 4.8.

Trong sơ đồ Hình 4.8, sau khi A tạo được chữ ký số  $DS_A$  và ghép với thông báo M, thì chúng được mã hóa bằng hệ mật RSA với khóa công khai của B ( $e_B$ ) và sau đó gửi đến B. Bên B sẽ tiến hành giải mã bằng khóa bí mật của B ( $d_B$ ) trước khi kiểm tra chữ ký số.



Hình 4.8. Sơ đồ chữ ký số RSA có bảo mật

### 4.3. HỆ MẬT DỰA TRÊN ĐỊNH DANH

#### 4.3.1. Ý tưởng cơ bản

Hệ mật dựa trên định danh do Shamir đề xuất [16] là một hệ mật bất đối xứng trong đó thông tin định danh của thực thể (tên riêng) đóng vai trò khoá công khai của nó. Trung tâm xác thực  $T$  được sử dụng để tính khoá riêng tương ứng của thực thể này. Trong các hệ mật khoá công khai thông thường mỗi người sử dụng có một cặp khoá  $(s, P)$  trong đó  $s$  là khoá bí mật (chỉ có người dùng này biết) còn  $P$  là khoá công khai mà mọi người đều có thể biết. Như vậy, các khoá công khai không cần phải giữ kín mà cần công bố rộng rãi. Tuy nhiên tính công khai này lại trở thành đối tượng cho các tấn công tích cực như việc thay khoá công khai giả vào vị trí khoá công khai thực trong danh bạ. Bởi vậy, ngoài cặp khoá  $(s, P)$  ta cần phải có chuỗi định danh  $I$  và không một dấu hiệu đảm bảo  $G$  để biết rằng  $P$  thực sự là khoá công khai của người dùng  $I$  và không phải là một kẻ giả mạo. Khi ta sử dụng các hệ mật dựa trên định danh, khoá công khai sẽ tương đương với định danh ( $P = I$ ). Còn dấu hiệu đảm bảo sẽ tương đương với khoá bí mật (tức là  $G = s$ ). Hệ thống này có nhiều đặc tính tốt do không phải lưu trữ chứng chỉ để kiểm tra.

Sau khi tính khoá riêng của một người dùng  $T$  sẽ chuyển khoá riêng cho người dùng đó trên một kênh riêng an toàn. khoá riêng này được tính không chỉ từ thông tin định danh của thực thể mà còn phải là một hàm của một thông tin riêng nào đó chỉ có  $T$  mới biết (Khóa riêng của  $T$ ). Đây là điều cần thiết nhằm tránh giả mạo và bắt chước. Điều chủ yếu là chỉ  $T$  mới có khả năng tạo các khoá riêng hợp lệ phù hợp với thông tin định danh.

#### 4.3.2. Sơ đồ trao đổi khoá Okamoto-Tanaka

Phần này mô tả tóm lược sơ đồ trao đổi khoá Okamoto-Tanaka [17] là một hệ thống phân phối khoá dựa trên định danh. Sơ đồ này gồm 3 pha sau:

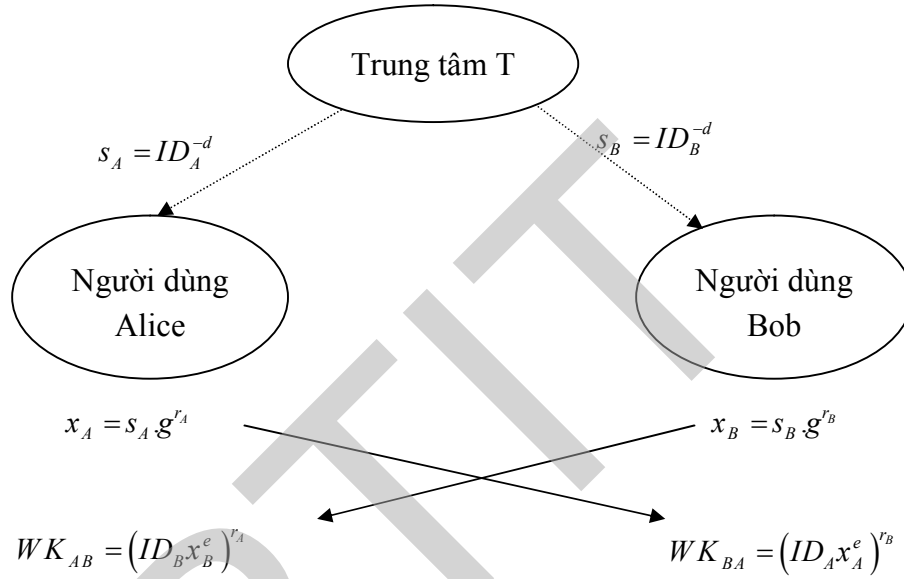
*Pha chuẩn bị.*

Trung tâm xác thực tin cậy chọn 2 số nguyên tố  $p$  và  $q$  và đưa công khai các giá trị  $n, g$  và  $e$ , trong đó  $n = pq$ ,  $g$  là phần tử sinh của cả  $Z_p^*$  và  $Z_q^*$ , còn  $e \in Z_{\lambda(n)}^*$ . Ở đây, hàm Carmichael của  $n$  được xác định như sau:

$$\lambda(n) = BCNN(p-1, q-1)$$

Cho  $d \in Z_{\lambda(n)}^*$  là khoá bí mật của trung tâm thỏa mãn điều kiện:

$$ed \equiv 1 \pmod{\lambda(n)}$$



Hình 4.9. Sơ đồ trao đổi khoá Okamoto-Tanaka

*Pha tham gia của người dùng.*

Cho  $ID_i$  là thông tin định danh của người dùng thứ  $i$  ( $i = A, B, C, \dots$ ). Cho  $s_i$  là khoá bí mật của người dùng  $i$  thỏa mãn:

$$s_i \equiv ID_i^{-d} \pmod{n}$$

Sau đó trung tâm  $T$  sẽ công bố  $(e, n, g, ID_i)$  và phân phát  $s_i$  tới mỗi người dùng  $i$  qua một kênh an toàn (hoặc bằng cách dùng thẻ).

*Pha tạo khoá chung.*

Ta giả sử ở đây rằng hai người dùng Alice và Bob muốn chia sẻ một khoá chung (chẳng hạn để dùng cho một hệ mật khoá bí mật).

Trước tiên Alice tạo một số ngẫu nhiên  $r_A$  và tính:

$$x_A \equiv s_A g^{r_A} \pmod{n}$$

và gửi nó cho Bob.

Tương tự, Bob tạo một số ngẫu nhiên  $r_B$  và tính:

$$x_B \equiv s_B g^{r_B} \pmod{n}$$

và gửi nó cho Alice.

Tiếp theo, Alice tính:

$$WK_{AB} = (ID_B x_B^e)^{r_A} \pmod{n}$$

Tương tự, Bob tính

$$WK_{BA} = (ID_A x_A^e)^{r_B} \pmod{n}$$

$WK_{AB}$  và  $WK_{BA}$  sẽ dùng làm khoá chung vì:

$$\begin{aligned} WK_{AB} &= (ID_B x_B^e)^{r_A} \pmod{n} \\ &= (ID_B (s_B g^{r_B})^e)^{r_A} \\ &= (ID_B (ID_B^{-d})^e g^{r_B e})^{r_A} \\ &= g^{e r_B r_A} \\ &= WK_{BA} \pmod{n} \end{aligned}$$

**Ví dụ 3.19:**

Hai bên liên lạc A và B chọn  $p = 19$ .

Tạo khóa: + A chọn  $(m, n) = (11, 5)$  thỏa mãn  $11 \times 5 = 55 \equiv 1 \pmod{18}$ .

+ B chọn  $(u, v) = (7, 13)$  thỏa mãn  $7 \times 13 = 91 \equiv 1 \pmod{18}$

Giả sử A cần gửi bản tin  $M = 6$  cho B. Quá trình truyền tin thực hiện như sau:

$A_{(11,5)}$	$B_{(7,13)}$
+ A tính $6^{11} \pmod{19} = 17$ và gửi cho B	+ B nhận 17 và tính: $17^7 \pmod{19} = 5$
+ A nhận 5 và tính: $5^5 \pmod{19} = 9$ và gửi 9 cho B	B gửi 5 cho A + B nhận 9 và giải mã ra bản tin M: $9^{13} \pmod{19} = 6 = M$

**Nhận xét:**

- Để tìm bản tin M thám mã phải giải bài toán logarit rời rạc, với trường hợp  $p$  lớn thì đây là hệ mật an toàn.
- Để truyền bản tin M thì phải thực hiện truyền 3 lần, do đó tốc độ mã  $R_{\text{mã}} = 1/3$  (thấp). Vì vậy hệ mật này chỉ thích hợp khi truyền các bản tin ngắn hoặc truyền khóa, phân phối khóa cho hệ mật khóa bí mật.

**3.4.2.3. Hệ mật Elgamal**

**a) Tạo khóa**

Mỗi bên liên lạc A, B tạo cho mình một cặp khóa công khai – bí mật theo các bước sau:

*Bước 1:* Chọn một số nguyên tố  $p$  lớn và  $\alpha$  là một phần tử nguyên thủy ( $\alpha \in \mathbb{Z}_p^*$ )

*Bước 2:* Chọn một số nguyên  $a$  ngẫu nhiên với  $1 < a < p - 1$  và tính

$$\alpha^a \pmod{p}$$

*Bước 3:* + Khóa công khai là bộ 3 số:  $(p, \alpha, \alpha^a)$

+ Khóa bí mật là:  $a$

**b) Mã hóa**

Giả sử B cần gửi bản tin M cho A, B sẽ thực hiện các bước sau:

*Bước 1:* B nhận khóa công khai của A:  $(p, \alpha, \alpha^a)$

*Bước 2:* B chọn số nguyên  $k$  ngẫu nhiên với  $1 < k < p - 1$  và tính:

$$\begin{cases} \gamma = \alpha^k \bmod p \\ \delta = M (\alpha^a)^k \bmod p \end{cases} \quad (3.20)$$

Giả sử bản tin đã được biểu thị dưới dạng một số nguyên  $M$  trong dải  $\{1, \dots, p-1\}$ .  
 Phép tính mũ trong (3.20) được tính bằng thuật toán nhân và bình phương theo modulo.

*Bước 3:* B gửi bản mã  $C = (\gamma, \delta)$  cho A.

Ta thấy bản mã  $C$  được ghép từ  $\gamma$  và  $\delta$  nên nó có độ dài bit bằng 2 lần độ dài của  $M$ , đây là nhược điểm của hệ mật này.

### c) Giải mã

A nhận bản mã  $C$  từ B và tiến hành giải mã theo các bước sau:

*Bước 1:* A sử dụng khóa bí mật  $a$  để tính:

$$\gamma^{p-1-a} \bmod p = \alpha^{-ak} \bmod p.$$

(Chú ý  $\gamma^{p-1-a} = \gamma^{-a} = (\alpha^k)^{-a} = \alpha^{-ak}$ )

*Bước 2:* A khôi phục bản rõ bằng cách tính:

$$\delta \gamma^{p-1-a} \bmod p = M \alpha^{ak} \alpha^{-ak} \bmod p = M$$

### Ví dụ 3.20:

*Tạo khoá.*

Bước 1: A chọn  $p = 17$  và phần tử nguyên thủy  $\alpha = 3$  của  $\mathbb{Z}_{17}^*$ .

Bước 2: A chọn khóa bí mật  $a = 6$  và tính  $\alpha^a \bmod p = 3^6 \bmod 17 = 15$

Bước 3: + Khóa công khai của A là bộ 3 số:  $(p, \alpha, \alpha^a) = (17, 3, 15)$

+ Khóa bí mật của A là:  $a = 6$

*Mã hoá*

Giả sử B cần gửi bản tin  $M = 7$  cho A.

Bước 1: B nhận khóa công khai của A:  $(p, \alpha, \alpha^a) = (17, 3, 15)$ .

Bước 2: B chọn số nguyên  $k = 4$  và tính:

$$\begin{cases} \gamma = \alpha^k \bmod p = 3^4 \bmod 17 = 13 \\ \delta = M (\alpha^a)^k \bmod p = 7 \cdot (15)^4 \bmod 17 = 10 \end{cases}$$

Bước 3: B gửi bản mã  $C = (\gamma, \delta) = (13, 10)$  cho A.



### Giải mã

A nhận bản mã  $C = (\gamma, \delta) = (13, 10)$  và tiến hành giải mã.

Bước 1: A sử dụng khóa bí mật  $a = 6$  để tính:

$$\gamma^{p-1-a} \bmod p = 13^{10} \bmod 17 = 16$$

Bước 2: A khôi phục bản rõ bằng cách tính:

$$\delta \gamma^{p-1-a} \bmod p = 10 \cdot 16 \bmod 17 = 7 = M$$

### Nhận xét:

- Để tìm khóa bí mật  $a$  (từ  $\alpha^a$ ) thám mã phải giải bài toán logarit rời rạc (tính  $a = \log_{\alpha} \alpha^a$ ), với trường hợp  $p$  lớn thì không thể giải được, hệ mật là an toàn.
- Hiệu quả truyền tin thấp, vì tốc độ mã chỉ đạt  $R_{\text{mã}} = 1/2$

## 3.5. BÀI TOÁN PHÂN TÍCH THỪA SỐ VÀ HỆ MẬT RSA

### 3.5.1. Bài toán phân tích thừa số

Theo định lý 1.1, với mỗi số nguyên  $n \geq 2$  ta luôn phân tích  $n$  được dưới dạng tích của lũy thừa của các số nguyên tố và phân tích này là duy nhất như sau:

$$n = p_1^{e_1} p_2^{e_2} \dots p_k^{e_k} \quad (3.21)$$

Trong đó  $p_i$  là các số nguyên tố khác nhau và  $e_i$  là các số nguyên dương.

Nếu  $n$  là tích của hai số nguyên tố  $n = pq$ , trong đó  $p, q$  là hai số nguyên tố lớn có độ lớn xấp xỉ nhau, thì việc phân tích  $n$  thành tích của  $p$  và  $q$  là bài toán khó. Hay là, nếu ta biết  $p, q$  thì tính  $n$  rất đơn giản, nhưng ngược lại biết  $n$  thì rất khó tìm  $p$  và  $q$ .

### 3.5.2. Hệ mật RSA (Rivest – Shamir – Adleman)

#### 3.5.2.1. Tạo khóa

Mỗi bên liên lạc A và B tự tạo cho mình một cặp khóa công khai – bí mật theo các bước sau đây:

*Bước 1:* Chọn 2 số nguyên tố lớn  $p$  và  $q$  có độ lớn tương đương.

*Bước 2:* Tính  $n = pq$  và  $\varphi(n) = \varphi(pq) = (p-1)(q-1)$

*Bước 3:* Chọn một số nguyên  $e$  ngẫu nhiên với  $1 < e < \varphi(n)$  thỏa mãn:  $(e, \varphi(n)) = 1$

*Bước 4:* Tính số nguyên duy nhất  $d$ ,  $1 < d < \varphi(n)$  thỏa mãn  $ed \equiv 1 \bmod \varphi(n)$ .

(Có thể sử dụng thuật toán Euclide mở rộng để tìm  $d$ )

*Bước 5:* + Khóa công khai công khai của A là cặp số:  $(n, e)$

+ Khóa bí mật của A là số  $d$ .

Chú ý: Các số nguyên  $e$  và  $d$  trong thuật toán tạo khoá RSA được gọi là số mũ mã hoá và số mũ giải mã còn số  $n$  được gọi là modulus. Ta thấy  $e$  và  $d$  là hai số nghịch đảo nên vai trò của chúng là giống nhau, tức là nếu khóa công khai là  $(n, e)$  thì khóa bí mật là  $d$ , ngược lại nếu khóa công khai là  $(n, d)$  thì khóa bí mật sẽ là  $e$ .

### 3.5.2.2. Mã hóa

Giả sử B cần gửi bản tin  $M$  cho A (với  $M \leq n$ ), B thực hiện các bước sau:

*Bước 1:* B nhận khóa công khai của A  $(n, e)$ .

*Bước 2:* B tính (mã hóa):

$$C \equiv M^e \pmod{n} \quad (3.22)$$

*Bước 3:* B gửi bản mã  $C$  cho A.

### 3.5.2.3. Giải mã

A nhận bản mã  $C$  từ B và tiến hành giải mã:

$$C^d \pmod{n} = M^{ed} \pmod{n} = M \quad (3.23)$$

**Nhận xét:**

- Thăm mã có  $e$  muốn biết  $d$  thì phải biết  $\varphi(n)$ , mà muốn biết  $\varphi(n)$  thì thăm mã phải thực hiện bài toán phân tích thừa số  $n = pq$  để biết  $p$  và  $q$ . Khi  $p$  và  $q$  là các số nguyên tố lớn thì thăm mã không thể giải được.
- Hiệu quả truyền tin của RSA cao ( $R_{\text{mã}} \approx 1$ ).
- Thuật toán RSA có tính bền vững, vì các lý do trên mà RSA là một trong các hệ mật được sử dụng rộng rãi trong vòng hơn 30 năm qua.

### Chú ý (Số mũ vận năng)

Số  $\lambda = BCNN(p-1, q-1)$  đôi khi được gọi là số mũ vận năng của  $n$ ,  $\lambda$  có thể được dùng thay cho  $\varphi(p-1)(q-1)$  khi tạo khoá RSA. Cần chú ý rằng  $\lambda$  là ước thực sự của  $\varphi$ . Sử dụng  $\lambda$  có thể thu được số mũ giải mã  $d$  nhỏ hơn (làm cho giải mã nhanh hơn). Tuy nhiên, nếu  $p$  và  $q$  được chọn ngẫu nhiên thì  $UCNN(p-1, q-1)$  sẽ khá nhỏ và bởi vậy  $\Phi$  và  $\lambda$  sẽ là các số có kích thước xấp xỉ.

### 3.5.2.4. Một số ví dụ

**Ví dụ 3.21:**

*Xây dựng các tham số cho hệ mật RSA (Tạo khoá)*

*Bước 1:* A chọn  $p = 7$  và  $q = 17$ .

*Bước 2:* A tính:

$$+ n = pq = 7 \times 17 = 119$$

$$+ \varphi(n) = (p-1)(q-1) = 6 \times 16 = 96$$

Bước 3: A chọn  $e = 5$  thỏa mãn  $(5, 96) = 1$  vì  $(5, 6) = 1$ ;  $(5, 16) = 1$

Bước 4: Tính  $d$  thỏa mãn:

$$ed \equiv 1 \pmod{\varphi(n)} \Rightarrow 5d \equiv 1 \pmod{96} \quad (3.24)$$

Với các giá trị  $p, q$  còn nhỏ ta có thể giải phương trình đồng dư để tìm  $d$ . Từ biểu thức (3.24) ta có phương trình sau:

$$5d = 1 + k96$$

Với  $k = 4$  tìm được:  $d = \frac{1 + 4 \cdot 96}{5} = 77$

Bước 5: + Khóa công khai của A là:  $(119, 5)$

+ Khóa bí mật của A là: 77

Cũng có thể chọn khóa công khai là  $(119, 77)$  và khóa bí mật là 5.

### Ví dụ 3.22:

Xây dựng một hệ mật RSA để gửi bản tin  $\mathbf{M} = \text{CRYPTOGRAPH}$  từ B đến A.

#### Tạo khóa:

Bước 1: A chọn  $p = 43, q = 59$ .

Bước 2: A tính  $n = pq = 43 \times 59 = 2537$

$$\text{Và } \varphi(n) = (p-1)(q-1) = 42 \times 58 = 2436$$

Bước 3: A chọn  $e = 1357$

Bước 4: A tính ra  $d = e^{-1} = 1357^{-1} = 517$

Bước 5: + Khóa công khai của A:  $(n, e) = (2537, 1357)$

+ Khóa bí mật của A:  $d = 517$

#### Mã hóa:

Bên B biến đổi bản tin  $\mathbf{M} = \text{CRYPTOGRAPH}$  như sau:

Biểu diễn dạng Hexa của M:

M =	C	R	Y	P	T	O	G	R	A	P	H
M <sub>HEX</sub> =	43	52	59	50	54	4F	47	52	41	50	48

Các giá trị hexa của các ký tự lấy từ bảng mã ASCII, ví dụ chữ cái C có mã hexa tương ứng là 43.

Sau đó chuyển đổi các số hexa thành số nhị phân, ví dụ số 43 gồm 2 số hexa là 4 và 3, khi chuyển sang nhị phân ta chuyển từng số một: số  $4_{Hex} \leftrightarrow 0100_{Bin}$ , số  $3_{Hex} \leftrightarrow 0011_{Bin}$  và như thế số 43 sẽ chuyển thành 8 bit nhị phân tương ứng là:  $43_{Hex} \leftrightarrow 0100.0011_{Bin}$ . Tiến hành chuyển đổi toàn bộ 11 chữ cái của M ta được biểu diễn nhị phân tương ứng của M như sau:

$$M_{Bin} = \underbrace{01000011}_{43} . \underbrace{01010010}_{52} . \underbrace{01011001}_{59} \dots \underbrace{01010000}_{50} . \underbrace{01001000}_{48} \quad (8 \times 11 = 88bit)$$

Có một chú ý là khi mã hóa thì giá trị  $M < n$ , với  $n$  tính được ở trên ta thấy:

$$n = 2537 > 2048 = 2^{11} \text{ như vậy mỗi lần chỉ mã hóa được tối đa 11 bit thông tin.}$$

Chuỗi bit của bản tin rõ gồm 88 bit, vậy B chia thành 8 khối, mỗi khối 11 bit như sau:

$$M_{Bin} = \underbrace{01000011010}_{M_1} . \underbrace{10010010110}_{M_2} \dots \underbrace{00001001000}_{M_8}$$

Tiếp theo B sẽ chuyển 8 khối 11 bit ( $M_1 \div M_8$ ) thành giá trị thập phân tương ứng:

$$M_{Dec} = \underbrace{538}_{M_1} . \underbrace{1174}_{M_2} . \underbrace{672}_{M_3} . \underbrace{1348}_{M_4} . \underbrace{1955}_{M_5} . \underbrace{1353}_{M_6} . \underbrace{42}_{M_7} . \underbrace{72}_{M_8}$$

B tiến hành mã hóa từng khối  $M_i$  để có các khối bản mã  $C_i$  tương ứng như sau:

$$\begin{cases} C_1 = M_1^e \bmod n = 538^{1357} \bmod 2537 = 905 \\ C_2 = M_2^e \bmod n = 1174^{1357} \bmod 2537 = 1307 \\ C_3 = M_3^e \bmod n = 672^{1357} \bmod 2537 = 1040 \\ C_4 = M_4^e \bmod n = 1348^{1357} \bmod 2537 = 1987 \\ C_5 = M_5^e \bmod n = 1955^{1357} \bmod 2537 = 750 \\ C_6 = M_6^e \bmod n = 1353^{1357} \bmod 2537 = 1567 \\ C_7 = M_7^e \bmod n = 42^{1357} \bmod 2537 = 1590 \\ C_8 = M_8^e \bmod n = 72^{1357} \bmod 2537 = 1093 \end{cases}$$

Cuối cùng B sẽ biến đổi các bản mã  $C_i$  thành các bit nhị phân và ghép lại thành chuỗi 88 bit và truyền chuỗi bit này đến A.

### **Giải mã:**

A nhận chuỗi các bit bản mã do B gửi lại tách thành các khối 11 bit và biến đổi thành giá trị thập phân tương ứng để có các khối bản mã  $C_i$ . Tiếp theo A sẽ giải mã cho từng khối bản mã  $C_i$  để có các khối  $M_i$  theo nguyên tắc giải mã:

$$M_i = (C_i)^d \bmod n$$

$$\text{Ví dụ: } M_1 = (C_1)^d \bmod n = 905^{517} \bmod 2537 = 538$$

Sau khi đã giải mã hết 8 khối  $C_i$ , A có 8 khối  $M_i$  và A sẽ thực hiện theo quy trình ngược lại bên B để có bản tin rõ cuối cùng là **M = CRYPTOGRAPH**

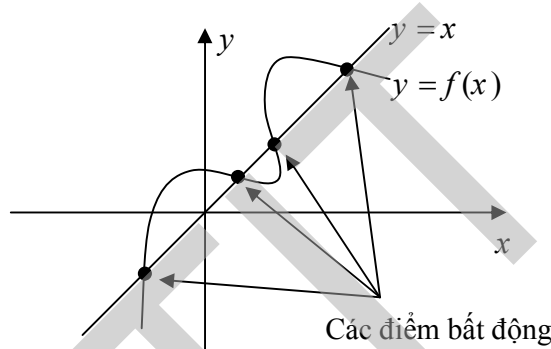
Việc mã hóa và giải mã của hệ mật RSA đều dựa vào hàm mũ, trong trường hợp các số mũ lớn, để tính toán được người ta sử dụng thuật toán nhân và bình phương để tính hàm lũy thừa modulo. Thuật toán này đã được trình bày trong mục 3.1.4.2.

### 3.5.3. Vấn đề điểm bất động trong RSA

Xét một ánh xạ  $y = f(x)$  (phép mã hóa trong hệ mật), tồn tại các điểm  $x$  mà sau khi mã hóa bản tin không thay đổi, tức là:

$$\exists x : x = f(x)$$

Các điểm này gọi là các điểm bất động.



Hình 3.6. Các điểm bất động của  $y = f(x)$

Hệ mật RSA cũng tồn tại các điểm bất động.

#### Ví dụ 3.23:

Xét hệ mật RSA với các tham số:  $(n, e) = (35, 17)$  với  $p = 5, q = 7$ . Giả sử bản tin  $M = 8$ , ta thấy:

$$C = M^e \bmod n = 8^{17} \bmod 35 = 8$$

Tức là  $M = C$  như vậy là không che giấu được thông tin.

Số các điểm bất động của một hệ mật RSA được xác định theo định lý sau đây:

#### Định lý 3.19:

Với hệ mật RSA có tham số khóa công khai là  $(n, e)$  với  $n = pq$  thì số bản tin không thể che giấu được (số điểm bất động) tính như sau:

$$N = \left[ 1 + \text{UCLN}(e-1, p-1) \right] \left[ 1 + \text{UCLN}(e-1, q-1) \right] \quad (3.25)$$

#### Ví dụ 3.24:

+ Với hệ mật RSA có  $(n, e) = (35, 3)$  với  $p = 5, q = 7$  thì số bản tin không che giấu được là:

$$N = [1 + UCLN(2, 4)][1 + UCLN(2, 6)] = 9$$

Các điểm bất động là:  $M = \{0, 1, 6, 14, 15, 20, 21, 29, 34\}$

Như vậy xác suất gặp phải các bản tin không che giấu được khoảng  $1/4$ .

+ Vẫn hệ mật RSA như trên nhưng ta thay đổi số  $e : (n, e) = (35, 17)$  thì lúc này số bản tin không che giấu được là:

$$N = [1 + UCLN(16, 4)][1 + UCLN(16, 6)] = 15$$

Các điểm bất động là:  $M = \{0, 1, 6, 7, 8, 13, 14, 15, 20, 21, 22, 27, 28, 29, 34\}$

Và xác suất gặp phải các bản tin không che giấu được khoảng gần  $1/2$ .

### 3.5.4. Hệ mật Rabin

#### 3.5.4.1. Tạo khoá

Mỗi bên liên lạc tạo cho mình một khoá công khai và một khoá bí mật tương ứng theo các bước sau:

*Bước 1:* Tạo 2 số nguyên tố lớn, ngẫu nhiên và phân biệt  $p$  và  $q$  có kích thước xấp xỉ nhau.

*Bước 2:* Tính  $n = pq$ .

*Bước 3:* + Khoá công khai là  $n$ .  
+ Khoá bí mật là các cặp số  $(p, q)$ .

#### 3.5.4.2. Mã hóa

B cần gửi bản tin  $M$  cho A, phải thực hiện các bước sau:

*Bước 1:* Nhận khoá công khai của A:  $n$

*Bước 2:* Tính  $C = M^2 \bmod n$ .

*Bước 3:* B gửi bản mã  $C$  cho A.

#### 3.5.4.3. Giải mã:

Để khôi phục bản rõ  $M$  từ  $C$ , A phải thực hiện các bước sau:

*Bước 1:* Tìm 4 căn bậc hai của  $C \bmod n$  là  $M_1, M_2, M_3, M_4$ .

*Bước 2:* Thông báo cho người gửi là một trong 4 giá trị  $M_1, M_2, M_3, M_4$ . Bằng một cách nào đó A sẽ quyết định  $M$  là giá trị nào.

*Chú ý:*

Tìm các căn bậc 2 của  $C \bmod n$ ,  $n = pq$  khi  $p \equiv q \equiv 3 \bmod 4$ . Trong trường hợp này, việc tìm 4 căn bậc 2 của  $C \bmod n$  được thực hiện khá đơn giản như sau:

Sử dụng thuật toán Euclide mở rộng để tìm các số nguyên  $a$  và  $b$  thoả mãn:  $ap + bq = 1$ . Chú ý rằng  $a$  và  $b$  có thể được tính trong giai đoạn tạo khoá.

Tính  $r = C^{(p+1)/4} \bmod p$ .

Tính  $s = C^{(q+1)/4} \bmod q$ .

Tính  $x = (aps + bqr) \bmod n$ .

Tính  $y = (aps - bqr) \bmod n$ .

Bốn giá trị căn bậc 2 của  $C \bmod n$  là:  $x, -x \bmod n, y$  và  $-y \bmod n$

### Ví dụ 3.25:

\* Tạo khoá.

Bước 1: Bên A chọn hai số nguyên tố  $p = 277, q = 331$ .

Bước 2: A tính  $n = pq = 91687$

Bước 3: + Khoá công khai của A là  $n = 91687$ .  
+ Khoá bí mật của A là cặp số  $(p = 277, q = 331)$ .

\* Mã hoá

Giả sử rằng 6 bit cuối cùng của bản tin gốc được lặp lại trước khi thực hiện mã hoá. Việc thêm vào độ thừa này nhằm giúp cho bên giải mã nhận biết được bản mã đúng.

Để mã hoá bản tin 10 bit  $\bar{M} = 1001111001$ , B sẽ lặp lại 6 bit cuối cùng của  $\bar{M}$  để có được bản tin 16 bit sau:  $M = 1001111001111001$ , biểu diễn thập phân tương ứng là  $M_{dec} = 40596$ .

Bước 1: B nhận khóa công khai của A:  $n = 91687$

Bước 2: B tính  $C = M^2 \bmod n = 40596^2 \bmod 91687 = 62111$

Bước 3: B gửi  $C$  cho A.

\* Giải mã

Để giải mã bản mã  $C$ , A thực hiện:

Bước 1: Tính bốn giá trị căn bậc 2 của  $C \bmod n$

$$M_1 = 69654, \quad M_2 = 22033, \quad M_3 = 40596, \quad M_4 = 51118$$

Biểu diễn nhị phân tương ứng của các số trên là:

$$M_1 = 10001000000010110, \quad M_2 = 101011000010001$$

$$M_3 = 1001111001111001, \quad M_4 = 1100011110101110$$

Vì chỉ có  $M_3$  mới có độ thừa cần thiết nên A sẽ giải mã  $C$  bằng  $M_3$  và khôi phục lại bản tin gốc là  $\bar{M} = 1001111001$ .

*Đánh giá hiệu quả*

Thuật toán mã hoá Rabin là một thuật toán cực nhanh vì nó chỉ cần thực hiện một phép bình phương modulo đơn giản. Trong khi đó, chẳng hạn với thuật toán RSA có  $e = 3$  phải cần tới một phép nhân modulo và một phép bình phương modulo. Thuật toán giải mã Rabin có chậm hơn thuật toán mã hoá, tuy nhiên về mặt tốc độ nó cũng tương đương với thuật toán giải mã RSA.

### 3.6. BÀI TOÁN XẾP BA LÔ VÀ HỆ MẬT MERKLE – HELLMAN

#### 3.6.1. Bài toán xếp ba lô

##### 3.6.1.1. Dãy siêu tăng

**Định nghĩa 3.25:** Dãy siêu tăng

Dãy các số nguyên dương  $(a_1, a_2, \dots, a_n)$  được gọi là dãy siêu tăng nếu  $a_i \geq \sum_{j=1}^{i-1} a_j$  với  $\forall i, 2 \leq i \leq n$

Ví dụ: dãy số  $(1, 2, 4, 8, 16, 32, 64, 128)$

##### 3.6.1.2. Bài toán xếp ba lô

Cho một tập hợp các gói có các trọng lượng khác nhau, liệu có thể xếp một số gói này vào ba lô để ba lô có một trọng lượng cho trước hay không. Về mặt hình thức ta có thể phát biểu bài toán trên như sau:

Cho tập các giá trị  $M_1, M_2, \dots, M_n$  và một tổng  $S$ . Hãy tính các giá trị  $b_i$  để:

$$S = b_1 M_1 + b_2 M_2 + \dots + b_n M_n$$

với  $b_i \in \{0, 1\}$

$b_i = 1$ : Có nghĩa là gói  $M_i$  được xếp vào ba lô.

$b_i = 0$ : Có nghĩa là gói  $M_i$  không được xếp vào ba lô.

Bài toán có  $2^n$  phương án ( $2^n$  vectơ nhị phân  $b$ ), bài toán với  $n$  lớn (vài trăm) thì không thể giải ở thời gian thực, hay đây là bài toán khó.



### 3.6.1.3. Giải bài toán xếp ba lô trong trường hợp dãy siêu tăng.

Trong trường hợp  $M = \{M_1, M_2, \dots, M_n\}$  là một dãy siêu tăng thì việc tìm  $b = (b_1, b_2, \dots, b_n)$  tương đương như bài toán tìm biểu diễn nhị phân của một số  $S$ . Biểu diễn này sẽ tìm được sau tối đa là  $n$  bước (độ phức tạp tương đương  $n$ ).

*Thuật toán giải:*

VÀO: Dãy siêu tăng  $M = \{M_1, M_2, \dots, M_n\}$  và một số nguyên  $S$  là tổng của một tập con trong  $M$

RA :  $(b_1, b_2, \dots, b_n)$  trong đó  $b_i \in \{0, 1\}$  sao cho:  $\sum_{i=1}^n b_i M_i = S$  (Phương án sắp xếp)

Bước 1:  $i \leftarrow n$

Bước 2: Chừng nào  $i \geq 1$  hãy thực hiện:

Nếu  $S \geq M_i$  thì

$b_i \leftarrow 1$

và  $S \leftarrow S - M_i$

ngược lại:  $b_i \leftarrow 0$

$i \leftarrow i - 1$

Bước 3: Return  $(b)$

**Ví dụ 3.26:**

Cho  $M = (1, 2, 4, 8, 16, 32, 64)_{n=7}$ ;  $S = 57$ , hãy tìm phương án sắp xếp  $b$ :

Bước 1:  $i = 7$

Bước 2:  $i = 7$ :  $57 < 64 \rightarrow b_7 = 0, i = 6$

$i = 6$ :  $57 > 32 \rightarrow b_6 = 1, S = 57 - 32 = 25; i = 5$

$i = 5$ :  $25 > 16 \rightarrow b_5 = 1, S = 25 - 16 = 9; i = 4$

$i = 4$ :  $9 > 8 \rightarrow b_4 = 1, S = 9 - 8 = 1; i = 3$

$i = 3$ :  $1 < 4 \rightarrow b_3 = 0; i = 2$

$i = 2$ :  $1 < 2 \rightarrow b_2 = 0; i = 1$

$i = 1$ :  $1 = 1 \rightarrow b_1 = 1, S = 1 - 1 = 0; i = 0$

Bước 3: ta có phương án sắp xếp:  $b = (b_1, b_2, \dots, b_7) = (1, 0, 0, 1, 1, 1, 0)$

### 3.6.2. Hệ mật Merkle - Hellman

Hệ này và các hệ liên quan dựa trên tính khó giải của bài toán tổng các tập con (bài toán này là bài toán NP đầy đủ - là một lớp khá lớn các bài toán không có giải thuật được biết trong thời gian đa thức). Tuy nhiên tất cả các hệ mật xếp ba lô khác nhau đều đã bị chứng tỏ là không mật (ngoại trừ hệ mật Chor-Rivest).

#### 3.6.2.1. Thuật toán tạo khoá

Mỗi bên liên lạc tạo cho mình một cặp khóa khoá công khai - khóa bí mật theo các bước sau đây:

*Bước 1:* Chọn một dãy siêu tăng  $\{M_1, M_2, \dots, M_n\}$  và một giá trị modulo  $M$  thỏa mãn:

$$M \geq \sum_{i=1}^n M_i \quad (M \text{ là phần tử thứ } n+1 \text{ của dãy siêu tăng})$$

*Bước 2:* Chọn một số nguyên ngẫu nhiên  $W$  ( $1 < W < M$ ) sao cho  $(W, M) = 1$ . (Để nhất là chọn  $M$  là số nguyên tố)

*Bước 3:* Tính  $a_i \equiv WM_i \pmod{M}$ , với  $i = 1, 2, \dots, n$

*Bước 4:* + Khóa công khai:  $A = (a_1, a_2, \dots, a_n)$

+ Khóa bí mật:  $(M, W, \{M_1, M_2, \dots, M_n\})$

#### 3.6.2.2. Mã hóa:

B cần gửi bản tin  $m = (m_1, m_2, \dots, m_n)$ ,  $m_i \in \{0, 1\}$  cho A. B thực hiện các bước sau:

*Bước 1:* B nhận khóa công khai của A:  $(M, W, \{M_1, M_2, \dots, M_n\})$

*Bước 2:* B tính  $C = \sum_{i=1}^n m_i a_i$

*Bước 3:* B gửi bản mã  $C$  cho A.

#### 3.6.2.3. Giải mã

A nhận bản mã  $C$  và tiến hành giải mã theo các bước sau:

*Bước 1:* A tính  $d = W^{-1}C \pmod{M} = \sum_{i=1}^n m_i M_i$

(Chú ý:  $d \equiv W^{-1}C \equiv W^{-1} \sum_{i=1}^n m_i a_i = W^{-1} \sum_{i=1}^n m_i W M_i = \sum_{i=1}^n m_i M_i$ )

Do  $(W, M) = 1$  nên sẽ  $\exists W^{-1}$  việc tìm  $W^{-1}$  có thể theo thuật toán Euclid mở rộng.

*Bước 2:* Sử dụng thuật giải bài toán xếp ba lô trong trường hợp dãy siêu tăng để tính

$$d = \sum_{i=1}^n m_i M_i$$

Và tìm lại được  $m = (m_1, m_2, \dots, m_n)$

**Ví dụ 3.27:**

*Bên A tạo khóa:*

*Bước 1:* A chọn ngẫu nhiên các giá trị  $M_i$  trong các dải số sau để tạo dãy siêu tăng:

$$M_1 \in [1, 16], M_2 \in [17, 32], M_3 \in [33, 64], M_4 \in [113, 128]$$

A có dãy siêu tăng:  $M = (5, 23, 57, 119)$

do  $\sum_{i=1}^4 M_i = 204$  nên A chọn  $M = 257$  (nguyên tố).

*Bước 2:* Chọn  $W = 113$ ,  $\exists W^{-1}$  vì  $(113, 257) = 1$ , và tính được

$$W^{-1} = W^{\Phi(M)-1} \bmod M = 113^{255} \bmod 257 = 116$$

*Bước 3:* Tính  $a_i \equiv W M_i \bmod M$ , với  $i = 1, 2, 3, 4$

$$a_1 = 113 \cdot 5 \bmod 257 = 51$$

$$a_2 = 113 \cdot 23 \bmod 257 = 29$$

$$a_3 = 113 \cdot 57 \bmod 257 = 16$$

$$a_4 = 113 \cdot 119 \bmod 257 = 83$$

*Bước 4:* + Khóa công khai:  $A = (a_1, a_2, a_3, a_4) = (51, 29, 16, 83)$

+ Khóa bí mật:  $(M, W, \{M_1, M_2, \dots, M_n\}) = (257, 113, \{5, 23, 57, 119\})$

*Mã hóa:*

B cần gửi bản tin  $m = (1, 1, 0, 1)$  cho A.

*Bước 1:* B nhận khóa công khai của A:  $(257, 113, \{5, 23, 57, 119\})$

*Bước 2:* B tính  $C = \sum_{i=1}^n m_i a_i = 1 \times 51 + 1 \times 29 + 0 \times 16 + 1 \times 83 = 163$

*Bước 3:* B gửi bản mã  $C = 163$  cho A.

*Giải mã:*

A nhận bản tin  $C = 163$  và giải mã:

*Bước 1:* A tính  $d = W^{-1} C \bmod M = 116 \cdot 163 \bmod 257 = 147$

*Bước 2:* Sử dụng thuật giải bài toán xếp ba lô trong trường hợp dãy siêu tăng để tính

$$d = \sum_{i=1}^n m_i M_i = 147$$

$$i = 4 \quad 147 > 119 \rightarrow m_4 = 1, S = 147 - 119 = 28, i = 3$$

$$i = 3: \quad 28 < 57 \rightarrow m_3 = 0; i = 2$$

$$i = 2: \quad 28 > 23 \rightarrow m_2 = 1; S = 28 - 23 = 5; i = 1$$

$$i = 1: \quad 5 = 5 \rightarrow m_1 = 1, S = 1 - 1 = 0; i = 0$$

Và bản tin sau giải mã là:  $m = (m_1, m_2, m_3, m_4) = (1, 1, 0, 1)$

### 3.6.3. Hệ mật Chor-Rivest (CR)

Hệ mật CR là hệ mật khoá công khai xếp ba lô duy nhất hiện nay không sử dụng phép nhân modulo để ngụy trang bài toán tổng tập con.

#### 3.6.3.1. Tạo khoá

Mỗi bên liên lạc tạo một khoá công khai và một khoá riêng tương ứng. A thực hiện các bước sau:

- (1) Chọn một trường hữu hạn  $F_q$  có đặc số  $q$ , trong đó  $q = p^h$ ,  $p \geq h$  và đối với nó bài toán logarit rời rạc là khó giải.
- (2) Chọn một đa thức bất khả quy định chuẩn ngẫu nhiên  $f(x)$  bậc  $h$  trên  $Z_p$ . Các phần tử của  $F_q$  sẽ được biểu diễn bằng các đa thức trong  $Z_p[x]$  có bậc nhỏ hơn  $h$  với phép nhân được thực hiện theo mod  $f(x)$ .
- (3) Chọn một phần tử nguyên thủy ngẫu nhiên  $g(x)$  của  $F_q$ .
- (4) Với mỗi phần tử của trường cơ sở  $i \in Z_p$ , tìm logarit rời rạc  $a_i = \log_{g(x)}(x + i)$  của các phần tử  $x + i$  theo cơ sở  $g(x)$ .
- (5) Chọn một phép hoán vị ngẫu nhiên  $\pi$  trên các số nguyên  $\{1, 2, \dots, p-1\}$ .
- (6) Chọn một số nguyên ngẫu nhiên  $d$ ,  $0 \leq d \leq p^h - 2$
- (7) Tính  $C_i = (a_{\pi(i)} + d) \bmod (p^h - 1)$ ,  $0 \leq i \leq p-1$ .
- (8) Khoá công khai của A là  $((C_0, C_1, \dots, C_{p-1}), p, h)$   
 Khoá riêng của A là  $(f(x), g(x), \pi, d)$ .

#### 3.6.3.2. Mã hoá

B cần mã hoá thông báo  $m$  để gửi cho A. B thực hiện các bước sau:

- (1) Nhập khoá công khai của A  $((C_0, C_1, \dots, C_{p-1}), p, h)$

(2) Biểu diễn thông báo như một xâu bit có độ dài  $\left\lceil \lg \binom{p}{h} \right\rceil$  trong đó:

$$\binom{p}{h} = \frac{p!}{h!(p-h)!}.$$

(3) Xem  $m$  như là biểu diễn nhị phân của một số nguyên. Biến đổi số nguyên này thành một vectơ nhị phân  $M = (M_0, M_1, \dots, M_{p-1})$  có độ dài  $p$  và có đúng  $h$  con 1 như sau:

(a) Đặt  $l \leftarrow h$

(b) For  $i$  from 1 to  $n$  do:

Nếu  $m \geq \binom{p-i}{l}$  thì đặt  $M_{i-1} \leftarrow 1, m \leftarrow m - \binom{p-i}{l}, l \leftarrow l - 1$ .

Nếu không thì đặt

$$M_{i-1} \leftarrow 0 \left( \begin{array}{ll} CY : \binom{n}{0} = 1 & n \geq 0 \\ \binom{0}{l} = 0 & l \geq 1 \end{array} \right)$$

(4) Tính  $C = \sum_{i=1}^{p-1} M_i c_i \bmod (p^h - 1)$ .

(5) Gửi bản mã  $C$  cho A.

### 3.6.3.3. Giải mã

Để khôi phục bản mã rõ  $m$  từ  $C$ , A phải thực hiện các bước sau:

(1) Tính  $r = (c - hd) \bmod (p^h - 1)$

(2) Tính  $u(x) = g^r(x) \bmod f(x)$

(3) Tính  $s(x) = u(x) + f(x)$  là một đa thức định chuẩn h trên  $Z_p$ .

(4) Phân tích  $s(x)$  thành các nhân tử bậc nhất trên  $Z_p$ .

$$s(x) = \prod_{j=1}^h (x + t_j) \text{ trong đó } t_j \in Z_p$$

(5) Các thành phần có giá trị 1 của vectơ M có các chỉ số là  $\pi^{-1}(t_j)$  với  $1 \leq j \leq h$ .

Các thành phần còn lại bằng 0

(6) Thông báo  $m$  được khôi phục lại từ M như sau

a) Đặt  $m \leftarrow 0, l \leftarrow h$

b) For  $i$  from 1 to  $p$  do:

Nếu  $M_{i-1} = 1$  thì đặt  $m \leftarrow m + \binom{p-i}{l}, l \leftarrow l-1$ .

*Chứng minh hoạt động giải mã:*

Ta thấy

$$\begin{aligned} u(x) &= g^2(x) \bmod f(x) \\ &\equiv [g(x)]^{c-hd} \equiv [g(x)]^{\left(\sum_{i=0}^{p-1} M_i c_i\right) - hd} \\ &\equiv [g(x)]^{\left(\sum_{i=0}^{p-1} M_i (a_{\pi(i)} + d)\right) - hd} \\ &\equiv [g(x)]^{\sum_{i=0}^{p-1} M_i a_{\pi(i)}} \bmod f(x) \\ u(x) &\equiv \prod_{i=0}^{p-1} [g(x)^{a_{\pi(i)}}]^{M_i} \equiv \prod_{i=0}^{p-1} (x + \pi(i))^{M_i} \pmod{f(x)} \end{aligned}$$

Vì  $\prod_{i=0}^{p-1} (x + \pi(i))^{M_i}$  và  $s(x)$  là các đa thức định chuẩn bậc  $h$  và đồng dư với nhau theo

modulo  $f(x)$  nên  $s(x) = u(x) + f(x) = \prod_{i=0}^{p-1} (x + \pi(i))^{M_i}$

Bởi vậy tất cả các căn bậc  $h$  của  $s(x)$  đều nằm trong  $Z_p$  và áp dụng  $\pi^{-1}$  đối với các căn này ta sẽ có các tọa độ của  $M$  là 1

**Ví dụ 3.28:**

**Tạo khoá:** A thực hiện các bước sau:

(1) Chọn  $p = 7$  và  $h = 4$ .

(2) Chọn đa thức bất khả quy  $f(x) = x^4 + 3x^3 + 5x^2 + 6x + 2$  có bậc 4 trên  $Z_7$ .

Các phần tử của trường hữu hạn  $F_{7^4}$  được biểu diễn bằng các đa thức trong  $Z_7[x]$ .

(3) Chọn phần tử nguyên thủy ngẫu nhiên  $g(x) = 3x^3 + 3x^2 + 6$ .

(4) Tính các logarit rời rạc sau:

$$a_0 = \log_{g(x)}(x) = 1028$$

$$a_1 = \log_{g(x)}(x+1) = 1935$$

$$a_2 = \log_{g(x)}(x+2) = 2054$$

$$a_3 = \log_{g(x)}(x+3) = 1008$$

$$a_4 = \log_{g(x)}(x+4) = 379$$

$$a_5 = \log_{g(x)}(x+5) = 1780$$

$$a_6 = \log_{g(x)}(x+6) = 223$$

(5) Chọn phép hoán vị ngẫu nhiên trên  $\{0,1,2,3,4,5,6\}$  như sau:

$$\begin{array}{lll} \pi(0) = 6 & \pi(3) = 2 & \pi(5) = 5 \\ \pi(1) = 4 & \pi(4) = 1 & \pi(6) = 3 \\ \pi(2) = 0 & & \end{array}$$

(6) Chọn số nguyên ngẫu nhiên  $d = 1702$

(7) Tính

$$C_0 = (a_6 + d) \bmod 2400 = 1925$$

$$C_1 = (a_4 + d) \bmod 2400 = 2081$$

$$C_2 = (a_0 + d) \bmod 2400 = 330$$

$$C_3 = (a_2 + d) \bmod 2400 = 1356$$

$$C_4 = (a_1 + d) \bmod 2400 = 1237$$

$$C_5 = (a_5 + d) \bmod 2400 = 1082$$

$$C_6 = (a_3 + d) \bmod 2400 = 310$$

(8) Khoá công khai của A là  $((C_0, C_1, C_2, C_3, C_4, C_5, C_6), p = 7, h = 4)$

Khoá bí mật của A là  $(f(x), g(x), \pi, d)$

### Mã hoá:

Đề mã hoá bản tin  $m = 22$  gửi cho A, B làm như sau:

(1) Nhận khoá công khai của A.

(2) Biểu diễn m như một xâu bit độ dài 5:  $m = 10110$  (Chú ý rằng  $\left\lceil \lg \binom{7}{4} \right\rceil = 5$ )

(3) Dùng phương pháp đã nêu ở trên bước c trong thuật toán trên để biến đổi m thành véc tơ nhị phân M có độ dài M:  $M = (1, 0, 1, 1, 0, 0, 1)$

(4) Tính  $C = (C_0 + C_2 + C_3 + C_6) \bmod 2400 = 1521$

(5) Gửi  $C = 1521$  cho A

### Giải mã

A nhận bản mã C và giải mã:

(1) Tính  $r = (c - hd) \bmod 2400 = 1913$

(2) Tính  $u(x) = g(x)^{1913} \bmod f(x) = x^3 + 3x^2 + 2x + 5$

(3) Tính  $g(x) = u(x) + f(x) = x^4 + 4x^3 + x^2 + x$

(4) Phân tích  $s(x) = x(x+2)(x+3)(x+6)$

(Do đó  $t_1 = 0, t_2 = 2, t_3 = 3, t_4 = 6$ )

(5) Các thành phần của M bằng 1 có các chỉ số

$$\pi^{-1}(0) = 2 \quad \pi^{-1}(2) = 3 \quad \pi^{-1}(3) = 6 \quad \pi^{-1}(6) = 0$$

Bởi vậy  $M = (1, 0, 1, 1, 0, 0, 1)$

(6) Sử dụng bước 6 trong thuật toán giải mã để biến đổi M thành số nguyên  $m = 22$  và như vậy khôi phục được bản rõ ban đầu

**Chú ý:**

- Hệ mật này được xem là an toàn nếu không bị lộ khoá bí mật.
- Có thể mở rộng hệ mật này cho trường hợp  $Z_p$  với  $p$  là lũy thừa của một số nguyên tố.
- Để làm cho bài toán logarit rời rạc là dễ giải, các tham số  $p$  và  $h$  phải chọn sao cho  $q = p^h - 1$  chỉ có các nhân tử có giá trị nhỏ.
- Trong thực tế kích thước khuyến nghị của các tham số là  $p \approx 200, h \approx 25$  (Ví dụ  $p = 197$  và  $h = 24$ )
- Trở ngại lớn nhất của thuật toán là khoá công khai với kích thước chừng  $p \cdot h \log p$  bit là quá lớn. Ví dụ với  $p = 197$  và  $h = 24$  khoá công khai có chừng 36.000 bit.

### 3.7. BÀI TOÁN MÃ SỬA SAI VÀ HỆ MẬT McELIECE

#### 3.7.1. Bài toán mã sửa sai

**Định nghĩa 3.26:**

Giải sử  $k, n$  là các số nguyên dương và  $k \leq n$ . Mã  $C(n, k)$  là một không gian  $k$  chiều của  $(\mathbb{F}_2)^n$  (không gian vectơ của tất cả các vectơ nhị phân  $n$  chiều).

Ma trận sinh  $G_{k \times n}$  của mã  $C(n, k)$  là ma trận nhị phân có kích thước  $k \times n$ , các hàng của ma trận này tạo nên cơ sở của C.

Ma trận  $H_{r \times n}$  với  $r = n - k$  là ma trận kiểm tra, với:  $G \cdot H^T = [0]$



Giả sử  $y, z \in (\mathbb{F}_2)^n$ , trong đó  $y = (y_1, y_2, \dots, y_n)$  và  $z = (z_1, z_2, \dots, z_n)$ . Ta xác định khoảng cách Hamming:  $d(y, z) = |\{i : 1 \leq i \leq n, y_i \neq z_i\}|$  tức là số các toạ độ mà ở đó  $y$  và  $z$  khác nhau.

Khoảng cách mã được định nghĩa như sau:

$$d = \min \{d(y, z)\} \quad (3.26)$$

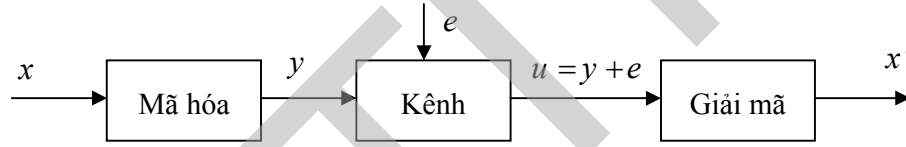
với  $y, z$  là hai từ mã khác nhau.

Mã sửa sai  $C(n, k)$  có khoảng cách  $d$  được ký hiệu là mã  $C(n, k, d)$ .

Mã sửa sai được dùng để sửa các sai ngẫu nhiên xảy ra khi truyền số liệu (nhị phân) qua kênh có nhiễu. Điều đó được thực hiện như sau: Giả sử  $G$  là một ma trận sinh đối với mã  $C(n, k, d)$ ,  $x$  là vectơ nhị phân  $k$  chiều cần truyền đi. Người gửi Alice sẽ mã hoá  $x$  thành một vectơ  $n$  chiều  $y = xG$  rồi truyền  $y$  qua kênh.

Có hai phương pháp giải mã:

\* Phương pháp "người láng giềng gần nhất"



$x \in (\mathbb{F}_2)^k$  là bản tin đầu vào ( $k$  bit)

$y \in (\mathbb{F}_2)^n$  là từ mã đầu ra ( $n$  bit):  $y = xG$

Giả sử Bob nhận được vectơ  $n$  chiều  $u$  không giống  $y$ , Bob sẽ giải mã  $u$  bằng chiến thuật giải mã "người láng giềng gần nhất". Theo chiến thuật này, Bob sẽ giải mã tất cả  $2^k$  vectơ đầu vào  $x$  để được tập từ mã  $|C| = 2^k$  vectơ, sau đó so sánh tập này với vectơ thu được  $u$  để tìm thấy  $y'$  có khoảng cách tới  $u$  nhỏ nhất. Và từ đó sẽ xác định vectơ  $k$  chiều  $x'$  sao cho  $y' = x'G$ . Bob hy vọng  $y = y'$  và do đó  $x' = x$  (tức là Bob tin rằng các sai số trên đường truyền đã được sửa).

Dễ dàng thấy rằng, nếu sai số trên đường truyền nhiều nhất là  $(d-1)/2$  thì trên thực tế chiến thuật này sẽ sửa được tất cả các sai.

Vì  $|C| = 2^k$  khi Bob so sánh  $u$  với mỗi từ mã, anh ta phải kiểm tra  $2^k$  vectơ là một số lớn theo hàm mũ. Nói cách khác, thuật toán này không phải là thuật toán chạy trong thời gian đa thức. Hay độ phức tạp của thuật toán theo hàm mũ, và khi giá trị  $k$  lớn thì nó sẽ là bài toán khó giải.

\* Phương pháp giải mã theo syndrome

Một biện pháp khác (tạo cơ sở cho nhiều thuật toán giải mã thực tế) dựa trên khái niệm về syndrom. Các hàng của ma trận kiểm tra  $H$  sẽ tạo cơ sở cho các phần bù trực giao của  $C$  (ký hiệu là  $C^\perp$ ) và được gọi là mã đối ngẫu với  $C$ . Nói cách khác, các hàng của  $H$  là những vectơ độc lập tuyến tính, còn  $GH^T = [0]_{k \times r}$  là một ma trận  $[0]$  cấp  $k \times r$ .

Cho vectơ  $y \in (\mathbb{F}_2)^n$ , ta xác định syndrom của  $y$  là  $s(y) = yH^T$ . Syndrom  $s(y)$  là một vectơ hàng có  $r = n - k$  thành phần.

**Định lý 3.20:**

Giả sử  $C$  là một bộ mã  $(n, k)$  có ma trận sinh  $G$  và ma trận kiểm tra tính chẵn lẻ  $H$ . Khi đó  $y \in (\mathbb{F}_2)^n$  là một từ mã khi và chỉ khi  $yH^T = [00\dots 0]$ .

Hơn nữa nếu  $y \in C, e \in (\mathbb{F}_2)^n$  và  $u = y + e$  thì  $uH^T = eH^T$  hay  $s(u) = s(e) = uH^T$ .

Định lý trên phát biểu rằng syndrom chỉ phụ thuộc vào các sai số mà không phụ thuộc vào từ mã cụ thể nào được truyền đi.

Điều này gợi ý tới một cách giải mã gọi là *giải mã theo syndrom*. Trước tiên Bob tính  $s(u) = uH^T$  nếu  $s(u) = [0]$ , thì anh ta có từ mã đúng  $y = u$  và tìm ra  $x$ . Nếu không thì Bob sẽ lần lượt tạo tất cả các vectơ sai  $e$  có trọng số 1. Với mỗi vectơ này, Bob tính  $s(e) = eH^T$ , nếu có một vectơ  $e$  nào đó thỏa mãn  $s(e) = eH^T = s(u)$  thì Bob sẽ có từ mã đúng là  $y = u - e$  và từ sẽ tìm lại được  $x$ . Nếu không có  $e$  nào thỏa mãn Bob tiếp tục làm như thế với các vectơ sai có trọng số 2, 3, ...,  $\lfloor (d-1)/2 \rfloor$  để tìm ra  $e$  thỏa mãn  $s(e) = s(u)$ .

Số các trường hợp sai  $N_e \leq t$ , tính như sau (với  $t$  là số sai khả sửa  $t = \lfloor (d-1)/2 \rfloor$ ):

$$N_e = \sum_{i=0}^t C_n^i \quad (3.27)$$

Như vậy, khi giải mã theo syndrom Bob phải so sánh  $s(u)$  với  $s(e)$  trong  $N_e$  trường hợp. Và theo (3.27) độ phức tạp của bài toán tương đương  $2^r$ , tức là vẫn theo hàm mũ, khi  $r$  lớn thì bài toán là khó giải.

### 3.7.2. Hệ mật McEliece

#### 3.7.2.1. Tạo khóa

Mỗi bên liên lạc tạo cho mình một cặp khóa công khai - khóa bí mật theo các bước sau:

*Bước 1:* Chọn một mã tuyến tính sửa sai  $(n, k, d)$  (khuyến nghị  $k \approx n/2$ ) với ma trận sinh  $G$ , bộ mã có một thuật toán giải mã hiệu quả.

*Bước 2:*

+ Chọn ma trận hoán vị  $P_n$  cấp  $n \times n$ , sao cho tồn tại ma trận nghịch đảo  $P_n^{-1}$  với:  $P_n P_n^{-1} = I$  (với  $I$  là ma trận đơn vị). Mỗi hàng và mỗi cột của  $P_n$  chỉ có duy nhất một con số 1.

+ Chọn một ma trận khả nghịch  $S_{k \times k}$  ( $\exists S^{-1}, SS^{-1} = I$ )

*Bước 3:* Tính  $G' = SGP$

*Bước 4:* + Khóa công khai:  $(G', t)$ .

+ Khóa bí mật:  $(S, P, G)$

Chú ý: Các tham số của mã Goppa có dạng  $n = 2^v, d = 2t + 1$  và  $k = n - vt$ . Để áp dụng trong thực tế cho một hệ mật khoá công khai, McEliece đề nghị chọn  $v = 10$  và  $t = 50$ . Điều này ứng với mã Goppa (1024, 524, 101). Mỗi bản rõ  $m$  là một vectơ nhị phân cấp 524 và mỗi bản mã  $C$  là một vectơ nhị phân cấp 1024. Khóa công khai  $G'$  là một ma trận nhị phân cấp  $524 \times 1024$ .

### 3.7.2.2. Mã hóa

Giả sử B cần gửi bản tin  $m$  cho A, B thực hiện các bước sau:

Bước 1: B nhận khóa công khai của A:  $(G', t)$

Bước 2: B tính  $y = mG' = xG', (x = m)$

Bước 3: B chọn  $e \in (\mathbb{F}_2)^n$  thỏa mãn  $W(e) \leq t$

Bước 4: B tính  $c = y + e$  (Chèn thêm vectơ sai)

Bước 5: B gửi bản mã  $c$  cho A.

### 3.7.2.3. Giải mã

A nhận bản mã  $c$  từ B và tiến hành giải mã theo các bước sau:

Bước 1: A tính:

$$\begin{aligned} u &= cP^{-1} = (y + e)P^{-1} = yP^{-1} + eP^{-1} \\ &= xG'P^{-1} + eP^{-1} \\ &= xSGPP^{-1} + eP^{-1} = xSG + eP^{-1} \end{aligned}$$

Hay 
$$u = x'G + eP^{-1} = y' + e'$$

Trong đó:  $e' = eP^{-1}$  là một vectơ sai nào đó.

Bước 2: A giải mã sửa sai từ mã  $u$  để tìm  $x'$ .

Bước 3: A tính  $x = x'S^{-1} = x = m$  để lấy lại bản tin rõ.

**Ví dụ 3.29:**

\* Tạo khóa: Bên A tạo khóa.

Bước 1: Chọn mã tuyến tính sửa sai  $(n, k, d) = (7, 4, 3)$ , với ma trận sinh:

$$G = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 1 & 0 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 \\ 0 & 0 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Và 
$$t = \left\lfloor \frac{d-1}{2} \right\rfloor = 1$$

Bước 2:

+ Chọn ma trận hoán vị  $P_n$  :

$$P_n = \begin{pmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

+ Chọn một ma trận khả nghịch  $S_{k \times k}$  :

$$S = \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 \\ 0 & 1 & 1 & 1 \\ 1 & 1 & 0 & 0 \end{pmatrix}$$

Bước 3: Tính  $G' = SGP$

$$G' = \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix}$$

Bước 4: + Khóa công khai:  $(G', t)$ .

+ Khóa bí mật:  $(S, P, G)$

\* Mã hóa:

Giả sử B cần gửi bản tin  $m = x = (1 \ 1 \ 0 \ 1)$  cho A.

Bước 1: B nhận khóa công khai của A:  $(G', t)$

Bước 2: B tính  $y = mG'$

$$y = (1 \ 1 \ 0 \ 1) \begin{pmatrix} 1 & 1 & 1 & 1 & 0 & 0 & 0 \\ 1 & 1 & 0 & 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 & 1 & 1 & 0 \end{pmatrix} = (0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0)$$

Bước 3: B chọn vectơ sai  $e = (0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0)$  có  $W(e) = 1 = t$

Bước 4: B tính  $c = y + e$

$$c = (0 \ 1 \ 1 \ 0 \ 0 \ 1 \ 0) + (0 \ 0 \ 0 \ 0 \ 1 \ 0 \ 0) = (0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0)$$

Bước 5: B gửi bản mã  $c$  cho A.

\* *Giải mã:*

Khi A nhận được bản mã  $c$ , A tiến hành giải mã:

Bước 1: A tính  $u = cP^{-1}$

$$u = (0 \ 1 \ 1 \ 0 \ 1 \ 1 \ 0) \begin{pmatrix} 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \end{pmatrix} = (1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 1)$$

Bước 2: B giải mã  $u = y' + e'$  bằng phương pháp syndrom tìm được:

$$y' = (1 \ 0 \ 0 \ 0 \ 1 \ 1 \ 0)$$

Cần để ý là  $e' \neq e$  do phép nhân với  $P^{-1}$

Sau đó B lập  $x' = (1 \ 0 \ 0 \ 0)$  (bốn thành phần đầu tiên của  $y'$ , vì mã sửa sai là mã hệ thống).

Bước 3:

Cuối cùng B tính ra bản rõ:

$$x = x'S = (1 \ 0 \ 0 \ 0) \begin{pmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 0 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 0 & 1 \end{pmatrix} = (1 \ 1 \ 0 \ 1) = m$$

### 3.8. ĐƯỜNG CONG ELLIPTIC VÀ CÁC HỆ MẬT LIÊN QUAN

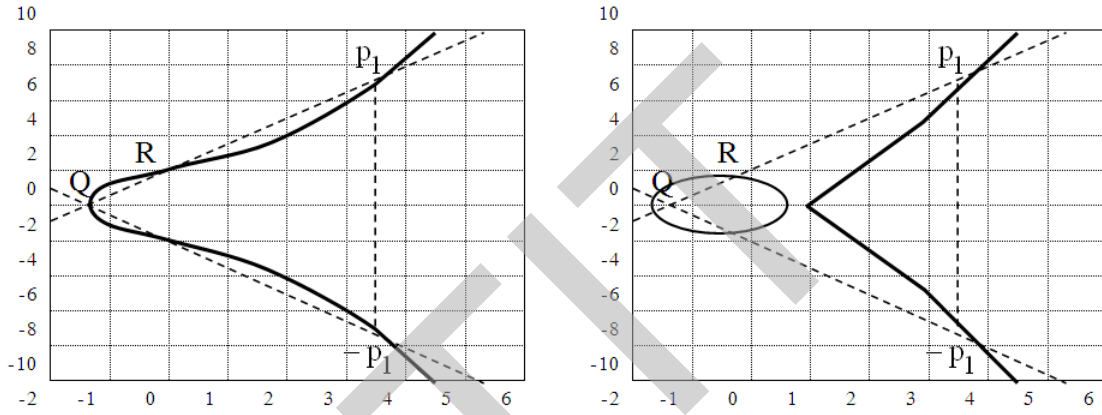
#### 3.8.1. Các đường cong Elliptic.

Một đường cong Elliptic là một phương trình bậc 3 có dạng sau:

$$y^2 + axy + by = x^3 + cx^2 + dx + e \quad (3.28)$$

Trong đó  $a, b, c, d, e$  là các số thực.

Trên các đường cong E ta xác định một phép cộng đặc biệt với một điểm O được gọi là điểm vô cực. Nếu trên đường thẳng cắt đường cong E ở ba điểm thì tổng của chúng bằng điểm vô cực O (điểm O này có vai trò như phần tử đơn vị trong phép cộng này). Hình 3.7 sau mô tả các đường cong  $E: y^2 = x^3 + 2x + 5$  và  $y^2 = x^3 - 2x + 1$



Hình 3.7. Các đường cong  $y^2 = x^3 + 2x + 5$  và  $y^2 = x^3 - 2x + 1$

#### 3.8.2. Các đường cong Elliptic trên trường Galois

**Định nghĩa 3.27:**

Đường cong Elliptic trên trường hữu hạn  $Z_p$  (với  $p$  là số nguyên tố) dạng Weiestrass được mô tả như sau:

$$y^2 \equiv x^3 + ax + b \pmod{p} \quad (3.29)$$

với  $x \in Z_p$ ;  $a, b$  nguyên dương.

Điều kiện tồn tại là:

$$\Delta = 4a^3 + 27b^2 \pmod{p} \neq 0 \quad (3.30)$$

**Ví dụ 3.30:**

Xét  $p = 17$  và đường cong elliptic:  $y^2 \equiv x^3 + x + 1 \pmod{17}$ .

$\Delta = 4 + 27 \pmod{17} = 14 \neq 0$  đảm bảo điều kiện tồn tại.

Xác định các thẳng dư bậc 2:  $Q_{17}$

$i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$i^2$	1	4	9	16	8	2	15	13	13	15	2	8	16	9	4	1

Vậy  $Q_{17} = \{1, 4, 9, 16, 8, 2, 15, 13\}$  và các căn bậc hai:

$$\begin{aligned}\sqrt{1} &= (1, 16); \sqrt{4} = (2, 15); \sqrt{9} = (3, 14); \sqrt{16} = (4, 13); \\ \sqrt{8} &= (5, 12); \sqrt{2} = (6, 11); \sqrt{15} = (7, 10); \sqrt{13} = (8, 9)\end{aligned}$$

Chú ý: Việc tính thặng dư bậc 2 và các căn bậc 2 theo các định nghĩa 3.18, 3.19; và các định lý 4.14 và 3.15.

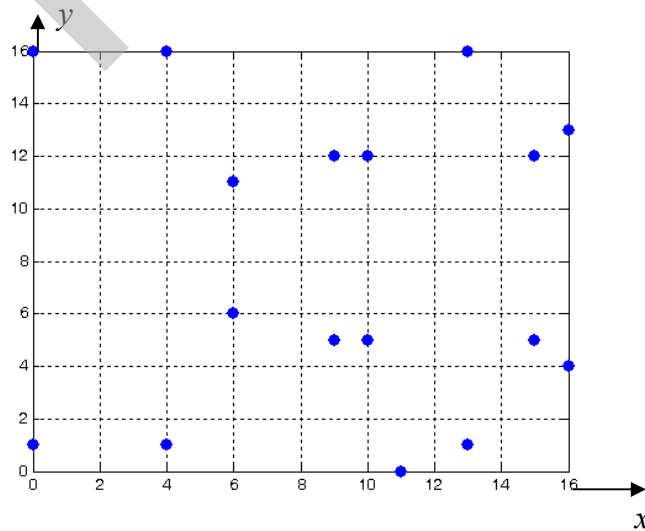
Từ đây tính được các cặp giá trị  $(x, y)$  trên đường cong elliptic như bảng dưới:

$x$	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$y^2$	1	3	11	14	1	12	2	11	11	8	8	0	7	1	5	8	16
$y^2 \in Q_{17}?$	Y	N	N	N	Y	N	Y	N	N	Y	Y	<u>N</u>	N	Y	N	Y	Y
$y_1$	1				1		6			5	5	0		1		5	4
$y_2$	16				16		11			12	12	0		16		12	13

Chú ý: mặc dù  $y^2 = 0$  không thuộc  $Q_{17}$  nhưng số 0 có căn bậc hai chính là 0.

Từ đây ta có được nhóm cộng  $E_{17}(1, 1)$  gồm 18 phần tử trên đường cong elliptic  $y^2 \equiv x^3 + x + 1 \pmod{17}$  như sau:

$$\begin{aligned}E_{17}(1, 1) = \{ & (0, 1), (0, 16), (4, 1), (4, 16), (6, 6), (6, 11), (9, 5), (9, 12), (10, 5) \\ & (10, 12), (11, 0), (13, 1), (13, 16), (15, 5), (15, 12), (16, 4), (16, 13), 0\}\end{aligned}$$



Hình 3.8. Tọa độ các điểm của nhóm cộng trên đường cong elliptic  $y^2 \equiv x^3 + x + 1 \pmod{17}$

Trong nhóm cộng này có thêm phần tử cuối cùng là phần tử zero (0). Nhóm này có cấp bằng 18 và có 6 phần tử nguyên thủy ( $\Phi(18) = 6$ ) để tạo nhóm cyclic.

### Tọa độ các điểm của nhóm được mô tả trong Hình 3.8

#### 3.8.3. Các phép toán cộng và nhân trên các nhóm E

Giả sử  $P = (x_1, y_1)$ ,  $Q = (x_2, y_2)$  là các điểm trong nhóm  $E_p(a, b)$ , 0 là điểm vô cực. Các quy tắc đối với phép cộng trên nhóm con  $E_p(a, b)$  như sau:

- (1)  $P + 0 = 0 + P = P$ .
- (2) Nếu  $x_2 = x_1$  và  $y_2 = -y_1$  tức là  $P = (x_1, y_1)$  và  $Q = (x_2, y_2) = (x_1, -y_1) = -P$  thì  $P + Q = 0$ .
- (3) Nếu  $Q \neq -P$  thì tổng  $P + Q = K(x_3, y_3)$  với tọa độ  $x_3, y_3$  của  $K$  xác định như sau:

$$\begin{aligned} x_3 &= \lambda^2 - x_1 - x_2 \bmod p \\ y_3 &= \lambda(x_1 - x_3) - y_1 \bmod p \end{aligned} \quad (3.31)$$

Trong đó:

$$\lambda = \begin{cases} \frac{y_2 - y_1}{x_2 - x_1} \bmod p, & \text{nếu } P \neq Q \\ \frac{3x_1^2 + a}{2y_1} \bmod p, & \text{nếu } P = Q \end{cases} \quad (3.32)$$

\* Cách xây dựng nhóm cộng trên đường cong elliptic

Các phần tử của nhóm được tính như sau:

- Phần tử zero:  $0(\infty, \infty)$  nằm ngoài tập hợp.
- Các phần tử khác tính theo quy tắc phép cộng các điểm trên đường cong Elliptic như trình bày trên đây.

**Ví dụ 3.31:**

Xây dựng nhóm  $E_p(a, b) = E_{17}(1, 1)$  từ phần tử nguyên thủy  $P(0, 1)$  và tìm tất cả các phần tử nguyên thủy của nhóm.

Theo nguyên tắc tính các điểm ở biểu thức (3.31) và (3.32) ta tính được các điểm của nhóm cộng như sau.

+ Điểm  $P(0, 1)$

+ Điểm  $2P = P + P$

$$\lambda = \frac{3 \cdot 0 + 1}{2 \cdot 1} = 2^{-1} \bmod 17 = 9 \bmod 17 \Rightarrow x_3 = 81 \bmod 17 = 13, y_3 = 1$$



Vậy  $2P(13,1)$

Chú ý: số 2 và 9 là hai số nghịch đảo, có nhiều cách tính nghịch đảo, sau đây là một cách tính theo nhóm nhân  $Z_{17}^*$  có phần tử sinh là 3.

Bảng 3.11. Nhóm nhân  $Z_{17}^*$  với phần tử sinh  $\alpha = 3$

$i$	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
$3^i \bmod 17$	3	9	10	13	5	15	11	16	14	8	7	4	12	2	6	1

+ Điểm  $3P = P + 2P$

$$\lambda = \frac{0}{12} = 0 \Rightarrow \begin{cases} x_3 = -13 \bmod 17 = 4 \\ y_3 = -1 \bmod 17 = 16 \end{cases}$$

$$\Rightarrow 3P(4,16)$$

+ Điểm  $4P = P + 3P$

$$\lambda = \frac{15}{4} = 15 \cdot 4^{-1} \bmod 17 = 15 \cdot 13 \bmod 17 = 8 \Rightarrow \begin{cases} x_3 = 9 \\ y_3 = 12 \end{cases}$$

$$\Rightarrow 4P(9,12)$$

+ Điểm  $5P = P + 4P$

$$\lambda = \frac{15}{9} = 15 \cdot 9^{-1} \bmod 17 = 15 \cdot 6 \bmod 17 = 5 \Rightarrow \begin{cases} x_3 = 16 \\ y_3 = 4 \end{cases}$$

$$\Rightarrow 5P(16,4)$$

+ Điểm  $6P = P + 5P$

$$\lambda = \frac{3}{16} = 3 \cdot 16^{-1} \bmod 17 = 3 \cdot 16 \bmod 17 = 14 \Rightarrow \begin{cases} x_3 = 10 \\ y_3 = 12 \end{cases}$$

$$\Rightarrow 6P(10,12)$$

+ Điểm  $7P = P + 6P$

$$\lambda = \frac{11}{10} = 11 \cdot 10^{-1} \bmod 17 = 11 \cdot 12 \bmod 17 = 13 \Rightarrow \begin{cases} x_3 = 6 \\ y_3 = 6 \end{cases}$$

$$\Rightarrow 7P(6,6)$$

+ Điểm  $8P = P + 7P$

$$\lambda = \frac{5}{6} = 5 \cdot 6^{-1} \bmod 17 = 5 \cdot 3 \bmod 17 = 15 \Rightarrow \begin{cases} x_3 = 15 \\ y_3 = 12 \end{cases}$$

$$\Rightarrow 7P(15,12)$$

$$+ \text{Điểm } 9P = P + 8P$$

$$\lambda = \frac{11}{15} = 11 \cdot 15^{-1} = 11 \cdot 8 \bmod 17 = 3 \Rightarrow \begin{cases} x_3 = 11 \\ y_3 = 0 \end{cases}$$

$$\Rightarrow 9P(11,0)$$

Từ điểm  $10P$  trở đi có thể tính theo quy tắc sau:  $P(x,y) = -P(x,-y)$

$$+ \text{Điểm } 10P = -8P \Rightarrow 10P(15,-12) = 10P(15,5). \text{ (Chú ý: } -12 \bmod 17 \equiv 5 \bmod 17)$$

$$+ \text{Điểm } 11P = -7P \Rightarrow 11P(6,-6) = 11P(6,11).$$

$$+ \text{Điểm } 12P = -6P \Rightarrow 12P(10,-12) = 12P(10,5).$$

$$+ \text{Điểm } 13P = -5P \Rightarrow 13P(16,-4) = 13P(16,13).$$

$$+ \text{Điểm } 14P = -4P \Rightarrow 14P(9,-12) = 14P(9,5).$$

$$+ \text{Điểm } 15P = -3P \Rightarrow 15P(4,-16) = 15P(4,1).$$

$$+ \text{Điểm } 16P = -2P \Rightarrow 16P(13,-1) = 16P(13,16).$$

$$+ \text{Điểm } 17P = -P \Rightarrow 17P(0,-1) = 17P(0,16).$$

$$+ \text{Điểm } 18P = 0.$$

Các điểm nguyên thủy của nhóm nhân được xác định như sau:

Nếu  $P$  là một điểm nguyên thủy thì  $kP$  sẽ là nguyên thủy với  $(k, |E_p(a,b)|) = 1$

Trong ví dụ này ta có  $|E_{17}(1,1)| = 18$  vậy các điểm nguyên thủy  $kP$  có  $(k, 18) = 1$ , hay  $k = \{1, 5, 7, 11, 13, 17\}$ .

Các điểm nguyên thủy là:  $P, 5P, 7P, 11P, 13P, 17P$  (6 điểm)

### 3.8.4. Các hệ mật trên đường cong elliptic

#### 3.8.4.1. Trao đổi thỏa thuận khóa Diffie - Hellman

Xét nhóm  $E_p(a,b)$  trên đường cong elliptic và  $P$  là điểm nguyên thủy  $P \in E_p(a,b)$ .

Quá trình trao đổi khóa diễn ra như sau:

A	B
+ A chọn ngẫu nhiên $x$ với điều kiện: $(1 < x < \#E_p(a,b))$ sau đó A tính $xP$ và gửi cho B.	+ B chọn ngẫu nhiên $y$ với điều kiện: $(1 < y < \#E_p(a,b))$ sau đó B tính $yP$ và gửi cho A.

+ A nhận $yP$ và tính ra khóa dùng chung: $K = x(yP) = xyP$	+ B nhận $xP$ và tính ra khóa dùng chung: $K = y(xP) = xyP$
--	--

**Ví dụ 3.32:** Xét  $E_{17}(1,1)$  và điểm nguyên thủy  $P(0,1)$  (Nhóm  $E_{17}(1,1)$  như đã tính ở ví dụ 3.31)

A	B
+ A chọn ngẫu nhiên $x = 3$ và tính: $3P = (4,6)$ và gửi cho B. + A nhận $5P$ và tính ra khóa dùng chung: $K = 3(5P) = 15P(4,1)$	+ B chọn ngẫu nhiên $y = 5$ và tính: $5P = (16,4)$ và gửi cho A. + B nhận $3P$ và tính ra khóa dùng chung: $K = 5(3P) = 15P(4,1)$

#### 3.8.4.2. Hệ mật Omura – Massey

Tạo khóa:

- + Hai bên liên lạc chọn trước nhóm  $E_p(a,b)$  làm khóa công khai
- + Khóa bí mật tạo như sau:
  - A chọn ngẫu nhiên hai số  $m, n$  làm khóa bí mật, sao cho  $m + n = \#E_p(a,b)$
  - B chọn ngẫu nhiên hai số  $u, v$  làm khóa bí mật, sao cho  $u + v = \#E_p(a,b)$

Giả sử A cần gửi bản tin  $M$  cho B, quá trình truyền tin bảo mật diễn ra như sau:

$A_{(m,n)}$	$B_{(u,v)}$
+ A tính $mP + M$ và gửi cho B	+ B nhận $mP + M$ và tính $(mP + M) + uP$ và gửi lại A.
+ A nhận $(mP + M) + uP$ và tính: $[(mP + M) + uP] + nP = M + uP$ và gửi cho B.	+ B nhận $M + uP$ và giải mã ra bản tin: $(M + uP) + vP = M$

**Ví dụ 3.33:**

Xét  $E_{17}(1,1)$  và điểm nguyên thủy  $P(0,1)$  (Nhóm  $E_{17}(1,1)$  như đã tính ở ví dụ 3.31).  
Bản tin  $M(4,16) = 3P$  (Chú ý phải có phép biến đổi bản tin thành một trong các điểm thuộc  $E_{17}(1,1)$ )

- + Khóa công khai:  $E_{17}(1,1)$

+ Khóa bí mật:

- A chọn ngẫu nhiên hai số  $(m, n) = (3, 15)$  với  $m + n = 18 = \#E_{17}(1, 1)$

- B chọn ngẫu nhiên hai số  $(u, v) = (5, 13)$  với  $u + v = 18 = \#E_{17}(1, 1)$

Quá trình truyền tin bảo mật:

$A_{(3,15)}$	$B_{(5,13)}$
+ A tính $mP + M = 3P + 3P = 6P(10, 12)$ và gửi cho B	+ B nhận $6P(10, 12)$ và tính $6P + 5P = 11P(6, 11)$ và gửi lại A.
+ A nhận $11P(6, 11)$ và tính: $11P + 15P = 8P(15, 12)$ và gửi cho B.	+ B nhận $8P(15, 12)$ và giải mã ra bản tin: $8P + 13P = 3P(4, 16) = M$

### 3.8.4.3. Hệ mật Elgamal

#### a) Tạo khóa

Mỗi bên liên lạc tạo cho mình một cặp khóa công khai – bí mật theo các bước sau:

*Bước 1:* Chọn nhóm  $E_p(a, b)$  và điểm nguyên thủy  $P$

*Bước 2:* Chọn ngẫu nhiên  $a$  với  $(1 < a < \#E_p(a, b))$  và tính  $aP = Q$

*Bước 3:* + Khóa công khai:  $(E_p(a, b), P, Q)$

+ Khóa bí mật:  $a$

#### b) Mã hóa

Giả sử B cần gửi bản tin  $M$  cho A, B thực hiện các bước sau:

*Bước 1:* B nhận khóa công khai của A:  $(E_p(a, b), P, Q)$

*Bước 2:* B chọn  $k$  ngẫu nhiên  $(1 < k < \#E_p(a, b))$  và tính:

$$\begin{cases} \gamma = kP \\ \delta = M + kQ \end{cases}$$

*Bước 3:* B gửi bản mã  $C = (\gamma, \delta)$  cho A.

#### c) Giải mã

A nhận bản mã  $C = (\gamma, \delta)$  và giải mã theo các bước:

*Bước 1:* A tính  $a\gamma = akP = kQ$

*Bước 2:* A tính  $\delta - kQ = M + kQ - kQ = M$

**Ví dụ 3.34:** Xét  $E_{17}(1,1)$  và điểm nguyên thủy  $P(0,1)$  (Nhóm  $E_{17}(1,1)$  như đã tính ở ví dụ 3.31). Bản tin  $M(15,5) = 10P$

\* *Bên A tạo khóa:*

*Bước 1:* A chọn  $a = 3$  và tính  $aP = 3P(4,16) = Q$

*Bước 2:* + Khóa công khai của A:  $(E_{17}(1,1), P(0,1), Q(4,16))$

+ Khóa bí mật:  $a = 3$

\* *Mã hóa:*

B gửi bản tin  $M = 10P(15,5)$  cho A

*Bước 1:* B nhận khóa công khai của A:  $(E_{17}(1,1), P(0,1), Q(4,16))$

*Bước 2:* B chọn  $k = 5$  ngẫu nhiên và tính:

$$\begin{cases} \gamma = kP = 5P(16,4) \\ \delta = M + kQ = 10P + 5Q = 10P + 5 \times 3P = 7P(6,6) \end{cases}$$

*Bước 3:* B gửi bản mã  $C = (\gamma, \delta) = (5P, 7P)$  cho A.

\* *Giải mã:*

A nhận bản mã  $C = (5P, 7P)$  và giải mã theo các bước:

*Bước 1:* A tính  $a\gamma = 3 \times 5P = 15P(4,1)$

*Bước 2:* A tính  $\delta - kQ = 7P - 15P = 10P(15,5) = M$

### 3.8.5. Độ an toàn của hệ mật trên đường cong Elliptic

Sức mạnh ECC nằm ở sự khó khăn đối với thám mã khi phải xác định số ngẫu nhiên bí mật  $k$  từ  $kP$  và  $P$ . Phương pháp nhanh nhất để giải bài toán này là phương pháp phân tích S - Pollard. Để phá ECC độ phức tạp tính toán khi dùng phương pháp S - Pollard là  $3,8.10^{10} MIPS$  - năm với kích thước khóa 150 bit (đây là số năm cần thiết với một hệ thống tính toán có tốc độ hàng triệu lệnh/giây). Để so sánh với phương pháp nhanh nhất phá RSA (là phương pháp sàng trừng số để phân tích hợp số  $n$  thành tích của 2 số nguyên tố  $p$  và  $q$ ) ta thấy rằng với  $n$  có kích thước 768 bit độ phức tạp tính toán là:  $2.10^8 MIPS$  - năm, với  $n$  có kích thước 1024 bit, độ phức tạp tính toán là  $3.10^{11}$  năm.

Nếu độ dài khóa của RSA tăng lên tới 2048 bit thì cần  $3.10^{20} MIPS$  - năm, trong khi đó với ECC chỉ cần độ dài khóa là 234 bit đã phải yêu cầu tới  $1,6.10^{28} MIPS$  - năm.

### 3.9. ƯU VÀ NHƯỢC ĐIỂM CỦA MẬT MÃ KHÓA CÔNG KHAI

#### 3.9.1. Ưu điểm

- + Mật mã khóa công khai không sử dụng kênh an toàn riêng để truyền khóa.
- + Dễ sinh khóa, bảo vệ và lưu trữ khóa.
- + Dễ xây dựng các dịch vụ an toàn khác như: xác thực, chữ ký số...

So sánh với mật mã khóa bí mật như bảng sau (giả sử ta xét  $n$  người dùng trên mạng):

Bảng 3.12. So sánh số lượng khóa của mật mã khóa bí mật và mật mã khóa công khai

$n$ người dùng	Số khóa cần tạo	Số khóa cần lưu trữ
Mật mã khóa bí mật	$\frac{n(n-1)}{2}$	$n-1$
Mật mã khóa công khai	$2n$	1

#### 3.9.2. Nhược điểm

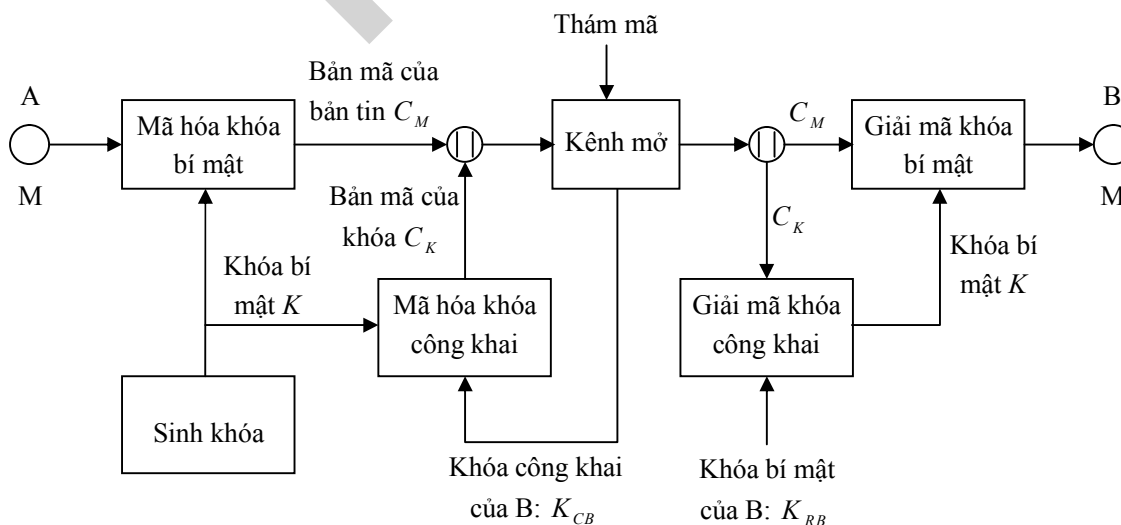
- + Nói chung mật mã khóa công khai phức tạp (phải xây dựng trên các trường lớn, mạch điện phức tạp, thời gian xử lý chậm hơn so với mật mã khóa bí mật)
- + Hiệu quả không cao (nói chung các hệ mật khóa công khai có tốc độ mã thấp)

Từ các nhược điểm này ta thấy mật mã khóa công khai khó thực hiện cho các dịch vụ nhạy cảm với độ trễ hoặc các dịch vụ di động.

### 3.10. XÂY DỰNG CÁC CHƯƠNG TRÌNH ỨNG DỤNG KIẾN TRÚC PGP

(PGP – Pretty Good Privacy: tạm dịch là “Riêng tư tốt đẹp”)

Từ các ưu và nhược điểm của mật mã khóa công khai và mật mã khóa bí mật, người ta kết hợp ưu điểm của hai loại mật mã này xây dựng nên một cấu trúc truyền tin PGP như Hình 3.9:



Hình 3.9. Sơ đồ kiến trúc PGP

Trong sơ đồ này, bản tin  $M$  được truyền đi bảo mật sử dụng mật mã khóa bí mật với khóa là  $K$ . Khóa bí mật  $K$  được mã hóa bằng hệ mật khóa công khai từ  $A$  đến  $B$  (sử dụng khóa công khai và bí mật của  $B$ ).

Việc mã hóa bản tin bằng mật mã khóa bí mật sẽ có ưu điểm tốc độ mã hóa nhanh, hiệu quả cao, còn việc truyền khóa sử dụng mật mã khóa công khai là để thay thế kênh an toàn tiết kiệm chi phí.

### BÀI TẬP CHƯƠNG 3

**Bài 3.1:** Sử dụng thuật toán Euclide mở rộng để tìm ước chung lớn nhất của hai số  $a = 1573$ ,  $b = 308$ .

**Bài 3.2:** Hãy tính  $3^{32} \bmod 23$  bằng cách dùng thuật toán nhân và bình phương có lập.

**Bài 3.3:** Hãy tính các căn bậc hai của  $12 \bmod 37$ .

**Bài 3.4:** Tìm tất cả các phần tử nguyên thủy của nhóm nhân  $Z_{19}^*$ .

**Bài 3.5:** Tìm phần tử nghịch đảo của 3 trong  $Z_{31}^*$ .

**Bài 3.6:** Với  $m, n, s \in \mathbb{N}$  và  $p_i$  là các số nguyên tố. Hãy chứng minh các tính chất sau của hàm Phi-Euler

- a)  $\varphi(p^s) = p^s \left(1 - \frac{1}{p}\right)$ .
- b)  $\varphi(m, n) = \varphi(m) \varphi(n)$  nếu  $\text{UCLN}(m, n) = 1$ .
- c)  $\varphi(n) = m \left(1 - \frac{1}{p_1}\right) \dots \left(1 - \frac{1}{p_r}\right)$  trong đó  $m = p_1^{e_1} \dots p_r^{e_r}$  là phân tích của  $m$  thành tích của thừa số nguyên tố.

**Bài 3.7:** Hãy tính  $\varphi(490)$  và  $\varphi(768)$ .

**Bài 3.8:** Giải hệ phương trình đồng dư sau:

$$5x \equiv 20 \bmod 6$$

$$6x \equiv 6 \bmod 5$$

$$4x \equiv 5 \bmod 77$$

**Bài 3.9:** Hãy dùng thuật toán Euclide mở rộng để tính các phần tử nghịch đảo sau:

a)  $17^{-1} \bmod 101$

b)  $357^{-1} \bmod 1234$

c)  $3125^{-1} \bmod 9987$

**Bài 3.10:** Ta nghiên cứu một số tính chất của các phần tử nguyên thủy:

a) 97 là một số nguyên tố. Hãy chứng minh rằng  $x \neq 0$  là một phần tử nguyên thủy theo modulo 97 khi và chỉ khi:

$$x^{32} \neq 1 \bmod 97 \text{ và } x^{48} \neq 1 \bmod 97$$

b) Hãy dùng phương pháp này để tìm phần tử nguyên thủy nhỏ nhất theo modulo 97.

c) Giả sử  $p$  là một số nguyên tố và  $p-1$  có phân tích ra lũy thừa của các nguyên tố sau:

$$p-1 = \prod_{i=1}^n p_i^{e_i}$$

Ở đây  $p_i$  là các số nguyên tố khác nhau. Hãy chứng tỏ rằng  $x \neq 0$  là một phần tử nguyên thủy theo modulo  $p$  khi và chỉ khi  $x^{(p-1)/p_i} \neq 1 \bmod p$  với  $1 \leq i \leq n$ .

**Bài 3.11:** Cho  $\phi_{13}$ , biết  $\alpha = 2$  là phần tử nguyên thủy của  $\phi_{13}^*$ . Hãy:

- Tìm tất cả các phần tử nguyên thủy của  $\phi_{13}^*$
- Giải bài toán logarit rời rạc: Tìm  $\log_{\alpha} y$  với  $\alpha$  là phần tử nguyên thủy,  $y \in \phi_{13}^*$
- Tìm các thặng dư bậc 2 của  $\phi_{13}^*$

**Bài 3.12:** Giải bài 3.11 trong các trường hợp sau:

- $\phi_{19}$ , biết  $\alpha = 2$  là phần tử nguyên thủy của  $\phi_{19}^*$ .
- $\phi_{23}$ , biết  $\alpha = 5$  là phần tử nguyên thủy của  $\phi_{23}^*$ .
- $\phi_{29}$ , biết  $\alpha = 2$  là phần tử nguyên thủy của  $\phi_{29}^*$ .

**Bài 3.13:** Tính khoá chung trong thủ tục thoả thuận khoá Diffie – Hellman với các trường hợp sau: (với  $p$  -nguyên tố,  $\alpha$  là phần tử nguyên thủy).

- $p = 11, \alpha = 7 \in Z_{11}^*$ .
- $p = 17, \alpha = 6 \in Z_{17}^*$ .
- $p = 19, \alpha = 10 \in Z_{19}^*$ .
- $p = 23, \alpha = 11 \in Z_{23}^*$ .
- $p = 29, \alpha = 8 \in Z_{29}^*$ .

**Bài 3.14:** Hãy xây dựng hệ mật Omura-Massey để thực hiện việc truyền khoá bí mật  $k$  từ A đến B:

- Tính số các trường hợp có thể lựa chọn tham số của hệ
- Lấy 1 ví dụ với khoá được chọn bởi A là  $k = 5$



Với các trường hợp sau:  $p = 11, 13, 17, 19, 23, 29$

**Bài 3.15:** Hãy xây dựng hệ mật ElGamal với mỗi trường hợp sau và thực hiện mã hoá bản tin  $M = 7$ . ( $p$  nguyên tố,  $\alpha$  là phần tử nguyên thủy)

- a)  $p = 11, \alpha = 8$       b)  $p = 13, \alpha = 11$
- c)  $p = 17, \alpha = 10$     d)  $p = 23, \alpha = 14$
- e)  $p = 29, \alpha = 10$     f)  $p = 31, \alpha = 12$

**Bài 3.16:** Ví dụ về hệ mật RSA. Cho  $p = 7$  và  $q = 17$ .

- a) Tính  $n$ .
- b) Cho  $e$  (số mũ mã hoá) bằng 5. Hãy tính số mũ giải mã  $d$ .
- c) Hãy mã hoá và giải mã cho các số 49 và 12.

**Bài 3.17:** Người ta biết rằng đối với hệ mật RSA, tập các bản rõ bằng tập các bản mã. Tuy nhiên bạn có cho rằng một số giá trị trong không gian thông báo (bản rõ) là không mong muốn?

**Bài 3.18:** Tìm  $n$  và số mũ giải mã  $d$  của mỗi hệ mật RSA có tham số dưới đây và mã hóa bản tin  $M = 5$  với mỗi trường hợp tương ứng:

- a)  $p = 13, q = 17, e = 11$
- b)  $p = 19, q = 17, e = 149$
- c)  $p = 19, q = 23, e = 85$
- d)  $p = 23, q = 29, e = 123$

**Bài 3.19:** Trong hệ mật Rabin, giả sử  $p = 199, q = 211$ .

- a) Xác định 4 căn bậc hai của  $1 \bmod n$ , trong đó  $n = pq$ .
- b) Tính bản mã của 32767.
- c) Xác định 4 bản giải mã có thể của bản mã trên.

**Bài 3.20:** Xét trường hợp đơn giản của hệ mật Merkle-Hellman sử dụng phép hoán vị đồng nhất. Giả sử dãy siêu tăng được chọn là (2, 3, 6, 13, 27, 52) giá trị ngẫu nhiên  $w$  được chọn là 31, modulo  $M$  được chọn là 105.

- a) Hãy xác định khoá bí mật.
- b) Bản tin ở dạng nhị phân có dạng 011000\_110101\_101110.  
Hãy tính bản mã và hãy giải mã để tìm lại bản tin ban đầu.

**Bài 3.21:** Đây là một ví dụ về hệ mật ElGamal áp dụng trong  $GF(3^3)$ . Đa thức  $x^3 + x^2 + 1$  là một đa thức bất khả quy trên  $Z_3[x]$  và bởi vậy  $Z_3[x]/(x^3 + x^2 + 1)$  chính là  $GF(3^3)$ . Ta có thể gán 26 chữ cái của bảng chữ cái tiếng Anh với 26 phần tử khác không của trường và

như vậy có thể mã hoá một văn bản thông thường theo cách truyền thống. Ta sẽ dùng thứ tự theo từ điển của các đa thức khác không để thiết lập sự tương ứng.

$A \leftrightarrow 1$	$B \leftrightarrow 2$	$C \leftrightarrow x$
$D \leftrightarrow x + 1$	$E \leftrightarrow x + 2$	$F \leftrightarrow 2x$
$G \leftrightarrow 2x + 1$	$H \leftrightarrow 2x + 2$	$I \leftrightarrow x^2$
$J \leftrightarrow x^2 + 1$	$K \leftrightarrow x^2 + 2$	$L \leftrightarrow x^2 + x$
$M \leftrightarrow x^2 + x + 1$	$N \leftrightarrow x^2 + x + 2$	$O \leftrightarrow x^2 + 2x$
$P \leftrightarrow x^2 + 2x + 1$	$Q \leftrightarrow x^2 + 2x + 2$	$R \leftrightarrow 2x^2$
$S \leftrightarrow 2x^2 + 1$	$T \leftrightarrow 2x^2 + 2$	$U \leftrightarrow 2x^2 + x$
$V \leftrightarrow 2x^2 + x + 1$	$W \leftrightarrow 2x^2 + x + 2$	$X \leftrightarrow 2x^2 + 2x$
$Y \leftrightarrow 2x^2 + 2x + 1$	$Z \leftrightarrow 2x^2 + 2x + 2$	

Giả sử Bob dùng  $\alpha = x$  và  $a = 11$  trong hệ mật ElGamal, khi đó  $\alpha^a = x + 2$ . Hãy chỉ ra cách mà Bob sẽ giải mã cho bản mã sau:

(K, H) (P,X) (N,K) (H, R) (T, F) (V, Y) (E, H) (F, A) (T, W) (J, D) (V, J).

**Bài 3.22:** Mã BCH (15, 7, 5) có ma trận kiểm tra sau:

$$H = \begin{pmatrix} 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 \\ 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 1 & 0 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 & 0 & 1 & 1 & 1 & 1 \end{pmatrix}$$

Hãy giải mã cho các vector nhận được sau bằng phương pháp giải mã theo syndrom:

- $r = (1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0)$
- $r = (1, 1, 0, 1, 1, 1, 1, 0, 1, 0, 1, 1, 0, 0, 0)$
- $r = (1, 0, 1, 0, 1, 0, 0, 1, 0, 1, 1, 0, 0, 0, 0)$

## CHƯƠNG 5. CÁC THỦ TỤC VÀ CÁC CHÚ Ý TRONG THỰC TẾ KHI SỬ DỤNG MÃ HOÁ

Trong các chương 3 và 4 ta đã xem xét các ví dụ về các hệ thống mật mã được coi là mật. Có hàng trăm phương pháp mã hoá khác nhau ngoài DES và RSA là hai hệ mật khoá công khai và khoá bí mật được thừa nhận rộng rãi nhất trong thực tế.

Tuy nhiên chỉ đơn giản là có và sử dụng một hệ mật mạnh là chưa đủ để đảm bảo mọi giao dịch sử dụng hệ mật đó được đảm bảo an toàn. Có những cách thức đúng hoặc không đúng khi sử dụng các phương pháp mã hoá. Hơn nữa các thuật toán này có thể được dùng để giải quyết các vấn đề mà bí mật hoặc xác thực chỉ là một phần của nó. Trong chương này ta sẽ nghiên cứu và đánh giá các kỹ thuật sử dụng mã hoá để thiết lập một kênh liên lạc mật giữa hai người dùng. Ta cũng khai thác các cách thích hợp để sử dụng mã hoá

### 5.1. CÁC THỦ TỤC: HÀNH VI CÓ THỨ TỰ

Các hệ thống mã hoá là một công cụ quan trọng trong an toàn máy tính, chúng cho phép bạn phát thông tin ở dạng được che giấu. Chúng được dùng để phát các tài liệu và số liệu trên một kênh có thể bị thu chặn. Bằng cách sử dụng các quy ước được thiết lập giữa hai bên với hệ mật có thể được dùng cho những mục đích khác với việc liên lạc an toàn. Các quy ước này được gọi là các thủ tục, chúng sẽ là chủ đề của phần sau:

#### 5.1.1. Định nghĩa thủ tục

Một thủ tục là một dãy các bước có thứ tự mà hai bên (hoặc nhiều hơn) phải thực hiện để hoàn tất một công việc nào đó. Các bên sử dụng thủ tục phải nhất trí hoàn tất thủ tục trước khi dùng nó. Thứ tự của các bước cũng quan trọng như hoạt động của mỗi bước. Người ta sử dụng các thủ tục để điều chỉnh hành vi và quyền lợi chung.

Sử dụng điện thoại là một ví dụ đơn giản của một thủ tục. Người quay số sẽ nghe cả âm chuông và tiếng nhắc máy khi kết nối được thiết lập. Trong thực tế, thủ tục ở đây là người nhận sẽ nói trước (chẳng hạn “xin chào”, hoặc “tôi nghe đây”). Người ta sẽ trả lời bằng một lời chào giới thiệu bản thân. Hai bên sau đó sẽ lần lượt trao đổi. Không có thủ tục này cả hai bên có thể nói cùng một lúc khi kết nối được thiết lập và không một ai có thể nghe được người khác.

Tương tự như ví dụ trên ta có thể thấy rằng một thủ tục tốt sẽ có các đặc tính sau:

- |                          |  |
|--------------------------|--|
| - Được thiết lập trước   | : Thủ tục được thiết kế hoàn chỉnh trước khi nó được sử dụng.          |
| - Có sự thống nhất chung | : Một thành viên nhất trí tuân thủ các bước trong thủ tục theo thứ tự. |
| - Không nhập nhằng       | : Không một ai có thể thực hiện sai một bước do không hiểu nó.         |

- Hoàn chỉnh : Đối với mọi tình huống có thể xảy ra đều phải có một hành động được mô tả trước cần thực hiện.

Các thủ tục cũng được dùng trong liên lạc giữa máy tính với máy tính. Một máy tính cần phải biết khi nào “nói” và “nghe” với máy đang liên lạc với nó và liệu nó đã nhận đủ thông tin chưa... Hiển nhiên là cả hai máy tính phải tuân theo cùng một thủ tục.

### 5.1.2. Các loại thủ tục

Các nhiệm vụ nhất định, chẳng hạn như thoả thuận hợp đồng, bầu cử, phân phối thông tin và thậm chí là chơi bài đều là các hoạt động của con người. Tuy nhiên nhiều nhiệm vụ kiểu này phụ thuộc vào người làm chứng để đảm bảo vị công bằng. Liệu bạn có tin vào một người nói rằng anh ta sẽ xóc các quân bài mà không nhìn vào chúng và đưa cho bạn? Liệu bạn có tin vào một người nếu bạn không quen biết và nếu số tiền đặt cược cao?

Xã hội hiện đại đòi hỏi việc sử dụng máy tính và liên lạc như những công cụ thương mại. Nhiều người sử dụng máy tính không có sự quen biết cá nhân đối với người quản lý và những người sử dụng khác trong hệ thống. Trong nhiều trường hợp việc liên lạc máy tính được thực hiện trên những khoảng cách lớn. Do tính vô danh và do khoảng cách người dùng sẽ không tin vào các nhà quản lý và những người dùng khác trong hệ thống. Để sử dụng máy tính một cách hiệu quả ta phải phát triển các thủ tục mà nhờ chúng hai người đa nghi có thể giao tiếp với nhau và tin vào sự công bằng.

Hơn nữa để điều chỉnh hành vi, các thủ tục còn phục vụ cho một mục đích rất quan trọng khác là các thủ tục phải tách quá trình hoàn tất một nhiệm vụ khỏi cơ chế thực thi nó. Một thủ tục sẽ chỉ xác định các quy tắc của hành vi. Bằng cách này ta có thể kiểm tra một thủ tục để tin rằng nó đạt kết quả mong muốn. Ta sẽ kiểm tra tính đúng đắn của quá trình ở mức cao.

Sau khi đã tin vào tính đúng đắn của thiết kế ta có thể áp dụng thủ tục bằng cách dùng một cơ chế nào đó (tức là dùng một ngôn ngữ riêng nào đó hoặc một hệ thống mã hoá). Áp dụng phải tách khỏi thiết kế. Bởi vậy ta chỉ cần kiểm tra rằng cơ chế sẽ phải ảnh hưởng tới thiết kế đúng đắn, ta không cần kiểm tra lại rằng ứng dụng sẽ giải quyết vấn đề mà thủ tục được thiết kế cho nó. Hơn nữa, sau này ta có thể thay đổi ứng dụng mà không ảnh hưởng tới thiết kế. Việc tách rời thiết kế khỏi các ứng dụng là một ưu điểm quan trọng trong việc dùng các thủ tục.

### 5.1.3. Các thủ tục có trọng tài

Trọng tài là một bên thứ ba vô tư được tin cậy để hoàn tất một giao dịch giữa hai bên không tin cậy nhau. Nếu bạn bán một chiếc xe cho một người lạ và anh ta đưa cho bạn một tấm séc thì bạn không có cách nào để biết rằng tấm séc này có giá trị không. Bạn muốn gửi tấm séc này và giữ xe lại trong ít ngày cho tới khi bạn tin chắc rằng séc không có vấn đề gì. Một người mua đa nghi sẽ không chịu như vậy vì bạn lại có cả xe và séc và biết đâu bạn có thể chuồn khỏi thành phố với chúng?

Giải pháp ở đây là sử dụng một bên thứ ba được tin nhiệm chẳng hạn một chủ nhà băng hoặc một luật sư làm trọng tài. Bạn trao chứng nhận sở hữu xe và chìa khoá xe cho trọng tài và người mua xe đưa séc cho trọng tài. Bạn có một sự đồng ý tay ba vào thời điểm séc được

xác nhận. Trọng tài sẽ gửi séc vào tài khoản của bạn. Nếu trong một thời hạn xác định séc được xác nhận thì trọng tài sẽ chuyển xe của bạn cho người mua. Nếu séc không được xác nhận thì trọng tài sẽ trả xe về cho bạn. Trong một thủ tục máy tính, trọng tài là một bên thứ ba tin cậy đủ đảm bảo sự công bằng. Trọng tài có thể là một người, một chương trình hoặc một máy tính. Ví dụ trong một mạng, trọng tài có thể là một chương trình chạy trên một máy trong mạng. Chương trình sẽ nhận và gửi các thông báo giữa các người dùng. Người dùng tin rằng khi trọng tài gửi tới một thông báo nói rằng nó tới từ A thì thông báo thực sự được gửi từ người dùng A. Khái niệm trọng tài là một khái niệm cơ bản đối với loại thủ tục an toàn được gọi là thủ tục có trọng tài.

Các thủ tục máy tính có trọng tài có một số nhược điểm:

- Hai bên không có khả năng tìm một bên thứ ba vô tư mà cả hai đều tin tưởng. Những người dùng đa nghi có lý khi họ không tin vào một trọng tài không được biết trên mạng.
- Việc duy trì hoạt động của trọng tài sẽ làm tăng chi phí cho những người sử dụng hoặc mạng chi phí này có thể rất lớn.
- Trọng tài sẽ gây nên thời gian giữ chậm khi liên lạc vì bên thứ ba phải thu, xem xét và rồi mới gửi đi đối với mỗi giao dịch.
- Nếu dịch vụ trọng tài được dùng quá nặng nề thì nó có thể trở thành một “nút cổ chai” trong mạng vì nhiều người dùng đều muốn có trọng tài riêng
- Tính bí mật trở nên dễ bị tổn thương vì trọng tài phải truy nhập tới nhiều thông tin nhạy cảm.

Vì những lý do trên mà người ta thường tránh dùng thủ tục này nếu có thể.

#### **5.1.4. Các thủ tục có phán xét**

Tương tự như trọng tài là ý tưởng sử dụng quan toà. Quan toà là một bên thứ ba có thể phán xét liệu một giao dịch có được thực hiện một cách công bằng hay không. Ví dụ công chứng viên là một bên thứ ba vô tư được tin cậy sẽ chứng thực rằng một tài liệu đã được ký một cách tự nguyện và xác nhận rằng anh ta có đầy đủ lý do để xác định rằng người ký là có thẩm quyền. Chữ ký của công chứng viên thường cần thiết đối với những tư liệu hợp lệ mà tính xác thực của nó sau này có thể bị nghi ngờ. Công chứng viên không thêm một tí gì vào giao dịch ngoài việc là một người làm chứng - một người sau này có thể kiểm tra khi có sự nghi ngờ.

Một số thủ tục máy tính sử dụng một kiểu tương tự công chứng viên để xây dựng bằng chứng công bằng. Với một thủ tục có khả năng phán xét, số liệu đủ là cần thiết để bên thứ ba vô tư phán xét được tính công bằng dựa trên bằng chứng. Bên thứ ba không chỉ có thể xác định liệu hai bên tranh chấp có sử dụng đúng không (tức là nằm trong các quy tắc của thủ tục) mà còn có thể xác định được ai là người gian lận.

Các thủ tục có phán xét sẽ xoay quanh các dịch vụ của bên thứ ba chỉ trong trường hợp có tranh chấp. Bởi vậy các thủ tục này có chi phí thấp hơn các thủ tục có trọng tài. Tuy nhiên chúng chỉ xác định được sai sót sau khi sai sót xảy ra.

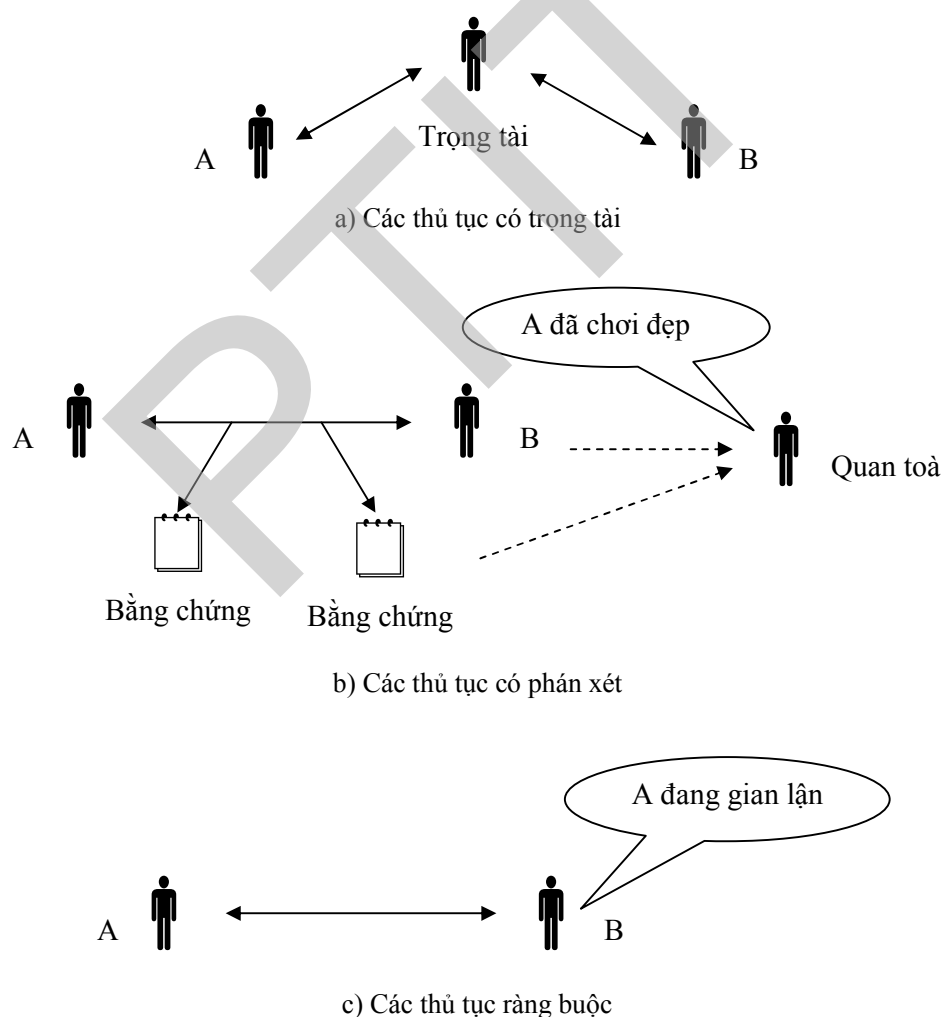
### 5.1.5. Các thủ tục tự ràng buộc

Một thủ tục tự ràng buộc là một thủ tục đảm bảo sự công bằng. Nếu một bên nào đó cố gắng gian lận thì điều này phải được bên kia thấy rõ. Không cần bất cứ một người ngoài nào để đảm bảo sự công bằng. Hiển nhiên là các thủ tục tự ràng buộc được ưu thích hơn cả. Tuy nhiên, không có một thủ tục tự ràng buộc đối với mọi tình huống.

Như vậy có ba mức các thủ tục:

- *Các thủ tục có trọng tài* trong đó một bên thứ ba tin cậy sẽ tham gia vào mỗi giao dịch để đảm bảo rằng cả hai bên đều sử dụng đúng đắn.
- *Các thủ tục có sự phán xét* trong đó một bên thứ ba có thể phán xét liệu cả hai bên có sử dụng đúng đắn hay không và nếu không thì bên nào là gian lận.
- *Các thủ tục tự ràng buộc* trong đó những mưu toan gian lận của một bên nào đó sẽ bị bên khác hiện.

Các loại thủ tục này được mô tả ở hình 5.1.



Hình 5.1. Các loại thủ tục

Sau đây ta sẽ hướng mỗi quan tâm vào việc sử dụng các thủ tục để giải quyết các bài toán trong an toàn mạng máy tính.

## 5.2. CÁC THỦ TỤC ĐỂ GIẢI QUYẾT CÁC VẤN ĐỀ

Bây giờ ta sẽ nghiên cứu việc sử dụng các thủ tục trên những bài toán thực tế. Rõ ràng là khi hai người tiếp xúc trực tiếp họ sẽ làm khác với khi có một máy tính giữa họ. Mặc dù có thể có những khác biệt về quan điểm nhưng chúng tôi vẫn muốn đưa ra những thủ tục cho một môi trường tự động mà nhờ nó con người có thể tiến hành các công việc hàng ngày như ký các hợp đồng, chi trả các hoá đơn, bỏ phiếu bầu cũng an toàn như các giao tiếp giữa người với người. Ta cũng nghiên cứu một số vấn đề mà có các thủ tục an toàn cho chúng.

### 5.2.1. Phân phối khoá

Thay đổi khoá mã hoá là một vấn đề quan trọng nhưng không dễ. Có thể thấy rằng thám mã sẽ càng có cơ may thành công nếu có càng nhiều bản mã. Bản mã từ các khoá khác nhau có thể giúp cho việc xác định cấu trúc của một thuật toán bí mật hoặc các khoá bí mật nhưng bản mã từ cùng “một khoá” lại giúp cho việc tìm giá trị của khoá. Như vậy, trên thực tế cần thay đổi khoá một cách định kỳ để đảm bảo lượng bản mã tạo từ một khoá bất kỳ không giúp ích đáng kể cho thám mã.

Tuy nhiên ta hãy xem xét một việc sử dụng mã hoá cho các chữ ký số trên các hợp đồng. Nếu bạn đưa ra hoặc thu nhận một hợp đồng đã được ký bằng chữ ký số thì bạn muốn giữ hợp đồng làm bằng chứng cho tới khi mọi tranh chấp có thể có đã được giải quyết. Điều này có thể là phải nhiều năm sau khi hợp đồng đã được thực hiện. Người gửi và người nhận cần phải giữ mọi khoá cần thiết để kiểm chứng giá trị các chữ ký của hợp đồng. Nếu người gửi thay đổi các khoá mã hoá hàng ngày hoặc thậm chí là hàng tháng thì vẫn phải lưu giữ hàng trăm hoặc hàng nghìn khoá. Và những khoá này phải được lưu giữ một cách an toàn. Điều này cho thấy không nên thay đổi khoá thường xuyên hơn mức cần thiết. Mặt khác, nếu một khoá bị mất hoặc bị lộ thì càng dùng ít khoá nguy cơ này càng ít. Bởi vậy việc chọn một tần suất thay đổi khoá thích hợp không phải là một công việc dễ dàng.

Một số thủ tục đã được phát triển cho việc phân phối khoá. Các thủ tục này phụ thuộc vào mức độ chia sẻ thông tin giữa người gửi và người nhận.

#### 5.2.1.1. Trao đổi khoá đối xứng không có Server (máy chủ)

Giả sử rằng hai người sử dụng đều có một bản sao của khoá mã hoá đối xứng (bí mật)  $K$  mà chỉ có họ biết. Khi lượng thông báo không lớn lắm hoặc nếu không có nguy cơ xâm nhập đáng kể thì họ có thể dùng  $K$  để trao đổi các thông báo.

Tuy nhiên nếu hai người muốn an toàn hơn họ có thể đồng ý thay đổi khoá thường kỳ thậm chí là dùng khoá khác nhau đối với mỗi thông báo. Để làm được điều này một trong hai người có thể tạo một khoá mới (được gọi là  $K_{\text{new}}$ ) rồi mã hoá nó bằng  $K$  và gửi  $E(K_{\text{new}}, K)$  tới người còn lại. Trong một số hệ mật  $K$  được gọi là khoá chủ hay mã hoá khoá và  $K_{\text{new}}$  được gọi là khoá phiên hay khoá lưu chuyển.

Nhược điểm của phương pháp này là hai người dùng phải cùng chia sẻ một khoá duy nhất của họ. Các cặp người dùng khoá cũng cần các khoá duy nhất và nói chung  $n$  người dùng phải cần tới  $n(n+1)/2$  khoá.

### 5.2.1.2. Trao đổi khoá đối xứng có Server

Một phương pháp khác là sử dụng dịch vụ phân phối khoá trọng tâm cho 2 người dùng. Ở thủ tục này số khoá sẽ giảm nhưng tính mềm dẻo của giải pháp cũng giảm đi một mức nào đó.

Giả sử Pablo và Ronê muốn có một khoá bí mật dùng để trao đổi các thông báo nhưng họ không có khoá chung. Tuy nhiên ta cũng coi rằng có một kho chứa khoá trung tâm sao cho Pablo và kho này có một khoá chung  $K_P$ , Ronê và kho có một khoá chung khác  $K_R$ . Như đã nêu trong [8], trước tiên Pablo sẽ gửi tới kho  $(P, R, I_P)$  báo danh tính của mình  $(P)$ , danh tính của người nhận  $(R)$  và một định danh  $I_P$  để đánh dấu kết quả, kho sẽ gửi trở lại thông tin của Pablo (giả sử rằng Pablo có thể gửi nhiều yêu cầu tới kho và cần một định danh để phân biệt) và để tránh các tấn công theo kiểu phát lại (các tấn công dạng này có thể buộc Pablo và Ronê dùng lại khoá trước). Thông tin này không cần phải mã hoá vì:

- Nếu thông báo được đọc bởi một người ngoài (Chẳng hạn Octavia) thì việc biết rằng Pablo và Ronê muốn trao đổi thông báo riêng cũng chẳng gây tổn hại gì.
- Nếu thông báo bị thu chặn và bị sửa đổi thì trường hợp xấu nhất có thể xảy ra là yêu cầu sẽ bị thay đổi từ yêu cầu Pablo/ Ronê sang yêu cầu Pablo/ Octavia, vì vậy sự thay đổi này sẽ được phát hiện ngay ở bước sau.
- Nếu thông báo bị Octavia thu chặn và không phát đi thì Pablo sẽ nhận thấy không có trả lời và sẽ có hành động đối phó.
- Nếu thông báo bị Octavia thu chặn và giả mạo tạo một câu trả lời thì Pablo sẽ nhận thấy rằng câu trả lời không được mã hoá đúng ở bước tiếp.

Trung tâm phân phối khoá sẽ tạo một khoá mã hoá mới cho Pablo và Ronê để sử dụng (ta gọi là  $K_{PR}$ ). Trung tâm phân phối sẽ gửi cho Pablo.

$$E\left(\left(I_P, R, K_{PR}, E\left((K_{PR}, P), K_R\right)\right), K_P\right) \quad (5.1)$$

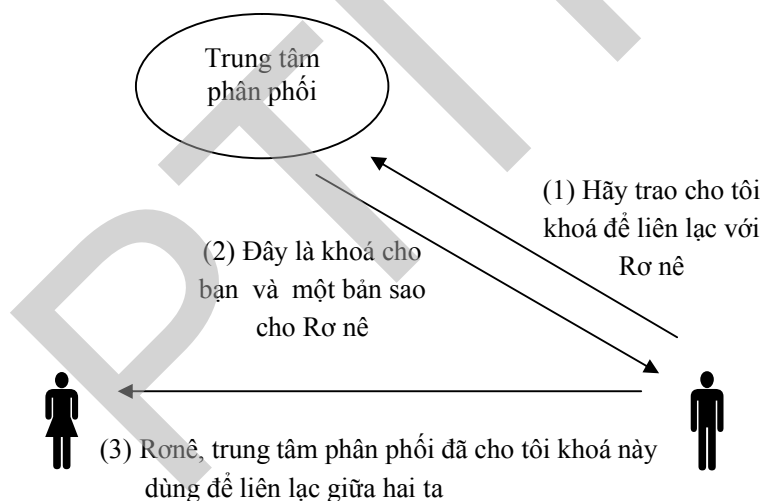
- Trong phần giao dịch này Pablo sẽ nhận 4 tin sau được mã hoá bằng  $K_P$  :
- Định danh thông báo  $I_P$ .
- Tính danh của Ronê  $R$ .
- Khoá liên lạc  $K_{PR}$ .
- Một xâu bao gồm tính danh của anh ta và cũng khoá đó được mã bằng khoá của trung tâm phân phối chung với Ronê,  $E\left((K_{PR}, P), K_R\right)$ .



Pablo không thể giải mã thông tin cuối cùng này nhưng anh ta có thể gửi nó cho Ronê. Một lần nữa, các tấn công của Octavia cũng không thành công vì:

- Nếu Octavia thay R bằng O trong thông báo gốc của Pablo để thay đổi tính danh của mình vào vị trí của Ronê thì trong trả lời từ trung tâm phân phối Pablo sẽ nhận thấy rằng tính danh của Ronê đã bị thay đổi thành tính danh của Octavia.
- Nếu Octavia thay P bằng O trong thông báo gốc của Pablo để thay tính danh của mình vào vị trí tính danh của Pablo. Khi đó Octavia (chứ không phải là Pablo) sẽ thu được trả lời, còn về phần mình Ronê sẽ xác định rằng Octavia (chứ không phải là Pablo) là người muốn liên lạc với mình (qua thông báo được mã bằng khoá chỉ có trung tâm phân phối và Ronê được biết).
- Nếu Octavia thu chặn thông báo từ trung tâm phân phối thì toàn bộ thông báo được mã bằng khoá  $K_P$  nên Octavia không thể biết được và cũng không thể thay đổi nó một cách logic.

Cuối cùng Pablo sẽ gửi cho Ronê  $E((K_{PR}, P), K_R)$ , khoá được xác nhận và được ký bởi trung tâm phân phối là khoá dùng để liên lạc với Pablo. Thủ tục này được chỉ ra trên hình 5.2.



Hình 5.2. Phân phối khoá qua trung tâm phân phối khoá

Thủ tục được đưa ra ở đây đòi hỏi phải luôn có trung tâm phân phối khoá. Việc thêm vào trung tâm này có thể gây ra hiện tượng tắc nghẽn (do nhiều người yêu cầu khoá), nó cũng là một mục tiêu rất hấp dẫn để qua mặt tại sao Octavia lại phải mất thời gian để thu chặn và thay đổi các thông báo riêng nếu như cô ta có thể qua mặt, giả mạo hoặc làm mất hiệu lực của trung tâm phân phối.

Tuy nhiên sơ đồ này tốt khi mở rộng vì khi thêm một người dùng mới chỉ cần thêm một khoá chung với trung tâm phân phối khoá. Hơn nữa, người dùng có thể thay đổi khoá thường xuyên theo mong muốn bằng cách đưa ra yêu cầu khoá mới cho trung tâm phân phối. Thủ tục này cũng thích hợp khi Pablo và Ronê không cần trao đổi liên lạc trước nếu họ đã được ghi nhận ở trung tâm phân phối.

### 5.2.1.3. Trao đổi khoá không đối xứng không có Server

Sử dụng mã hoá không đối xứng (khóa công khai) sẽ làm giảm nhu cầu đối với các khóa riêng và cũng không làm giảm khả năng dễ bị tấn công của kho chứa trung tâm. Giả sử Pablo và Ronê muốn trao đổi một thông báo, mỗi người đều có một cặp khóa riêng và khóa công khai và mỗi người đều có thể truy nhập được vào khóa công khai của người khác. Ta ký hiệu các khóa riêng và khóa công khai của Pablo là  $E_P$  và  $D_P$ , các khóa tương ứng này của Ronê là  $E_R$  và  $D_R$ .

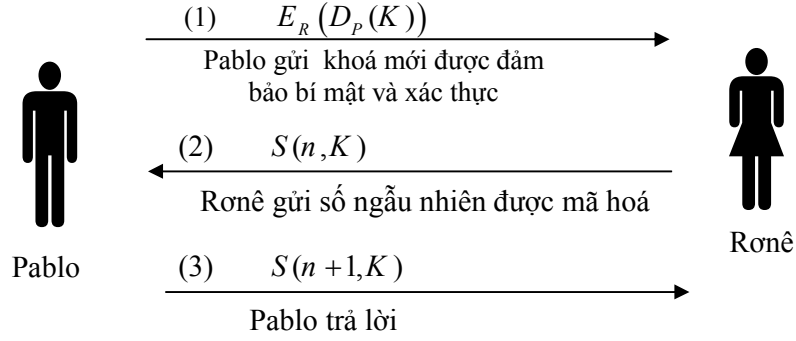
Pablo có thể gửi trực tiếp  $E_R(M)$  tới Ronê. Tuy nhiên các thuật toán mã hoá công khai điển hình thực hiện chậm hơn hẳn so với các thuật toán mã hoá đối xứng (thường kém tới một vài bậc). Bởi vậy, trong khi bản thân việc mã hoá là một gánh nặng đối với tốc độ liên lạc thì mã hoá khóa công khai lại là một gánh nặng tới mức không thể chấp nhận được. Vì lý do đó, Pablo và Ronê có thể không muốn sử dụng thuật toán khóa công khai để bảo vệ tổng thể quá trình liên lạc.

Pablo có thể chọn một khóa mã hoá  $K_{PR}$  của một thuật toán đối xứng  $S$  để sử dụng với Ronê. Nếu Pablo gửi  $E_R(K_{PR})$  tới Ronê thì chỉ có Ronê mới có thể đọc được nó vì chỉ có Ronê mới có khóa riêng  $D_R$  để giải ra khóa. Hơn nữa việc sử dụng mã hoá khóa công khai cho công việc này chỉ trong một thời gian ngắn không ảnh hưởng nhiều tới việc sử dụng các tài nguyên tính toán. Tuy nhiên Ronê không thể đảm bảo được rằng khóa này thực sự tới từ Pablo ngay cả khi anh ta gửi  $E_R(P, K_{PR})$ . Những thông báo này có thể bị giả mạo bởi Octavia.

Pablo có thể gửi  $E_R(D_P(K_{PR}))$  tới Ronê. Chỉ có Ronê mới có thể giải mã được lớp ngoài, bởi vậy Ronê được đảm bảo về tính bí mật của thông báo. Tương tự, chỉ có Pablo mới có thể áp dụng mã hoá lớp trong ( $D_P(\quad)$ ), do đó Ronê được đảm bảo về tính xác thực của nó. Bởi vậy thông báo này sẽ chuyển một khóa xác thực và tin cậy.

Tuy nhiên bằng việc trao đổi khóa đối xứng hai bên cũng trao đổi một thông báo kiểm tra để chứng tỏ rằng mỗi bên đã nhận đúng một khóa mới (tức là không phải một khóa được dùng lại). Sau khi thu khóa của Pablo, Ronê phải gửi một số ngẫu nhiên đã mã  $n$ , Pablo sẽ giải mã số này và để chứng tỏ rằng mình đã làm việc đó anh ta sẽ gửi trả lại bản mã của số  $(n+1)$ . Thủ tục này được mô tả trên hình 5.3.

Thủ tục này trả lời cho câu hỏi Pablo và Ronê phải làm thế nào để nhận được các khóa công khai của nhau một cách bí mật. Ta sẽ xét câu trả lời này trong trường hợp có trung tâm phân phối khóa.

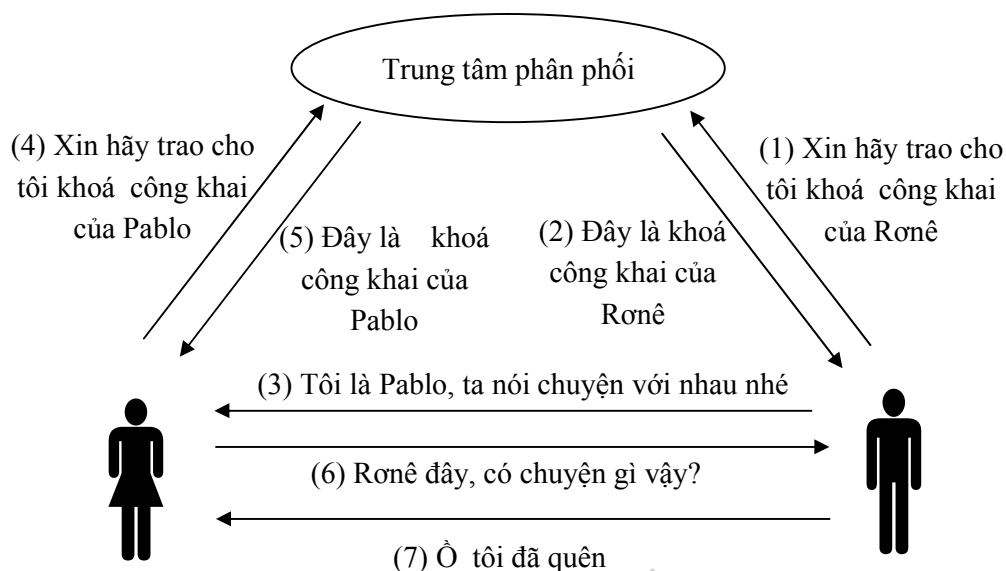


Hình 5.3. Thủ tục trao đổi khoá đối xứng

#### 5.2.1.4. Trao đổi khoá không đối xứng có Server

Thủ tục này là sự mở rộng trực tiếp của thủ tục sử dụng trung tâm phân phối khoá đối xứng. Pablo sẽ gửi một thông báo tới trung tâm phân phối nói rằng anh ta muốn liên lạc với Ronê ( $P/R$ ). Như đã nói ở trên, thông báo này có thể sử dụng ở dạng rõ. Trung tâm phân phối sẽ trả lời bằng  $D_D(E_R, R)$  (khoá công khai của Ronê và tính danh của cô ta). Thông báo này được mã bằng khoá riêng của trung tâm phân phối. Không có bí mật nào chứa trong thông báo này ngoài tính toàn vẹn. Bởi vậy, bất cứ ai có khoá công khai của trung tâm phân phối đều có thể đọc nó nhưng chỉ có trung tâm phân phối mới có thể tạo ra được thông báo này. Pablo bây giờ đã có khoá công khai của Ronê và có thể bắt đầu liên lạc trực tiếp với Ronê nhưng thủ tục chưa kết thúc. Nếu Pablo gửi một thông báo bằng khoá công khai của Ronê thì anh ta tin rằng chỉ có Ronê mới có thể đọc được nó. Tuy nhiên Ronê không có khoá dùng để trả lời ngay cả khi Pablo gửi khoá công khai của mình kèm theo thông báo, Ronê chẳng có lý do nào để tin rằng đó là khoá của Pablo chứ không phải là của Octavia.

Pablo sẽ gửi  $E_R(P, I_P)$  tới Ronê chứa tính danh của anh ta và một thông tin phụ dùng cho thông báo trả lời của Ronê. Bây giờ Ronê sẽ liên lạc với trung tâm phân phối theo cách mã Pablo đã làm để nhận được khoá của anh ta. Ronê gửi  $(R, P)$  và nhận trở lại  $D_D(E_R, R)$ . Cô ta gửi cho Pablo  $E_P(I_P, I_R)$  để cho Pablo biết rằng cô ta đã thu được thông báo của ta, đã nhận được khoá công khai của anh ta và đang sẵn sàng liên lạc.  $I_P$  báo cho Pablo biết rằng thông báo này tới từ Ronê (bởi vì anh ta đã gửi nó cho Ronê sau khi đã mã nó bằng mã công khai của Ronê). Pablo sẽ hoàn tất thủ tục bằng việc gửi cho Rone thông báo  $M$  và gửi  $I_R$  để chứng tỏ rằng thông báo này thật sự là từ Pablo, tức là nó có sau khi cô ta đã có khoá công khai của Pablo nhận từ trung tâm. Tóm lại, Pablo sẽ gửi cho Ronê  $E_R(M, I_R)$  (Thông thường  $M$  là một khoá mã hoá đối xứng cho cả hai người sử dụng. Nếu không, họ có thể tiếp tục gửi các tính danh  $I$  để tiếp tục đảm bảo tính xác thực). Thủ tục này có 7 bước như được mô tả trên hình 5.4.



Hình 5.4. Thủ tục trao đổi khoá không đối

Trung tâm phân phối làm thế nào để có được các khoá? Trung tâm phân phối có thể công bố khoá công khai của mình một cách rộng rãi và bất kỳ ai muốn ghi nhận vào trung tâm này chỉ cần đưa khoá và tính danh đã được mã bằng khoá của trung tâm. Do hiệu năng, độ tin cậy và kích cỡ mà có thể có nhiều trung tâm phân phối. Ở các bước 2 và 5, nếu trung tâm phân phối không có các khoá được yêu cầu thì nó có thể dàn xếp với các trung tâm khác để thu nhận và phân phối lại các khoá. Người dùng không cần biết quá trình thoả thuận này. Hai hoặc nhiều trung tâm có thể dùng như bộ phận dự phòng của trung tâm khác để nếu một trung tâm bị quá tải hoặc hỏng thì trung tâm khác có thể cung cấp các giá trị khoá như vậy. Cũng như vậy, một khi Pablo và Roné có khoá công khai của nhau thì họ có thể tự lưu trữ khoá này, chỉ khi một bên muốn thay đổi khoá công khai thì mới phải yêu cầu tới trung tâm phân phối.

Trung tâm phân phối khoá là một khâu quyết định trong thủ tục: nó phải hoạt động bất cứ khi nào có nhu cầu về khoá. Tuy nhiên, cần chú ý là trung tâm phân phối sẽ gửi cho Roné  $D_D(E_P, P)$  bao gồm khoá và tính danh của Pablo để đảm bảo rằng khoá này thực sự tới từ trung tâm phân phối (bởi vì đây là kết quả được mã hoá bằng  $D_D$  là khoá riêng của trung tâm phân phối). Pablo có thể yêu cầu khoá riêng của mình ở bất cứ thời điểm thích hợp nào và sẽ nhận được kết quả tương tự từ trung tâm phân phối. Sau đó anh ta có thể chuyển các bản sao của kiểu xác thực này đối với khoá của mình tới bất cứ ai mà anh ta muốn thiết lập liên lạc. Theo cách này, tính khả dụng liên tục của trung tâm phân phối không còn quan trọng nữa.

Câu hỏi còn lại là điều gì làm cho ta tin rằng các khoá là xác thực, tức là các khoá này là đúng của người có tính danh tương ứng. Liệu Octavia có thể đưa ra khoá công khai của mình mà tuyên bố rằng đó là khoá của Roné không? Điều cần nói ở đây là ta phải làm thế nào để ràng buộc một cách chắc chắn tính danh và khoá công khai theo một cách thực sự tin cậy? Câu hỏi này được gọi là chứng thực.

#### 5.2.1.5. Chứng thực

Cũng như mọi người chúng ta sẽ thiết lập sự tin cậy ở mọi lúc. Với con người ta thường xác định họ qua các đặc điểm như giọng nói, khuôn mặt hoặc chữ viết. Tuy nhiên, đôi khi ta

trả lời điện thoại và nghe thấy những câu đại loại: "Tôi là đại diện của chính quyền địa phương..." hoặc "Tôi thay mặt tổ chức từ thiện...". Tùy theo nội dung của cuộc gọi mà ta quyết định liệu có nên tin vào người gọi hay phải tìm một cách kiểm tra độc lập chẳng hạn như nhờ cảnh sát kiểm tra số máy đã gọi đến trong danh bạ điện thoại. Ta cũng có thể tìm kiếm thông tin phụ từ người gọi: ("Cô ta mặc áo màu gì"). Ta có thể hành động để ngăn chặn kẻ lợi dụng ("Tôi sẽ gửi séc trực tiếp tới tổ chức từ thiện của bạn"). Tuy nhiên đối với những công việc giao tiếp kiểu này ta đều có một ngưỡng tin cậy nào đó, một mức độ để ta có thể sẵn lòng tin tưởng vào một người chưa được biết.

Đối với liên lạc điện tử ta cần phải xây dựng những phương pháp để hai bên thiết lập được sự tin cậy mà không phải gặp gỡ. Trong thương mại rất cần những ứng dụng này. Công ty chế tạo máy Acorn gửi tới cho công ty thép Big một đơn đặt hàng 10.000 tấm thép cần được đóng hàng xong trong một tuần và sẽ trả tiền trong vòng 10 ngày. Đơn đặt hàng được in theo mẫu của Acorn và được ký bởi một người có tên là Helen Smudge - đại diện thương mại. Big có thể bắt đầu chuẩn bị thép. Big có thể kiểm tra mức độ tài khoản của Acorn, để quyết định xem có nên gửi hàng mà không được trả tiền trước hay không. Nếu nghi ngờ, Big có thể điện thoại cho Acorn và yêu cầu nói chuyện với bà Smudge ở bộ phận thương mại.

Tuy nhiên thường là Big sẽ đóng hàng mà không cần biết ai sẽ là bà Smudge, liệu bà ta có phải là đại diện thương mại không và liệu bà ta có thẩm quyền để đặt những hợp đồng có tầm cỡ như vậy hay không. Đôi khi những giao dịch kiểu này được thực hiện qua Fax nên Big thậm chí cũng không có cả chữ ký gốc trên tài liệu. Các giao dịch kiểu này thường xuyên xảy ra. Ở đây sự tin cậy được dựa trên tính xác thực (khuôn dạng bản in, chữ ký), thông tin phụ (báo cáo tài chính) và mức độ khẩn cấp (Acorn yêu cầu có thép nhanh).

Tất cả chúng ta đã từng cho bạn bè vay một thứ gì đó. Giả sử bạn gặp ở một bữa tiệc một người nói rằng cô ta là bạn của một người bạn và cô ta yêu cầu bạn vay tiền. Bạn trao đổi với cô ta một số chi tiết và bạn có thể hài lòng biết rằng cô ta có biết đôi chút về người bạn này. Dựa trên mối quan hệ đó bạn có thể quyết định cho cô ta vay tiền. Sự tin tưởng ở đây là dựa vào vẻ ngoài đáng tin cậy của cô ta và sự quen biết của cô ta với người bạn của bạn. Câu chuyện tương tự có thể tiếp tục với một số người bạn của một người bạn ("Tôi sống cạnh Anand và anh ta thường nói về Clara là người láng giềng của anh, anh có biết cô ấy không?"). Bạn có thể cho một người bạn của một người bạn vay tiền nếu bạn cảm thấy đủ tin cậy với chuỗi tình bạn này. Mối quan hệ "bạn của" không dễ dàng chuyển bắc cầu như vậy.

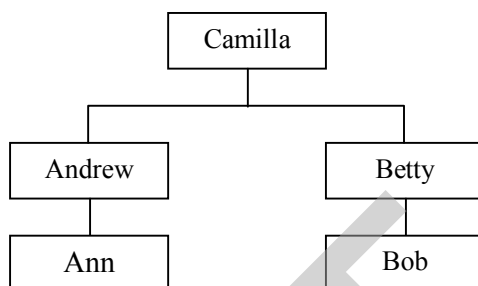
Tuy nhiên, trong thực tế có thể thấy được một số điều đảm bảo trong các ví dụ trên. Cảnh sát sẽ đảm bảo tính xác thực của người gọi. Acorn sẽ đảm bảo rằng bà Smudge là đại diện thương mại của họ (bằng cách chuyển cuộc gọi tới bà ta ở bộ phận thương mại) và theo một nghĩa nào đó, công ty điện thoại sẽ đảm bảo tính xác thực của bộ phận cảnh sát hoặc của Acorn qua danh sách của họ trong danh bạ điện thoại. Những "đảm bảo" này có thể được dùng là cơ sở để tin cậy trong các quan hệ thương mại mà hai bên không biết nhau.

Một công ty lớn có thể có một vài bộ phận, mỗi bộ phận có thể có một số phòng, mỗi phòng có thể có một vài đề án và mỗi một đề án có thể có một số nhóm tác nghiệp (với những khác biệt về tên, số mức và cấp độ hoàn chỉnh của cấu trúc). Người điều hành cao nhất có thể không biết tên và mặt của mỗi nhân viên trong công ty nhưng trường nhóm tác nghiệp phải

biết tất cả các thành viên trong nhóm mình và lãnh đạo dự án phải biết tất cả các trưởng nhóm tác nghiệp...

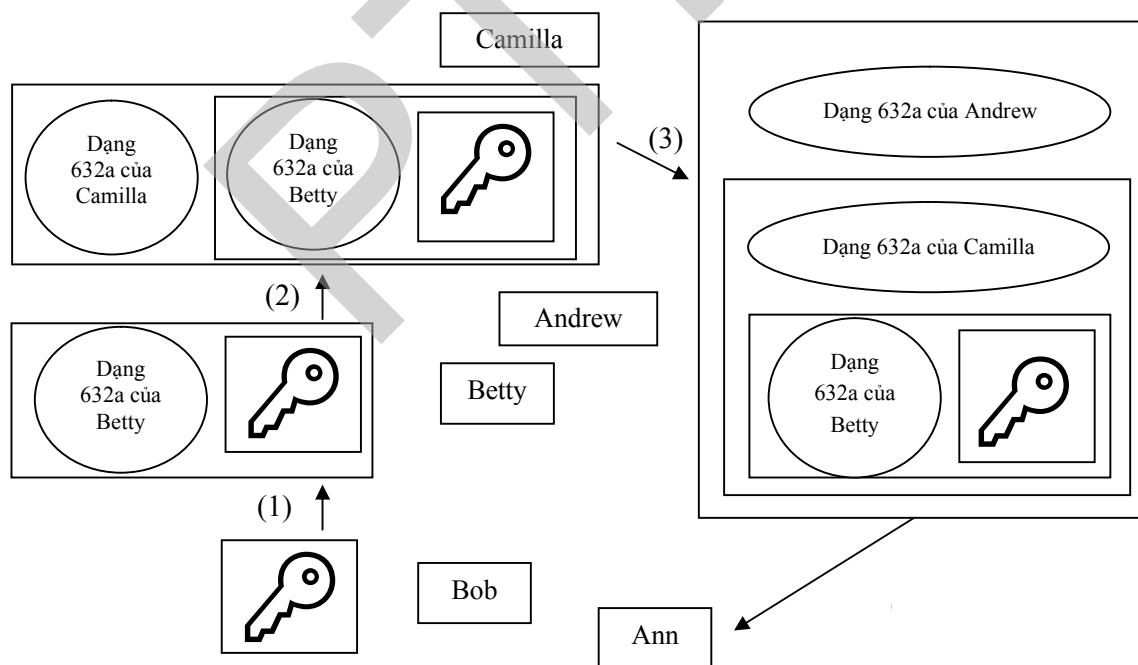
Xem xét cơ cấu tổ chức của một công ty có thể cho ta thấy sự tin cậy được xây dựng ra sao. Giả sử Ann và Bob gặp nhau. Bob nói rằng anh ta cũng làm trong công ty với Ann.

Trong công ty của họ Andrew và Betty là hai trưởng nhóm tác nghiệp trong cùng một dự án mà Camilla là giám đốc và Ann làm việc cho Andrew, Bob làm việc cho Betty. Các yếu tố này có thể giúp cho Ann và Bob tin cậy nhau. (Các quan hệ về mặt tổ chức được chỉ ra ở hình 5.5).



Hình 5.5. Tổ chức trong một công ty giả định

Ann sẽ hỏi Andrew: Bob là ai, Andrew hoặc hỏi Betty nếu anh ta biết Betty, nếu không anh ta sẽ hỏi Camilla, (Camilla hỏi Betty), Betty trả lời Bob làm việc cho mình. Câu trả lời này sẽ đi theo con đường ngược với các câu hỏi. Tuy nhiên nếu Bob nằm trong tác nghiệp khác thì có thể cần phải leo cao hơn theo cây tổ chức này cho tới khi tìm thấy điểm chung.



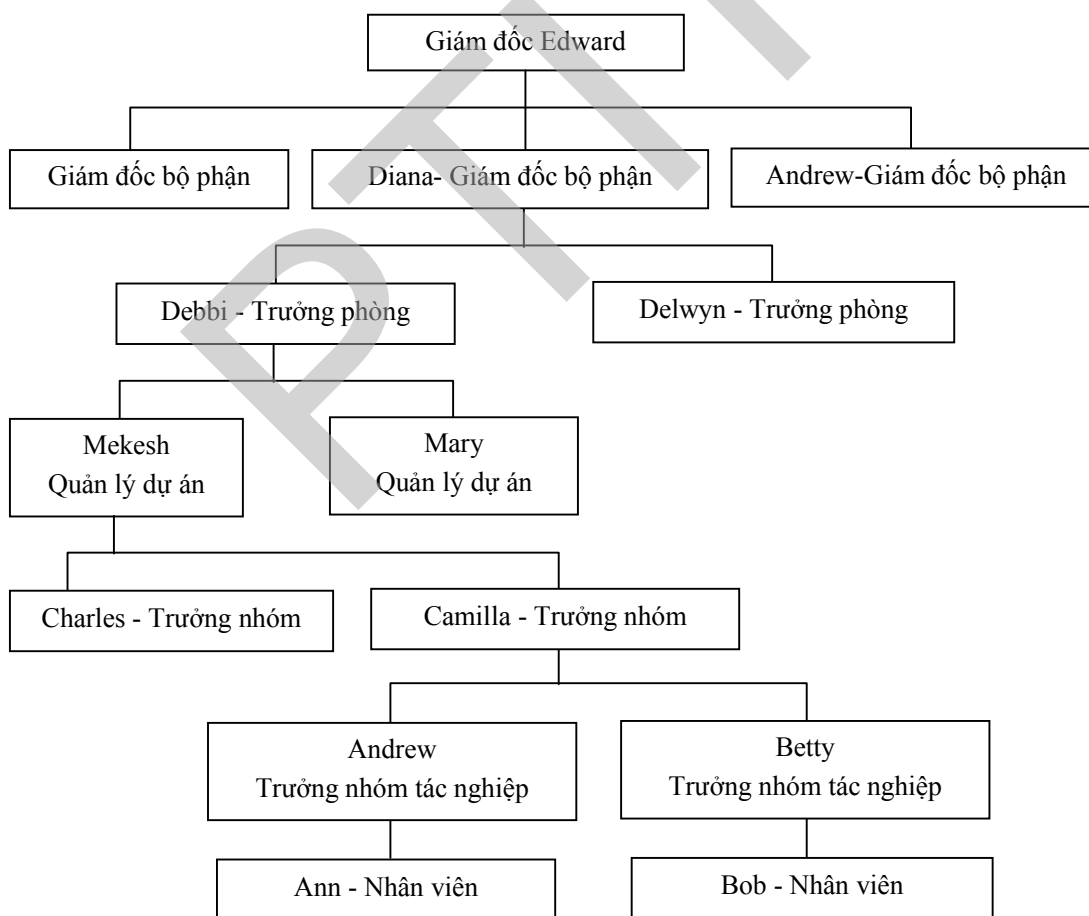
Hình 5.6. Bob chuyển khoá cho Ann

Ta có thể sử dụng một quá trình tương tự để trao đổi khoá mật mã. Nếu Bob và Ann muốn liên lạc, Bob có thể trao khoá công khai của mình cho Betty là người sẽ chuyển nó cho Camilla hoặc trao trực tiếp cho Andrew - là người trao nó cho Ann. Tuy nhiên cách thức trong thực tế không diễn ra đúng như vậy, khoá phải được gắn thêm một ghi chú nói rằng nó là từ

Bob ở dạng thông tin trong danh bạ điện thoại hoặc ở dạng 947 của Hồ sơ cá nhân (tương tự như mẫu 2a/TCTW Hồ sơ công chức Việt Nam). Và nếu dạng 947 được sử dụng thì Betty phải gắn thêm thông tin dạng 632a. Camilla cũng sẽ gắn thêm một thông tin dạng 632a khác và cuối cùng là Andrew cũng làm như vậy (Xem hình 5.6).

Chuỗi các thông tin dạng 632a này muốn nói rằng "Tôi là Betty và tôi đã nhận khoá này và có gắn kèm hồ sơ cá nhân của một người mà tôi biết là Bob". "Tôi là Camilla và tôi đã nhận khoá này có gắn kèm hồ sơ cá nhân và trích lục dạng 632a từ một người mà tôi biết là Betty" và cứ như vậy. Khi Ann nhận được khoá cô ta có thể xem lại chuỗi bằng chứng này và kết luận với một đảm bảo hợp lý là khoá này thực sự đã tới từ Bob. Thủ tục này là một cách để thu được một khoá công khai có chứng thực (kết hợp giữa khoá và một tính danh tin cậy). Mô hình này làm việc tốt trong một công ty vì luôn có một người nào đó là chung đối với hai nhân viên bất kỳ, thậm chí nếu hai nhân viên nằm ở các bộ phận khác nhau tới mức mà người chung là Giám đốc công ty.

Tuy nhiên quá trình này sẽ gặp trở ngại do mỗi khi Ann và Bob muốn liên lạc thì cả Ann, Andrew, Camilla, Betty và Bob đều phải tham gia. Nếu Betty đi công tác hoặc Andrew bị ốm thì thủ tục sẽ chệch choạc. Thủ tục này cũng không làm việc tốt nếu giám đốc không thể làm được bất cứ một việc quan trọng nào do phải mất thời gian vào việc tạo các biểu 632a.



Hình 5.7. Cấu trúc công ty mở rộng

Để khắc phục vấn đề thứ nhất Bob có thể yêu cầu được cung cấp đầy đủ các mẫu biểu 632a từ giám đốc xuống anh ta. Sau đó Bob có thể trao bản sao của chúng tới bất kỳ ai trong

công ty cần tới khoá của mình. Thay cho phải làm từ đáy tới đỉnh chung, Bob sẽ bắt đầu từ đỉnh và nhận chuỗi đầy đủ các mẫu biểu của mình. Anh ta sẽ nhận các chữ ký cần thiết vào bất cứ lúc nào mà các cấp trên của anh ta rồi việc, như vậy họ không nhất thiết phải có mặt khi anh ta muốn trao khoá công khai đã được chứng thực của mình. Thay cho việc bắt đầu từ đáy (từ các nhân viên tác nghiệp) và làm việc dần tới đỉnh của cây (Giám đốc) ta sẽ bắt đầu từ đỉnh. Như vậy, Bob sẽ có một khoá đã được chứng thực trước để có thể sử dụng được không hạn chế trong tương lai. Ta sẽ mở rộng cấu trúc của công ty giả định từ giám đốc tới các mức (Xem hình 5.7).

Giám đốc công ty sẽ viết một bức thư cho mỗi giám đốc bộ phận nói rằng "Tôi là Edward - chủ tịch công ty. Tôi chứng nhận cho Diana là giám đốc bộ phận là người mà tôi biết rõ và tôi trao cho Diana quyền chứng nhận cho các thuộc cấp của mình". Tương tự mỗi giám đốc sẽ sao lại thư này và kèm theo thư của mình... Bob sẽ nhận được một tập thư từ chủ tịch xuống tới trưởng nhóm tác nghiệp của mình, mỗi thư được kết nối theo tên tới thư tiếp. Nếu mỗi nhân viên trong công ty đều có một tập thư như vậy thì hai nhân viên bất kỳ muốn trao đổi các khoá đã chứng thực thì họ chỉ cần so sánh các tập thư này của nhau: cả hai tập thư sẽ có ít nhất một người chung là Edward. Chẳng hạn Bob và Ann sau khi so sánh thấy rằng các thư này là như nhau sau Camilla còn phần còn lại ở bên dưới thì Bob biết rằng phần của Ann đã được Camilla chứng thực vì nó giống như phần của anh ta và Ann cũng biết như vậy. Mỗi người đều biết phần còn lại là đúng vì nó được gắn với nhau bằng một dây liên tục các tên và chữ ký.

Thủ tục này về mặt điện tử có thể biểu thị dễ hơn trên giấy. Với giấy tờ thì phải đảm bảo chống giả mạo nhằm tránh để một thư nào đó trong tập thư bị thay thế và để đảm bảo rằng khoá công khai ở đáy được gắn kết với dây này. Về mặt điện tử thì toàn bộ các công việc này được làm thông qua các chữ ký số và các hàm băm (hash). Có thể coi Kohnfelder là người đầu tiên đưa ra ý tưởng này và sau đó Merkle đã mở rộng nó trong bài báo của mình và gần đây hơn là trong chuẩn quốc tế X.509.

Khoá công khai và tính danh của người dùng được gắn với nhau thành một chứng chỉ và rồi nó lại được ký bởi một người nào đó để chứng thực tính đúng đắn của mỗi liên kết này. Trước tiên Edward chọn một cặp khoá công khai và đặt phần công khai ở nơi mà mọi người trong công ty có thể nhận được nó và giữ lại phần bí mật của mình. Sau đó một giám đốc bộ phận (chẳng hạn Diana) sẽ tạo cặp khoá công khai của mình rồi đặt khoá công khai trong một thông báo cùng với tính danh và chuyển nó một cách an toàn tới Edward. Edward sẽ ký nó bằng cách tạo ra một giá trị băm của thông báo (tóm lược thông báo) và rồi mã hoá thông báo và hàm băm này bằng khoá riêng của mình. Bằng cách ký thông báo như vậy Edward sẽ khẳng định rằng khoá công khai (của Diana) và tính danh (cũng của Diana) trong thông báo là của cùng một người. Thông báo này được gọi là chứng chỉ của Diana. Các trưởng phòng của Diana đều tạo các thông báo bằng khoá công khai của họ. Diana sẽ ký và tóm lược (băm) chúng rồi chuyển lại đồng thời gắn thêm vào một bản sao chứng chỉ mà mình đã nhận được từ Edward. Bằng cách này mọi người có thể kiểm tra chứng chỉ của người phụ trách bằng cách xuất phát từ khoá công khai đã biết của Edward để giải mã chứng chỉ của Diana nhằm thu được khoá công khai (và tính danh) của Diana và dùng khoá này để giải mã cho chứng chỉ của người phụ trách. Hình 5.8 chỉ ra cách tạo các chứng chỉ cho Diana và Delwyn. Quá trình này cứ tiếp tục như vậy cho tới Ann và Bob. Hình H.5.9 ta thấy rằng chứng chỉ của Bob là chứng



chỉ riêng của anh ta kèm theo tất cả các chứng chỉ các xếp từ người phụ trách trực tiếp cho tới ông giám đốc công ty.

Đề tạo chứng chỉ của Diana

Diana tạo và chuyển tới Edward

Tên : Diana
Chức vụ: Giám đốc bộ phận
Khoá công khai: 17EF83CA..

Edward thêm vào

Tên : Diana	Giá trị
Chức vụ: Giám đốc bộ phận	băm
Khoá công khai: 17EF83CA..	128C4

Edward ký bằng khoá riêng của anh ta

Tên : Diana	Giá trị
Chức vụ: Giám đốc bộ phận	băm
Khoá công khai: 17EF83CA..	128C4

Đề tạo chứng chỉ của Delwyn

Delwyn tạo và chuyển tới Diana

Tên : Delwyn
Chức vụ: Trưởng phòng
Khoá công khai: 3AB3882C...

Diana thêm vào

Tên : Delwyn	Giá trị
Chức vụ: Trưởng phòng	băm
Khoá công khai: 3AB3882C...	48CFA

Diana ký bằng khoá riêng của mình

Tên : Delwyn	Giá trị
Chức vụ: Trưởng phòng	băm
Khoá công khai: 3AB3882C...	48CFA

Diana gắn thêm chứng chỉ của mình.







Tên : Delwyn	Giá trị
Chức vụ: Trưởng phòng	băm
Khoá công khai: 3AB3882C...	48CFA
Tên : Diana	Giá trị
Chức vụ: Giám đốc bộ phận	băm
Khoá công khai: 17EF83CA..	128C4

Đây là chứng chỉ của Delwyn

Hình 5.8. Các chứng chỉ có ký

Trong ví dụ này, các chứng chỉ được đưa ra theo tuyến quản lý. Tuy nhiên không nhất thiết phải có hoặc tuân theo cấu trúc quản lý để sử dụng việc ký chứng thực. Bất cứ một ai được coi là có thẩm quyền đều có thể ký một chứng chỉ. Ví dụ, nếu bạn muốn xác định xem bậc một người nào đó coi đúng là có bằng đại học hay không thì bạn không cần gặp giám đốc hoặc viện trưởng, thay vào đó bạn chỉ cần đi tới cơ quan lưu trữ. Để kiểm tra công việc của một người nào đó bạn phải gửi tới cơ quan nhân sự hoặc người phụ trách nhân sự. Và để kiểm tra xem liệu một người nào đó có sống ở một địa chỉ xác định thì bạn có thể hỏi cơ quan lưu trữ hành chính công cộng. Một công chứng viên sẽ chứng nhận tính hợp pháp của một chữ ký trên một tài liệu một số công ty có cơ quan an ninh, một số công ty có một cơ quan nhân sự tách biệt đối với mỗi bộ phận hoặc mỗi một nhà máy. Các cán bộ phụ trách của bộ phận này có thể ký xác nhận cho những người trong phạm vi quản lý của họ. Các phân cấp tự nhiên tồn tại trong xã hội và các phân cấp tương tự cũng có thể được sử dụng để tạo hiệu lực cho các chứng chỉ. Vấn đề duy nhất trong cấu trúc phân cấp là cần phải trông cậy vào mức đỉnh. Toàn bộ chuỗi xác thực là an toàn vì mỗi chứng chỉ sẽ chứa khoá phải mã cho chứng chỉ tiếp sau ngoại trừ mức đỉnh. Trong một công ty việc đặt niềm tin vào giám đốc là một điều

hợp lý. Tuy nhiên nếu các chứng chỉ trở nên được sử dụng rộng rãi trong thương mại điện tử thì mọi người phải có khả năng trao đổi các chứng chỉ một cách an toàn qua các công ty, các tổ chức và các quốc gia.

-  Được mã bằng khoá riêng của Betty
-  Được mã bằng khoá riêng của Camilla
-  Được mã bằng khoá riêng của Mukesh
-  Được mã bằng khoá riêng của Delwyn
-  Được mã bằng khoá riêng của Diana
-  Được mã bằng khoá riêng của Edward

Tên : Bob Chức vụ: Nhân viên Khóa công khai: 7013F82A...	Giá trị băm 60206
Tên : Betty Chức vụ: Trưởng nhóm Khóa công khai: 2068A6CD...	Giá trị băm 00002
Tên : Camilla Chức vụ: Trưởng ban Khóa công khai:...	Giá trị băm
Tên : Mukesh Chức vụ: Giám đốc dự án Khóa công khai: 47E0F0008...	Giá trị băm 16802
Tên : Delwyn Chức vụ: Trưởng phòng Khóa công khai: 3AB3882C...	Giá trị băm 48CFA
Tên : Diana Chức vụ: Giám đốc bộ phận Khóa công khai: 17EF83CA...	Giá trị băm 128C4

Hình 5.9. Chuỗi các chứng chỉ

Internet là một tập hợp rộng lớn của các mạng dùng trong liên lạc bên ngoài các công ty, các tổ chức và quốc tế (cũng như bên trong các công ty, các tổ chức và một đất nước). Internet được chạy (nói một cách chính xác hơn là được phối hợp) bởi một nhóm nhỏ được gọi là Cộng đồng Internet. Cộng đồng Internet đã đề nghị rằng nó là người ký cho các chứng chỉ với khoá công khai cho liên lạc Internet. Cộng đồng này sẽ ký các chứng chỉ cho Nhà chức trách chứng thực của cảnh sát và cơ quan này lại chứng thực cho các chứng chỉ cho các cơ quan nhà nước, các doanh nghiệp, các nhà trường, các tổ chức khác và cho các cá nhân. Quan điểm này có lẽ là một quan điểm đúng đắn nhất.

Bây giờ ta sẽ lướt qua một số quan điểm về phân phối khoá từ phân phối qua trao đổi trực tiếp tới phân phối qua trung tâm phối có chứng thực. Mỗi loại đều có ưu nhược điểm riêng. Những điểm cần nhớ trong các thủ tục này (cũng như những điểm khác mà ta sẽ nghiên cứu) là:

- Có những hạn chế hoạt động nào? Chẳng hạn thủ tục có cần một phương tiện khả dụng liên tục ví như trung tâm phân phối khoá hay không?
- Có những yêu cầu tin cậy nào? Ai và những thực thể nào phải được tin cậy để hoạt động đúng?

- Có những gì để bảo vệ tránh khỏi những trục trặc? Liệu một kẻ bên ngoài có thể giả mạo một thực thể nào đó trong thủ tục và gây tổn hại tới tính an toàn? Liệu có một kẻ đồng lõa nào đó trong thủ tục thực hiện lừa đảo mà không bị phát hiện?
- Hiệu quả của thủ tục ra sao? Một thủ tục yêu cầu một số bước để thiết lập một khoá mã hoá sẽ được sử dụng nhiều lần là một vấn đề, một vấn đề khác là phải qua một số bước mất rất nhiều thời gian để có một khoá sử dụng một lần.
- Sử dụng một thủ tục có dễ dàng không? Cần chú ý rằng độ phức tạp trong các ứng dụng máy tính có thể khác với độ phức tạp trong các ứng dụng thủ công. Ta sẽ đưa ra một số mô tả cho một vài thủ tục, một số thủ tục sẽ được đưa ra trong các ví dụ cuối chương. Bạn cần phải ghi nhớ những điểm này khi nghiên cứu những thủ tục còn lại trong chương này.

### 5.2.2. Các chữ ký số

Ta sẽ nghiên cứu một ứng dụng điển hình trong máy tính thể hiện một nhu cầu thông thường của con người. Lệnh chuyển tiền từ một người này tới một người khác. Về căn bản đây là một dạng séc đã được máy tính hoá. Theo truyền thống ta đã hiểu giao dịch này sẽ được thực hiện như thế nào ở dạng giấy tờ.

- Séc là một đối tượng xác định có tư cách giao dịch thương mại.
- Chữ ký trên séc sẽ xác nhận tính xác thực bởi vì chắc chắn chỉ có người ký hợp pháp mới có thể tạo được chữ ký này.
- Trong trường hợp có sự giả mạo bất hợp pháp thì một bên thứ ba có thể được gọi và để phán xét tính xác thực.
- Séc bị huỷ để nó không thể sử dụng lại.
- Séc giấy không thể thay đổi được hay hầu hết các kiểu thay đổi đều có thể dễ dàng phát hiện được.

Việc giao dịch thương mại bằng séc phụ thuộc vào các đối tượng xác định ở dạng được mô tả trước.

Đối với các giao dịch trên máy tính không tồn tại các đối tượng xác định (séc). Bởi vậy việc chấp nhận chi trả bằng máy tính đòi hỏi một mô hình khác. Ta sẽ xem xét các yêu cầu trong một tình thế tương tự từ phía nhà băng và phía người sử dụng.

Sandy gửi cho ngân hàng của mình một thông báo cho phép ngân hàng chuyển 100 USD cho Tim. Ngân hàng của Sandy phải có khả năng kiểm tra và chứng tỏ được rằng thông báo thực sự đến từ Sandy nếu sau đó cô ta không nhận là mình đã gửi nó. Ngân hàng cũng muốn biết rằng toàn bộ thông báo này là của Sandy và nó không thể sửa đổi. Về phần mình, Sandy cũng muốn chắc chắn rằng ngân hàng của cô ta không thể giả mạo những thông báo đó là thông báo mới, không phải là một thông báo trước đó đã được sử dụng lại và nó phải không bị sửa đổi trong khi truyền. Việc sử dụng các tín hiệu điện tử thay cho giấy sẽ làm phức tạp thêm giao dịch này.

Chữ ký số là một thủ tục tạo ra một hiệu quả tương tự như chữ ký thực. Nó là một dấu hiệu mà chỉ có người gửi mới có thể tạo ra nhưng những người khác có thể dễ nhận thấy được rằng nó là của người gửi. Giống như chữ ký thực, chữ ký số được dùng để xác nhận nội dung của một thông báo.

Các chữ ký phải thoả mãn các điều kiện cơ bản sau:

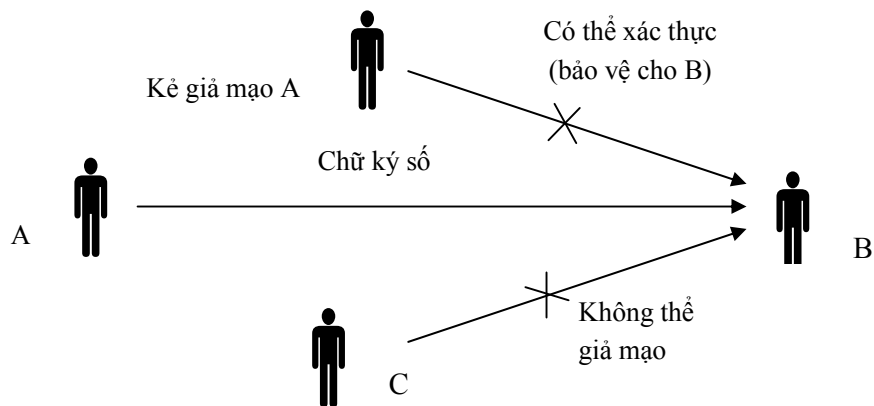
- Không thể giả mạo. Nếu P ký thông báo M bằng chữ ký  $S(P, M)$  thì không một ai có thể tạo được cặp  $[M, S(M, P)]$
- Xác thực. Nếu R nhận được cặp  $[M, S(M, P)]$  được coi là của P thì R có thể kiểm tra được rằng chữ ký có thực sự là của P hay không? Chỉ có P mới có thể tạo được chữ ký này và chữ ký được "gắn chặt" với M. Hai yêu cầu đầu tiên này là những trở ngại chính trong giao dịch qua máy tính. Hai tính chất bổ trợ sau là những tính chất mong muốn đối với giao dịch được hoàn tất nhờ chữ ký số:
- Không thể thay đổi. Sau khi được phát M không thể thay đổi bởi S, R hoặc bởi một kẻ thu trộm nào.
- Không thể sử dụng lại. Một thông báo trước đó được đưa ra sẽ ngay lập tức bị R phát hiện.

Trước tiên ta sẽ trình bày một cơ chế thoả mãn hai yêu cầu đầu tiên, sau đó ta sẽ thêm vào đó giải pháp để thoả mãn được các yêu cầu khác.

#### **5.2.2.1. Các chữ ký số khoá đối xứng**

Với hệ thống mã hoá riêng việc giữ bí mật khoá sẽ đảm bảo tính xác thực của thông báo cũng như độ mật của nó. Nếu Sandy và ngân hàng có một khoá mã hoá chung thì Sandy có thể mã hoá yêu cầu chuyển tiền của mình. Ngân hàng có thể tin vào tính xác thực của thông báo vì không một ai ngoài Sandy có khoá này.

Tuy nhiên hệ thống mã với khoá đối xứng truyền thống không tránh khỏi được sự giả mạo: Ngân hàng có thể tạo ra thông báo như vậy vì ngân hàng cũng có khoá này. Bởi vậy ngân hàng có thể tin chắc vào tính xác thực của thông báo, nhưng ngân hàng không có sự bảo vệ chống lại sự chối bỏ (không thừa nhận việc gửi thông báo). Vì cả Sandy và ngân hàng đều có cùng một khoá nên cả Sandy và ngân hàng đều có thể tạo một thông báo bất kỳ. Khi Sandy nhận một thông báo từ ngân hàng, cô ta tin vào tính xác thực của nó nhưng cô ta có thể tin vào bất cứ một ai khác. Xác thực nhưng không thể là không chối bỏ là một đặc điểm của mã hoá đối xứng. Với các hệ mật đối xứng như chuẩn mã dữ liệu (DES) cần phải có một trọng tài để tránh khỏi việc giả mạo.

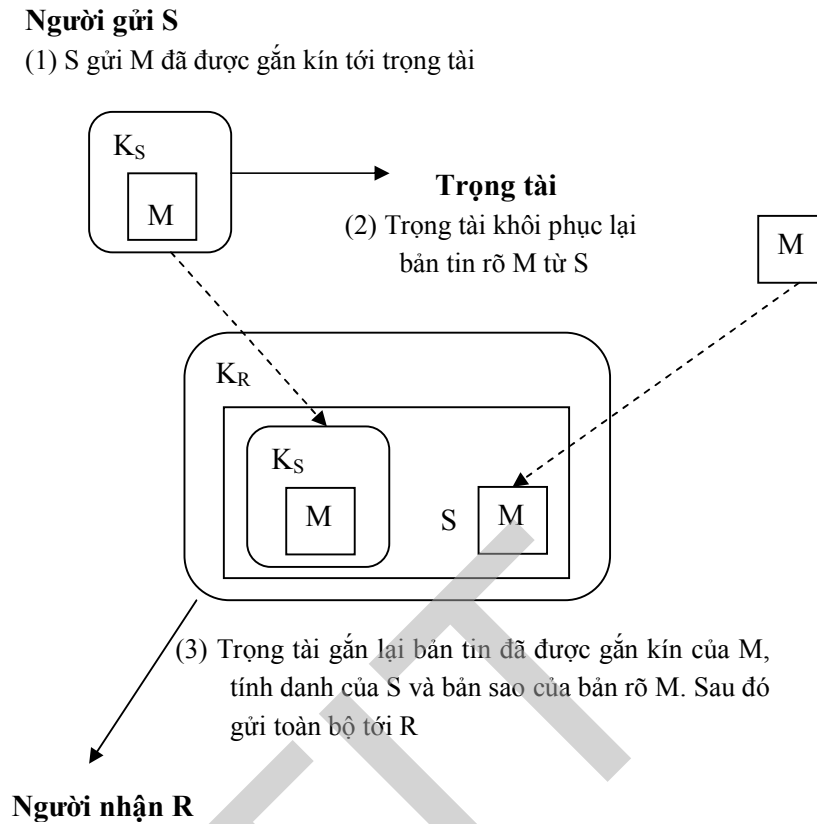


Hình 5.10. Những yêu cầu đối với chữ ký số

Sau đây là một phác thảo của thủ tục chữ ký số. Giả sử S là người gửi (Sandy), A là trọng tài và R là người nhận (Ngân hàng). S có khoá  $K_S$  chung với A và R có khoá  $K_R$  chung với A. Giả sử S và R đã thoả thuận trước về khuôn dạng cho một chữ ký số. S muốn gửi một thông báo M cho R với các yêu cầu phụ trợ là thông báo này không thể giả mạo về tính xác thực của nó có thể kiểm tra được.

Như trên hình 5.11, trước tiên S sẽ gửi  $E(M, K_S)$  tới trọng tài A. A sẽ gửi ra M bằng cách dùng  $K_S$ . Sau khi kiểm tra thấy rằng thông báo này thực sự là từ S (vì trọng tài có thể giải nó bằng khoá  $K_S$  của Sandy), A sẽ gửi  $E((M, S, E(M, K_S)), K_S)$  tới R. R sẽ thu toàn bộ thông báo M đã được mã bằng khoá  $K_R$  để R có thể giải mã và xử sự theo nội dung của thông báo. R cũng thu thông báo S của A, thông báo này nói rằng A chứng thực là thông báo tới từ S. R cũng thu được  $E(M, K_S)$ , đây là thông báo mà R không thể giải mã vì nó được mã bằng khoá  $K_S$ . Tuy nhiên R sẽ giữ lại một bản sao của M và  $E(M, K_S)$  phòng trường hợp có sự tranh chấp sau này. Ở đây điều kiện xác thực được thoả mãn vì R tin tưởng vào trọng tài (là người nói rằng thông báo tới từ S). Tính chất không thể giả mạo cũng được thoả mãn vì nếu sau đó S tuyên bố rằng đó là bản giả mạo thì S sẽ đưa ra M và  $E(M, K_S)$ . Trọng tài có thể mã hoá lại M bằng  $K_S$  và xác nhận rằng chỉ có S (hoặc trọng tài - người mà ta coi là trung thực) mới có thể tạo được  $E(M, K_S)$  vì nó được mã bằng  $K_S$ . Bởi vậy trọng tài có thể chứng thực rằng S (hoặc một ai đó có khoá  $K_S$ ) đã gửi M.

Thủ tục này sẽ tạo ra một hệ thống thực sự mạnh hơn yêu cầu, thông báo đã được phát ở dạng đã mã hoá ngay cả khi không cần phải bí mật trong mọi tình huống. Như ta đã thấy, một số thuật toán mã hoá tiêu tốn khá nhiều thời gian và sử dụng nó sẽ làm giảm tốc độ truyền thông báo. Bởi vậy ta muốn có một thủ tục khác không cần phải mã hoá toàn bộ thông báo.



Hình 5.11. Chữ ký số khoá đối xứng có trọng tài

#### 5.2.2.2. Các chữ ký số không có mã hoá

Nếu S và R không quan tâm tới độ mật thì họ có thể thoả thuận dùng hàm niêm phong mật mã như một chữ ký. Niêm phong là một con tem, con dấu hoặc một nhãn gắn cố định vào tư liệu để chứng tỏ tính xác thực của nó. Hàm niêm phong là một hàm toán học chịu tác động bởi mọi bit đầu vào của nó. Ví dụ, các bytes của một thông báo có thể được dùng như các số và tổng của tất cả các bytes của một thông báo có thể tính toán được. Tổng này là duy nhất đối với một thông báo vì bất kỳ một sự thay đổi nào của thông báo cũng đều gây nên sự thay đổi của tổng này. Hàm niêm phong có thể là một hàm băm hoặc một hàm mã hoá một chiều, hoặc nó cũng có thể là một hàm khác phụ thuộc vào toàn bộ đầu vào và dễ tính toán.

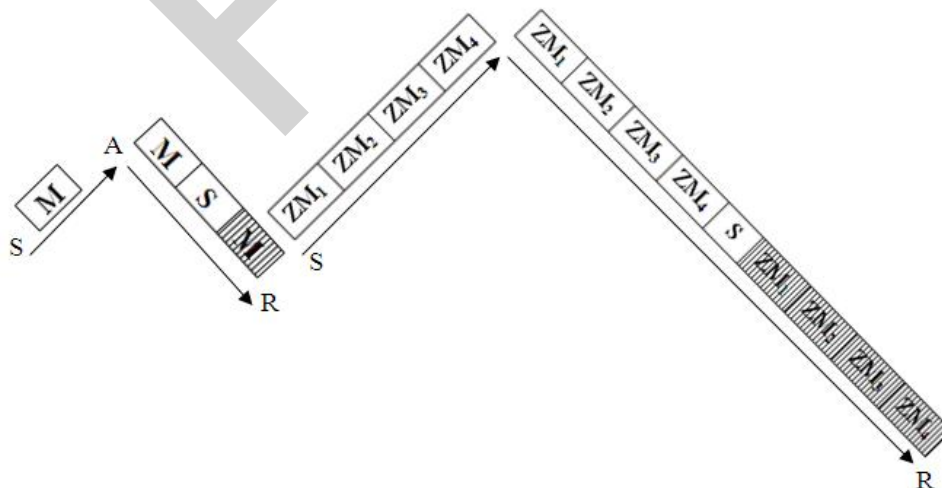
Giả sử S và R đều đăng ký một hàm niêm phong của mình với trọng tài: giả sử  $f_S$  và  $f_R$  là hai hàm này. Sau đó, S gửi M và  $f_S(M)$  tới A. A cũng tính  $f_S(M)$  theo bản sao của M đã nhận được từ S. Nếu hai giá trị này của  $f_S(M)$  trùng nhau thì thông báo từ S được coi là xác thực. Sau đó, A gửi M, S,  $f_S(M)$  và  $f_R(M, S)$  cho R. R không thể hiểu được  $f_S(M)$ , nhưng R sẽ giữ lại nó làm bằng chứng báo rằng S đã gửi thông báo M. S sẽ kiểm tra tính đúng đắn của  $f_R(M, S)$  để biết tính xác thực của thông báo.

### 5.2.2.3. Ngăn chặn sử dụng lại hoặc biến đổi

Cả hai giải pháp trên đều thoả mãn các đòi hỏi về tính xác thực và chống giả mạo. Bởi vậy, chúng đều là các thủ tục chữ ký số chấp nhận được. Tuy nhiên, ta còn muốn tránh được việc sử dụng lại hoặc biến đổi một thông báo cũ. Mặc dù R không thể tạo các thông báo mới bằng  $K_S$  hoặc  $f_S$  nhưng bên thu có thể sử dụng lại các thông báo cũ. Ví dụ, bên thu có thể lưu lệnh cũ trả cho Tim 100 USD hàng tháng. Hơn nữa, biết kỹ thuật mã hoá nhưng không biết khoá bên thu vẫn có thể cắt các mẫu từ các thông báo cũ và dán chúng với nhau để tạo nên một thông báo mới. Việc kiểm tra giấy tờ sẽ giải quyết vấn đề này như sau: ngân hàng sẽ đóng dấu huỷ séc và trả nó lại cho người gửi để người gửi biết không thể sử dụng lại séc này. Ta cần có một cách để tạo một chữ ký máy tính tự huỷ tương tự để không thể sử dụng lại được.

Để minh chứng việc sử dụng một thông báo, ta phải tạo một phần của chữ ký vào một nhãn thời gian. Ví dụ, nếu thông báo của Sandy có ghi ngày tháng và thời gian gửi thì ngân hàng không thể tạo lại thông báo tương tự vào tuần sau mà không bị Sandy và những người khác phát hiện ra sự giả mạo. Nhãn thời gian không phải là thời gian và ngày tháng theo nghĩa đen mà là một mã bất kỳ không lặp lại, chẳng hạn như tăng số loạt. Điều này sẽ gây khó khăn cho việc tạo lại.

Để tránh việc cắt thông báo thành nhiều mẫu và sử dụng lại một mẫu nào đó, Sandy có thể làm cho mỗi mẫu đều phụ thuộc nhãn thời gian. Ví dụ, với DES (thuật toán này sẽ mã các khối văn bản 64 bit thành một đầu ra 64 bit). Sandy có thể mã thời gian và ngày tháng ở 8 bit đầu của mỗi khối và chỉ để 56 bit làm nội dung của thông báo. Vì tất cả 64 bit của mỗi khối ra đều phụ thuộc vào toàn bộ 64 bit vào nên ngân hàng không thể gắn 56 bit thông báo với một nhãn thời gian 8 bit khác. Quá trình này được chỉ ra trên hình 5.12. Như sẽ thấy ở cuối chương, chế độ liên kết khối mã của DES cũng là một phương pháp khác nhằm tránh việc sử dụng lại một khối của một thông báo vào một thông báo khác.



Hình 5.12. Chữ ký số khoá đối xứng

Những giải pháp trên là khá phức tạp. Chúng yêu cầu phải luôn có một trọng tài trong mỗi giao dịch và để đảm bảo bí mật thì thông báo phải được mã hai lần. Rất may là thủ tục khoá công khai sẽ cho ta tạo một phương pháp đơn giản hơn.

#### 5.2.2.4. Thủ tục khoá công khai

Các hệ thống khoá công khai là các hệ thống thích hợp nhất cho chữ ký số. Để đơn giản về ký hiệu, ta giả sử rằng  $E(M, K_U)$  là mã hoá khoá công khai cho người dùng  $U$  và phép biến đổi khoá riêng cho  $U$  được ký hiệu là  $D(M, K_U)$ . Ta cũng có thể coi  $E$  là phép biến đổi riêng (vì chỉ có  $U$  mới có thể giải mã nó) và coi  $D$  là một phép biến đổi xác thực (vì chỉ có  $U$  mới có thể tạo ra nó). Tuy nhiên, cần nhớ rằng với một số thuật toán không đối xứng (như RSA),  $D$  và  $A$  là giao hoán. Bởi vậy:

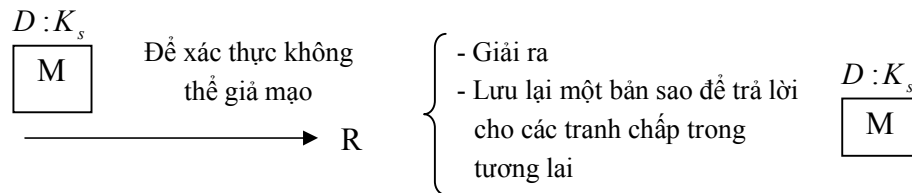
$$D(E(M, -), -) = M = E(D(M, -), -)$$

Nếu  $S$  muốn gửi  $M$  cho  $R$  thì  $S$  sẽ dùng phép biến đổi xác thực để tạo  $D(M, K_S)$ . Sau đó  $S$  gửi  $D(M, K_S)$  cho  $R$ .  $R$  sẽ giải mã thông báo bằng phép biến đổi khoá công khai của  $S$  ( $S$  tính  $E(D(M, K_S), K_S) = M$ , xem hình 5.13). Vì chỉ có  $S$  mới có thể tạo ra một thông báo có nghĩa dưới dạng  $E(-, K_S)$  nên thông báo này phải thực sự là từ  $S$ . Phép kiểm tra này thoả mãn yêu cầu về tính xác thực.

$R$  sẽ lưu giữ  $D(M, K_S)$ . Nếu sau đó  $S$  khẳng định rằng thông báo này là giả mạo (không phải từ  $S$ ) thì  $R$  chỉ cần đưa ra  $M$  và  $E(M, K_S)$ . Bất kỳ ai cũng có thể kiểm tra được rằng vì  $D(M, K_S)$  được chuyển thành  $M$  nhờ dùng phép biến đổi khoá công khai của  $S$  nhưng chỉ có  $S$  mới có thể tạo ra  $D(M, K_S)$  nên  $D(M, K_S)$  phải là từ  $S$ . Phép kiểm tra này thoả mãn yêu cầu không thể giả mạo.

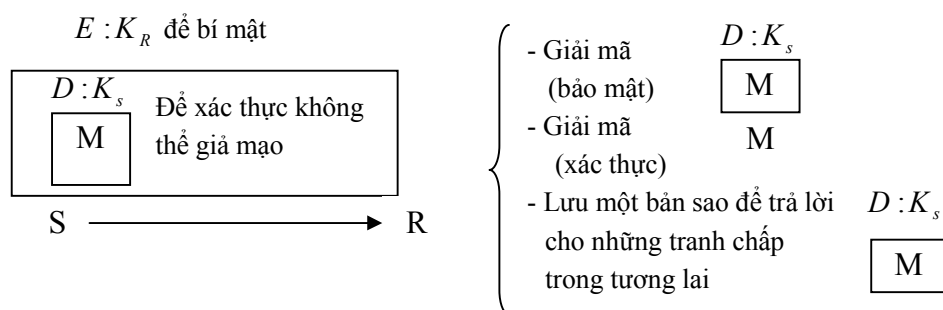
Giải pháp khoá công khai hiển nhiên là ít phức tạp hơn nhiều so với giải pháp khoá riêng. Một nhược điểm của giải pháp này là: thông báo là xác thực nhưng không đảm bảo tính bí mật. Tức là, bất cứ ai biết khoá công khai của  $S$  đều có thể dịch được thông báo này. Ta có thể khắc phục nhược điểm này bằng cách mã hoá 2 lần.

Vì  $S$  cần phải gửi  $M$  cho  $R$  nên tính xác thực có thể được đảm bảo bằng cách dùng khoá riêng của  $S$ , còn tính bí mật có thể được đảm bảo bằng khoá công khai của  $R$ . Ví dụ,  $S$  có thể gửi  $E(D(M, K_S), K_R)$  tới  $R$  (như được chỉ ra trên hình 5.13). Vì chỉ có  $S$  mới có thể tạo ra  $D(-, K_S)$  nên thông báo này phải từ  $S$ . Nhưng vì chỉ có  $R$  mới có thể giải mã cho  $E(-, K_R)$  nên nội dung của thông báo vẫn được giữ bí mật cho tới khi  $R$  giải nó. Như vậy, với hai phép mã hoá, ta sẽ có một thủ tục đảm bảo cả tính riêng tư và tính xác thực.



Hình 5.13. Chữ ký số không đối xứng





Hình 5.14. Sử dụng hai phép mã hoá trong chữ ký số không đối xứng

Như đã nêu ở trên, các nhân thời gian có thể được dùng để đảm bảo chống lại việc sử dụng lại. Một hàm niêm phong cũng có thể khắc phục được việc thay thế các mẫu trong bản mã.

### 5.2.3. Giao kèo về khoá

Giả sử bạn làm mất chìa khoá nhà. Với đôi chút bất tiện, bạn có thể gọi một thợ khoá để mở khoá cho bạn, và thậm chí có thể làm cho bạn một chìa khoá khác. Bạn cần phải thuyết phục người thợ khoá để bạn vào nhà. Tuy nhiên, nếu bạn có một giấy tờ xác nhận nào đó, hoặc những người láng giềng đảm bảo cho thẩm quyền của bạn thì bạn không cần phải thuyết phục người thợ khoá.

Nếu bạn mất khoá của két bạc thì vấn đề sẽ phức tạp hơn đôi chút và người thợ khoá phải mất nhiều thời gian hơn. Tuy nhiên, trong một thời gian tương đối ngắn, bạn vẫn có thể mở được. Sở dĩ như vậy vì các loại khoá nhà và khoá két bạc được thiết kế chỉ để chống lại các kiểu tấn công trong một khoảng thời gian ngắn: một hoặc tối đa hai ngày. An toàn về vật lý đóng một phần quan trọng trong việc đảm bảo an toàn cho các ngôi nhà và các công sở. Ta mong muốn có thể ngăn chặn một cuộc tấn công kéo dài trong nhiều giờ.

Bản mã không có tính an toàn về mặt vật lý. Trên thực tế, nó được dùng để liên lạc trên các kênh không an toàn, nơi mà các thám mã có thể thu được các bản mã và phân tích nó trong nhiều giờ, nhiều ngày, nhiều tuần và thậm chí là hàng năm. Bởi vậy, các thuật toán mã hoá phải đủ mạnh để chống lại được với các kiểu tấn công. Do đó, không có một "người thợ khoá" nào có thể tìm được các khoá bị mất hoặc bị quên. Nếu bạn làm việc trong một công ty và bạn đã mã hoá các file của mình thì vẫn có người có nhu cầu chính đáng để truy nhập vào khi bạn rời khỏi cơ quan trong một chuyến công tác hay trong các ngày nghỉ, hoặc khi bạn ốm hay khi bạn không còn ở công ty nữa. Bạn cũng có thể quên mất khoá của bạn. Đây là một trong các ưu điểm của giao kèo khoá và tại sao nó lại thích hợp trong những tình huống như vậy: giao kèo khoá sẽ cung cấp một phương tiện để đảm bảo an toàn số liệu ở một mức độ thích hợp nhưng vẫn cho phép những người dùng hợp pháp khác truy nhập vào.

#### 5.2.3.1. Giao kèo khoá Clipper

Quan điểm Clipper là một ví dụ về một thủ tục giao kèo khoá. Bản chất của thuật toán này là chia một khoá thành  $n$  mẫu rồi giao cho  $n$  đối tượng sao cho  $k$  người nào trong số này đều có thể giải mã nhưng không có  $(k-1)$  người nào có thể giải mã được. Thủ tục này được gọi là thủ tục  $k$  trong  $n$ . Đối với thủ tục Clipper,  $k = n = 2$ . (Trường hợp tổng quát hơn khi

$n > 2$  đã được Shamir nghiên cứu, ông đã đưa ra một giải pháp dựa trên việc tìm các nghiệm của một đa thức bậc  $n$ . Tuy nhiên, Blakley đã độc lập giải quyết vấn đề trên theo một phương pháp khác; chương 9 trong [9] của Simon đã có một tổng quan khá hay về các sơ đồ khác.)

Những yêu cầu đối với một thủ tục giao kèo khoá bao gồm:

- Liên lạc phải xác định được nguồn mã hoá của nó.
- Tính danh chỉ cho biết đơn vị mã hoá mà không để lộ khoá mã hoá riêng.
- Có thể phục hồi được khoá nhờ thủ tục  $k$  trong  $n$ .

Thủ tục mã hoá Clipper thoả mãn được các yêu cầu trên. Với thủ tục Clipper, đơn vị mã hoá xuất phát sẽ tạo ngẫu nhiên một khoá phiên 80 bit. Nó sẽ mã hoá khoá 80 bit này bằng một khoá duy nhất của mỗi đơn vị vật lý cụ thể có gắn số loạt đối với mỗi đơn vị vật lý và mã hoá cả hai bằng một khoá được chia sẻ bởi tất cả các thiết bị tương tự; nhóm được mã hoá này được gọi là LEAF (nhóm truy nhập hợp lệ).

Nó cũng sử dụng thủ tục trao đổi khoá riêng của mình để chuyển khoá này tới nơi thu. Tất cả các đơn vị Clipper đã được lập trình trước từ nhà máy với một khoá mã hoá đối xứng được chia sẻ (gọi là khoá gia đình). Clipper sử dụng một thủ tục trao đổi khoá 3 giai đoạn để trao đổi khoá. Bài toán này có những đòi hỏi sau:

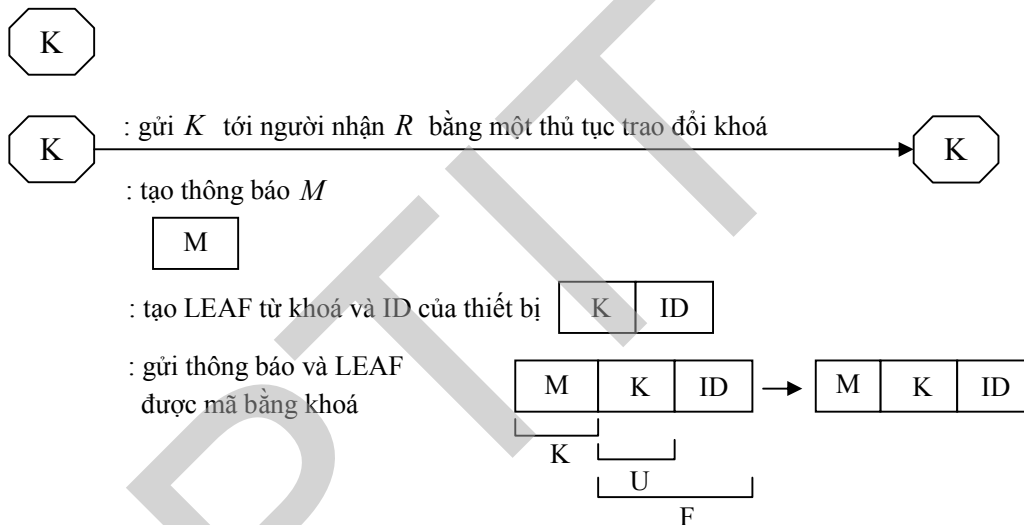
- Các khoá mã hoá chỉ được phát ở dạng đã mã hoá.
- Việc trao đổi giữa các đơn vị Clipper phải chống được việc sử dụng lại.
- Hai đơn vị Clipper trong trao đổi phải xác thực nhau.

Thuật toán cần 3 bước để trao đổi khoá cộng thêm 4 bước để xác thực nhau. Thuật toán này do Bellovin và Merritt đưa ra [10]. Bên gửi  $S$  và bên nhận  $R$  chia sẻ một mật khẩu chung  $P$ , ngoài ra chúng có một thuật toán mã hoá không đối xứng  $E$  và một thuật toán mã hoá đối xứng chung  $Y$ .

1.  $S$  tạo một khoá công khai ngẫu nhiên  $E_S$ .  $S$  mã hoá  $E_S$  bằng  $Y_P$  và gửi  $Y_P(E_S)$  cho  $R$ .
2.  $R$  biết  $P$  và dùng nó để giải mã và nhận được  $E_S$ .  $R$  sẽ tạo một khoá phiên ngẫu nhiên  $X$ .  $R$  chuyển  $X$  cho  $S$  bằng cách mã hoá kép bằng  $Y_P$  và  $E_S$ :  $R$  gửi  $E_S(Y_P(X))$ .
3.  $S$  dùng  $E_S$  và  $P$  để thu được  $X$ . Lúc này, việc trao đổi khoá đã hoàn tất. Bây giờ,  $S$  và  $R$  phải đảm bảo rằng họ đã có khoá mới  $X$  để sử dụng.
4.  $S$  tạo một xâu ngẫu nhiên  $M$  và gửi  $E_X(M)$  cho  $R$ .
5.  $R$  giải mã thông báo này để thu được  $M$ .  $R$  chọn một xâu ngẫu nhiên  $N$  khác và chuyển trả lại cho  $S$  xâu ghép đã được mã hoá bằng khoá  $X$ :  $E_X((M, N))$ .

6.  $S$  giải mã thông báo này để thu được  $(M, N)$ . Nếu phần đầu  $M$  chính là xâu mà  $S$  đã gửi thì  $R$  đã được xác thực đối với  $S$ .  $S$  gửi trả  $E_x(N)$  cho  $R$ .
7. Nếu  $R$  thu được  $N$  đã gửi ở bước 4 thì  $S$  đã được xác thực đối với  $R$ . Sau khi Clipper đã hoàn thành việc trao đổi khoá, nó bắt đầu lưu chuyển thông báo với bên thu (mỗi thông báo có kèm theo LEAF). Như được mô tả trên H 5.15, một thông báo có kèm theo LEAF sẽ cho phép bên nhận hợp thức thu được khoá được dùng để mã hoá.
2. Bằng thủ tục này, các đối tác hợp thức thu chặn thông báo có thể dùng khoá gia đình  $F$  để giải mã đoạn cuối cùng của  $E((E(K, U) + ID), F)$  và thu được ID là số của thiết bị đã tạo ra khoá mã hoá thông báo. Họ có thể đệ trình lên một "toà án" để nhận được  $U$  mà từ nó, họ có thể thu được  $K$  từ  $E(K, U)$ .

Người gửi  $S$  : chọn khoá phiên  $K$



Hình 5.15. Trao đổi thông báo Clipper

Tuy nhiên, thủ tục này có một số điểm không mong muốn. Trước hết là, một khi các đối tác hợp thức đã nhận được  $U$  từ "toà án" thì giá trị này sẽ cho phép họ truy nhập tới mọi thông báo đã được gửi bởi thiết bị này (trong quá khứ hoặc trong tương lai), thậm chí cả khi thiết bị này đã được mua hoặc bán một cách hợp pháp cho một người khác. Cũng vậy, một khi các đối tác hợp thức đã có  $U$  thì họ có thể tạo các thông báo mang danh người gửi, bởi vì  $E$  là một thuật toán không đối xứng. Tuy nhiên, những người sử dụng có thể tránh được tình huống này nếu dùng mã hoá bội: các đối tác hợp thức chỉ có thể giải mã  $E(M, K)$  để tìm ra cái mà người gửi đã mã (là  $M$ ) trước khi chuyển nó tới Clipper để mã. Tức là,  $M$  phải là  $E'(M, K')$  đối với một phép mã hoá  $E'$  khác hoặc một khoá  $K'$  khác. Cần chú ý rằng, việc đặt phép mã hoá Skipjack chỉ ở trong phần cứng nhằm chống trả có hiệu quả việc thám mã bằng ứng dụng mật mã.

### 5.2.3.2. *Giao kèo khoá bằng phần mềm*

Các ứng dụng mật mã bằng phần mềm có cả ưu và nhược điểm so với ứng dụng phần cứng. Chúng dễ sửa đổi hoặc dễ làm phù hợp theo các yêu cầu của các ứng dụng riêng biệt. Chúng có thể được gắn vào hoặc được tích hợp vào các ứng dụng có thể được cài đặt hoặc chạy trên một máy tính bất kỳ. Các sai sót nếu có có thể dễ dàng được sửa và chi phí để sửa không đáng kể. Tuy nhiên, khả năng thay được cũng là một bất lợi lớn nhất: chúng có thể bị thay đổi bởi một sự cố hoặc bởi một chương trình ác ý nào đó nằm trên phương tiện lưu trữ chúng, trên bộ nhớ hoặc khi thực hiện chương trình. Ta cũng đã từng thấy rằng chỉ một thay đổi rất nhỏ, thậm chí chỉ là 1 bit cũng có thể gây nên hậu quả nghiêm trọng đối với sức mạnh của thuật toán mật mã. Tuy vậy, tính mềm dẻo của phần mềm vẫn làm cho nó có thể sử dụng được trong một số tình huống.

Việc mã hoá có giao kèo khoá dựa trên phần mềm cũng hoạt động tương tự như giao kèo khoá Clipper. Một cơ quan giao kèo sẽ lập trình trước một ứng dụng mật mã (phần mềm) với một khoá gia đình và một ID của thiết bị. Ứng dụng này sẽ được bán cho một nhà chế tạo phần mềm để gắn nó vào một sản phẩm phần mềm. Mỗi bản sao của phần mềm (tương tự như đối với một thiết bị phần cứng riêng) phải có ID riêng của nó. Sản phẩm sẽ nhận vào bản rõ và một khoá mã hoá, tạo ra bản mã và một LEAF. Phần mềm thu sẽ kiểm tra sự tồn tại của LEAF và giải mã. Người ta thấy rằng sử dụng mã hoá khoá công khai trong thủ tục giao kèo khoá là thích hợp hơn cả. Denning và Branstad [11] đã trình bày một số kết quả nghiên cứu lý thú về các thủ tục giao kèo khoá bằng phần mềm.

### 5.2.4. *Chơi bài qua thư tín*

Trong phần này, ta sẽ xem xét một tình huống tương tự khi cần đảm bảo cả bí mật lẫn xác thực. Ta xét một trò chơi bài mà những người tham gia không nhìn thấy nhau (chẳng hạn, chơi bài qua thư tín). Ngay cả khi việc chơi bài qua thư tín không phải là một công việc quan trọng thì thủ tục chơi vẫn có những ứng dụng quan trọng. Phần khó khăn nhất của việc chơi bài qua thư tín là việc đảm bảo rằng các quân bài phải được chia và phân phối một cách công bằng. Trong phần sau, ta sẽ sử dụng thủ tục này để phân phối các khoá mã hoá, trong trường hợp này, các quân bài chính là các khoá. Việc đặt quân bài sẽ dễ hơn cho việc giải thích thủ tục.

Giả sử Ann và Bill quyết định chơi bài qua thư tín. Ann là người chia và xóc các quân bài. (Để đơn giản, giả sử chỉ có 10 quân bài và mỗi bên chơi bài nhận 5 quân. Thủ tục này có thể dễ dàng mở rộng cho bộ bài truyền thống 52 quân).

#### 5.2.4.1. *Thủ tục phân phối*

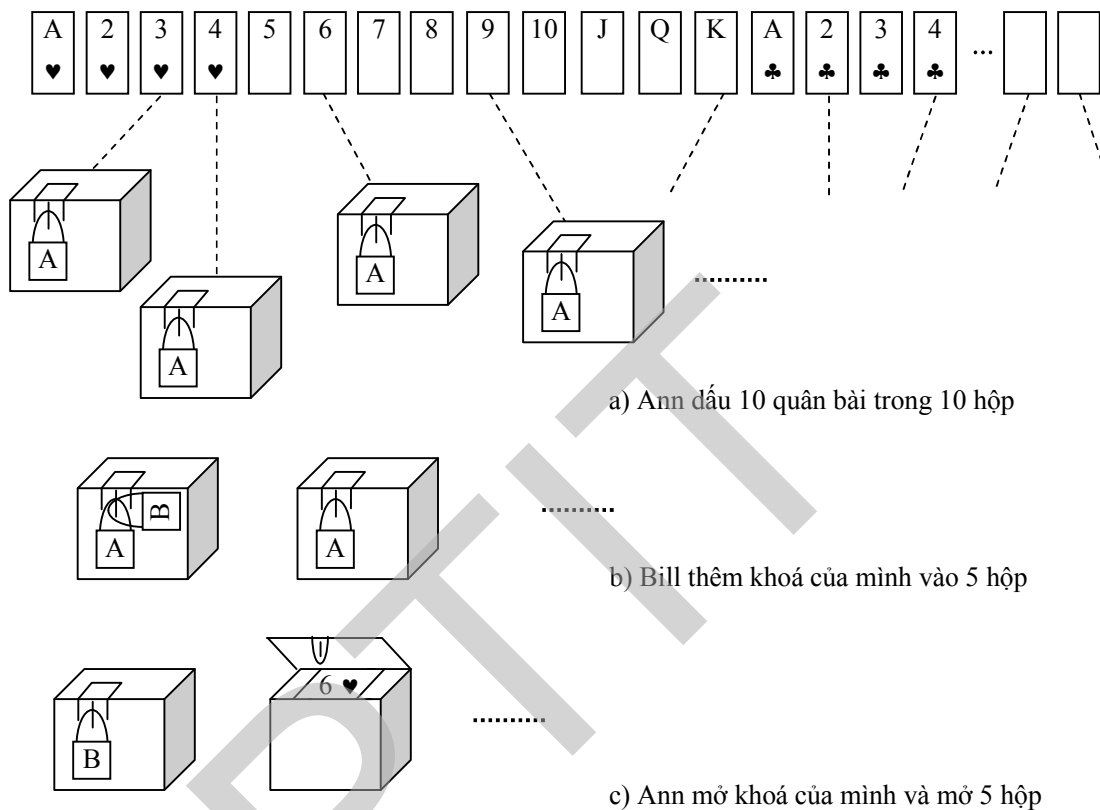
Để xáo bài, Ann sẽ đặt chúng theo một thứ tự tùy ý và đặt mỗi quân bài vào một hộp không đánh dấu sẵn rồi khoá từng hộp lại. Sau đó, Ann sẽ gửi 10 hộp cho Bill. Vì các hộp đều như nhau nên Bill sẽ chọn 5 hộp và đặt thêm một khoá thứ hai vào các hộp này. 5 hộp này là lựa chọn của Bill và Bill không động chạm đến 5 hộp còn lại (chỉ có khoá của Ann trên đó). Sau đó, Bill sẽ gửi trả lại Ann cả 10 hộp.

Theo các hộp đã nhận, Ann có thể xác định được Bill có xử sự đúng đắn không: anh ta đã chỉ chọn 5 hộp và không đụng chạm tới các quân bài (vì các khoá của Ann vẫn còn

nguyên). Ann sẽ mở cả 10 khoá của mình. Khi đó, Ann có 5 quân bài mà Bill không động tới (đây là những quân bài mà Ann đã chọn).

Ann sẽ gửi trả 5 hộp còn lại vẫn còn khoá của Bill. Bill sẽ mở khoá cho 5 hộp này. Ann không biết các quân bài này vì Ann không được mở các khoá của Bill. Bill (tin vào sự công minh của việc xáo bài) sẽ tiếp tục chơi.

Quá trình trên được mô tả trên hình 5.16.



Hình 5.16. Thủ tục chơi bài

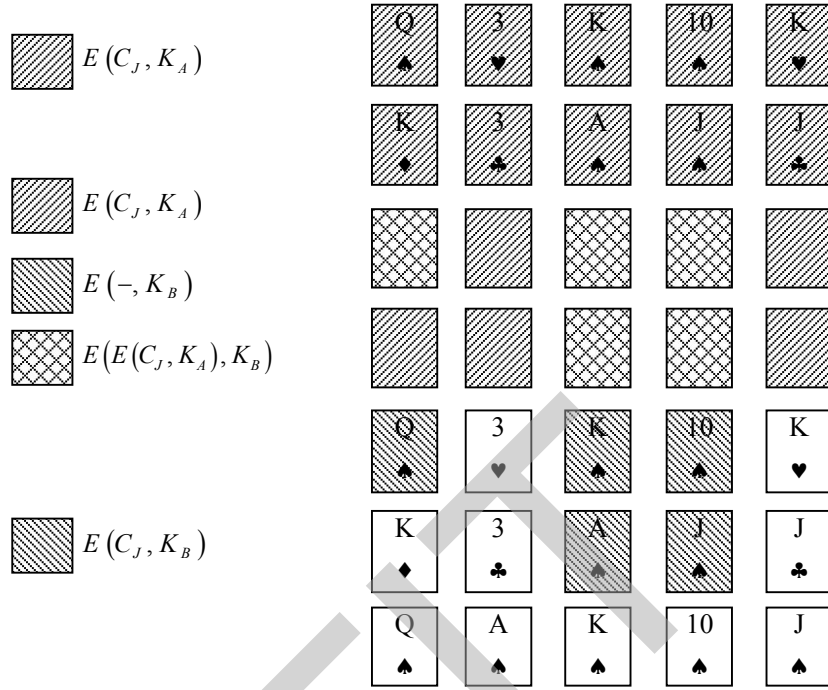
#### 5.2.4.2. Áp dụng khoá đối xứng

Thủ tục này làm việc dễ dàng với các hệ thống mã hoá truyền thống. Để khái quát hoá, ta sử dụng thuật ngữ "thông báo" thay cho "quân bài" trong thủ tục. Thủ tục này được mô tả trên hình 5.17.

Ann sẽ "xáo" các thông báo và mã hoá chúng bằng khoá  $K_A$  (khoá của Ann). Sau đó, Ann sẽ gửi 10 thông báo đã mã này cho Bill. Khi nhận được chúng, Bill sẽ chọn ngẫu nhiên 5 cái. Bill không thể xác định được các thông báo này vì chúng đã được mã bằng khoá của Ann. Bill sẽ gửi trả lại cho Ann những thông báo không được chọn.

Bill sẽ nhận 5 thông báo còn lại (ta ký hiệu chúng là  $B_1, \dots, B_5$ ). Bill sẽ gửi  $E(B_1, K_B), \dots, E(B_5, K_B)$  cho Ann. Chú ý rằng Ann đã gửi các thông báo đã mã cho Bill, các  $B_i$ , chính là  $E(c_j, K_A)$  (thông báo  $c_j$  được mã bằng khoá của Ann). Bởi vậy, trên thực

tế Bill đã gửi  $E(E(c_j, K_A), K_B)$  cho Ann. Phép mã hoá với khoá của Bill sẽ tạo nên một khoá kép trên các thông báo này để đảm bảo rằng Bill sẽ nhận trở lại những thông báo này và Ann không thể biết nội dung của chúng.



Hình 5.17. Thủ tục chơi bài với phép mã hoá

Bây giờ Ann đã nhận được 5 thông báo được mã một lần  $E(c_j, K_A)$  và 5 thông báo được mã bằng hai khoá  $E(E(c_j, K_A), K_B)$ . Ann sẽ mở khoá 5 thông báo cần giữ bằng cách giải mã 5 thông báo  $E(c_j, K_A)$ . Cô ta cũng giải mã cho 5 thông báo mà Bill đã chọn bằng cách tạo ra  $D(E(E(c_j, K_A), K_B), K_A)$ . Nếu phép mã hoá và giải mã là giao hoán thì:

$$D(E(E(c_j, K_A), K_B), K_A) = E(D(E(c_j, K_A), K_A), K_B) = E(c_j, K_B)$$

Như vậy, các thông báo này bây giờ chỉ còn được "gắn" bởi khoá của Bill. Sau đó, Ann sẽ gửi lại những thông báo này cho Bill để giải chúng và tiếp tục trò chơi. Trên hình 5.17, các thông báo được xem là những quân bài. (Trên thực tế, thủ tục này có một yếu điểm nếu Ann chỉ cung cấp cho Bill 10 thông báo để lựa chọn. Hãy xem các bài tập ở cuối chương để thấy rõ).

#### 5.2.4.3. Áp dụng khoá công khai

Ta sử dụng  $D()$  để ký hiệu cho phép biến đổi riêng (bí mật) và  $E()$  để ký hiệu cho phép biến đổi chung (công khai). Trước tiên, Ann sẽ "khóa" các thông báo bằng phép biến đổi

công khai; sau đó Bill sẽ khoá tiếp các thông báo được chọn bằng phép biến đổi công khai của mình. Ann sẽ mở khoá công khai của mình để thu được thông báo được chọn.

#### 5.2.4.4. Phân phối khoá: Một ứng dụng của thủ tục

Thủ tục không bị hạn chế bởi tình huống khá phi thực tế là phân phối các quân bài bằng máy tính. Thay vào đó, ta xét một hệ thống mã hoá yêu cầu các khoá đặc biệt để không phải bất cứ số nào cũng là một khoá chấp nhận được. (Một số ứng dụng của DES đòi hỏi mỗi bit thứ 8 của khoá phải là bit kiểm tra chẵn của 7 bit đứng trước. Điều này dễ dàng được thực hiện với một khoá 56 bit bất kỳ. Tuy nhiên, điều này khác với điều một số nhị phân 56 bit bất kỳ có thể là khoá. Như đã mô tả trong chương 4, các thuật toán RSA, Merkle - Hellman và ElGamal đều yêu cầu một cặp khoá đặc biệt chỉ có thể do người sử dụng thuật toán xác định).

Nhìn chung, những người sử dụng không thể hoặc không muốn tự mình tạo khoá. Bởi vậy, việc cung cấp khoá cho người dùng cũng là một vấn đề. Một giải pháp là phải có một trung tâm để tạo và phân phối các khoá. Tuy nhiên, lại có những lo lắng xác đáng về việc giữ bí mật cho các kho giữ và gán các khoá.

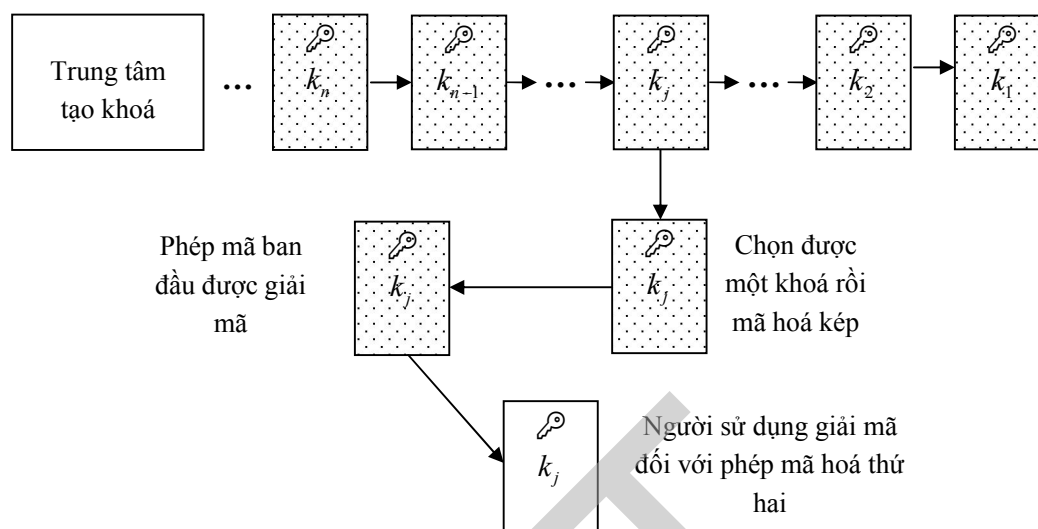
Cần phải có một thủ tục cho phép người dùng nhận được một khoá mới trong khi không một ai (kể cả kho lưu trữ) biết được khoá mà người dùng đã nhận là khoá nào. Ở đây có hai vấn đề:

- Không một ai có thể xác định được khoá riêng của người dùng cho ra khoá công khai của mình. Ta không thể xuất bản một "thư mục khoá" và cho phép mọi người chọn khoá ngẫu nhiên. Nếu có một danh bạ như vậy thì người ta có thể tìm một khoá riêng phù hợp với một khoá công khai ngay khi một người sử dụng chọn một cặp khoá và đưa ra khoá công khai. Bởi vậy, tính riêng tư của một khoá riêng phải được đảm bảo khi người sử dụng lựa chọn một khoá mới.
- Trung tâm phân phối khoá phải không biết khoá mà người dùng đã chọn là khoá nào? Trung tâm không chỉ đơn giản là phân phối một cặp khoá cho một người dùng, người dùng phải có một sự lựa chọn nào đó để trung tâm không thể biết chắc khoá nào là khoá mà người dùng đã chọn. Điều quan trọng là ta muốn trung tâm phân phối khoá đặt các khoá trong các "phong bì" được gắn kín trong một "thùng chứa" và để cho mỗi người dùng nhặt một phong bì.

Một trung tâm phân phối khoá tốt phải là một nguồn trung tâm tạo ra và mã hoá các khoá (giống như Ann trong ví dụ trước). Ann sẽ đưa ra một dòng liên tục các khoá mã hoá đã được mã. Vì các khoá đều được mã nên không ai có thể xác định được các khoá này.

Một người sử dụng riêng muốn có một khoá mới (giống như Bill) sẽ cho phép một ít khoá đã mã hoá của Ann chuyển tới và rồi chọn ngẫu nhiên một khoá  $E(k_i, K_A)$ . Sau đó (sau khi đã có đủ số khoá để Ann không thể biết được khoá mà Bill đã chọn), Bill sẽ mã hoá khoá đã chọn để tạo ra  $E(E(k_i, K_A), K_B)$ . Bill sẽ gửi khoá đã được mã hoá hai lần này cho Ann. Ann sẽ giải mã khoá này để bây giờ nó chỉ còn được mã bởi khoá của Bill và gửi nó trở lại cho Bill. Cuối cùng, Bill sẽ giải mã để sử dụng khoá này. Quá trình này tương tự như thủ

tục chơi bài đã được mô tả ở H5.18. Cần chú ý rằng đối với hệ thống mã khoá công khai, mỗi khoá từ Ann thực sự là một cặp khoá  $E((k_{P \cup B}, k_{PRIV}), K_A)$ .



Hình 5.18. Thủ tục phân phối khoá mật

### 5.2.5. Bỏ phiếu bằng máy tính

Một vấn đề tương tự xoay quanh việc phát một thông báo xác thực nhưng không xác định được dấu vết. Chẳng hạn, cho phép một người (tham gia vào một thí nghiệm) trả lời một câu hỏi kín theo cách vô danh tính. Trong một trường hợp khác, các cử tri có thể phải được lập danh sách bằng máy tính. Một ví dụ thứ ba có liên quan đến các giao dịch riêng tư tự động (chẳng hạn như các tài khoản trong ngân hàng Thụy Sĩ), trong đó việc chuyển tiền là xác thực nhưng không thể truy tìm trở lại được người tạo giao dịch.

Trong cả ba trường hợp này ta cần có một thủ tục để đảm bảo bí mật và hợp thức. Mỗi thông báo trong hệ thống phải xuất phát từ một người sử dụng hợp pháp nhưng không một người sử dụng nào có thể gắn kết được với một thông báo riêng một khi nó được phát. Các yêu cầu bí mật của thủ tục này bao gồm:

- Chỉ có những người dùng hợp pháp mới có thể phát các thông báo.
- Ở một thời điểm, mỗi người dùng chỉ có thể phát một thông báo.
- Không một ai có thể xác định được ai là người đã gửi một thông báo riêng.

Demilo và Merritt đã thiết kế một thủ tục như vậy. Để chỉ rõ hoạt động của một thủ tục, ta sẽ xét một ví dụ bỏ phiếu. Giả sử có ba cử tri là Jan, Keith và Lee bỏ phiếu "Có" hoặc "Không" cho một vấn đề tranh cãi nào đó. Giả sử mỗi cử tri có hai hàm mã hoá khoá công khai. Hàm thứ nhất  $E$  là một hàm mã hoá công khai thông thường nhưng hàm thứ hai,  $R$ , sẽ gắn thông báo trong một xâu ngẫu nhiên và rồi mã hoá kết quả.  $D$  là hàm giải mã cho  $E$ , còn  $Q$  là hàm giải mã cho  $R$ . Vì có nhiều lớp hàm nên ta sẽ ký hiệu  $E_U$  là phép biến đổi của cá nhân  $U$ .



Hai người có thể có phiếu bầu như nhau. Thủ tục này phải cho phép mỗi người dùng biết được phiếu bầu của mình nhưng không thể biết được phiếu bầu của các cử tri khác cho dù có cùng ý kiến. Phép mã hoá  $R$  với xâu ngẫu nhiên phải cung cấp độ bảo mật mong muốn và khả năng phân biệt. Về hình thức, các phiếu bầu được gửi kín ví hai phiếu như nhau đều được gắn với các xâu ngẫu nhiên khác nhau. Tuy nhiên, mỗi cử tri đều có thể xác định đúng phiếu bầu của mình. Vì mỗi phép giải mã  $R$  của mỗi người là của riêng từng người nên chỉ có người gửi mới có thể xác nhận rằng phiếu của mình có nằm trong số phiếu được tính hay không?

#### 5.2.5.1. Thủ tục bỏ phiếu

Mỗi cử tri chọn một phiếu  $v$  và tính:

$$R_J \left( R_K \left( R_L \left( E_J \left( E_K \left( E_L (v) \right) \right) \right) \right) \right) \right)$$

bằng cách dùng các phép mã hoá khoá công khai. Tất cả các phiếu đã mã hoá sẽ được gửi cho cử tri thứ nhất là Jan. Jan sẽ kiểm tra xem lá phiếu của mình có nằm trong số các phiếu đã nhận hay không? Jan sẽ giải mã cho mức mã hoá đầu tiên cho tất cả các lá phiếu:

$$Q_J \left( R_J \left( R_K \left( R_L \left( E_J \left( E_K \left( E_L (v) \right) \right) \right) \right) \right) \right) = R_K \left( R_L \left( E_J \left( E_K \left( E_L (v) \right) \right) \right) \right)$$

vì  $Q_J (R_J (x)) = x$

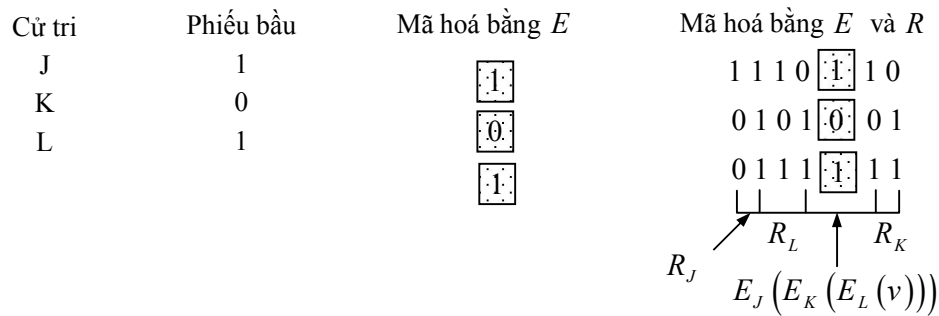
Bây giờ Jan sẽ gửi các lá phiếu tới Keith. Keith sẽ kiểm tra lá phiếu của mình và giải mã mức một để tạo ra:

$$Q_K \left( R_K \left( R_L \left( E_J \left( E_K \left( E_L (v) \right) \right) \right) \right) \right) = R_L \left( E_J \left( E_K \left( E_L (v) \right) \right) \right)$$

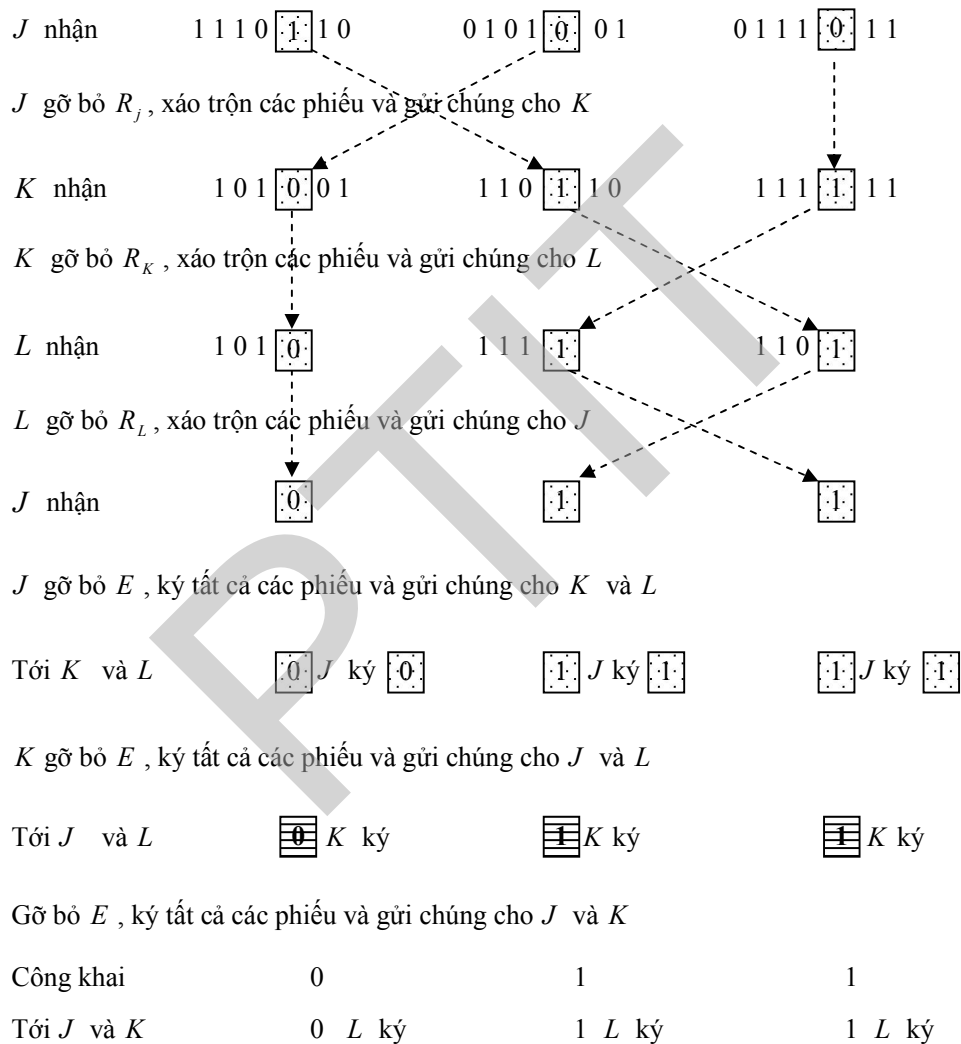
Keith sẽ gửi kết quả này cho Lee. Lee sẽ gửi  $E_J (E_K (E_L (v)))$  của tất cả các  $v$  cho Jan. Lee cũng ký các biểu quyết bằng một chữ ký số và gửi chữ ký này cho Jan và Keith.

Jan sẽ giải mã thêm một mức nữa và kiểm tra để thấy rằng biểu quyết của mình là vẫn còn nằm trong tập. Jan sẽ gửi các lá phiếu cho Keith và gửi các chữ ký số của các lá phiếu cho Keith và Lee. Keith sẽ thu được  $E_K (E_L (v))$  từ Jan. Từ đó, Keith sẽ giải mã để tạo ra  $E_L (v)$  cho Lee. Lee sẽ giải mã  $E_L$  và công bố các kết quả.

Thủ tục này sẽ làm việc vì quá trình phân tích là mỗi chuỗi 6 liên kết:  $J \rightarrow K \rightarrow L \rightarrow J \rightarrow K \rightarrow L$ . Các kết quả ở mỗi kết nối của chuỗi có thể được làm công khai mà không phá huỷ tính vô danh tính của một lá phiếu bất kỳ. Hơn nữa, trong ba liên kết cuối của chuỗi, bất cứ ai cũng có thể đi "ngược" trở lại nhưng chỉ có một người có thể đi "xuôi". Tức là, giả sử Jan biến đổi các lá phiếu và chuyển chúng cho Keith. Chỉ có Jan mới có thể thực hiện phép biến đổi này. Nhưng Keith hoặc bất cứ một ai khác có thể thực hiện phép biến đổi ngược ( $J \leftarrow K$ ) để xem liệu cái mà Jan đã chuyển cho Keith có phù hợp với cái mà Jan đã bắt đầu xử lý. Điều này nhằm kiểm tra công việc của nhau và ngăn chặn sự gian lận của bất cứ ai.



$J$ ,  $K$  và  $L$  tính các phiếu đã mã hoá của họ;  $K$  và  $L$  gửi các lá phiếu của mình cho  $J$ .



Hình 5.19. Thủ tục bỏ phiếu

### 5.2.5.2. Phân tích thủ tục

Ta sẽ phân tích mỗi bước trong thủ tục này. Trước tiên, ta thấy không có thông tin nào để gắn một lá phiếu với một cá nhân nào và như vậy, các lá phiếu vẫn giữ được bí mật. Thứ hai là mỗi cá nhân chỉ có thể bỏ phiếu được một lần vì chỉ có 3 lá phiếu và mọi người sẽ chứng thực ở vòng đầu tiên là lá phiếu của mình nằm trong 3 lá phiếu này. Cuối cùng, không

một ai ngoài 3 người có thể bỏ phiếu được vì một trong 3 cử tri sẽ thấy được rằng mình đã bị mất phiếu.

Giả sử ở vòng thứ hai có một người nào đó quyết định can thiệp bất chính vào một lá phiếu. Một khi kết quả đã được công bố thì 3 cử tri có thể giải mã các phiếu kết quả bằng  $R$  để kiểm tra rằng các phiếu này phù hợp với tập đã gửi ban đầu. Bởi vì mỗi một cử tri đều ký cho tập các lá phiếu cần được chuyển tiếp nên có thể xác định được ai đã bóp méo các kết quả.

Hình 5.19 mô tả một ví dụ về thủ tục này. Để đơn giản các phép mã hoá kiểu  $E$  được mô tả bằng các vạch trực tiếp còn các phép mã hoá kiểu  $R$  được mô tả bằng cách Jan thêm vào 1 bit ngẫu nhiên về phía trái, Keith thêm vào 2 bit ngẫu nhiên về phía phải và Lee thêm vào 3 bit ngẫu nhiên về phía trái.

### 5.2.6. Chuyển giao không nhớ

Thủ tục sau đây là một công cụ sẽ được sử dụng trong nhiều thủ tục phức tạp hơn sau này. Ở đây, bài toán xoay quanh việc gửi một trong hai thông báo với một hạn chế là không một ai (kể cả người gửi và người nhận) biết được thông báo nào đã được gửi. Một ví dụ của bài toán này là việc gieo đồng tiền từ xa. Một người (người gửi) sẽ tung đồng tiền và viết kết quả (xấp hoặc ngửa). Người khác (người thu) sẽ viết ra dự báo kết quả. Sau đó hai người sẽ gặp nhau và trao đổi các biên bản. Khi đó, nếu các kết quả trên các biên bản giống nhau thì người thu sẽ thắng, ngược lại là người gửi sẽ thắng.

Giả sử rằng hai người không thể gặp nhau để trao đổi các biên bản. Hãy thử hình dung tình hình sau: Pete và Nancy rủ nhau đi chơi vào buổi tối. Trên điện thoại. Pete đề nghị tung một đồng tiền, nếu xuất hiện mặt ngửa thì Pete sẽ trả tiền ăn và tiền vé xem phim, nếu xuất hiện mặt sấp thì Nancy sẽ trả. Nancy trả lời rằng ý kiến đó rất hay đối với cô ta. Pete biết rằng nếu Nancy tung đồng tiền thì cô ta sẽ báo rằng "mặt ngửa xuất hiện" bất chấp kết quả ra sao; Pete biết điều đó vì anh ta cũng sẽ xử sự như vậy. Tuy nhiên, họ đã quyết định qua điện thoại để người còn lại phải ra ngân hàng nhận tiền cho buổi tối.

Vấn đề này được gọi là vấn đề chuyển giao không nhớ. Pete muốn gửi một trong hai thông báo tới Nancy với xác suất xuất nhận của mỗi thông báo là 0.5. Thông báo đầu tiên của Pete có thể là "Tôi sẽ trả. Pete". Nếu Nancy chỉ cho anh ta thông báo này thì anh ta sẽ có trách nhiệm chi trả, thông báo khác mà Nancy có thể nhận là một sự bóp méo vô nghĩa, trong trường hợp này cô ta sẽ phải chi trả. Nếu xác suất nhận được của mỗi một trong các thông báo của Pete là 0.5 thì nó tương đương với việc tung đồng tiền một cách thông minh.

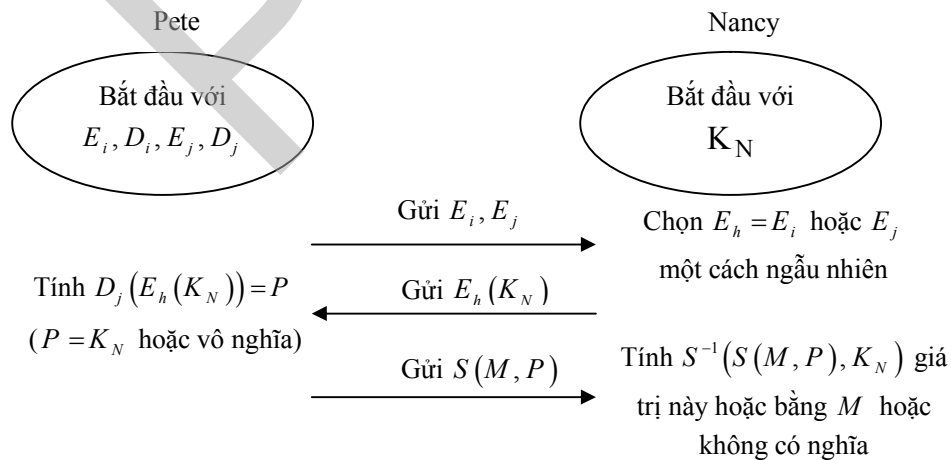
#### 5.2.6.1. Thủ tục chuyển

Thủ tục ban đầu do Blum đưa ra [12]. Giải pháp của Even [13] đưa ra rất dễ hiểu, mặc dù nó cần có một số khoá mã hoá. Sau đây là các bước của thủ tục:

1. Pete chọn hai cặp khoá mã hoá công khai (tổng cộng là 4 khoá). Ta ký hiệu các khoá này là  $E_i, D_i, E_j$  và  $D_j$ .  $E_i$  là một phép biến đổi công khai với khoá  $i$  và  $D_i$  là phép biến đổi riêng tương ứng của nó. (Tức là  $D_i(E_i(M)) = M$  với thông báo  $M$  bất kỳ).

2. Nancy chọn một khoá  $K_N$  cho một thuật toán mã hoá đối xứng  $S$ .
3. Pete gửi cả hai khoá công khai  $(E_i, E_j)$  cho Nancy và giữ lại các khoá riêng.
4. Nancy chọn ngẫu nhiên một khoá và gửi  $E_h(K_N)$  cho Pete, tức là cô ta mã hoá khoá đối xứng  $K_N$  của mình bằng một khoá công khai  $E_h$ .
5. Pete chọn ngẫu nhiên hoặc  $i$  hoặc  $j$  (giả sử Pete chọn  $j$ ), Pete tính  $P = D_j(E_h(K_N))$ . Đối với Pete thì đây là một khâu nhị phân, bởi vậy anh ta không thể biết liệu  $j$  có phải là giá trị mã Nancy đã chọn hay không?. Nếu  $h = j$  thì  $P$  là khoá  $K_N$  của Nancy, ngược lại thì  $P$  là một khâu nhị phân vô nghĩa.
6. Pete tính  $S$  ("Tôi sẽ trả. Pete",  $P$ ) rồi gửi cho Nancy, cùng với giá trị  $j$ .
7. Nancy giải mã thông báo của Pete bằng  $K_N$ . Cô ta đã chọn  $h$  và Pete đã chọn  $j$ . Nếu  $h = j$  thì cô ta thắng, khi giải mã đúng thông báo của Pete, cô ta sẽ gửi  $M$  và  $h$  cho Pete để chứng thực rằng cô ta đã chọn đúng. Nếu  $h \neq j$  thì cô ta thua, kết quả giải mã chỉ là một xâu bit vô nghĩa.
8. Sau khi người thắng cuộc đã được xác định, Pete sẽ trao các khoá riêng là  $D_i$  và  $D_j$  cho Nancy. Từ các khoá này, cô ta có thể kiểm tra xem  $h$  đã chọn là  $i$  hay  $j$  và xem liệu  $j$  mà Pete đã dùng có phải là một trong hai giá trị này hay không? Nếu  $h = j$  và cô ta không nhận được thông báo thì cô ta có thể kết luận rằng Pete đã ăn gian.

Các bước của thủ tục này được mô tả trên hình 5.20.



Hình 5.20. Chuyển không nhớ

### 5.2.6.2. Phân tích thủ tục

Thủ tục này sẽ làm việc như thế nào? Trước tiên, Nancy chọn một trong các khoá của Pete, cô ta sẽ "ngụy trang" nó và trả lại cho Pete. Pete không biết khoá mà Nancy đã chọn bởi vì anh ta không thể biết  $K_N$ . Bởi vậy, khi anh ta chọn  $j$  thì cơ may để anh ta chọn đúng khoá

mà Nancy đã chọn chỉ là 1 trong 2. Anh ta sẽ mã hoá thông báo mà Nancy phải giải đúng để thuyết phục anh ta là anh ta đã chọn đúng khoá của cô ta. Nếu họ cùng chọn một khoá thì khoá  $K_N$  của Nancy sẽ giải mã được thông báo, ngược lại thì  $K_N$  sẽ tạo ra một khâu vô nghĩa. Sau đó, khi gặp nhau họ có thể trao đổi các khoá và kiểm tra được mỗi bên có tuân theo đúng các quy tắc của thủ tục hay không?

Dĩ nhiên là việc tung đồng tiền qua điện thoại không phải là một công việc rất quan trọng. Tuy nhiên, thủ tục này là cơ sở cho 2 thủ tục mà ta sẽ xem xét dưới đây.

### 5.2.7. Ký thoả thuận

Giả sử rằng Charles và Diane cùng thống nhất với nhau về một vấn đề nào đó và muốn ký một thoả thuận để xác nhận sự đồng ý của họ. Cả hai phải cam kết thực hiện một công việc nào đó theo thoả thuận nhưng người nào cũng chỉ muốn cam kết khi người khác cũng làm như vậy. Ví dụ, Charles cam kết bán xe của anh ta cho Diane nếu Diane đồng ý trao cho anh ta một cổ phần trong việc kinh doanh bán bánh pizza của cô ấy.

Charles ở California và Diane ở New York. Charles sẽ không ký thoả thuận trước và gửi nó cho Diane, bởi vì điều đó sẽ đặt số phận của anh ta vào bàn tay của Diane: anh ta đã đồng ý bán xe, bởi vậy anh ta phải đưa nó ra khỏi gian hàng. Tuy nhiên, Diane có thể xé bản thoả thuận để mặc Charles với chiếc xe; mặt khác Diane có thể ký thoả thuận bắt buộc Charles phải sản xuất xe. Charles sẽ bị ràng buộc bởi thoả thuận ngay khi anh ta ký. Tuy nhiên, anh ta không biết liệu Diane có ký hay không, bởi vậy, cô ta không bị ràng buộc bởi bản thoả thuận. Tình hình cũng tương tự nếu Diane ký trước: cô ta sẽ bị ràng buộc còn Charles thì không.

Trong thực tế, điều này được khắc phục khi cả hai bên cùng ký. Ở đây chúng ta muốn thiết lập một thủ tục để ký các thoả thuận bằng máy tính. Giải pháp này nhằm tránh phải gặp gỡ nhau và ký trên giấy.

Có một giải pháp phải nhờ tới một bên thứ 3 tin cậy: Charles và Diane, mỗi người phải ký vào một bản sao của bản thoả thuận để gửi cho một người thứ 3 tin cậy, người này sẽ giữ hai bản sao có một chữ ký này. Đây là bằng chứng chứng tỏ rằng họ đều muốn ký thoả thuận. Khi bên thứ 3 công bố rằng anh ta đã nhận được các bản sao được ký từ mỗi bên thì Charles sẽ ký một bản sao khác để gửi cho Diane và Diane cũng ký vào một bản sao khác để gửi cho Charles. Charles và Diane phải báo cho bên thứ 3 biết và bên thứ ba sẽ huỷ các thoả thuận chỉ có một chữ ký. Tuy vậy, ta vẫn muốn không phải sử dụng tới bên thứ 3 nếu có thể.

Bởi vậy, một thủ tục thoả thuận cần có hai điều:

- Cam kết. Sau một thời điểm nhất định, cả hai bên sẽ bị ràng buộc bởi bản thoả thuận cho tới một thời điểm nào đó.
- Tính không thể giả mạo. Các chữ ký trên bản thoả thuận phải chứng tỏ được là xác thực. Tức là mỗi bên phải có khả năng chứng minh rằng chữ ký của bên còn lại là xác thực.

Thủ tục phải có khả năng thu được hai vấn đề nêu trên một cách gián tiếp, không cần phải mặt đối mặt. Một thủ tục gián tiếp là thích hợp trong tình huống đã được máy tính hoá.

### 5.2.7.1. Với giấy và bút

Một thủ tục gián tiếp có thể hoạt động theo cách sau. Thủ tục này có thể không được sử dụng trên thực tế, nhưng nó tương tự với một thủ tục máy tính được mô tả ở phần sau. Charles và Diane sẽ chuyển tới lui ít nhất 3 bản thoả thuận. Mỗi người giữ một bản làm bằng chứng về sự việc đã xảy ra còn một bản đang được chuyển. Việc ai giữ bản sao nào cho ai để giữ một bản làm bằng chứng là rất rõ ràng, bởi vậy ta sẽ không mô tả các bản sao khác nhau lưu chuyển tới lui.

Charles sẽ viết chữ cái đầu tiên trong tên của mình lên bản thoả thuận là chữ "C" và gửi nó cho Diane. Cô ta sẽ viết chữ "D" và gửi trở lại. Charles sẽ viết một chữ cái tiếp theo trong tên của mình "h" và cứ như vậy họ tiếp tục thực hiện trao đổi bằng cách mỗi lần gắn thêm một chữ cái trong chữ ký. Chỉ với chữ "C" trên bản thoả thuận, Charles biết rằng không một ai có thể yêu cầu anh ta thực hiện bản thoả thuận. Tuy nhiên, mỗi chữ cái là một hành động chứng tỏ sự trung thực mà anh ta muốn tỏ cho Diane thấy. Diane cũng sẽ đáp ứng bằng những hành động tương tự.

Sau một vài chữ cái trên mỗi chữ ký, Diane biết rằng cô ta có thể thuyết phục Quan toà là Charles thực sự đã bị ràng buộc bởi bản thoả thuận. Cô ta giải thích các tình huống và hy vọng rằng Quan toà sẽ đồng ý. Ở đây vẫn còn có một độ bất định: Charles không bị ràng buộc bởi các chữ cái "C" mà anh ta chỉ bị ràng buộc bởi chữ ký đầy đủ của mình. Tuy nhiên, thời điểm mà anh ta bắt đầu bị ràng buộc là không rõ ràng: khi có  $\frac{1}{2}$ ,  $\frac{2}{3}$  hay  $\frac{3}{4}$  số chữ cái trong chữ ký?

Bởi vì họ không biết chắc đúng vào thời điểm nào họ bị ràng buộc nên cả hai đều lo ngại họ sẽ bị ràng buộc ở mỗi lần, bởi vậy họ không có lý do nào để không tiếp tục thủ tục. Họ cũng biết rằng càng tiếp tục thì xác suất chứng tỏ rằng bên kia cũng bị ràng buộc càng lớn, bởi vậy mối quan tâm lớn nhất của họ là phải tiếp tục công việc. Ngoài ra, cả hai đều muốn ký thoả ước bằng bất cứ cách nào nhưng phải đảm bảo họ bị ràng buộc cùng lúc. Tính không chắc chắn của điểm cam kết là cơ sở của giải pháp máy tính.

### 5.2.7.2. Tóm lược về thủ tục gián tiếp

Thủ tục máy tính làm việc khá giống cách trên. Các bên trao đổi thông báo thoả thuận theo các mẫu bằng cách sử dụng thủ tục chuyển không nhớ. Cần nhớ lại là việc chuyển không nhớ là cách phát một trong hai thông báo để không một ai (cả người phát lẫn người nhận) biết chắc thông báo nào đã được phát. Bởi vậy, không một bên nào biết được một nửa thông báo đã được phát bởi vì với việc chuyển không nhớ, không một ai biết chính xác cái mà người khác đã nhận được.

Giả sử mỗi chữ ký của mỗi người được chia thành 4 khối và không thể ghi nhận chữ ký bằng cách nhìn vào nó. Charles và Dianne thực hiện việc chuyển không nhớ mỗi người 8 lần Charles sẽ gửi mỗi khối một lần theo  $i$  và một lần theo  $j$ . Diane cũng làm như vậy.

Trong 3 lần đầu, Charles cảm thấy an toàn vì Diane không có đủ 4 khối trong chữ ký của anh ta. Charles sẽ gửi khối thứ 4. Với khối này, Diane có thể có một chữ ký đầy đủ nhưng cô ta không thể biết điều đó. Trên thực tế, xác suất để cô ta nhận đủ 4 khối của chữ ký là khá

thấp (nhỏ hơn 0.02). Nếu cô ta dừng lại thì Charles có thể đã có  $\frac{3}{4}$  chữ ký của cô ta và có thể thuyết phục Quan toà rằng Diane đã phải bội lại thoả thuận. Diane không có sự lựa chọn nào khác ngoài việc tiếp tục. Khi họ tiếp tục công việc, họ sẽ ngày càng tiến gần hơn tới việc đảm bảo thoả thuận.

Ở một thời điểm bất kỳ sau 4 vòng không một ai có thể dừng lại, bởi vì cô ta hoặc anh ta có thể đã phát đi một chữ ký đầy đủ và bởi vậy, đã bị ràng buộc bởi bản thoả thuận. Do đó, điều quan tâm lớn nhất của cả hai là phải tiếp tục công việc cho tới khi kết thúc, là khi các chữ ký đầy đủ đã được trao đổi hoàn toàn xác định.

### 5.2.7.3. Thủ tục ký thoả thuận gián tiếp

Sau đây là thủ tục do Even và các cộng sự đưa ra [13].

- 1) Charles chọn ngẫu nhiên  $2n$  khoá của một hệ mật khoá đối xứng (chẳng hạn, DES). Ta ký hiệu chúng là  $c_1, c_2, \dots, c_{2n}$ . Các khoá được gom thành các cặp sau  $(c_1, c_{n+1}), (c_2, c_{n+2}), \dots$ . Đây chỉ là một cách ký hiệu chứ không có sự phân biệt nào giữa  $c_i$  và  $c_{n+i}$ .
- 2) Với mỗi khoá, Charles tính  $C_i = E(S, c_i)$  đối với một thông báo chuẩn nào đó mà nội dung của nó là không có liên quan. Charles sẽ gửi  $C_1$  tới  $C_{2n}$  cho Diane. Tức là Diane có dạng đã mã hoá của thông báo chuẩn dưới mỗi khoá.
- 3) Charles nhất trí rằng anh ta sẽ bị ràng buộc bởi bản thoả thuận nếu Diane có thể đưa ra cả hai khoá  $c_i$  và  $c_{n+i}$  với  $i$  bất kỳ. (Chú ý: theo mô tả của Even, mỗi  $C_i$  được gọi là một thách đố  $S$  và khoá  $(c_i, c_{n+i})$  được gọi là giải pháp).
- 4) Diane lặp lại 3 bước này theo cách tương tự với các khoá  $d_i$  và các thông báo đã mã  $D_i$ .
- 5) Charles sẽ gửi mỗi cặp  $c_i$  và  $c_{n+i}$ ,  $1 \leq i \leq n$  cho Diane qua thủ tục chuyển không nhớ. Tức là, Charles sẽ gửi hoặc  $c_i$ , hoặc  $c_{n+i}$  cho Diane nhưng cả Charles lẫn Diane đều không biết chắc khoá đã nhận được. Diane cũng làm tương tự với tất cả các  $d_i$  và  $d_{n+i}$  với  $1 \leq i \leq n$ . Tới lúc này, Charles và Diane mỗi người đã có một nửa bí mật của người khác.
- 6) Gọi  $l$  là độ dài của mỗi  $c_i$  hoặc  $d_i$ .

For  $1 \leq j \leq l$  begin

Charles phát bit thứ  $j$  của tất cả các  $c_i$ ,  $1 \leq i \leq n$  cho Diane

Diane phát bit thứ  $j$  của tất cả các  $d_i$ ,  $1 \leq i \leq n$  cho Charles

End

- 7) Giả sử rằng không một ai kết thúc sớm, ở cuối thư mục cả hai đều có tất cả  $l$  bit của tất cả các bí mật của nhau và thoả thuận đã được ký.

Khi bước 6 bắt đầu, cả hai bên Charles và Diane đều có một nửa bí mật của người khác, tuy nhiên Diane vẫn không xác định được khoá  $c_i$  từ thông báo đã mã  $C_i$  của nó cho dù Diane đã

có  $c_{n+i}$ . Vì bước này đang tiếp diễn nên Diane sẽ nhận được 1 bit của  $c_i$ , rồi bit tiếp theo, rồi bit tiếp nữa... Nếu Diane sử dụng một phép tấn công mạnh để xác định  $c_i$  thì mỗi bit cô ta nhận được sẽ làm giảm nhẹ công việc của cô ta đi một nửa.

Ở thời điểm mà cô ta nghĩ rằng công việc cần làm còn lại là đủ dễ dàng để cô ta có thể xác định được  $c_i$  với  $i$  nào đó. Bởi vậy cô ta có thể ngừng việc gửi các bit cho Charles. Cũng vào lúc đó, Charles cũng có một lượng thông tin tương đương về mỗi  $d_i$ . Nếu Diane có thể dễ dàng xác định được  $c_i$  thì Charles cũng có thể dễ dàng xác định được  $d_k$  với  $k$  nào đó. Bởi vậy, thời gian mà Diane cần để gian lận cũng cho Charles thời gian để tìm ra giải pháp của cô ta để cả hai đều bị ràng buộc vào bản thoả thuận.

### 5.2.8. Thư tín được chứng thực

Việc gửi thư tín có chứng thực là vấn đề cuối cùng mà ta sẽ phát triển một giải pháp thủ tục. Giả sử Gina muốn gửi một thông báo cho Hal nhưng cô ta muốn chứng tỏ rằng Hal đã nhận được thông báo. Cô ta không muốn để lại thông báo cho Hal mà không nhận được phúc đáp. Dĩ nhiên là Hal sẽ không ký phúc đáp cho tới khi anh ta thực sự có thông báo. Các bưu tá và trọng tài có thể giải quyết được vấn đề này. Tuy nhiên, như thường lệ, ta muốn có một giải pháp không cần có trọng tài. Giải pháp này rất giống với thủ tục ký thoả thuận mà ta vẫn nêu. Trên thực tế, giải pháp này cũng do Even và các cộng sự đưa ra [13].

Giả sử thông báo mà Gina chuyển tới Hal là  $M$ . Cô ta sẽ phát  $M$  cho Hal ở dạng được mã hoá và sẽ chỉ trao cho Hal khoá mã hoá khi cô ta đã nhận được một thông tin xác nhận thích hợp từ Hal báo về việc anh ta đã nhận được  $M$ . Hal chỉ biết được  $M$  khi anh ta nhận được khoá cho phép anh ta thu được bản rõ  $M$ .

1) Gina chọn ngẫu nhiên  $n+1$  khoá của một hệ mật đối xứng (chẳng hạn DES). Ta ký hiệu chúng là  $g_0, g_1, \dots, g_n$ . Cô ta cũng tính  $g_{n+i} = g_0 \oplus g_i$  với  $1 \leq i \leq n$  (dấu  $\oplus$  chỉ phép hoặc loại trừ - hay phép cộng theo modulo 2).

2) Gina tính  $G = E(M, g_0)$  (mã hoá thông báo  $M$  bằng khoá  $g_0$ ). Cô ta gửi  $G$  cho Hal. Vì với mỗi  $i$ :

$$\begin{aligned} g_{n+i} &= g_0 \oplus g_i \\ g_{n+i} + g_i &= g_0 \oplus g_i \oplus g_i \\ &= g_0 \oplus 0 = g_0 \end{aligned}$$

Bởi vậy Hal có thể xác định được  $M$  nếu biết một cặp  $(g_i, g_{n+i})$  bất kỳ.

3) Gina tính và gửi cho Hal  $G_i = E(S, g_i)$  với  $1 \leq i \leq 2n$  trong đó  $S$  là một thông báo chuẩn nào đó mà nội dung của nó là không quan trọng. Vào lúc này Hal đã có thông báo đã mã  $G$  và một thông báo tiêu chuẩn  $S$  được mã bằng  $2n$  khoá khác nhau của Gina.

4) Hal chọn  $2n$  khoá  $h_1, h_2, \dots, h_{2n}$ . Hal tính  $H_i = E(S, h_i)$  với  $1 \leq i \leq 2n$ . Hal gửi  $H_i$  cho Gina.



5) Hal gửi một thông báo cho Gina nói rằng anh ra sẽ báo việc nhận bản vẽ của  $G$  nếu Gina có thể tạo ra một trong các cặp của Hal ( $h_i$  và  $h_{n+i}$ ) và tất cả các  $g_j$  với  $1 \leq j \leq 2n$ .

Khi nói điều này, Hal nói rằng nếu Gina có thời gian để xác định một trong các cặp khoá của anh ta ( $h_i$  và  $h_{n+i}$ ) thì anh ta cũng có thời gian để xác định một trong các cặp khoá của cô ta ( $g_m$  và  $g_{n+m}$ ). Chừng nào có được một cặp khoá, Hal có thể tính được khoá mã hoá  $g_0$  và thu được thông báo  $M = D(G, g_0)$ .

Điều kiện thứ hai trong tuyên bố của Hal đảm bảo rằng Gina đã xử sự công minh, tức là mỗi cặp đều dẫn ra được  $g_0$ .

6) Cũng như trong thủ tục ký thoả thuận. Gina và Hal sẽ trao đổi các cặp  $(g_1, g_{n+1})$ ,  $(g_2, g_{n+2})$ , ... và  $(h_1, h_{n+1})$ ,  $(h_2, h_{n+2})$ , ... theo thủ tục chuyển không nhớ. Tức là, Hal sẽ nhận, hoặc  $g_1$ , hoặc  $g_{n+1}$  nhưng cả Hal lẫn Gina đều không biết là giá trị nào? Gina sẽ nhận một nửa trong mỗi cặp  $(h_i, h_{n+i})$  mà không biết giá trị nào.

7) Cũng như trong thủ tục ký thoả thuận, giả sử  $l$  là độ dài của mỗi khoá  $g_i$ .

For  $j = 1$  to  $l$  begin

Gina gửi cho Hal bit thứ  $j$  của mỗi  $g_i$  với  $1 \leq i \leq 2n$ .

Hal gửi cho Gina bit thứ  $j$  của mỗi  $h_i$  với  $1 \leq i \leq 2n$ .

end

Khối lượng công việc cần làm để phá mã của người khác cũng giống như trong việc ký thoả thuận. Bởi vậy, tốt hơn cả là tiếp tục công việc. Mặc dù cả hai thủ tục ký thoả thuận và thư tín có chứng thực là khá phức tạp đối với con người nhưng chúng lại dễ dàng thích hợp trong liên lạc máy tính.

### 5.3. SỬ DỤNG MÃ HOÁ NHƯ THỂ NÀO

Trong những phần trước, ta đã nghiên cứu một số ví dụ về các thủ tục để sử dụng trong máy tính. Ta đã coi các thủ tục là các dãy có thứ tự của các bước để giao tiếp giữa hai bên. Các phương pháp đã đưa ra các cách thông minh trong khi vẫn giữ được tính bí mật, tính vô danh tính hoặc tính riêng tư.

Các thủ tục sử dụng mã hoá như một công cụ để thu được tính bí mật, tính xác thực hoặc tính toàn vẹn, mặc dù trọng tâm là nằm ở trên thủ tục và các đặc tính an toàn cần đạt được chứ không phải ở bản thân phép mã hoá. Các thủ tục (không phụ thuộc vào một thuật toán mã hoá riêng nào) chỉ giả định sự tồn tại của một kiểu mã hoá (đối xứng hoặc không đối xứng) có thể có một tính chất chung (chẳng hạn như tính giao hoán).

Trong phần này ta sẽ nghiên cứu việc sử dụng các phép mã hoá trong thực tiễn. Mặc dù các ứng dụng thực tiễn này không hình thức như các thủ tục nhưng chúng sẽ hướng dẫn ta trong việc sử dụng các phép mã hoá để thực hiện các nhiệm vụ trên máy tính. Một số tình huống trong máy tính chỉ liên quan tới một người sử dụng. Ở đây chẳng cần tới một quy tắc

nào vì không có một sự không nhất trí nào về tính đúng đắn hoặc về sự tuân thủ đối với một quy tắc. Trong những trường hợp này, tính bí mật và tính xác thực là những vấn đề quan trọng. Những tình huống này được gọi là các thực tế mã hoá hay các kỹ thuật mã hoá. Chúng là chủ đề của phần còn lại trong chương này.

Chúng ta phải thông thạo về các đặc tính, các ưu nhược điểm của thuật toán mã hoá. Khi đó ta có thể chọn các thuật toán một cách đúng đắn và sử dụng những phương pháp mã hoá thích hợp. Trong phần này ta sẽ xem xét các tiêu chuẩn để phân xét các hệ thống mã hoá riêng. Hai sơ đồ chủ yếu cần xem xét là RSA và DES. Ta cũng nghiên cứu những hạn chế chung của các hệ thống khoá đối xứng và khoá không đối xứng.

Ta biết rằng, Shannon đã đề xuất các tiêu chuẩn chung cho các hệ thống mã hoá. Mặc dù các tiêu chuẩn này đã được xây dựng trước khi phổ cập các máy tính - modem nhưng chúng vẫn còn có khả năng áp dụng vào những vấn đề cần quan tâm đến ngày nay. Ta sẽ nhắc lại các tiêu chuẩn này.

- Mức độ bí mật cần thiết sẽ xác định khối lượng công việc thích hợp để mã hoá và giải mã.
- Tập các khoá không liên quan tới độ phức tạp.
- Ứng dụng phải đơn giản như có thể được.
- Các sai sót trong mã hoá phải không lan truyền và gây nên ngưng trệ những thông tin tiếp sau trong thông báo.
- Kích thước của bản mã phải không lớn hơn kích thước của thông báo gốc.

Tiêu chuẩn đầu tiên là cơ sở của các hệ mật. Tiêu chuẩn thứ hai là quan trọng hơn khi các khoá (và toàn bộ hệ thống mã hoá) được tính bằng tay. Với các máy tính để thực hiện một công việc phức tạp hay tẻ nhạt thì độ phức tạp của việc chọn khoá không phải là vấn đề. Trên thực tế, với các thuật toán toán học như RSA hay ElGamal thì các khoá là rất phức tạp theo nghĩa chỉ có những số nguyên hoặc những cặp số nguyên nhất định mới có thể là khoá. Tuy nhiên, khi việc trao đổi khoá là cần thiết hoặc mong muốn thì việc phân phối khoá tới những người sử dụng cũng là một khó khăn. Ứng dụng một sơ đồ mã hoá vẫn cần thiết phải đơn giản. Tuy nhiên, với việc sử dụng máy tính, các thuật toán trước kia không có khả năng thực hiện thì nay đã trở nên có thể ứng dụng. Việc lan truyền sai được mô tả trong tiêu chuẩn thứ tư vẫn còn phải lưu tâm. Cuối cùng, tiêu chuẩn về kích thước bản mã không còn quan trọng nữa.

Ta sẽ xem xét các tiêu chuẩn này chi tiết hơn trong phần sau.

### **5.3.1. Mức độ bảo mật**

Cuộc tranh luận liên quan tới việc xác nhận lại DES là một ví dụ tốt cho sự thích ứng của bảo mật ngày nay. Việc DES có thể bị thám nếu có đủ thời gian và tài nguyên tính toán vẫn chưa đủ để người ta loại bỏ nó. (Có thể coi DES chính thức bị loại vào cuối năm 1998, Triple - DES tạm thời thay cho DES trong khi chờ đợi chuẩn mới AES ra đời (AES - Advanced Encryption Standard - Chuẩn mật mã tiên tiến) dự kiến đưa ra vào năm 2001, các

thảo luận trên mạng đang thúc tiến việc xây dựng AES có khoá 256 bit, chẳng hạn như các thuật toán RC-6, CAST-256,...).

Từ 1979, Hellman đã chứng tỏ rằng một thám mã có chủ đích có thể giải mã một thông báo được mã bằng DES. Mặc dù giá các chip giải mã ngày càng rẻ nhưng chi phí để xây dựng một máy giải mã DES vẫn còn lớn tới mức khó có thể có một công ty tư nhân nào, hoặc thậm chí một tổ chức tội phạm nào hay một cơ quan chính phủ nào dám bỏ ra. Một số người sử dụng mã hoá chỉ cần giữ được bí mật đối với những thám mã tình cờ chứ không nhất thiết giữ được bí mật đối với các thám mã thực sự muốn phá. Ví dụ, một công ty có thể mã hoá các dữ liệu có liên quan tới một dòng sản phẩm mới để duy trì ưu thế cạnh tranh. Các ngân hàng mã hoá các thông báo để chuyển các tài khoản nhằm tránh các sửa đổi bất hợp pháp và để xác định việc truyền giả mạo các sai chứ không nhằm bảo vệ tính riêng tư. Một người sử dụng mã hoá file chứa chương trình nguồn nhằm tránh các sửa đổi bất hợp pháp.

Trong những trường hợp này và trong nhiều trường hợp khác, giá trị của thông tin được mã hoá phải được cân nhắc với chi phí thám mã. Khi đó, DES vẫn được coi là đủ mức an toàn. Đối với hệ mật RSA, tình hình cũng tương tự như vậy. Kể từ khi được đưa vào năm 1978 tới nay, vẫn chưa có một thách đố nghiêm trọng nào được đưa ra đối với tính an toàn của nó.

Tóm lại, tính bảo mật của một hệ mật mã phải thích hợp với mức độ quan trọng của dữ liệu cần bảo vệ. Người sử dụng các hệ thống mật mã phải xem xét giá trị của dữ liệu cần bảo vệ khi chọn một hệ thống mật mã.

### **5.3.2. Quản lý khoá**

Đối với các hệ thống mã hoá phức tạp hơn, các khoá không thể chỉ được chọn đơn giản một cách ngẫu nhiên. Các hệ thống khoá công khai thường cần 2 khoá trong một cặp được xác định rất cẩn thận. Ngay cả đối với các thuật toán đối xứng, việc tồn tại các khoá yếu cũng không cho phép tạo khoá hoàn toàn ngẫu nhiên. Vì các lý do này nên các khoá thường được tạo từ một trung tâm tạo khoá.

Khi xem xét tới vấn đề trên, việc phân phối khoá có thể là một nhiệm vụ chính. Khi tới lúc cần thay đổi khoá, mỗi một chủ sở hữu khoá phải được thông báo về việc thay đổi khoá và tất cả những người dùng phải bắt đầu sử dụng khoá mới đồng thời (để tránh có một số thông báo lại được mã hoá bằng khoá cũ thay vì bằng khoá mới và nhằm phân biệt giữa chúng).

Việc thay thế khoá cũng là một vấn đề vì các khoá cũ phải được lưu giữ để cho phép giải mã các thông báo (được mã bằng các khoá cũ) được lưu lại.

### **5.3.3. Các khoá bị mất (bị lộ)**

Cả hai hệ thống khoá đối xứng và không đối xứng đều nhạy cảm đối với việc mất khoá, lộ khoá hoặc bị đánh cắp khoá. Cách xử lý duy nhất được biết là phải kiểm tra định kỳ. Người sử dụng phải biết khi nào một khoá có thể bị tổn hại. Nếu điều đó xảy ra thì người sử dụng phải báo cho kho trung tâm hoặc cho tất cả những ai có thể nhận các thông báo từ họ rằng khoá có khả năng đã bị lộ. Tất cả những nơi thu nhận các thông báo từ người sử dụng này phải được biết rằng họ phải nghi ngờ mọi thông báo nhận được sau ngày báo. Họ cũng phải

biết rằng họ không được gửi bất kỳ một thông báo nào nữa được mã hoá bằng khoá công khai của người đã bị lộ khoá.

Thủ tục này (không thực sự là một thủ tục tối ưu) là một thủ tục khá hữu dụng. Một người dùng muốn rút lại một thoả thuận đã ký chỉ cần tuyên bố đã lộ khoá riêng ở thời điểm trước ngày ký thoả thuận. Bởi vì khoá riêng đã được coi là lộ nên bất cứ ai cũng có thể làm như người dùng (đã bị mất khoá).

#### **5.3.4. Độ phức tạp mã hoá**

Ta sẽ xem xét tiếp tới tính phức tạp để thực hiện một phép mã hoá. Có hai vấn đề chính ở đây là độ trễ trước khi thuật toán mã hoá có thể bắt đầu tạo ra bản mã và độ chậm của bản thân thuật toán mã hoá.

##### **5.3.4.1. Độ trễ trước khi mã hoá**

Độ trễ trước khi mã hoá bắt đầu phụ thuộc vào loại mã là mã khối hay mã dòng và kích thước của khối. Như đã mô tả trong chương 2, các mã dòng là thích hợp để thuật toán mã hoá có thể mã hoá ngay mỗi khi có một ký tự xuất hiện. Kém thích hợp hơn một chút là các mã dòng làm việc trên các khối. Mỗi ký tự không thể được mã hoá ngay khi nó xuất hiện nhưng các ký tự mới sẽ được giữ cho tới khi nhận được một khối (ví dụ 8 ký tự 8 bit hay 64 bit đối với DES). Các hàm mã hoá ít thích hợp nhất là các hàm mã toàn bộ thông báo rõ phải được nhận trước khi có thể bắt đầu việc mã hoá.

Nhiều thuật toán mật mã thông thường (bao gồm RSA, Merkle - Hellman, ElGamal, DES và Skipjack) là các mật mã khối. Với DES, một khối gồm 64 bit. Với RSA, không có độ dài khối bắt buộc (mặc dù các khối ngắn sẽ có độ mật hạn chế). Độ dài khối lớn nhất cũng là một tham số lựa chọn đối với người áp dụng.

Các nhà nghiên cứu phát triển RSA khuyến nghị độ dài khối nên từ 100 tới 200 bit. Các độ dài khối của DES và RSA thích hợp cho các ứng dụng trong thương mại.

##### **5.3.4.2. Tốc độ mã hoá**

Một số thuật toán mã hoá thực hiện một khối lượng công việc không đổi đối với một ký tự được mã. (Chẳng hạn, với các mật mã thay thế, tất cả công việc chủ yếu là quá trình tra cứu bảng). DES, Skipjack, RSA, Merkle - Hellman và ElGamal mặc dù là các thuật toán phức tạp hơn nhiều nhưng đều thực hiện cùng một lượng tính toán đối với một khối. Tất cả các thuật toán này hoạt động theo một thời gian tỉ lệ với kích thước của bản rõ, chỉ có hằng số tỉ lệ là khác nhau.

Tốc độ mã hoá là một tham số quan trọng vì nó đảm bảo rằng nếu một thuật toán mã hoá có thể mã hoá xong một khối trước khi nhận được khối thứ hai thì thuật toán này sẽ không làm giảm tốc độ của ứng dụng. Luôn có các ứng dụng phần cứng (chip) đối với hầu hết các thuật toán thông dụng đảm bảo làm việc với các tốc độ hợp lý.

Các thuật toán khoá công khai chậm hơn rất nhiều so với các thuật toán khoá đối xứng. Các thuật toán đối xứng đều dựa trên các phép toán hữu hiệu như các phép tra cứu bảng, dịch, chuyển vị và các phép logic (như VÀ, HOẶC, cộng mod 2...). Tất cả các phép toán này đều có các chức năng phần cứng tiêu chuẩn. Các thuật toán không đối xứng đều xây dựng trên các

bài toán khó. Với các khoá có độ dài càng lớn thì độ phức tạp tính toán càng tăng theo tỉ lệ hàm mũ. Các số liệu trong bảng B 5.1 cho ta thấy rằng mã hoá đối xứng thực hiện nhanh hơn từ 1000 tới 5000 lần so với mật mã khoá công khai.

### 5.3.5. Lan truyền sai

Các sai trong truyền dẫn là chủ yếu và gây tác hại nghiêm trọng. Ta cũng biết rằng các mạng cục bộ hoặc các mạng ở xa thường chịu các sai truyền dẫn. Các sai này có thể không báo cho ta biết có kẻ thu chặn. Các thuật toán mã hoá lý tưởng phải không bị ảnh hưởng của các sai của mạng. Thông thường, mạng sẽ chịu trách nhiệm xác định các cuộc liên lạc bị sai và phát lại các thông báo có lỗi sao cho người dùng không phải quan tâm đến.

Bảng 5.1: Tốc độ mã hoá

Thuật toán	Phần cứng (bits/s)	Phần mềm (bits/s./MIPS)
Mã hoá RSA	220K	0.5
Mã hoá RSA	220K	32K
MD4	220K	1300K
DES	1,26 (1,2. 10 <sup>9</sup> )	400K

Một sai thực sự (do sự thay đổi của thám mã) phải thể hiện rõ trên bản mã sao cho một sự thay đổi bất kỳ đều dễ dàng thấy được. Không có một thuật toán thông dụng nào có cơ chế chống xâm nhập. Nhờ các cơ chế này, thuật toán mã hoá sẽ bị bóp méo tới mức thậm chí chỉ cần thay đổi 1 bit đầu ra cũng sẽ dẫn tới thông báo được giải mã bị bóp méo nghiêm trọng. Nếu bản rõ là một đoạn văn thì sự thay đổi sẽ được thấy rất rõ. Nếu bản rõ là dữ liệu nhị phân thì sự thay đổi có thể không bị phát hiện.

### 5.3.6. Kích thước bản mã

Kích thước của bản mã được tạo thành cũng là một tham số quan trọng. Trong một số trường hợp, bản mã có thể nằm trong cùng một không gian với bản rõ. Khi đó, việc tăng một chút về kích thước cũng là điều không thể chấp nhận được. Cần chú ý rằng các mật mã khối phải mã hoá đồng thời một khối đầy đủ. Mặc dù có thể dễ dàng độn thêm vào khối cuối cùng không đầy đủ để tăng kích thước nhưng việc độn thêm này sẽ làm tăng kích thước của văn bản ở đầu ra. DES và Skipjack tạo ra một đầu ra có kích thước đúng bằng kích thước đầu vào hoặc hơi lớn hơn một chút. (Một vector khởi tạo hoặc các bit độn làm đầy đủ cho khối cuối cùng sẽ làm tăng tối đa là một hoặc hai khối). Trong các thuật toán công khai, đầu ra có quan hệ với kích thước đầu vào nhưng không nhất thiết phải tỉ lệ với kích thước này.

## 5.4. CẢI THIẾN ĐỘ MẬT CỦA HỆ MẬT

DES và RSA được xem là các thuật toán mã hoá an toàn qua nhiều năm nghiên cứu của các nhà thám mã. Các thuật toán Skipjack và ElGamal cũng được coi là an toàn mặc dù chúng chưa được thử thách lâu như DES và RSA. Mặc dù thám mã khó có thể tìm được nội dung của các thông báo được mã bằng các thuật toán trên nhưng việc mô tả những tấn công có khả

năng đối với một hệ mật sẽ chứa đựng nhiều yếu điểm tiềm năng hơn việc phá bí mật của một thông báo. Trong phần này, ta sẽ xem xét một số phương pháp tấn công này.

#### 5.4.1. Ngăn ngừa và phát hiện sai

Trong DES, mỗi khoá làm một thực thể. Một thám mã đã nắm được khuôn dạng của các thông báo của người gửi có thể sửa đổi các thông báo mà không cần phải phá mã.

##### 5.4.1.1. Lặp lại khối

Ví dụ, ta xét tình huống sau. Hai ngân hàng thoả thuận trao đổi điện tử các thông tin về chuyển tiền trao đổi giữa hai ngân hàng. Để bảo mật, họ sử dụng các số liệu được mã hoá. Họ thoả thuận phát các bản ghi có khuôn dạng xác định như sau:

Chủ tài khoản	Số tài khoản	Lượng tiền chuyển
24 bytes	8bytes	8 bytes

Các bản ghi này gồm 40 bytes = 320 bits = 5 khối 64 bit (xem hình 5.21).

Giả sử John có thể thu trộm kênh dữ liệu giữa hai ngân hàng. Ngày đầu tiên, John với tư cách chủ tài khoản sẽ chuyển 100 USD từ một ngân hàng này sang một ngân hàng khác. Ngày tiếp theo anh ta cũng làm như vậy. Ở cả hai ngày, anh ta vẫn đều thu chặn và ghi lại quá trình truyền dẫn từ ngân hàng này tới ngân hàng khác. Giả sử rằng cả hai lần truyền dẫn đều được gửi bằng cùng một khoá mã hoá và cả hai lần đều bắt đầu bằng một bản ghi. John biết rằng cả hai lần truyền đều có 3 khối biểu thị tên của mình, một khối biểu thị số tài khoản và một khối biểu thị lượng tiền chuyển. John cũng biết rằng 5 khối này ở cả hai ngày đều như nhau. Việc xác định dữ liệu trong các trường này chỉ là một quá trình tìm kiếm hai phần giống nhau. Công việc tẻ nhạt này có thể thực hiện dễ dàng bằng máy tính.

	Khối 1	Khối 2	Khối 3	Khối 4	Khối 5
Đã mã hoá các bytes	Tên			Tài khoản	Số lượng
	1		24	2532	3340

Hình 5.21.

Ngày tiếp theo, John kiểm tra giả định của mình bằng cách gửi qua một giao dịch khác với một lượng tiền khác. John lại tìm kiếm 4 khối giống với 4 khối trước đó, tuy nhiên các bản ghi này sẽ khác nhau ở khối thứ 5. Nếu chỉ có một tập các khối phù hợp với mẫu này thì John có dạng đã mã tên, số tài khoản của anh ta.

Bằng cách chèn thêm các số liệu vào đường truyền thay cho việc đọc các thông báo từ nó, John bây giờ có thể thay thế một người bất kỳ và số tài khoản của anh ta bằng tên và số tài khoản của mình, riêng số lượng tiền thì không đụng chạm tới. John không cần biết ai là người nhận tiền và lượng tiền được nhận là bao nhiêu, anh ta chỉ đơn giản là thay vào tên vào tên và số tài khoản của mình và quan sát ngân sách trong tài khoản của mình tăng trưởng. Kỹ thuật này được gọi là kỹ thuật lặp lại khối. Trong kỹ thuật này, các khối đã mã từ một lần truyền

dẫn sẽ được gửi lại ở lần truyền dẫn thứ hai. Để sử dụng kỹ thuật này, người thu chặn không cần phải phá mã.

Ở cuối mỗi lần truyền, các ngân hàng thường gửi tổng các chuyển khoản nhưng John sẽ không can thiệp vào, bởi vậy, tổng số sẽ cân bằng. Nếu John may mắn thì ít nhất là một trong những lần truyền đã sửa đổi sẽ làm anh ta giàu to. Các khách hàng có thể không nhận thấy thiếu một lần chuyển (khi John chuyển sang tài khoản của mình) tới một tháng sau khi John bắt đầu can thiệp. John sẽ tiếp tục trò chơi này trong vòng tối đa 1 tháng và rút tiền mặt ra khỏi tài khoản của mình.

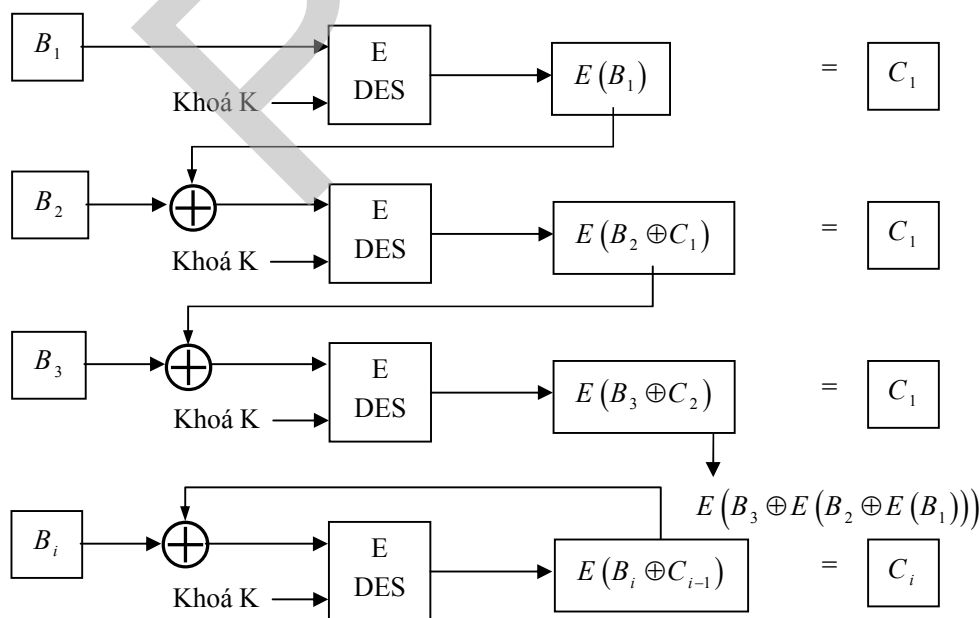
John có thể coi là người trong cuộc theo nghĩa anh ta biết khuôn dạng truyền, tần suất thay đổi khoá, các truyền dẫn này thường xảy ra như thế nào... John cũng biết cách đảm bảo an toàn khi có sự cố. Kỹ thuật được dùng ở đây đã đơn giản hoá đôi chút nhằm làm cho việc giải thích dễ dàng hơn và tránh các hướng dẫn chi tiết cho kẻ xấu lợi dụng.

Do cách xử lý mỗi khối của bản rõ là độc lập nên DES và các mật mã khối khác đều bị ảnh hưởng bởi loại tấn công này. Rất may là có một giải pháp khắc phục dễ dàng, được gọi là giải pháp xích khối.

#### 5.4.1.2. Giải pháp xích khối

Ta biết rằng khi cộng mod 2 một chuỗi nhị phân với chính nó thì kết quả sẽ bằng 0 và phép "hoặc có loại trừ" này là một phép toán giao hoán. Bởi vậy, với 2 chuỗi bất kỳ  $a$  và  $b$ , ta có:

$$\begin{aligned}(a \oplus b) \oplus a &= (a \oplus a) \oplus b \\ &= 0 \oplus b \\ &= b\end{aligned}$$



Hình 5.22. Ví dụ về giải pháp xích khối

Với giải pháp xích khối, mỗi khối cần phát sẽ được kết hợp bằng phép cộng mod 2 với các khối trước. Nếu các khối là  $B_1, B_2, B_3$  và các hàm mã hoá là  $E(\ )$  thì các khối sau sẽ được phát:

$$\begin{aligned} C_1 &= E(B_1) \\ C_2 &= E(E(B_1) \oplus B_2) = E(C_1 + B_2) \\ C_3 &= E(E(E(B_1) \oplus B_2) \oplus B_3) = E(C_2 + B_3) \end{aligned}$$

Quá trình này được mô tả trên hình 5.22.

Bên nhận sẽ giải mã khối đầu tiên nhận được  $C_1$  như thông thường. Bên nhận sẽ giải mã khối thứ hai nhận được (chính là  $C_2 = E(C_1 + B_2)$ ) và thu được  $C_1 + B_2$ . Từ  $C_1$  đã nhận được trước kia, bên thu sẽ tính  $(C_1 \oplus B_2) \oplus C_1$ . Ta có:

$$(C_1 \oplus B_2) \oplus C_1 = (C_1 \oplus C_1) \oplus B_2 = 0 \oplus B_2 = B_2$$

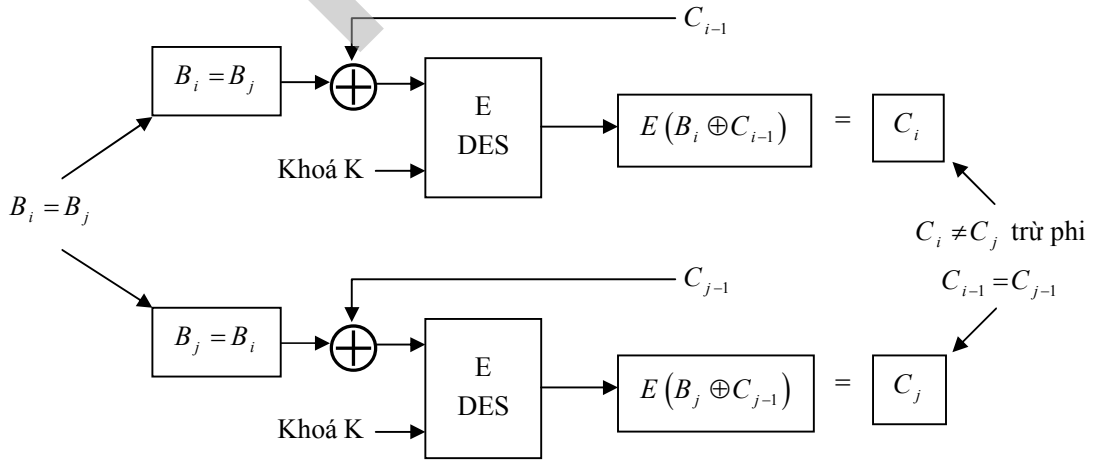
Đây là giá trị rõ của khối thứ hai. Các khối tiếp sau cũng được xử lý tương tự.

Như đã nêu ra trên hình 5.23, ta thấy rằng các khối bản rõ như nhau được phát tách biệt hoặc được phát trong cùng một lần truyền không nhất thiết tạo ra cùng một bản mã. Có sự khác nhau như vậy vì mỗi khối sẽ phụ thuộc vào tất cả các khối ở phần trước. Vì vậy, thám mã không thể lặp lại các khối nhất định từ một đường truyền này sang một lần truyền khác hoặc thậm chí là chỉ tìm các khối xuất phát từ cùng một bản rõ.

Đối với DES, một khối không làm lộ khối khác vì:

$$E(C_1 \oplus B_2) \neq E(C_1) \oplus E(B_2)$$

Thậm chí nếu thám mã biết rằng  $C_2 = E(C_1 \oplus B_2)$  và đã nhận được  $C_1$  thì thông tin này cũng không bị để lộ  $B_2$  hoặc thậm chí chỉ là  $E(B_2)$ .



Hình 5.23. Mã hoá các khối bản rõ như nhau



### 5.4.1.3. Giá trị ban đầu

Chế độ xích khối sẽ che giấu các khối như nhau. Tuy nhiên, đối với một số thông báo một số khối đầu tiên có thể lại tuân theo một khuôn dạng định trước. Với cùng một khoá mã hoá, tất cả các thông báo dạng này sẽ lại tạo ra cùng một bản mã ở đầu ra.

Nếu điều này không thể chấp nhận được thì ta có thể dùng một giá trị ban đầu để nâng cao độ an toàn. Giá trị này là một xâu ngẫu nhiên nào đó được dùng như khối khởi đầu. Chẳng hạn, xâu này có thể là thời gian, ngày tháng hiện thời hoặc là một khối ngẫu nhiên. Xâu này phải khác nhau đối với mỗi thông báo được mã hoá. Cả bên gửi và bên nhận đều phải giải mã khối này nhưng chỉ dùng nó để giải mã cho các khối tiếp theo (khối này không chứa dữ liệu).

Bằng cách sử dụng một vector khởi tạo, thông báo  $B_1B_2B_3\dots$  sẽ trở thành thông báo  $IB_1B_2B_3\dots$ , khối đầu tiên được gửi là  $E(I)$ , trong đó  $I$  là giá trị ban đầu. Khối tiếp theo được gửi là  $E(E(I) \oplus B_1)$ . Thậm chí, nếu hai khối giống nhau nằm trong các thông báo khác nhau (ví dụ  $B_1$ ) thì chúng cũng không tạo ra các bản mã giống nhau (bởi vì  $I$  là khác nhau đối với mỗi thông báo nên  $E(E(I) \oplus B_1)$  sẽ luôn khác nhau).

### 5.4.2. Mã hoá một chiều

Một số phép mã hoá phụ thuộc vào một hàm khó tính toán. Ví dụ, xét hàm lập phương  $y = x^3$ . Có thể dễ dàng tính  $x^3$  bằng tay. Hàm ngược  $\sqrt[3]{y}$  khó tính toán hơn nhiều. Hàm  $y = x^2$  không có hàm ngược vì có hai khả năng đối với  $\sqrt[2]{y}$  là  $+x$  và  $-x$ . Các hàm như vậy (dễ tính hơn nhiều so với các hàm ngược của chúng) được gọi là các hàm một chiều.

#### 5.4.2.1. Sử dụng mã hoá một chiều

Các hàm một chiều đặc biệt hữu ích trong xác thực. Các mật khẩu thường được dùng để kiểm tra xem một người sử dụng đang cố gắng đăng nhập có phải là một người dùng hợp lệ hay không? Việc đưa ra công khai bảng mật khẩu của các người dùng là nguy hiểm đối với hệ thống máy tính, bởi vậy nhiều hệ thống đã sử dụng hàm một chiều để mã hoá bảng mật khẩu. Hệ thống sẽ lưu  $f(pw)$  khi người sử dụng nhận được một khẩu mới  $pw$ , trong đó  $f$  là hàm một chiều. Khi có một người sử dụng đăng nhập thì hệ thống sẽ hỏi mật khẩu. Người sử dụng đưa vào  $t$  và hệ thống sẽ lưu mật khẩu được mã bằng khoá là chính mật khẩu đó.

Hệ thống mật khẩu là an toàn vì không thể không có khả năng tính được  $f^{-1}$ . Thám mã phải tìm bảng mật khẩu của người dùng là  $f(pw)$ . Tuy nhiên, thông tin này không cho phép thám mã suy ra  $pw$ . Hơn nữa, thám mã có thể thử các mật khẩu khác nhau cho đến khi tìm thấy một từ  $w$  sao cho  $f(w) = f(pw)$ . Với các mật khẩu có độ dài lớn được chọn từ một bộ ký tự lớn thì kiểu tấn công này được xem là không khả thi.

#### 5.4.2.2. Niêm phong mật mã

Một tính chất hữu dụng khác của mật mã là có khả năng bảo vệ dữ liệu khỏi những can thiệp bất hợp pháp. Ví dụ, trong một file dữ liệu thông thường, một dữ liệu nào đó, một bản ghi nào đó hoặc toàn bộ file có thể bị sửa đổi mà không bị phát hiện. Ở đây, ta chỉ xem xét tới các file mặc dù những tình huống tương tự cũng có thể áp dụng đối với một bản ghi, một trường hoặc một byte riêng lẻ.

Mã hoá thường dùng để bảo mật. Tuy nhiên, trong một số trường hợp, tính toàn vẹn lại là một yếu tố quan trọng hơn. Ví dụ, trong một hệ thống trích xuất dữ liệu, điều quan trọng là phải biết rằng bản sao được trích xuất là đúng với tư liệu đã lưu trữ. Tương tự, trong một hệ thống liên lạc an toàn, nhu cầu truyền dẫn các thông báo đúng có thể còn vượt trên các nhu cầu về bí mật. Mã hoá có thể đảm bảo được tính toàn vẹn cũng như tính bí mật.

Trong hầu hết các file thường không có sự ràng buộc nào giữa các yếu tố, tức là mỗi byte, mỗi bit hay mỗi ký tự là độc lập với nhau trong file. Việc thay đổi một giá trị sẽ ảnh hưởng tới tính toàn vẹn của file, tuy nhiên điều này rất khó phát hiện được.

Mật mã có thể được dùng để niêm phong một file (giống như bao nó bằng một vỏ chất dẻo) sao cho bất kỳ một sự thay đổi nào đều bị phát hiện. Một trong những kỹ thuật này là tính một hàm mật mã của file (thường được gọi là hàm băm (hash) hay tổng kiểm tra). Trong chương 4, ta đã nghiên cứu một số thuật toán băm. Hàm này phải phụ thuộc vào tất cả các bit của file cần niêm phong sao cho bất kỳ một sự thay đổi nào thậm chí chỉ là một bit riêng lẻ cũng làm thay đổi kết quả tổng kiểm tra.

Giá trị tổng kiểm tra được lưu cùng với file. Sau đó, mỗi khi file được sử dụng thì tổng kiểm tra sẽ được tính lại. Nếu tổng kiểm tra tính được phù hợp với giá trị đã lưu thì điều đó có nghĩa là file đã không bị sửa đổi.

Một hàm mật mã (chẳng hạn DES) là đặc biệt thích hợp cho việc niêm phong cho các giá trị vì các tham số không thể biết phải thay đổi giá trị được lưu như thế nào để phù hợp với dữ liệu cần sửa đổi. Như đã mô tả ở trên, chế độ xích khối của DES sẽ tạo ra một đầu ra mà mỗi khối sẽ phụ thuộc vào giá trị của các khối trước. Tổng kiểm tra mật mã của file có thể là khối cuối cùng trong phép mã hoá DES ở chế độ xích khối vì rằng khối này sẽ phụ thuộc vào tất cả các khối khác.

#### 5.4.2.3. Xác thực

Ta nhớ lại một câu chuyện cổ kể về hai người cắt một đồng tiền làm đôi, mỗi người giữ một nửa và rời ra đi. Anh ta nói rằng: "Nếu có một người đưa tin mang đến cho bạn một cái gì đó kèm theo nửa đồng tiền này thì bạn sẽ biết rằng đó là của tôi". Với mã hoá, ta có thể làm tương tự. Trong giao tiếp cá nhân, khi nói về một người mà chúng ta biết, ta có nhiều cách để đảm bảo rằng người mà chúng ta nói chính là người mà ta đã biết. Ta ghi nhận giọng nói, phong cách, các mẫu ngữ điệu. Trong giao tiếp máy tính, ta không có nhiều chứng cứ để đảm bảo cho ta việc nhận dạng.

Phương thức xác thực chủ yếu là một mật khẩu: đó là một từ hoặc một câu mà chỉ có một người biết. Hệ thống máy tính sẽ coi rằng một người nào đó biết mật khẩu sẽ chính là

người có mật khẩu đó. Như ta sẽ thấy sau này, các mật khẩu có thể coi là an toàn, mặc dù nếu được sử dụng không đúng thì chúng chỉ bảo vệ được chút ít.

Mã hoá là một dạng khác của xác thực. Nếu bạn nhận được một thông báo đã mã hoá mà bạn có thể giải mã bằng khoá chỉ có bạn và người khác biết thì thông báo là xác thực. Trừ phi sơ đồ mã hoá đã bị khoá hoặc mã đã bị lộ, còn không thì thông báo chỉ có thể được tạo bởi người có khoá.

Khẳng định này có thể mở rộng cho toàn bộ thông báo. Với một thuật toán mã hoá mạnh thì không ai có thể thay thế một câu nào đó trong thông báo và cũng không thể dán các mẫu thông báo cũ với nhau. Chẳng hạn, chế độ xích khối mật mã đã xét ở trên không cho phép thay bất kỳ một khối nào trong thông báo. Thông báo đã nhận là được gửi bởi một người đã biết và không thể bị sửa đổi trước khi thu nó.

#### **5.4.2.4. Các nhãn thời gian**

Một vấn đề an toàn khác cần đặt ra là có thể có một thông báo đã bị thu chặn và bây giờ đang được phát lại. (Tình huống này tương tự như việc đổi tiền bằng một bản sao của séc mà người gửi không biết đó là một bản sao). Trong tình huống này, người gửi và người nhận có thể gắn với mỗi thông báo một nhãn thời gian hoặc một số thông báo.

Số thông báo là một con số được gắn vào một thông báo. Thám mã không có cách nào để biết được các bit của số này nằm ở vị trí nào trong thông báo hoặc không thể biết cách thay đổi các bit để tạo ra dạng mã hoá của số tiếp sau hoặc không thể biết cách thay đổi các bit này mà không làm gián đoạn việc giải mã phần còn lại của thông báo. Các số thông báo này không thể bị thay thế, thay đổi hoặc giả mạo. Người nhận phải duy trì việc đếm các số thông báo đã nhận được. Nếu hai người sử dụng một tập các số thì người nhận có thể ngay lập tức biết được liệu có thông báo nào trước thông báo hiện thời đã bị mất hoặc bị chậm trễ vì số được mã hoá của thông báo hiện thời phải lớn hơn số được mã hoá của thông báo trước.

Nếu người gửi có nhiều thông báo thì có thể gặp phải vấn đề số thông báo quá dài. Vì lý do này, người ta thường đặt lại bộ đếm số thông báo khi nó đạt tới giá trị quá lớn (thường là sau khi độ dài của số thông báo vượt quá 30 bit). Trong trường hợp này, tất cả các bên thu phải được thông báo rằng số thông báo được gửi tiếp theo sẽ được đặt lại về một số nhỏ (chẳng hạn là 0).

Còn một vấn đề khác là với mỗi cặp người gửi - người nhận phải có một bộ đếm riêng. Thông thường, người gửi hoặc người nhận phải duy trì nhiều bộ đếm khác nhau cho nhiều đối tượng khác nhau.

Có một vấn đề nữa là một người gửi sẽ có một bộ tạo số thông báo duy nhất cho các thông báo tới mọi đối tượng. Nếu người gửi gửi thông báo cho hai người A và B thì thông báo 1 có thể đi tới A, thông báo 2 tới B, các thông báo 3 và 4 tới A, thông báo 5 tới B...

Không một bên thu nào có thể biết được thông báo nào đã bị mất vì các số thông báo chỉ là một dãy tăng chứ không nhất thiết phải là tất cả các số trong một dải nào đó. Tuy nhiên, bất cứ lúc nào, bên thu cũng có thể yêu cầu phát lại thông báo trước. Số thông báo cần phải được mã hoá hoặc phải được bảo vệ bằng một cách nào đó để ngăn chặn việc sửa đổi.

Các nhãn thời gian và ngày tháng có thể xem là một cách mềm dẻo hơn đôi chút. Chúng là các dấu hiệu về thời gian và ngày tháng mà thông báo được gửi với mức chính xác đủ để cho không có hai thông báo nào có cùng một dấu hiệu. Các nhãn này không phải đặt lại.

Bên thu phải đồng bộ về thời gian rất chặt chẽ với bên gửi. Khi truyền nhanh, nếu các đồng bộ của bên thu và bên phát không đồng bộ thì nhãn thời gian của người gửi có thể chậm hơn thời gian hiện thời của bên nhận. Có thể cho phép có một sai lệch nhỏ đối với thời gian đồng bộ hoặc phải ghi nhận rằng các nhãn thời gian của bên gửi luôn nhanh hơn một chút.

Bây giờ, chúng ta sẽ nghiên cứu các khía cạnh khác nhau của DES trong các ứng dụng riêng trong việc đảm bảo tính toàn vẹn, tránh sử dụng lại và đảm bảo bí mật. Các ứng dụng này có thể thu được từ nhiều hệ thống mã khoá khác bởi vì chúng chỉ phụ thuộc vào những tính chất chung của phép mã hoá.

## 5.5. CÁC CHẾ ĐỘ MÃ HOÁ

Bây giờ chúng ta nghiên cứu một số phương pháp khác nhau mà phép mã hoá có thể sử dụng. Các nguyên tắc chung ở đây có thể áp dụng cho hầu hết các thuật toán mã hoá thông thường (kể cả đối xứng và không đối xứng). Một dạng sử dụng DES đã được trình bày trong chương 3, được gọi là chế độ quyền mã điện tử (ECB). Trong những phần sau ta sẽ trình bày các chế độ sử dụng DES khác.

### 5.5.1. Chế độ xích khối mật mã (CBC)

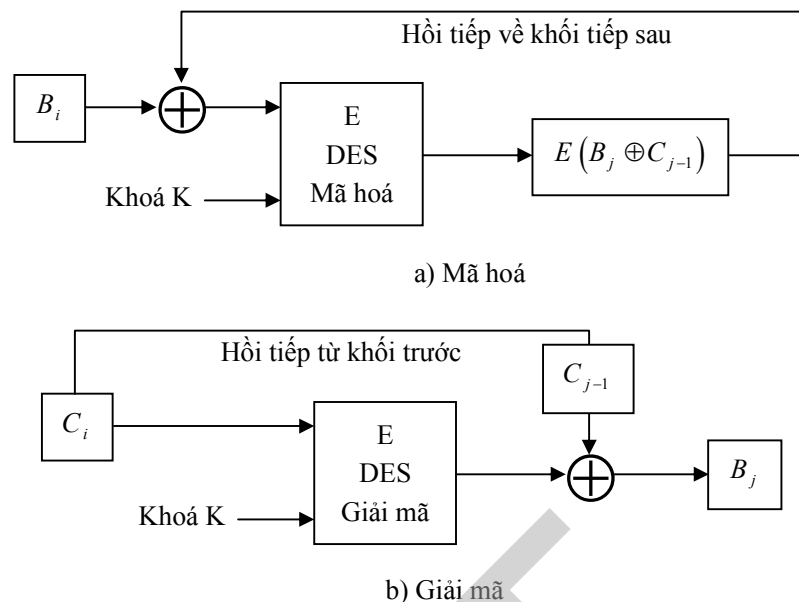
Như đã nêu ở trên, chế độ CBC bắt đầu bằng một vector khởi tạo ngẫu nhiên (IV), vector này được mã hoá và được gửi như khối 1. Sau đó, vector này sẽ được cộng mod 2 với khối bản rõ đầu tiên ( $E(p_1 \oplus IV, k)$ ). Kết quả này sẽ được mã làm khối 2. Sau đó, khối bản mã thứ hai này sẽ được cộng mod 2 với khối bản rõ tiếp theo trong thông báo nhằm không thể thay thế khối bản mã bằng một khối khác mà không bị phát hiện. Chế độ CBC đã được chỉ ra trên hình 5.24.

Chế độ CBC cũng có tính chất tự sửa sao cho một sự thay đổi trong khối  $c_i$  chỉ ảnh hưởng tới việc giải mã cho các khối  $p_i$  và  $p_{i+1}$ . Tuy nhiên, các khối  $p_{i+2}$  và tiếp theo sẽ không bị tác động. Sau hai khối hàm cộng mod 2 sẽ huỷ bất kỳ một sai nào. (Cần nhớ rằng, đối với một khâu bất kỳ  $x$ ,  $x \oplus x = 0$ ). Tính chất tự sửa là thuận tiện vì một sai số trong truyền dẫn hoặc mã hoá sẽ không phá huỷ một lượng lớn bản mã.

### 5.5.2. Chế độ hồi tiếp mật mã (CFB)

Bản chất định hướng theo khối của DES là không thuận tiện với 2 lý do sau.

- Thứ nhất là khối cuối cùng phải được độn thêm. Bởi vậy, kích thước của bản mã kết quả có thể lớn hơn một chút so với bản rõ tương ứng.
- Thứ hai là không thể thực hiện việc mã hoá nếu chưa nhận đủ 64 bit của một khối. Bởi vậy, tốc độ mã hoá cho một ký tự có thể phụ thuộc vào tốc độ nhận của một vài ký tự tiếp sau. Có một số ứng dụng đòi hỏi phải mã hoá ngay từng ký tự. Ví dụ, trong một môi trường mạng an toàn, người dùng phải phát mỗi ký tự như nó được đưa vào từ đầu cuối.

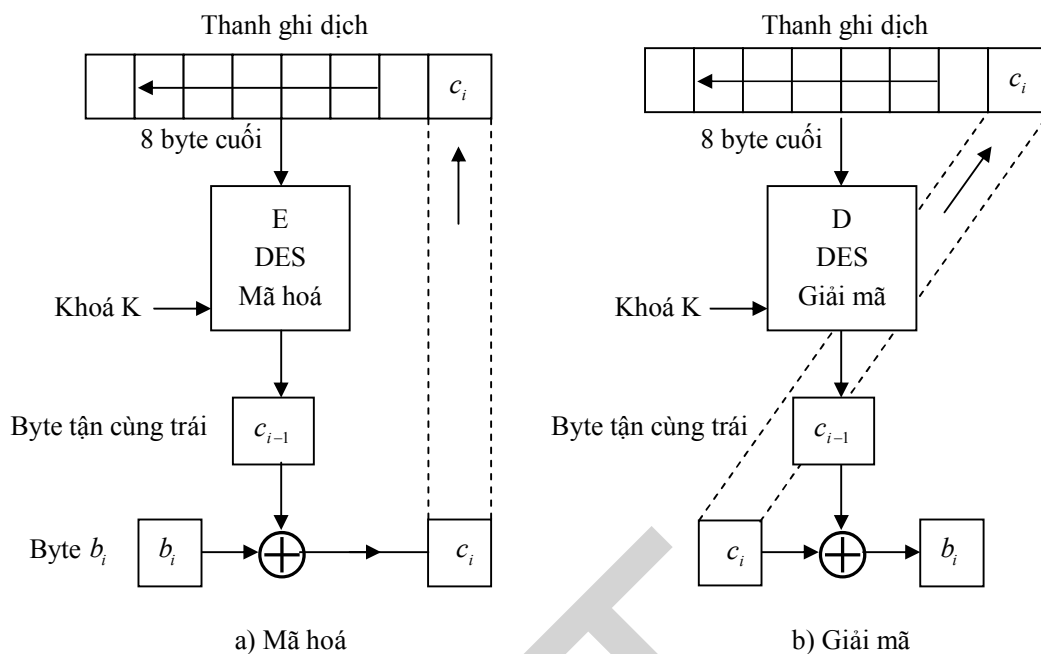


Hình 5.24. Chế độ CBC

Chế độ CFB hoạt động trên một hàng đợi 64 bit. Ban đầu, hàng đợi được làm đầy bằng vector khởi tạo (giống như vector IV trong chế độ CBC). Hàng đợi này sẽ được mã hoá và 8 bit tận cùng bên trái của kết quả sẽ được cộng mod 2 với ký tự đầu tiên cần mã hoá. 8 bit được mã hoá này sẽ được phát đi. 8 bit này cũng được chuyển vào vị trí 8 bit tận cùng bên phải của hàng đợi. Tất cả các bit khác sẽ được chuyển sang trái 8 bit và 8 bit tận cùng bên trái sẽ bị loại bỏ. Quá trình tương tự cũng được thực hiện ở đầu thu. Thủ tục này được chỉ ra trên hình 5.25.

Ưu điểm của chế độ CFB là có thể mã hoá ngay lập tức một ký tự. Cũng như trong chế độ CBC, mỗi ký tự cũng ảnh hưởng tới tất cả các ký tự tiếp sau sao cho một sự thay đổi tới một ký tự bất kỳ trong quá trình truyền cũng tác động tới các ký tự kế tiếp. Một sai trong quá trình mã hoá sẽ tác động tới ký tự đang được chuyển và cũng tác động tới 8 ký tự tiếp theo bởi vì nó sẽ nằm trong hàng đợi ở 8 phép mã hoá ký tự sau đó cho tới khi ký tự sai được đẩy ra khỏi ô tận cùng bên trái trong hàng đợi.

Hai chế độ này mở rộng cơ bản tính hiệu quả của DES. Chúng khắc phục được 2 hạn chế chủ yếu của DES: hạn chế khi mã hoá các bản rõ lặp lại và sự giữ chậm để đợi mã hoá đồng thời một khối văn bản. Không có những hạn chế này, DES có chất lượng tương đương với các thuật toán mã hoá dòng.



Hình 5.25. Chế độ CFB

### 5.5.3. Hai khoá cho hiệu quả tương đương một khoá 112 bit

Mặc dù độ dài khoá của DES được coi là đủ dài nhưng một số người vẫn không tin tưởng vào độ dài 56 bit của nó. Sau đây là một phương pháp nhằm tăng độ dài hiệu quả của khoá. Phương pháp này không đòi hỏi phải thay đổi bản thân thuật toán, nó thích hợp trong trường hợp thuật toán được áp dụng bởi thiết bị phần cứng hoặc trong trường hợp phần mềm có những đoạn không thể sửa đổi.

Do đã có những khảo sát về độ mật của một khoá 56 bit nên hợp lý hơn cả là nên xem xét việc sử dụng 2 khoá. Nếu bằng một cách nào đó, một phép tìm kiếm vét cạn đã tìm được một khoá thì cần phải thời gian gấp đôi để phá khoá thứ hai. Đáng tiếc là điều này không đơn giản như vậy. Merkle [14] đã chứng tỏ rằng hai khoá 56 bit dùng liên tiếp có thể bị phá bằng phép tấn công theo bản rõ chọn lọc trong  $2^{57}$  phép thử chứ không phải là trong  $2^{112}$  phép thử như mong muốn. Bởi vậy, phép mã hoá lần hai hầu như không làm tăng độ mật.

Tuchman [15] tính rằng hai khoá được dùng theo một cách đặc biệt sẽ làm tăng độ mật. Tuchman sử dụng một kỹ thuật được gọi là DES bội ba (Triple DES) do Matyas và Meyer đưa ra. TDES được IBM sử dụng khi mã hoá các khoá chủ trong một số hệ thống mã hoá của họ. Với hai khoá ( $K_1$  và  $K_2$ ), người gửi sẽ mã hoá bằng  $K_1$ , giải mã bằng  $K_2$  và lại mã hoá bằng  $K_1$ . Bên thu sẽ giải mã bằng  $K_1$ , mã hoá bằng  $K_2$  và lại giải mã bằng  $K_1$ .

Quan điểm này rất đáng xem xét trong sử dụng với một thiết bị mã hoá tự động (có thể bằng phần cứng hoặc phần mềm). Nếu thiết bị cần hai khoá và người dùng chỉ muốn dùng một khoá thì người dùng sẽ cung cấp  $K_1$  ba lần. Thiết bị sẽ mã bằng  $K_1$ , giải mã bằng  $K_1$  (trở về bản rõ ban đầu) và cuối cùng lại mã hoá bằng  $K_1$ . Theo cách này, một thiết bị có thể dùng cho cả các phép mã hoá đơn và kép.

## 5.6. TÓM LƯỢC VỀ CÁC THỦ TỤC VÀ CÁC ỨNG DỤNG THỰC TẾ

Trong chương này, chúng ta đã xem xét một số vấn đề quan trọng trong sử dụng các hệ thống liên lạc và các hệ thống máy tính. Ta đã nghiên cứu các thủ tục mà có thể tách biệt một cách có hiệu quả giữa thiết kế của một giải pháp với ứng dụng của giải pháp đó.

Chúng ta cũng đã xác định những nhiệm vụ đối với chúng, các thủ tục phải được thiết kế. Các nhiệm vụ này có ứng dụng trong thương mại điện tử, liên lạc cá nhân và những hoạt động thông thường khác mà ta muốn tự động thực hiện. Các nhiệm vụ này là:

- Ký bằng chữ ký số
- Chơi bài bằng tâm linh (phân phối khoá)
- Giao kèo khoá
- Bỏ phiếu
- Chuyển không nhớ (tung đồng tiền)
- Ký thoả thuận
- Thư tín có chứng thực

Các thủ tục đã giải quyết những vấn đề thường đòi hỏi khi giao tiếp mặt mặt giữa con người. Tuy nhiên, bằng cách thiết kế cẩn thận, ta đã có thể đảm bảo được tính bí mật và xác thực trong khi vẫn giữ được tuyệt đối trung thực.

Như đã nói ở trên, việc tách biệt thiết kế ra khỏi ứng dụng là điều đáng mong muốn. Các thủ tục (có thể phân tích và kiểm tra được) là một phương tiện thiết kế tốt bởi vì tính đúng đắn của một thủ tục có thể được xác định không liên quan tới bất kỳ một ứng dụng riêng nào của nó.

Đối với mỗi thủ tục, trước tiên ta khai thác được vấn đề cần giải quyết. Sau đó, ta xác định những yêu cầu bảo mật cần phải thoả mãn với thủ tục. Tiếp theo, ta thiết kế một thủ tục thoả mãn được các yêu cầu bảo mật. Cuối cùng, chúng ta thực hiện việc phân tích thủ tục để khẳng định rằng nó đã thoả mãn được các yêu cầu.

Chúng ta cũng xem xét những ứng dụng thực tế thích hợp khi sử dụng mã hoá. Chỉ sử dụng mã hoá thì không đủ đảm bảo được độ mật, tính riêng tư hoặc xác thực. Tuy vậy, bằng cách sử dụng đúng phép mã hoá sẽ mang lại cho ta những kết quả đó. Ta cũng đánh giá các hệ thống mã hoá DES và RSA theo các tiêu chuẩn kinh điển của Shannon và các tiêu chuẩn bảo mật khác. Cuối cùng, ta nghiên cứu các kỹ thuật mật mã nhằm hạn chế các sai sót không bị phát hiện (chẳng hạn như sử dụng lại) thông qua các kỹ thuật như nhân thời gian...

## BÀI TẬP CHƯƠNG 5

**Bài 5.1:** Tại sao người ta không muốn có trọng tài trong một thủ tục trao đổi bí mật?

**Bài 5.2:** Cho một ví dụ về một thủ tục tự ràng buộc trong đời sống thực tế?

**Bài 5.3:** Hàm niêm phong mật mã trước tiên được đề xuất (Tổng giá trị số của tất cả các byte trong một thông báo) có một yếu điểm nghiêm trọng: việc trao đổi vị trí của hai byte trong thông báo sẽ không được phát hiện bởi hàm này. Hãy đề xuất một hàm khác không có yếu điểm này.

**Bài 5.4:** Phép mã hoá xếp ba lô của Merkle Hellman không phải là một ánh xạ "vào", tức là có một số nhị phân nào đó không phải là kết quả áp dụng phép mã hoá xếp ba lô lên một đoạn bản rõ. Với những thủ tục nào thì đặc tính này sẽ gây ra vấn đề? Hãy giải thích?

**Bài 5.5:** Có một yếu điểm trong cơ chế niêm phong mật mã được nêu ở đây. Nếu R có thể tính  $f_S$  để kiểm tra xem tư liệu M đã nhận được có phải là tư liệu đã phát hay không thì R cũng có thể tính  $f_S$  để giả mạo một chữ ký số. Hãy đề xuất một giải pháp cho vấn đề này.

**Bài 5.6:** Hãy mô tả một thủ tục trao đổi các bí mật một cách công bằng theo cách thiết lập của con người (không phải theo kiểu máy tính). Hai người muốn trao đổi thông tin bí mật nhưng không một ai muốn đưa một bí mật mà không nhận được một bí mật từ phía bên kia.

a) Các yêu cầu bí mật của tình huống này là gì?

b) Thế nào là một thủ tục trao đổi bí mật công bằng?

**Bài 5.7:** Hãy liệt kê các yêu cầu của một sơ đồ chữ ký số khoá bí mật. Có thể có một yêu cầu nào trong các yêu cầu này cần thoả mãn với một thủ tục có trọng tài? Tại sao có, tại sao không? Có thể có một yêu cầu này cần thoả mãn trong một thủ tục ràng buộc? Tại sao có, tại sao không?

**Bài 5.8:** Hãy trình bày một thủ tục chữ ký số sử dụng mã hoá đối xứng sao cho người gửi và người nhận không phải để lộ nội dung thông báo của họ cho trọng tài.

**Bài 5.9:** Hãy giải thích tại sao thủ tục chữ ký số dùng mã hoá khoá công khai sẽ cản trở bên thu giả mạo thông báo từ người gửi bằng cách sử dụng khoá công khai của người gửi.

**Bài 5.10:** Hãy chỉ ra rằng hệ mật RSA có cả hai tính chất sau: vừa là ánh xạ giao hoán, vừa là ánh xạ "vào". Tại sao hai tính chất này lại cần thiết đối với một thủ tục chữ ký số khoá công khai?

**Bài 5.11:** Một hàm băm (hash) sẽ rút gọn một khối số liệu (lớn) thành một tóm lược (nhỏ hơn). Điều này có nghĩa là không phải mọi sửa đổi có thể có đối với số liệu gốc đều được phát hiện bởi hàm băm. Tuy nhiên, một thuật toán nén văn bản cũng rút gọn số liệu lớn thành dạng được nén nhỏ hơn. Liệu một thuật toán nén có khả năng phát hiện được tất cả những thay đổi trong văn bản gốc? Tại sao có, tại sao không? Liệu một thuật toán nén văn bản có thể sử dụng như một hàm băm hay không? Tại sao có, tại sao không?



## CHƯƠNG 6. CÁC CHUẨN VÀ ÁP DỤNG

### 6.1. BẢO MẬT THƯ ĐIỆN TỬ SỬ DỤNG PRETTY GOOD PRIVACY (PGP)

#### 6.1.1. Mở đầu

PGP là một hiện tượng nổi bật. PGP chủ yếu được kiến tạo bởi Phil Zimmerman. PGP cung cấp dịch vụ xác thực và bí mật có thể dùng cho E-mail và cho việc lưu trữ file. Về cơ bản Phil Zimmerman đã làm được các công việc sau:

Chọn được các thuật toán mật mã tốt để tạo dựng các modun.

Tích hợp các thuật toán này vào một ứng dụng độc lập với hệ điều hành và bộ xử lý. Ứng dụng này sử dụng một tập nhỏ các lệnh để dùng.

- Tạo gói phần mềm với đầy đủ tài liệu kể cả mã nguồn cung cấp miễn phí trên mạng Internet.
- Thỏa thuận với một công ty (ViaCrypt - Hiện nay là Network Associates) để cung cấp phiên bản thương mại với giá rẻ.

PGP đã phát triển mạnh mẽ và hiện nay đang được sử dụng rộng rãi (được xem như một chuẩn bảo mật E-mail cho cộng đồng Internet).

Một số lý do thúc đẩy sự phát triển của PGP:

- Có nhiều phiên bản miễn phí có thể chạy được trên nhiều nền như: DOS/Windows, UNIX, Macintosh... Ngoài ra còn có một sản phẩm thương mại có sự trợ giúp đầy đủ cho khách hàng.
- PGP dựa trên các thuật toán đã được xem xét kỹ và được coi là cực mật như:
  - + Các thuật toán mã hóa công khai RSA, DSS, Diffie-Hellman.
  - + CAST-128, IDEA, TDEA cho mật mã đối xứng.
  - + Hàm băm SHA-1
- Có lực lượng đông đảo những người dùng từ các công ty tới các cá nhân muốn giao tiếp an toàn với các người khác bên ngoài thế giới qua Internet và qua các mạng khác.
- Không bị khống chế và ràng buộc bởi bất kỳ một chính phủ nào hoặc bởi bất cứ một cơ quan tiêu chuẩn nào.

#### 6.1.2. Ký hiệu

- $K_s$  : Khóa phiên được dùng cho mã hóa đối xứng.
- $K_{Ra}$  : Khóa riêng của user A được dùng trong sơ đồ mã hóa công khai.
- $K_{Ua}$  : Khóa công khai của user A được dùng trong sơ đồ mã hóa công khai.

- EP : Mã hóa công khai.
- DP : Giải mã khóa công khai.
- EC : Mã hóa khóa đối xứng.
- DC : Giải mã khóa đối xứng.
- H : Hàm băm.
- || : Phép ghép.
- Z : Nén dùng thuật toán ZIP.
- R64 : Biến đổi sang khuôn dạng ASCII cơ số 64.

Chú ý:

- + Thuật ngữ khóa bí mật là khóa cùng cặp với khóa công khai trong mật mã hóa công khai .
- + Thuật ngữ khóa riêng là khóa của mật mã đối xứng.

### 6.1.3. Mô tả hoạt động

Không kể tới việc quản lý khóa hoạt động của PGP gồm 5 dịch vụ: Xác thực, bí mật, nén, tương thích E-mail và phân đoạn.

Bảng 6.1: Tóm lược các dịch vụ của PGP

Chức năng	Các thuật toán được dùng	Mô tả
Chữ ký số	DSS/SHA hoặc RSA/SHA	Mã băm của thông báo được tạo bằng cách dùng SHS-L. Tóm lược thông báo này được mã bằng DSS hoặc RSA với khóa riêng của người gửi và được kèm với thông báo
Mã hóa thông báo	CAST hoặc IDEA hoặc TDEA với 3 khóa với Diffie-Hellman hoặc RSA	Thông báo được mã bằng CAST-128/IDEA/3DES với khóa phiên một lần được tạo bởi người gửi. Khóa phiên được mã bằng Diffie-Hellman hoặc RSA với khóa công khai của bên thu và được gửi kèm với thông báo.
Nén	ZIP	Thông báo có thể được nén (để lưu giữ hoặc để truyền) bằng ZIP
Tương thích E-mail	Biến đổi cơ số 64	Để cung cấp tính tương thích với các ứng dụng thư điện tử, thông báo đã mã có thể được biến đổi thành 1 xâu ASCII sử dụng phép biến đổi cơ số 64
Phân đoạn		Để phù hợp với hạn chế về kích thước lớn nhất của thông báo, PGP sẽ thực hiện việc phân đoạn và sắp xếp lại.

### **6.1.3.1. Xác thực**

- Người gửi tạo một thông báo.
- Thuật toán SHA-1 được dùng để tạo một mã băm 160 bit của thông báo.
- Mã băm được mã hóa bằng RSA sử dụng khóa riêng của người gửi và kết quả được gắn vào thông báo.
- Bên thu sử dụng RSA với khóa công khai của người gửi để giải mã và khôi phục lại mã băm.
- Bên thu tạo một mã băm mới do thông báo và so sánh nó với mã băm đã được giải mã. Nếu hai mã này phù hợp thì thông báo được coi là xác thực.

Việc kết hợp SHA-1 và RSA tạo nên một sơ đồ chữ ký có hiệu quả. Nhờ độ mật của RSA bên thu được đảm bảo rằng chỉ có người chủ của khóa bí mật tương ứng mới có thể tạo ra chữ ký.

Nhờ thuật toán băm SHA-1 bên thu cũng đảm bảo rằng không một ai có thể tạo được một thông báo mới mà thông báo đó lại có cùng mã băm. Còn một tùy chọn khác là sử dụng DSS/SHA-1.

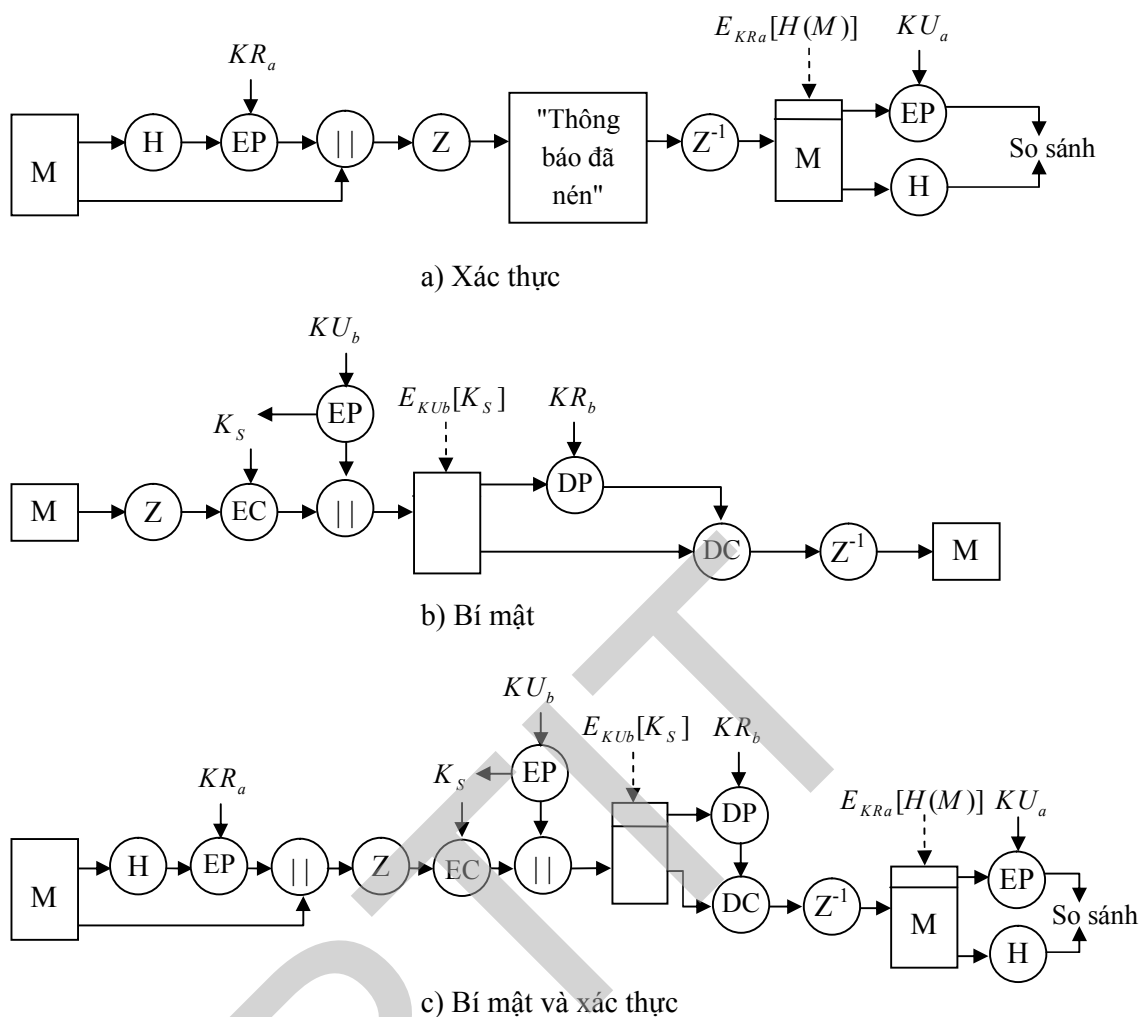
### **6.1.3.2. Bí mật**

Bí mật là một dịch vụ khác của PGP. Nó thực hiện mã hóa các thông báo cần phát hoặc cần được lưu giữ tại chỗ như một file. Trong cả hai trường hợp có thể dùng một trong các mã pháp CAST-128, IDEA hoặc TDEA. Chế độ hồi tiếp khối mã 64 bit được sử dụng.

Trong PGP mỗi khóa riêng chỉ được dùng một lần. Tức là một khóa mới sẽ được tạo ra như một số ngẫu nhiên 128 bit. Vì chỉ được dùng một lần nên khóa phiên được gắn vào thông báo và được truyền cùng thông báo. Để bảo vệ khóa, nó được mã hóa bằng khóa công khai của người nhận.

Quá trình này được mô tả như sau:

- Người gửi tạo một thông báo và một số ngẫu nhiên 128 bit được dùng làm khóa phiên cho chính thông báo này.
- Thông báo được mã hóa bằng CAST-128 (hoặc IDEA/ TDEA) với khóa phiên này.
- Khóa phiên được mã bằng RSA nhờ dùng khóa công khai của người nhận và được gắn vào thông báo.
- Bên thu dùng RSA với khóa bí mật của mình để giải mã và khôi phục lại khóa phiên.
- Khóa phiên được dùng để giải mã thông báo.



Hình 6.1. Các chức năng của PGP

### 6.1.3.3. Bí mật và xác thực

Có thể sử dụng cả hai dịch vụ này cho cùng một thông báo. Trước tiên một chữ ký được tạo ra đối với một thông báo rõ và được gắn vào thông báo. Sau đó thông báo rõ và chữ ký sẽ được mã bằng CAST-128 (hoặc IDEA/ TDEA) và khóa phiên được mã bằng RSA (hoặc ElGamal) theo khóa công khai của người nhận.

### 6.1.3.4. Nén

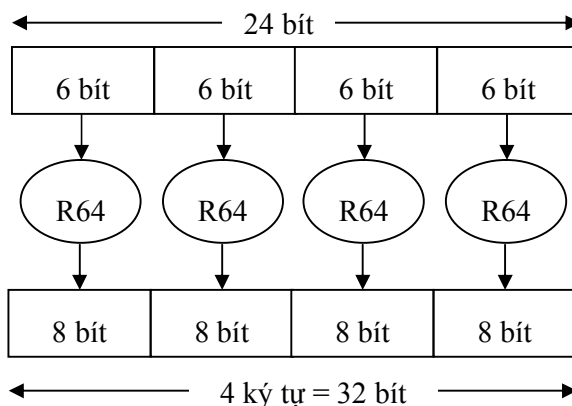
Theo ngầm định, PGP sẽ nén thông báo sau khi ký nhưng trước khi mã. Điều này nhằm tiết kiệm bộ nhớ để truyền thư điện tử hoặc để lưu trữ file. Ngoài ra, điều này còn thuận tiện cho khi cần kiểm tra.

Chú ý:  $Z$ : nén,  $Z^{-1}$ : Giải nén.

### 6.1.3.5. Tương thích E-mail.

Nhiều chương trình E-mail chỉ cho phép dùng các khối chứa các văn bản ASCII. Nhằm phù hợp với hạn chế này PGP có cung cấp dịch vụ biến đổi dòng bit chứa các dãy 8 bit tùy ý

thành dòng bit gồm các ký tự ASCII. Sơ đồ được dùng cho phép biến đổi này là sơ đồ biến đổi cơ số 64, mỗi nhóm 3 bytes dữ liệu nhị phân được biến đổi thành 4 ký tự ASCII.



Hình 6.2.

Bảng 6.2: Mã hóa cơ số 64

Giá trị 6 bit	Mã hoá ký tự	Giá trị 6 bit	Mã hoá ký tự	Giá trị 6 bit	Mã hoá ký tự	Giá trị 6 bit	Mã hoá ký tự
0	A	16	Q	32	g	48	w
1	B	17	R	33	h	49	x
2	C	18	S	34	i	50	y
3	D	19	T	35	j	51	z
4	E	20	U	36	k	52	0
5	F	21	V	37	l	53	1
6	G	22	W	38	m	54	2
7	H	23	X	39	n	55	3
8	I	24	Y	40	o	56	4
9	J	25	Z	41	p	57	5
10	K	26	a	42	q	58	6
11	L	27	b	43	r	59	7
12	M	28	c	44	s	60	8
13	N	29	d	45	t	61	9
14	O	30	e	46	u	62	+
15	P	31	f	47	v	63	/
						(pad)	=

Sử dụng phép biến đổi cơ số 64 này sẽ làm mở rộng thông báo chừng 33%. Tuy nhiên do phần khóa phiên và chữ ký không lớn, mật khác thông báo rõ đã được nén. Trên thực tế quá trình nén đã dư sức bù cho phần mở rộng này.

Ví dụ: Khi dùng ZIP, tỷ số nén vào khoảng 2 lần. Nếu ta bỏ qua các thành phần khóa và chữ ký tương đối nhỏ thì hiệu quả chung của cả nén và mở rộng đối với 1 file có độ dài X là:

$$1,33.0,5.X=0,665.X.$$

Như vậy file kết quả vẫn được nén 33% (1/3).

#### **6.1.3.6. Phân đoạn và sắp xếp lại.**

Các chương trình E-mail thường hạn chế độ dài lớn nhất của thông báo (Chẳng hạn 50.000 bytes).

Một thông báo dài hơn phải được phân đoạn thành các đoạn nhỏ hơn, mỗi đoạn phải được gửi tách biệt.

Để phù hợp với hạn chế này PGP sẽ tự động chia một thông báo quá lớn thành các đoạn đủ nhỏ để gửi qua E-mail.

Việc phân đoạn được thực hiện sau tất cả quá trình khác (bao gồm cả phép biến đổi cơ sở 64). Bởi vậy thành phần khóa phiên và thành phần chữ ký chỉ xuất hiện một lần ở đầu của đoạn đầu tiên.

Ở phía thu, PGP phải tách ra tất cả các header của E-mail và lắp ráp lại toàn bộ khối ban đầu trước khi thực hiện các công đoạn còn lại.

### **6.2. GIAO DỊCH ĐIỆN TỬ AN TOÀN - SET**

#### **6.2.1. Mở đầu**

SET là một đặc tả về mã hóa và an toàn được xây dựng nhằm bảo vệ các giao dịch bằng thẻ tín dụng trên Internet.

Phiên bản hiện thời SETv1 được phát triển với sự tham gia của các công ty IBM, Microsoft, Netscape, RSA, Terisa và Verisign. Tới năm 1998, các phần mềm tương thích với SET đã xuất hiện. Bản thân SET không phải là một hệ thống chi trả. Nó là một tập các thủ tục an toàn và các khuôn dạng an toàn cung cấp cho những người sử dụng hạ tầng chi trả bằng thẻ tín dụng hiện thời trên mạng mở (Internet) theo một cách an toàn.

Về cơ bản SET cung cấp 3 dịch vụ:

- Cung cấp một kênh liên lạc an toàn giữa các bên thực hiện giao dịch.
- Cung cấp xác thực nhờ sử dụng các chứng chỉ số X509v.3.
- Đảm bảo tính riêng tư vì thông tin chỉ khả dụng đối với các bên vào lúc cần thiết và ở nơi cần thiết.

SET được mô tả trong 3 cuốn sách được xuất bản năm 1997:

- Q1: Mô tả thương mại (80 trang).
- Q2: Hướng dẫn cho lập trình viên (629 trang).
- Q3: Định nghĩa thủ tục hình thức (262 trang).

#### **6.2.2. Mô tả SET**

##### **6.2.2.1. Các yêu cầu thương mại**

Q1 mô tả các yêu cầu thương mại cho quá trình chi trả an toàn và các thẻ tín dụng trên Internet và các mạng khác.

- *Cung cấp sự bí mật cho các thông tin đặt hàng và chi trả.*

Điều cần thiết là phải đảm bảo cho người sở hữu thẻ tín dụng rằng thông tin này là an toàn và chỉ có bên nhận hợp pháp mới có thể truy nhập được. Tính bí mật này cũng làm giảm nguy cơ gian trá của bên thực hiện giao dịch khác hoặc của bên thứ ba ác ý. SET sử dụng mã hóa để cung cấp khả năng này.

- *Đảm bảo tính toàn vẹn của tất cả các dữ liệu đã phát*

Tức là phải đảm bảo rằng không thể xảy ra bất cứ một sự thay đổi nào trong nội dung khi truyền các thông báo SET.

Ở đây chữ ký số được dùng để cung cấp tính toàn vẹn.

- *Cung cấp sự xác thực rằng chủ sở hữu thẻ tín dụng (card) là người sử dụng hợp pháp của tài khoản.*

Một cơ chế kết nối chủ sở hữu thẻ tới một số tài khoản riêng làm giảm khả năng gian lận.

Các chữ ký số và các chứng chỉ được dùng để kiểm tra rằng chủ thẻ là người dùng hợp pháp của một tài khoản hợp lệ.

- *Cung cấp sự xác thực rằng người bán (nhà buôn) có thể chấp nhận các giao dịch dùng thẻ qua mối quan hệ của nó với một cơ quan tài chính. Chủ thẻ phải có khả năng xác định các nhà buôn mà họ thực hiện các giao dịch.*

Lại một lần nữa ở đây SET sử dụng các chữ ký số và các chứng chỉ.

- *Đảm bảo sử dụng các kỹ thuật thiết kế hệ thống tốt nhất và các thực tế an toàn tốt nhất để bảo vệ các bên hợp pháp tham gia giao dịch thương mại điện tử.*

SET là một đặc tả đã được kiểm tra kỹ dựa trên các thuật toán và các thủ tục mật mã an toàn cao.

- *Tạo một thủ tục và các khuôn dạng của SET là độc lập với phần cứng, hệ điều hành và phần mềm Web*

#### **6.2.2.2. Các đặc điểm chủ chốt của SET**

- *Tính bí mật của thông tin*

Tài khoản của chủ thẻ và thông tin chi trả phải được bí mật. Một điểm quan trọng và thú vị của SET là nó tránh cho nhà buôn khỏi phải biết số thẻ tín dụng, số này chỉ dùng cho ngân hàng.

DES được dùng để đảm bảo tính bí mật

- *Tính toàn vẹn của dữ liệu*

Thông tin chi trả được gửi từ chủ thẻ tới nhà buôn bao gồm: thông tin đặt hàng, số liệu cá nhân và các lệnh chi trả.

SET đảm bảo rằng các nội dung thông báo này không bị biến đổi trong khi truyền. Các chữ ký số RSA sử dụng các mã băm SHA-1 cung cấp tính toàn vẹn cho thông báo.

Một số thông báo cũng được bảo vệ bằng HMAC dùng SHA-1.

- *Xác thực tài khoản của chủ thẻ.*

SET cho phép nhà buôn kiểm tra rằng chủ thẻ có phải là người dùng hợp pháp không và số tài khoản có hợp lệ không.

SET dùng các chứng chỉ số X509v3 với các chữ ký số RSA cho mục đích này.

- *Xác thực nhà buôn SET cung cấp cho chủ thẻ* có thể kiểm tra được rằng nhà buôn có quan hệ với cơ quan tài chính cho phép chấp nhận các thẻ chi trả.

### **6.2.2.3. Các bên tham gia giao dịch trong SET**

- *Chủ thẻ tín dụng.*

Trong môi trường điện tử các khách hàng sẽ giao tiếp với các nhà buôn từ máy tính cá nhân trên Internet. Chủ thẻ tín dụng là chủ nhân hợp pháp của thẻ chi trả (chẳng hạn Master card và Visa card)

- *Nhà buôn:* là một người hoặc một tổ chức có hàng hóa hoặc các dịch vụ để bán cho các chủ thẻ tín dụng. Các hàng hóa và các dịch vụ này được mời chào trên trang Web hoặc qua thư điện tử.

Một nhà buôn chấp nhận các thẻ chi trả phải có mối quan hệ với một ngân hàng (kho bạc).

- *Nhà phát hành:* Đây là một cơ quan tài chính (chẳng hạn ngân hàng) cung cấp thẻ tín dụng. Nhà phát hành phải chịu trách nhiệm đối với việc chi trả các khoản của nhà buôn.

- *Cổng chi trả:* Đây là chức năng được điều hành bởi kho bạc. Cổng chi trả sẽ thực hiện giao tiếp giữa SET và các mạng chi trả. Nhà buôn sẽ trao đổi các thông báo SET với cổng chi trả qua Internet, còn cổng chi trả sẽ có một số kết nối mạng hoặc kết nối trực tiếp tới hệ thống xử lý tài chính của kho bạc.

- *Máy chủ chứng thực:* Đây là một thực thể tin cậy đưa ra các chứng chỉ khóa công khai cho các chủ thẻ tín dụng, cho các nhà buôn và cho các cổng chi trả. Sự thành công của SET sẽ tùy thuộc vào sự tồn tại của hạ tầng máy chủ chứng thực này.

Bây giờ chúng ta sẽ mô tả một cách ngắn gọn dãy các biến cố cần có cho một giao dịch:

- *Khách hàng mở một tài khoản:* Khách hàng sẽ nhận một tài khoản cho thẻ tín dụng.

- *Khách hàng nhận một chứng chỉ:* Sau phép kiểm tra về tính danh, khách hàng sẽ nhận một chứng chỉ số được ký bởi ngân hàng. Chứng chỉ sẽ kiểm tra khóa công khai RSA của khách hàng và ngày hết hạn của nó. Nó cũng thiết lập một quan hệ (được đảm bảo bởi ngân hàng) giữa cặp khóa của ngân hàng và thẻ tín dụng của anh ta.

- *Các nhà buôn có các chứng chỉ của riêng họ:* Một nhà buôn chấp nhận một loại card chi trả nhất định phải có hai chứng chỉ cho hai khóa công khai mà nhà buôn có: Một để ký nhận các thông báo và một để trao đổi khóa. Nhà buôn cần có một đảm bảo chứng chỉ khóa công khai của cổng chi trả.

- *Khách hàng đặt một đơn hàng:* Trước tiên khách hàng vào trang Web của nhà buôn để chọn hàng và xác định giá. Sau đó khách hàng sẽ gửi một danh sách các hạng mục cần mua



tới nhà buôn. Nhà buôn sẽ gửi trở lại một đơn hàng chứa danh sách các mặt hàng, giá của chúng, giá tổng cộng và số của đơn hàng.

- *Nhà buôn được kiểm tra thêm vào đơn hàng* : Nhà buôn sẽ gửi một bản sao chứng chỉ của nó để khách hàng có thể kiểm tra được là anh ta làm việc với một kho hàng hợp pháp.

- *Đơn hàng chi trả được gửi*: Khách hàng sẽ gửi cả thông tin về đơn hàng và thông tin chi trả tới nhà buôn cùng với chứng chỉ của mình. Đơn hàng sẽ xác nhận việc mua các mặt hàng trong đơn hàng, thông tin chi trả được mã hóa theo cách mà nhà buôn không thể đọc được. Chứng chỉ của khách hàng sẽ làm cho nhà buôn có thể kiểm tra được khách hàng.

- *Nhà buôn yêu cầu quyền chi trả*: Nhà buôn gửi thông tin chi trả tới công chi trả đòi hỏi quyền nhận một khoản tín dụng thích hợp của khách hàng đủ cho vụ mua bán đó..

- *Nhà buôn xác nhận đơn hàng*: Nhà buôn sẽ gửi xác nhận về đơn hàng cho khách hàng.

- *Nhà buôn cung cấp hàng hóa và dịch vụ*: Nhà buôn vận chuyển hàng hóa hoặc cung cấp dịch vụ cho khách hàng.

- *Nhà buôn đòi hỏi chi trả*: Đòi hỏi này sẽ được gửi đến công chi trả là nơi không chế toàn bộ quá trình chi trả.

#### 6.2.2.4. Chữ ký kép

Mục đích của chữ ký kép là liên kết 2 thông báo gửi cho hai bên nhận khác nhau. Trong trường hợp này khách hàng muốn gửi thông tin đặt hàng (OI) tới nhà buôn và thông tin chi trả (PI) tới ngân hàng.

Nhà buôn không cần biết số thẻ tín dụng của khách hàng và ngân hàng không cần biết chi tiết đơn hàng của khách hàng.

Khách hàng cần giữ hai thông tin này tách biệt nhưng hai mục này phải kết nối theo cách để có thể được dùng để giải quyết các tranh chấp khi cần. Kết nối này cần thiết để khách hàng có thể chứng tỏ rằng chi trả này là cho đơn hàng đó chứ không phải cho các hàng hóa hoặc dịch vụ khác. Để thấy rõ hơn ta phải giả sử rằng: khách hàng phải cho nhà buôn hai thông báo: OI có ký và PI có ký. Và nhà buôn gửi PI tới ngân hàng. Nếu nhà buôn có giữ một OI khác của khách hàng thì nhà buôn có thể tuyên bố rằng đó là OI đi kèm với PI này chứ không phải là OI thực sự. Mỗi kết nối sẽ tránh được sự nhập nhèm đó.

$$DS = E_{KR_C} [H(H(PI) / H(OI))]$$

$KR_C$  : Khóa chữ ký riêng của khách hàng.

Bây giờ giả sử rằng nhà buôn có chữ ký kép (PS), có OI và tóm lược thông báo của PI (PIMD). Nhà buôn cũng có khóa công khai của khách hàng lấy từ chứng chỉ của khách hàng. Sau đó nhà buôn có thể tính hai đại lượng sau:

$$H(PIM / H(OI))$$

và

$$D_{KU_C}(DS)$$

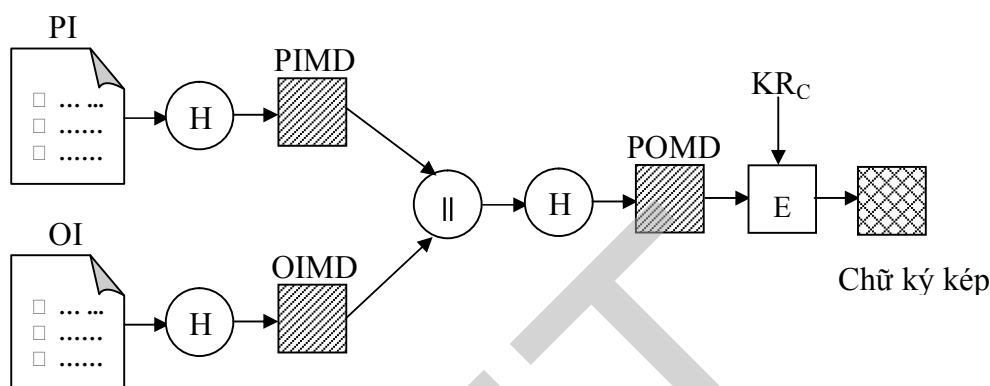
$KU_C$  : Khóa chữ ký công khai của khách hàng.

Nếu hai đại lượng này bằng nhau thì nhà buôn đã kiểm tra được chữ ký.

Tương tự, nếu ngân hàng có  $\begin{cases} DS, PI, OIMD \\ KU_C \end{cases}$

Khi đó ngân hàng có thể tính:  $H(H(PI) // OIMD)$  và  $D_{KU_C}(DS)$ .

Nếu hai đại lượng này bằng nhau thì ngân hàng đã kiểm tra được chữ ký.



Hình 6.3. Cấu trúc của chữ ký kép

PI : Thông tin chi trả

OI : Thông tin đặt hàng

H : Hàm băm

|| : Phép ghép

PIMD : Tóm lược thông báo của PI

OIMD : Tóm lược thông báo của OI

POMD: Tóm lược thông báo của lệnh chi trả

E : Phép mã hóa (RSA)

$KR_C$  : Khóa chữ ký riêng của khách hàng.

### 6.3. ỨNG DỤNG XÁC THỰC - KERBEROS

#### 6.3.1. Mở đầu

Kerberos là một dịch vụ xác thực được xây dựng từ dự án Athena của MIT. Vấn đề mà Kerberos muốn giải quyết là: Trong một môi trường phân tán mở, ở đó các trạm làm việc muốn truy nhập các dịch vụ trên các máy chủ phân tán qua mạng. Ta muốn các máy chủ có khả năng hạn chế truy nhập đối với các người dùng hợp lệ và có thể xác thực các yêu cầu đối với mọi dịch vụ. Trong môi trường này một trạm làm việc không thể tự xác định được đúng các người dùng của nó cho các dịch vụ mạng.

Trên thực tế có ba nguy cơ sau:

- Người sử dụng có thể sử dụng một trạm làm việc nào đó và giả mạo là một người dùng khác.
- Người sử dụng (user) có thể thay đổi địa chỉ mạng của một trạm làm việc để các yêu cầu được gửi từ trạm này xuất hiện như thể từ trạm mạo danh.
- User có thể "nghe trộm" các trao đổi và sử dụng kiểu tấn công sử dụng lại để truy nhập vào một máy chủ hoặc phá vỡ hoạt động.

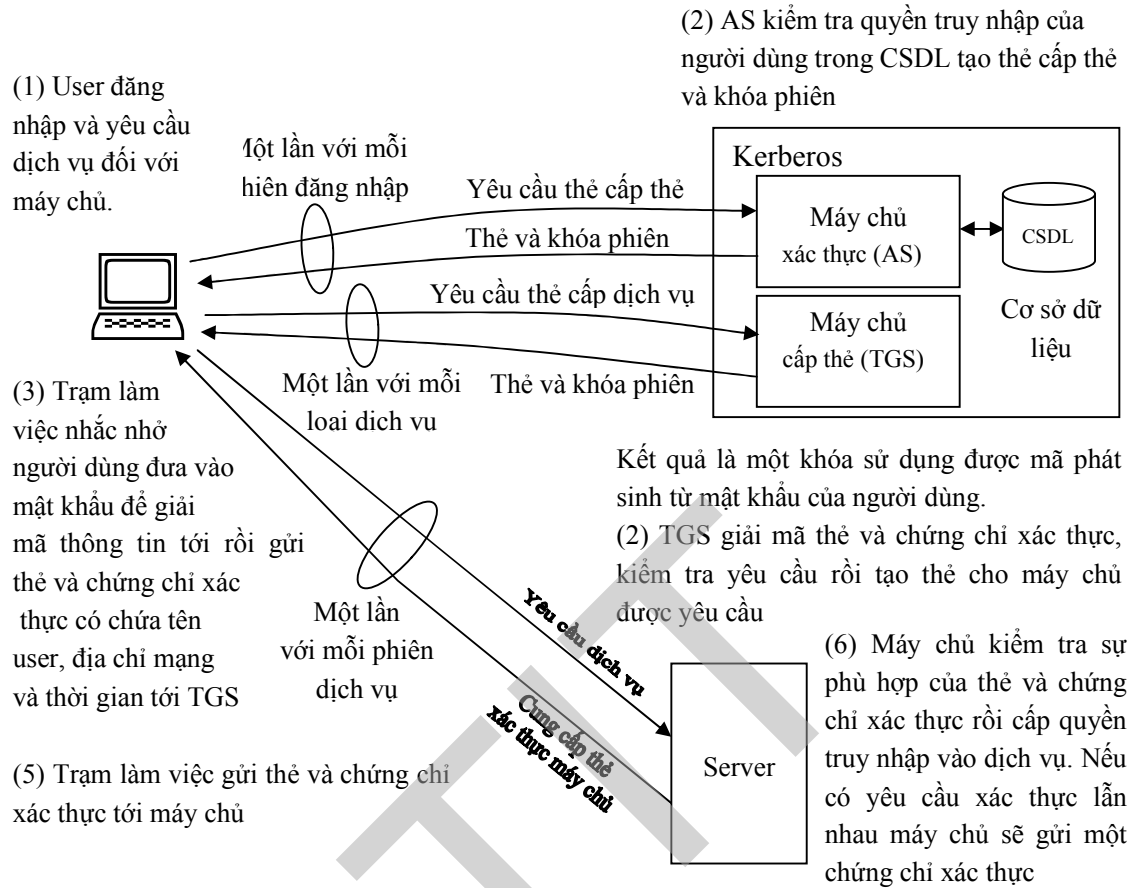
Trong các trường hợp này người dùng bất hợp pháp có thể truy nhập được tới các dịch vụ và dữ liệu mà anh ta không được phép truy nhập. Khác với việc xây dựng các thủ tục xác thực ở mỗi máy chủ, Kerberos cung cấp một dịch vụ xác thực tập trung có nhiệm vụ xác thực các user đối với máy chủ (server) và ngược lại. Cần chú ý rằng Kerberos chỉ sử dụng mật mã đối xứng mà không dùng mật mã khóa công khai.

Có hai phiên bản Kerberos là Kv.4 và Kv.5. Kv.5 được đề trình xem như một chuẩn được khuyến nghị cho Internet.

Các yêu cầu chính đặt ra cho Kerberos:

- An toàn : Kẻ thu trộm trên mạng không có khả năng thu được thông tin cần thiết để mạo danh. Hơn nữa, Kerberos phải đủ mạnh để đối phương mạnh không thể tìm thấy được các yếu điểm.
- Tin cậy: Đối với mọi dịch vụ không chế truy nhập có sử dụng Kerberos, việc thiếu tính sẵn sàng của dịch vụ Kerberos cũng có nghĩa là thiếu tính sẵn sàng của các dịch vụ được trợ giúp. Bởi vậy Kerberos phải rất tin cậy và phải sử dụng kiến trúc máy chủ phân tán có hệ thống dự phòng.
- Trong suốt: Về mặt lý tưởng khi người dùng đưa ra mật khẩu vào họ không thể biết được rằng quá trình xác thực đang xảy ra.
- Có khả năng mở rộng: Kerberos phải có khả năng phục vụ một số lượng lớn các máy chủ và máy trạm. Bởi vậy Kerberos phải có kiến trúc phân tán và modun.

### 6.3.2. Kerberos V.4



Hình 6.4. Tóm lược các trao đổi thông báo của Kv.4

a) Trao đổi dịch vụ xác thực: Để thu nhận thẻ cấp thẻ.

- (1)  $C \rightarrow AS : ID_C // ID_{tgs} // TS_1$  (Nhãn thời gian).
- (2)  $AS \rightarrow C : E_{K_C} [K_{C, tgs} // ID_{tgs} // TS_2 // LT_2 \text{ (thời gian sống) } // \text{Thẻ}_{tgs}]$   
 $\text{Thẻ}_{tgs} = E_{K_{tgs}} [K_{C, tgs} // ID_C // AD_C // ID_{tgs} // TS_2 // LT_2]$

b) Trao đổi dịch vụ cấp thẻ: Để thu nhận thẻ cung cấp dịch vụ.

- (3)  $C \rightarrow TGS : ID_V \text{ Thẻ}_{tgs} // \text{Authenticator}_C$  ( $ID_V$  : ID của dịch vụ yêu cầu)
- (4)  $TGS \rightarrow C : E_{K_{tgs}} [K_{C, v} // ID_V // TS_2 // \text{Thẻ}_v]$   
 $\text{Thẻ}_{tgs} = E_{K_{tgs}} [K_{C, tgs} // ID_C // AD_C // ID_{tgs} // TS_2 // LT_2]$   
 $\text{Thẻ}_v = E_{K_V} [K_{C, v} // ID_C // AD_C // ID_V // TS_2 // LT_4]$   
 $\text{Authenticator}_C = E_{K_{tgs}} [ID_C // AD_C // TS_3]$

c) Trao đổi xác thực Máy trạm/ máy chủ: Để thu nhận dịch vụ.

(5)  $C \rightarrow V$  : Thẻ<sub>V</sub> // Authenticator<sub>C</sub>.

(6)  $V \rightarrow C$  :  $E_{K_{C,V}} [TS_5 \parallel H]$  (để xác thực lẫn nhau)

$$\text{Thẻ}_V = E_{K_V} [K_{C,V} \parallel ID_C \parallel AD_C \parallel ID_V \parallel TS_2 \parallel LT_4]$$

$$\text{Authenticator}_C = E_{K_{TGS}} [ID_C \parallel AD_C \parallel TS_5]$$

Bảng trên cho ta thấy kỹ thuật phân phối khoá phiên.

Trước hết khách hàng (client) gửi một thông báo tới AS yêu cầu truy nhập vào TGS. AS trả lời bằng một thông báo được mã bằng khóa trích xuất từ mật khẩu của người dùng ( $K_C$ ) có chứa thẻ. Thông báo này cũng chứa một bản sao của khóa phiên  $K_{C,TGS}$  cho C và TGS, vì khóa phiên này nằm bên trong thông báo được mã bằng  $K_C$  nên chỉ có máy trạm của người dùng mới có thể đọc được nó. Khóa phiên này cũng nằm trong Thẻ<sub>TGS</sub> mà TGS có thể đọc được. Như vậy khóa phiên đã được phân phối an toàn cho C và TGS. Cần chú ý rằng một số thông tin phụ đã được thêm vào pha hội thoại đầu tiên. Thông báo (1) chứa nhãn thời gian  $TS_1$  để AS biết rằng thông báo là đúng lúc. Thông báo (2) chứa một số yếu tố của thẻ ở dạng mà C có thể truy nhập. Điều này cho phép C xác nhận rằng thư này là của TGS và biết được thời gian hết hạn của nó.

Khi có thẻ và khóa phiên C đã có thể truy nhập vào TGS.

Trước hết C gửi tới TGS một thông báo chứa thẻ +  $ID_V$  (định danh của dịch vụ yêu cầu) (thông báo (3)). Hơn nữa, C cũng phát một thẻ xác thực Authenticator bao gồm  $ID_C$  và  $AD_C$  (địa chỉ của người dùng C) và một nhãn thời gian  $TS_3$ . Không giống như thẻ có thể dùng lại, thẻ xác thực chỉ dùng một lần và có thời gian sống ngắn. TGS có thể giải mã thẻ bằng khóa mà nó chia sẻ với AS. Thẻ này báo rằng người dùng C đã được cung cấp khóa phiên  $K_{C,TGS}$ . Thực ra, thẻ này báo rằng "Người sử dụng khóa  $K_{C,TGS}$  phải là C".

TGS dùng khóa phiên để giải mã thẻ xác thực. Sau đó TGS có thể kiểm tra tên và địa chỉ từ thẻ xác thực từ nội dung của thẻ và từ địa chỉ mạng của thông báo tới. Nếu mọi điều là phù hợp thì TGS đã được đảm bảo rằng người gửi thẻ thực sự là chủ nhân của thẻ. Thực ra thẻ này báo rằng "ở thời điểm  $TS_3$  tôi sử dụng  $K_{C,TGS}$ ". Cần chú ý rằng thẻ không chứng minh định danh của bất cứ ai nhưng là một phương pháp để phân phối các khóa một cách an toàn. Chính thẻ xác thực sẽ chứng minh định danh của client. Vì thẻ xác thực chỉ dùng một lần và có thời gian dùng ngắn nên đối phương khó lòng có thể ăn cắp thẻ và thẻ xác thực để dùng lại.

TGS trả lời bằng thông báo (4) có dạng như thông báo (2).

Thông báo được mã bằng khóa phiên chia sẻ giữa TGS và C, nó chứa một khóa phiên cần chia sẻ giữa C và máy chủ V, định danh của V;  $ID_V$ , nhãn thời gian của thẻ. Bản thân thẻ cũng chứa khóa phiên này.

Lúc này C có thể cấp dịch vụ có thể dùng lại đối với V. Khi C trình thẻ này trong thông báo (5) nó cũng gửi kèm theo một thẻ xác thực (Authentication)

Máy chủ V có thể giải mã thẻ, khôi phục khóa phiên và giải mã thẻ xác thực khi cần xác thực lẫn nhau, máy chủ có thể trả lời như thông báo (6). Máy chủ trả lại giá trị nhãn thời gian lấy từ thẻ xác thực được tăng thêm 1 và được mã bằng khóa phiên. C có thể giải mã thông báo này để khôi phục lại nhãn thời gian đã được tăng. Vì thông báo được mã bằng khóa phiên nên C được đảm bảo rằng nó chỉ có thể được tạo bởi V. Nội dung của thông báo đảm bảo với C rằng thông báo này không phải là dùng lại thông báo cũ. Cuối cùng, ở cuối quá trình, client và server cùng chia sẻ một khóa bí mật. Khóa này có thể được dùng để mã hóa các thông báo trong tương lai cho hai bên hoặc để trao đổi khóa phiên ngẫu nhiên mới.

*Giải thích các yếu tố trong thủ tục Kv.4.*

	<b>a) Trao đổi dịch vụ xác thực</b>
Thông báo (1)	Client yêu cầu thẻ cấp thẻ.
$ID_C$	Báo cho AS định danh của người dùng từ client này
$ID_{TGS}$	Báo cho AS biết rằng người dùng yêu cầu truy nhập TGS
$TS_1$	Cho phép AS kiểm tra rằng đồng hồ của client được đồng bộ với AS
Thông báo (2)	AS trả về thẻ cấp thẻ
$E_{K_C}$	Mã hóa dựa trên mật khẩu của user, cho phép AS và client kiểm tra mật khẩu và bảo vệ nội dung của thông báo (2)
$K_{C,TGS}$	Bản sao của khóa phiên được tạo bởi AS mà client có thể sử dụng để thực hiện trao đổi bí mật giữa client và TGS (không yêu cầu chúng phải chia sẻ một khóa cố định)
$ID_{TGS}$	Xác nhận rằng thẻ này là cho TGS
$TS_2$	Báo cho client về thời gian mà thẻ này đệ trình.
$LT_2$	Báo cho client về thời gian sống của thẻ này
Thẻ $_{TGS}$	Thẻ cần được client sử dụng để truy nhập TGS
	<b>b) Trao đổi dịch vụ cấp thẻ</b>
Thông báo (3)	Client yêu cầu thẻ cấp dịch vụ
$ID_V$	Báo cho TGS rằng user yêu cầu truy nhập tới máy chủ V.
Thẻ $_{TGS}$	Đảm bảo cho TGS rằng user này được xác thực bởi AS.
Authenticator <sub>C</sub>	Được tạo bởi client để xác nhận tính hợp lệ cho thẻ.
Thông báo (4)	TGS trả về thẻ cấp dịch vụ.
$E_{K_{C,TGS}}$	Khóa được chia sẻ giữa C và TGS dùng bảo vệ nội dung của thông báo (4)
$K_{C,TGS}$	Bản sao của khóa phiên được tạo bởi AS.
$ID_V$	Xác nhận rằng thẻ này là cho máy chủ V
$TS_4$	Báo cho client về thời gian mà thẻ này được đệ trình.

Thẻ $v$	Thẻ được dùng bởi client để truy nhập vào máy chủ V.
Thẻ $_{tgs}$	Có khả năng dùng lại để người dùng không phải vào lại mật khẩu.
$E_{K_{tgs}}$	Thẻ được mã hóa bằng khóa chỉ có AS và TGS biết nhằm tránh phá rối (thu trộm)
$K_{C,tgs}$	Bản sao của khóa phiên mà TGS có thể truy nhập được dùng để giải mã thẻ xác thực nhờ đó xác thực thẻ
$ID_C$	Báo chủ sở hữu hợp lệ của thẻ này
$AD_C$	Tránh việc dùng thẻ từ một trạm làm việc khác với trạm đã yêu cầu thẻ
$ID_{tgs}$	Đảm bảo cho máy chủ rằng nó đã giải mã đúng cho thẻ.
$TS_2$	Báo cho TGS về thời gian mà thẻ này được đệ trình.
$LT_2$	Tránh dùng lại sau khi thẻ đã hết hạn.
Authenticator $_C$	Đảm bảo cho TGS rằng người trình thẻ đúng là client mà thẻ đã được trình cho nó, có thời gian sống ngắn để tránh dùng lại.
$E_{K_C,tgs}$	Thẻ xác thực được mã bằng khóa chỉ có client và TGS biết nhằm tránh thu trộm.
$ID_C$	Phải phù hợp với ID trong thẻ để xác thực thẻ.
$AD_C$	Phải phù hợp với địa chỉ trong thẻ để xác thực thẻ
$TS_2$	Báo cho TGS về thời gian mà thẻ xác thực này được tạo ra.
<b>c) Trao đổi xác thực Máy trạm/ máy chủ (Client/Server)</b>	
Thông báo (5)	Client yêu cầu dịch vụ
Thẻ $v$	Đảm bảo với máy chủ rằng người dùng này đã được AS xác nhận
Authenticator $_C$	Được tạo bởi client để xác nhận tính hợp lệ cho thẻ.
Thông báo (6)	Xác thực (tùy chọn) của server đối với client
$E_{K_C,v}$	Đảm bảo cho C rằng thông báo này là từ V
$TS_{5+1}$	Đảm bảo cho C rằng đây không phải là sự dùng lại của hồi đáp cũ
Thẻ $v$	Có thể dùng lại để client không cần yêu cầu thẻ mới từ TGS đối với mỗi truy nhập tới cùng máy chủ.
$E_{K_v}$	Thẻ được mã hóa bằng khóa chỉ có TGS và server biết nhằm tránh thu trộm.
$K_{C,v}$	Bản sao của khóa phiên mà client có thể truy nhập, được dùng để giải mã thẻ xác thực nhằm xác thực thẻ
$ID_C$	Báo chủ sở hữu hợp lệ của thẻ này
$AD_C$	Tránh việc dùng thẻ từ một trạm làm việc khác với trạm đã yêu cầu thẻ.
$ID_v$	Đảm bảo cho server rằng nó đã giải mã đúng cho thẻ.
$TS_4$	Báo cho server về thời gian mà thẻ này được đệ trình.
$LT_4$	Tránh dùng lại sau khi thẻ đã hết hạn.

$Authenticator_C$	Đảm bảo cho server rằng người trình thẻ đúng là client mà thẻ đã được đưa cho nó, có thời gian sống ngắn để tránh dùng lại.
$E_{K_C, v}$	Thẻ xác thực được mã bằng khóa chỉ có server và client biết nhằm tránh thu trộm.
$ID_C$	Phải phù hợp với ID trong thẻ để xác thực thẻ.
$AD_C$	Phải phù hợp với địa chỉ trong thẻ để xác thực thẻ
$TS_S$	Báo cho server về thời gian mà thẻ xác thực này được tạo.

## BÀI TẬP CHƯƠNG 6

**Bài 6.1:** Nêu ý tưởng thiết kế một hệ mật bảo vệ các file dữ liệu trên cơ sở phân tích PGP?

**Bài 6.2:** Cấu tạo của chữ ký kép? Ý nghĩa của chữ ký kép trong thương mại điện tử?

**Bài 6.3:** Vai trò của máy chủ xác thực trong Kerberos?

**Bài 6.4:** Hãy mô tả quá trình trao đổi dịch vụ xác thực trong K.V.4?



## PHỤ LỤC 1 - MÃ NGUỒN DES

```
#define EN0    0        /* MODE == encrypt */
#define DE1    1        /* MODE == decrypt */

typedef struct {
    unsigned long ek[32];
    unsigned long dk[32];
} des_ctx;

extern void deskey(unsigned char *, short);
/*
    hexkey[8]    MODE
    * Sets the internal key register according to the hexadecimal
    * key contained in the 8 bytes of hexkey, according to the DES,
    * for encryption or decryption according to MODE.
    */

extern void usekey(unsigned long *);
/*
    cookedkey[32]
    * Loads the internal key register with the data in cookedkey.
    */

extern void cpkey(unsigned long *);
/*
    cookedkey[32]
    * Copies the contents of the internal key register into the storage
    * located at &cookedkey[0].
    */

extern void des(unsigned char *, unsigned char *);
/*
    from[8]      to[8]
    * Encrypts/Decrypts (according to the key currently loaded in the
    * internal key register) one block of eight bytes at address `from'
    * into the block at address `to'. They can be the same.
    */

static void scrunch(unsigned char *, unsigned long *);
static void unscrunch(unsigned long *, unsigned char *);
static void desfunc(unsigned long *, unsigned long *);
static void cookey(unsigned long *);

static unsigned long KnL[32] = { 0L };
static unsigned long KnR[32] = { 0L };
static unsigned long Kn3[32] = { 0L };
static unsigned char Df_Key[24] = {
    0i01,0x23,0x45,0x67,0x89,0xab,0xcd,0xef,
    0xfe,0xdc,0xba,0x98,0x76,0x54,0x32,0x10,
    0x89,0xab,0xcd,0xef,0i01,0x23,0x45,0x67 };

static unsigned short bytebit[8] = {
    0200, 0100, 040, 020, 010, 04, 02, 01 };

static unsigned long bigbyte[24] = {
    0x800000L,    0x400000L,    0x200000L,    0x100000L,
    0x80000L,     0x40000L,     0x20000L,     0x10000L,
    0x8000L,      0x4000L,      0x2000L,      0x1000L,
    0x800L,       0x400L,       0x200L,       0x100L,
```

```
0x80L,      0x40L,      0x20L,      0x10L,
0x8L,       0x4L,       0x2L,       0x1L   };

/* Use the key schedule specified in the Standard (ANSI X3.92-1981). */

static unsigned char pc1[56] = {
    56, 48, 40, 32, 24, 16,  8,   0, 57, 49, 41, 33, 25, 17,
    9,  1, 58, 50, 42, 34, 26, 18, 10,  2, 59, 51, 43, 35,
    62, 54, 46, 38, 30, 22, 14,   6, 61, 53, 45, 37, 29, 21,
    13,  5, 60, 52, 44, 36, 28,  20, 12,  4, 27, 19, 11,  3 };

static unsigned char totrot[16] = {
    1, 2, 4, 6, 8, 10, 12, 14, 15, 17, 19, 21, 23, 25, 27, 28 };

static unsigned char pc2[48] = {
    13, 16, 10, 23,  0,  4,      2, 27, 14,  5, 20,  9,
    22, 18, 11,  3, 25,  7,     15,  6, 26, 19, 12,  1,
    40, 51, 30, 36, 46, 54,     29, 39, 50, 44, 32, 47,
    43, 48, 38, 55, 33, 52,     45, 41, 49, 35, 28, 31 };

void deskey(key, edf)      /* Thanks to James Gillogly & Phil Karn! */
unsigned char *key;
short edf;
{
    register int i, j, l, m, n;
    unsigned char pc1m[56], pcr[56];
    unsigned long kn[32];

    for ( j = 0; j < 56; j++ ) {
        l = pc1[j];
        m = l & 07;
        pc1m[j] = (key[l >> 3] & bytebit[m]) ? 1 : 0;
    }
    for( i = 0; i < 16; i++ ) {
        if( edf == DE1 ) m = (15 - i) << 1;
        else m = i << 1;
        n = m + 1;
        kn[m] = kn[n] = 0L;
        for( j = 0; j < 28; j++ ) {
            l = j + totrot[i];
            if( l < 28 ) pcr[j] = pc1m[l];
            else pcr[j] = pc1m[l - 28];
        }
        for( j = 28; j < 56; j++ ) {
            l = j + totrot[i];
            if( l < 56 ) pcr[j] = pc1m[l];
            else pcr[j] = pc1m[l - 28];
        }
        for( j = 0; j < 24; j++ ) {
            if( pcr[pc2[j]] ) kn[m] |= bigbyte[j];
            if( pcr[pc2[j+24]] ) kn[n] |= bigbyte[j];
        }
    }
    cookey(kn);
    return;
}

static void cookey(rawl)
register unsigned long *rawl;
{
    register unsigned long *cook, *raw0;
    unsigned long dough[32];
```

```
    register int i;

    cook = dough;
    for( i = 0; i < 16; i++, rawl++ ) {
        raw0 = rawl++;
        *cook  = (*raw0 & 0x00fc0000L) << 6;
        *cook |= (*raw0 & 0x00000fc0L) << 10;
        *cook |= (*raw1 & 0x00fc0000L) >> 10;
        *cook++ |= (*raw1 & 0i00000fc0L) >> 6;
        *cook  = (*raw0 & 0x0003f000L) << 12;
        *cook |= (*raw0 & 0x0000003fL) << 16;
        *cook |= (*raw1 & 0x0003f000L) >> 4;
        *cook++ |= (*raw1 & 0i0000003fL);
    }
    usekey(dough);
    return;
}

void cpkey(into)
register unsigned long *into;
{
    register unsigned long *from, *endp;
    from = KnL, endp = &KnL[32];
    while( from < endp ) *into++ = *from++;
    return;
}

void usekey(from)
register unsigned long *from;
{
    register unsigned long *to, *endp;
    to = KnL, endp = &KnL[32];
    while( to < endp ) *to++ = *from++;
    return;
}

void des(inblock, outblock)
unsigned char *inblock, *outblock;
{
    unsigned long work[2];

    scrunch(inblock, work);
    desfunc(work, KnL);
    unscrunch(work, outblock);
    return;
}

static void scrunch(outof, into)
register unsigned char *outof;
register unsigned long *into;
{
    *into  = (*outof++ & 0xffL) << 24;
    *into |= (*outof++ & 0xffL) << 16;
    *into |= (*outof++ & 0xffL) << 8;
    *into++ |= (*outof++ & 0xffL);
    *into  = (*outof++ & 0xffL) << 24;
    *into |= (*outof++ & 0xffL) << 16;
    *into |= (*outof++ & 0xffL) << 8;
    *into |= (*outof  & 0xffL);
    return;
}
```

```
static void unscrun(outof, into)
register unsigned long *outof;
register unsigned char *into;
{
    *into++ = (*outof >> 24) & 0xffL;
    *into++ = (*outof >> 16) & 0xffL;
    *into++ = (*outof >> 8) & 0xffL;
    *into++ = *outof++ & 0xffL;
    *into++ = (*outof >> 24) & 0xffL;
    *into++ = (*outof >> 16) & 0xffL;
    *into++ = (*outof >> 8) & 0xffL;
    *into = *outof & 0xffL;
    return;
}

static unsigned long SP1[64] = {
    0#01010400L, 0#00000000L, 0#00010000L, 0#01010404L,
    0#01010004L, 0#00010404L, 0#00000004L, 0#00010000L,
    0#00000400L, 0#01010400L, 0#01010404L, 0#00000400L,
    0#01000404L, 0#01010004L, 0#01000000L, 0#00000004L,
    0#00000404L, 0#01000400L, 0#01000400L, 0#00010400L,
    0#00010400L, 0#01010000L, 0#01010000L, 0#01000404L,
    0#00010004L, 0#01000004L, 0#01000004L, 0#00010004L,
    0#00000000L, 0#00000404L, 0#00010404L, 0#01000000L,
    0#00010000L, 0#01010404L, 0#00000004L, 0#01010000L,
    0#01010400L, 0#01000000L, 0#01000000L, 0#00000400L,
    0#01010004L, 0#00010000L, 0#00010400L, 0#01000004L,
    0#00000400L, 0#00000004L, 0#01000404L, 0#00010404L,
    0#01010404L, 0#00010004L, 0#01010000L, 0#01000404L,
    0#01000004L, 0#00000404L, 0#00010404L, 0#01010400L,
    0#00000404L, 0#01000400L, 0#01000400L, 0#00000000L,
    0#00010004L, 0#00010400L, 0#00000000L, 0#01010004L };

static unsigned long SP2[64] = {
    0x80108020L, 0x80008000L, 0#00008000L, 0#00108020L,
    0#00100000L, 0#00000020L, 0x80100020L, 0x80008020L,
    0x80000020L, 0x80108020L, 0x80108000L, 0x80000000L,
    0x80008000L, 0#00100000L, 0#00000020L, 0x80100020L,
    0#00108000L, 0#00100020L, 0x80008020L, 0#00000000L,
    0x80000000L, 0#00008000L, 0#00108020L, 0x80100000L,
    0#00100020L, 0x80000020L, 0#00000000L, 0#00108000L,
    0#00008020L, 0x80108000L, 0x80100000L, 0#00008020L,
    0#00000000L, 0#00108020L, 0x80100020L, 0#00100000L,
    0x80008020L, 0x80100000L, 0x80108000L, 0#00008000L,
    0x80100000L, 0x80008000L, 0#00000020L, 0x80108020L,
    0#00108020L, 0#00000020L, 0#00008000L, 0x80000000L,
    0#00008020L, 0x80108000L, 0#00100000L, 0x80000020L,
    0#00100020L, 0x80008020L, 0x80000020L, 0#00100020L,
    0#00108000L, 0#00000000L, 0x80008000L, 0#00008020L,
    0x80000000L, 0x80100020L, 0x80108020L, 0#00108000L };

static unsigned long SP3[64] = {
    0#00000208L, 0#08020200L, 0#00000000L, 0#08020008L,
    0#08000200L, 0#00000000L, 0#00020208L, 0#08000200L,
    0#00020008L, 0#08000008L, 0#08000008L, 0#00020000L,
    0#08020208L, 0#00020008L, 0#08020000L, 0#00000208L,
    0#08000000L, 0#00000008L, 0#08020200L, 0#00000200L,
    0#00020200L, 0#08020000L, 0#08020008L, 0#00020208L,
    0#08000208L, 0#00020200L, 0#00020000L, 0#08000208L,
    0#00000008L, 0#08020208L, 0#00000200L, 0#08000000L,
    0#08020200L, 0#08000000L, 0#00020008L, 0#00000208L,
```

```
0#00020000L, 0#08020200L, 0#08000200L, 0#00000000L,
0#00000200L, 0#00020008L, 0#08020208L, 0#08000200L,
0#08000008L, 0#00000200L, 0#00000000L, 0#08020008L,
0#08000208L, 0#00020000L, 0#08000000L, 0#08020208L,
0#00000008L, 0#00020208L, 0#00020200L, 0#08000008L,
0#08020000L, 0#08000208L, 0#00000208L, 0#08020000L,
0#00020208L, 0#00000008L, 0#08020008L, 0#00020200L };
```

```
static unsigned long SP4[64] = {
0#00802001L, 0#00002081L, 0#00002081L, 0#00000080L,
0#00802080L, 0#00800081L, 0#00800001L, 0#00002001L,
0#00000000L, 0#00802000L, 0#00802000L, 0#00802081L,
0#00000081L, 0#00000000L, 0#00800080L, 0#00800001L,
0#00000001L, 0#00002000L, 0#00800000L, 0#00802001L,
0#00000080L, 0#00800000L, 0#00002001L, 0#00002080L,
0#00800081L, 0#00000001L, 0#00002080L, 0#00800080L,
0#00002000L, 0#00802080L, 0#00802081L, 0#00000081L,
0#00800080L, 0#00800001L, 0#00802000L, 0#00802081L,
0#00000081L, 0#00000000L, 0#00000000L, 0#00802000L,
0#00002080L, 0#00800080L, 0#00800081L, 0#00000001L,
0#00802001L, 0#00002081L, 0#00002081L, 0#00000080L,
0#00802081L, 0#00000081L, 0#00000001L, 0#00002000L,
0#00800001L, 0#00002001L, 0#00802080L, 0#00800081L,
0#00002001L, 0#00002080L, 0#00800000L, 0#00802001L,
0#00000080L, 0#00800000L, 0#00002000L, 0#00802080L };
```

```
static unsigned long SP5[64] = {
0#00000100L, 0#02080100L, 0#02080000L, 0x42000100L,
0#00080000L, 0#00000100L, 0x40000000L, 0#02080000L,
0x40080100L, 0#00080000L, 0#02000100L, 0x40080100L,
0x42000100L, 0x42080000L, 0#00080100L, 0x40000000L,
0#02000000L, 0x40080000L, 0x40080000L, 0#00000000L,
0x40000100L, 0x42080100L, 0x42080100L, 0#02000100L,
0x42080000L, 0x40000100L, 0#00000000L, 0x42000000L,
0#02080100L, 0#02000000L, 0x42000000L, 0#00080100L,
0#00080000L, 0x42000100L, 0#00000100L, 0#02000000L,
0x40000000L, 0#02080000L, 0x42000100L, 0x40080100L,
0#02000100L, 0x40000000L, 0x42080000L, 0#02080100L,
0x40080100L, 0#00000100L, 0#02000000L, 0x42080000L,
0x42080100L, 0#00080100L, 0x42000000L, 0x42080100L,
0#02080000L, 0#00000000L, 0x40080000L, 0x42000000L,
0#00080100L, 0#02000100L, 0x40000100L, 0#00080000L,
0#00000000L, 0x40080000L, 0#02080100L, 0x40000100L };
```

```
static unsigned long SP6[64] = {
0x20000010L, 0x20400000L, 0#00004000L, 0x20404010L,
0x20400000L, 0#00000010L, 0x20404010L, 0#00400000L,
0x20004000L, 0#00404010L, 0#00400000L, 0x20000010L,
0#00400010L, 0x20004000L, 0x20000000L, 0#00004010L,
0#00000000L, 0#00400010L, 0x20004010L, 0#00004000L,
0#00404000L, 0x20004010L, 0#00000010L, 0x20400010L,
0x20400010L, 0#00000000L, 0#00404010L, 0x20404000L,
0#00004010L, 0#00404000L, 0x20404000L, 0x20000000L,
0x20004000L, 0#00000010L, 0x20400010L, 0#00404000L,
0x20404010L, 0#00000000L, 0#00004010L, 0x20000010L,
0#00400000L, 0x20004000L, 0x20000000L, 0#00004010L,
0x20000010L, 0x20404010L, 0#00404000L, 0x20400000L,
0#00404010L, 0x20404000L, 0#00000000L, 0x20400010L,
0#00000010L, 0#00004000L, 0x20400000L, 0#00404010L,
0#00004000L, 0#00400010L, 0x20004010L, 0#00000000L,
0x20404000L, 0x20000000L, 0#00400010L, 0x20004010L };
```

```
static unsigned long SP7[64] = {
    0#00200000L, 0#04200002L, 0#04000802L, 0#00000000L,
    0#00000800L, 0#04000802L, 0#00200802L, 0#04200800L,
    0#04200802L, 0#00200000L, 0#00000000L, 0#04000002L,
    0#00000002L, 0#04000000L, 0#04200002L, 0#00000802L,
    0#04000800L, 0#00200802L, 0#00200002L, 0#04000800L,
    0#04000002L, 0#04200000L, 0#04200800L, 0#00200002L,
    0#04200000L, 0#00000800L, 0#00000802L, 0#04200802L,
    0#00200800L, 0#00000002L, 0#04000000L, 0#00200800L,
    0#04000000L, 0#00200800L, 0#00200000L, 0#04000802L,
    0#04000802L, 0#04200002L, 0#04200002L, 0#00000002L,
    0#00200002L, 0#04000000L, 0#04000800L, 0#00200000L,
    0#04200800L, 0#00000802L, 0#00200802L, 0#04200800L,
    0#00000802L, 0#04000002L, 0#04200802L, 0#04200000L,
    0#00200800L, 0#00000000L, 0#00000002L, 0#04200802L,
    0#00000000L, 0#00200802L, 0#04200000L, 0#00000800L,
    0#04000002L, 0#04000800L, 0#00000800L, 0#00200002L };

static unsigned long SP8[64] = {
    0x10001040L, 0#00001000L, 0#00040000L, 0x10041040L,
    0x10000000L, 0x10001040L, 0#00000040L, 0x10000000L,
    0#00040040L, 0x10040000L, 0x10041040L, 0#00041000L,
    0x10041000L, 0#00041040L, 0#00001000L, 0#00000040L,
    0x10040000L, 0x10000040L, 0x10001000L, 0#00001040L,
    0#00041000L, 0#00040040L, 0x10040040L, 0x10041000L,
    0#00001040L, 0#00000000L, 0#00000000L, 0x10040040L,
    0x10000040L, 0x10001000L, 0#00041040L, 0#00040000L,
    0#00041040L, 0#00040000L, 0x10041000L, 0#00001000L,
    0#00000040L, 0x10040040L, 0#00001000L, 0#00041040L,
    0x10001000L, 0#00000040L, 0x10000040L, 0x10040000L,
    0x10040040L, 0x10000000L, 0#00040000L, 0x10001040L,
    0#00000000L, 0x10041040L, 0#00040040L, 0x10000040L,
    0x10040000L, 0x10001000L, 0x10001040L, 0#00000000L,
    0x10041040L, 0#00041000L, 0#00041000L, 0#00001040L,
    0#00001040L, 0#00040040L, 0x10000000L, 0x10041000L };

static void desfunc(block, keys)
register unsigned long *block, *keys;
{
    register unsigned long fval, work, right, leftt;
    register int round;

    leftt = block[0];
    right = block[1];
    work = ((leftt >> 4) ^ right) & 0#0f0f0f0fL;
    right ^= work;
    leftt ^= (work << 4);
    work = ((leftt >> 16) ^ right) & 0#0000ffffL;
    right ^= work;
    leftt ^= (work << 16);
    work = ((right >> 2) ^ leftt) & 0x33333333L;
    leftt ^= work;
    right ^= (work << 2);
    work = ((right >> 8) ^ leftt) & 0#00ff00ffL;
    leftt ^= work;
    right ^= (work << 8);
    right = ((right << 1) | ((right >> 31) & 1L)) & 0xffffffffL;
    work = (leftt ^ right) & 0xaaaaaaaaL;
    leftt ^= work;
    right ^= work;
```

```
leftt = ((leftt << 1) | ((leftt >> 31) & 1L)) & 0xffffffffL;

for( round = 0; round < 8; round++ ) {
    work = (right << 28) | (right >> 4);
    work ^= *keys++;
    fval = SP7[ work & 0x3fL];
    fval |= SP5[(work >> 8) & 0x3fL];
    fval |= SP3[(work >> 16) & 0x3fL];
    fval |= SP1[(work >> 24) & 0x3fL];
    work = right ^ *keys++;
    fval = SP8[ work & 0x3fL];
    fval |= SP6[(work >> 8) & 0x3fL];
    fval |= SP4[(work >> 16) & 0x3fL];
    fval |= SP2[(work >> 24) & 0x3fL];
    leftt ^= fval;
    work = (leftt << 28) | (leftt >> 4);
    work ^= *keys++;
    fval = SP7[ work & 0x3fL];
    fval |= SP5[(work >> 8) & 0x3fL];
    fval |= SP3[(work >> 16) & 0x3fL];
    fval |= SP1[(work >> 24) & 0x3fL];
    work = leftt ^ *keys++;
    fval = SP8[ work & 0x3fL];
    fval |= SP6[(work >> 8) & 0x3fL];
    fval |= SP4[(work >> 16) & 0x3fL];
    fval |= SP2[(work >> 24) & 0x3fL];
    right ^= fval;
}

right = (right << 31) | (right >> 1);
work = (leftt ^ right) & 0xaaaaaaaaL;
leftt ^= work;
right ^= work;
leftt = (leftt << 31) | (leftt >> 1);
work = ((leftt >> 8) ^ right) & 0#00ff00ffL;
right ^= work;
leftt ^= (work << 8);
work = ((leftt >> 2) ^ right) & 0x33333333L;
right ^= work;
leftt ^= (work << 2);
work = ((right >> 16) ^ leftt) & 0#0000ffffL;
leftt ^= work;
right ^= (work << 16);
work = ((right >> 4) ^ leftt) & 0#0f0f0f0fL;
leftt ^= work;
right ^= (work << 4);
*block++ = right;
*block = leftt;
return;
}

/* Validation sets:
 *
 * Single-length key, single-length plaintext -
 * Key      : 0123 4567 89ab cdef
 * Plain    : 0123 4567 89ab cde7
 * Cipher   : c957 4425 6a5e d31d
 *
 *****/

void des_key(des_ctx *dc, unsigned char *key){
    deskey(key,EN0);
}
```

```
        cpkey(dc->ek);
        deskey(key,DE1);
        cpkey(dc->dk);
    }
    /* Encrypt several blocks in ECB mode. Caller is responsible for
       short blocks. */
    void des_enc(des_ctx *dc, unsigned char *data, int blocks){
        unsigned long work[2];
        int i;
        unsigned char *cp;
        cp = data;
        for(i=0;iek);
            unscrun(work,cp);
            cp+=8;
        }
    }

    void des_dec(des_ctx *dc, unsigned char *data, int blocks){
        unsigned long work[2];
        int i;
        unsigned char *cp;

        cp = data;
        for(i=0;idk);
            unscrun(work,cp);
            cp+=8;
        }
    }

    void main(void){
        des_ctx dc;
        int i;
        unsigned long data[10];
        char *cp, key[8] = {0#01,0x23,0x45,0x67,0x89,0xab,0xcd,0xef};
        char x[8] = {0#01,0x23,0x45,0x67,0x89,0xab,0xcd,0xe7};

        cp = x;

        des_key(&dc, key);
        des_enc(&dc, cp, 1);
        printf("Enc(0..7,0..7) = ");
        for(i=0;i<8;i++) printf("%02x ", ((unsigned int) cp[i])&0#00ff);
        printf("\n");

        des_dec(&dc, cp, 1);

        printf("Dec(above,0..7) = ");
        for(i=0;i<8;i++) printf("%02x ", ((unsigned int)cp[i])&0#00ff);
        printf("\n");

        cp = (char *) data;
        for(i=0;i<10;i++)data[i]=i;

        des_enc(&dc, cp, 5); /* Enc 5 blocks. */
        for(i=0;i<10;i+=2) printf("Block %01d = %08lx %08lx.\n",
                                   i/2, data[i], data[i+1]);

        des_dec(&dc, cp, 1);
        des_dec(&dc, cp+8, 4);
        for(i=0;i<10;i+=2) printf("Block %01d = %08lx %08lx.\n",
                                   i/2, data[i], data[i+1]);
    }
}
```