Mathematical Modelling of Engineering Problems
Vol. 11, No. 10, October, 2024, pp. 2763-2769
Journal homepage: http://iieta.org/journals/mmep

Advancing Security: A Novel Approach with Linear Bézier Curve in Menezes-Vanstone Elliptic Curve Cryptosystem

Dua M. Ghadi

Wasit Education Directorate, Ministry of Education, Baghdad 10011, Iraq

Corresponding Author Email: mdua1093@gmail.com

https://doi.org/10.18280/mmep.111018

Received: 1 October 2023
Revised: 12 February 2024
Accepted: 25 February 2024
Available online: 31 October 2024

Keywords:
Menezes-Vanstone Elliptic Curve Cryptosystem, linear Bézier, Elliptic Curve Cryptography, encryption, decryption, complexity, randomness, cryptosystem enhancement, security

ABSTRACT

Elliptic Curve Cryptography (ECC) and the related cryptographic systems have become widespread alternatives to provide secure channels of communication within our ever-expanding networked universe. Cryptosystems play an important role in keeping the privacy of information when it is sent over the network or stored. But as threats and resources grow exponentially higher the only solution is to constantly develop and make more complex systems to remain safe. The Menezes-Vanstone Elliptic Curve Cryptosystem (MVECC) is a cryptographic scheme based on elliptic curve mathematics, designed for data encryption and transmission. A linear Bézier curve is a mathematical construct used in computer graphics, and it's defined by a set of control points. This paper presents an extension to MVECC with a linear Bézier curve equation. This is implemented using a modification of the linear Bézier curve equation to propose the encryption/decryption equations. Also, define a new secret key κ, with u (where $u \in [0,1]$) and create it with scalar multiplication in ECC by the public key R_k of MVECC. The purpose of this paper is to enhance the general complexity of the MVECC for key generation, encryption, and decryption operations. This research paper aims to increase the complexity of MVECC by incorporating the behavior of the linear Bézier curve equation. The test of randomness of the ciphertext of the proposed method is depends on several tests issued by the National Institute of Standards and Technology (NIST). The timing of the proposed method has been tested, and the required processes have been calculated. Results show that the proposed modification has the complexity and randomness to provide higher security levels.

1. INTRODUCTION

Networks have become ubiquitous to all aspects of life and particularly because of fast technical progress. They help us obtain information, talk to each other, do business online, and benefit from many other online activities. However, networks also create some threats and risks to protecting and privacy of our data. So, there is the requirement of secure methods to shield our data from unauthorized access. Cryptography is the science concerned with hiding and verifying information. It includes protocols, algorithms and techniques to encrypt and decrypt data in a secure method and letting each element of the data be validated.

Cryptography is mention to protocols, algorithms and techniques that can encode a message or decode a ciphertext in such a way as to protect the privacy of the message during its digital transmission. The Bézier curve is one of the basic mathematical concepts common to both computer graphics and design. In the field of automobile design invented between 1958 and 1960 by French scientists Pierre Bézier and Paul de Casteljau, who took advantage of the polynomial properties that is convenient to manipulate, control or analyze [1].

The theory of asymmetric cryptography began to appear in 1986, the year that Miller [2] and Koblitz [3] introduced the idea of elliptic curves (EC) [4], where EC has become the basis for offering novel methods of improved data protection, encryption and secure transmission. It was Miller's seminal paper back in 1987 [2], and Koblitz's paper a year earlier in 1986 [3], that saw elliptic curves introduced into cryptography at its birth. Since then, elliptic curves have been subjected to intense research in the field of cryptography, resulting in many encryption algorithms that provide stronger security. ECC is dependent on the discrete logarithm issue and has the potential to be applied using elliptic curves in EC protocols such as Diffie-Hellman key exchange [5].

Central to the discussion of elliptic curve cryptosystems is the challenge posed by the Elliptic Curve Discrete Logarithm Problem (ECDLP) including the MVECC. The security of ECC is based on the assumption that it is computationally infeasible to find the discrete logarithm of a point on an elliptic curve with respect to a publicly known base point that is the attacker attempts to discover the private key by trying all possible values. The difficulty of solving ECDLP is directly depend on size of the key of the EC. Therefore, larger key sizes are needed to keep the level of security. Mathematically, given two points P and B on the curve, finding an integer 'n' such that $B = nP$ is believed to be computationally hard. To break an ECC system, an attacker would need to find the private key (the integer n) from point P to perform unauthorized decryption [6].

Understanding the impacts of ECDLP were efficiently savable, it could make ECC cryptosystems unsafe. This research paper introduces a novel approach by incorporating MVECC with a linear Bézier curve to create a highly intricate cryptosystem and against evolving threats. The primary aim of this paper is to enhance the complexity of an elliptic curve cryptosystem. The proposed modification hinges on the generation of a new key and modifying the encryption and decryption procedures. By doing so, it significantly elevates the difficulty level for potential attackers attempting to solve DLP. This heightened complexity arises from the utilization of two private keys, the first generated by MVECC and the second created secretly through $u \in [0,1]$ and scalar multiplication between the sender and receiver, further increasing the cryptosystem's security.

2. RELATED WORKS

Many researchers have developed new methods and improvements to many cryptosystems. Some of them developed Bézier-curve-based cryptosystems. These studies have used Bézier curves with some cryptosystems, as in our research paper, where we use a linear Bézier curve with one of elliptic curve cryptosystems, which is MVECC. We present some of them, such as in our previous work in 2020 [7], where we presented an improvement of MVECC based on the quadratic Bézier curves, which is dependent on the quadratic Bézier equation in encryption and decryption processes. Srividya and Akhila [8] presented a cryptography technique based on the quartic Bézier curve over Galois field GF (Pm) to improve security data while maintaining low computational complexity. To modify ElGamal ECC, Ghadi [9] has used a quadratic Bézier curve for encrypting key and sending it with ciphertext to the receiver side based on PGP behaviour. In the study conducted by Abd El-Latif et al. [10], to enhance the chaotic key sequence, an expanded sequence employing a chaotic map and Bernstein-Bézier curve is shown. The scrambling is performed in the DCT domain using changeable control parameters. To increase safety, image pixels' gray values are widely used by applying a keystream derived from the chaotic map and plain image's enlarged key sequence.

For researchers who modified and developed the MVECC, we present some of them, one of which was our previous work in 2020 [7], which has been mentioned in the paragraph above. Rahouma [11] proposed a modified variant of the MVECC using multiple curves with corresponding keys. Messages are divided into blocks and protected through encryption with a random sequence generator and modulus calculation. The system keys include encryption and decryption keys and random sequence generator coefficients. Sagheer [12] introduced Elliptic Curves Cryptography (ECC) as a recent public key cryptosystem and proposes three modified techniques based on elliptic curves for the ElGamal cryptosystem. These techniques demonstrate a significant reduction in calculation time and increased ciphertext confusion compared to the original methods. ECC offers comparable security to RSA with shorter keys, improves storage and performance, and outperforms other secure ciphers like RSA, ElGamal encryption and decryption, and MVECC techniques by eliminating the need for inverse operations and plaintext embedding. Sadiq and Kadhim [13] proposed a new variation of the MVECC to enhance encryption efficiency and security. Unlike the original ElGamal scheme, MVECC avoids encoding plaintext in the elliptic curve, but it requires computing inverses, affecting efficiency. The proposed variation mixes plaintext units, increasing confusion, and only requires one inverse operation, making it more efficient and secure than MVECC.

These studies affirm the effectiveness of employing Bézier curves in cryptographic applications and modifying MVECC schemes. In our paper, we apply the linear Bézier curve equation to the encryption and decryption processes of MVECC. Additionally, we introduce a new secure key for encrypting and decrypting data.

3. LINEAR BÉZIER CURVES

Bézier curves of degree m is regarded as having a control point sequence of $m+1$ [1]. These points describe Bézier curves [14]. $B_j{}^m(u)$ symbolizes a Bézier curve of degree m that has $m+1$ control points with mixture functions.

$$B_j{}^m(u) = \binom{m}{j}(1-u)^{m-j}u^j,\ j = 0,1,2,\ldots,m$$

Such that $\binom{m}{j} = \frac{m!}{j!(m-j)!}$, for $B_j{}^m(u)$ is a jth Bernstein polynomial with degree m. Then, the Bezier curve equation is [1]:

$$q(u) = \sum_{j=0}^{m}\binom{m}{j}(1-u)^{m-j}u^j L_j$$

$$q(u) = \sum_{j=0}^{m} B_j{}^m(u)\,L_j, \quad 0 \le u \le 1$$

$$q(u) = B_0{}^m(u)L_0 + B_1{}^m(u)L_1 + B_2{}^m(u)L_2 + \cdots + B_j{}^m(u)L_j$$

Therefore, the linear Bezier curve is identified with two control points, L_0 and L_1, and its equation is:

$$q(u) = (1-u)\,L_0 + uL_1, \text{ for } u \in [0,1] \qquad (1)$$

In the interval [0, 1], the linear Bezier curve is recognized. So, the starting point is $q(0) = L_0$ and the ending point is $q(1) = L_1$, assuming the Bezier curve interpolates between the first and last control points [14].

In general, Bézier curves provide suitable performance in cryptography, based on some of the research that has been mentioned in Section 2. Its simplicity can contribute to computational efficiency and complexity, which enhances the overall security of the MVECC. In short, the importance of linear Bézier curves lies in their mathematical simplicity and their ability to provide a new approach to cryptosystem processes. As mentioned earlier, ECC including MVECC, depend on the difficulty of solving ECDLP. In this paper, the proposed modification can increase the complexity of these problems, making them more computationally difficult for attackers.

4. ELLIPTIC CURVE CRYPTOGRAPHY

Miller [2] and Koblitz [3] presented Elliptic Curve Cryptography (ECC). This cryptographic technique allows the implementation of various systems such as Diffie-Hellman key exchange, the ElGamal public key encryption, and signature systems.

ECC can provide security similar to existing public-key techniques while using shorter key lengths [15]. RSA and ECC are the most well-known cryptosystems in public key cryptography, and they get the same security protections, with the exception that ECC works with shorter keys than RSA (160-256 bit vs. 1024-3072 bit). It depends on the (DLP), which could be implemented in EC methodologies such as Diffie-Hellman key exchange using elliptic curve. ECC systems have benefits over RSA and discrete logarithm systems in terms of fewer calculations and smaller signatures and keys. Elliptic curves have a stronger mathematical connection than RSA and discrete logarithm schemes [5, 7].

By Eq. (2), elliptic curve E over finite prime field F_p and it symbolize $E(F_p)$, can be defined and arithmetic operations on its points can be performed as follows [9, 16, 17]:

$$E: y^2 = x^3 + ax + b \quad mod\, p \tag{2}$$

Consider the following: p is an odd prime that means that ($p > 2,3$), a and b must satisfy the formula $(4a^3 + 27b^2)\ mod\ p \neq 0$ in F_p, and each point $P = (x, y)$ in E. The mathematical operations on the points of E are summarized below:

1) The Addition Operation: If $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ are points on E, then $P + Q = (x_3, y_3)$ and calculated as below:

$$\lambda = \frac{(y_2 - y_1)}{(x_2 - x_1)} \tag{3}$$

$$x_3 = (\lambda^2 - x_1 - x_2) mod\, p \tag{4}$$

$$y_3 = [\lambda(x_1 - x_3) - y_1] mod\, p \tag{5}$$

2) Doubling Operation: If $P = (x_1, y_1)$ is a point on E, the doubling P is calculated as: $2P = P + P = (x_3, y_3)$, such that:

$$\lambda = \frac{(3\, {x_1}^2 + a)}{(2\, y_1)} mod\, p \tag{6}$$

$$x_3 = (\lambda^2 - 2\, x_1) mod\, p \tag{7}$$

$$y_3 = [\lambda(x_1 - x_3) - y_1] mod\, p \tag{8}$$

3) Scalar Multiplication Operation: If $P = (x, y)$ is a point on E and let n be a positive integer, the scalar multiplication of P is the repeated addition to P itself, as shown: $\underbrace{nP = P + P}_{n\text{-times}} + \cdots + P$.

4.1 Menezes-Vanstone Elliptic Curve Cryptosystem

This cryptosystem employs ECC to ensure secure data transmission between transmitters and receivers [7, 15]. The MVECC does not work with discrete logarithms. Users will very certainly succeed in embedding data once a curve has a point on it [13]. MVECC messages $M = (m_1, m_2)$ and ciphertexts $C = (c_1, c_2)$ can alternatively be randomized ordered pairs of nonzero elements (they just don't have to contain points in E) [18]. The MVECC algorithm is designed as below:

Setting Up the Cryptosystem Between Sender and Receiver:

1) Public $E: y^2 = x^3 + ax + b\ mod\, p$, where a, b, and p satisfy: $(4a^3 + 27b^2)\ mod\ p \neq 0$.
2) Public base point $P \in E(F_p)$.

```
Algorithm 1: Key Production

Receiver's Keys Production
Inputs: $P \in E(F_p)$.
Outputs: Public Key $R_k$.
  1) Randomly, choose a private integer $n$.
  2) Generate a public key $R_k$ by calculating:
$nP = R_k$.

Sender's Keys Production
Inputs: Public Key $R_k$.
Outputs: Secret Key $S_k = (s_1, s_2)$.
  1) Randomly, choose a private integer $e$.
  2) Generate the key $S_k = (s_1, s_2)$ using $e$ and $R_k$:
     $S_k = eR_k$.
```

```
Algorithm 2: Encryption (The Sender)

Inputs: Message $M = (m_1, m_2)$, $S_k$, $P \in E(F_p)$.
Outputs: Ciphertext $C = (c_1, c_2)$, $eP$.
  1) Choose message $M = (m_1,\ m_2)$.
  2) Encrypt $M$ with $S_k$ by:
         $c_1 = m_1 * s_1\ mod\, p$
         $c_2 = m_2 * s_2\ mod\, p$
  3) Calculate $eP$.
  4) Send $\{C, eP\}$ to the receiver.
```

```
Algorithm 3: Decryption (The Receiver)

Inputs: Ciphertext $C = (c_1, c_2)$, $eP$, Private Key $n$.
Outputs: Message $M = (m_1, m_2)$.
  1) Receive $\{C, eP\}$ from the sender.
  2) Generate $S_k = n(eP)$.
  3) Decrypt $C$ with $S_k$ by:
         $m_1 = c_1 * s_1^{-1}\ mod\, p$
         $m_2 = c_2 * s_2^{-1}\ mod\, p$
  4) The message $M = (m_1, m_2)$.
```

5. THE PROPOSED METHOD OF MVECC

The algorithm below performs the proposed technique for MVECC encryption and decryption using the Eq. (1). Through Eq. (1), a cryptographic equation for enhancing MVECC is created and used in encrypting messages and decrypting ciphertext with the generated secret key $S_k = (s_1, s_2)$ and $u \in [0,1]$ which is considered to be an agreed-upon random secret key by sender and receiver.

In proposed algorithm, we used S_k as a control point instead of a Bézier point. We only chose $u \in [0,1]$ and computed it using module p, then used scalar multiplication with public key R_k and $u \in [0,1]\ mod\, p = \Omega$ to generate the new key $\kappa = (\kappa_1, \kappa_2)$ and keep it secret. The process of encrypting and decrypting messages is only done through these two secret keys.

The algorithm below explains the proposed encryption and decryption technique as well as how to use the keys $S_k =$

(s_1, s_2) and $\kappa = (\kappa_1, \kappa_2)$ encrypting and decrypting procedures.

Setting Up the Proposed Cryptosystem Between Sender and Receiver

1) Public $E: y^2 = x^3 + ax + b \mod p$, where a, b, and p satisfy: $(4a^3 + 27b^2) \mod p \neq 0$.
2) Public base point $P \in E(F_p)$.
3) Random secret $u \in [0,1]$.

```
Algorithm 4: Proposed Key Production
Receiver's Keys Production
Inputs: $P \in E(F_p)$.
Outputs: Public Key $R_k$, Secret number $\Omega$, Secret key $\kappa = (\kappa_1, \kappa_2)$.
    1) Randomly, choose a private integer $n$.
    2) Generate a public key $R_k$ by calculating: $nP = R_k$.
    3) Compute secret number by $u \in [0,1] \mod p = \Omega$.
    4) Generate a secret key $\kappa$ using his public key $R_k$ by calculating: $\Omega R_k = \kappa = (\kappa_1, \kappa_2)$.
Sender's Keys Production
Inputs: Public key $R_k$.
Outputs: Secret key $S_k = (s_1, s_2)$, Secret number $\Omega$, Secret key $\kappa = (\kappa_1, \kappa_2)$.
    1) Randomly, choose a private integer $e$.
    2) Generate the key $S_k = (s_1, s_2)$, using $e$ and $R_k$: $S_k = eR_k$.
    3) Compute secret number by $u \in [0,1] \mod p = \Omega$.
    4) Generate a secret key $\kappa$ using his public key $R_k$ by calculating: $\Omega R_k = \kappa = (\kappa_1, \kappa_2)$.
```

```
Algorithm 5: Proposed Encryption (The Sender)
Inputs: Message $M = (m_1, m_2)$, $S_k$, $\kappa$, $P \in E(F_p)$, private integer $e$.
Outputs: Ciphertext $C = (c_1, c_2)$, $eP$.
    1) Choose message $M = (m_1, m_2)$.
    2) Encrypt $M$ with $S_k$ and $\kappa$ by:
           $c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$
           $c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1 \mod p$
    3) Calculate $eP$.
    4) Send $\{C, eP\}$ to the receiver.
```

```
Algorithm 6: Proposed Decryption (The Receiver)
Inputs: Ciphertext $C = (c_1, c_2)$, $eP$, Private key $n$, κ.
Outputs: Message $M = (m_1, m_2)$.
    1) Receive $\{C, eP\}$ from the sender.
    2) Generate $S_k = n(eP)$.
    3) Decrypt $C$ with $S_k$ and $\kappa$ by:
           $m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \mod p$
           $m_2 = (c_2 - s_1 * \kappa_1) * {s_2}^{-1} + \kappa_2 \mod p$
    4) The message $M = (m_1, m_2)$.
```

5.1 Analyzing encryption and decryption

As in Section 3, the linear Bézier curve is described in Eq. (1), and based on this equation, we created equations for encryption and decryption processes. The following explains how they created these equations, which exist in the proposed algorithms 5 and 6.

Depending on the behavior of Eq. (1), we use the message point $M = (m_1, m_2)$, the secret keys $S_k = (s_1, s_2)$ and $\kappa = (\kappa_1, \kappa_2)$ to get the equations of encryption as follows: $c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$.

Similarly, we get $c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1 \mod p$.

We developed equations for decryption processes and used Eq. (1) to find message point $M = (m_1, m_2)$:

$$c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$$

By simplification:

$$c_1 - s_2 * \kappa_2 = (m_1 - \kappa_1) * s_1$$
$$[c_1 - s_2 * \kappa_2 = (m_1 - \kappa_1) * s_1] * {s_1}^{-1}$$
$$c_1 * {s_1}^{-1} - s_2 * \kappa_2 * {s_1}^{-1} = m_1 - \kappa_1$$
$$m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1$$

We get the equations of decryption:

$$m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \mod p$$

Similarly, in case $c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1$, we get

$$m_2 = (c_2 - s_1 * \kappa_1) * {s_2}^{-1} + \kappa_2 \mod p$$

The theorem below proves that these encryption and decryption equations work:

Theorem: Let $M = (m_1, m_2)$ be the message point, $S_k = (s_1, s_2)$ and $\kappa = (\kappa_1, \kappa_2)$ be keys, then

$$c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$$
$$c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1 \mod p$$

If and only if

$$m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \mod p$$
$$m_2 = (c_2 - s_1 * \kappa_1) * {s_2}^{-1} + \kappa_2 \mod p$$

Proof: Suppose that $m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \mod p$

By substitution, we get

$$m_1 = \left((m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 - s_2 * \kappa_2\right) * {s_1}^{-1} + \kappa_1 \mod p$$

Simplifying,

$$m_1 = (m_1 * s_1 - \kappa_1 * s_1 + s_2 * \kappa_2 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \mod p$$
$$m_1 = m_1 * s_1 * {s_1}^{-1} - \kappa_1 * s_1 * {s_1}^{-1} + \kappa_1 \mod p$$
$$m_1 = m_1 * 1 - \kappa_1 * 1 + \kappa_1 \mod p$$
$$m_1 = m_1 \mod p$$

Also, suppose that $c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$

Simplifying,

$$c_1 = ((c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \mod p$$
$$c_1 = (c_1 * {s_1}^{-1} - s_2 * \kappa_2 * {s_1}^{-1}) * s_1 + s_2 * \kappa_2 \mod p$$
$$c_1 = c_1 * {s_1}^{-1} * s_1 - s_2 * \kappa_2 * {s_1}^{-1} * s_1 + s_2 * \kappa_2 \mod p$$
$$c_1 = c_1 * 1 - s_2 * \kappa_2 * 1 + s_2 * \kappa_2 \mod p$$
$$c_1 = c_1 \mod p$$

In the same way, in case of c_2 and m_2.

5.2 Implementation

We demonstrate how the proposed method can be used between sender and receiver by generating keys and encrypting and decrypting data, as illustrated below:

Setting Up the Proposed Cryptosystem Between Sender and Receiver

1) Public $E(F_{313})$: $y^2 = x^3 + 71x + 203\ mod\ 313$, where 71, 203, and 313 satisfy:

$$(4(71)^3 + 27(203)^2) \text{ mod } 313 = 223 \neq 0.$$

2) Public base point $P = (300,\ 250) \in E(F_{313})$.
3) Random secret $u = 0.85 \in [0,1]$.

Algorithm 7: Implementation of Proposed Key Production

```
Receiver's Keys Production
Inputs: $P = (300, 250) \in E(F_{313})$.
Outputs: Public key $R_k = (53, 259)$, Secret number
    $\Omega = 173$, Secret key $\kappa = (235, 161)$.
  1) Randomly, choose a private integer $n = 180$.
  2) Generate a public key $R_k$ by calculating: $R_k = nP = 180(300,250) = (53,259)$.
  3) Compute secret number by $u \in [0,1]\ mod\ p = \Omega$:
     $\Omega = 0.85\ mod\ 313 = 85 * 100^{-1}\ mod\ 313 = 173$.
  4) Generate a secret key $\kappa$ using $R_k$ by calculating $\Omega R_k = \kappa$: $\kappa = 173(53, 259) = (235, 161)$.

Sender's Keys Production
Inputs: Public key $R_k = (53, 259)$.
Outputs: Secret key $S_k = (53, 54)$, Secret number
    $\Omega = 173$, Secret key $\kappa = (235, 161)$.
  1) Randomly, choose a private integer $e = 223$.
  2) Generate the key $S_k = (s_1, s_2)$, using $e$ and $R_k$:
       $S_k = eR_k = 223(53, 259) = (53, 54)$.
  3) Compute secret number by $u \in [0,1]\ mod\ p = \Omega$:
     $\Omega = 0.85\ mod\ 313 = 85 * 100^{-1}\ mod\ 313 = 173$.
  4) Generate a secret key $\kappa$ using $R_k$ by calculating: $\Omega R_k = \kappa$: $\kappa = 173(53, 259) = (235,161)$.
```

Algorithm 8: Implementation of Proposed Encryption (The Sender)

```
Inputs: $S_k = (53, 54)$ , $\kappa = (235, 161)$ , $P = (300, 250)$,
    $e = 223$.
Outputs: Ciphertext $C = (226, 242)$, $eP = (115, 267)$.
  1) Choose message $M = (200, 300)$.
  2) Encrypt $M$ with $S_k$ and $\kappa$ by:
       $c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2\ mod\ p$
         $= (200 - 235) * 53 + 54 * 161\ mod\ 313$
         $= 266$.
       $c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1\ mod\ p$
         $= (300 - 161) * 54 + 53 * 235\ mod\ 313$
         $= 242$.
  3) Calculate $eP = 223(300, 250) = (115, 267)$.
  4) Send $\{C, eP\}$ to the receiver.
```

Algorithm 9: Implementation of Proposed Decryption (The Receiver)

```
Inputs: Ciphertext $C = (266,242)$, $eP = (115, 267)$,
    $n = 180$, $\kappa = (235, 161)$.
Outputs: Message $M = (200, 300)$.
  1) Receive $\{C, eP\}$ from the sender.
  2) Generate $S_k = n(eP)$:
       $= 180(115, 267) = (53, 54)$.
  3) Decrypt $C$ with $S_k$ and $\kappa$ by:
       $m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1\ mod\ p$
         $= (266 - 54 * 161) * 53^{-1} + 235\ mod\ 313$
         $= (266 - 54 * 161) * 189 + 235\ mod\ 313$
         $= 200$.
       $m_2 = (c_2 - s_1 * \kappa_1) * {s_2}^{-1} + \kappa_2\ mod\ p$
         $= (242 - 53 * 235) * 54^{-1} + 161\ mod\ 313$
         $= (242 - 53 * 235) * 29 + 161\ mod\ 313$
         $= 300$.
  4) The message $M = (200, 300)$.
```

6. RESULTS AND DISCUSSION

The proposed method has been applied to the software MATLAB R2018b (version 9.5.0.944444), designed for 64-bit systems on a Core i3 PC with a CPU speed of 2.00GHz and 4GB of RAM. The timing in seconds for 1000 and 3000 characters of encryption and decryption procedures for the proposed method has been compared with the original MVECC, as illustrated in Table 1.

Table 1. The processing times for MVECC and proposed algorithm in seconds

The Method	Encryption		Decryption	
	1000 Character	**3000 Character**	**1000 Character**	**3000 Character**
MVECC	0.002992	0.006163	0.095308	0.316671
Proposed Method	0.003684	0.023156	0.097800	0.341695

Table 2. The required processes are in encryption and decryption procedures for proposed algorithm

The Method	Encryption		Decryption	
Proposed Method	Mult.	4	Mult.	4
	Add.	2	Add.	2
	Sub.	2	Sub.	2
	Inv.	None	Inv.	2

Table 3. The required processes are in encryption and decryption procedures for MVECC

The Method	Encryption		Decryption	
MVECC	Mult.	2	Mult.	2
	Add.	None	Add.	None
	Sub.	None	Sub.	None
	Inv.	None	Inv.	2

The research paper presents an analysis of the number of required processes of the proposed algorithms, with a comparison to the original MVECC, and summarizes them in Tables 2 and 3. By counting the number of required processes for encrypting and decrypting procedures, (Mult.) was used for multiplication processing, (Add.) for addition processing, (Sub.) for subtraction processing, and (Inv.) for inverse processing.

Each individual operation (Mult., Add., Sub., and Inv.) takes a certain amount of time to compute. The proposed method requires more of these operations than MVECC, inherently increasing the total processing time. Encryption involves processing the entire message. As message size grows, the time difference between performing a few additional operations and many additional operations becomes more noticeable. This explains why the disparity in processing times increases significantly for the 3000-character message compared to the 1000-character message, as shown in Table 1.

To assess the proposed method's performance in terms of randomness, we exposed it to the randomness tests outlined by NIST [19]. We selected these tests based on the length of the encrypted message (1000 bits). According to established standards in randomness testing, a p-value equal to or exceeding 0.01 signifies randomness; otherwise, the test is considered failed. Analyzing the results of this randomness revealed that the proposed method produced randomness, indicating that it generated a highly random sequence of numbers, as illustrated in Table 4.

Table 4. The randomness tests of an encrypted message

Test	Results
Frequency	0.447884478264112
Block Frequency	0.814985131123346
Runs	0.737900261941727
Longest Run of Ones in Block	0.0943445489438835
Discrete Fourier Transform	0.561657715013042
Non-Overlapping Template Matching	0.987271934931365
Serial	0.138877328519302
Approximate Entropy	0.722639614775567
Cumulative Sums	0.589897787755803

6.1 Computational complexity

In this subsection of exploratory results, Encryption and decryption procedures for the proposed algorithm are computed by computational complexity and compared with the MVECC. The O-notation has proved particularly effective in assisting analysts in classifying methods according to their efficiency and in leading algorithm developers in their hunt for the "best" algorithms for major difficulties [20].

Therefore, the addition of two s-bit numbers demands s-bit operations. that illustrate as follow:

$$O_{(s)} = T(s - bit + s - bit)$$

Decimal digits are used for input of magnitude n: $T(n + n) = O(\log n)$, and $\log_2 n$ equals the number of bits in n.

Because s addition operations are required, multiplying two s-bit binary numbers require $s^2(s * s)$. It is to say:

$$T(s - bit * s - bit) = O(s^2)$$

Decimal digits are used for input of magnitude n: $T(n * n) = O(\log n)^2$, and $\log_2 n$ equals the number of bits in n.

The following is the computational complexity of MVECC and the proposed algorithms:

1) The MVECC Complexity

i. Encryption Process:

$$c_1 = m_1 * s_1 \; mod \; p$$
$$c_2 = m_2 * s_2 \; mod \; p$$

Then

$$T(c_1) = O(\log n)^2 \quad \textit{bit operation}$$
$$T(c_2) = O(\log n)^2 \quad \textit{bit operation}$$

Due to modular multiplication of encryption process.

ii. Decryption Process:

$$c_1 = m_1 * {s_1}^{-1} \; mod \; p$$
$$c_2 = m_2 * {s_2}^{-1} \; mod \; p$$

Then

$$T(c_1) = O(\log n)^2 + O(\log n)^3 \quad \textit{bit operation}$$
$$T(c_2) = O(\log n)^2 + O(\log n)^3 \quad \textit{bit operation}$$

Due to modular multiplication and modular inverse calculation of decryption process.

$T({s_1}^{-1})$ and $T({s_2}^{-1})$ equal $O(\log n)^3$ through an extension of Euclid's method.

2) The Proposed Method Complexity

iii. Encryption Process:

$$c_1 = (m_1 - \kappa_1) * s_1 + s_2 * \kappa_2 \;\; mod \; p$$
$$c_2 = (m_2 - \kappa_2) * s_2 + s_1 * \kappa_1 \;\; mod \; p$$

Then

$$T(c_1) = O(2(\log n)^2) + O(2 \log n) \quad \textit{bit operation}$$
$$T(c_2) = O(2(\log n)^2) + O(2 \log n) \quad \textit{bit operation}$$

Due to more modular multiplication and additions of encryption process.

iv. Decryption Process:

$$m_1 = (c_1 - s_2 * \kappa_2) * {s_1}^{-1} + \kappa_1 \;\; mod \; p$$
$$m_2 = (c_2 - s_1 * \kappa_1) * {s_2}^{-1} + \kappa_2 \;\; mod \; p$$

Then

$$T(c_1) = O(2(\log n)^2) + O(2 \log n) + O(\log n)^3 \; \textit{bit operation}$$
$$T(c_2) = O(2(\log n)^2) + O(2 \log n) + O(\log n)^3 \; \textit{bit operation}$$

Similar to encryption, plus modular inverse calculation of decryption process.

Because of the computational complexity, the proposed algorithms has a slightly higher complexity than MVECC due to the additional operations.

7. CONCLUSIONS

In conclusion, this research paper introduces a pioneering and enhanced algorithm designed to bolster the security and intricacy of the MVECC cryptosystem through the implementation of a two-key encryption approach. By modifying the key production, encryption, and decryption procedures of the MVECC system with the utilization of the linear Bézier equation, the proposed method successfully achieves heightened complexity and randomness according to the randomness tests and computational complexity. The proposed method has an increased the processing time due to its higher number of operations than MVECC. The research paper contributes substantially to the field of cryptographic systems by presenting an algorithm that offers an innovative avenue for enhancing both the security and complexity of the MVECC cryptosystem.

REFERENCES

[1] Marsh, D. (2005). Applied Geometry for Computer Graphics and CAD. Springer Science & Business Media.

[2] Miller, V.S. (1985). Use of elliptic curves in cryptography. In Conference on the Theory and Application of Cryptographic Techniques, Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 417-426. https://doi.org/10.1007/3-540-39799-X_31

[3] Koblitz, N. (1987). Elliptic curve cryptosystems. Mathematics of Computation, 48(177): 203-209. http://doi.org/10.1090/S0025-5718-1987-0866109-5

[4] Kataria, B., Jethva, H.B., Shinde, P.V., Banait, S.S., Shaikh, F., Ajani, S. (2023). SLDEB: Design of a secure and lightweight dynamic encryption bio-Inspired model for IoT networks. International Journal of Safety and Security Engineering, 13(2): 325-331. https://doi.org/10.18280/ijsse.130214

[5] Paar, C., Pelzl, J., Preneel, B. (2010). Understanding Cryptography: A Textbook for Students and Practitioners. Springer-Verlag, Berlin.

[6] Menezes, A. (2001). Evaluation of security level of cryptography: The elliptic curve discrete logarithm problem (ECDLP). University of Waterloo, 14.

[7] Ghadi, D.M., Adil, A.R. (2020). Improvement of Menezes-Vanstone Elliptic Curve Cryptosystem based on quadratic Bézier curve technique. Journal of Computer Science, 16(5): 715-722. http://doi.org/10.3844/jcssp.2020.715.722

[8] Srividya, B.V., Akhila, S. (2014). Novel cryptosystem based on Bezier curve using GF (p m). In International Conference on Circuits, Communication, Control and Computing, Bangalore, India, pp. 304-307. http://doi.org/10.1109/CIMCA.2014.7057811

[9] Ghadi, D.M. (2021). A novel encryption system based on PGP using elliptic curve cryptosystem and Bézier curve for secure information exchange. Journal of Physics: Conference Series. IOP Publishing, 1999(1): 012111. http://doi.org/10.1088/1742-6596/1999/1/012111

[10] Abd El-Latif, A.A., Niu, X., Wang, N. (2011). Chaotic image encryption using Bezier curve in DCT domain scrambling. In Digital Enterprise and Information Systems: International Conference, DEIS 2011, London, UK, pp. 30-41. https://doi.org/10.1007/978-3-642-22603-8_3

[11] Rahouma, K. (2006). A modified Menezes-Vanstone elliptic curve multi-Keys cryptosystem. Identity, 1(4): 6-8.

[12] Sagheer, A.M. (2012). Elliptic curves cryptographic techniques. In 2012 6th International Conference on Signal Processing and Communication Systems, Gold Coast, QLD, Australia, pp. 1-7. https://doi.org/10.1109/ICSPCS.2012.6507952

[13] Sadiq, A.T., Kadhim, N.J. (2009). Enhanced Menezes-Vanestone elliptic curves cryptosystem. Al-Nahrain Journal of Science, 12(1): 162-165.

[14] Sederberg, T.W. (2012). Computer aided geometric design. Computer Aided Geometric Design Course Notes. http://hdl.lib.byu.edu/1877/2822.

[15] Menezes, A.J., Vanstone, S.A. (1993). Elliptic curve cryptosystems and their implementation. Journal of Cryptology, 6: 209-224. http://doi.org/10.1007/BF00203817

[16] Mogollon, M. (2007). Cryptography and security services: Mechanisms and applications. IGI Global.

[17] Jyotsna K.M., Sarkar A., Bose A., Halder S. (2017). Elliptic curve cryptosystem (ECC). Review of Computer Engineering Studies, 4(2): 67-69. https://doi.org/10.18280/rces.040205

[18] Al-Saffar, N.H., Said, M.R.M., Rushdan, M. (2013). On the mathematical complexity and the time implementation of proposed variants of elliptic curves cryptosystems. International Journal of Cryptology Research, 4(1): 42-54.

[19] Bassham III, L.E., Rukhin, A.L., Soto, J., Nechvatal, J.R., Smid, M.E., Barker, E.B., Leigh, S.D., Levenson, M., Vangel, M., Banks, D.L., Heckert, N.A., Dray, J.F., Vo, S. (2010). Sp 800-22 rev. 1a. A statistical test suite for random and pseudorandom number generators for cryptographic applications. National Institute of Standards & Technology.

[20] Yan, S.Y. (2013). Number Theory for Computing. Springer Science & Business Media.