A Rapid Cryptography Algorithm Based on Chaos and Public Key

Yun-peng Zhang*
School of Software and Microelectronics Northwestern Polytechnical University
Xi"an, Shannxi, 710072, P.R.China
Melbourne eResearch Group, The University of Melbourne
Barry St, PARKVILLE, Victoria 3010 Australia
poweryp@163.com

Xia Lin and Qiang Wang
School of Software and Microelectronics Northwestern Polytechnical University
Xi"an, Shannxi, 710072, P.R.China
xgdwq@126.com

Richard O. Sinnott
Melbourne eResearch Group, The University of Melbourne
Barry St, PARKVILLE, Victoria 3010 Australia
rsinnott@unimelb.edu.cn

***Abstract*——This paper is based on the combined application of chaotic and public key cryptosystems to support rapid cryptography algorithms, leveraging their individual advantages. To demonstrate this we designed a rapid knapsack public key algorithm [1] utilizing double sequences, which themselves use a sequence of random numbers generated from interference of Logistic [2] and Chebychev [3] chaotic mapping. We also designed a chaotic pseudo-random number generator while analyzing its characteristics and used it for the implementation of the algorithm. Through simulation and comparisons with related algorithms transversely and longitudinally, we have developed a cryptographic algorithm with higher efficiency and better security.**

***Index Terms*— Cryptography, Public-key, Chaotic systems, Performance**

I. Introduction

A public-key cryptosystem separates encryption and decryption operations so that both sides are able to establish secure communications without exchanging keys in advance. This aspect of transmission and storage of keys, has largely reduced the quantity of keys required in multi-users' communications and saved considerable system resources. Chaotic systems also offer several advantages features to cryptographic communications: ergodicity, miscibility, certainty and sensitivity to initial conditions [4], all of which are related to the diffusion and strength of cryptographic approaches. The combination of these two approaches is thus highly desirable.

At present, research based on chaotic private-key cryptosystems is more mature than that based on the chaotic public-key cryptosystem [5]. Indeed several security problems with existing algorithms are still waiting to be solved [6]. Features of the encryption scheme used and the design of asymmetric encryption systems ultimately illustrate that public-key encryption speeds are slower than the speed of private key encryption systems, subsequently impeding the development and roll out of public-key cryptosystems.

In [7] the authros proposed a high density knapsack public-key cryptosystem algorithm which is relatively efficient, but beyond the author's theoretical analysis and experiments, the overall security of this approach needs to be improved. The author designed a rapid public-key cryptosystem algorithm utilizing chaotic systems to generate interference backpack vectors, and a chaotic pseudo-random generator which was subsequently applied to the algorithm implementation. Through simulation and comparisons with related algorithms transversely and longitudinally, it was found that this algorithm was excellent with an overall higher efficiency and better security.

II. Chinese Remainder Theorem

The Chinese remainder theorem [8] has been widely used in cryptography, primarily for simplifying computations of public-key cryptosystems. The obvious advantage of the algorithm is that it can hide the data and can be designed as a one-way limitation. The theorem itself can be described as follows. Let set n_1 n_2, ... n_k be co-prime and all positive integers where:

$$\begin{cases} x \equiv x_1 \bmod n_1 \\ x \equiv x_2 \bmod n_2 \\ \quad \dots \\ x \equiv x_k \bmod n_k \end{cases} \tag{1}$$

The set {0, 1, 2, …n-1} has a unique solution, that is: $x=\sum_{i=1}^{k} x_i M_i s_i \bmod n$, with $M_i = n/n_i$, while $s_i \equiv M_i^{-1}(\bmod n_i)$ (i=0, 1, 2, ..., k)

The Chinese Remainder Theorem as given in the miracle repository is given by: *bool crt_init (big_chinese* c, int r, big* moduli)*, *void crt (big_chinese* c, big* u, big x)* and *void crt_end* (big_chinese *c)*. These functions, initialize the big_chinese function using a modulus index; calculate to get the result x, and then end the operation.

III. Description of Algorithm

This algorithm utilizes the easiness of the super-increasing knapsack problem as its limitation and uses the double backpack sequence A and B in the process of its construction. It then makes the super increasing sequence S as one of a subset of a backpack, for instance, $S \subset B$, before using the Chinese remainder theorem to hide the two sequences to get a new sequence *X*, and applying necessary modulus transformation to *X* to get *X**. Through this, the overall security is enhanced since issue *X** is a public key while the related parameters and *S* are private keys. Experiments show that the speed of the algorithm is very fast in both of encryption and decryption. It is also easy to generate keys with the software implementation.

The main steps of the generation of keys, encryption and decryption are shown below:

A. The production of the key(shown in Figure 1):

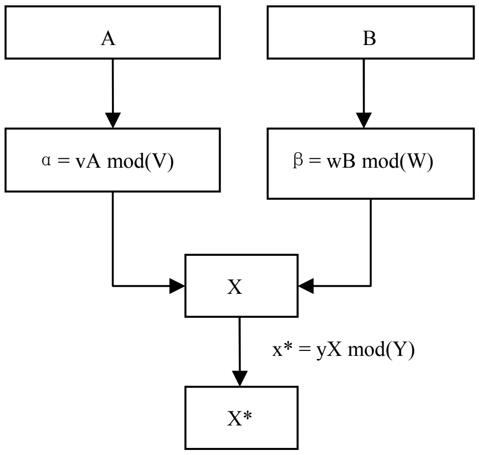

Figure 1. The production of the key

(1) Choose two sequences *A* and *B*, $A = (a_1, ..., a_n, ..., Z_1, ..., Z_u)$, $B = (a_1, ..., a_n, s_1, ..., s_u)$, in which, $t = n + u$. Besides, *B* satisfies that vector *S* is super increasing backpack vector. That is, $s_i > \sum_{j=1}^{u} s_j, (1<j<u)$; $Z_1, ... Z_u$, satisfy that $Z_1 = Z_2 = ... Z_u$, at the same time $Z_i > \sum_{i=1}^{u} a_i$;

(2) Choose modulus $V>|A|$ and $W>|B|$, $|A|$ and $|B|$ respectively is the sum of all the elements in A and B. Generate two random secret parameters *v* and *w* to satisfy gcd(*v*, *V*) = gcd (*w*, *W*) = 1, let α = vA mod (*V*), *β=wB mod(W);*

(3) Choose two modulus *p* and *q* which are coprime , plus, $p>|\alpha|$, $q>|\beta|$. $|\alpha|$ *and* $|\beta|$ respectively represents the sum of all elements in α and β. Let $N = pq$;

(4) Use Chinese remainder theorem to calculate *X*, aiming at (in order to) hiding α and β into *X*, and hiding A and B indirectly, that is:

$$\begin{cases} X = \alpha \bmod p \\ X = \beta \bmod q \end{cases} \tag{2}$$

5) Choose modulus $Y>|X|$, $|X|$ is the sum of the elements in vector *X*. Select the secret parameter y to make gcd(*y, Y*)=1, let *X* = yX mod(Y).*
public key: *(X*, n, u)*
private key: *(A, B, p, q, v, w, V, W, y, Y, Z)*

B. Encryption(shown in Figure 2 below):

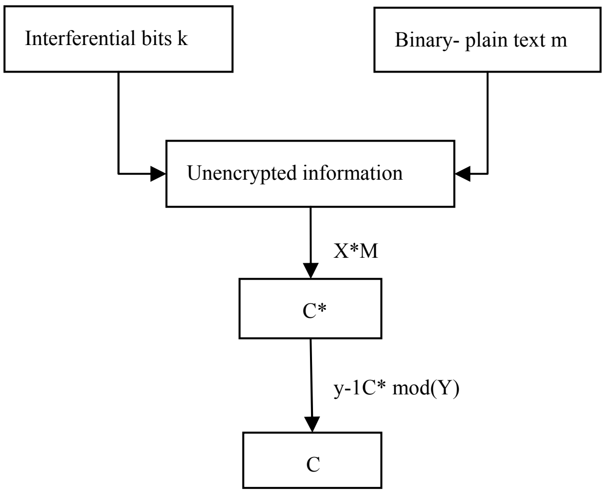

Figure 2. The process of the encryption

The plain text sequence is *m = (m₁, m₂, ..., m_u) ∈{0, 1}U* . We choose the random sequence *k =(k₁, ..., k_n) ∈{0, 1}n* , then combine the *k* and *m* to be an interfering-plaintext sequence $M = (k_1, ...k_n, m_1, m_2..., m_u)$ which is prepared for encryption .

After encryption, the cipher text is

$$C^*=X^*M=\sum_{i=1}^{t} X_i^* M_i \tag{3}$$

C. Decryption (shown in Figure 3 below):

(1) Calculate the cipher text $C=y^{-1}C^* \bmod(Y)$.
(2) Pick modulus *p* and *q* with the cipher text *C*, namely, $Dp = C \bmod p = XM \bmod p = \alpha M$, *similarly,* $Dq = \beta M$.

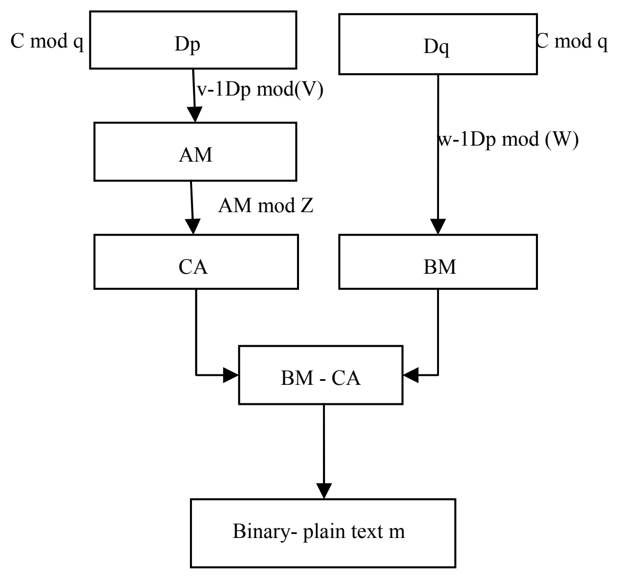

Figure 3. The process of the decryption

(3) Apply modulus-reverse transformation to α and β, that is $AM = \mathrm{v}^{-1}Dp\mathrm{mod V} = \mathrm{v}^{-1}\alpha M \bmod \mathrm{V}$, similarly we can get $BM = \mathrm{w}^{-1}Dq \bmod \mathrm{W}$

(4) Calculate :

$$\mathrm{CA} = \mathrm{AM} \bmod \mathrm{Z} = \sum_{\mathrm{i}=1}^{\mathrm{n}} \mathrm{a}_i k_i + \sum_{j=1}^{u} Zm_j (\mathrm{mod} Z) = \sum_{i=1}^{n} a_i k_i$$

(5) From the structure of *A* and *B*, we can know that

$$(BM\text{-}CA) = \sum_{i=1}^{u} s_i m_i \qquad (4)$$

Because *s* is super increasing backpack vector, that is, (*BM-CA)* can get the plaintext via working it out.

D. Algorithm Instructions:

(1) The choice of the parameters: The selection of the sequences *A* and *B* needs to meet the condition that is mentioned in the steps of the production of key in the algorism description above. Elements in *A* and *B* can be produced randomly, so they can be generated by the chaotic pseudo-random number generator [9]. Meanwhile, the interfering-bit *k* of the plaintext can be generated totally randomly.

(2) We may apply random transformation *f* (we can make choices freely while implementing) to the subsequence *S* of sequence *B*, after which, we can work out the plaintext through the inverse -transformation off.

(3) The cipher can add a secret parameter $\mathrm{h} \in \mathrm{Z}^+$, making $B = hB$ and $BM/h - CA$ in decryption. Thus the security of the algorism is strengthened in some degrees, for exhausting *h* is almost impossible.

E. An algorithm paradigm

Next, a complete algorithm paradigm will be given just as an example of encryption and decryption. Because of not selecting parameters strictly, the security of the paradigm will not be discussed.

(1) The production of the key:

a. Assume $n = 3$, $u = 6$, so $t = 9$; Let $A = (16, 50, 43, 112, 112, 112, 112, 112, 112)$, then $|A|=781$; $B=(16, 50, 43, 12, 17, 33, 74, 157, 316)$, then $|B|=718$. Among that, $s = (12, 17, 33, 74, 157, 316)$ is super-increasing knapsack sequence, $Z=112$;

b. Choose modulus $V = 863$, $W = 907$, let the secret parameter $v = 321$, $w = 58$. We can compute:
$\alpha = (vA) \bmod V = 321(16, 50, 43, 112, 112, 112, 112, 112, 112) \bmod 863$
$= (821, 516, 858, 569, 569, 569, 569, 569, 569)$,
$|\alpha| = 5609$;
$\beta = (wB) \bmod W = 58\ (16, 50, 43, 12, 17, 33, 74, 157, 316) \bmod (907)$
$=(21, 179, 680, 696, 79, 100, 664, 36, 188)$,
$|\beta|=2643$;

c. Choose two modulus which are coprime :$p = 5623$, $q = 2647$, then $N = 14884081$;

d. Use Chinese remainder theorem to calculate $X = (7361328, 7957061, 13434205, 7557881, 11392767, 5376157, 9638391, 9537177, 7096795)$, $|X|=63871155$;

e. Choosing modulus $Y = 6371219$, $y = 3976354$ and calculating:
$X^* = (24438697, 13965907, 13679387, 60510775, 54487483, 61877154, 49001559, 38858722, 42691726)$, then:
The public key is (X^*, 3, 6)
The private key is ($A, B, p, q, v, w, V, W, y, Y, Z$);

(2) Encryption:

Assume that the binary form of the plain text is $m = 101101$, after adding the interference signal: $k = 110$, then $M = (1, 1, 0, 1, 0, 1, 1, 0, 1)$, we can work out the cipher text $C^*=252485818$.

(3) Decryption:

a. Computing $C = (y\text{-}1C^*) \bmod Y = (3605945 * 252485818) \bmod 63871219 = 44987613$;

b. Computing $Dp = \alpha M = 3613$, $Dq=\beta M = 1848$;

c. Applying modulus-reverse transformation to α and β , we can get $AM = (164*3613) \bmod V = 514$, $BM = (735*1848) \bmod (W) = 501$;

d. Computing $CA=541 \bmod 112 =66$;

e. $BM\text{-}CA=435$, and *BM-CA* is a super-increasing knapsack set, using the process of solving the super-increasing knapsack problem, the plain text m = 101101 can be worked out.

IV. Analysis and Comparison of the Algorithm

A. Efficiency analysis

Suppose that the length of plaintext *M* is *k*, actually, the process of encryption is an addition, of which the time complexity is O (*k*). While in the course of decryption, a inverse modulus operation is carried out firstly; the liner operation of modulus is done(did) twice in the next step; then the liner operation of modulus is carried out once;

finally we can figure it out to get the plaintext, namely, execute the comparison operation with super increasing sequences for k times; each of their time complexity is O (k^2), O (k), 2* O (k^2), O (k) and O (k). So the total time complexity is 3*O (k^2) + 4*O (k). After ignoring the lower order, it is 3*O (k^2).

The complexities of the traditional public-key algorithms, RSA [10] and ElGamal [11], are both cubic. If the modulus length they use is 1, 024 bits in binary , the encryption of RSA will cost the bit operation about 10^9 times, whereas due to the use of operation and two modulus by computing a module, the encryption speed of ElGamal is about a half of that of RSA. This algorithm acquires addition for n-times in the process of encryption. If n = 1, 000, it will only take about 10^3 times bit operation to complete encrytion, which demonstrates the encryption speed is very fast. Besides, in the decryption process it will totally acquire 3*O (t^2) + 4*O(t) times calculations, which is about 3*10^6 times-operation, and 300 times faster than RSA public-key algorithms.

B. *Security analysis*

(1) In the aspects of the key, through the modulus Y, the randomness of vector A is able to cover the vector X, which is obtained by Chinese remainder theorem directly. The attacker cannot get X before getting Y and y, in such case he cannot estimate the size of the modulus N, let alone to get p and q by decomposing N, which means that he cannot get the information of the keys. Thus, the security is enhanced. In addition, $k = (k_1 , ..., k_n)$ is the random interferential signal that is independent from the plaintext and mixed with the value of plaintext calculation when being calculated. To the attackers who cannot distinguish the message of plaintext from interfering signal, what they can get at most is the encrypted information of interfered plaintext. Never could the attacker find out the corresponding numerical size in the plaintext.

(2) This algorithm is also able to resist the attacks of low-density subsets and LDA. According to the L^3– lattice base reduction algorithm, if the algorithm can always find a basis of the shortest nonzero lattice vector, so if $d = \frac{t}{\max \log_2 x_i}$ (note, $t = n + u$), and d < 0.9408, it is very likely to be attacked efficiently. Attack may be like this:

$$\begin{pmatrix} 1 & o & \lambda x_1 \\ & \cdots & \cdots \\ o & 1 & \lambda x_2 \\ \frac{1}{2} & \cdots \ \frac{1}{2} & \lambda S \end{pmatrix} \qquad (5)$$

λ is a proper integer which satisfies:

$$\lambda > \frac{1}{2}\sqrt{t}\text{ , S= } \sum_{i=1}^{t} x^*_i M_i$$

The cipher text is worked out by $C = XM$. Assuming that the lengths of the modulus q and p are nearly identical, namely, about W (bits), and the size of the modulus $N= p \bullet q$ is about $2log_2 W$ (from $N = p \bullet q$ we can know:

$log_2 N = log_2 (pq)$. That is $d = \frac{t}{2\log_2 W}$ .When $d > 1$, it requires $n + u > 2log_2 W$, that is, $u > 2log_2W - n$. When this condition is met, we can get $d > 1$; therefore it can resist L^3'attack. Thus, if the plaintext is the random interference of 120 bits-plaintext, $n = 20$, then only if the length of modulus N is less than 70 bits can it resist the attacks of the low-density subsets . It indicates that the security of this algorithm can be controlled.

The Table 1 display the comparison among the improved algorithm in references [1], MH algorithm and this algorithm .Meanwhile, to the time spent on encryption and decryption, we compare it to RSA-1024.

There is also a defect in this algorithm that the decryption of the cipher text by vector A and B, thus the strength of the security is weakened. An excellent design of algorithm should make decryption also depend entirely on sequences A and B. Two solutions can be adopted to weaken its influence on the safety of algorithm. One is to add a secret parameter h, making $B = hB$ which is impossible to search h exhaustively for the attackers; Another one is to prevent the leakage of private-key, no longer to store it after using vector, and recover it by using some certain methods.

V. THE GENERATING OF THE CHAOTIC PSEUDO-RANDOM NUMBER GENERATOR

During the process of generating the key in the algorithm, we specially designed a chaotic pseudo-random generator. It can produce the random elements of sequence A and B in the algorithm above with the method of interference of Logistic and Chebychev chaotic mapping. The random sequence satisfies the safety aspect of the requirements of cryptography and it provides the big integer for the use of cryptographic algorithm.

The Figure *4* is the flow chart of the generator.
The instructions of the process described below:

(1) Input the system parameters of Logistic, Chebychev mapping and the initial state respectively.
(2) x_i and x_che_i are global, which means that the value is changing with the system iteration.
(3) When x_che_i is less than 0, the system will enter into Logistic mapping of which the random parameter is $\mu(\mu\ [3.9, 4.0])$ for a iteration; otherwise entering the

TABLE 1: THE COMPARISON OF ALGORITHM CHARACTERISTICS

algorithm	Resist LDA?	Encryption time	Decryption time	Security can be controlled?
MH	NO	10^3	10^6	NO
References[1]	YES	10^3	10^8	YES
This algorithm	YES	10^3	10^6	YES
RSA-1024	--	10^9	10^9	YES

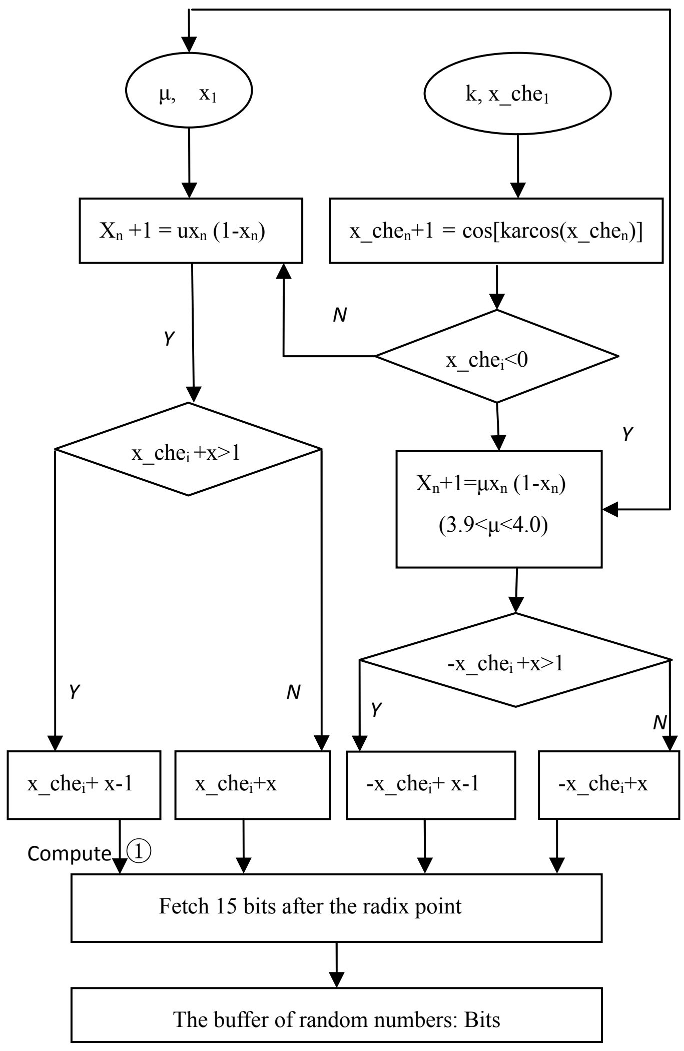

Figure 4. The flow chart of the generator

original mapping for iteration.

(4) When using the chaotic pseudo-random generator, the system should be iterated several times at first. After making the system in a stable state, the data is to begin to be filled in random buffer - bits.

(5) We only need to fetch the data from the Bits when necessary. For example, if we need to generate a random number which is in between double-digits and 50 digits, we just need to fetch the data from the Bits sequentially, and then convert it to a large number.

(6) The generator can generate not only decimal stream, but also binary sequence. A simple way is to output 1 when meeting the condition $((-)x\_che+1>1)$; otherwise , output 0.

The random generator can not only generate the decimal data flow, but also can be used to produce '01' sequence of binary. We only need to extract the data from Bits when putting it to use. Through bit-count based-test, byte-count based tests, self-correlated tests and the value distributed tests, it thoroughly shows good random stochastic characteristics.

V. CONCLUSION

In this paper we design a rapid public-key algorithm by making use of chaotic system which produces backpack interfering vector and using the Chinese remainder theorem to hide the double sequences; we also designed a chaotic pseudo-random generator to be applied to this algorithm. Through simulation and comparisons with some related algorithms transversely and longitudinally, this algorithm is excellent with a higher efficiency and better, controllable security, thus making up for the shortcoming of security which is put forward in the reference [7].

ACKNOWLEDGEMENTS

This work is supported by Aero-Science Fund of China (2009ZD53045), Science and Technology, Development Project of Shaanxi Province Project(2010K06-22g), Basic research fund of Northwestern Polytechnical University (GAKY100101), and R Fund of College of Software and Microelectronics of Northwestern Polytechnical University (2010R001).

REFERENCES

[1] Z. Yunpeng, L. Xia, L. Xi, An improved high-density knapsack-type public key cryptosystem, 5th International Conference on Software and Data Technologies, INSTICC, 2010, pp.127-133

[2] XIE Jianquan, XIE Qing, YANG Chunhua, HUANG Dazu, Security Analysis and Improvement of an Encryption Based on Logistic Map, Journal of Chinese Computer Systems, Vol.31, No.6, 2010

[3] JING Yong-wen, SONG Li-Qin, Substation information encryption method based on chaotic series, Journal of North China Electric Power University, Vol.36, No.6, 2009

[4] Z. Yunpeng, Z. Fei, Z. Zhengjuni, A Color Image Encryption Algorithm Based on Chaotic Chebychev and Variable-Parameters Logistic Systems, Journal of Nothwestern Polytechnical University, vol.28, no 4 (2010), pp.628-632

[5] S. Xi, The research and implementation of public-key algorithms based on chaos. University of Chongqing, Reading, MA, 2006.

[6] Han Li-dong, Liu Ming-jie, Bi Jing-guo, Security Analysis of Two Knapsack-Type Public Key Cryptosystems, Journal of Electronics & Information Technology, Vol.32, No.6, Jun.2010

[7] W. Bao-cang, H. Yu-pu, Knapsack-Type Public-Key Cryptosystem with High Density. Journal of Electronics & Information Technology, vol.28, no.12 (2006), pp.2390-2393

[8] J. Xue-juan, the origin and the solution of the "Chinese remainder theorem". Journal of jiujiang University(natural sciences), no.3(2004), pp.102-108

[9] Y. Zhen-Biao, F. Jiu-Chao, A method for generating chaotic-spectrum sequences and their optimized selection algorithm, ACTA PHYSICA SINICA, vol.57, no.3(2008), pp.1409-1415

[10] CHEN Cai-sen, WANG Tao, ZHANG Yuan-yuan, ZHAO Xin jie, Timing Attacks and Defenses on RSA Public-key Algorithms, Computer Engineering, Vol.35, No.2, January 2009.

[11] Rivcst R L, Gau M J. Adlcman L M. A method for obtaining digital signature and public key cryptosystems [J]. Communications of the ACM, 1978, 21(2): 120-126.