Cryptographic Computation Using ElGamal Algorithm in 32-bit Computing System

Nurul 'Atiqah Rosly, Mohd Zafran Abdul Aziz, Habibah Hashim,Syed Farid Syed Adnan,Mohd Anuar Mat Isa
Faculty of Electrical Engineering,
Universiti Teknologi MARA,
40450 Shah Alam, Malaysia
syed_farid@salam.uitm.edu.my

***Abstract*— Smart Computing provides ease of use in term of mobility in the low power computing devices such as smartphone, tablet and experimental board (e.g. RaspberryPi). Low power constraint and low computation capability require a lightweight security system and networking protocol. This paper presents an analysis of ElGamal encryption system for securing data communication through the computer. The objectives of this paper are to study ElGamal algorithm in 32-bit integer computation and to implement data encryption using the ElGamal algorithm using 32-bit integers. ElGamal is a continuation of Diffie-Hellman key exchange algorithm. However, ElGamal encryption system uses asymmetric key encryption and decryption. ElGamal uses a public key to encrypt and private a key to decrypt messages where private key and public key are different. An experiment was conducted to evaluate the maximum number of integers that can be computed in 32 bit computer system using standard 32 bit GCC compiler in Debian 6. Furthermore, an exploration of computing capabilities and performance measurement of the ElGamal algorithm using C language is presented in this paper.**

Keywords-component; ElGamal, Diffie-Hellman, 32-bits integers, DHKE, key exchange protocol, asymmetric, GMP, cryptography, security, trust, privacy, encryption, decryption, public key, precision computation, greater common divisor (GCD), Extended Euclidean.

I. INTRODUCTION

A. Overview

The work proposed in this paper aims to improve the overall security, trust and privacy for implementation in boot loader or firmware of embedded devices [2]. This paper focuses on using the Diffie Hellman and ElGamal algorithm since the Key Exchange used is assymmetric as such is more secure than using symmetric key algorithm. ElGamal algorithm in 32-bit computation was studied to deterrnine on how the data have been secured during the encryption and decryption using the ElGamal algorithm. Other than that, the objective of this paper is to implement data encryption using the ElGamal algorithm with the 32-bit computation. As for now, previous research has been focused on finding out the maximum number that can be to computed using the ElGamal algorithm in 32-bit computation. In an addition to that, computer capabilities in computing numbers greater than 2^{32} was also investigated. Based on the experiments that have been done and data that have been collected, the maximum number that can be computed using ElGamal in 32-bit computation is limited to the number below 2^{32}.

B. Diffie-Hellman Key Exchange (DHKE)

"New Directions in Cryptography" is a paper by Diffie and Hellman that presented a secure key agreement protocol called DHKE that can be carried out over public communication channel [3]. Diffie-Hellman is the earliest practical implementation of cryptographic key exchange [4]. DHKE is a method that allows two entities to share a secret key over insecure communication channels, although they have no prior knowledge of each other [2].

C. ElGamal Encryption System

ElGamal public-key encryption is a continuation of Diffie-Hellman key exchange protocol. This system was defined by Taher Elgamal in 1984. There are three main components which are the key generated; the encryption and decryption algorithms. ElGamal is characterized as asymmetric cryptography and it is commonly known that the encryption and decryption speed is slower than in a symmetrical algorithm [5].

D. Confidentiality, Integrity and Authentication (CIA)

Confidentiality, Integrity and Authentication (CIA) are core principles of information security. These principles are ideally from Donn Parker in 2002 [6]. Basically from Donn Parker the elements of information are divided into six. The examples of the elements are authentication, confidentiality, and authentication. Confidentiality refers to the prevention of information disclosure to unauthorized system or individuals. The information that can be accessed is only limited to certain types of information. Integrity refers to a respectable of character. Integrity needs people to maintain and measure data consistency and accuracy over its entire life-cycle. In other words, the data cannot be unmodified, undetected and unauthorized. Authenticity is necessary to make sure the data, transaction, communication or documents are genuine. In Non-repudiation concept, the person that does the transaction need to responsible for the thing that have been done. Let says, A is using B's account to a transaction. Although B denying that it is not being done by him but B will responsible for the action that have been done by A [7].

E. *Security, Trust and Privacy*

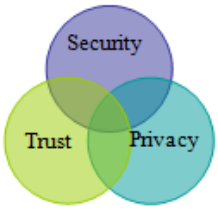

FIGURE 1. THREE MAIN COMPONENT IN INFORMATION SECURITY

There is no a specific meanings or standard for trust and privacy. Fig. 1 shows the relationship between trust, privacy and security. Basically, trust is an acceptance of any vulnerability either it is a positive expectation of behavior or intentions. [8] Definition of security is to maintain the confidentiality, the availability of information and integrity [9].

II. METHODOLOGY AND ANALYSIS

First, Diffie–Hellman key distribution scheme is reviewed [1]. The secret is made by raising the prime number a to an exponent which can range from 2 until 32. In this case, since these experiments are testing until 32-bits integer so that the maximum numbers that are used are until 32. When the prime number p had a as a primitive root in the Eq. (1) shows the range of values generated,

$$a \bmod p, a^2 \bmod p, \ldots, a^{32} \bmod p \tag{1}$$

In this algorithm, there are two parties that wish to exchange a key indicated as A and B. Random number X_a for user A will be selected to calculate the public key Y_a in Eq. (2).

$$Y_a \equiv a^{Xa} \bmod p \tag{2}$$

Similarly, user B will select random number X_b independently for Eq. (3) to calculate the public key Y_b.:

$$Y_b \equiv a^{Xb} \bmod p \tag{3}$$

The X values will be kept as private for each side while the Y values are available publicly to the other side. The calculation for user A using Eq. (4), to calculate the shared secret is as follows:

$$K \equiv (Y_b)^{Xa} \bmod p \tag{4}$$

and calculation of user shared secret for B in Eq. (5)as follows:

$$K \equiv (Y_a)^{Xb} \bmod p \tag{5}$$

This will produce the same result for both calculations, Eq.(6), as proven below:

$$\begin{aligned} K &\equiv (Y_b)^{Xa} \bmod p \\ &\equiv (a^{Xb} \bmod p)^{Xa} \bmod p \\ &\equiv (a^{Xb})^{Xa} \bmod p \\ &\equiv (a^{Xa})^{Xb} \bmod p \\ &\equiv (a^{Xa} \bmod p)^{Xb} \bmod p \\ &\equiv (Y_a)^{Xb} \bmod p \end{aligned} \tag{6}$$

Both of the users have now exchanged with each other their secret key. The opponents only know the values of p, a, Y_a and Y_b, since the value X_a and X_b are private. To determine the key, the opponents have to solve a discrete logarithms problem. Discrete logarithm problem is very difficult to solve compared to exponential modulo a prime that is easier to calculate, that is a fact that security are based on Diffie-Hellman key exchange algorithm.To know the value of X_a and X_b ,the attacker has to solve the discrete logarithm as in Eq. (7).

$$a \equiv b^x \bmod p \tag{7}$$

Secret values can be established to be used in exchanging data in public network by using the Diffie-Hellman key exchange algorithm [10].

A common method for calculating modular division is Extended Euclidean Algorithm [11].This algorithm is used to find out the greatest common divisor of two integers. In this experiment, gcd and Extended Euclidean Algorithm were used to find inverse function to decrypt C_1 ciphertext.

ElGamal algorithm is a continuation from a Diffie-Hellman Algorithm and Extended Euclidean Algorithm. The coding that has been created is a combination of Diffie-Hellman Algorithm and Extended Euclidean Algorithm.

First, choose a prime number within the range provided by the 32-bit integer known as p .Then choose a value for g, withint the range of the 32-bit integer but the number must be lower than value of p. For example, there are two users named Along and Busu.

Along has to choose any random integer a such that $0 < a < p$. Then Along will compute Eq.(8) .

$$\mathrm{A} \equiv g^a \bmod p \tag{8}$$

The public key for Along is A, g and p. While her private key is a. Along will receive an encrypted short message from Busu. The encrypted message that has been sent by Busu include the random integer B such that $0 < b < p$. Then he will perform a similar computation as Along by Eq. (9).

$$\mathrm{B} \equiv g^b \bmod p \tag{9}$$

After that Busu will send his encrypted message to Along. The message that has been send to Along will decrypted by computing Eq. (10).

$$m \equiv (C_1^a)^{-1} \cdot C_2 \pmod{p} \quad (10)$$

Value of C_1 and C_2 are as in Eq (11) :

$$C_1 = A,$$
$$C_2 = m \cdot B^a \pmod{p} \quad (11)$$

Even if there are any attempts on the ciphertext , without knowing a , it will not be possible to solve Eq. (10).

III. RESULT AND DISCUSSION

To discover the performance of the 32-bit ElGamal Algorithm, an experiment has been conducted. There are three types of time performance using time function in Linux. The three time function in Linux is real %e, user %u, sys %s. %e is for elapsed real time. %u is for the total number of CPU-seconds that the process spent in user mode and %s for total number of CPU-second that the process spent in kernel mode. Data have been collected for all three time functions. Data for elapsed real time are shown in Table I.

TABLE I: PERFORMANCE IN ELAPSED REAL TIME

Input					Performance (seconds)			
p	**g**	**m**	**a**	**b**	**10,000**	**25,000**	**50,000**	**100,000**
2	1	2	5	7	0.007	0.012	0.021	0.038
3	2	3	5	7	0.007	0.012	0.021	0.038
5	3	5	5	7	0.007	0.012	0.021	0.038
7	5	7	5	7	0.007	0.013	0.021	0.040
11	7	11	5	7	0.007	0.012	0.021	0.038
13	7	13	5	7	0.007	0.012	0.021	0.039
17	7	17	5	7	0.007	0.012	0.022	0.039
19	7	19	5	7	0.007	0.012	0.022	0.040
23	7	23	5	7	0.007	0.012	0.022	0.040

Table I shows the sample data that have been taken. For input value, $g < p$ and $a > b$. Performance measurement is collected for 10000, 25000, 50000 and 100000 times looping. Fig. 2 shows the data performance in real time. In Fig. 2, it is shown that all data performance in elapsed of real time are similar with each other although the input value and looping time are different. This is because in real time all other elapsed time slices that are used by other processes are included. So the times are not accurate to represent the specific time performance for data that have been executed.

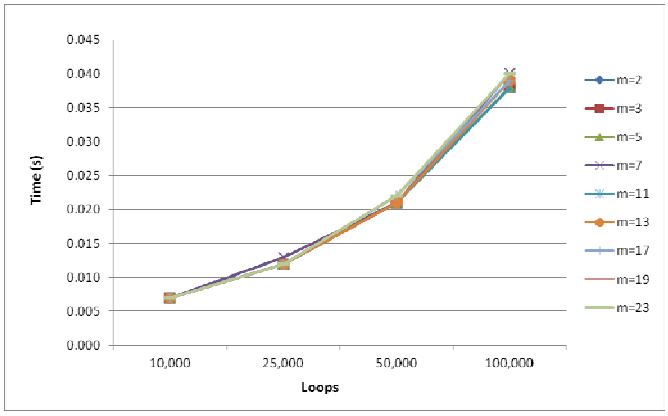

FIGURE 2. DATA PERFORMANCE IN REAL TIME

TABLE II: PERFORMANCE IN USER MODE

Input					Performance (seconds)			
p	**g**	**m**	**a**	**b**	**10,000**	**25,000**	**50,000**	**100,000**
2	1	2	5	7	0.008	0.012	0.012	0.032
3	2	3	5	7	0.008	0.012	0.016	0.040
5	3	5	5	7	0.008	0.008	0.016	0.032
7	5	7	5	7	0.000	0.012	0.020	0.040
11	7	11	5	7	0.008	0.008	0.020	0.032
13	7	13	5	7	0.004	0.012	0.012	0.036
17	7	17	5	7	0.008	0.008	0.020	0.028
19	7	19	5	7	0.008	0.004	0.016	0.032
23	7	23	5	7	0.004	0.004	0.016	0.036

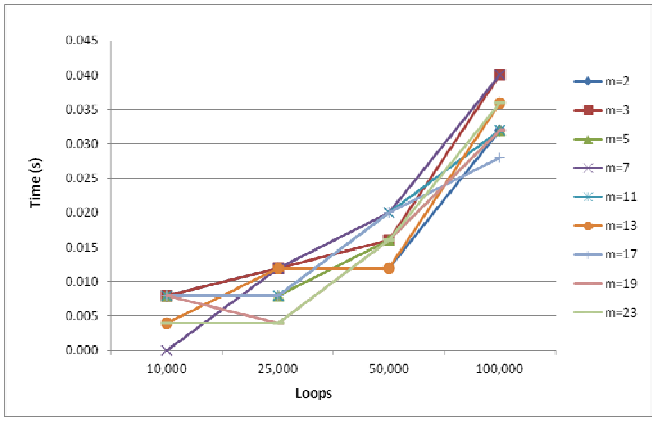

FIGURE 3. PERFORMANCE IN USER MODE

Data collected in Table II is for performance of data in user mode. The graph plotted by the data is shown in Fig. 3. In Fig. 3, data performances in user mode display the same result for 50,000 times looping. On other looping times, there are slightly different results. In user mode, the times taken are times in user mode only. The times for other processes are blocked from being counted within this process. So the user time is more accurate than in real time mode.

TABLE III PERFORMANCE IN KERNEL MODE

Input					Performance (seconds)			
p	g	m	a	b	10,000	25,000	50,000	100,000
2	1	2	5	7	0.007	0.012	0.021	0.038
3	2	3	5	7	0.007	0.012	0.021	0.038
5	3	5	5	7	0.007	0.012	0.021	0.038
7	5	7	5	7	0.007	0.013	0.021	0.040
11	7	11	5	7	0.007	0.012	0.021	0.038
13	7	13	5	7	0.007	0.012	0.021	0.039
17	7	17	5	7	0.007	0.012	0.022	0.039
19	7	19	5	7	0.007	0.012	0.022	0.040
23	7	23	5	7	0.007	0.012	0.022	0.040

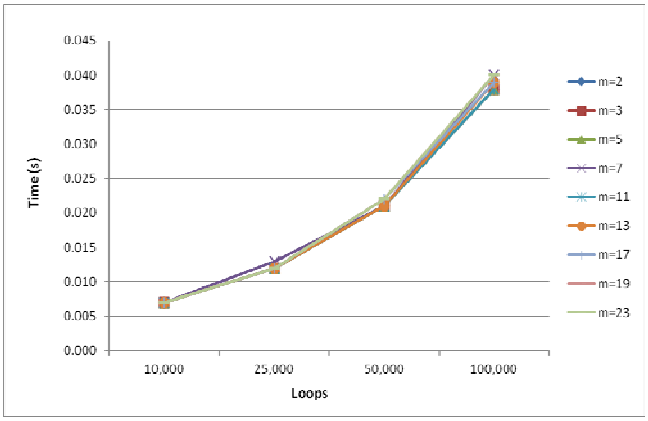

FIGURE 4. PERFORMANCE IN KERNEL MODE

Fig. 4 shows the performance in kernel mode based on data in Table III. Data collected shows that performance data are different with each other. Time in kernel mode is counted only in CPU mode which is within the kernel process. Therefore, the combination time in kernel mode and user mode, relative execution are the accurate time counted to refer as performance compared to real time mode. As shown in fig. 5 based on data in table IV, the time shows how much CPU time given to execute this algorithm based on time slice.

TABLE IV RELATIVE EXECUTION

Input					Performance (seconds)			
p	g	m	a	b	10,000	25,000	50,000	100,000
2	1	2	5	7	0.015	0.024	0.033	0.07
3	2	3	5	7	0.015	0.024	0.037	0.078
5	3	5	5	7	0.015	0.02	0.037	0.07
7	5	7	5	7	0.007	0.025	0.041	0.08
11	7	11	5	7	0.015	0.02	0.041	0.07
13	7	13	5	7	0.011	0.024	0.033	0.075
17	7	17	5	7	0.015	0.02	0.042	0.067
19	7	19	5	7	0.015	0.016	0.038	0.072
23	7	23	5	7	0.011	0.016	0.038	0.076

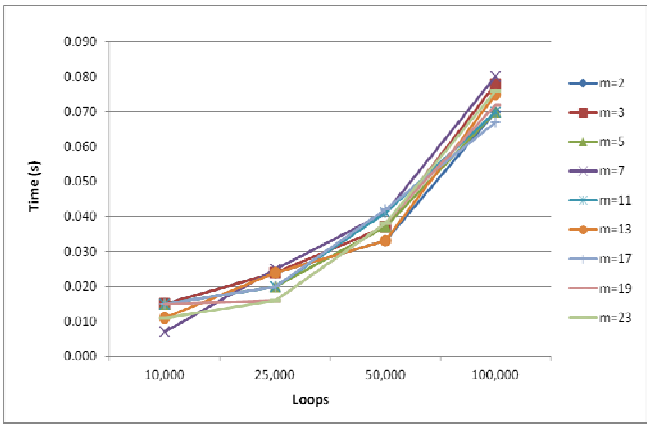

FIGURE 5. RELATIVE EXECUTION

IV. RESULT AND ANALYSIS

Data performances have been collected for three different times. In conclusion data in real time are not suitable as a reference to be implemented because in real time mode this project is not being implemented on the networking process. Real time mode output is including other processing time, so the output time is not accurate. The actual times for the specific process are counted by using a combination of time in user mode and system mode.

There are a few algorithms can be used as a technique to secure data through computer such as Diffie-Hellman key exchange and ElGamal algorithms. ElGamal is an improvement from Diffie-Hellman Key Exchange Protocol. In order to prove the ElGamal capabilities to secure data through a computer, an experiment have been performed and the result shows that an integer number equal or less than 32-bit is not capable to secure data through 32-bit computer system. For an improvement, to implement numbers bigger than 32-bit integer needs to use 64-bit computer. However, the usage of standard 32-bit or 64-bit GCC integer is not enough to meet security requirements for current cryptographic key strength which is at least 1024 bits. For that reason, implementation based on modern ElGamal needs to use 2^{1024} bits computing integer that is using GMP bignum library to replace standard GCC integer number. GMP library allowed for C and C++ codes to computes integer numbers greater than 32-bits for 32-bit or 64-bit computer.

ACKNOWLEDGMENT

The authors would like to acknowledge the Ministry Of Science Technology & Innovation, Malaysia for supporting the study by providing the research grant under "100-RMI/SF 16/6/2 (18/2012)".

REFERENCES

[1] T. ElGamal, "A Public Key Cryptosystem and a Signature Scheme Based on Discrete Logarithm", IEEE Transaction on Information Theory , volume IT-31, No.4, July 1985

[2] M. A. M. Isa, N. N. Mohamad, H. Hashim, S. F. S. Adnan, J.A. Manan and R. Mahmod, "A Lightweight and Secure TFTP Protocol for Smart Enviroment" in 2012 International Symposium on Computer Application and Industrial Electronics (ISCAIE 2012), Kota Kinabalu Malaysia, December 2012.

[3] J.F.Raymond and A. Stiglic, "Security Issues in the Diffie-Hellman Key Agreement Protocol", IEEE Transactions on Information Theory, 2000.

[4] Y. Guang-Ming, C. Jin-Ming, L. Ya-Feng and M. DA-Ming, "An efficient improved group key agreement protocol based on Diffie-Hellman key exchange," *Advanced Computer Control (ICACC), 2010 2nd International Conference on* , Vol.2, no., pp.303,306, March 2010.

[5] H. Chen, X. Shen,Y. Lv and J. Lin, "An Improvement of ElGamal Digital Signature Scheme," Information Engineering and Computer Science, 2009. ICIECS 2009. International Conference on , vol., no., pp.1,5, Dec 2009.

[6] Y. S. Feruza and T.Kim "IT Security Review: Privacy, Protection, Access Control", International Journal of Multimedia and Ubiquitous Engineering ,Vol. 2, No. 2, April, 2007

[7] "InformationSecurity",http://en.wikipedia.org/wiki/Information_security, Mac 2013.

[8] D. Rousseau, S. Sitkin, R. Burt and C. Camerer , " Not so Different after All: a Cross-discipline View of Trust", Academy of Management Review, 23 (3): 393-404, 1998.

[9] ISO (2005) 27001 : Information Security Management-Specification With Guidance for Use.

[10] N. Li, "Research on Diffie-Hellman Key Exchange Protocol" , 2010 2nd International Conference on Computer Engineering and Technology, 2010.

[11] D.E. Knuth, "The Art of Computing Programming", Vol. 2, Seminumerical Algorithms, third Ed., Addison-Wesley, Reading, MA, 1998.