Accelerating Asymmetric-Key Cryptography using Parallel-key Cryptographic Algorithm (PCA)

Thongpon Teerakanok[1] and Sinchai Kamolphiwong[2]
[1]Buranarumluk school, Trang, Thailand
[2]Department of Computer Engineering, Faculty of Engineering, Prince of Songkla University, Thailand
Email: thongpon.te@gmail.com[1] and ksinchai@coe.psu.ac.th[2]

***Abstract*- In this day, the necessity of security and protection of computer information on network has become more and more important. In cryptography, asymmetric-key is the framework with high flexibility which can be applied to most of cryptographic systems using today. To reduce the encryption and decryption time which is the main drawback of asymmetric-key cryptography, we have proposed a new mechanism called "Parallel-key Cryptographic Algorithm (PCA)" which accelerates the cryptographic system in encryption and decryption process and strengthens the system against *Brute force attack*. We have shown that, in our practical experiment results, our proposed algorithm can perform faster in encrypting and decrypting message than other asymmetric-key cryptographic algorithm; RSA (Rivest, Shamir and Adleman). In our theoretical analysis, we have shown that PCA can provide the security against *Brute force attack* better than other algorithms. Furthermore, PCA can be applied to parallel computing and cryptographic mode such as Cipher Block Chaining (CBC) and Interleaved Cipher Block Chaining (ICBC) for higher efficiency.**

Keywords: PCA, RSA, Brute force attack, Cipher block chaining, CBC, ICBC

I. INTRODUCTION

In the field of computer security, there is a large quantity of papers discussing on cryptography. In cryptography, there are two types of mechanisms based on key usage [1]: symmetric-key, and asymmetric-key. In symmetric-key cryptography, the same key is used for both encryption and decryption. This mechanism requires a sender and a receiver agree on a secret key before they can communicate securely [2]. From this requirement, there is a big disadvantage on a key exchanging and a limitation on the number of keys which is up to n^2, where n is a number of participants in a group. As we will see, a large number of keys needs to be maintained in group [3]. While in asymmetric-key cryptography, a number of keys which need maintaining in group is only $2n$. Asymmetric-key mechanism, however, has a big disadvantage over symmetric-key in terms of encryption and decryption process. Our proposed mechanism will come over this issue.

There is a number of multiple-key encryptions proposed in literatures. Most of them work on the digital signature [4]. In the study of parallel encryptions, it can perform in and purpose to many applications e.g. message authentication (signcryption) [5], [6]. For instance, some of them are: the solutions of solving key agreements or key exchanges problem, and the cryptographic algorithm for Color Images were proposed in [7].

In modern information block cipher, the decryption algorithm must be the inverse of encryption algorithm, and both algorithms have to use the corresponding key to each other (for asymmetric-key) so that receiver can recover the plaintext sent by sender. There is a mode of operation, called "Cipher Block Chaining (CBC)", which each plaintext block is exclusive-ored with the previous ciphertext block. This method can prevent an algorithm from replay attack which is the main method in attacking of traditional Electronic Code Book (ECB) [2].

This paper has proposed a new cryptographic algorithm using the feature of asymmetric-key cryptography, and the advantage of adding a couple of a key pair to accelerate the algorithm. Our proposed algorithm is strengthen against the problem of *Factorization attack* and *Brute force attack*, as well as it allows users to communicate faster and more secure through public insecure channel, such as internet. This makes our proposed mechanism more attractive.

This paper is organized as follows: The proposed algorithm of our scheme is described in section 2. Section 3 shows the results of our experiment. The experiment result analysis is given in section 4. Section 5 shows how our scheme can improve of prevention on attacking. At the end, we conclude our paper in section 5.

II. PROPOSED ALGORITHM

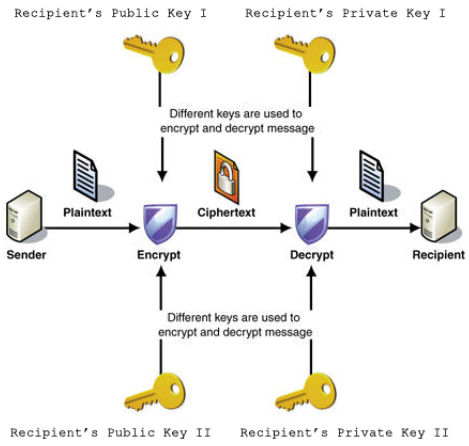

Figure 1 Parallel-key Cryptographic Algorithm (PCA)

978-1-4244-3388-9/09/$25.00 ©2009 IEEE

Figure 1 shows the main process of our proposed algorithm, so called Parallel-key Cryptographic Algorithm (PCA). The protocol in sending plaintext from a sender to a recipient contains a few methods which can be classified as follows:

- ***Key generating process*:** This is a method of generating a couple of key pair. The first key pair has a key length more than the second one.
- ***Encryption*:** This is a method allowing users to create a ciphertext (or secret message) by using two public-keys encrypting a plaintext (or an original clear text) with a special random pattern called "*Extension*". First, in our experiment, we assume that the encryption performed by PCA is the same as RSA with two key pairs. Then, the process will randomly separate plaintext into two parts (for each public-key is encrypting a message). After that we encrypt the two parts with two public-keys, as shown in Figure 2.

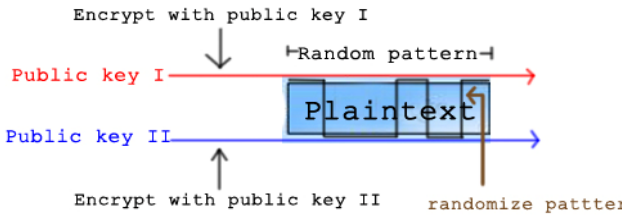

Figure 2 Encryption process of Parallel-key Cryptographic Algorithm

Figure 2 shows the encryption process of two Public-keys (called Parallel-key) which each key will encrypt plaintext randomly, and the process of two keys is independent. Key length of each key pair is not equal to each other. Firstly, key pair has a long term key same as RSA (called "*Major key*"), so we use the length of *Major key* as the representative key length of this algorithm. Another one has a short term of key (called "Minor key"), this key is the key to accelerate encryption and decryption process by using cryptographic method mentioned above. Only true recipient can decrypt cipher text correctly by using the *Extension* which is sent to recipient as attachment following the ciphertext. At the end of encryption process, the *Extension* mentioned above has to be encrypted by both public-keys (from *Major* and *Minor key*). From this idea of encrypting plaintext with two public-keys (as shown in Figure 2), now the attackers must do much more than only to break equations of two asymmetric-key pairs and to get the original plaintext correctly.

- ***Decryption:*** First, decryption process has to decrypt the *Extension* with two private-keys (from *Major* and *Minor key*). After the *Extension* decrypted with two private-keys, now we have to decrypt the ciphertext by following to the extension which already decrypted, just like the encryption process. The reason for adding *Minor key* in encryption and decryption process is to accelerate the algorithm without reducing the level of security. A speed of algorithm depends on key lengths. Obviously, using two pairs of keys with a shorter key length of one key pair will accelerate this algorithm. By this principle, now PCA is able to encrypt and decrypt messages confidentially just like other cryptographic algorithms. The decryption process of PCA is shown in Figure 3 below:

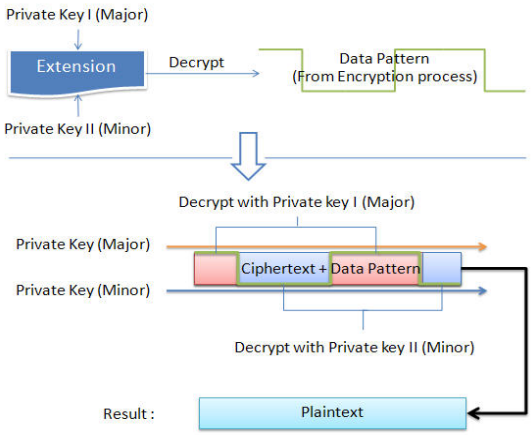

Figure 3 Decryption process of Parallel-key Cryptographic Algorithm

III. Experiment Results

In our experiments, the proposed algorithm was tested on a PC equipped with a dual-core processor at 2.4 GHz and 3 GB of RAM on MS-Windows OS. In this experiment, we used *Major key* to encrypt 50% of plaintext blocks. The length of *Major key* is 1024 bits which equal to the key length of RSA used in this experiment. We have found that, our algorithm used encryption and decryption time less than RSA, as shown in the table below.

TABLE I
Experiment Results

Algorithm	Key Length	Time of Process (s)	
		Encryption	Decryption
PCA	512 bits (Minor)	2.945646	13.7334545
	640 bits (Minor)	3.202464	15.0622222
	768 bits (Minor)	3.592111	17.1712626
	896 bits (Minor)	4.093798	19.9112525
	1024 bits (Minor)	4.562020	23.6499191
RSA	1024 bits	4.535666	23.68255

In practical experiment result, the key length of PCA is represented by *Major key* length which is the main key pair in preventing algorithm from Brute force attack.

From the experiment result, it shows that PCA can encrypt and decrypt messages faster than RSA. In practical, *Minor key* length should longer or equal to 512 bits to prevent the *Factorization Attack*. By using the feature of parallel-key mechanism, for more speed, PCA can be applied to parallel computing and some cryptographic mode e.g. Interleaved Cipher Block Chaining (ICBC) for more efficiency [8], [9].

IV. ATTACKS ANALYSIS

There is a number of attacks on Asymmetric-key cryptography which PCA based on. Hence, attacks on Asymmetric key are the same types on PCA. The important types of attack on PCA can be categorized as follows:

- *Factorization Attack*
- *Brute force attack*

***Factorization Attack*:** a type of attack which tries to find the factors of a large number (called "*Public Modulus*", *n*) with the polynomial time complexity in a short possible period of time [2], [10]. By using the parallel-key pair (two key pairs), PCA can prevent the message from attacker in more secure because PCA has two key pairs. Then, attacker has to do much more work in breaking two asymmetric-key pair with *Public Modulus* of their own.

***Brute force attack*:** a type of attack which tries to find all possible private-key for complete decryption. In practical, the small private-key *d*, would make the decryption faster. But, from the study of key size, Wiener showed that a small size of *d* is easy to break with the special type of attack based of *continuous fraction* [11], [12]. Then, for RSA, the recommendation is to have $d \geq 1/3n^{1/4}$ to prevent low decryption exponent attack, where *n* is a *Public Modulus*. By using a benefit of parallel-key mechanism, PCA can fight against *Brute force attack* better than common asymmetric-key cryptographic algorithm. In theoretical calculation of *Brute force attack*, we assume that *Major key* and *Minor key* length are 1024 and 512 bits, respectively. In this calculation, PCA will be compared with RSA (1024 bits) by assuming that the *attack on both algorithms operated on the same computer which can perform 1 million operations per second.*

First, in RSA 1024 bits, the value of Public Modulus (*n*) should be 2^{1024}. Then, from the recommendation of RSA mentioned above, we assume that private-key *d* value is equal to $1/3n^{1/4}$ which is the lowest value that still prevent algorithm from low decryption exponent attack. Hence, the maximum value of private-key (d_{RSA}) should be $\frac{2^{256}}{3}$. Thus, the time of *Brute force* on RSA 1024 bits (t_{RSA}) is shown as follow:

$$t_{RSA} = \frac{d_{RSA}}{y} = \frac{\frac{2^{256}}{3}}{(365\times24\times3600)\times10^6} = 1.224 \times 10^{63} \text{ years} \quad (1)$$

From (1), *y is a number of operations in attacking which can be operated in one year with 1 million operations/second processor.* After the calculation of *Brute force attack* on RSA, in PCA, the summation of private-key value (d_{PCA}) from *Major* and *Minor key* should be $d_{major} + d_{minor}$. Hence, the simple calculation of time *Brute force attack* on *Major* and *Minor key* ($t_{major+minor}$) should be $\frac{d_{PCA}}{y}$.

After the adversary tries to decrypt a ciphertext with all possible private-key values (from both key pairs, *Major* and *Minor key*), each case of decryption produce a decrypted message which only two of them (one from the attack on *Major* and the other from *Minor key*) are match to each other. Thus, in the next step of *Brute force attack* on PCA, attacker has to choose one decrypted message from the result of attack on *Major key* and another one from the attack of *Minor key*.
In the attack of *Major* and *Minor key*, the number of decrypted messages should equal to d_{major} and d_{minor} respectively. Therefore, time of selecting two corresponding messages (t_{select}) mentioned above is shown as follow:

$$t_{select} = \frac{\prod_{i=1}^{h} d_{Major_1} \times \prod_{i=1}^{k} d_{Minor_1}}{y} \quad (2)$$

Where *h* and *k* are a number of *Major* and *Minor key* respectively.

After the process of selection, the next step to complete *Brute force attack* on PCA is to find a *reliable plaintext* by combining two corresponding decrypted messages mentioned above. In this theoretical calculation, we assume that the RSA and PCA operate on *l* blocks of plaintext. The method of selecting and combining two corresponding messages is shown in Figure 3.

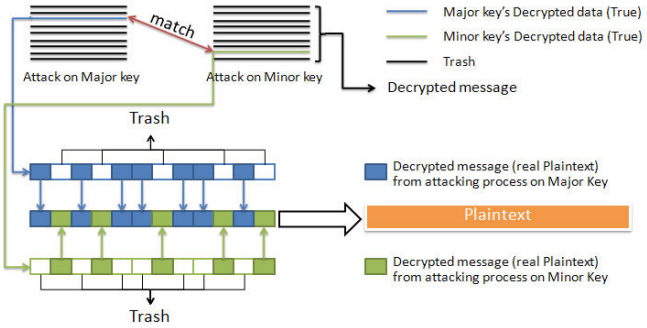

Figure 4 the method of selecting and combining two decrypted messages

As we will see, the number of possibility in combining two blocks of messages is up to $(h+k)!$. Thus, the time of combining ($t_{combine}$) decrypted messages from two key pairs should be $\frac{2^l}{y}$.

From processes of *Brute force attack* on PCA, the summation of time (t_{PCA}) is shown as follow:

$$t_{PCA} = t_{major+minor} + (t_{select} \times t_{combine}) \quad (3)$$

Hence,

$$t_{PCA} = \frac{d_{major} + d_{minor} + (\prod_{i=1}^{h} d_{Major_i} \times \prod_{i=1}^{k} d_{Minor_i}) + (h+k)^l}{y}$$

From (3), time of *Brute force attack* on PCA with 1024 bits (*Major*), 512 bits (*Minor*) and 128 blocks data compared with RSA is shown in Table II:

TABLE II
TIME OF BRUTE FORCE ATTACK

Algorithm	Time of Brute force attack (years)
PCA 512 bits (Minor)	$(1.224 \times 10^{63}) + (3.596 \times 10^{24}) + (1.497 \times 10^{126})$
RSA 1024 bits	1.224×10^{63}

From the difference of time in Brute force attack on RSA and PCA, it shows that PCA can provide better security against *Brute force attack.* As we will see, in the calculation of *Brute force attack* on PCA and RSA, PCA can provide security of a plaintext longer approximately than RSA 1.497×10^{126} years by attacking with 1 million operations /second computer.

V. Results Analysis

From the practical experiment result on encryption and decryption of PCA, it shows that PCA can perform better result in both encryption and decryption process without lacking of security. The acceleration over RSA in encryption and decryption processes of PCA (Ac_{PCA}) is calculated as follow:

$$Ac_{PCA} = (1 - \frac{Encryption\ or\ Decryption\ Time}{RSA}) \times 100\% \qquad (4)$$

Following to (4), the accelerations of PCA over RSA are shown in table below:

TABLE III
Acceleration of PCA

Algorithm	Key Length	Acceleration of PCA (%)	
		Encryption	Decryption
PCA	512 bits (Minor)	35.0559%	42.0102%
	640 bits (Minor)	29.3937%	36.3994%
	768 bits (Minor)	20.8030%	27.4940%
	896 bits (Minor)	9.7420%	15.9243%
	1024 bits (Minor)	-0.5810%	0.1377%

From Table III, it shows that *PCA can perform faster in encryption and decryption than RSA about 35.0559% and 42.0102%, with 512 bits length Minor key, for the most efficient.* The reason why the process of PCA is faster than RSA is the speed of algorithm bases on key length. In addition, the change of time relating to key length is the relation in Exponential function; hence, using a shorter key length takes a shorter time.

Moreover, from the theoretical calculation of *Brute force attack*, we will see that the time for *Brute force attack* on PCA is longer than RSA for 1.497×10^{126} years. Hence, from the result of theoretical calculation on *Brute force attack*, PCA can provide a security against *Brute force attack* better than RSA.

In addition, the using of parallel-key, *PCA can be performed on a computer with Multiprocessor and applied to some modes of operations with parallelism* such as Cipher Block Chaining (CBC) mode and Interleaved Cipher Block Chaining (ICBC) mode *for higher speed and security.*

VI. Conclusion and Further Work

This paper has proposed a new cryptographic algorithm, so called "Parallel-key Cryptographic Algorithm (PCA)", based on Asymmetric-key cryptography. By using parallel-key (a couple of key pairs), instead of using a single key pair, the algorithm is strengthened against the problem of *Factorization Attack* and *Brute force attack* which is main types of attacks on asymmetric-key cryptography. As a result, PCA can encrypts and decrypts message faster than RSA (which is one of the well known algorithm) about 53.98% and 72.44% respectively. In our simulation of Brute force attack on both algorithms (PCA and RSA), it shows that the time used by *Brute force attack* on PCA is longer than RSA for 1.497×10^{126} years. Moreover, this algorithm is more flexible of sending or transferring data through communication networks or insecure channels without any problems of key agreements and limitation of the number of key which need to be maintained in group.

Acknowledgment

First, thanks to Mr.Robert Elz and Dr.Nitida Elz, for important comments and suggestions. Thanks for a support from National Electronics and Computer Technology Center (NECTEC) and National Science and Technology Development Agency (NSTDA) under Junior Science Talent Project (JSTP) project.

References

[1] Atul Kahate, *CRYPTOGRAHY and NETWORK SECURITY*, Tata McGraw-Hill Publishing Company Limited, 2003

[2] Behrouz A. Forouzan, *Cryptography and Network Security*, McGraw-Hill International Edition, 2008

[3] Schneier, Bruce, *Applied Cryptography* Second Edition: protocols, algorithms, and source code in C, John Wiley & Sons, Inc., 1996

[4] David Pointcheval, Jacques Stern, "*Security Proofs for Signature Schemes*", E'cole Normale Supe'rieure Laboratoire d'informatique, 45, rue d'Ulm, 75230 Paris Cedex 05

[5] Benoˆıt Libert, Jean-Jacques Quisquater, and Moti Yung, "Parallel Key-Insulated Public Key Encryption Without Random Oracles", UCL, Micorelectronics Laboratory, Crypto Group (Belgium) and RSA Labs and Columbia University (USA)

[6] Ying Wang, Chunyan Han and Yuanyi Liu, "A Parallel Encryption Algorithm for Color Images Based on Lorenz Chaotic Sequences", *IEEE transaction on information Theory*, 1-4244-0332-4/06, pp.9744-9747,2006

[7] Yun Chen, Xin Chen, YiMu,"A Parallel Key Generation Algorithm for Efficient Diffie-Hellman Key Agreement", *IEEE Transactions on information Theory*, 1-4244-0605-6/06, pp.1393– 1395, 2006

[8] Praveen Dongara and T. N. Vijaykumar, "Accelerating Private-Key Cryptography via Multithreading on Symmetric Multiprocessors", *Proceeding of IEEE Int'l Symp. Performance Analysis of Systems and Software (ISPASS 03)*, IEEE Press, 2003, pp. 58-69.

[9] R. M. Dansereau, S. Jin, R. A. Goubran, "Reducing Packet Loss in CBC Secured VoIP using Interleaved Encryption", *Proceeding of IEEE CCECE/CCGEI*, Ottawa, May, 1-4244-0038-4 , 2006

[10] Dan Boneh and Ramarathnam Venkatesan, *Breaking RSA may be easier than factoring*, Advances in Cryptology — EUROCRYPT '98 (Kaisa Nyberg, ed.), LNCS, no. 1403, IACR, Springer, May 1998, pp. 59–71.

[11] Boneh, D., "Twenty Years of Attacks on the RSA Cryptosystem", Notice of the AMS, 46: 2003-2013

[12] Boneh, D.; Durfee, G., "Cryptanalysis of RSA with private key d less than $N^{0.292}$", *IEEE Transactions on Information Theory*, Volume 46, Issue 4, Jul 2000 pp.1339 - 1349

[13] Andrej Dujella, "Cryptanalysis A variant of Wiener's attack on RSA with small secret exponent", *ACM Communications in Computer Algebra*, Volume 42 , Issue 1-2 (March/June 2008), pp. 50-51, 2008