Journal of Discrete Mathematical Sciences & Cryptography
ISSN 0972-0529 (Print), ISSN 2169-0065 (Online)
Vol. 26 (2023), No. 1, pp. 215–229
DOI : 10.1080/09720529.2021.1930656

Increasing security to public key cryptography for point-to-point communication

Kritsanapong Somsuk *
Department of Computer and Communication Engineering
Faculty of Technology
Udon Thani Rajabhat University, UDRU
Udon Thani 41000
Thailand

Chalida Sanemueang
Office of Academic Resources and Information Technology
Udon Thani Rajabhat University, UDRU
Udon Thani 41000
Thailand

Abstract

In the beginning of 2020, the new idea of public key cryptography was proposed. This method is from the combining between RSA and Chebyshev maps. It is very difficult to break this system because it is based on both of Integer Factorization Problem (IFP) and Chaotic maps Discrete Logarithm Problem (CMDL). However, Chebyshev polynomials taking many computation costs (many modular multiplications and modular subtraction operations) are the equation to find the result of Chebyshev maps. In this paper, the new improvement of public-key cryptography is proposed for point-to-point communication. The main key is to reduce time and increase the security level. In fact, time is reduced by replacing some Chebyshev maps with modular exponentiation equations using only modular multiplication processes. In addition, the security level can be increased, because it is based on three problems, IFP, CMDL and Discrete Logarithm Problem (DLP). Although, DLP is included in the proposed method, it does not affect the computation time in both of encryption and decryption sides, because DLP is selected to exchange the RSA's public key. The experimental results show that time can be reduced in both sides especially in decryption side. The reason is that the private key of the compared method must be assigned too large, it is larger than

*E-mail: kritsanapong@udru.ac.th

the traditional RSA's private key. On the other hand, the private key for the proposed method is similar to RSA's private key.

Subject Classification: *94A60, 00A69, 68M25.*

Keywords: *RSA, Chebyshev maps, IFP, CMDL, DLP.*

1. Introduction

Long distance communication via computer network is the most popular way to exchange the information because of the fast process. However, computer network is known as the insecure channel. It becomes the system's drawback. Attackers can use this weak point to break the system. With this reason, the security for network communication is very important. Cryptography [1] is one of the security algorithms for protecting the information over the communication channel. It is divided into two different techniques. The first technique is symmetric key cryptosystem. It uses the same key, is called the secret key, in both of encryption and decryption sides. AES [2-3] is the strongest algorithm in this group. The advantage is about the high security and low computation costs. However, the problem of this group is about finding the channel to exchange the secret key between sender and receiver especially when they are far from each other. From the problem of the first technique, in 1976, the first method of asymmetric key cryptography or public key cryptography [4] which is the second technique was proposed by W. Diffie and M. Hellman. It is called Diffie-Hellman protocol. This method is selected to solve the problem of symmetric key cryptography, key - exchanging. The security is based on Discrete Logarithm Problem (DLP) [19] that is about the difficulty to find the exponent from the public base and result. Later, the next algorithm which is called RSA [5] was presented in 1978. In fact, RSA has more features than Diffie-Hellman protocol such as data encryption and digital signature. In fact, RSA is depended on Integer Factorization Problem (IFP) [6 - 9]. It means that if intruders want to break RSA, they have to factor the modulus as prime numbers at first before recovering the private key.

In 2020, N. Tahat et.al. [10] proposed the new idea of public key cryptography to increase the security level. This system is from the combination between RSA and Chebyshev maps [17]. That means it is based on both of IFP and Chaotic maps Discrete Logarithm Problem (CMDL) [18]. Therefore, the difficulty to break this system is increased.

However, Chebyshev polynomials take high computation costs consisting of many steps to compute both of modular multiplication and modular subtraction.

The aim of this paper is to propose the new public key cryptosystem. This method is modified from Tahat's method. In fact, the computation time to finish the process will be decreased by replacing some processes which use multi-Chebyshev polynomials (using many modular multiplication and modular subtraction operations) with only one modular exponentiation. Furthermore, the security level is increased because the proposed method is based on three problems consisting of DLP, IFP and CMDL.

2. Related Works

2.1 *RSA*

RSA [5] is one of the best well known public key cryptography. It was proposed by R. Rivest, A. Shamir and L. Adleman in 1978. The security is based on integer factorization problem. RSA is divided into three processes. The first process is key generation. After finishing this process, all public parameters are e and n, where e is the public key and $n = p*q$ is the modulus. On the other hand, all private parameters are p, q, $\Phi(n)$ and d, where p, q are two prime numbers, $\Phi(n)$ is Euler' s totient function and d is the private key. The second process is encryption. Sender uses e to encrypt the plaintext, m, by the following equation: $c = m^e \bmod n$, where c is ciphertext. The last process is decryption. In this process, receiver has to recover m by using the decryption equation: $m = c^d \bmod n$.

2.2 *Traditional Modular Exponentiation*

Traditional Modular Exponentiation (TME) is the easiest method to find the result from modular exponentiation equation. Assuming, TME is selected to find $t = a^b \bmod c$, where a, b and $c \in \mathbb{Z}$, the process is as following:

First, Assigning $t = a$, then

Step 1: find $a^2 \bmod c$, $t = a*t \bmod c = a^2 \bmod c$, → 1 modular multiplication

Step 2: find $a^3 \bmod c$, $t = a*t \bmod c = a*a^2 \bmod c = a^3 \bmod c$, → 1 modular multiplication

Step 3: find $a^4 \bmod c$, $t = a*t \bmod c = a*a^3 \bmod c = a^4 \bmod c$, → 1 modular multiplication

Therefore,

Step *b*-1: find $a^b \bmod c$, $t = a*t \bmod c = a*a^{b-1} \bmod c = a^b \bmod c$ → 1 modular multiplication

Therefore, TME is always takes $b-1$ modular multiplication processes where b is represented as the exponent. In addition, many algorithms which are more efficient than TME were already proposed to speed up modular exponentiation such as Fast Exponentiation [11] and Square-and-Multiply Algorithm [12].

2.3 *Square-and-Multiply Algorithm*

Square-and-Multiply Algorithm is one of the methods to compute modular exponentiation. The number of loops is based on the binary system and Hamming Weight of the exponent. In fact, total loops are equal to $l + h$, where l, h are the length and Hamming Weight of the exponent, in order. Assuming Square-and-Multiply Algorithm is selected to find $t = a^b \bmod c$, where $b = \sum_{i=0}^{m} b_i * 2^i$. The algorithm is as following:

Algorithm: Square-and-Multiply Algorithm

Input: a, b, c

Output: t

1. $z_0 \leftarrow 1, y_0 \leftarrow a, i \leftarrow 0$
2. While $i \leq m$ do
3. $\quad y_{i+1} \leftarrow y_i^2 \bmod c$
4. $\quad$ IF $b_i = 1$ then
5. $\quad\quad z_{i+1} \leftarrow z_i * y_i \bmod c$
6. $\quad$ Else
7. $\quad\quad z_{i+1} \leftarrow z_i$
8. $\quad$ End IF
9. $\quad i \leftarrow i+1$
10. End While
11. $t \leftarrow z_{i+1}$

In fact, if b is very large, computing modular exponentiation by using Square-and-Multiply Algorithm is very faster than TME.

2.4 *New RSA's Private Key*

In general, d is always larger than e to avoid attacking easily. However, it effects to the slow process in decryption side especially when Hamming Weight is very high. In 2016, the new RSA's private key [13] was proposed to speed up decryption process. In fact, the condition to increase speed is that the Hamming Weight of the new private key must be less than the Hamming Weight of d. Assuming, the new private key is computed from $d_t * e \bmod \Phi(n) = x$, where d_t is the new private key. Therefore, the decryption equation must be changed as $m = (c^{dt} \bmod n)^{1/x}$.

2.5 *Chebyshev Maps*

Chaotic maps [14] are known as the complex nonlinear dynamical system. These maps can be chosen to apply with many symmetric key cryptosystems. Chebyshev maps are the one-dimension map. In addition, they can be chosen to apply with some public key cryptosystems such as RSA. The Chebyshev polynomial in the field Z_n is as following:

Assigning m is represented as an integer, where $1 < m$, and x is an integer in finite field $(\mathbb{Z}_n)$, then Chebyshev polynomial, T_m is:

$$T_m(x) = (2xT_{m-1}(x) - T_{m-2}(x)) \bmod n \tag{1}$$

Where, $T_0(x) = 1$ and $T_1(x) = x$

Furthermore, the semi-group which is the significant property of Chebyshev polynomial is as following:

$$T_a(T_b(x)) \bmod n = T_{ab}(x) \bmod n \tag{2}$$

In fact, Chebyshev maps are based on CMDL. The process to compute $T_m(x)$, where x, $T_{m-1}(x)$ and $T_{m-2}(x)$ are the public parameters, is very easy. On the other hand, computing m from $T_m(x)$ is very difficult.

2.6 *Combining RSA and Chebyshev Maps*

In 2020, N. Tahat et.al. [10] proposed the new idea of public key cryptosystem which is from the combining between RSA and Chebyshev maps. That means this algorithm is based on both of IFP and CMDL. There are three processes as following:

1) **Key generation:** the process to generate the keys for encryption and decryption processes. It is divided as 9 steps as follows:

Step 1: Generate two prime numbers randomly, p and q

Step 2: Compute the modulus, n, $n = p*q$

Step 3: Compute Euler' s totient function, $\Phi(n) = (p^2 - 1)(q^2 - 1)$

Step 4: Select the public key e, where $1 < e < \Phi(n)$ and $\gcd(e, \Phi(n)) = 1$

Step 5: Compute the private key d from: $e*d \bmod \Phi(n) = 1$

Step 6: Select two integers randomly, a and b, where $0 \leq a, b < \Phi(n)$

Step 7: Select two integers randomly, h and t, where $h, t \in \mathbb{Z}_n^*$

Step 8: Compute $y_1 = T_{a^2}(h) \bmod n$

Step 9: Compute $y_2 = T_{b^2}(t) \bmod n$

The public parameters = $\{n, e, y_1, y_2, h, t\}$

The private parameters = $\{p, q, a, b, d\}$

2) **Encryption process:** the process to convert m as the unreadable message by using the following steps:

Step 1: Select $r \in \mathbb{Z}_n^*$ randomly

Step 2: Compute $s_1 = T_e(r) \bmod n$

Step 3: Select two integers randomly, c and f, where $c, f \in \mathbb{Z}_n$

Step 4: Compute $s_2 = T_c(h) \bmod n$

Step 5: Compute $s_3 = T_f(t) \bmod n$

Step 6: Compute $s_4 = mT_c(y_1)T_f(y_2)T_e(r+1) \bmod n$

Ciphertext = $\{s_1, s_2, s_3, s_4\}$

3) **Decryption process:** the process to recover m from s_1, s_2, s_3 and s_4 as follows:

Step 1: Recover r from $r = T_d(s_1) \bmod n$

Step 2: Compute $R = s_4 T_e^{-1}(r + 1) \bmod n$

Step 3: Compute $u = T_{a^2}(s_2) \bmod n$

Step 4: Compute $v = T_{b^2}(s_3) \bmod n$

Step 5: Recover m from $m = (Ru^{-1}v^{-1}) \bmod n$

In fact, the way to recover m by attackers is to solve both of IFP and CMDL. Moreover, r, c and f in encryption process will be left suddenly after all of them are chosen for implementation. Furthermore, they will be generated again in the next encryption process to make the system more secure.

3. The Proposed Method

The aims of this paper are to decrease time and increase security level for the public key cryptosystem which is from the combining between RSA and Chebyshev maps. This method is applied with only point-to-point communication. Before discussing the concept of the proposed method, two problems of the combining between RSA and Chebyshev maps [10] will be mentioned at first:

Problem 1: it seems that d cannot be generated from $\Phi(n) = (p-1)(q-1)$, the example is shown in Table 1.

Assuming H is the representation of the following parameters, $p = 11$, $q = 37$, $n = 407$, $\Phi_1(n) = (p-1)(q-1) = 360$, $\Phi_2(n) = (p^2 - 1)(q^2 - 1) = 164160$, $e = 23$, $d_1 = 47$ (d_1 is from $d_1 * e \bmod \Phi_1(n) = 1$), $d_2 = 35687$ (d_2 is from $d_2 * e \bmod \Phi_2(n) = 1$), $a = 19$, $b = 31$, $h = 11$, $t = 23$, $r = 59$, $c = 62$ and $f = 35$, all decrypted messages, $m = 1$ to 406 by using the method [10] with d_1 and d_2 are shown in table 1.

Table 1

Decryption message from *H'* s parameters

Original Plaintext	Decrypted message	
	$d_1 = 47$	$d_2 = 35687$
1	353	1
2	299	2
3	245	3
4	191	4
5	137	5
6	83	6
7	29	7
8	382	8
9	328	9
10	274	10
…	…	…
405	108	405
406	54	406

The information in table 1 shows that if d_1 is chosen, the error has occurred 100%. On the other hand, the accuracy becomes 100% when d_2 is chosen.

Therefore, the disadvantage is about time to finish decryption process because d (d_2) is too large.

Assuming, $T_{m-2}(x)$ and $T_{m-1}(x)$ are known, where m is a prime number, the process to find $T_m(x)$ takes 1 modular multiplication and 1 modular subtraction. That means, the cost is larger than modular multiplication. It implies that the process to compute $a^b \bmod c$, where a, b and $c \in \mathbb{Z}$ and $a, b < c$, by using the TME takes $b - 1$ modular multiplication processes but the process to compute $T_b(a) \bmod c$ takes $b - 1$ modular multiplication processes and $b - 1$ modular subtraction processes, without using semi-group property. Therefore, it concludes that modular exponentiation computing ($a^b \bmod c$) by using TME takes time less than the process to find $T_b(a)$. The conclusion of the first problem is about the large value of d and the high cost of $T_b(a)$'s computing which must be implemented in many parts.

Problem 2: it is based on only both of IFP and CMDL.

In fact, the main ideas of the proposed method are divided into two parts. The first part is to speed up both of encryption and decryption processes by replacing Chebyshev maps with modular exponentiation operation. In addition, step 2 and step 6 in encryption process and step 1 and step 2 in decryption process are changed from Chebyshev maps as modular exponentiation. Moreover, the process in key generation must be modified a little. The second part is to use Diffie-Hellman protocol or the other improved techniques to exchange e to hide this value from the intruders. Assigning Alice and Bob want to communicate each other secretly, the proposed method is divided into four processes as following:

Process 1 (Key-Exchanging protocol): the process to generate e for point-to-point communication by using Diffie-Hellman protocol or the other improved techniques. However, e must be an odd number. After finishing this process, e will be share between sender and receiver.

Process 2 (Key generation): Alice generates the keys by using the following steps:

Step 1: Generate two prime numbers randomly, p and q

Step 2: Computing the modulus, n, $n = p^*q$

Step 3: Computing Euler' s totient function, $\Phi(n) = (p - 1)(q - 1)$

Step 4: The process will return to step 2 whenever gcd(e, $\Phi(n)$) $\neq 1$. On the other hand, jumping to the step 5.

Step 5: Compute private key d from: $e*d \bmod \Phi(n) = 1$

Step 6: Select two integers randomly, a and b, where $0 \leq a, b < \Phi(n)$

Step 7: Select two integers randomly, h and t, where $h, t \in \mathbb{Z}_n^*$

Step 8: Compute $y_1 = T_{a^2}(h) \bmod n$

Step 9: Compute $y_2 = T_{b^2}(t) \bmod n$

The public parameters = $\{n, y_1, y_2, h, t\}$

The private parameters = $\{p, q, a, b, d\}$

Point-to-point communication's parameter = $\{e\}$

Process 3 (Encryption process): Bob converts m as the unreadable message by using the following steps:

Step 1: Select $r \in \mathbb{Z}_n^*$ randomly

Step 2: Compute $s_1 = r^e \bmod n$

Step 3: Select two integers randomly, c and f, where $c, f \in \mathbb{Z}_n$

Step 4: Compute $s_2 = T_c(h) \bmod n$

Step 5: Compute $s_3 = T_f(t) \bmod n$

Step 6: Compute $s_4 = mT_c(y_1)T_f(y_2)(r+1)^e \bmod n$

Ciphertext = $\{s_1, s_2, s_3, s_4\}$ which will be sent to Alice

Process 4 (Decryption process): Alice will recover m from s_1, s_2, s_3 and s_4 by using the following steps:

Step 1: Recover r from $r = s_1^d \bmod n$

Step 2: Compute $R = s_4(r+1)^{-e} \bmod n$, using Extended Euclidean Algorithm [15 – 16] to finish modular inverse

Step 3: Compute $u = T_{a^2}(s_2) \bmod n$

Step 4: Compute $v = T_{b^2}(s_3) \bmod n$

Step 5: Recover m from $m = (Ru^{-1}v^{-1}) \bmod n$

Theorem m can be recovered by using the proposed method

Proof: From,

$$(Ru^{-1}v^{-1}) \bmod n = s_4(r+1)^{-e} T_{a^2}^{-1}(s_2)\ T_{b^2}^{-1}(s_3) \bmod n$$

$$= mT_c(y_1)T_f(y_2)(r+1)^e(r+1)^{-e}\ T_{a^2}^{-1}(s_2)\ T_{b^2}^{-1}(s_3) \bmod n$$

Because, $s_2 = T_c(h) \bmod n$, then

$$T_{a^2}(s_2) \bmod n = T_{a^2}\ T_c(h) \bmod n$$

$$= T_c T_{a^2}(h) \bmod n$$

$$= T_c(y_1) \bmod n$$

Because, $s_3 = T_f(t) \bmod n$, then

$$T_{b^2}(s_3) \bmod n = T_{b^2}\ T_f(t) \bmod n$$

$$= T_f T_{b^2}(t) \bmod n$$

$$= T_f(y_2) \bmod n$$

Then,

$$(Ru^{-1}v^{-1}) \bmod n = mT_c(y_1)T_f(y_2)(r+1)^e(r+1)^{-e}\ T_c^{-1}(y_1)\ T_f^{-1}(y_2) \bmod n = m$$

□

Example: Assuming, after Alice and Bob agree each other to share $e = 19$ and Bob wants to send the secret message to Alice. The process for Bob and Alice is as following:

Key generation process: Alice's keys

Step 1: $p = 17$ and $q = 53$

Step 2: $n = 901$

Step 3: $\Phi(n) = 832$

Step 4: Because gcd(19, 832) = 1, then we can continue jump to Step 5

Step 5: $d = 219$

Step 6: $a = 12$ and $b = 23$

Step 7: $h = 13$ and $t = 17$

Step 8: $y_1 = 596$

Step 9: $y_2 = 612$

Encryption process: Bob encrypts message, $m = 111$, before sending to Alice

Step 1: $r = 29$

Step 2: $s_1 = 555$

Step 3: $c = 30$ and $f = 34$

Step 4: $s_2 = 456$

Step 5: $s_3 = 560$

Step 6: $s_4 = 202$

Decryption process: Alice recovers m from the following parameters, s_1, s_2, s_3, s_4, which are sent from Bob

Step 1: $r = 29$

Step 2: $R = 6060$

Step 3: $u = 562$

Step 4: $v = 866$

Step 5: $m = 111$

In fact, the proposed method has two advantages as following

Advantage 1: it is based on DLP, IFP and CMDL. Therefore, if attackers want to recover m, then they have to find e (DLP's solving) p, q (IFP's solving) and a, b, c and f (CMDL's solving). Therefore, it is very difficult to solve three problems in polynomial time.

Advantage 2: the speed of the proposed method to finish the processes in both of encryption process and decryption process is faster than the same processes in [10]. The reason is that some processes using multi-Chebyshev polynomials are replaced by only one modular exponentiation. Moreover, d in [10] is very larger than the proposed method, because it is generated from $\Phi(n) = (p^2 - 1)(q^2 - 1)$.

4. Experimental Results

In this section, the comparison about the costs to compute modular multiplication and modular subtraction in both of encryption and decryption sides between the proposed method and the method in [10], RSA + Chebyshev maps, is discussed. All parameters are from the example 1. However, for the compared method, d which is in example 1, cannot

Table 2

The comparison between the proposed method and the method in [10]

Method	Encryption's costs		Decryption's costs	
	Modular Multiplication	Modular Subtraction	Modular Multiplication	Modular Subtraction
RSA+ Chebyshev maps (Method in [10])	163	160	341198	341195
The Proposed Method	163	124	910	671

be chosen. Therefore, d and $\Phi(n)$ must be changed (only the compared method) as 340507 and 808704, respectively.

The information in Table 2 shows that the proposed method takes both of modular multiplication and modular subtraction processes less than the compared method in both of encryption and decryption sides, especially in decryption side that d for the compared method must be assigned very large. In fact, modular inverse is not included, because this cost is stable, 3 modular inverses in decryption process.

However, the example to find the number of both of modular multiplication and modular subtraction processes in encryption process of the proposed method is as following, assigning M is the representation of number of modular multiplication processes and S is the representation of number of modular subtraction processes:

Step 1: select r → 0M, 0S
Step 2: find s_1 → 18M, 0S ($e = 19$)
Step 3: select, c and f → 0M, 0S
Step 4: find s_2 29M, 29S ($c = 30$)
Step 5: find s_3 33M, 33S ($f = 34$)
Step 6: find s_4 83M, 62S ($c = 30$, $f = 34$, $e = 19$ and 3M)

Therefore, in encryption process, the proposed method takes 163 number of modular multiplication processes and 124 number of modular subtraction processes.

Table 3

The comparison about Security Level and Time between the proposed method and the method in [10]

Method	Security Level	Time
RSA+ Chebyshev maps (Method in [10])	HIGH (CMDL + IFP)	Very SLOW
The Proposed Method	Very HIGH (CMDL + DLP + IFP)	SLOW

Moreover, the number of modular multiplication processes to finish modular exponentiation can be reduced by using the other method such as Square-and-Multiply Algorithm. For example, step 2 in encryption process of the proposed method, there are only 7 modular multiplication processes left ($e = 19 = 10011_2$) when Square-and-Multiply Algorithm are chosen instead of TME.

Assuming d has high Hamming Weight but d_t ($d_t * e \bmod \Phi(n) = x$) has low Hamming Weight and x is a small positive integer, then the equation in step 1 of the proposed method's decryption can be changed as the decryption equation in [13].

The information in Table 3 is the conclusion about the security level and time to finish process. It is shown that the proposed method is more secure than the method in [10]. Furthermore, it is also faster than this compared method.

5. Conclusion

In this paper, the new improvement of public key cryptosystem for point-to-point communication is proposed to decrease computation time and increase security level. In fact, the method in [10] is the latest public key algorithm. It is based on both of IFP and CMDL. Nevertheless, the proposed method is different because it is based on three problems consisting of DLP, CMDL and IFP. That means it is very difficult to break this system. In addition, the speed does not increase because Diffie-Hellman protocol is only chosen to exchange the public key, it does not affect to both of encryption and decryption processes. On the other hand, time to finish process is decreased. The reason is that some steps which are based on Chebyshev polynomials are replaced by modular exponentiation. In fact, when the same loops are required, modular exponentiation is faster than Chebyshev maps, because modular subtraction processes which

are not included in modular exponentiation computing are required in Chebyshev maps.

References

[1] S. Kanoje, D. Mukhopadhyay and S. Girase, "User Profiling for University Recommender System using Automatic Information Retrieval", *Procedia Computer Science*, vol.78, pp. 5 – 12, 2016.

[2] L. Li, J. Fang, J. Jiang, L. Gan, W. Zheng, H. Fu and G. Yang, "Efficient AES implementation on Sunway TaihuLight supercomputer: A systematic approach", *Journal of Parallel and Distributed Computing*, vol.138, pp.178 – 189, 2020.

[3] D. Gerault, P. Lafourcade, M. Minier and C. Solnon, "Computing AES related-key differential characteristics with con-straint programming", *Artificial Intelligence*, vol. 278, pp.1 – 32, 2020.

[4] W. Diffie and M.E. Hellman, "New directions in cryptography", *IEEE Transactions on Information Theory*, vol. 22, pp. 644 – 654, 1976.

[5] R.L. Rivest, A. Shamir and L. Adleman, "A method for obtaining digital signatures and public key cryptosystems", *Communications of ACM*, vol. 21, pp. 120 – 126, 1978.

[6] K. Somsuk, "The improvement of initial value closer to the target for Fermat's factorization algorithm", *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 21(7 – 8), pp. 1573 – 1580, 2018.

[7] K. Somsuk and K. Tientanopajai, "An Improvement of Fermat's Factorization by Considering the Last m Digits of Modulus to Decrease Computation Time", *International Journal of Network Security*, vol. 19, pp. 99 – 111, 2017.

[8] M.E. Wu, R. Tso and H.M. Sun, "On the improvement of Fermat factorization using a continued fraction technique", *Future Generation Computer Systems* vol. 30(1), pp.162 – 168, 2014.

[9] G. Pandey, "Polynomial selection in number field sieve for integer factorization", *Perspectives in Science*, vol. 8, pp. 101 – 103, 2016.

[10] N. Tahat, A. A. Tahat, M. Abu-Dalu, R.B. Albadarneh, A.E. Abdallah and O.M. Al-Hazaimeh, "A new RSA public key encryption scheme with chaotic maps", *International Journal of Electrical and Computer Engineering*, vol. 10(2), pp.1430 – 1437, 2020.

[11] J.A. Buchman, "Introduction to Cryptography", *Springer-Verlag*, Germany, 2004.

[12] O. Nibouche, M. Nibouche and A. Bouridane, "Highspeed FPGA implementation of RSA encryption algorithm", in: *Proc. International Conference on Electronics, Circuits and Systems*, 2003, pp. 204-207.

[13] K. Somsuk, "The improving decryption process of RSA by choosing new private key", in: *Proc. International Conference on Information Technology and Electrical Engineering*, 2016, pp. 1 - 4.

[14] D. Arroyo, G. Alvarez, S. Li, C. Li and J. Nunez, "Cryptanalysis of a discrete-time synchronous chaotic encryption system", *Physics Letters A*, vol. 372(7), pp. 1034 – 1039, 2008.

[15] M.M. Asad, L. Marouf, Q. A. Al-Haija and A. Alshuaibi, "Performance Analysis of 128-bit Modular Inverse Based Extended Euclidean Using Altera FPGA Kit", *Procedia Computer Science*, vol. 160, pp. 543 – 548, 2019.

[16] Q. Zhou, C. Tian, H. Zhang, J. Yu and F. Li, "How to securely outsource the extended euclidean algorithm for large-scale polynomials over finite fields", *Information Sciences*, vol. 512, pp. 641 – 660, 2020.

[17] A. Shakiba, "Security analysis for chaotic maps-based mutual authentication and key agreement using smart cards for wireless networks", *Journal of Information and Optimization Sciences*, vol. 40(3), pp. 725 - 750, 2019.

[18] M.A. Khan , H.P. Mazumdar and S.D. Jabeen, "Anti-synchronization phenomenon of discrete chaotic maps using linear transformations", *Journal of Information and Optimization Sciences*, vol. 41(8), pp. 1757-1769, 2020.

[19] N. Tahat, A. K. Alomari , O.M. Al-Hazaimeh and M.F. Al-Jamal, "An efficient self-certified multi-proxy signature scheme based on elliptic curve discrete logarithm problem", *Journal of Discrete Mathematical Sciences and Cryptography*, vol. 23(4), pp. 935-948, 2020.

Received November, 2020

Revised April, 2021