Proceedings of the Jangjeon Mathematical Society
28 (2025), No. 4, pp. 443 - 468

www.jangjeon.or.kr
http://dx.doi.org/10.17777/pjms2025.28.4.006

INNOVATIVE CRYPTOGRAPHIC TECHNIQUE USING RHOTRIX MAPPING AND ELLIPTIC CURVES OVER FINITE FIELDS

SHALINI GUPTA, RUCHI NARANG, AND KRITIKA GUPTA

Abstract. This research paper presents a novel cryptographic scheme that leverages elliptic curves, hash functions, and involutory rhotrix structures to enable secure and authenticated communication between two parties. Each party computes a specific private key using the other's public key, ensuring mutual authentication and secure key derivation. A permuted ASCII table is generated by mapping ASCII values to elliptic curve points, hashing these points, and sorting the hash values, thus ensuring security and randomness. The coordinates of specific private keys are used to construct maximum distance separable (MDS) involutory rhotrix for secure data transformation. This approach ensures secure message exchange, mutual authentication, and data integrity, making it suitable for various cryptographic applications. Furthermore, we implement this scheme in Python, and through rigorous testing, we measure the time required for encryption and decryption, demonstrating the efficiency and practical feasibility of our cryptographic approach.

2000 Mathematics Subject Classification. 12E20; 94A60.

Keywords and phrases. Elliptic curve cryptography, rhotrices.

SUBMISSION DATE: 30.07.2024

1. Introduction

In the current digital era, ensuring secure data transmission is crucial as communication technology continues to evolve. Text encryption plays a vital role in maintaining data confidentiality during transmission. Elliptic Curve Cryptography (ECC) has emerged as a cornerstone of modern cryptographic systems, particularly in text encryption and mapping schemes. ECC is based on the mathematical properties of elliptic curves over finite fields, offering a powerful framework for encryption and data integrity. One of the primary advantages of ECC in text encryption is its ability to provide robust security with much smaller key sizes compared

to traditional algorithms like Rivest-Shamir-Adleman (RSA). This efficiency is particularly important for applications with limited resources, such as mobile devices and Internet of Things (IoT) devices.

In the realm of research on ECC-based mapping and text encryption schemes, researchers explore various approaches to enhance the security, efficiency, and applicability of these cryptographic methods. Their goal is to develop algorithms and protocols that are not only resilient against a wide range of cryptographic attacks but also efficient in terms of computational resources, making them suitable for diverse applications. Optimization of efficiency and performance is critical, with a particular emphasis on creating ECC schemes that require minimal computation and memory-crucial for resource-constrained environments like IoT devices.

A cornerstone of this research is security analysis, involving rigorous evaluation of ECC-based schemes' resistance to attacks, including the increasing threat of quantum attacks. Scalability is another key aspect, as researchers work on applying ECC securely in large-scale systems, distributed environments, and cloud computing. Additionally, privacy and anonymity are significant considerations, leading to investigations into ECC protocols that enable anonymous communication and protect user privacy through techniques like homomorphic encryption. Effective key management strategies are also vital, addressing challenges related to key generation, distribution, and revocation to maintain secure ECC systems. The benefits of ECC-based text encryption and mapping schemes are substantial. ECC offers robust security based on the elliptic curve discrete logarithm problem, and its efficiency allows for faster computations and reduced resource requirements. This efficiency enhances secure communication protocols, enabling quicker and more secure data exchanges. However, there are challenges to consider. Proper implementation of ECC requires a deep understanding of its mathematical foundations, as errors can lead to vulnerabilities.

The technical aspects of mapping text characters to points on an elliptic curve involve encoding, finite field arithmetic, point compression, error checking, security measures, padding, compression techniques, conversion back to text, performance optimization, cryptographic hashing, resistance to known attacks, and implementation security. Accurate mapping of text characters to points on an elliptic curve is crucial for ensuring security and avoiding vulnerabilities. This process must be executed with precision and appropriate security measures to prevent weaknesses. Key management is another critical aspect of ECC-based schemes. Secure key storage and exchange protocols are essential for

maintaining security. Traditional ECC-based text encryption schemes often rely on secure channels or protocols to share the code table during initialization. However, these methods can be vulnerable to interception or man-in-the-middle attacks, especially when a secure channel is not available. Adversaries may exploit weaknesses in transmission or compromise the code table, leading to decryption errors or unauthorized access to sensitive information.

1.1. **Objectives of the present work.** The primary objectives of the present work are

(1) To perform a comprehensive background study and enhance the literature review on secure and efficient authenticated encryption and mapping schemes using ECC.
(2) To generate a permuted ASCII table using the cryptographic properties of elliptic curves and hash functions.
(3) To propose a novel rhotrix mapping, which maps the characters of a message to points on an elliptic curve. This approach enhances cryptographic security by generating and mapping the points on the elliptic curve. By using elliptic curve coordinates as keys, it ensures robustness. These coordinates act as seed values for a random number generator, ensuring reproducibility while strengthening resistance against unauthorized access. Furthermore, it eliminates the traditional need for sharing a code table between sender and receiver, thus reducing communication overhead.
(4) Additionally, the proposed methodology will be tested and validated across various performance metrics including encryption time, decryption time, and resilience against diverse attacks.

1.2. **Organisation of the present work.** The rest of the work is organized as follows: Section 2 enlightens related works. Section 3 illustrates preliminaries necessary for understanding the paper. Section 4 highlights the proposed methodology. Section 5 provides the algorithm of the proposed encryption scheme. Section 6 presents experimentation, results, and performance analysis of the proposed scheme. Finally, Section 7 concludes the paper with future scope.

2. Related work

In cryptography, various techniques ensure secure communication between sender and receiver, restricting message visibility to authorized parties only. Significant advancements in elliptic curve cryptography

(ECC) have seen widespread implementation across encryption, decryption, digital signatures, authentication, and key agreement [1, 2, 3, 9, 29, 33, 41, 44]. This literature review comprehensively examines different encryption models, detailing their strengths, weaknesses, and potential applications. Hankerson et al. [11] introduced a method which maps each character of a message to points on an elliptic curve by multiplying the generator point G with the ASCII value of the character. However, this approach was vulnerable to frequency analysis attacks due to repetitive encrypted characters in the ciphertext. Amounas and El Kinani [13] proposed a permutation scheme using a non-singular matrix to prevent such attacks, encrypting the mapped message with ECC. Their method, however, faced vulnerability to chosen plaintext attacks if the matrix used for multiplication was discovered.

Paragas et al. [22] developed a Hill cipher chain algorithm based on modified Hill cipher principles, employing XOR and shift operations for block encryption alongside RADIX 64 encoding. Bh et al. [36] utilized the Koblitz technique to probabilistically map messages onto an elliptic curve, susceptible to collision attacks. King [8] presented a deterministic approach based on binary string interpretation to map messages onto elliptic curves, ensuring each message element is translated into a secure subgroup point. Muthukuru and Sathyanarayana [21] proposed a mapping technique using XOR operations to map fixed-length strings to elliptic curve points, although it was vulnerable to man-in-the-middle and chosen plaintext attacks. Almajed et al. [15] emphasized the role of padding bits in ECC mapping, investigating optimal sizes to enhance performance and security. Kumar et al. [12] introduced a method for secure and authenticated communication using ECC over finite fields, where plaintext characters associated with elliptic curve points were encrypted using shared keys.

Almajed and Almogren [17] proposed an authenticated encoding and mapping scheme leveraging ECC metrics, demonstrating resilience against attacks and optimized performance through appropriate padding sizes and reduced encoding and decoding operations. However, their use of Cipher Block Chaining (CBC) mode hindered parallel processing across multiple processors or cores. These studies collectively illustrate the evolving landscape of ECC in cryptographic applications, highlighting various methodologies, their vulnerabilities, and their potential contributions to enhance secure communication protocols.

Kamalakannan and Tamilselvan [43] devised an encryption technique where each character of the plaintext message was converted to its corresponding ASCII value, which was then mapped to specific affine points

on an elliptic curve. These points underwent further modification using a matrix mapping scheme and were ultimately secured using the ElGamal encryption method. Keerthi and Surendiran [24] proposed a scheme where the plaintext message was first converted into its ASCII values, then transformed into the hexadecimal format and organized into specific groups based on the input size. The order of these hexadecimal values was reversed, and the final ciphertext was generated through scalar multiplication on an elliptic curve. Naji et al. [27] introduced an encryption process involving the conversion of each plaintext character into a 16-bit decimal representation, followed by segmentation into larger integer segments for encryption using a shared key. This method substituted characters with affine points, eliminating the need for costly mapping techniques.

Nadeem et al. [30] presented an efficient algorithm designed to protect data from unauthorized access and misuse, particularly in cloud computing environments. Their approach involved converting plaintext to ASCII byte values, generating keys using a Non-Deterministic Random Bit Generator (NRBG), XNORing keys with plaintext bits, and applying a matrix cipher encryption algorithm. This ensured that unique keys derived from plaintext were used for decryption, thereby enhancing data security across different stages to produce ciphertext. Tiwari and Kim [16] introduced a novel approach inspired by DNA-based ECC, where DNA genome sequences were utilized to assign values to various character sets within a message. This mapping employed randomized character assignments based on pseudo-random data, crucial for securing encrypted messages. However, the scheme did not address the encoding process converting the message into numerical values for mapping to DNA sequences, potentially exposing it to specific encryption attacks.

Cahyono et al. [19] developed encryption and decryption algorithms using Type IVa MP-Wavelet analysis and synthesis techniques. Their encryption key strategy involves ensuring the total channel count across all levels exceeds the number of characters in the plaintext, enhancing security against brute-force attacks. The decryption key incorporates this encryption key and a sequence derived from binary encoding of detail components, further fortifying resistance to cryptographic attacks. This algorithm is noted for its efficiency due to its linear complexity relative to the plaintext's character count. Sattar et al. [25] introduced a novel text encryption scheme combining elliptic curve cryptography with max-plus algebra-based wavelet transform, aimed at bolstering security and efficiency in data protection.

Fu et al. [20] focused on enhancing security in Hyperledger Fabric by replacing the Elliptic Curve Digital Signature Algorithm (ECDSA) with the SM2 algorithm. Their approach optimizes signature processes, reduces time complexity, and enhances overall system performance by integrating SM2 into Fabric's Blockchain Cryptographic Service Provider (BCCSP) module. Gupta et al. [7] explored Business-to-Business (B2B) methods in healthcare, addressing challenges in managing extensive smart device data. Their study aimed to improve data transmission, patient access, and care quality while emphasizing security concerns and proposed techniques to safeguard sensitive healthcare data. Zhao et al. [32] introduced the Lightweight Reputation-Based Consensus Mechanism tailored for securing Industrial IoT (IIoT) data. Sharma et al. [34, 35] proposed a new text encryption scheme, and image encryption scheme using mapping of the characters to a point on an elliptic curve.
Genç and Afacan [45] proposed an innovative approach converting each character in a message into its hexadecimal Unicode representation. This method differs from ASCII by encompassing a broader range of characters and employing hexadecimal values spanning from one to six digits. Singh and Singh [26] presented a technique mapping message characters to affine points on an elliptic curve through ASCII value grouping and employing the Big Integer function to derive mapped points.
Sengupta and Ray [6] explored various message mapping schemes within elliptic curve cryptography, discussing vulnerabilities to cryptanalysis and proposing guidelines for effective and secure message mapping. They introduced a novel scheme designed to resist frequency analysis and other forms of cryptanalysis.
Das and Giri [37] proposed two novel algorithms for converting input messages into elliptic curve points, which reduces the communication and computational costs while improving performance, particularly for handling large text inputs. Azhar et al. [40] introduced a novel encryption scheme using the Pell sequence and elliptic curves. Initially, plaintext was scrambled through cyclic shifts, followed by obscuring the text using the Pell sequence, a weight function, and binary encoding. Further confusion was added through permutations on elliptic curves, ensuring resilience against key sensitivity and statistical attacks. Murtaza et al. [14] proposed a scheme to develop a fast and secure S-box generator for lightweight cryptography, utilizing small elliptic curves and binary sequences for text encryption.
Ullah et al. [18] presented efficient S-box and pseudo-random number generators based on ordered Mordell elliptic curves, demonstrating their ability to produce numerous distinct and cryptographically strong

S-boxes and random sequences with low time and space complexity. Rihartanto et al. [39] devised a text encryption scheme incorporating bit-based cube rotation for achieving confusion and diffusion, resulting in ciphertext with high avalanche effect and low correlation coefficient, ensuring effective encryption independent of original content. Kordov [23] introduced a cryptographic system employing a pseudorandom generator derived from two chaotic maps, demonstrating its resistance against various attacks through rigorous cryptographic analysis. Ataş and Güler [31] proposed an encryption-decryption method using the fractional-order Rössler chaotic system for image, sound, and text data, enhancing security through Master-Slave synchronization of the chaotic system. Ivanova et al. [42] developed a text encryption algorithm utilizing two Clifford attractors to enhance message transfer security, validated through rigorous security testing.

In the realm of message encryption into elliptic curve points, traditional methods often rely on code tables or keys shared between sender and receiver, introducing communication overhead and security vulnerabilities. Addressing these challenges, the proposed scheme simplifies encryption by directly mapping message characters to elliptic curve points using rhotrices. This method ensures data integrity and authenticity by leveraging elliptic curve coordinates as the basis for a predictable random number generator. Additionally, the use of involutory rhotrices enhances the security, reinforcing the encrypted data's security. Thus, this method offers a robust alternative in elliptic curve cryptography for secure communication without the complexities associated with traditional encryption methods.

3. Preliminaries

3.1. **Rhotrix** [5]**.** Ajibade defined a 3−dimensional rhotrix, which is in some way between 2×2 and 3×3 matrices as follows:

$$R = \left\langle \begin{array}{ccc} & a & \\ b & h(R) & d \\ & e & \end{array} \right\rangle$$

where $h(R)$ represents the heart of rhotrix and $a, b, d, e \in R$, denote components of an odd-dimensional rhotrix. This notion was expanded

upon by Mohammed et al. [4] to encompass a rhotrix of order n.

$$R_n = \left\langle \begin{array}{ccccccc} \cdots & \cdots & \cdots & a_1 & \cdots & \cdots & \cdots \\ \cdots & \cdots & a_2 & a_3 & a_4 & \cdots & \cdots \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ a_{\frac{t-n+2}{2}} & \cdots & \cdots & a_{\frac{t+1}{2}} & \cdots & \cdots & a_{\frac{t+n}{2}} \\ \cdots & \cdots & \cdots & \cdots & \cdots & \cdots & \cdots \\ \cdots & \cdots & a_{t-3} & a_{t-2} & a_{t-1} & \cdots & \cdots \\ \cdots & \cdots & \cdots & a_t & \cdots & \cdots & \cdots \end{array} \right\rangle$$

where $t = \frac{n^2+1}{2}$, $a_i \in R$. In this context $h(R) = \frac{a_{t+1}}{2}$, serves as the heart of R_n.
Generally, a rhotrix is denoted as $R = \langle h(R), a_i \rangle$, $1 \leq i \leq t$, $i \neq \frac{t+1}{2}$.

3.2. **Finite Field** [38]**.** A non-empty finite set F is defined as a finite field if it is an abelian group under both addition and multiplication.In computing, positive integers are typically represented as n-bit words, where n can be $8, 16, 32, 64$, and so forth. Consequently, the maximum range of integers is $2^n - 1$. A Galios field, denoted as $GF(2^n)$ having a finite field of 2^n elements while $GF\{p\}$ encompasses the set Z_p, where p represents the largest prime number less than 2^n. The elements within this field consist of n-bit words, as elaborated in the works of Hun and Yen [10] and Hell and Johnson [28].

3.3. **Elliptic Curve.** In elliptic curve cryptography, a specific form of elliptic curve defined over a finite field $\mathbb{F}_p$ is of interest. This field is defined by a prime number p. The elliptic group modulo p, denoted as $E_p(a, b)$, is particularly relevant for cryptographic applications. Equation (1) establishes the condition for selecting an appropriate elliptic curve:

$$4a^3 + 27b^2 \pmod{p} \not\equiv 0 \tag{1}$$

Here, a and b are two non-negative integers less than p. The notation $E_p(a, b)$ signifies the elliptic group modulo p, whose elements (x, y) are pairs of non-negative integers less than p. Equation (2) represents the general form of an elliptic curve:

$$y^2 = x^3 + ax + b \tag{2}$$

In summary, elliptic curve cryptography involves selecting elliptic curves satisfying equation (1) over the finite field $\mathbb{F}_p$, where p is a prime. The general form of such curves is given by equation (2).
Two fundamental group operations performed on elliptic curves over prime fields $GF(p)$ are defined as follows:

(1) Addition: If $a,\ b\ \in GF(p)$ then, $a\ +\ b\ =\ r$, where $r\ \in\ GF(p)$ and equals $(a\ +\ b)$modulo p.
(2) Multiplication: If $a,\ b\ \in GF(p)$ then, $a\ *\ b\ =\ r$, where $r\ \in\ GF(p)$ and equals $(a\ *\ b)$ modulo p.
(3) Inversion: If a is a non-zero element in $GF(p)$, then $a^{-1}\ =\ c\ \in\ GF(p)$ such that $a\ *\ c\ =\ 1$ modulo p.

For elliptic curves over prime fields $GF(p)$ with $p \geq 3$, the parameters a and b must satisfy the condition $4a^3 + 27b^2\ \not\equiv\ 0\ (\text{mod}\ p)$. Points on an elliptic curve are typically represented using affine coordinates. If we consider a point $P(x, y)$ on an elliptic curve, then the negative of a point P is its reflection in the $x-$axis, i.e., $-P\ =\ (x, -y)$. For addition, consider $P_1\,(x_1, y_1)$ and $P_2(x_2, y_2)$ are two points on an elliptic curve, then the addition of two points is given by

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

where,

$$x_3 = \lambda^2 - x_1 - x_2,$$
$$y_3 = \lambda\,(x_1 - x_3) - y_1$$

and

$$\lambda = \frac{y_2 - y_1}{x_2 - x_1}.$$

Doubling of a point $P\,(x_1, y_1)$, having coordinates (x_2, y_2) is given by formula

$$x_2 = \lambda^2 - 2x_1,$$
$$y_2 = \lambda(x_1 - x_2) - y_1$$

and

$$\lambda\ =\ \frac{3{x_1}^2 + a}{2y_1}.$$

3.4. **ECC Encryption Scheme** [26]**.** Suppose Alice and Bob are two parties who want to communicate a message securely. They agree on using a specific elliptic curve and a generator point G.

(1) Alice chooses her private key n_A.
(2) Bob chooses his private key n_B.
(3) Alice's public key is generated as $P_A = n_A G$.
(4) Bob's public key is generated as $P_B = n_B G$.

When Alice wants to send a message M to Bob, she uses Bob's public key and a random integer t for encryption. Alice creates the ciphertext $T = \{tG,\ M + tP_B\}$. Different values of the random integer t will generate different ciphertexts for the same message, making decryption without

the private key difficult. Bob can decrypt the message using his private key. He subtracts $n_B(tG)$ from $M + tP_B$ to retrieve the original message M.

3.5. **Elliptic Curve Diffie-Hellman Key Exchange** [11]**.** Let private keys of Alice and Bob be m and n respectively. Let G be the generator of elliptic curve. Alice's public key is mG and Bob's public key is nG. These public keys are exchanged over an open channel. Alice then multiplies her private key with Bob's public key, and Bob multiplies his private key with Alice's public key. Both Alice and Bob will obtain the same result. This method of key exchange between users is known as the Elliptic Curve Diffie-Hellman (ECDH) key exchange.

4. Proposed Methodology

In the proposed scheme, we present a novel method for creating a customized permuted ASCII table that introduces advanced cryptographic security through the use of hashing. Each ASCII value is mapped to a point on an elliptic curve, hashed using a cryptographic hash function like SHA-256, and sorted to generate the permuted table. Additionally, we incorporate specific private keys to enhance security. Furthermore, we introduce an additional layer of security by incorporating specific private keys. This specific private key serves a dual purpose: first, it provides the receiver with a robust means of verifying that the ciphertext originates exclusively from the legitimate sender. Second, this specific private key is employed in generating the elements of MDS involutory rhotrix.

4.1. **System Model.** Suppose Alice and Bob are two communicating parties and they want to share a message between them. Both, Alice and Bob agree on a common elliptic curve with prime order generator point T.

(1) Alice chooses a random number n_A in such a way that it lies between $[1, l-1]$, where l is the order of generator point T. Alice keeps n_A as her private key.
(2) Alice computes her public key as $P_A = n_A T$.
(3) Bob selects a large random number n_B in such a way that it lies between $[1, l-1]$. He keeps n_B as his private key.
(4) Public key of Bob is $P_B = n_B T$. The public keys of both the parties are published.
(5) Alice calculates her specific private key, $A_S = n_A P_B = (S_x, S_y)$.
(6) Bob calculates his specific private key, $B_S = n_B P_A = (S_x, S_y)$.
(7) Alice selects a random positive integer k of 128 bits.

4.2. **ASCII Table Generation.** Generating a permuted ASCII table involves a sophisticated process that leverages the cryptographic properties of elliptic curves and hash functions. By mapping each ASCII value (ranging from 0 to 255) to a unique point on an elliptic curve, and subsequently hashing the coordinates of these points, we create a deterministic permutation of the ASCII table. This approach ensures a high level of security and randomness, making it suitable for cryptographic applications and secure data processing. In this section, we highlight the steps of generating these elliptic curve points, hashing them, and sorting the hashed values to obtain a permuted ASCII table.

(1) **Hashing Elliptic Curve Points**
Each ASCII value (0 to 255) is mapped to a point on the elliptic curve using scalar multiplication with a generator point T. The coordinates (x, y) of each elliptic curve point are hashed using a cryptographic hash function like SHA-256. This hashing step converts the coordinates into fixed-length hexadecimal strings (hash digests).

(2) **Sorting by Hashed Values**
The hashed values (hexadecimal strings) are sorted in ascending order. Sorting is based on the lexicographical order of hexadecimal strings. Each hexadecimal character ('0'-'9' then 'a'-'f') is compared in sequence to determine the order. Sorting ensures that the permuted ASCII values are arranged in a deterministic order determined by the hashed representations of their corresponding elliptic curve points.

(3) **Extracting Permuted ASCII Values**
After sorting the hashed values, the permuted ASCII values are extracted based on the sorted order of these hash digests. The index of each hashed value in the sorted list determines the position of the corresponding ASCII value in the permuted sequence.

Illustration
Let's say we have hashed values (in hexadecimal) of elliptic curve points mapped from ASCII values:

1. ASCII 'A' (decimal 65) maps to point 65 T, which hashes to $'2a3b...c4d'$.
2. ASCII 'B' (decimal 66) maps to point 66 T, which hashes to $'1f2e...a5b'$.
3. ASCII 'C' (decimal 67) maps to point 67 T, which hashes to $'4c1d...b2a'$.

If after hashing and sorting, the order of these hashed values is:

- '1f2e...a5b' (for ASCII 'B')
- '2a3b...c4d' (for ASCII 'A')

- '4c1d...b2a' (for ASCII 'C')

Then, the permuted ASCII values would be in the order: 'B', 'A', and 'C'.
The permuted ASCII values are arranged in an order determined by the lexicographical sorting of their hashed representations. This ensures a consistent and deterministic method of permuting ASCII values based on the cryptographic properties of the elliptic curve and hash function used.
The permuted ASCII table is given in Annexure - A.

4.3. **Construction of MDS Involutory Rhotrix.** Consider a 3-dimensional involutory rhotrix,

$$I_3 = \left\langle \begin{array}{ccc} & a_{11} & \\ a_{21} & 1 & a_{12} \\ & a_{22} & \end{array} \right\rangle.$$

Since this is an involutory rhotrix, it must satisfy following properties:

$$a_{22} = -a_{11}$$
$$a_{12}a_{21} = 1 - {a_{11}}^2.$$

The value of $1-{a_{11}}^2$ is split into two factors, with one factor designated as a_{12} and the other as a_{21}. To calculate a_{11}, we XOR the first 64 binary bits of S_y with the last 64 binary bits of S_y and then convert it to decimal form which only Alice and Bob can determine using their private keys and public keys of each other. This method allows us to assign all the values and construct an involutory rhotrix, ensuring it remains private to Alice and Bob. For longer messages, we can use more than one involutory rhotrix.

4.4. **Mapping Scheme.** In our novel mapping scheme for character-to-point transformation on the elliptic curve, we introduce a streamlined approach that eliminates the need for a shared code table and enables the simultaneous mapping of fifteen characters to a single elliptic curve point.
Here is the process:

(1) Alice breaks the message into blocks of fifteen characters each: $m_1, m_2, m_3, \ldots, m_n$. If the last block has fewer than fifteen characters, it will be padded with zero characters.
(2) Alice converts the first block m_1 of fifteen characters into its ASCII values. Each ASCII value is then mapped to its corresponding value in the permuted ASCII table. These permuted

ASCII values are converted into their binary representations, forming a 128-bit binary string.

(3) Alice converts the x and $y-$coordinates of specific private key into binary bits of length 128 characters of both coordinates.
(4) Alice writes random integer k as 128 bits and XOR all 128 bits of random integer k, first block of message m_1, S_x and S_y. She denotes the resultant by x_1.
(5) Alice substitutes this point on the elliptic curve to get $y-$coordinate. Alice gets first mapping point. In case $y-$coordinate does not exist at point x_1, then by adding 1 to x_1, Alice again tries to find $y-$coordinate, and will continue this process till Alice gets the $y-$coordinate. She denotes the first mapping point as $\left(M_{x_1}, M_{y_1}\right)$.
(6) Rhotrix mapping: For rhotrix mapping, Alice assigns row-wise entries, first mapping point as first entry of the rhotrix, and second mapping point as second entry and so on to get the rhotrix R_3 (say). For the construction of first element of (MDS) involutory rhotrix, she performs XOR operation between the first 64 binary bits of S_y and last 64 binary bits of S_y. Subsequently, converting it to decimal form, the resulting element will be considered as first element of involutory rhotrix and remaining elements will be calculated according to the algorithm that we have defined to construct involutory rhotrix I_3.
(7) Rhotrix multiplication: Now Alice multiplies rhotrix R_3 with involutory rhotrix I_3 to get M. Denoting each entry of M by sequence of points.

4.5. **Encryption.** After mapping, each message point is then encrypted to a pair of cipher points T_1 and T_2.

(1) Alice will use the same random number k to compute T_1 as:

$$T_1 = kT.$$

(2) Computes Cipher point T_2 as:

$$T_2 = M + kP_B.$$

4.6. **Authentication.** Adding digital signature authentication ensures the integrity and authenticity of the encryption. It guarantees that the message has not been tampered with during transmission, providing an additional layer of security. Alice performs the following steps to achieve authentication.

(1) Calculates hash value h_C using the following algorithm

$$T = kA_S = (u, v).$$

As u and v both have 128 bits, representing the first 64 bits and last 64 bits of u as u_1 and u_2 respectively, and the first 64 bits and last 64 bits of v as v_1 and v_2 respectively.

(2) Computes S

$$S = u_1 \oplus u_2 \oplus v_1 \oplus v_2.$$
$$h_S = SHA_{256}(S).$$

(3) Similarly, computes hash value H using hash h_C of cipher text T_2.

$$h_C = SHA_{256}(C).$$
$$H = h_S \oplus h_C.$$

(4) Calculates R by performing XOR operation between S_x and S_y as follows:

$$R = S_x \oplus S_y.$$

(5) Calculates k' using the parameter k as follows:

$$k' = k \oplus R.$$

(6) Evaluates the digital signature (V, W) as follows:

$$V = SHA_{256}(H),$$
$$W = (k' + V)(\mathrm{mod} p).$$

(7) Sends the digital signature (V, W) and ciphered text T_2 to Bob.

4.7. **Verification.** For verification Bob will proceed as follows:

(1) Calculates R as

$$\begin{aligned} R &= S_x \oplus S_y. \\ k &= (W - V) \oplus R \\ &= (k' + V - V) \oplus R \\ &= k' \oplus R \\ &= k. \end{aligned}$$

(2) Bob calculates hash value h_s using same algorithm,

$$T = kB_S = (u, v).$$

As u and v both are of 128 bits, representing first 64 bits and last 64 bits of u as u_1 and u_2 respectively and first 64 bits and last 64 bits of v as v_1 and v_2 respectively.

(3) Here,

$$S = u_1 \oplus u_2 \oplus v_1 \oplus v_2.$$
$$h_S = SHA_{256}(S).$$

(4) Similarly, computs hash value H using hash value h_C of cipher text T_2.

$$h_C = SHA_{256}(C).$$
$$H = h_S \oplus h_C.$$

(5) Calculates V from computed hash value H. If,

$$V = SHA_{256}(H),$$

then the signature is verified.
Hence,

$$V = V.$$

4.8. **Decryption.** When Bob gets the cipher text T_2 for each message point M, he decrypts the message using the expression

$$\begin{aligned} T_2 - n_B T_1 &= M + kP_B - n_B T_1 \\ &= M + kP_B - n_B kT \\ &= M + kn_B T - n_B kT \\ &= M. \end{aligned}$$

To decode and retrieve the original characters of the message corresponding to the point M, Bob employs the following steps:

(1) Bob calculates R_3 using following step:

$$R_3 = I_3 M.$$

(2) From R_3, Bob gets all the mapping points. For the retrieval of first block of original message, he will XOR random points k, M_{x_1}, S_x, and S_y.
(3) After this, m_1 will be changed to decimal form, then from the permuted ASCII table original value will be retrieved and the first block of the original message will be decrypted.
(4) Similarly, Bob will decrypt all the blocks of message and hence complete message will be retrieved.

5. Encryption Algorithm of The Proposed Scheme Using ECC

Key Generation:

(1) Alice chooses private key n_A.
(2) Bob chooses private key n_B.
(3) Public key generated by Alice: $P_A = n_A T$.
(4) Public key generated by Bob: $P_B = n_B T$.
(5) Specific private key generated by Alice: $A_S = n_A P_B = (S_x, S_y)$.
(6) Specific private key generated by Bob: $B_S = n_B P_A = (S_x, S_y)$.
(7) Alice selects a random integer k such that $2 \leq k \leq l - 1$.
(8) Both Alice and Bob compute the following parameters:

$$kT = (u, v), \quad g = a_1 \oplus a_2 \oplus a_3 \oplus a_4, \quad h = a_5 \oplus a_6 \oplus a_7 \oplus a_8,$$

$$i = a_9 \oplus a_{10} \oplus a_{11} \oplus a_{12}, \quad j = a_{13} \oplus a_{14} \oplus a_{15} \oplus a_{16}.$$

Encryption Process (Alice):

(1) Constructs a ASCII permuted table.
(2) Constructs an (MDS) involutory rhotrix.
(3) Converts the message into blocks of fifteen characters each: $m_1, m_2, m_3, \ldots, m_n$.
(4) Converts the first block of message m_1 to its corresponding value in the permuted ASCII table, then converts to the binary bits of length 128 characters.
(5) Converts the random integer k, the first block of message m_1, $x-$coordinate S_x and $y-$coordinate S_y of specific private keys into binary bits of length 128 characters.
(6) XOR all 128 bits of random integer k, first block of message m_1, S_x, and S_y.
(7) Computes first mapping point $\left(M_{x_1}, M_{y_1}\right)$.
(8) Computes rhotrix R_3.
(9) $R_3 = I_3 M$.
(10) Compute $T_1 = kT$.
(11) Computes cipher point $T_2 = M + kP_B$.

Decryption Process (Bob):

(1) Constructs an ASCII permuted table using the same procedure.
(2) Constructs an MDS involutory rhotrix using same algorithm.
(3) Calculates $T_1 = kT$.
(4) Computes $T_2 - n_B T_1$ to retrieve M.
(5) Evaluates $R_3 = I_3 M$.
(6) Computes $M_{x1} \oplus k \oplus S_x \oplus S_y = m_1$.
(7) Recovers the original value of m_1 from the permuted ASCII table.

6. Experimentation, Results, and Analysis

6.1. **Experimental Setup.** The proposed scheme was implemented using Intel Core i7 9th Generation CPU 2.00 GHz, 16 GB RAM, 1 TB SSD using Python 3 64-bit software.

6.2. **Results and Analysis.** In this section, we present the empirical results obtained from implementing our proposed encryption scheme in Python. We have measured the time taken for encryption and decryption processes to evaluate the performance and efficiency of the algorithm. These measurements were conducted to provide a comprehensive analysis of the computational overhead introduced by our cryptographic approach.

6.2.1. *Encryption and Decryption Time.* To evaluate the performance of the proposed scheme, we conducted several tests to measure the time taken for both encryption and decryption. The times recorded provide insight into the computational efficiency of the algorithm under different conditions. Table 1 below presents the detailed results of our performance tests and results are illustrated in Figure 1.

Table 1. Encryption and Decryption Time of Proposed Scheme

No. of characters in the message	Encryption Time(sec)	Decryption Time(sec)
40	0.000504	0.000282
500	0.006788	0.003719
1000	0.010684	0.004864
2000	0.017564	0.009867

These results indicate that the proposed scheme operates within acceptable time frames, making it suitable for practical applications. The encryption and decryption times are consistently low, demonstrating the efficiency of the algorithm.

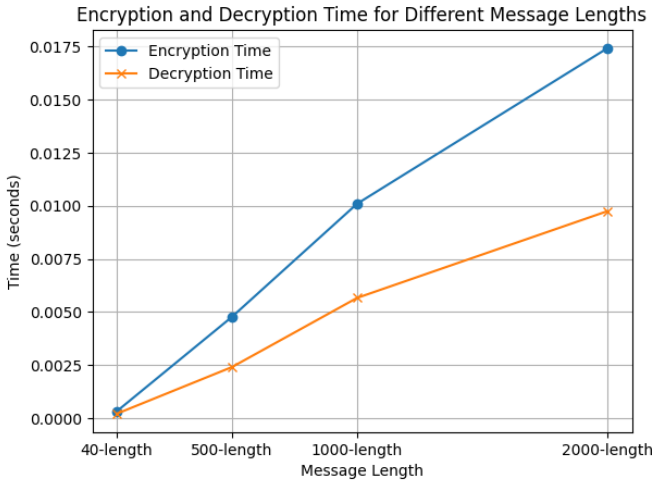

FIGURE 1. Encryption Decryption Time for Different Message Length

6.2.2. *Timing Attack.* A timing attack is a form of side-channel attack where an adversary attempts to compromise ciphertext by analyzing the varying response times for different messages. Figure 2 illustrates the encryption time for different data sets of the same size (100 characters) over multiple trials. Each line represents a different character set (A, B, C, and D), showing the variations in encryption time across four trials. From the graph, it is evident that the execution time is different for different same size data. This demonstrates the algorithm's efficiency and stability in handling data of varying content but identical size.

These results suggest that the proposed encryption scheme is robust and performs efficiently across different types of data, making it suitable for practical applications where performance is a critical factor.

6.2.3. *Time Complexity.* The most significant operations are the scalar multiplications in ECC (steps 3, 4, 5, 6, 8, 18, and 22), each with a time complexity of $O(\log n)$, where n represents the size of the scalar (private key or random integer). Splitting the message into blocks (step 11) has a linear time complexity $O(m)$, where m is the length of the message. Thus, the overall time complexity of the algorithm is dominated by the scalar multiplications and the message length, resulting in $O(m + \log n)$. The time complexity $O(m + \log n)$ reflects an efficient and practical balance between cryptographic security and performance,

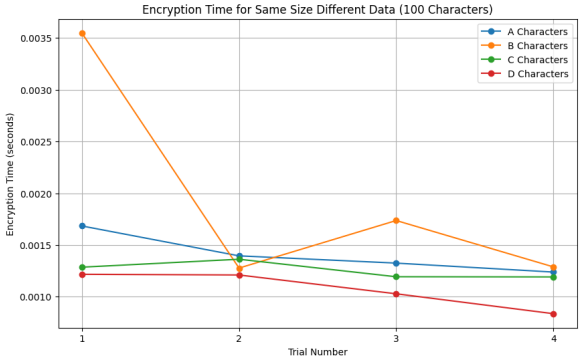

FIGURE 2. Encryption Time for Same Size Different Data

making the algorithm suitable for modern cryptographic applications.

6.3. **Security Analysis.** The security of the proposed encryption scheme is paramount to ensuring the confidentiality, integrity, and authenticity of the communicated messages between Alice and Bob. This section provides a comprehensive analysis of the security features inherent in our scheme. We will examine the robustness and resilience of the system against various potential attacks.

6.3.1. *Keyspace Analysis.* Keyspace analysis is crucial in assessing the security of a cryptographic encryption scheme. The size of the key significantly impacts security, and while the algorithm used is commonly known, opting for a larger key is generally a prudent choice. However, increasing key size also adds to the computational load. The Elliptic Curve Discrete Logarithm Problem (ECDLP), employed in Elliptic Curve Cryptography (ECC), is computationally difficult, enabling users to utilize smaller keys compared to traditional cryptographic methods without compromising security. Our implementation uses a 256-bit key length, which is robust enough to withstand basic attacks. For enhanced security, we can increase the key length.

6.3.2. *Known Plaintext Attack.* If an attacker knows the plaintext, the encryption method, and some plaintext-ciphertext pairs, our encryption scheme still resists known plaintext attacks. This is because it uses an

MDS involutory rhotrix, which only the sender and receiver can calculate using their private keys. This ensures that even when the same message is encrypted, the resulting ciphertexts cannot be generated. Therefore, known-plaintext attacks won't work, as the involutory rhotrix prevents the attacker from spotting patterns or links between the plaintext and ciphertext to decode the message.

6.3.3. *Known Ciphertext Attack.* If an attacker knows the ciphertext and the encryption scheme but doesn't have the receiver's private key, decrypting the message through brute force is extremely time-consuming due to the very large key size. This makes a brute force attack take years, making a ciphertext-only attack practically impossible. As a result, even if the attacker somehow manages to decrypt the message, the information would likely be outdated and irrelevant. Therefore, a ciphertext-only attack is not a feasible approach in this scenario.

6.3.4. *Chosen Plaintext Attack.* The use of an algorithm that employs elliptic curve coordinates and private keys to calculate all entries of the involutory rhotrix adds a layer of unpredictability to the mapping process. Even if an attacker can choose plaintexts for encryption, predicting the mapping to specific points on the curve remains challenging.

6.3.5. *Chosen Ciphertext Attack.* The proposed encryption scheme is resistant to chosen ciphertext attacks (CCA) due to several security measures. The use of involutory rhotrix in the decryption process further ensures the security of the ciphertext, reducing the risk of plaintext exposure even if an attack succeeds. Constructing an (MDS) involutory rhotrix using secure key agreement protocols like ECDH guarantees that each entry is unique, making it difficult for attackers to compromise other keys. This encryption method also makes it hard for attackers to deduce the relationship between changes in ciphertext and plaintext. Additionally, the scheme uses secure decryption procedures that can detect and reject altered ciphertexts, enhancing its defense against potential CCA.

6.3.6. *Collision Attack.* A collision attack occurs when two different inputs produce the same hash value or ciphertext after being processed by a cryptographic hash function or encryption algorithm. This encryption scheme primarily relies on Elliptic Curve Cryptography (ECC), and collision resistance is mainly associated with the cryptographic hash functions used in ECC or other components. Our proposed scheme is deterministic in nature, making it resistant to collision attacks.

6.4. **Comparative Analysis.** In this section, we present a detailed comparison of our proposed encryption scheme against existing schemes based on encryption and decryption times. The results demonstrate the efficiency of our proposed scheme. The encryption and decryption times are critical for evaluating the performance of an encryption scheme. Table 2 compares the encryption and decryption times of our proposed scheme with those of existing schemes.

TABLE 2. Comparison of Encryption and Decryption Times

Scheme	Number of Characters in P	Encryption Time (sec)	Decryption Time (sec)
Proposed	500	0.0050	0.0025
Ref [9]	374	0.0950	0.0040
Ref [21]	409	0.0800	0.0600
Ref[38]	409	0.0930	0.1400
Ref [44]	409	1.950	0.8300

From Table 2, it is evident that our proposed scheme significantly outperforms existing schemes in terms of encryption and decryption times. Our scheme achieves encryption and decryption within milliseconds, whereas existing schemes take substantially longer, especially for larger texts.

6.5. **Future Scope.** Future work will focus on further optimizing the scheme, exploring additional applications, and assessing its performance in various real-world scenarios. We can also explore trying this encryption scheme for image encryption, which could potentially protect visual data in secure communication, medical imaging, and digital media protection. The research contributes to the field of cryptography by providing a novel approach that enhances the security and efficiency of data encryption, addressing some of the limitations of existing methods.

7. CONCLUSION

In conclusion, this research introduces a novel cryptographic scheme that integrates elliptic curves, hash functions, and involutory rhotrix structures to facilitate secure and authenticated communication between parties. By leveraging each party's computation of a specific private key using the other's public key, the scheme ensures mutual authentication and robust key derivation. The scheme employs a permuted ASCII table to enhance security and randomness, mapping ASCII values to elliptic curve points, hashing these points, and sorting the hash values.

This process underpins the secure transformation of data using involutory rhotrices constructed from specific private key coordinates. Message encryption involves converting blocks into permuted ASCII values, then into a 128-bit binary string, which is combined with random integers and private key coordinates via XOR to derive elliptic curve points for encryption. Each message point is encrypted into a pair of cipher points, accompanied by a digital signature to ensure data integrity during transmission. Our Python implementation and rigorous testing have demonstrated the efficiency and practical feasibility of the proposed approach for various cryptographic applications. This scheme represents a significant advancement in cryptographic protocols, offering robust security measures while maintaining computational efficiency in real-world applications. Additionally, for future work, there is potential for further research to explore the integration of various data formats into the proposed framework and to evaluate its effectiveness in terms of security and computational efficiency.

References

[1] A. Joux, *A one round protocol for tripartite Dife-Hellman*, J. Cryptol. 17(4) (2004), 263-276.

[2] A. K. Lenstra and E. R. Verheul, *Selecting cryptographic key sizes*, J. Cryptol. 14 (2001), 255-293.

[3] A. M. Johnston and P. S. Gemmell, *Authenticated key exchange provably secure against the man-in the-middle attack*, J. Cryptol. 15 (2002), 139-148.

[4] A. Mohammed, E. A. Ezugwu and B. Sani, *On generalization and algorithmatization of heart-based method for multiplication of rhotrices*, Int. J. Comput. Inform. Syst. 2 (2011), 46-49.

[5] A. O. Ajibade, *The concept of rhotrix in mathematical enrichment*, Int. J. Math. Edu. Sci. Tech. 34 (2003), 175-179.

[6] A. Sengupta and U. K. Ray, *Message mapping and reverse mapping in elliptic curve cryptosystem*, Sec. Commun. Networks 9(18) (2016), 5363-5375.

[7] B. B. Gupta, A. Gaurav and P. K. Panigrahi, *Analysis of security and privacy issues of information management of big data in B2B based healthcare systems*, J. Bus. Res. 162 (2023), 113859.

[8] B. King, *Mapping an arbitrary message to an elliptic curve when defined over* $GF(2^n)$, Int. J. Network Sec. 8(2) (2009), 169-176.

[9] B. V. Varun, M. V. Abhishek, A. C. Gangadhar, and U. Purushotham, *Implementation of encryption and decryption algorithms for security of mobile devices*, 19th Int. Conf. Comm. Technol. (ICCT) (2019), 1391-1395. IEEE

[10] C. C. Hun and J. C. Yen, *A new cryptography systems and its VISI realization*, J. Syst. Architect. 49 (2003), 355-367.

[11] D. Hankerson, S. Vanstone and A. Menezes, *Guide to Elliptic Curve Cryptography*, Springer-Verlag New York (2004).

[12] D. S. Kumar, C. Suneetha and A. Chandrasekhar, *Encryption of data using elliptic curve over finite fields*, Int. J. Distrib. Parallel Syst. 3(1) (2012), 301.

[13] F. Amounas and E. H. El Kinani, *Fast mapping method based on matrix approach for elliptic curve cryptography*, Int. J. Inf. Network Sec. (IJINS) 1(2) (2012), 54-59.

[14] G. Murtaza, N. A. Azam and U. Hayat, *Designing an efficient and highly dynamic substitution-box generator for block ciphers based on finite elliptic curves*, Sec. Commun. Networks. (2021), 1-14.

[15] H. Almajed, A. Almogren and M. Alabdulkareem, *iTrust-A trustworthy and efficient mapping scheme in elliptic curve cryptography*, Sensors 20(23) (2020), 6841.

[16] H. D. Tiwari and J. H. Kim *Novel method for DNA-based elliptic curve cryptography for IoT devices*, Elec. Tele. Research Inst. J. 40(3) (2018), 396-409.

[17] H. N. Almajed and A. S. Almogren, *SE-ENC: a secure and efficient encoding scheme using elliptic curve cryptography*, Access 7 (2019), 175865-175878. IEEE

[18] I. Ullah, N. A. Azam and U. Hayat, *Efficient and secure substitution box and random number generators over Mordell elliptic curves*, J. Inf. Sec. Appl. 56 (2021), 102619.

[19] J. Cahyono, D. Adzkiya and B. Davvaz, *A cryptographic algorithm using wavelet transforms over max-plus algebra*, J. King Saud. Univ. Comput. Inf. Sci. 34(3) (2022), 627–635.

[20] J. Fu, W. Zhou and S. Zhang, *Fabric blockchain design based on improved SM2 algorithm*, Int. J. Semant Web Inf. Syst. (IJSWIS) 19(1) (2023), 1-13.

[21] J. Muthukuru and B. Sathyanarayana, *Fixed and variable size text based message mapping techniques using ECC*, Global J Comp Sci Technol. 12(3) (2012), 12-18.

[22] J. R. Paragas, A. M. Sison and R. P. Medina, *Hill cipher modification: A simplified approach*, 11th Int. Conf. Commun. Softw. Networks. (ICCSN) (2019), 821-825. IEEE

[23] K. Kordov, *Text encryption algorithm for secure communication*, Int. J. Appl. Math. 34(4) (2021), 705.

[24] K. Keerthi and B. Surendiran, *Elliptic curve cryptography for secured text encryption*, Int. Conf. Circ. Power Comput. Technol. (ICCPCT) (2017), 1–5. IEEE

[25] K. A. Sattar, T. Haider, U. Hayat and M. D. Bustamante, *An efficient and secure cryptographic algorithm using elliptic curves and max-plus algebra-based wavelet Transform*, Appl. Sci. 13(14) (2023), 8385.

[26] L. D. Singh and K. M. Singh, *Implementation of text encryption using elliptic curve cryptography*, Procedia Comput. Sci. 54 (2015), 73–82.

[27] M. A. Naji, D. A. Hammood, H. A. Atee, R. S. Jebur, H. A. Rahim and R. B. Ahmad, *Cryptanalysis cipher text using new modeling: Text encryption using elliptic curve cryptography*, AIP Conf. Proceed. 2203(1) (2020), 020003.

[28] M. Hell and T. Johnson, *Breaking the strem ciphers F-FCSR-H and F-FCSR-16 in real time*, J. Cryptol. 24 (2011), 427-445.

[29] M. Jaiswal and K. Lata, *Hardware implementation of text encryption using elliptic curve cryptography over 192 bit prime feld*, Int. Conf. Adv. Comput. Commun. Inf. (ICACCI) (2018), 343-349. IiEEE

[30] M. Nadeem, A. Arshad, S. Riaz, S. W. Zahra, A. K. Dutta, M. Al Moteri and S. Almotairi, *An efficient technique to prevent data misuse with matrix cipher encryption algorithms*, Comput. Mater. Continua. 74(2) (2023), 4059-4079.

[31] M. T. Ataş and H. Güler, *Real-time encryption/decryption algorithm with a fractional chaotic system of various data: Image, speech, and text*, Int. J. Appl. Comput. Math. 8(4) (2022), 161.

[32] M. Zhao, C. Shi and Y. Yuan *Blockchain-based lightweight authentication mechanisms for industrial internet of things and information systems*, Int. J. Semant Web Inf. Syst. (IJSWIS) 20 (2024), 1-30.
[33] N. Koblitz, *Elliptic curve cryptosystems*, Math. Comput. 48(177) (1987), 203-209.
[34] P. L. Sharma, S. Gupta, H. Monga, A. Nayyar, K. Gupta and A. K. Sharma, *TEXCEL: text encryption with elliptic curve cryptography for enhanced security*, Multimed. Tools. Appls. (2024), 1-29.
[35] P. L. Sharma, S. Gupta, A. Nayyar, M. Harish, K. Gupta and A. Sharma *ECC based novel color image encryption methodology using primitive polynomial*, Multimed. Tools. Appls. (2024), 1-40.
[36] P. Bh, D. Chandravathi and P.P. Roja, *Encoding and decoding of a message in the implementation of Elliptic Curve cryptography using Koblitz's method*, Int. J. Comput. Sci. Eng. 2(5) (2010), 1904-1907.
[37] P. Das and C. Giri, *An efficient method for text encryption using elliptic curve cryptography*, 8th Int. Adv. Comput. Conference. (IACC) (2018), 96-101. IEEE
[38] R. Lidl and H. Niederreiter, *Finite fields*, Cambridge University Press Cambridge Second Edition (1997).
[39] R. Rihartanto, D. S. B. Utomo, H. Februariyanti, A. Susanto and W. Khafdhah, *Bit-based cube rotation for text encryption*, Int. J. Electr. Comput. Engg. 13(1) (2023), 709.
[40] S. Azhar, N. A. Azam and U. Hayat, *Text encryption using pell sequence and elliptic curves with provable security*, Comput. Contin. 71 (2022), 4972-4989.
[41] S. M. C. Vigila and K. Muneeswaran *Implementation of text based cryptosystem using elliptic curve cryptography*, 1st Int. Conf. Adv. Comput. (2009), 82-85. IEEE
[42] T. Ivanova, B. Stoyanov and D. Dobrev, *Secure text encryption based on cliford attractors*, 31st National Conf. with Int. participation (TELECOM) (2023), 1-4. IEEE
[43] V. Kamalakannan and S. Tamilselvan, *Security enhancement of text message based on matrix approach using elliptical curve cryptosystem*, Procedia Mater. Sci. 10 (2015), 489-496.
[44] V. S. Miller, *Use of elliptic curves in cryptography*, Adv. Cryptogr. CRYPTO'85 (Lect Notes Comput Sci.) (1986), 218.
[45] Y. Genç and E. Afacan, *Implementation of new message encryption using elliptic curve cryptography over finite fields*, Int. Congr. Adv. Tech. Engg. (ICOTEN) (2021), 1-6. IEEE

Annexure -A : Permuted values of Standard ASCII Table

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17	18	19	20
167	216	228	234	182	64	44	154	236	178	163	97	189	116	38	223	204	179	191	35	211
21	22	23	24	25	26	27	28	29	30	31	32	33	34	35	36	37	38	39	40	41
233	92	219	150	124	112	22	164	37	131	208	40	20	135	226	177	238	79	210	161	121
233	92	219	150	124	112	22	164	37	131	208	40	20	135	226	177	238	79	210	161	121
42	43	44	45	46	47	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62
127	16	49	56	90	39	62	94	55	142	119	83	188	107	153	227	60	181	10	253	143
63	64	65	66	67	68	69	70	71	72	73	74	75	76	77	78	79	80	81	82	83
41	220	146	25	229	199	113	72	71	81	152	58	45	170	87	69	6	249	137	195	217
84	85	86	87	88	89	90	91	92	93	94	95	96	97	98	99	100	101	102	103	104
139	80	67	118	126	218	185	175	75	165	68	33	110	224	54	122	48	155	91	130	225
105	106	107	108	109	110	111	112	113	114	115	116	117	118	119	120	121	122	123	124	125
230	209	93	147	23	50	85	201	156	105	7	250	193	96	115	32	17	166	4	247	240
126	127	128	129	130	131	132	133	134	135	136	137	138	139	140	141	142	143	144	145	146
160	31	73	104	120	14	30	13	103	109	24	200	180	11	254	114	27	99	144	190	157
147	148	149	150	151	152	153	154	155	156	157	158	159	160	161	162	163	164	165	166	167
65	88	3	246	47	123	222	70	2	245	231	108	237	76	82	213	43	241	149	1	244
168	169	170	171	172	173	174	175	176	177	178	179	180	181	182	183	184	185	186	187	188
192	89	57	132	46	129	59	63	61	53	140	98	21	184	9	252	95	215	205	19	207
189	190	191	192	193	194	195	196	197	198	199	200	201	202	203	204	205	206	207	208	209
36	8	251	187	101	136	128	202	176	169	172	100	138	242	221	173	145	84	29	34	42
210	211	212	213	214	215	216	217	218	219	220	221	222	223	224	225	226	227	228	229	230
158	111	134	0	243	159	141	151	77	86	12	255	74	186	18	212	194	148	133	162	196
231	232	233	234	235	236	237	238	239	240	241	242	243	244	245	246	247	248	249	250	251
198	66	235	117	171	51	197	203	174	183	206	239	168	52	15	106	26	232	78	125	28
252	253	254	255																	
102	214	5	248																	

TABLE 3. Annexure -A : Permuted values of Standard ASCII Table

Customized ASCII Table

NUL	SOH	STX	ETX	EOT	ENQ	ACK	BEL	BS	HT	LF	VT	FF	CR	SO	SI	DLE	DC1	DC2	DC3	DC4
167	216	228	234	182	64	44	154	236	178	163	97	189	116	38	223	204	179	191	35	211
NAK	SYN	ETB	CAN	EM	SUB	ESC	FS	GS	RS	US	SPACE	!	"	#	$	%	&	‘	(	)
233	92	219	150	124	112	22	164	37	131	208	40	20	135	226	177	238	79	210	161	121
*	+	,	-	.	/	0	1	2	3	4	5	6	7	8	9	:	;	<	=	>
127	16	49	56	90	39	62	94	55	142	119	83	188	107	153	227	60	181	10	253	143
?	@	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
41	220	146	25	229	199	113	72	71	81	152	58	45	170	87	69	6	249	137	195	217
T	U	V	W	X	Y	Z	[	\	]	^	_	‘	a	b	c	d	e	f	g	h
139	80	67	118	126	218	185	175	75	165	68	33	110	224	54	122	48	155	91	130	225
i	j	k	l	m	n	o	p	q	r	s	t	u	v	w	x	y	z	{	|	}
230	209	93	147	23	50	85	201	156	105	7	250	193	96	115	32	17	166	4	247	240
~	DLT	€		‚	ƒ	„	…	†	‡	^	‰	Š	‹	Œ		Ž			‘	’
160	31	73	104	120	14	30	13	103	109	24	200	180	11	254	114	27	99	144	190	157
“	”		–	—	~	™	š	›	œ		ž	Ÿ		¡	¢	£	¤	?	¦	§
65	88	3	246	47	123	222	70	2	245	231	108	237	76	82	213	43	241	149	1	244
¨	©	ª	«	¬		®	¯	°	±	²	³	´	µ	¶	·	¸	¹	º	»	¼
192	89	57	132	46	129	59	63	61	53	140	98	21	184	9	252	95	215	205	19	207
½	¾	¿	À	Á	Á	Ã	Ä	Å	Æ	Ç	È	É	Ê	Ë	Ì	Í	Î	Ï	Ð	Ñ
36	8	251	187	101	136	128	202	176	169	172	100	138	242	221	173	145	84	29	34	42
Ò	Ó	Ô	Õ	Ö	×	Ø	Ù	Ú	Û	Ü	Ý	Þ	ß	à	á	â	ã	ä	å	æ
158	111	134	0	243	159	141	151	77	86	12	255	74	186	18	212	194	148	133	162	196
ç	è	é	ê	ë	ì	í	î	ï	ð	ñ	ò	ó	ô	õ	ö	÷	ø	ù	ú	û
198	66	235	117	171	51	197	203	174	183	206	239	168	52	15	106	26	232	78	125	28
ü	ý	þ	ÿ																	
102	214	5	248																	

TABLE 4. Customized ASCII Table

Department of Mathematics & Statistics Himachal Pradesh University Shimla, India-171005
Email address: shalini.garga1970@gmail.com

Department of Mathematics Government College Karsog, India-175011
Email address: ruchinarang8878@gmail.com

Department of Mathematics & Statistics Himachal Pradesh University Shimla, India-171005
Email address: kritika993@gmail.com