PHÂN TÍCH VÀ CẢI TIẾN PHƯƠNG PHÁP MẬT MÃ HÓA KHÓA CÔNG KHAI

Phan Thị Ngọc Mai*, Nguyễn Thị Thuỳ Trang

Trường Đại học Công nghiệp Thực phẩm Tp.HCM

*Email: *maiptn@cntp.edu.vn*

Ngày nhận bài: 08/03/2017; Ngày chấp nhận đăng: 12/04/2017

TÓM TẮT

Trong bài báo này chúng tôi khảo sát phương pháp mã hóa khóa công khai dựa vào định lý số dư Trung Hoa và phương trình ma trận trong quá trình phát sinh khóa, mã khóa và giải mã do Baocang WANG và Yongzhuang WEI, Yupu HU đưa ra năm 2009 [1] và cùng với các nội dung cần cải tiến sau đây: phát sinh nhanh các ma trận khóa, phát sinh nhanh khóa, hiện thực và so sánh với phương pháp gốc và với RSA.

Từ khóa: số nguyên tố và mã hóa khóa công khai, số dư Trung Hoa.

1. TỔNG QUAN

Mã hóa khóa công khai đã được ứng dụng rộng rãi trong thực tế và thu hút được sự quan tâm nghiên cứu của nhiều nhà khoa học trên thế giới. Tuy nhiên hầu hết các hệ thống mã hóa khóa công khai thông dụng như RSA, ElGamal gặp vấn đề về tốc độ phát sinh cũng như mã hóa khóa tương đối chậm. Nguyên nhân chính của vấn đề này là do trong quá trình phát sinh và mã khóa hóa, giải mã đều tính toán các số nguyên và số nguyên tố rất lớn.

Có nhiều phương pháp giải quyết vấn đề trên đã được đề xuất, một trong những phương pháp là vận dụng định lý số dư Trung Hoa và phương trình ma trận trong quá trình phát sinh khóa, mã hóa và giải mã do Baocang WANG và Yongzhuang WEI, Yupu HU đưa ra năm 2009 [1].

Hầu hết các hệ thống mã hóa khóa công khai như RSA, ElGamal thực hiện tính toán với các số nguyên lớn hàng trăm chữ số. Độ phức tạp trong việc mã hóa, giải mã các hệ thống mã hóa khóa công khai này tỉ lệ thuận với độ lớn của các số nguyên tham gia vào việc tạo khóa mã hóa và khóa công khai. Do đó để hệ mã an toàn, cần tăng kích thước của các số nguyên. Mặt khác, khi kích thước của các số nguyên cần xử lý lớn thì thời gian xử lý về các mặt phát sinh và mã hóa khóa cũng tăng lên. Thông tin cần mã hóa ngày càng đa dạng và có khối lượng lớn, đòi hỏi hệ mã giảm thiểu thời gian xử lý.

Trên thực tế, tốc độ hiệu quả cũng như tính bảo mật là các vấn đề được quan tâm. RSA và ElGamal là hai thuật toán còn tồn tại trong môi trường yêu cầu tính bảo mật khắc nghiệt trong thập kỷ qua. Tuy nhiên cả hai thuật toán trên đều có tốc độ khá chậm vì tính toán dựa trên hàm mũ là hàm sử dụng nhiều tài nguyên.

Để hỗ trợ giải quyết các vấn đề trên, bài báo "Phân tích và cải tiến phương pháp mật mã hóa khóa công khai" tập trung vào việc xây dựng và đề xuất một số thuật toán tối ưu hóa nhằm tăng hiệu năng của quá trình phát sinh khóa dựa vào định lý số dư Trung Hoa và ứng dụng trong phân tích, cải tiến hiệu năng tính toán của quá trình phát sinh khóa.

2. CƠ SỞ LÝ THUYẾT

2.1. Định lý về số dư Trung hoa

Cho tập số nguyên tố $p_1, p_2, \dots, p_k$ đôi một nguyên tố cùng nhau [2], với mỗi bộ số nguyên bất kỳ $x_1, x_2, \dots, x_k$ thì hệ phương trình đồng dư.

$$\begin{cases} x \equiv x_1 \ (mod\ p_1) \\ x \equiv x_2 \ (mod\ p_2) \\ \quad \dots \\ x \equiv x_k \ (mod\ p_k\ ) \end{cases}$$

luôn có nghiệm duy nhất $x \in Z_N$, $N = p_1.p_2 \dots p_k$, được xác định như sau:

$$x = x_1p_2p_3 \dots p_kc_1 + p_1x_2p_3 \dots p_kc_2 + p_1p_2x_3 \dots p_kc_3 + \dots + p_1p_2 \dots p_{k-1}x_kc_k(mod\ N)$$

Với c_i được tính $c_i = N^{-1}(mod\ p_i)$ với $i = 1, \dots, k$ có thể biểu diễn dưới dạng $(x_1, x_2, \dots, x_k)$ và ngược lại từ $(x_1, x_2, \dots, x_k)$ ta tìm được duy nhất một x.

Chứng minh tính duy nhất và tồn tại có thể tham khảo [2].

Ví dụ: Tìm x thỏa mãn hệ phương trình đồng dư sau:

$$\begin{cases} x \equiv 2(mod\ 3) \\ x \equiv 3(mod\ 5) \\ x \equiv 4(mod\ 7\ ) \end{cases}$$

Ta tính như sau: $p_1 = 3, p_2 = 5, p_3 = 7, N = p_1.p_2.p_3 = 3.5.7 = 105$

$$x_1 = 2, x_2 = 3, x_3 = 4, c_1 = (5.7)^{-1}(mod\ 3) = 2^{-1}(mod\ 3) = 2,$$

$$c_2 = (3.7)^{-1}(mod\ 5) = 1^{-1}(mod\ 5) = 1,$$

$$c_3 = (3.5)^{-1}(mod\ 7) = 1^{-1}(mod\ 7) = 1$$

Vậy ta suy ra: $x = 2.5.7.2 + 3.3.7.1 + 3.5.4.1 = 263 = 53\ (mod\ 105)$

Thuật toán về định lý số dư Trung Hoa

Input, $[x_1, x_2, \dots, x_k], [p_1, p_2, \dots, p_k]$

Output: x

(1) $N = p_1 * p_2 * \dots * p_k a$

(2) $x = 0$

(3) For $i = 0\ to\ k$ Do $c_i = \left(\frac{N}{p_i}\right)^{-1} mod\ p_i$

(4) End For

(5) For $i = 0\ to\ k$ Do $x = x + x_i \left(\frac{N}{p_i}\right) c_i$

(6) End For

(7) Return x

Trong thuật toán trên ta sử dụng tính nghịch đảo của một số nguyên. Ta tính như sau: gọi y là nghịch đảo của $x(mod\ n)$ nếu: $x * y \equiv 1(mod\ n) \Leftrightarrow x * y - kn = 1$

2.2. Sinh nhanh ma trận khả nghịch

2.2.1. Ma trận khả nghịch

Cho A là ma trận vuông cấp n, một ma trận B gọi là ma trận nghịch đảo của A

$$AB = BA = I_n$$

Khi đó A được gọi là ma trận khả nghịch.

2.2.2. Cách tìm ma trận khả nghịch

a. Dựa vào định thức

$$A^{-1} = \frac{1}{Det(A)} A^{*T}$$

với $a_{ij}^{*} = (-1)^{i+j} \det(M_{ij})$, M_{ij} lấy từ ma trận A bằng cách bỏ đi hàng i cột j.

b. Dựa vào phép biến đổi sơ cấp trên hàng

$[A_n \mid I_n]$ (I_n là ma trận đơn vị cấp n)

$$[A_n \mid I_n] = \left(\begin{array}{cccc|cccc} a_{11} & a_{12} & \dots & a_{1n} & 1 & 0 & \dots & 0 \\ a_{21} & a_{22} & \dots & a_{2n} & 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots & \dots & \dots & \dots & \dots \\ a_{n1} & a_{n2} & \dots & a_{nn} & 0 & 0 & \dots & 1 \end{array}\right)$$

c. Dựa vào cách giải hệ phương trình

U là ma trận tam giác trên (Upper Matrix), L là ma trận tam giác dưới (Lower Matrix).

$$L.L^{-1} = I$$

$$L.L^{-1} = LX = L[X_1,...,X_n] = I_n \Leftrightarrow \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \dots & \dots & \dots & \dots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \\ \dots \\ x_n \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

$$LX_i = \begin{pmatrix} 0 \\ \dots \\ 1_i \\ \dots \\ 0 \end{pmatrix} \Leftrightarrow \begin{pmatrix} l_{11} & 0 & \dots & 0 \\ l_{21} & l_{22} & \dots & 0 \\ \dots & \dots & l_{jj} & \dots \\ l_{n1} & l_{n2} & \dots & l_{nn} \end{pmatrix} \begin{pmatrix} x_{i1} \\ \dots \\ x_{ij} \\ x_{in} \end{pmatrix} = \begin{pmatrix} 0 \\ \dots \\ 1 \\ 0 \end{pmatrix}$$

$$x_{i1} = 0, \;\; x_{ij} = (1 - \sum_{k=1}^{j} l_{jk} x_k) / l_{jj}$$

Như vậy nếu $\boldsymbol{A = LU}$ khả nghịch thì $\boldsymbol{A^{-1} = U^{-1}L^{-1}}$

2.3. Sinh ma trận và tìm ma trận nghịch đảo

2.3.1. Sinh ma trận

Trong bài báo này chúng tôi đưa ra cách cài đặt thuật toán phát sinh ma trận A thông qua hai ma trận U và L. Trong đó U là ma trận tam giác trên, L là ma trận tam giác dưới.

U khả nghịch $\Leftrightarrow \prod_{i=0}^{n} u_{ii} \neq 0$

L khả nghịch $\Leftrightarrow \prod_{i=0}^{n} l_{ii} \neq 0$

$A = U.L$ cũng khả nghịch.

Thuật toán phát sinh ma trận A thông qua hai ma trận U, L trên trường Z_p

(1) U là ma trận vuông tam giác trên

U khả nghịch $\Leftrightarrow \left(\prod_{i=1}^{n} u_{ii} \right) \% p \neq 0 \Leftrightarrow 0 < u_{ij} < p$

(2) L là ma trận vuông tam giác dưới

L khả nghịch $\Leftrightarrow \left(\prod_{i=1}^{n} l_{ii} \right) \% p \neq 0 \Leftrightarrow 0 < l_{ij} < p$

(3) $A = U.L$

2.3.2. Tìm ma trận nghịch đảo

Thuật toán tìm ma trận nghịch đảo B thông qua hai ma trận U, L

(1) Tìm U^{-1}

(2) Tìm L^{-1}

(3) $B = A^{-1} = (LU)^{-1} = U^{-1}.L^{-1}$

Một số ma trận tam giác đặc biệt

Định nghĩa ma trận đối hợp: Cho ma trận vuông A_{nxn}, A được gọi là ma trận đối hợp (Involation Matrix) nếu $A^2 = I$

Trong quá trình phát sinh ngẫu nhiên hai ma trận U, L có những trường hợp:

$$U^2 = U.U = I_n \text{ hay } L^2 = L.L = I_n$$

$$\begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_{nn} \end{pmatrix} \begin{pmatrix} u_{11} & u_{12} & \dots & u_{1n} \\ 0 & u_{22} & \dots & u_{2n} \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_{nn} \end{pmatrix} = \begin{pmatrix} u_{11}^2 & 0 & \dots & 0 \\ 0 & u_{22}^2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_{nn}^2 \end{pmatrix} = I_n$$

$$\Leftrightarrow \begin{pmatrix} u_{11}^2 & 0 & \dots & 0 \\ 0 & u_{22}^2 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & u_{nn}^2 \end{pmatrix} = \begin{pmatrix} 1 & 0 & \dots & 0 \\ 0 & 1 & \dots & 0 \\ \dots & \dots & \dots & \dots \\ 0 & 0 & \dots & 1 \end{pmatrix}$$

$$\Leftrightarrow \begin{pmatrix} u_{11}^2 = 1 \\ u_{22}^2 = 1 \\ \dots \\ u_{nn}^2 = 1 \end{pmatrix} \Leftrightarrow u_{ii} = u_{ii}^{-1} (i = 1,2, \dots, n)$$

Để giải quyết vấn đề trên ta làm như sau: Khi phát sinh một phần tử u_{ii} nào đó dùng thuật toán Euclide tính u_{ii}^{-1}nếu có u_{ii} nào đó thỏa $u_{ii} = u_{ii}^{-1}$ thì phát sinh lại u_{ii}. Như vậy đảm bảo ma trận tam giác sau khi phát sinh có $U^2 \neq I$.

3. HỆ THỐNG MẬT MÃ HÓA KHÓA CÔNG KHAI

3.1. Lịch sử ra đời

Một hệ thống mật mã hóa khóa công khai được các nhà khoa học Whitfield Diffie và Martin Hellman đưa ra năm 1976, dựa trên công trình trước đó của Ralph Merkle về phân phối khóa công khai.

3.2. Thuật toán mật mã hóa khóa công khai

Tạo khóa:

- Phát sinh n số nguyên tố phân biệt chẳng hạn $n = 6, p_1, p_2, \dots, p_6$ mỗi số có chiều dài khoảng 333b. Tính $N = p_1 * p_2 * \dots * p_6$.

- Phát sinh ngẫu nhiên hai ma trận ma trận vuông U, L gồm 6 hàng 6 cột giá trị u_{ij}, l_{ij}, khoảng 30b.

- Tìm A: $A = LU$

- Tìm ma trận nghịch đảo của ma trận A: $B = A^{-1} = (LU)^{-1} = U^{-1}.L^{-1}$

- Dựa vào định lý số dư Trung Hoa tính $a_1, a_2, \dots, a_6$

- Tính $ba_6 \equiv 1(mod\ n), b_1 \equiv ba_1(mod\ n), \dots, b_5 \equiv ba_5(mod\ n)$

- Khóa công khai: $(n, b_1, b_2, \dots, b_5)$, khóa bí mật: $(a_6, p_1, p_2, \dots, p_6, B)$

Mã hóa: chiều dài của bản rõ m mỗi lần mã hóa là 1500b

Chia bản rõ m thành 5 khối $m = (m_1, m_2, \dots, m_5)^T$

Chọn ngẫu nhiên số nguyên r có chiều dài 300b

$$c \equiv b_1m_1 + b_2m_2 + \dots + b_5m_5 + r(mod\ n)$$

Giải mã:

Tính $s \equiv a_6c(mod\ n),\ s_1 \equiv s(mod\ p_1),\ s_2 \equiv s(mod\ p_2), \dots$

$$s_6 \equiv s(mod\ p_6)$$
$$(m_1, m_2, \dots, m_5, r)^T = B(s_1, s_2, \dots, s_6)^T$$

Khôi phục lại bản rõ m ban đầu.

3.2.1. Một số vấn đề khó khăn trong quá trình phát sinh khóa

Khi phát sinh ma trận A để tìm ma trận nghịch đảo của ma trận A ta sử dụng một trong các cách đã trình bày trong chương 2 phần 2.3. Nhưng vấn đề đặt ra là các phần tử của ma trận nghịch đảo có thể chứa giá trị là những số thực dẫn tới việc mã hóa gặp nhiều khó khăn, thông tin có thể bị mất mát trong quá trình mã hóa. Sau đây là một số cách giải quyết để tìm được ma trận nghịch đảo mà đã đưa ra để sau khi tìm ma trận nghịch đảo giá trị những phần tử trong ma trận là những số nguyên.

(1) Phát sinh ma trận vuông A có định thức bằng 1. Khi đó khi tìm ma trận nghịch đảo các phần tử của B hiển nhiên sẽ là các số nguyên.

(2) Tìm ma trận nghịch đảo của ma trận A dựa vào phương pháp đã được trình bày trong phần 2.2.2.

(3) Chúng ta gọi B là ma trận nghịch đảo của A theo modulo p_i nào đó, p_1 chẳng hạn. Khi đó khi giải mã ta phải trả lại tính theo công thức sau:

$$(m_1, m_2, \dots, m_5, r)^T = B(s_1, s_2, \dots, s_6)^T(mod\ p_1)$$

Trong quá trình cài đặt ứng dụng, chúng tôi sử dụng phương pháp thứ 3, đó là tính toán dựa trên Zp.

3.2.2. Đề xuất cách phát sinh ma trận

Theo thuật toán đã đề ra ở trên, ma trận A là một ma trận phát sinh ngẫu nhiên sau đó ta

phải tính nghịch đảo B của A. Áp dụng phương pháp tìm ma trận nghịch đảo ở chương 2 phần 2.3. Như vậy ta sẽ không phát sinh trực tiếp A mà sẽ tính được A thông qua việc phát sinh 2 ma trận tam giác: Ma trận tam giác trên U và ma trận tam giác dưới L.

3.2.3. Tính bảo mật của hệ thống mã hóa khóa công khai

a. Hiệu suất bảo mật thông tin

Trong thực tế, bên cạnh việc bảo mật, hiệu quả bảo mật thông tin của một hệ thống mật mã hóa khoá công khai cần được xem xét. Nếu tỉ lệ hiệu quả bảo mật thông tin thấp thì sẽ tốn nhiều không gian lưu trữ và tài nguyên tính toán nhiều. Vì vậy hiệu quả bảo mật thấp thì sẽ không đạt được tốc độ và yêu cầu của mã hóa khóa công khai của chúng ta.

Bây giờ ta sẽ ước lượng tỉ lệ mã hóa. Trong quá trình mã hóa chiều dài nhị phân L_m của bản rõ là 1500b. Chú ý rằng chiều dài nhị phân của bản mã c xấp xỉ bằng $L_{c\approx}6 * 333 = 1998b$. Do đó chúng ta có thể tính được tỉ lệ mã hóa của phương pháp chúng ta đề xuất là $R = L_m/L_c \approx 1500/1998 \approx 0.75$. Theo đó hiệu suất bảo mật của thuật toán là 4:3 so sánh hiệu suất bảo mật này với thuật toán RSA và El Gamal. Chú ý rằng RSA là 1:1 và của El Gamal là 2:1 từ đó ta thấy rằng hiệu suất bảo mật của chúng ta là tương đối thấp.

b. Độ phức tạp của thuật toán

Phần này ta sẽ nói về độ phức tạp của thuật toán trong quá trình mã hóa và giải mã.

Trong thuật toán mà chỉ thực hiện trên năm phép chia và năm phép cộng modulo. Độ phức tạp của thuật toán nhân modulo là bình phương, độ phức tạp của phép cộng là tuyến tính. Do đó tổng độ phức tạp của thuật toán mã hóa là $O(k^2)$ với k là tham số bí mật. Ta cũng phải chú ý rằng độ phức tạp của thuật toán mã hóa khóa trong RSA và ElGamal là bậc ba.

So sánh trên cho ta thấy rằng thuật toán mã khóa đang khảo sát nhanh hơn RSA và ElGamal.

4. THỬ NGHIỆM VÀ ĐÁNH GIÁ

Bảng 1: So sánh thời gian phát sinh ma trận của bài toán gốc với bài toán cải tiến.

Cấu hình máy tính	Số lần chạy	Thời gian chạy trung bình	
		Bài toán gốc	Bài toán cải tiến
AMD Turion X2 Dual-Core RM-72, 2.10GHz, RAM 4GB	1000	3128,386	2371,228
Intel Core Duo T5270, 1.40GHz, RAM 2GB	1000	2370,145	1780,231
Pentium (R) Dual – Core E550, 2.80 GHz, RAM 2GB	1000	1399,311	979,436

Bảng 2: So sánh thời gian phát sinh khóa của bài toán gốc với bài toán cải tiến.

Cấu hình máy tính	Số lần chạy	Thời gian chạy trung bình	
		Bài toán gốc	Bài toán cải tiến
AMD Turion X2 Dual-Core RM-72, 2.10GHz, RAM 4GB	1000	4838,898	3849,868
Intel Core Duo T5270, 1.40GHz, RAM 2GB	1000	4158,335	2950,664
Pentium (R) Dual – Core E550, 2.80 GHz, RAM 2GB	1000	2019,617	1599,743

Bảng 3: So sánh thời gian phát sinh khóa của bài toán cải tiến với RSA.

Cấu hình máy tính	Số lần chạy	Thời gian chạy trung bình	
		Bài toán gốc	Bài toán cải tiến
AMD Turion X2 Dual-Core RM-72, 2.10GHz, RAM 4GB	1000	2507,455	3849,868
Intel Core Duo T5270, 1.40GHz, RAM 2GB	1000	2352,212	2950,664
Pentium (R) Dual – Core E550, 2.80 GHz, RAM 2GB	1000	1351,309	1599,743

Bảng 4: So sánh thời gian mã hóa, giải mã của bài toán cải tiến với RSA.

Dữ liệu/byte	Thời gian mã hóa		Thời gian giải mã	
	RSA	Bài toán cải tiến	RSA	Bài toán cải tiến
719	1567	406	1344	250
1949	2730	1154	2102	546
4053	3390	1747	2496	1341
5328	3510	2621	2907	1511
6721	3785	2699	3140	1560

5. KẾT LUẬN

5.1. Những kết quả đạt được

Chứng minh tính đúng đắn của thuật toán.

Các kết quả nghiên cứu và ứng dụng bước đầu đã thực hiện được mục đích của bài nghiên cứu. Bằng cách ứng dụng phương trình ma trận và định lý về số dư Trung Hoa trong việc xây dựng hệ thống mã hóa khóa công khai.

Thuật toán đề xuất cho kết quả tốt hơn về thời gian so với bài toán gốc về các mặt phát sinh ma trận, phát sinh khóa.

Chương trình thử nghiệm được xây dựng nhằm chứng minh tính khả thi của các kết quả nghiên cứu.

5.2. Hướng phát triển

Các kết quả của bài nghiên cứu có thể được áp dụng trong nhiều hệ thống mã hóa khóa công khai khác nhau và tiếp tục được cải tiến để có được tốc độ thực thi tốt hơn.

Các kết quả có thể được áp dụng trên nhiều hệ thống bảo mật, thực hiện trong các giao dịch thương mại, thực hiện tạo và xác thực chữ ký điện tử.

Tác giả mong muốn có thể tiếp tục phát triển để đưa các kết quả đã nghiên cứu vào ứng dụng trong thực tế.

TÀI LIỆU THAM KHẢO

1. Bùi Xuân Hải, Trần Ngọc Hội, Trịnh Thanh Đèo, Lê Văn Luyện (2009). Đại số tuyến tính và ứng dụng, NXB Đại học Quốc gia TP.HCM.
2. Nguyễn Đình Thúc, Bùi Doãn Khanh (2006), Mã hóa thông tin lý thuyết và ứng dụng, NXB Lao động và xã hội.
3. Phạm Hữu Khang (2005). Lập trình C#, NXB Lao động Xã hội
4. A. Menezes, P. Van Oorschot, and S. Vanstone, Efficient Implementation, Handbook of Applied Cryptography, Chapter 2, 14, CRC Press, 1996.
5. Baocang WANG, Yongzhuang WEI, Yupu HU (2009), Fast public-key encryption scheme based on Chinese remainder theorem.
6. Rivest R L, Shamir A, Adleman L M. A method for obtaining digital signature and public key cryptosystems. Communications of the ACM, 1978, 21(2): 120-126
7. Solovay, R., and Strassen, V. A Fast Monte - Carlo test for primality. SIAM J. Comptnt. (March 1977), 84 - 85
8. http://www.codeproject.com/KB/cs/biginteger.aspx
9. http://www.codeproject.com/KB/cs/JIBitArray.aspx

ABSTRACT

ANALYSE AND IMPROVE PUBLIC KEY CRYPTOGRAPY METHODS

Phan Thi Ngoc Mai*, Nguyen Thi Thuy Trang

Ho Chi Minh city University of Food Industry

*Email: *maiptn@cntp.edu.vn*

In this article, we examine public key cryptography methods based on the Chinese Remainder Theorem and matrix equation in the process of key generation, encryption and decryption by Baocang WANG and Yongzhuang WEI, Yupu HU published in 2009 and along with the contents to the following improvements: generate fast the key matrix, generate fast key, deploy and compare it to the original mode with RSA method.

Keyword: prime number and public key cryptography, Chinese remainder theorem.