A Large Scalar Multiplication Algorithm using Modified Pell Numbers for Key Generation

Fudailah Duemong[1] and Ladda Preechaveerakul[2]

ABSTRACT: Modern Cryptographic methods consist of two parts, a key and an algorithm, which are used to encrypt and decrypt data. The key is an essential part that works with the algorithm. The security of encryption schemes depends on the key size (key length) and the longer the key, the better the security it provides. Applying an elliptic curve for key agreement provides a high-performance architecture and high security. The main process for calculating key points in Elliptic Curve Cryptography (ECC) is called scalar multiplication, which relates to point addition and point doubling. An efficient algorithm, proposed as the Large Scalar Multiplication Algorithm using Modified Pell Numbers (LSMA-MPN), was introduced to speed up the calculation of points on elliptic curves during large scalar multiplications. This system also reduced computation time by applying Modified Pell numbers in a 2×2 matrix representation. The experimental results show that computation time was reduced by approximately 67% in comparison with the computation time required by a general algorithm.

Keywords: Elliptic Curve Cryptography, Scalar Multiplication, Modified Pell Numbers, Matrix Representation

DOI: 10.37936/ecti-cit.2021152.227427

Article history: received December 1, 2019; revised April 19, 2020; accepted June 23, 2020; available online April 29, 2021

1. INTRODUCTION

Valuable information is routinely transferred over the Internet. However, most channels that we currently use to transfer information are not secure [1,2]. The emerging communication network known as the Internet of Things (IoT) [3] makes daily life easier. Smart devices are connected to each other via wireless networks. These connected devices generate a tremendous amount of information held in Internet storage such as cloud services. The security is sues are considerable. Although most cloud provide offer security processes, cloud users still have concerns about important information or sensitive data [4], such as personal identification numbers, financial, health, and educational information. One of the techniques used to secure data or information is cryptography [5, 6]. Cryptography [5, 7] is the art and science of encoding messages to make them non-readable. Basic cryptographic terms used in this research are plaintext, which is a message in an understandable or readable form; ciphertext, which is the same message in an unreadable form; encryption, which is the process of transforming plaintext into ciphertext; and decryption, which is the process of transforming ciphertext back into plaintext.

Many researchers have proposed cryptographic algorithms to encrypt and decrypt data. These cryptographic processes consist of two elements: a key and an algorithm. Usually, the algorithm used for encryption and decryption is available to everybody. However, to secure encrypted data, a key is necessary. Symmetric keys are simpler and take less time because the same key is used for encryption and decryption of data. The disadvantage of symmetric keys is that all users (senders and receivers) must share the key to encrypt data before they can decrypt it. On the other hand, when using asymmetric keys, the sender uses one key to encrypt the data and the receiver uses another key to decrypt it. Asymmetric key algorithms are used in communication channels, especially over the Internet.

[1,2]The authors are with the Computer Science Program, Faculty of Science, Prince of Songkla University, Thailand., E-mail: 5910230010@psu.ac.th and ladda.p@psu.ac.th

[1]The scholarship of the first author is supported by the Strategic Scholarships Fellowships Frontier Research Networks (Specific for Southern Region), from the Office of the Higher Education Commission, Ministry of Higher Education, Science, Research and Innovation.

The most commonly used algorithms are RSA and ECC (Elliptic Curve Cryptography). Each cryptographic algorithm provides different strength of security or security level, depending on the algorithm and the key size used. The National Institute of Standards and Technology (NIST) recommends security levels in bits [8] which estimate the computational operations to find a solution. The main problem of conventional cryptographic algorithms is that the key size has to be sufficiently large in order to meet the high security level. For example, at a 128-bit security level, the ECC-based algorithms use a 256-bit key, whereas RSA algorithm uses a 3072-bit key. This shows that ECC-based algorithms use smaller-sized keys than RSA [9] and as a result, ECC-based algorithms have gradually replaced RSA [10]. Although much research into data security has focused on the generation of small-sized keys, the strength of the cryptographic process also depends on the algorithm used to generate the key. ECC is now applied for key eneration, and the main process is scalar multiplication. The efficiency of the algorithm depends on the cost of computing scalar multiplication. However, if the scalar number is large, the number of arithmetic operations, especially multiplications, is increased. Therefore, this research proposes a new cryptographic key generation algorithm by applying Modified Pell Numbers that reduces the time needed for large scalar multiplication but securely exchanges keys over an unsecure channel and offers the same level of security as longer keys.

This paper gives the background knowledge and related work in Section 2 and 3, respectively. A new method of calculating large scalar multiplication for Elliptic Curve Cryptography is proposed in Section 4. We discuss the efficiency analysis of the proposed method in Section 5 before presenting the conclusion in Section 6.

2. MATHEMATICAL BACKGROUND

In this section, some basic conceptual terms are presented.

2.1 Elliptic Curve Scalar Multiplication

Elliptic Curve Cryptography is a widely used approach, since it provides a high level of security with a smaller key size. To generate a key, key points are calculated along an elliptic curve using scalar multiplication. Definition 1 describes a point operation on an elliptic curve. All operations on the elliptic curve are performed in the order of the prime field [2, 11, 12], as shown in Fig.1 and Definition 2.

Definition 1: Let E be an elliptic curve and Q be a scalar multiplication on E. Then Q is derived by adding point P on E by k times and is defined as

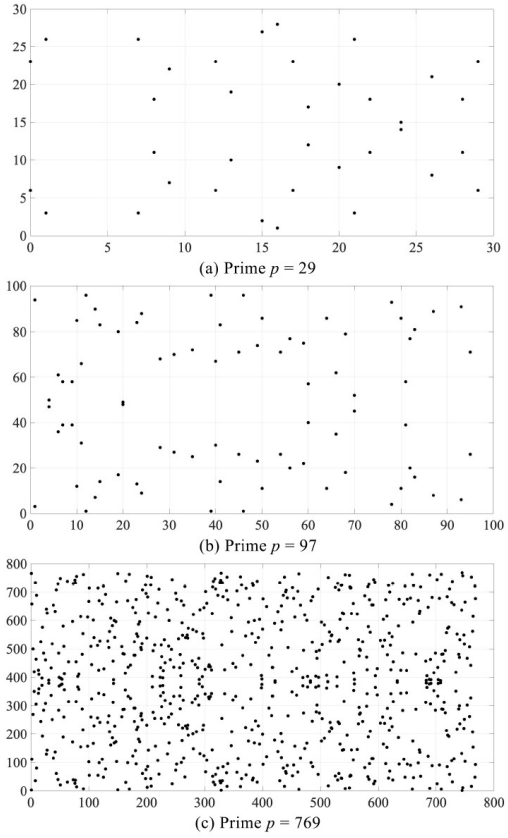

Fig. 1: *Rational points of* $Ep : y^2 = x^3 + x + 7$ *with (a) prime* $p = 29$*, (b) prime* $p = 97$*, and (c) prime* $p = 769$*.*

$$Q = kP = \sum_{i=1}^{k} P \tag{1}$$

where k is a positive integer and P is a point on the elliptic curve.

Definition 2: Let E_p be an elliptic curve over a finite field F and F_p be an integer modulo p from a prime field. Then E_p is defined as

$$E_p : \{(x, y) \in F_p | y^2 = x^3 + ax + b\} \ mod \ p \tag{2}$$

where p is a prime number, and the discriminant is $4a^3+27b^2 \neq 0$ to ensure that the curve is nonsingular.

Basically, scalar multiplication on an elliptic curve consists of 2 arithmetic operations: point addition and point doubling. Definition 3 and Example 1 refer to point addition while Definition 4 and Example 2 refer to point doubling.

Definition 3: Let point addition be adding two points from different co-ordinates on an elliptic curve. Let $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ be two different co-ordinates and $P_3(x_3, y_3) = Q \in E(F_p)$. The point addition operation is defined as

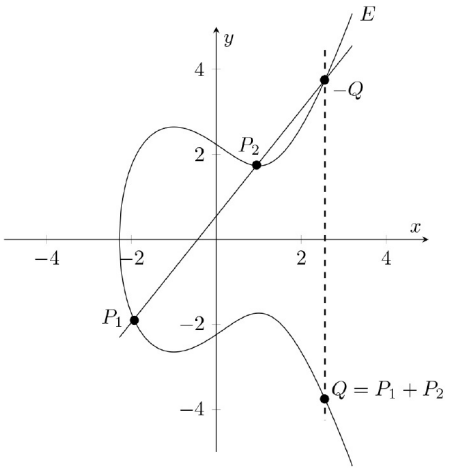

Fig.2: *Point addition on the elliptic curve* $y^2 = x^3 - 3x + 5$.

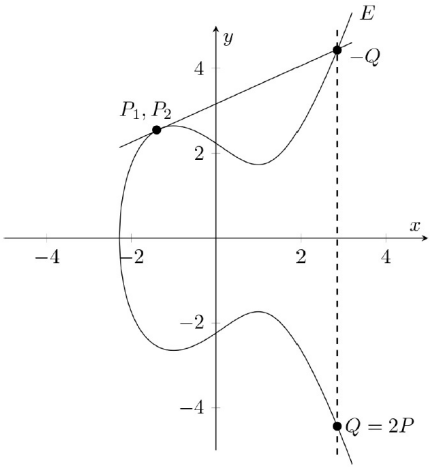

Fig.3: *Point doubling on the elliptic curve* $y^2 = x^3 - 3x + 5$.

$$Q = P_3(x_3, y_3) = P_1(x_1, y_1) + P_2(x_2, y_2) \quad (3)$$

Assuming that $P_1 \neq P_2$ and that $P_1, P_2 \notin \infty$, the line through P_1 and P_2 is depicted in Fig. 2, where

$$x_3 = \lambda^2 - x_1 - x_2 \ mod \ p \quad (4)$$

$$y_3 = \lambda(x_1 - x_3) \ mod \ p \quad (5)$$

$$\lambda = \frac{y_2 - y_1}{x_1 - x_2} \ mod \ p \quad (6)$$

Example 1: Let an elliptic curve be $E_{17} : y^2 = x^3 + x + 13 \ mod \ 17$. Choose $P_1(x_1, y_1) = (1, 7)$ and $P_2(x_2, y_2) = (3, 14)$. From Definition 3, the point addition $P_3(x_3, y_3)$ is

$$\begin{aligned}
\lambda &= \frac{14 - 7}{3 - 1} \ mpd \ 17 \\
&= \left\{ \frac{7 \ mod \ 17}{2 \ mod \ 17} \ mod \ 17 \right\} \\
&= \{(7 \ mod \ 17) \times (2^{-1} \ mod \ 17)\} \ mod \ 17 \\
&= \{(7) \times (9)\} \ mod \ 17 \\
&= 63 \ mod \ 17 \\
&= 12
\end{aligned}$$

$$\begin{aligned}
x_3 &= (12)^2 - 1 - 3 \ mpd \ 17 \\
&= 140 \ mod \ 17 \\
&= 4
\end{aligned}$$

$$\begin{aligned}
y_3 &= 12(1 - 4) - 7 \ mpd \ 17 \\
&= (-43) \ mod \ 17 \\
&= 8
\end{aligned}$$

Therefore, on the elliptic curve E_{17}, the result of $Q = P_1 + P_2$, in other words $P_3(x_3, y_3)$, is (4, 8).

Definition 4: Let point doubling be adding the same points from the same co-ordinates on an elliptic curve. Let $P_1(x_1, y_1)$ and $P_2(x_2, y_2)$ be 2 points from co-ordinates such that $P_1(x_1, y_1) = P_2(x_2, y_2)$, $P_3 = 2P_1 = Q \in E(F_p)$. The point addition operation is defined by

$$Q = P_3(x_3, y_3) = p_1(x_1, y_1) + P_1(x_1, y_1) = 2P_1 \quad (7)$$

Assuming that $P_1 = P_2$, the line through P_1 and curve E is depicted in Fig. 3, where

$$x_3 = \lambda^2 - 2x_1 \ mod \ p \quad (8)$$

$$y_3 = \lambda(x_1 - x_3) - y_1 \ mod \ p \quad (9)$$

$$\lambda = \frac{3(x_1)^2 + a}{2y_1} \ mod \ p \quad (10)$$

Example 2: Let an elliptic curve be $E_{17} : y^2 = x^3 + x + 13 \ mod \ 17$. Choose $P_1(x_1, y_1) = (1, 7)$. From Definition 4, the point doubling to derive $P_3(x_3, y_3)$ is performed as follows:

$$\begin{aligned}\lambda &= \frac{3(1)^2+1}{2(7)}\ mpd\ 17\\ &= \left\{\frac{4\ mod\ 17}{14\ mod\ 17}\ mod\ 17\right\}\\ &= \{(4\ mod\ 17)\times(14^{-1}\ mod\ 17)\}\ mod\ 17\\ &= \{(4)\times(11)\}\ mod\ 17\\ &= 44\ mod\ 17\\ &= 10\end{aligned}$$

$$\begin{aligned}x_3 &= (10)^2-2(1)\ mpd\ 17\\ &= 98\ mod\ 17\\ &= 13\end{aligned}$$

$$\begin{aligned}y_3 &= 10(1-13)-7\ mpd\ 17\\ &= 9\ mod\ 17\\ &= 9\end{aligned}$$

Therefore, on the elliptic curve E_{17}, the result of $Q = 2P_1$, in other words $P_3(x_3, y_3)$, is (13, 9) .

2.2 Modified Pell Numbers

The Fibonacci sequence [13] is a set of numbers in which each number is the sum of the two predecessors. Pell numbers [14] are based on Fibonacci numbers except the number in a Pell sequence is greater than the number in a Fibonacci sequence at the same index. A Pell and Modified Pell sequence [15] are defined in Definition 5 and 6, respectively. Definition 7 describes how the ratio of two consecutive Pell numbers gets closer to the silver ratio as the i^{th} Pell number approaches infinity.

Definition 5: [15] Let the sequence of Pell numbers be

$$n_1, n_2, n_3, n_4, n_5, \ldots, n_i, \ldots \tag{11}$$

where n_i is the number at the i^{th} index in the Pell sequence. The i^{th} Pell number is defined as follows:

$$n_i = \begin{cases} 0 & \text{if } i = 1;\\ 1 & \text{if } i = 2;\\ 2_{n_{i-1}+n_{i-2}} & \text{otherwise.}\end{cases} \tag{12}$$

Therefore, the recurrence relation of the Pell sequence can be shown as

0, 1, 2, 5, 12, 29, 70, 169, 408, 985, 2378,...

Definition 6: [15] Let the sequence of Modified Pell numbers be

$$m_1, m_2, m_3, m_4, m_5, \ldots, m_i, \ldots \tag{13}$$

where m_i is the number at the i^{th} index in the Modified Pell sequence. The i^{th} Modified Pell numbers is defined as follows:

$$m_i = \begin{cases} 1 & \text{if } i = 1, 2;\\ 2_{m_{i-1}+m_{i-2}} & \text{otherwise.}\end{cases} \tag{14}$$

Therefore, the recurrence relation of the Modified Pell sequence can be shown as

1, 1, 3, 7, 17, 41, 99, 239, 577,...

Definition 7: [16] Let φ be the silver ratio that is denoted by the ratio of two consecutive Pell numbers in sequence. The silver ratio can be defined by the following rule:

$$\varphi = \lim_{i\to\infty}\frac{n_i}{n_{i-1}} = 1+\sqrt{2} \tag{15}$$

where $n_i, n_{i-1} \in Z^+$

3. RELATED WORK

Every encryption and decryption process consists of an algorithm and a key. Generally, the algorithm used for both processes is revealed to everyone, but the key makes the process of cryptography secure. The strength of cryptographic algorithms is normally based on the difficulty of knowing the pattern of the algorithm without additional information, and the key size. Consequently, efforts to make keys more secure have usually generated large keys. One way to solve this problem is ECC [17, 18]. Koblitz and Miller [19] introduced the ECC technique to reduce key size, basing their innovation on the concept of the Elliptic Curve Discrete Logarithm Problem (ECDLP). Compared with Rivest-Shamir-Adleman (RSA) [20], ECC offerred the same level of security using a smaller key size [21, 22]. An elliptic curve is applied for key agreement [23, 24] in both symmetric and asymmetric key algorithms.

Keys schemes generated from ECC use much fewer bits than those generated for cryptographic systems that are based on integer factorization and discrete logarithm problems. Many researchers have exploited this advantage of ECC and implemented it in cryptographic algorithms and various tasks of key agreement. The main process in ECC is a scalar multiplication (Q) defined in Definition 1. If k is large, the number of multiplication operations is increased. Therefore, current research has focused on reducing the number of these arithmetic operations. An elliptic curve used for scalar multiplication is defined over a finite field which could be binary or a prime field [25, 26]. The basic methods used for scalar multiplication over a binary field are the right-to-left binary (RLB) and the left-to-right binary (LRB) methods. There are three general methods for scalar multiplication over prime fields [27–29]: the double-and-add method, the NAF method, and the window method. The cost of scalar multiplication methods depends on the cost of arithmetic operations.

Some research has applied elliptic curves over binary fields. Susantio and Muchtadi-Alamsyah [27] separated plaintext into several blocks using a sim-

ple version of the Elliptic Curve Integrated Encryption Scheme (ECIES). Javeed, et al. [30] designed a method of providing new high-performance hardware for calculating scalar multiplication based on parallel modular multiplication operations implemented on Xilinx Virtex-6, Virtex-5, and Virtex-4 FPGA platforms. However, some research has applied elliptic curves over prime fields for large scalar multiplication. Panchbhai and Ghodeswar [31] explained the Verilog implementation by digitizing serial computations on elliptic curves to calculate point addition and point doubling for faster computation and more efficient algorithms. Phalakarn, et al. [28] extended the idea of optimal representation for a right-to-left parallel to calculate the scalar multiplication on an elliptic curve based on NAF and devised a mathematical model to reduce computation time by using parallel computing. Additionally, one of the interesting methods used for the ECC approach is Fibonacci numbers. Fibonacci numbers have been applied in both the process of algorithms and key generation to reduce the computation time. Raphael and Sundaram [32] designed a secured communication using Fibonacci numbers and Unicode symbols to generate encryption and decryption algorithms. Plaintext was converted into ciphertext by replacing one character with another based on Fibonacci numbers and then the ciphertext was converted to Unicode symbols. Mukherjee and Samanta [33] presented a new technique to encrypt data using Fibonacci numbers hidden in an image. Agarwal, et al. [34] then extended the two-phase encryption of Raphael and Sundaram [32] by converting plaintext using characters based on Fibonacci numbers and Unicode symbols and then encrypting the Unicode symbols into an image. Zhang and Tan [35] explained the use of Fibonacci numbers in scalar multiplication operations for key generation. They used the Zeckendorf and Pell representation to construct Binary Scalar Multiplication (BSM). Liu, et al. [36] then proposed new addition and doubling formulas, called Fibonacci-type Addition Chain (DFAC) algorithm, to reduce computation time. For the same reason, Duemong and Preechaveerakul [37] proposed a new approach of calculating large scalar multiplication that was based on Pell numbers.

As the decrement of computation time for large scalar multiplication is an important part of key generation, this paper introduces an efficient algorithm by applying Modified Pell numbers in a 2×2 matrix representation.

4. LARGE SCALAR MULTIPLICATION ALGORITHM USING MODIFIED PELL NUMBERS

Elliptic curve cryptography provides a high level of security with a smaller key size. Scalar multiplication on an elliptic curve is the main process for calculating key points. If the scalar k is large, the number of arithmetic operations, especially multiplications, is increased. This complexity increases computation time. To reduce the computation time, we propose a new algorithm called Large Scalar Multiplication Algorithm using Modified Pell Numbers ($LSMA-MPN$). $LSMA-MPN$ consists of two main processes: creating a table of points at the numbers in the Modified Pell sequence ($P4MP$ table), and calculating scalar multiplication on an elliptic curve.

Theorems 1 - 2 are presented next, then Example 3 explains how to find the index of a number in the Modified Pell sequence. Example 4 explains how to find the numbers in the sequence.

Theorem 1: Let m_i be the number at the index i in a Modified Pell sequence $(1, 1, 3, 7, 17, 41, 99, 239, \ldots)$ where $m, i \in Z^+$, and let δ be the normalization ratio of two consecutive Modified Pell numbers in the sequence. Then a relation of two consecutive Modified Pell numbers in the sequence is $1+\sqrt{2}$ denoted by

$$\lim_{i\to\infty} \frac{m_i}{m_{i-1}} = 1+\sqrt{2} \tag{16}$$

Proof:

Assume that δ is the normalization ratio of two consecutive Modified Pell numbers in the sequence, then

$$\delta = \lim_{i\to\infty} \frac{m_i}{m_i - 1}$$

where $i \in Z^+$.
From (14), for $i \geq 3$;

$$m_i = 2m_{i-1} + m_{i-2}$$

Thus, $$\frac{m_i}{m_{i-1}} = 2 + \frac{m_{i-2}}{m_{i-1}}$$

Suppose $$\delta_i = \frac{m_i}{m_{i-1}},$$

then $$\delta_i = 2 + \frac{m_{i-2}}{m_{i-1}},$$

when $i = 3; \delta_3 = 2 + \frac{m_1}{m_2} = 2 + \frac{1}{1} = 2$

$i = 4; \delta_4 = 2 + \frac{m_2}{m_3} = 2 + \frac{1}{3} = 2.3333$

$i = 5; \delta_5 = 2 + \frac{m_3}{m_4} = 2 + \frac{3}{7} = 2.4286$

$i = 6; \delta_6 = 2 + \frac{m_4}{m_5} = 2 + \frac{7}{17} = 2.4118$

$i = 7; \delta_7 = 2 + \frac{m_5}{m_6} = 2 + \frac{17}{41} = 2.4146$

$i = 8; \delta_8 = 2 + \frac{m_6}{m_7} = 2 + \frac{41}{99} = 2.4141$

$i = 9; \delta_9 = 2 + \frac{m_7}{m_8} = 2 + \frac{99}{239} = 2.4142$

As i approaches infinity; therefore, $\lim_{i\to\infty} = 1+\sqrt{2}$

From now on, we will define the index i in the Modified Pell sequence (m_i) as follows:

$$i = \lfloor 1 + (\mu \times \log(m_i) + \omega) \rfloor \tag{17}$$

where ω, μ, and δ are defined as

$$\omega = \log \sqrt{2} \times \mu,$$

$$\mu = \frac{1}{\log \delta},$$

$$\delta = 1 + \sqrt{2}$$

To reduce time calculating scalar multiplication on an elliptic curve, the ratio in Theorem 1 is applied to find the index of a number in the Modified Pell sequence using the floor function (the largest integer less than or equal to the given number) as shown in (17).

Example 3: To find the index i of the number 135 in the Modified Pell sequence by using (17), the operation is performed along these lines:

$$\begin{aligned} i &= \lfloor 1 + \left(\frac{1}{\log(1+\sqrt{2})} \times \log(135) + \log \sqrt{2} \times \frac{1}{\log(1+\sqrt{2})} \right) \rfloor \\ &= \lfloor 1 + (2.6125 \times 2.1303 + 0.1505 \times 2.3125) \rfloor \\ &= \lfloor 1 + 5.9587 \rfloor \\ &= \lfloor 6.9587 \rfloor \\ &= 6 \end{aligned}$$

In fact, via rounding, the closest number to 135 s 99 at the index 7 in the Modified Pell sequence (*see Table* 1; $m_7 = 99$). However, using (17) to reduce computation time, the closest number to 135 is 41 at the index 6 (*see Table* 1; $m_6 = 41$) and this index will be used to the last index of 135 in the Modified Pell Sequence.

Table 1: *Modified Pell number at index i.*

i	1	2	3	4	5	6	7	8	9
m_i	1	1	3	7	17	41	99	239	577

We then have applied a 2 × 2 matrix to speed up calculating the numbers of the Modified Pell sequence from index 1 (m_1) to index i (m_i) as shown in Theorem 2 and presented the time complexity in Lemma 1.

Theorem 2: Let m_i be the number at the index i in a Modified Pell sequence where the number of the Modified Pell sequence m_i where $i \geq 4$, is derived from a matrix M^i based on the following 2× 2 matrix:

$$M^i = NM^{i-1}; M^i = \begin{bmatrix} m_i & m_{i-1} \\ m_{i-1} & m_{i-2} \end{bmatrix}, \tag{18}$$

and a matrix N that can be defined by

$$N = \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}, \tag{19}$$

and for $i \geq 3$;

$$m_i = 2m_{i-1} + m_{i-2}$$

proof:

The following property of the initial matrix M^i at initial index $i = 4$ is

$$\begin{aligned} M^4 &= NM^3 \\ &= \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} m_3 & m_2 \\ m_2 & m_1 \end{bmatrix} \\ &= \begin{bmatrix} 2m_3 + m_2 & 2m_2 + m_1 \\ m_3 & m_3 \end{bmatrix} \\ &= \begin{bmatrix} m_4 & m_3 \\ m_3 & m_2 \end{bmatrix} \end{aligned}$$

Therefore, the number of the Modified Pell sequence m_i in the matrix M^i is true for $i = 4$.

Suppose the number of the Modified Pell sequence at index i (m_i) in the matrix M^i is true for $i = j$.

Then,

$$M^j = \begin{bmatrix} m_{i-1} & m_{i-2} \\ m_{i-2} & m_{i-3} \end{bmatrix} = NM^{j-1} \tag{20}$$

Now, we will show it is true for $i = j + 1$ by using (20)

$$\begin{aligned} M^{j+1} &= M.M^j \\ &= M(NM^{j-1}) \\ &= NM^j \\ &= \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} m_{i-1} & m_{i-2} \\ m_{i-2} & m_{i-3} \end{bmatrix} \\ &= \begin{bmatrix} 2m_{i-1} + m_{i-2} & 2m_{i-2} + m_{i-3} \\ m_{i-1} & m_{i-2} \end{bmatrix} \\ &= \begin{bmatrix} m_i & m_{i-1} \\ m_{i-1} & m_{i-2} \end{bmatrix} \end{aligned}$$

Thus it is true for $i = j + 1$. By the principle of mathematical induction, the number of the Modified Pell sequence of index i (m_i) in the matrix M^i is true for all $i \geq 4$.

Example 4: Find the Modified Pell number at the index 7 (m_7) in the sequence. Using Theorem 2, the number at the index 7 is

$$\begin{aligned} M^7 &= NM^{(7-1)} \\ &= \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} m_6 & m_5 \\ m_5 & m_4 \end{bmatrix} \end{aligned}$$

So, we can simply multiply matrix N with matrix M^6 to give the result.

$$M^7 = \begin{bmatrix} 99 & 41 \\ 41 & 17 \end{bmatrix} = \begin{bmatrix} m_7 & m_6 \\ m_6 & m_5 \end{bmatrix}$$

Therefore, the number of the Modified Pell sequence at the index 7 (m_7) in matrix M^7 is 99. The set of numbers in the sequence from index 1 to 7 is {1, 1, 3, 7, 17, 41, 99}.

Lemma 1: Let $T(n)$ be a function over the positive numbers defined by the recurrence that finds the numbers (m_i) in the matrix M^i from Theorem 2. Then, the time complexity of calculating the numbers in Modified Pell sequence is $O(\log_2 n)$, where n is the number of indices of the Modified Pell sequence.

Proof:

From Theorem 2, a matrix M^i based on 2×2 matrix is

$$\begin{bmatrix} m_i & m_{i-1} \\ m_{i-1} & m_{i-2} \end{bmatrix}$$

and a matrix N is

$$\begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix}$$

We use matrix multiplication for multiplying the matrix M to recursively compute M^{i+1} where $i \geq 4$, and $m,\ i \in Z^+$ expressed as

$$\begin{aligned} M^{i+1} &= NM^i \\ &= \begin{bmatrix} 2 & 1 \\ 1 & 0 \end{bmatrix} \begin{bmatrix} m_i & m_{i-1} \\ m_{i-1} & m_{i-2} \end{bmatrix} \\ &= \begin{bmatrix} 2m_i + m_{i-1} & 2m_{i-1} + m_{i-2} \\ m_1 & m_{i-1} \end{bmatrix} \end{aligned}$$

and i represents the i^{th} power of the matrix M as $i = 1$;

$$\begin{aligned} M^1 &= M^{1/2} \cdot M^{1/2} \\ &= M^1 \end{aligned}$$

$i = 2$;

$$\begin{aligned} M^2 &= M^{2/2} \cdot M^{2/2} \\ &= M^1 \cdot M^1 \end{aligned}$$

$i = 3$;

$$\begin{aligned} M^3 &= M^{3/2} \cdot M^{3/2} \\ &= M^{6/2} \\ &= M^3 \\ &= M^1 \cdot M^1 \cdot M^1 \end{aligned}$$

and for any i;

$$M^i = \prod_1^i M^1 \tag{21}$$

Algorithm 1 Creating table of points at the numbers in Modified Pell sequence ($P4MP$ table)

Input: Point $P_0(x, y)$, scalar k, last index of k (li)
Output: $P4MP(m, P, t)$

$P4MP(m_1, P_1, t_1) \leftarrow (1, P_0, 0)$
$P4MP(m_2, P_2, t_2) \leftarrow (1, P_0, 0)$
$i \leftarrow 2$
while $(i \leq li)$ **do**
  $i \leftarrow i + 1$
  $P4MP(m_i, P_i, t_i) \leftarrow (m_i, 2 \cdot P_{i-1} + P_{i-2}, 0)$
end while
return $P4MP$

Algorithm 2 Calculate the scalar multiplication of scalar k ($Q = kP$)

Input: Point $P_0(x, y)$, scalar k, last index of k (li), $P4MP(m, P, t)$
Output: The scalar multiplication (Q) of $kP := Q(x, y)$

$i \leftarrow li$
while $i > 0$ and $k \neq 0$ **do**
  if $k \geq 2 \times P4MP(m_i)$ **then**
    $P4MP(t_i) \leftarrow 2$
    $k \leftarrow k - 2 \times P4MP(m_i)$
  else if $k \geq P4MP(m_i)$ **then**
    $P4MP(t_i) \leftarrow 1$
    $k \leftarrow k - P4MP(m_i)$
  end if
  $i \leftarrow i - 1$
end while
$Q(x, y) \leftarrow (0, 0)$
for $i \leftarrow 1$ to li **do**
  if $P4MP(t_i) \neq 0$ **then**
    $Q \leftarrow Q + (P4MP(P_i) \cdot P4MP(t_i))$
  end if
end for
return Q

Hence, the number of multiplications $T(n)$ satisfies

$$T(n) = T(\frac{n}{2}) + 1 \tag{22}$$

where $n = 2^i$, and $i = \log_2 n$ from which we find $T(n)$ and the answer of calculating the numbers in Modified Pell sequence is $O(\log_2 n)$.

4.1 Creating Modified Pell Table

The process of creating a table of points P at the numbers in the Modified Pell sequence ($P4MP$ table) stores the number at the index i (m_i) in the sequence, the point P derived from scalar multiplication at the index i (P_i), and the number of times to calculate the scalar multiplication at the point P_i (t_i). The creation of a $P4MP$ table requires finding the last index (li) of scalar k using (17), calculating each of

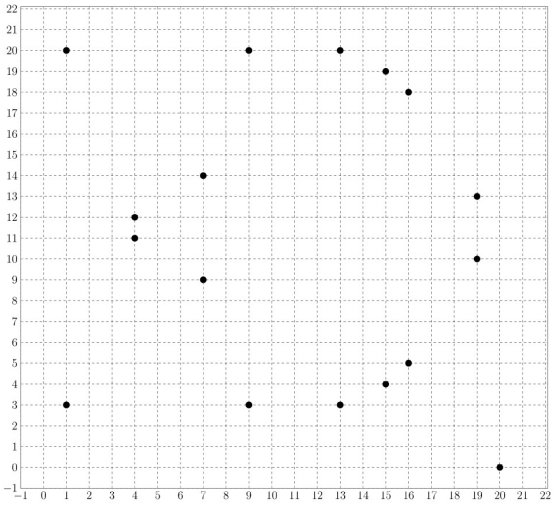

Fig.4: *The set of points of* $E_{23} : y^2 = x^3 + x + 7 \ mod \ 23$.

the Modified Pell numbers from index 1 to the last index (li) using Theorem 2, and finding the point P on the elliptic curve related to each index i using Definition 3 and Definition 4, respectively. Algorithm 1 outlines the process.

4.2 Calculating Large Scalar Multiplication

The scalar multiplication (Q) of scalar k at point P on an elliptic curve (kP) uses the $P4MP$ table. The process is described in Algorithm 2 and Example 5 demonstrates how $LSMA - MPN$ works.

Example 5: Suppose an elliptic curve $E_{23} : y^2 = x^3 + x + 7 \ mod \ 23$. Find Q for the scalar $k = 293$ and the starting point $P_0 = (7, 9)$. Fig. 4 shows the graph of E_{23}. The set of points on this curve is {(1,3), (1,20), (4,11), (4,12), (7,9), (7,14), (9,3), (9,20), (13,3), (13,20), (15,4), (15,19), (16,5), (16,18),(19,10), (19,13), (20,0)}.

Following Algorithm 1 to create the $P4MP$ table, the last index i of scalar $k = 293$ in the Modified Pell sequence is calculated using (17) as follows:

$$
\begin{aligned}
li &= \lfloor 1 + \left(\frac{1}{\log(1+\sqrt{2})} \times \log(293) + \log\sqrt{2} \times \frac{1}{\log(1+\sqrt{2})} \right) \rfloor \\
&= \lfloor 1 + (2.6125 \times 2.4669 + 0.1505 \times 2.6125) \rfloor \\
&= \lfloor 1 + 6.8379 \rfloor \\
&= \lfloor 7.8379 \rfloor \\
&= 7
\end{aligned}
$$

Then, we find the Modified Pell numbers (m_i) from index 1($i = 1$) to the last index ($i = li$) via the 2×2 matrix shown in Theorem 2 (Example 4). Next, we calculate point Pi on the elliptic curve related to the number mi using Definition 3 and 4, respectively. The results are shown in Table 2.

Table 2: *The P4MP table of* $k = 293$.

i	1	2	3	4	5	6	7
m_i	1	1	3	7	17	41	99
P_i	(7,9)	(7,9)	(4,11)	(16,5)	(7,14)	(19,10)	(20,0)
t_i	0	0	0	0	0	0	0

After the $P4MP$ table has been created, Algorithm 2 is applied with two sub-processes. The first sub-process determines the number of times to multiply the points (t_i) of $k = 293$. The sub-process starts from the last index in the $P4MP$ table ($l_i = 7$) which is related to 99. We multiply 99 by 2 (*i.e.*, $2 \times 99 = 198$) and subtract product from k to find the remaining scalar k: (*i.e.*, 293 - 198 = 95). We then read the number at the previous index in the $P4MP$ table to verify that m_i is less than or equal to the remaining scalar k. The number at index 6 (m_6) is 41, and we multiply 41 by 2 (*i.e.*, $2 \times 41 = 82$). The process is continued to find the remaining scalar k is 13 (*i.e.*, 95 - 82 = 13) and the number at index 4 (m_4) is found. Thereafter, we define index 4 and the remaining scalar k is 6 (i.e., 13 - 7 = 6). Finally, using t_i from Table 3, the number at index 3 (m_3) is applied two times (*i.e.*, $2 \times 3 = 6$). The second sub-process calculates the scalar multiplication $Q = 293P$ using ti in the $P4MP$ table (see Table 3). Consequently, the scalar multiplication of $293P$ consists of $2(99P) = (7, 9)$; $2(41P) = (9, 3)$; $7P = (16, 5)$; and $2(3P) = (1, 3)$. Therefore, the co-ordinates of $293P = 2(99P) + 2(41P) + 1(7P) + 2(3P)$ yields (1,3).

Table 3: *The number of times in P4MP of* $k = 293$.

i	1	2	3	4	5	6	7
m_i	1	1	3	7	17	41	99
P_i	(7,9)	(7,9)	(4,11)	(16,5)	(7,14)	(19,10)	(20,0)
t_i	0	0	2	1	0	2	2

5. EFFICIENCY ANALYSIS

This section presents the computation time by applying the 2 × 2 matrix which compares the results with the iteration method, the time complexity of $LSMA - MPN$ algorithm, experimental results and discussion.

5.1 2 × 2 Matrix representation

We calculated large scalar multiplication using Modified Pell numbers and applied our 2 × 2 matrix representation to reduce computation time. Figure 5 describes the comparison of computation time between the iteration method and our matrix representation method. The result shows that the computation time by applying our matrix representation method to calculate scalar multiplication takes less time than the iteration method, especially when the index i is very large.

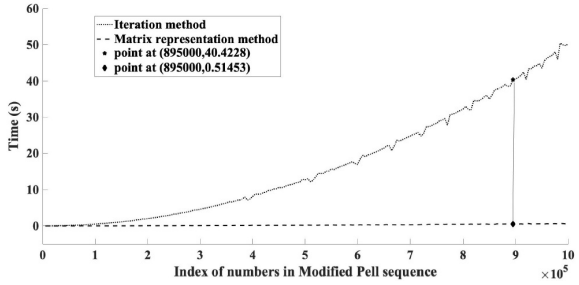

Fig.5: *Comparison of the computation time between the iteration method and our matrix representation method.*

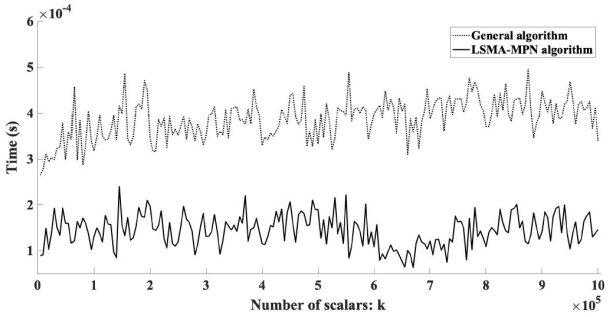

Fig.6: *Computation time of the scalar multiplication on the elliptic curve between the general and LSMA - MPN algorithm.*

5.2 Time Complexity

To discusses the time complexity, Lemma 2 is defined for the general algorithm. For $LSMA - MPN$ algorithm, we define Lemma 3 and 4.

Lemma 2: For the general algorithm, the cost of calculating scalar multiplication is $O(k)$.

Proof: The cost of the scalar multiplication depends on the cost of arithmetic operations. The operations of point addition and point doubling are $O(1)$. The process to calculate kP is used k times to calculate the scalar multiplication. Equation (23) shows the time complexity of the general algorithm.

$$O(1 \times k) = O(k) \tag{23}$$

Therefore, the time complexity of the general algorithm is $O(k)$.

Lemma 3: For the $LSMA - MPN$ algorithm, the best case of calculating scalar multiplication is $O(\log_2 n)$.

Proof: The best case for the $LSMA - MPN$ algorithm occurs when a scalar k equals to the number of last index (m_{li}) in $P4MP$ table. The time to calculate the scalar multiplication k using the 2×2 matrix to find the number in $P4MP$ table is $O(\log_2 n)$ where $n = l_i$ and $l_i < k$.

Lemma 4: For the $LSMA - MPN$ algorithm, the worst case of calculating scalar multiplication is $O(n)$.

Proof: The worst case for the $LSMA - MPN$ algorithm includes the time to create the $P4MP$ table using Algorithm 1, which is $O(\log_2 n)$ where $n = l_i$, the time for finding the numbers of calculating point doubling in the $P4MP$ table which is $O(n)$, and the time to calculate the scalar multiplication k which is $O(n)$. The time complexity of the worst case is shown in (24)

$$O(\log_2 n + n + n) = O(n) \tag{24}$$

Therefore, the time complexity of the worst case for the $LSMA - MPN$ algorithm is $O(n)$.

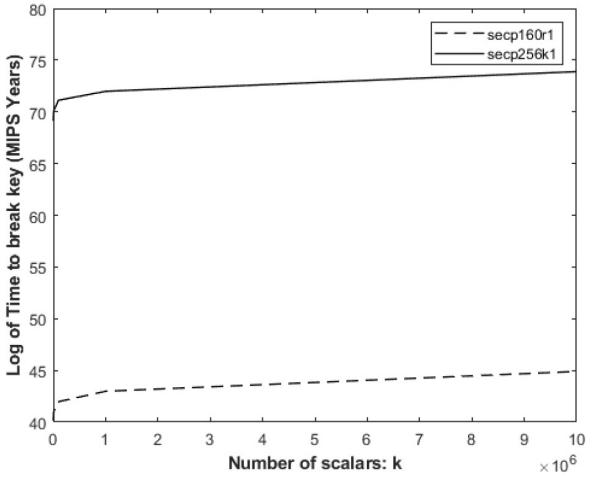

Fig.7: *Time to break key using standardized curves:secp160r1 and secp256k1.*

5.3 Experimental Results

This work compared the performance of a general algorithm to the $LSMA - MPN$ algorithm in calculating scalar multiplications on an elliptic curve implemented in Python. The hardware used in this experiment was a computer with a 3.1 GHz Core i5 processor and 16 GB memory. Using both the general algorithm and the $LSMA - MPN$ algorithm, we compared computation times for multiplications of scalars k from k = 5,000 to k = 1,000,000. Figure 6 shows the high speed of scalar multiplication using $LSMA - MPN$, which was also more efficient than the general algorithm in all of the elliptic curve point multiplications. The experimental result shows that $LSMA - MPN$ reduces the computation time by approximately 67% in comparison with the general algorithm.

In addition, to achieve reasonable security, we compared the time required to break a key for a 160-bit key and a 256-bit key. The values are computed in MIPS years. A MIPS year represents the computing

time of 1 year on a machine capable of performing one million instructions per second. The scalar multiplication for two SECG-standardized elliptic curves over a prime field, $secp160r1$ for 160-bit key (80-bit security level) and $secp256k1$ for 256-bit key (128-bit security level) [38, 39] were implemented. Figure 7 shows the time taken on a brute force attack for multiplication of scalar k from $k = 1; 000$ to $10; 000; 000$. To clarify the result in Fig. 7, Table 4 displays some scalars (k) with the times to break a key. The experimental result indicated that at the same key size

Table 4: *Examples of time to break key using standardized curves: secp160r1 and secp256k1.*

Number of scalars: k	Time to break key (MIPS Years)	
	secp160r1	**secp256k1**
5,000	1.45×10^{40}	1.53×10^{69}
10,000	1.16×10^{41}	1.35×10^{70}
50,000	1.58×10^{41}	2.05×10^{70}
100,000	9.47×10^{41}	1.33×10^{71}
500,000	1.02×10^{42}	1.49×10^{71}
1,000,000	9.61×10^{42}	9.62×10^{71}

6. CONCLUSION

To calculate large scalar multiplication on the elliptic curve, we proposed an $LSMA - MPN$ algorithm based on Modified Pell numbers and 2×2 matrix representation to speed up the large scalar multiplication of point addition and point doubling. The results showed that the proposed algorithm processed scalar multiplications at high speed and was more efficient than a general algorithm in all of the elliptic curve point multiplications. Using our proposed algorithm to calculate Modified Pell numbers in a sequence, we also demonstrated a difference in execution time between the iteration and our matrix representation methods. Additionally, we compared the time required to break keys for high security using two SECG-standardized elliptic curves over a prime field: $secp160r1$ and $secp256k1$, which are 80-bit level security level and 128-bit security level, respectively. This finding proves that at the same key size with a large scalar (k), breaking a key required much more computing time. In future work, we will compare the $LSMA - MPN$ algorithm efficiency with other related algorithms. Furthermore, we will apply $LSMA - MPN$ that is suitable for the generated key to propose a new efficient cryptographic algorithm for encryption and decryption of data.

ACKNOWLEDGMENT

This research work was financially supported by the Strategic Scholarships Fellowships Frontier Research Networks (Specific for Southern Region), the Higher Education Commission, Ministry of Higher Education, Science, Research and Innovation. The authors also would like to acknowledge Mr. Thomas Duncan Coyne for proofreading the article.

References

[1] C. Easttom, "Modern Cryptography: Applied Mathematics for Encryption and Information Security," *McGraw-Hill Education*, 2015.

[2] C. Paar and J. Pelzl, "Understanding Cryptography," *Springer Berlin Heidelberg*, 2010.

[3] B. AlBelooshi, E. Damiani, K. Salah, and T. Martin, "Securing Cryptographic Keys in the Cloud: A Survey," *IEEE Cloud Computing*, Vol. 3, No. 4, pp. 42-56, 2016.

[4] L. Yibin, K. Gai, L. Qiu, M. Qiu, and Z. Huid, "Intelligent cryptography approach for secure distributed big data storage in cloud computing," *Information Sciences*, Vol. 387, pp. 103-115, 2017.

[5] W. Stallings, "Cryptography and Network Security: Principles and Practice, 7^{th} ed.," *Pearson Education Limited*, October 11, 2016.

[6] B. Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C," *John Wiley & Sons Inc.*, New York, United States, 1995.

[7] S. Y. Yan, "Computational Number Theory and Modern Cryptography, 1st ed.," *Higher Education Press*, November 27, 2012.

[8] Certicom Research, "Standards for Efficient Cryptography, SEC 1: Elliptic Curve Cryptography," Certicom Corp., Version 1.0, September 20, 2000.

[9] M. Bedoui, B. Bouallegue, B. Hamdi, and M. Machhout, "An Efficient Fault Detection Method for Elliptic Curve Scalar Multiplication Montgomery Algorithm," *Proceeding of 2019 IEEE International Conference on Design & Test of Integrated Micro & Nano-Systems (DTS)*, pp. 1-5, 2019.

[10] S. Chandel, et al., "A Multi-dimensional Adversary Analysis of RSA and ECC in Blockchain Encryption," *Proceeding of FICC 2019: Advances in Information and Communication*, pp. 988-1003, 2019.

[11] V. S. Miller, "Use of elliptic curves in cryptography," *Proceeding of CRYPTO 1985: Advances in Cryptology-CRYPTO'85 Proceedings*, pp. 417-426, 1985.

[12] F. Akhter, "Faster Scalar Multiplication Algorithm to Implement a Secured Elliptic Curve Cryptography System," *International Journal of Advanced Computer Science and Applications*, Vol. 7, No. 1, pp. 639-644, 2016.

[13] A. Overmars and S. Venkatraman, "A New Method of Golden Ratio Computation for Faster Cryptosystems," *Proceeding of 2017 Cybersecurity and Cyberforensics Conference (CCC)*, pp. 915, 2017.

[14] J. J. Bravo, P. Das, S. Guzman, and S. Laisham, "Powers in products of Terms of Pell's and Pell-

Lucus Sequences," *Indian Statistical Institute*, Delhi Centre, New Delhi, India, pp. 1-10, 2010.

[15] A. F. Horadam, "Applications of Modified Pell Numbers to Representations," *Ulam Quaterly*, Vol. 3, No. 1, pp. 34-53, 1994.

[16] V. W. Spinadel, "The metallic means family and renormalization group techniques," *Trudy Inst. Mat. i Mekh. UrO RAN*, Vol. 6, No. 1, pp. 173-189, 2000.

[17] U. Priyatharsan, P. L. Rupasinghe, and I. Murray, "A New Elliptic Curve Cryptographic System over the Finite Fields," *Proceeding of 6^{th} National Conference on Technology and Menagement (NCTM)*, pp. 164-169, 2017.

[18] R. Harkanson and Y. Kim, "Applications of Elliptic Curve Cryptography A light introduction to elliptic curves and a survey of their applications," *Proceeding of 12^{th} Annual Conference on Cyber and Information Security Research (CISRC)*, pp. 1-7, 2017.

[19] N. Koblitz, "Elliptic curve cryptosystems," *Mathematics of Computation*, Vol. 48, No. 177, pp. 203-209, 1987.

[20] K. Gupta and S. Silakari, "ECC over RSA for Asymmetric Encryption: A Review," *IJCSI International Journal of Computer Science Issues*, Vol. 8, Issue 3, No. 2, pp. 370-375, 2011.

[21] M. Alam, I. Jahan, L. J. Rozario, and I. Jerin, "A Comparative Study of RSA and ECC and Implementation of ECC on Embedded Systems," *International Journal of Innovative Research in Advanced Engineering (IJIRAE)*, Vol. 3, Issue 3, pp. 86-93, 2016.

[22] B. Akanksha and A. Arun, "Providing Security, Integrity and Authentication Using ECC Algorithm in cloud storage," *Proceeding of 2017 International Conference on Computer Communication and Informatics (ICCCI)*, pp. 1-5, 2017.

[23] S. Gajbhiye, S. Karmakar, and M. Sharma, "Study of Finite Field over Elliptic Curve: Arithmetic Means," In *the International Journal of Computer Applications*, Vol. 47, No. 17, pp. 32-38, 2012.

[24] T. S. Mathew and S. Suresh, "Comparative Analysis of AES and ECC in Automated Metering Infrastructure," *Proceeding of 2017 international Conference on Wireless Communications, Signal Processing and Networking (WiSPNET)*, pp. 934-938, 2017.

[25] S. Gueron and V. Krasnov, "Fast prime field elliptic-curve cryptography with 256-bit primes," *Journal of Cryptographic Engineering*, Vol. 5, pp. 141-151, 2015.

[26] M. Bluhm and S. Gueron, "Fast software implementation of binary elliptic curve cryptography," *Journal of Cryptographic Engineering*, Vol. 5, pp. 215-226, 2015.

[27] D. R. Susantio and I. Muchtadi-Alamsyah, "Implementation of Elliptic Curve Cryptography in Binary Field," *Journal of Physics: Conference Series*, Vol. 710, pp. 1-9, 2016.

[28] K. Phalakarn, K. Phalakarn, and V. Suppakitpaisarn, "Optimal Representation for Right-to-Left Parallel Scalar and Multi-Scalar Point Multiplication," *International Journal of Networking and Computing*, Vol. 8, No. 2, pp. 166-185, 2018.

[29] D. Khleborodov, "Fast elliptic curve point multiplication based on window Non-Adjacent Form method," *Applied Mathematics and Computation*, Vol. 334, pp. 41-59, 2018.

[30] K. Javeed, X. Wang, and M. Scott, "High performance hardware support for elliptic curve cryptography over general prime field," Microprocessors and Microsystems, No. 50, pp. 331-342, 2017.

[31] M. M. Panchbhai and U. S. Ghodeswar, "Implementation of Point Addition & Point Doubling for Elliptic Curve," *Proceeding of International Conference on Communication and Signal Processing*, pp. 746-749, 2015.

[32] A. J. Raphael and V. Sundaram, "Secured Communication through Fibonacci Numbers and Unicode Symbols," *International Journal of Scientific & Engineering Research*, Vol. 3, Issue 4, pp. 1-5, 2012.

[33] M. Mukherjee and D. Samanta, "Fibonacci Based Text Hiding Using Image Cryptography," *Lecture Notes on Information Theory*, Vol. 2, No. 2, pp. 172-176, 2014.

[34] P. Agarwal, N. Agarwal, and R. Saxena, "Data Encryption through Fibonacci Sequence and Unicode Characters," *International Journal of Computer Science and Information Technology*, Vol. 5, No. 2, pp. 79-82, 2015.

[35] N. Zhang and S. Tan, "Elliptic Curve Scalar Multiplication Based on Fibonacci Number," *Proceeding of 5^{th} International Conference on Intelligent Networking and Collaborative Systems*, pp. 507-510, 2013.

[36] S. Liu, G. Qi, and X. A. Wang, "Fast and Secure Elliptic Curve Scalar Multiplication Algorithm Based on a Kind of Deformed Fibonacci-type Series," *Proceeding of 10^{th} International Conference on P2P, Parallel, Grid, Cloud and Internet Computing*, pp. 398-402, 2015.

[37] F. Duemong and L. Preechaveerakul, "Applying Pell Numbers for Efficient Elliptic Curve Large Scalar Multiplication," [*Proceeding of 22^{nd} International Computer Science and Engineering Conference (ICSEC)*, pp. 1-4, 2018.

[38] Certicom Research, "Standards for Efficient Cryptography, SEC 2: Recommended Elliptic Curve Domain Parameters," *Certicom Corp.*, Version 2.0, January 27, 2010.

[39] R. Herold and M. K. Rogers, "Encyclopedia of Information Assurance," *Auerbach Publications*, New York, December, 2010.

Fudailah Duemong received her B.Sc. degree in Applied Mathematics from Prince of Songkla University (Pattani Campus), Thailand in 2007. She received her M.Sc. degree in Computer Science from Prince of Songkla University (Hat Yai Campus), Thailand in 2010. She has been a lecturer in the Department of Computer at Yala Rajabhat Univeristy, Yala, Thialand since June 2011. She is currently a Ph.D. candidate in Computer Science at Prince of Songkla University (Hat Yai Campus), Thailand. Her research interests include Modern cryptography based on Mathematical theory, algorithms for Cryptography, Data Mining and Database.

Ladda Preechaveerakul received her B.Sc. degree in Mathematics from Prince of Songkla University (Hat Yai Campus), Thailand in 1989. She received her M.Sc. degree in Applied Statistics from National Institute of Development Administration in 1994 and her Ph.D. degree in Computer Science from Chulalongkorn University, Thailand in 2006. Currently, she works as an Assistant Professor at Faculty of Science, Prince of Songkla University, Hat Yai, Songkhla, Thailand. Her research interests include Information Security and Mobile Computing.