



OPEN ACCESS

EDITED BY

Shitharth Selvarajan,
Leeds Beckett University, United Kingdom

REVIEWED BY

Rakesh Kadu,
Shri Ramdeobaba College of Engineering and
Management, India
Hariprasath Manoharan,
Panimalar Institute of Technology, India

*CORRESPONDENCE

Ramireddy Nava Teja Reddy
✉ rnavateja2233@gmail.com
Ch. Rami Reddy
✉ crreddy229@gmail.com
Amr Yousef
✉ a.yousef@ubt.edu.sa
Ahmed Emara
✉ a.emara@ubt.edu.sa

RECEIVED 04 February 2024

ACCEPTED 21 May 2024

PUBLISHED 04 July 2024

CITATION

Nava Teja Reddy R, Kavitha M, Reddy GS,
Yousef A, AboRas KM, Emara A and Reddy CR
(2024) Mayfly optimistic hyperelliptic curve
cryptosystem. *Front. Comput. Sci.* 6:1381850.
doi: 10.3389/fcomp.2024.1381850

COPYRIGHT

© 2024 Nava Teja Reddy, Kavitha, Reddy,
Yousef, AboRas, Emara and Reddy. This is an
open-access article distributed under the
terms of the [Creative Commons Attribution
License \(CC BY\)](https://creativecommons.org/licenses/by/4.0/). The use, distribution or
reproduction in other forums is permitted,
provided the original author(s) and the
copyright owner(s) are credited and that the
original publication in this journal is cited, in
accordance with accepted academic practice.
No use, distribution or reproduction is
permitted which does not comply with these
terms.

Mayfly optimistic hyperelliptic curve cryptosystem

Ramireddy Nava Teja Reddy^{1*}, M. Kavitha¹, G. Sudarsana Reddy²,
Amr Yousef^{3*}, Kareem M. AboRas⁴, Ahmed Emara^{3*} and
Ch. Rami Reddy^{5,6*}

¹Department of CSE, Koneru Lakshmaiah Education Foundation, Vaddeswaram, India, ²Department of Computer Science and Engineering, SVR Engineering College, Nandyal, India, ³Electrical Engineering Department, University of Business and Technology, Jeddah, Saudi Arabia, ⁴Department of Electrical Power and Machines, Faculty of Engineering, Alexandria University, Alexandria, Egypt, ⁵Department of Electrical and Electronics Engineering, Joginpally B. R. Engineering College, Hyderabad, India, ⁶Applied Science Research Center, Applied Science Private University, Amman, Jordan

Various applications use asymmetric cryptography to secure communications between both parties, and it raises the main issue of generating vast amounts of computation and storage. Thus, elliptic curve cryptography (ECC) is a methodology that emerged to overcome this issue using its low computation and generation of small keys with its strong encryption strategy. ECC is becoming mandatory and used mostly for public key encryption protocols. ECC has expanded cumulative acceptance in practice due to the reduced bit magnitude of operands compared to RSA for safety level. Previously, protocols designed for ECC suggested calculation of scalar development and it was accomplished in finite fields as projective, affine, and Jacobian simulations of coordinates. Arithmetic operations in a limited area establish the core benefits of the ECC algorithm. Even though ECC generated an issue of complex key generation using its curve formation, to overcome this issue a hyperelliptic curve cryptosystems (HECC) is proposed in this study. HECC perform ECC in the Public Key Cryptography (PKC) domain. This study presented an optimization-based key generation and made a random selection of integers for encrypting the message. Selecting a prime number as the private key and multiplying it to the encrypted message to generate a public key is done. This encrypted message is mapped to the curve to check whether it satisfies the curve equation or not. Once an encrypted message is obtained, it is then sent to a second party for pursuing the message. On the side of the second party, a reverse process called decryption takes place. Thus, a secured transmission of data communication takes place. Implementing this algorithm in MATLAB resulted in 94% accuracy and an error of 6%, which was a higher performance ratio than previous methods.

KEYWORDS

elliptic curve cryptography, public key encryption, finite field, Jacobian models, hyperelliptic curve cryptosystems and private key

1 Introduction

Information security is important in today's world as every transaction becomes digitized. For legal and profitable trading, integrity, confidentiality, non-reputability, and integrity of the associated data are necessary (Mrabet et al., 2016). It was done using cryptographic methods and integrated cryptographic methods must satisfy all these cryptographic concerns. The requirement of these properties made a secure

communication system and brought a revolutionary change in data management (Zhou et al., 2019). Devices in billions such as smartphones, wearable devices, and smart cars are linked to the internet as they are able to share data with each other in a communication system. Procedures are mostly resource-forced; generally, information is deposited in the cloud and devices and people are permitted to upload and take the information anytime and wherever until the time of using the internet (Mahto and Yadav, 2017). However, it caused difficulty that made to verifying the safety of information. Direct control of data organization with data owner was enforced to strictly control the information access and protect it from the cloud as clouds cannot be completely reliable. To allow and secure information in an openly dispersed computing field, encryption of data before storing it in the cloud was necessary (Liu et al., 2015). Thus, the symmetric and asymmetric key generation phase occurs that realizes the purpose of encryption. Nonetheless, this type of well-formed encryption required requests to share a collective period key in progress among information user and owner (Naresh et al., 2018). Sharing information in IoT structures was made difficult due to its repeated encryption, and data in each session of encryption were shared; thus, it created a necessity of complicated key management protocol that incurs storage overhead and high computation. Similarly for asymmetric encryption, there is the necessity of obtaining each data owner's public key, repeatedly encrypting data and storing it in various encrypted photocopies of identical information in the cloud (Sahoo et al., 2013). Wireless systems require resource-constrained devices such as smart cards, smartphones, RFID tags, and small processors. These devices play a vital character in controlling the safety of satellite communication, e-commerce, e-banking, IoT applications, and embedded systems (Mahmood et al., 2018). However, problems were raised while executing safety for a communication system.

Cryptography is the art and science of encrypting data using encryption algorithms to make communications (such as credit card information and emails) safe when they are transferred across a network. In network security, several encryption algorithms are utilized. Asymmetric-key cryptography (public key cryptography) involves using pairs of keys, a public key for encryption, and a private key for decryption. The public key is widely distributed, while the private key is kept secret. Examples include RSA (Rivest–Shamir–Adleman) and elliptic curve cryptography (ECC). Lightweight cryptography refers to cryptographic algorithms and protocols specifically designed to operate efficiently on resource-constrained devices, such as those found in IoT (Internet of Things), RFID (Radio-Frequency Identification), smart cards, and embedded systems.

Lightweight cryptographic algorithms are designed to have low computational overhead, requiring minimal processing power and memory resources. This enables them to run efficiently on devices with limited computing capabilities. Lightweight cryptography aims to minimize memory usage, allowing it to operate on devices with limited RAM and storage capacity. This characteristic is crucial for resource-constrained devices where memory is often at a premium. Lightweight cryptographic algorithms are designed to have a compact code size, reducing the amount of storage space required for implementation. This is essential for devices with limited program memory. Lightweight

cryptography emphasizes fast cryptographic operations, enabling rapid encryption, decryption, and other cryptographic functions. This characteristic is vital for real-time applications and scenarios where low latency is critical. Lightweight cryptographic algorithms are optimized to minimize energy consumption, making them suitable for battery-powered and energy-constrained devices. By reducing the computational workload, these algorithms help extend the battery life of devices. Despite their focus on efficiency, lightweight cryptographic algorithms aim to provide adequate security for the intended application. While they may not offer the same level of security as their more computationally intensive counterparts, they are designed to resist common cryptographic attacks and threats relevant to their deployment scenarios.

The need for secure data transfer in wireless networks led to the development of cryptographic algorithms (He and Zeadally, 2014). There are a lot of drawbacks to these algorithms, such as the fact that they increase processing time and computational complexity while simultaneously increasing power consumption. This highlights the need to create a robust cryptographic method that circumvents these constraints. A method that used two distinct keys, the public and private keys, was proposed by public key cryptography (PKC; Kumar et al., 2019). As the user chose the private key, no one else was able to decipher it. By using a reversible mathematical method, the public key is generated from the private key and made publicly available to all users. You may use both keys for encryption and decryption as they are compatible with each other. It was necessary to use very safe methods as the private key was never disclosed, in contrast to symmetric key encryption. Two main categories for this kind of encryption are ECC and Rivest–Shamir–Adelman (RSA; Ding et al., 2018). An increase in key size has the potential to lessen the security risks associated with RSA, but doing so has increased the complexity of the key generation, encryption, and decryption schemes, necessitating more storage space and computing power. Real-time experiments involve analyzing the efficiency of ECC implementations in terms of throughput, latency, and resource utilization. Throughput measures the number of ECC operations processed per unit of time, while latency measures the time taken to complete a single ECC operation. Resource utilization assesses the hardware resources (e.g., logic gates and memory) consumed by the ECC implementation. Real-time experiments aim to evaluate the performance of ECC algorithms under various operating conditions, such as different key sizes, curve parameters, and cryptographic operations (e.g., key generation, encryption, decryption, and digital signatures). Performance metrics include execution time, power consumption, and area utilization. Real-time experiments also involve assessing the security of ECC implementations against known cryptographic attacks, including side-channel attacks, fault attacks, and implementation-specific vulnerabilities. Security analysis helps identify potential weaknesses and refine the ECC implementation to enhance its resistance to attacks (Shitharth et al., 2023a).

Hyperelliptic curve cryptography (HECC) extends the ideas of elliptic curve cryptography (ECC) by using hyperelliptic curves, instead of elliptic curves. Hyperelliptic curves used in cryptography are typically defined over a finite field \mathbb{F}_q and are expressed by the equation:

$$C: y^2 + h(x)y = f(x)$$

where $h(x)$ and $f(x)$ are polynomials over \mathbb{F}_q . The degree of $f(x)$ is $2g+1$ or $2g+2$ for a curve of genus g . In cryptographic applications, the genus g is usually small, typically 1 or 2. The group used for cryptographic purposes is derived from the Jacobian of the curve, which represents equivalence classes of divisors on the curve. A divisor is a formal sum of points on the curve. The Jacobian of a curve of genus g is a g -dimensional abelian variety, which forms a group. The complexity of group operations in the Jacobian of a hyperelliptic curve is higher than that in the elliptic curve group, but it can still be made efficient with careful implementation.

The private key in HECC is a randomly selected integer k from the range $[1, n-1]$, where n is the order of the subgroup used for cryptographic operations. The public key is calculated as $P = [k]Q$, where Q is a base point on the Jacobian of the hyperelliptic curve, and $[k]$ denotes the scalar multiplication of Q by k . These operations in HECC can be analogous to those in ECC, such as ElGamal encryption, where messages are embedded into points on the Jacobian. Similarly, digital signature algorithms like ECDSA can be adapted for HECC, using the properties of the Jacobian for signing and verification processes (Bhageerath Chakravorthy et al., 2021; Shitharth et al., 2023b).

The well-known PKC system is RSA's Achilles' heel when it comes to wireless systems, whereas ECC provides superior security with smaller key sizes, less processing time, and less computing complexity (Kapoor et al., 2008). As a result, a new approach to public key cryptography known as hyperelliptic curve cryptosystem (HECC) has evolved (Mani and Gopi, 2013). An elliptic curve method with a genus of 2 or higher is called a hyperelliptic curve. When the genus of a HEC is 1, we obtain an elliptic curve. Superior to ECC in many respects, HECC boasts reduced key sizes, reduced computing loads, increased speeds, reduced bandwidth needs, and reduced power usage. Engineering applications such as cellular phones, web servers, chip cards, electronic commerce, and cryptocurrencies all make effective use of HECC (Gueron and Krasnov, 2015). Because of these benefits, HECC is an excellent option for lightweight cryptosystems and is easy to implement in hardware and software.

The major contributions of this study are as follows:

- Producing a secured encryption system based on HECC with compacted key size and optimal key selection.
- An encryption system uses the hyperelliptic curve to minimize weight in code words that are hard to sample and make this encryption system protect from several attacks.
- Flaws of different encryption schemes are overwhelmed by applying block cipher modes of operation during encryption.
- To provide a security analysis-based efficient scheme that satisfies security requirements in the communication system.

Considering the attacks that happened for previous algorithms and assuming that there was no more error to attack MO-HECC algorithm is developed. Error caused by key parameters are solved in this scheme, thus improving the

security efficiency of this algorithm than any other methods. The rest of the article is organized as follows: Section 2 discusses the previous studies related to the proposed algorithm and its drawbacks are presented. Section 3 briefs about the background study of methods used for its benefits in the proposed model, and Section 4 discusses the proposed MO-HECC cryptography and its methodology for secured communication. Section 5 presents the experimental result of proposed and existing methods of cryptography. Section 6 concludes the whole manuscript.

2 Literature review

Pan et al. (2016) have presented an Elliptic curve Digital Signature Algorithm (ECDSA) and produced an extraordinary realization server termed Guess. It has a 256-bit key size on a Linux-operated product computer, a harnessing desktop graphics processing unit (GPU), and a highlighted cryptographic accelerator. It demonstrated involvement in exploiting the computer influence of guess and cannot give power to tenants that also includes optimization considering different software and hardware factors. However, it comprises rigorous checks on guess flexibility to side-channel attacks, and in turn, it increases the cost with respect to elliptic curves. Sowjanya et al. (2019) provided a key strategy based on ECC that is lightweight, uses a refresh/update method, and does not rely on bilinear pairing. Direct attribute and user revocation are part of the authority-driven key distribution process. It is protected by the decisional Diffie–Hellman problem under an elliptic curve. The use of semi-functional cipher texts and keys in this algorithm's safety proof, nevertheless, adds difficulty due to its foundation in pairing and orthogonal subgroups that are composite or premier ordering categories. Khan et al. (2019) have designed an ECC-established mutual verification protocol for smart grid establishment using the biometric methodology. It was developed in an SG environment and developed using ECC cryptography. One limitation of this method is that it is suitable only for smart grids and not for any other practical applications. Ibrahim and Alharbi (2020) have developed a safe dynamic S-box resulting Henson map algorithm. It used a dynamic S-box to build an image encryption system that consists of hash verification and decryption steps suitable for cipher attacks. Hash assists as an image-reliant keystream and is collectively used as an image-reliant S-box struggle as plain text attacks. It has protected encryption keys using ECC, however, in the situation of temporary retrieval of keys using S-box failed due to statistical attacks. De Rango et al. (2020) have presented a dynamic IOT retreat model depending on standard IoT edge design in which nodes interchange messages using a message queue telemetry transport procedure. It tends to intensify safety by modifying data tempering, replay attacks, and eavesdropping using ECC. However, it mainly focused on the application of elliptic curves for both fog and cloud calculating with the objective of reducing energy, rather unfortunately it faced moderation of DoS attacks in fog calculation. Zhang et al. (2020) have presented a public key encryption algorithm founded on elliptic codes that help us to reduce the size of confidential key and minder attacks calculated on numerous expectations that unfortunately turned into a drawback to the system. The lowest

weight code word was not efficiently sampled, and the points evaluated were large which made them invalid. However, its failure to perform without using conventional transform techniques was an issue. Wu et al. (2017) have developed an asymmetric image encryption process for key collections and the number of keys in confidential data transmission among some people. It was tiny and the key transmission system was comparatively secure and simple. Initially, it compressed plain images, and then, color images were encrypted using an advanced 4D cat map monitored by asymmetric encryption that was operated depending on ECC encryption, and at last encrypted image was diffused universally. However, this algorithm was affected by plaintext attacks and also cipher text attacks that decrypted the quality of the image.

Almajed and Almogren (2019) have presented a SE-Enc procedure that was effectively working on ECC that encodes information and maps them to an elliptic curve. Methods that avoid encoding procedures get affected by encryption defects and also it identifies the encoding phase and makes it resistant to encryption attacks. Proof-based security analysis was conducted to highlight the degree to which the system was secured. However, this method unfortunately failed to train the implications of response encryption stuff at the mapping phase causing vulnerability to security. Mehrabi et al. (2020) have presented residue number system-based hardware design of ECC for two relations of elliptic curves that were termed as twisted and short Edwards curves. It developed specified executions for SECP256K1 and ED25519 ordinary curves. It proposed an RNS proto-type design to support speed elliptic curve point addition (ECPA), point doubling, and tripling. It provides an ECC point development procedure on the Xilinx FPGA platform. However, it requires various scalar multiplication methods that were employed for the ECC state machine and it reduced point multiplication redundancy. Rawat and Deshmukh (2020) have developed a computation-effective group key algorithm that established common keys for groups and also for subgroups. It used an advanced group key arrangement using a hierarchical key tree, and thus, it attempted to develop a key using a hierarchical key tree approach. Rather, this technique failed to reduce communication costs and also increased security thread even after securing the message with a key. Naresh et al. (2018) have presented a discrete algorithm problem-dependent lightweight secured data transfer system using HECC. It used different genus curves for different prime fields and different types of constraint networks. It used a Jacobean matrix for decryption and encryption to change the scope of the field. Nevertheless, it is detected that the application of genus 4 HECC on an embedded microprocessor that this cryptosystem is healthier suitable for usage in embedded atmospheres than to common purpose processors. Liu et al. (2021) have presented a demodulation frequency band selection method using the Mayfly optimization algorithm. It was used to diagnose bearing fault in the interference from random pulses, and its variance static index has been applied to evaluate energy spectrum distribution that determined central frequency. However, energy from the impulse component was difficult to concentrate on a series of spectrum lines, and thus, mayfly optimization failed to serve efficiently for this application (Ellappan et al., 2023; Syed et al., 2024).

3 Background information on HECC and optimization

Methods used to develop the proposed design of the MO-HECC algorithm and its benefits are discussed below.

3.1 HECC cryptography

HECC is used in a wide range due to its low computation and its advantage in resulting similar level of cryptographic toughness as additional public key protocols with exact minor size keys. Substantial size in different key sizes means that devices with lesser calculation abilities accomplish efficiently. Similarly, the HECC algorithm is based on discrete logarithm construction of elliptic curves terminated on finite fields. A helpful phenomenon of HECC is that it is recommended to interchange keys and provide safe communications between two connections; rather, they show the difference in encryption method. HECC is used to both encrypt and sign messages.

$$y^2 \equiv x^3 + ax + b \pmod{p} \quad (1)$$

In this Equation (1), $a, b \in \mathbb{Z}_p$ and $4a^3 + 27b^2 \neq 0 \pmod{p}$ it is obtained for the hyperelliptic curve equation satisfies $= 1$, where h is a polynomial with $\deg h \leq g$. Thus, an elliptic curve should be non-singular which means a plot has no vertices. Group actions performed on the elliptic curve are given as addition/multiplication or modulation. Group calculation is the HECC main action defined as numerous instances of group point doubling. It consists of d as an integer number known as the private key and $G = (x_i, y_i)$ as a base point on an elliptic curve. A hyperelliptic curve satisfying the above conditions can be given as shown in Equation (2).

$$C: Y^2 + h(X)Y \quad (2)$$

From Equation (2), it is predicted that each hyperelliptic curve contains a single point at infinity and an operation dG is performed to double the time for G , which gives another point (x_j, y_j) termed as public key. Selected two coordinates $P = (x_1, y_1)$ and $Q = (x_2, y_2)$ as performing operations on (P, Q) results in (x_3, y_3) using a random variable Γ , it might be addition, scalar multiplication, inverse operations, and subtraction. For a point $P(x_1, y_1)$ on a hyperelliptic curve, by selecting a point in the curve, it becomes $[x, -y, -h(x)]$. Thus, the security of HECC is based on the hardness of the logarithm problem of these selected operations and each operation has a different computational cost. HECC works in four phases to discuss keys and secure communication. These phases are listed as follows:

- First phase—Generating parameters for elliptic curve and calculating Private key value.
- Second phase—Numerical Encoding of the message.
- Third phase—Hashing the message for security.
- Fourth phase—Mapping the cipher text to an elliptic curve.

The main calculation complicated in the first phase is the computation of the public key, and it is derived as the product of d and G . This multiplication is upgraded in several ways by improving the selection of the private key using an optimistic technique. The second phase is based on the method recommended to convert message characters into numbers using the ASCII coding technique. Each message must be encoded in a way that prevents encryption attacks. Similarly, the third phase involves hashing the message using the signature to ensure security. It is the responsibility of the last phase, sometimes known as the fourth phase, to map the encoded message to an elliptic curve. The mapping of the message to a curve that safeguards against encryption attacks is an essential step.

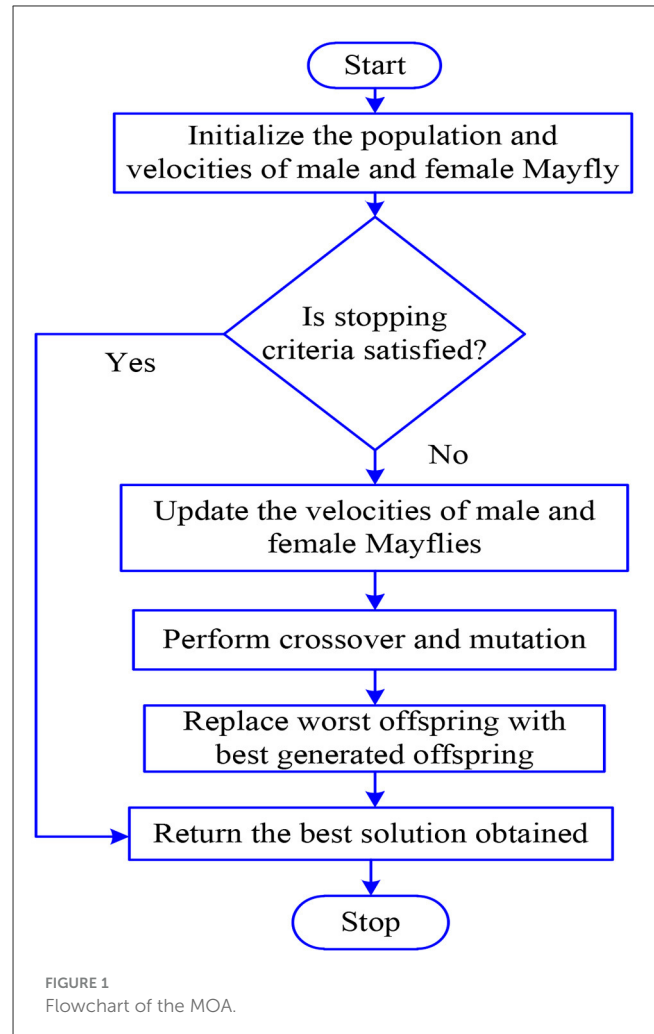
3.2 Mayfly optimization

The Mayfly optimization approach (MOA) provides a strong hybrid algorithmic framework like that of mayflies, which was inspired by their unique behavior. This mathematical model is based on the social behavior of mayflies, with a particular emphasis on the mating process of these insects. For the sake of this algorithm, it is assumed that once mayflies emerge from their eggs, they are already adults and that only the most robust mayflies succeed in surviving.

The MOA stands out due to its unique combination of swarm intelligence and evolutionary algorithms, inspired by flight behavior and mating process of mayflies. Despite the existence of other well-known metaheuristic optimization methods like Particle Swarm Optimization (PSO), Genetic Algorithm (GA), Firefly Algorithm (FA), and Differential Evolution (DE), the MOA demonstrates superior performance in terms of convergence rate and speed as evidenced by comparisons with these methods. The utilization of nuptial dance and random flight processes of MOA effectively enhances the balance between exploration and exploitation properties, aiding in escaping local optima. Therefore, the innovative approach and successful integration of nature-inspired behaviors of Mayfly Algorithm make it a compelling choice for optimization tasks, despite the presence of other proven bio-inspired algorithms in the field (Zervoudakis and Tsafarakis, 2020; Rabie et al., 2024).

The ideal solution to the issue is shown by the position of each mayfly in the region of search space.

The answer to the issue is represented by the position of each mayfly in the search space. Based on the fitness function, this method locates the optimal location, which is often referred to as the best CH. In this mathematical model, nuptial dance and the movement of mayflies are taken into consideration within the limit that has been pre-defined. Moreover, MOA determines the crossover value between mayflies, which allows for the optimal location to be obtained consequently. There is a flow chart of the MOA shown in Figure 1. In the beginning, two different groups of mayflies are produced at random and then positioned in the issue space as a potential solution, which is represented by a vector of d dimensions. The velocity of a mayfly and the change in its location are both taken into consideration after the fitness function $f(x)$ has been evaluated to discover a solution to the issue. The



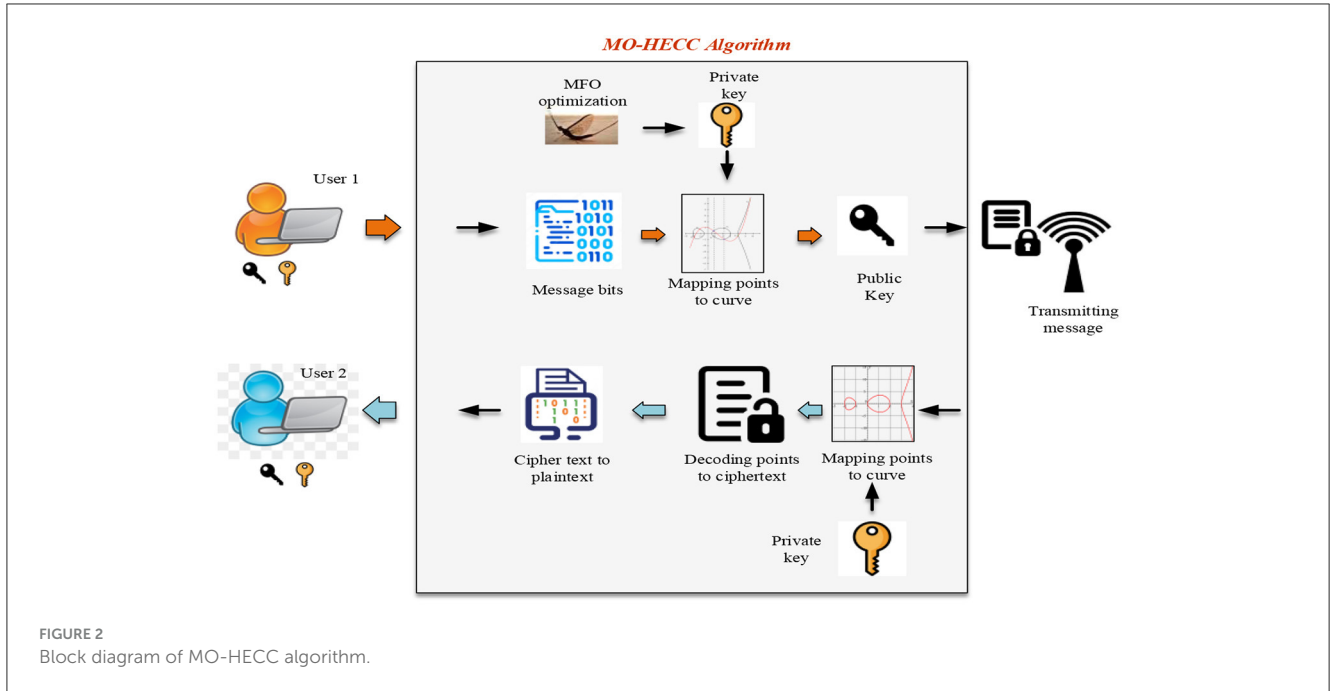
flying direction of every mayfly is in a dynamic direction, and every mayfly adjusts its position so that it is closer to the best and the best position that any mayfly in the swarm has attained in the past. This best position is used to determine who will be picked for the CH post. To achieve the highest possible energy level among the nodes, it is necessary to decrease the distance between them and pick the CH with the lowest energy consumption rate (ECR).

3.2.1 Movement of male mayflies

In the process of initialization, both male and female mayflies were carried out independently. To modify its location, each mayfly takes into account not only its own experiences but also the experiences of its neighbors. It is expected that the current position of mayfly is y_j^t and search space is represented as j at a time t . A change in position occurs when velocity is added to the location that is already being tracked. It is possible to express it in the form of Equation (3).

$$y_j^{t+1} = y_j^t + V_{jk}^{t+1} \tag{3}$$

$$V_{jk}^{t+1} = V_{jk}^t + c_1 e^{-\beta r^2 p (pbest_{jk} - V_{jk}^t)} + c_2 e^{-\beta r^2 g (gbest_{jk} - V_{jk}^t)} \tag{4}$$



In Equation (4), c_1 and c_2 are positive attraction constraints. $pbest_{jk}$ is the finest location of Mayfly i has always stayed? Here, β is represented by a fixed visibility coefficient, r^2mf reserve among female and male mayflies, and fm random walk coefficient. Allowing for minimization problems $pbest$ can be considered as given in Equation (5).

$$pbest_j = \begin{cases} y_j^{t+1}, & \text{if } (y_j^{t+1}) < f(pbest_j) \\ \text{is kept same,} & \text{otherwise} \end{cases} \quad (5)$$

Similarly, $gbest$ at times step t can be given as Equation (6).

$$gbest \in \{pbest_1, pbest_2, \dots, pbest_N | f(pbest)\} = \min\{f(pbest_1), f(pbest_2), \dots, f(pbest_N)\} \quad (6)$$

Assuming that N is the total number of male mayflies that are present in the swarm. You may compute the Cartesian distance between male and female mayflies by using the formula that was described previously.

$$r^2mf = \sqrt{\sum_{k=1}^n (y_{jk} - Z)^2} \quad (7)$$

Here in Equation (7), Z matches to $pbest$ or $gbest$. At this point, the values for the male and female mayflies have been initialized to their respective positions. A discussion on the evaluation of the fitness function for the proposed model about the optimization of the MOA will take place in the next section.

3.2.2 Movement of female mayflies

According to the outcome of the present solution, the attraction process between men and females happens. The women who

TABLE 1 Key generation algorithm.

Procedure 1: Key generation
Input: d and G ; // Select a private key and Prime order value//
Output: Public key Q
 Call **Procedure 6** //MFO optimization //
 $P = nG$; $n=21$;
 $Q = P.d$
Output: Q ;
End

TABLE 2 Message-encoding algorithm.

Procedure 2: Message-Encoding Algorithm
Input: The message M and Q
Output: Encoded message
 Get (M) // Message Bit//
 Integer m ; // Point in the elliptic curve //
 Cipher text C_1, C_2 ;
 $C_1 = P$;
 $C_2 = m + rQ$;
 $C = C_1 + C_2$;
 Enc $K(C, C_1, C_2)$;

perform the best are attracted to the men who perform the best, and this process continues until all of the potential partners have been located. It is possible to provide the current location of female mayflies as follows using Equation (8):

$$z_j^{t+1} = z_j^t + V_{jk}^{t+1} \quad (8)$$

It is possible to determine velocity when female mayflies are taken into consideration by assuming that the best female attracts

TABLE 3 Encryption mapped points using a public key.

<p>Procedure 3: Encrypting Mapped Points Using Public Key Algorithm Input: M_p, P // Taking input of mapped points and public key// Output: E_p // Encrypted Points // Do ($M_p = 0$) { Get (P) Get (M_p) Out E_p</p>

TABLE 4 Decryption using a public key.

<p>Procedure 4: Decrypting Encrypted Points Using Shared Key Algorithm Input: E_p Output: M_p Get (E_p) Subtract P from E_p Repeat for all encrypted points; Output: Mapped points</p>

TABLE 5 Algorithm to convert binary values to plaintext.

<p>Procedure 5: Converting Binary Values to Plaintext Input: Binary values Output: The plain text message M Get binary values; Convert each 8 bits into its corresponding char; Repeat for all blocks; For each N char aggregate to single block; Output message</p>
--

TABLE 6 Pseudocode for proposed MEC-MFO algorithm.

<p>Procedure 6: MFO optimization Input: population size, maximum iteration value Output: Best fitness value $f = (f_1, f_2, f_3, \dots, f_n)$ Initialize population and velocity of male and female mayflies randomly Evaluate population value and find gbest for iterations = 1 to maximum iterations do for $i=1$ to population size do update pbest Evaluate and update velocities of male and female mayflies End For Sort mayflies and rank them Perform crossover Generate male and female offspring Mutate offspring Replace worst gbest with pbest Update gbest End</p>

the best male and that the second-best female attracts the second-best male. There is an update to the velocity of a female, and the equation may be written using Equation (9) as follows:

$$V_{jk}^{t+1} = \begin{cases} \text{if } fitness(Z_j) > fitness(Y_j) \\ g^* V_{yz} + c_2^* e^{-\beta r^2 m f^*} (Z_{jk} - Y_{jk}) \\ \text{else if } fitness(Z_{jk}) < fitness(Y_{jk}) \\ g^* V_{yz}^t + fl^* r \end{cases} \quad (9)$$

where V_{jk}^{t+1} is the j th constituent of the k th velocity of the k th female mayfly at the time t , Z_{jk} is the location of a female mayfly in

element k at time t , Y_{jk} is the j th constituent of the location of male mayfly k at time t , c_2 is formerly a well-defined attraction constant, g is the gravity coefficient, r is an accidental value $\varepsilon [-1, 1]$, and rmf is the Cartesian space among female and male mayflies. fl is a random walk coefficient in the instance of a female not being attracted to a male $fl^* r = fl_0 \times \delta^{itr}$. Here, it and δ are two previously well-defined variables.

3.2.3 Crossover between mayflies

A crossover operation is carried out by determining the first selection of a male mayfly, followed by the identification of a female mayfly. The selection process is based on the fitness value of the animals, with the best male breeds being paired with the best female breeds. Equations (10), (11) illustrate the offspring that are created as a result of crossover.

$$offspring\ 1 = r_{of}^* male + (1 - r_{of})^* female \quad (10)$$

$$offspring\ 2 = r_{of}^* female + (1 - r_{of})^* male \quad (11)$$

Both a random selection and a selection based on the fitness function are both viable options. In this context, the male stands for the male parent, while the female stands for the female parent. r_{of} is a random significance within a particular area. At the beginning, the velocity of the offspring is supposed to be zero.

3.2.4 Mutation of offspring

The selection of a randomly distributed energy that is ordinarily distributed as offspring that are accessible for mutation is done to circumvent the issue of premature network collapse. It is necessary to perform mutations between freshly created offspring to improve the capacity of algorithm for exploration. Offspring are capable of being altered and may be administered using Equation (12) as below:

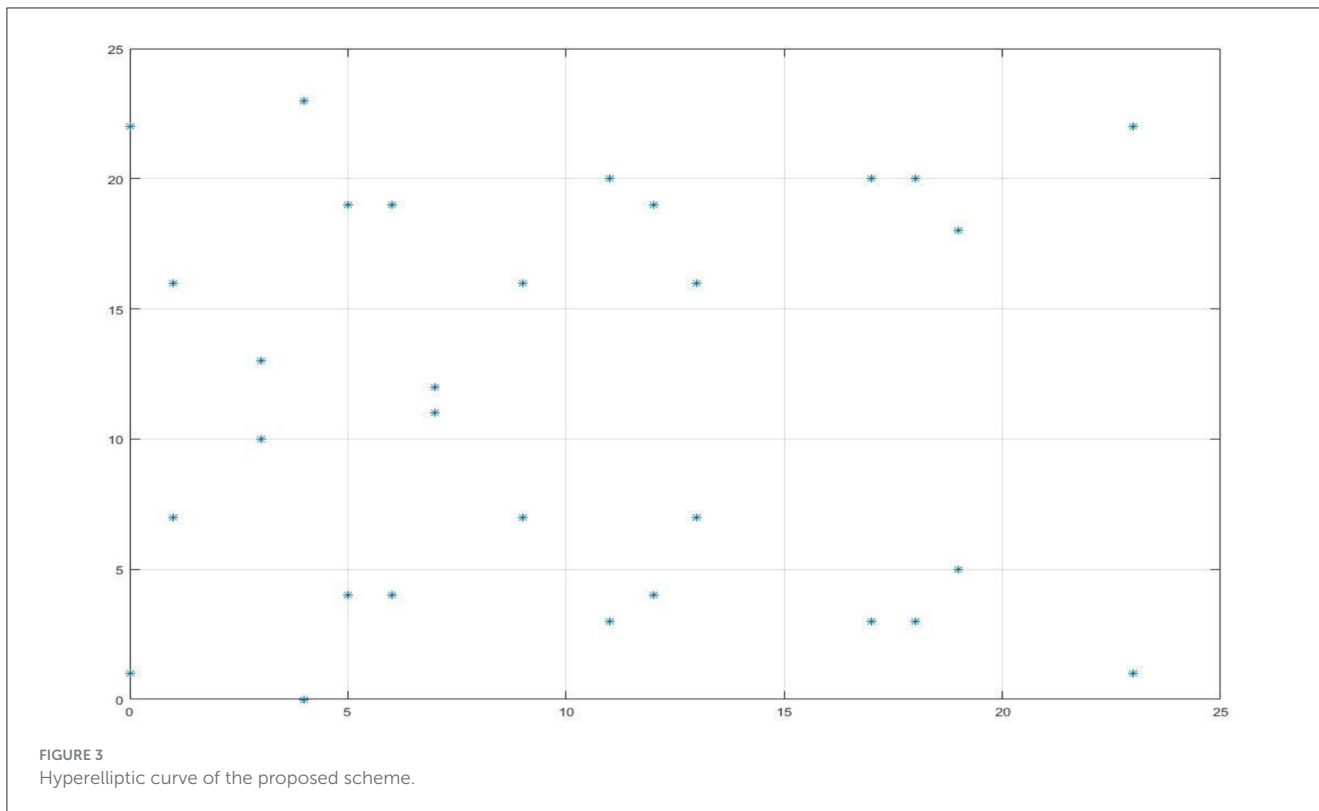
$$offspring_n^* = offspring_n + \eta N_n(0, 1) \quad (12)$$

Here, η is the normal deviation value of the standard distribution and $N_n(0, 1)$ is a typical normal distribution with mean =0 and variance =1. Moreover, male and female represent the parents. It is worth revealing that the preparatory speed of an offspring is expected to be zero.

4 Description of proposed method

Encryption of the data in communication under HECC cryptography with the help of MOA is developed in this study. Usually, keys are generated based on the selection of random prime numbers in the population space.

While selecting this random number as a key value, there are a lot of limitations with respect to attacks, and thus, an inherent solution to select this random coordinate value is developed using



MOA optimization. The proposed method contains eight phases: generating system parameters, encoding the message, mapping to the elliptic curve, encrypting the mapped points, decryption phase, and decoding phase. The main focus of this study is to select an optimal value from the curve for building an efficient key structure using MOA optimization. From Figure 2, it is understandable that if one user desires to direct a message to an additional user. The message is obtained from the first user, and it is then mapped to the curve using a private key. Private key selection is made optimal using the MFO algorithm, and then, the private key is added with the message encrypted to generate public key. The public key is a shared key between two users based on key agreement protocol. The second user receives the message, maps the points obtained to the curve eliminates the key bits, and separates the message bits to be decrypted. The decrypted message is then converted to plaintext to get the original message. Similarly, the second user also sent a message to the first user in the same way using the MO-HECC cryptography for secured communication.

4.1 Generating system parameters

In this phase, the generation of shared secret keys is done using optimization. This key is recommended to encrypt mapped points on an elliptic curve. The contributor needs to generate a mutual key in order to encrypt mapped points on an elliptic curve. Using this private key, a public key is generated, and both the sender and receiver approve on shared key. Initially, sender and receiver accept on a same elliptic curve with a generator point G . The largest prime number is selected using optimization denoted as d and used as

TABLE 7 MO-HECC key size comparison.

Key values	Key size (bytes)
3	1
139	1.5
1,009	1.8
9,973	2
10,000,019	3.5
9,999,999,929	5

the private key and calculates a point $P = dG$ as the public key. A number is said to be a prime number if it fulfills the situation.

- 1) $d - 1$ has a large prime factor r
- 2) $d + 1$ a large prime factor S
- 3) $r - 1$ have prime factors of t

Hence, the Fitness function to assess a large prime number can be communicated as exposed in Equation (13):

$$Fitness\ function\ Max\ \{f(x(t))\} = |(r_d + s_d + t_d)| \quad (13)$$

Fitness function $f[x(t)]$ is assessed to clarify excessive individual value. Equation (16), r_d denotes a large prime factor of $(d - 1)$, respectively. S_d represents a large prime factors of d . Similarly, t_d represents a large prime factor of r_d , respectively. The key generation algorithm is presented in Table 1.

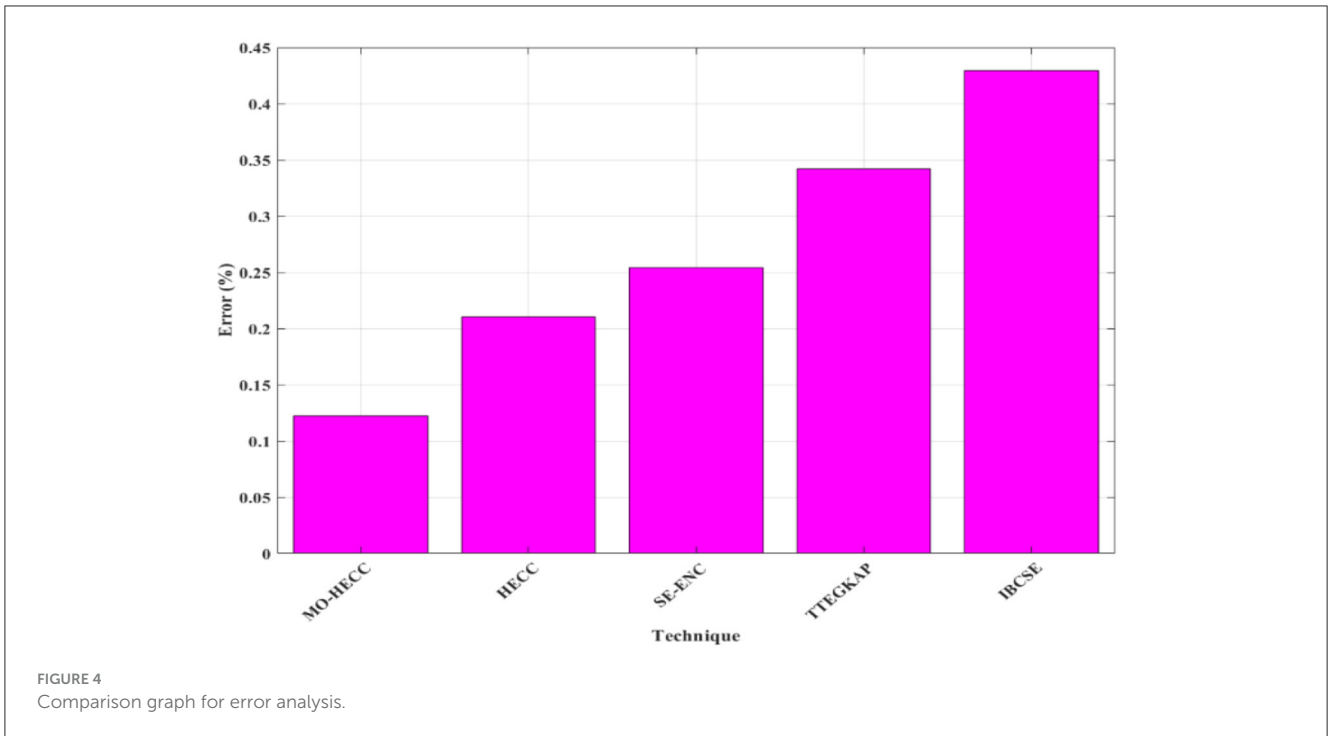


FIGURE 4 Comparison graph for error analysis.

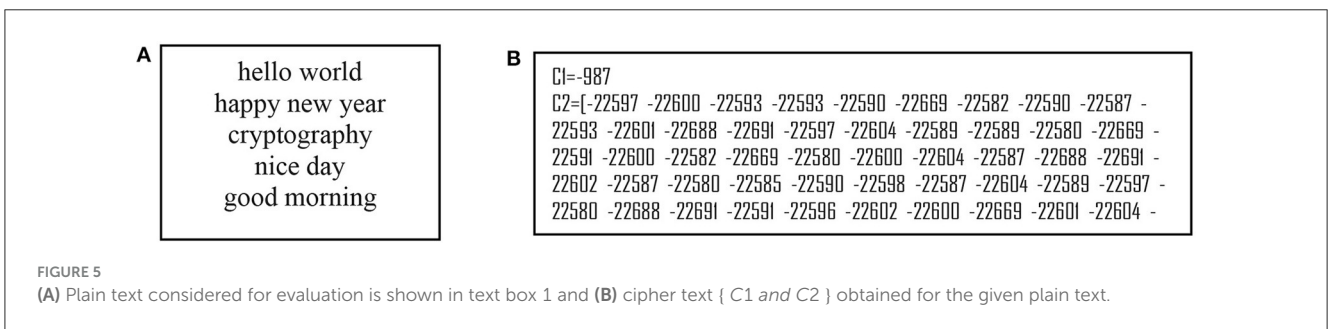


FIGURE 5 (A) Plain text considered for evaluation is shown in text box 1 and (B) cipher text { C1 and C2 } obtained for the given plain text.

4.2 Encryption phase

In this phase, a random integer e is selected such that $e \in \mathbb{Z}_p$ and computed $R = eG$. From the two parts of the message $P(x_1, y_1)$ and $Q(x_2, y_2)$, two coordinates are obtained, and it was performed with addition for transmission and R results (x_3, y_3) . The message is obtained and divided the block into N characters and B blocks. These blocks are reduced to 8-bit code after separating it into two cipher texts P and Q . These P and Q are performed in addition to obtaining a private key. After obtaining this key, Q is then encrypted along with the private key to produce an encrypted message. After performing encryption, the encrypted values are then recorded to an elliptic curve to make the algorithm more secure from encryption attacks. The message encoding algorithm is shown in Table 2.

4.3 Mapping to hyperelliptic curve

Mapping a message to HEC means that it must satisfy the curve equation. In the proposed method, each block from the encoded

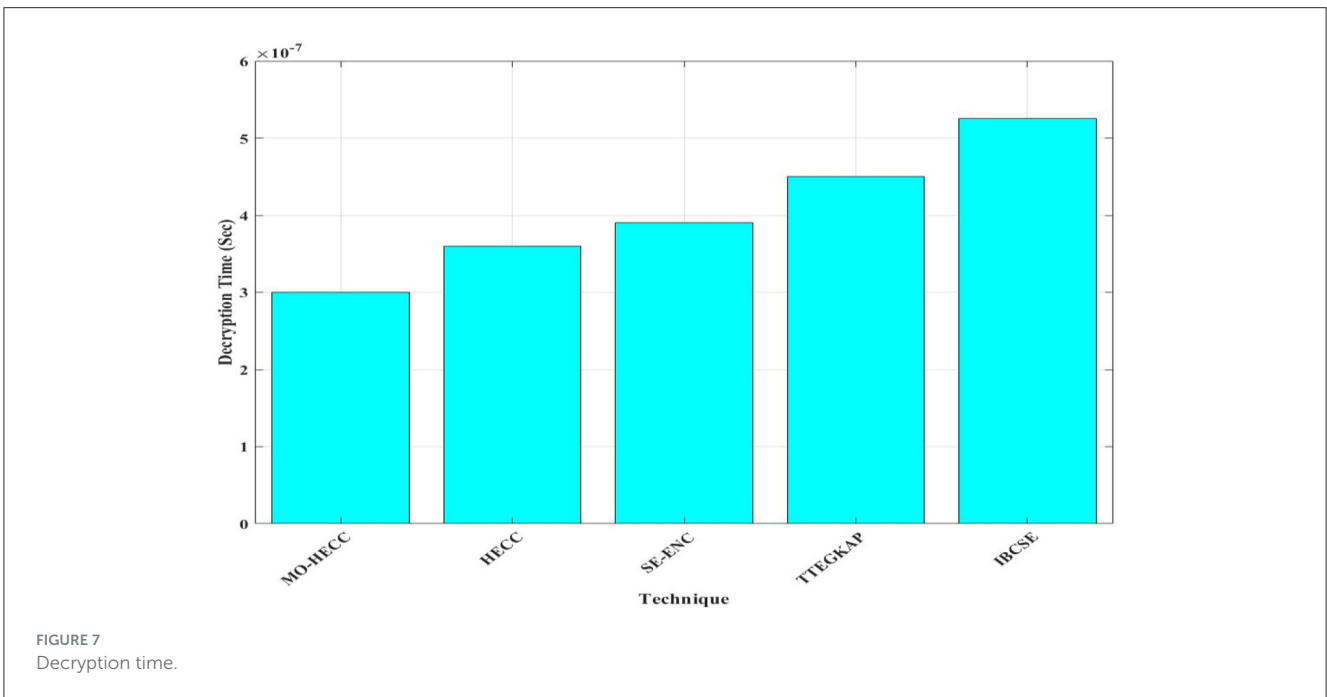
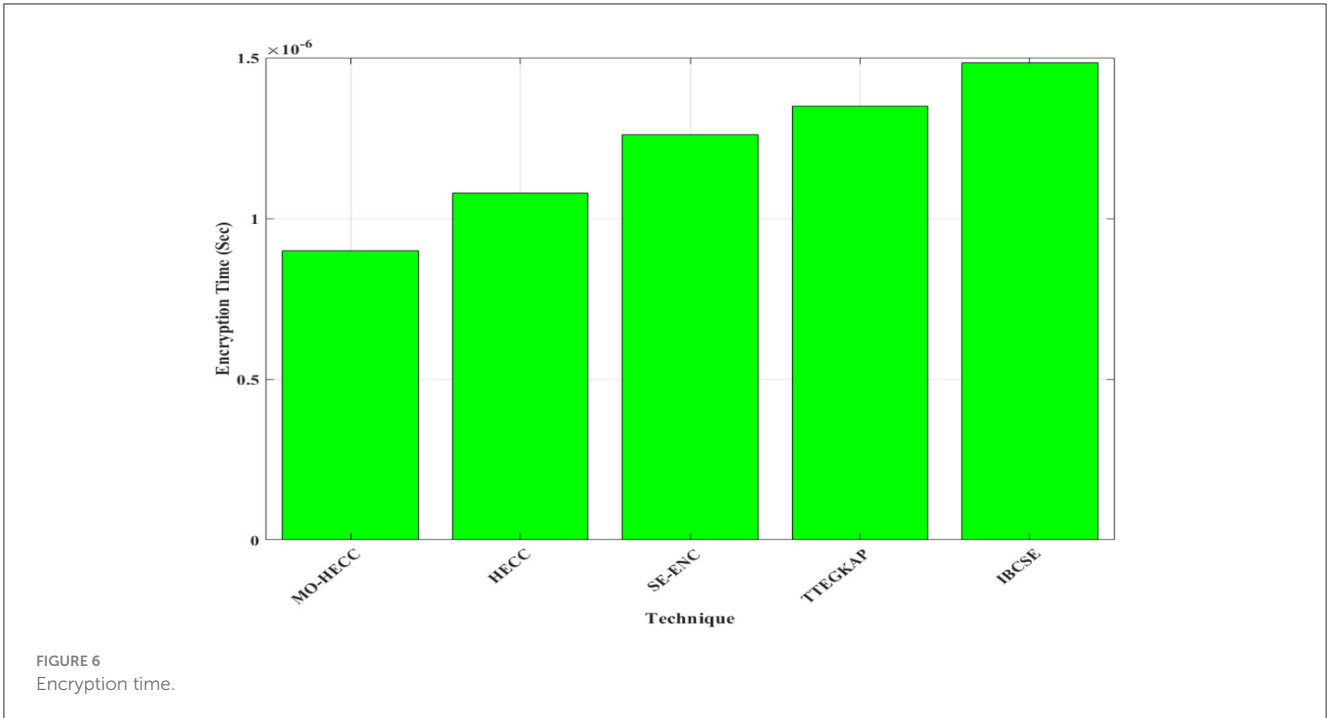
message, the block contains a binary value converted to a decimal value. Map R to s via Mumford's illustration for the points of $R(x_3, y_3)$. The function can be given as in Equation (14).

$$\theta : J(F_p) \rightarrow F_p \ni (e, n) = s \tag{14}$$

From Equation (14), F_p represents the function field and $J(F_p)$ represents a reduced divisor. The function θ is defined by:

$$\theta(R) = \begin{cases} 1, & \text{if } R = [x^2 + ax + b \text{ mod } p] \\ 0, & \text{if } D = [1, 0] \end{cases} \tag{15}$$

From the Equation (15), the value obtained from is evaluated for different a and b values until the result becomes 1. Each value gets mapped to the elliptic curve, respectively. The messages are attained from the user and plotted directly to the elliptic curve over a finite field. In this finite field, a table of operations is performed, and then, these mapped points are encrypted to an encoded stage. This results in a pair of cipher coordinates. It is mapped to the curve and satisfies the condition as explained in the next section.



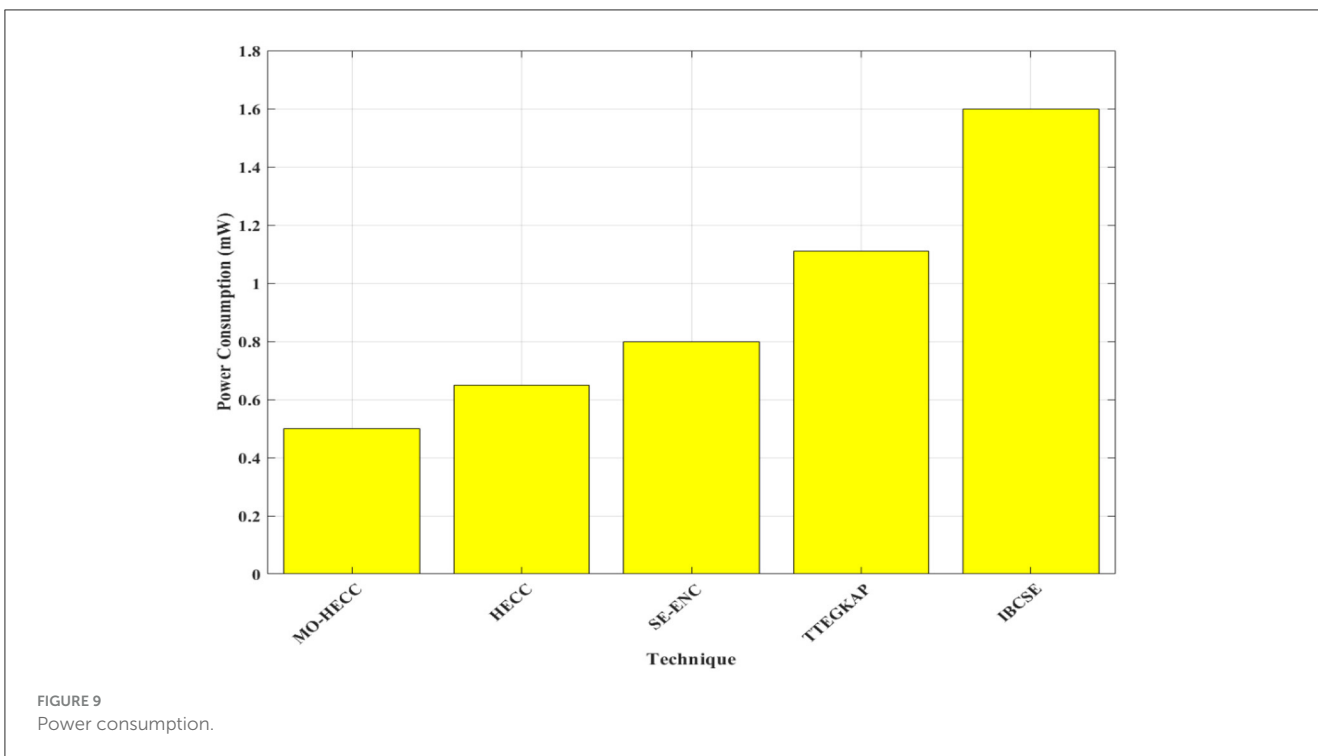
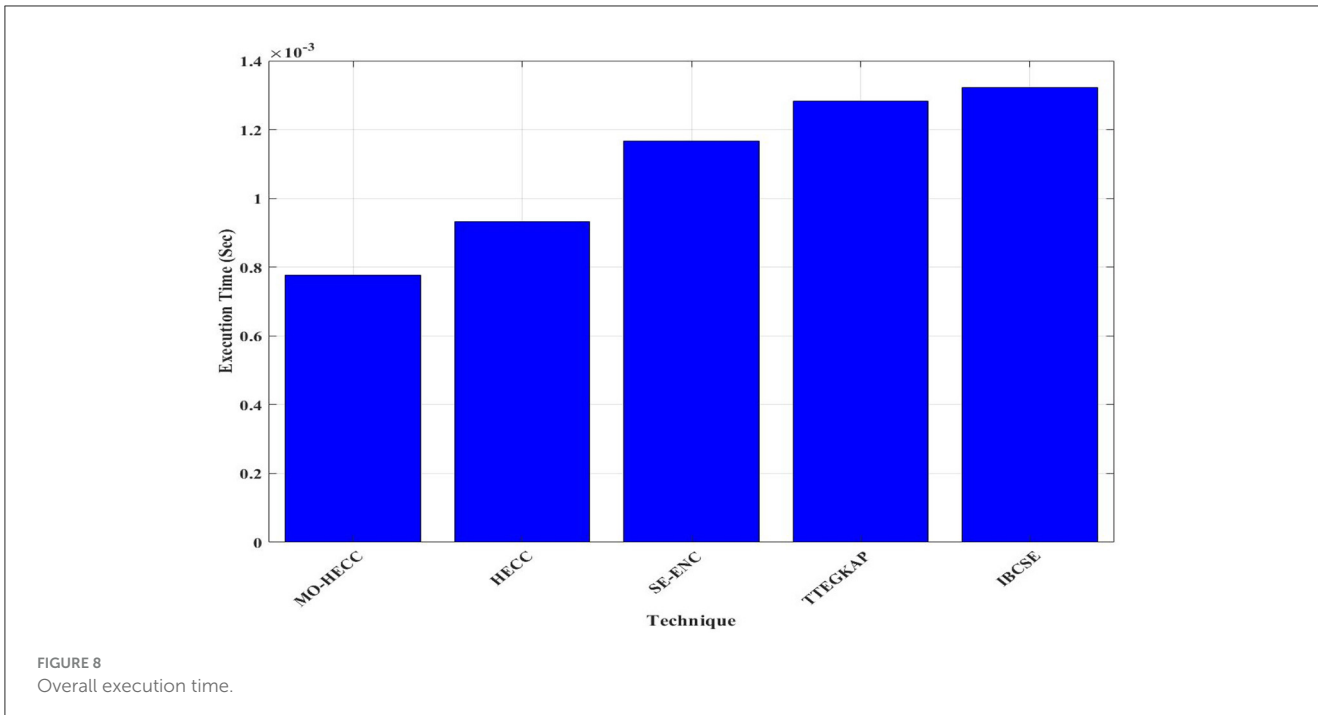
4.4 Encryption on mapped points

Mapping points to an elliptic curve is completed, and it must be qualified to be multiplied with the private key to advance the elliptic curve discrete logarithm problem (ECDLP) as hard as possible as shown in Table 3. Each point that is mapped is added with a shared key generated. Therefore, it is made cryptographically hard to regain mapped points without knowing the shared key. Thus, the process of adding a shared key with all mapped points

is repeated. Similarly, after encryption, the message is transmitted to the other party.

4.5 Decryption phase

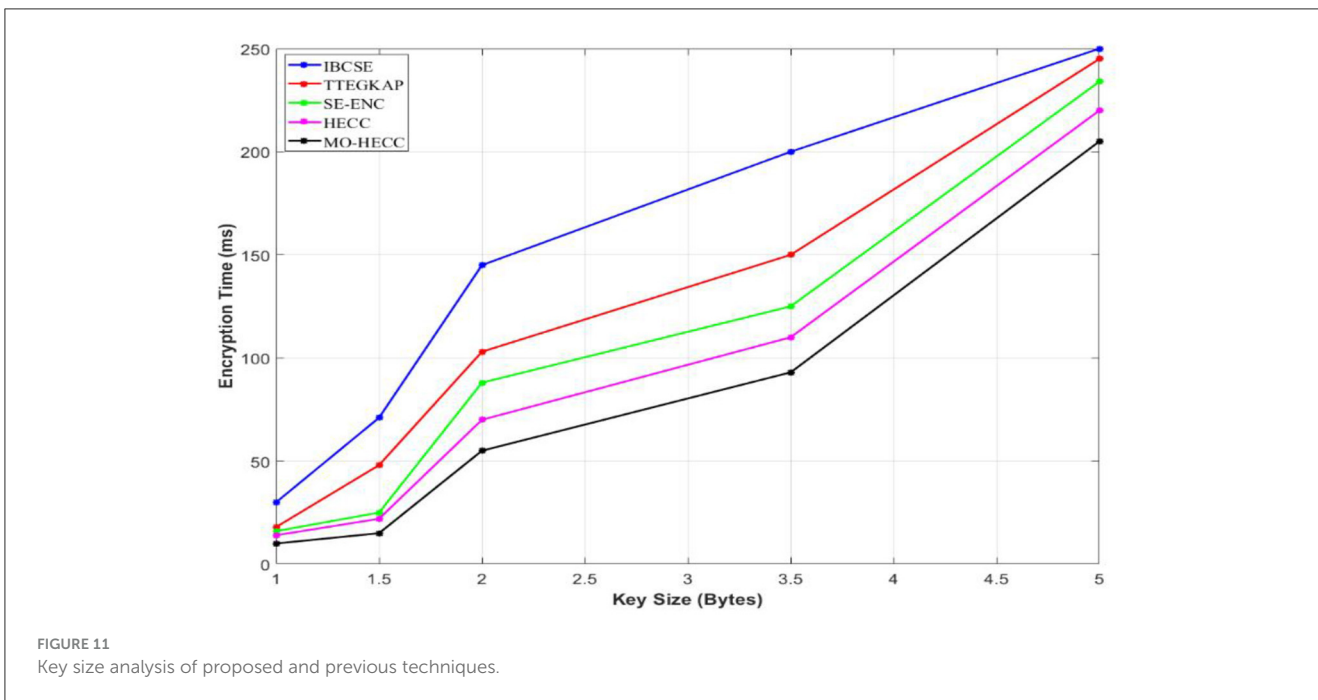
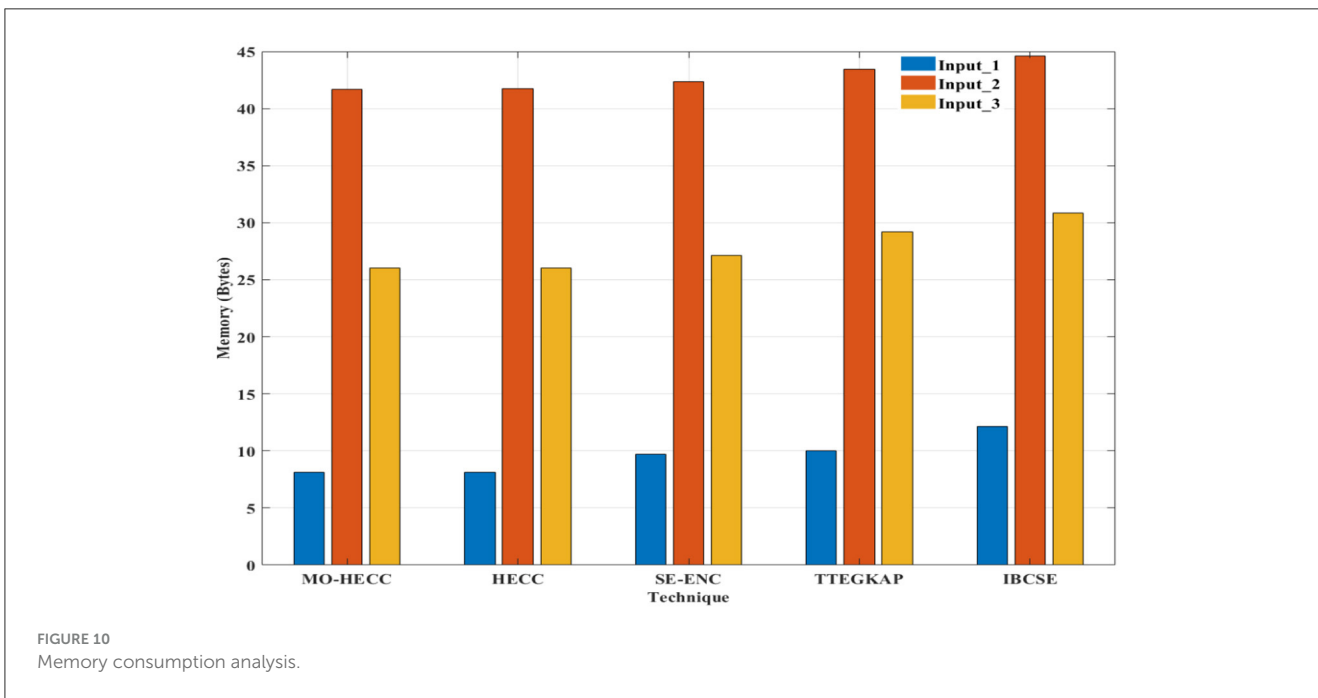
The reverse of the previous phases is done in the following phase. In the proposed method, to decrypt the encrypted points the recipient must subtract the shared key similar to the encryption



phase. After obtaining a set of mapped points, these points are obtained in the form of two pairs denoted as x and y . Here, y is only used in mapping the points to an elliptic curve and x left with a decoded part. Convert x to binary value, and the inverse process of transmission side was performed. The description using a public key is shown in Table 4.

4.6 Converting decoded data into plaintext

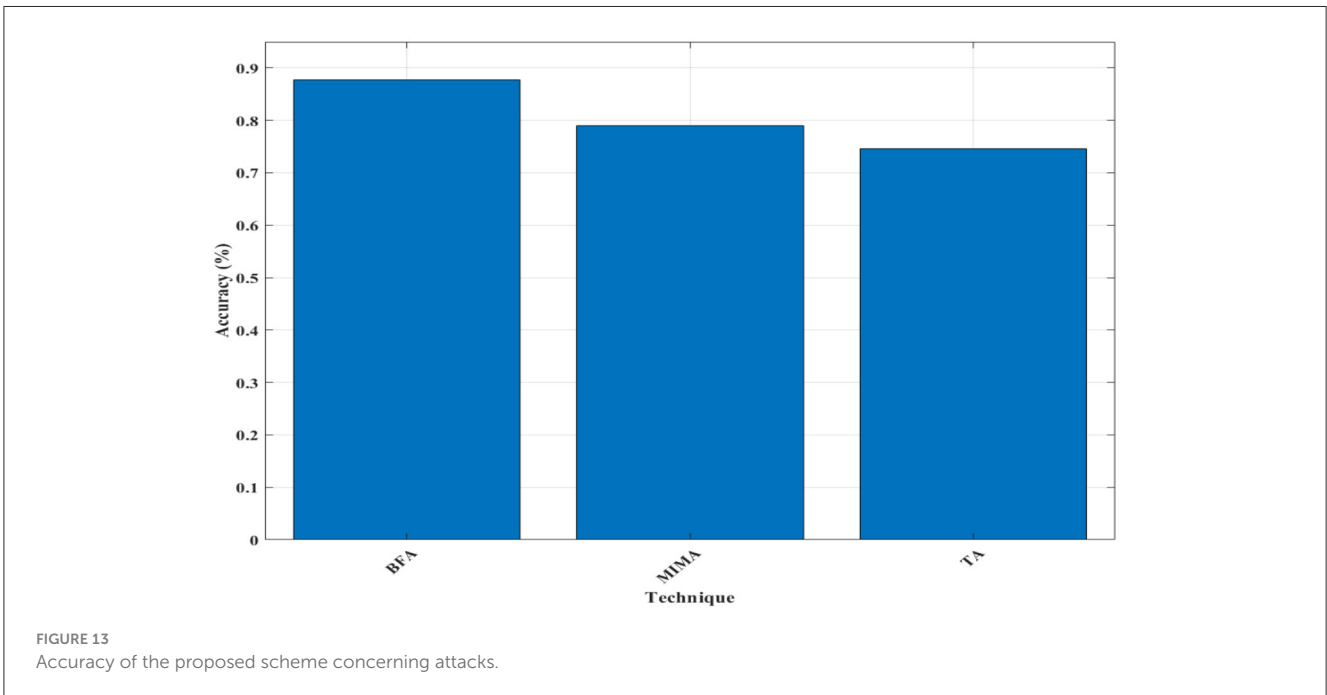
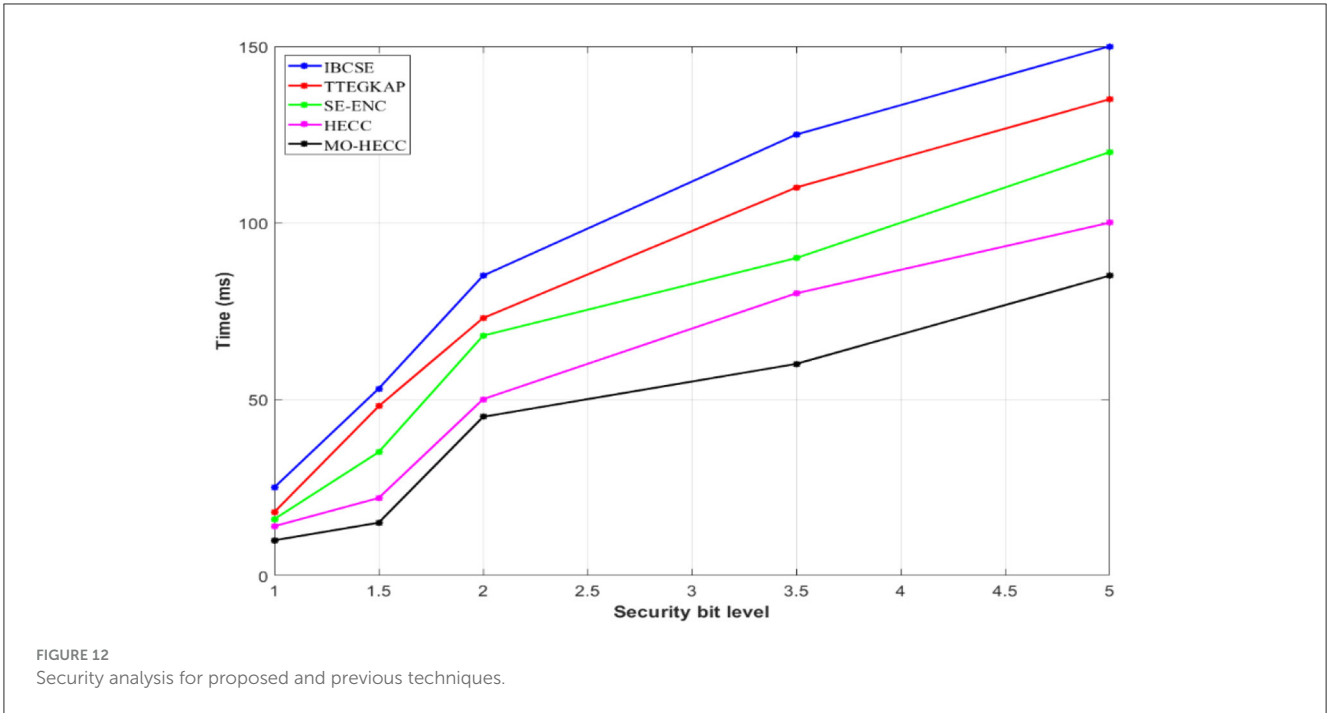
The last step is concerned with transforming the binary values into the characters that correspond to those values. It is the plaintext message M that is represented by these characters. It is possible for the decryption method to successfully recover the



message M if the access matrix of the cipher text is satisfied by a set of attribute secret keys that are currently in possession of a particular user. Finally, the message transmitted via the transmitter gets decrypted with the utilization of a shared key. The algorithm to convert binary values to plain text is depicted in Table 5.

4.6.1 Process of the proposed algorithm for optimization

The MFO procedure shown in Table 6 is recommended to optimize the d as it is key pairs in the article. Primarily, the random parameters are produced (d, e) pairs toward key parameters.



Step 1: Generate random parameters and integer values selected from the elliptic curve. Perform computations on message bit by partitioning it with respect to blocks.

Step 2: Optimal selection of the private key is set as an objective function, and the result is obtained as the best random prime number from the curve; then, message bits are subjected to a map in the curve and process is repeated until the optimal value is found.

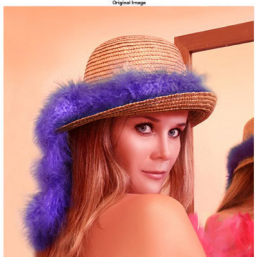
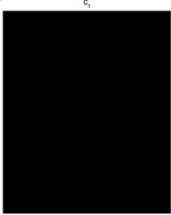

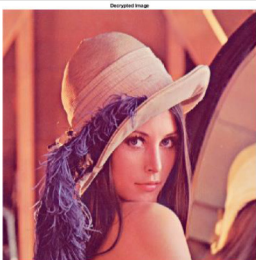

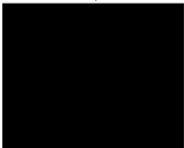

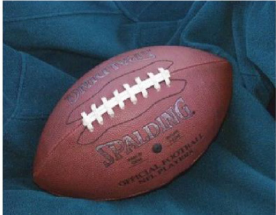
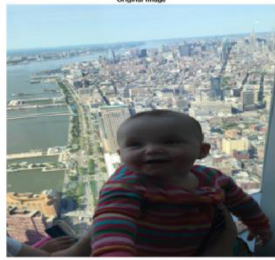



Step 3: Messages are mapped to the elliptic curve, and this process is repeated until are points are mapped to the curve.

Step 4: Mapped points in the curve are encrypted using the public key. The public key is attained by performing operations taking the private key and random integer as input.

Step 5: Encrypted points satisfy the curve equation to prevent encryption attacks. The encrypted points are transmitted via the transmitter to the other end user. This process is repeated for all message transmitters.

Step 6: On the other end, the user obtains a message, and the user has a public key for decryption. The message is mapped to a curve and subtracted with key points.

TABLE 8 Image encryption using MO-HECC.

S. no	Original image	Encrypted image	Decrypted image
1.	 <p>Lena image</p>	 	
2.	 <p>Football image</p>	 	
3.	 <p>Baby image</p>	 	

NFL Football (reproduced from FutureNJGov via Wikimedia Commons, licensed under CC BY- SA 3.0 File:Wilsonnflfootball.png - Wikimedia Commons).

Baby image is taken from Polygonal region of interest - MATLAB (mathworks.com).

Lena (reproduced from Roberto Bittencourt, CC BY-SA 2.5 BR, via Wikimedia Commons).

TABLE 9 Image encryption time analysis result.

Dimensions of the image	Size of the image	Encryption time				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	3.000622	2.8140	2.6726	2.5013	1.8021
256*320	25.8 KB	0.915505	0.8790	0.8588	0.8496	0.6431
700*600	793 KB	2.8848	2.6658	2.5655	2.4564	2.2741

Step 7: After subtracting from key points, message points are obtained that are decrypted to get the binary value of the message.

Step 8: Binary value is then converted to characters to obtain the original message.

5 Experimental results

Results are evaluated by implementing proposed MO-HECC and previous algorithms such as the HECC system, Se-Enc (Secure and efficient encoding scheme using ECC) Ternary tree-based group key agreement protocol over elliptic curve for the dynamic

group (TTEGKAP) and IBCSE-identity-based combined signature and encryption. This algorithm is implemented in MATLAB of Intel® Core™ i5-3330S CPU@2.70 GHz with installed memory 8 GB and 64-bit operating system.

5.1 Performance analysis of MO-HECC algorithm

When analyzing cryptographic systems, several metrics are crucial for assessing their efficiency and security. These include error analysis, encryption time, decryption time, execution time,

TABLE 10 Image decryption time analysis result.

Dimensions of the image	Size of the image	Decryption time				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	1.309932	1.48018	1.5013	1.51531	1.6122
256*320	25.8 KB	0.456	0.542	0.578	0.601	0.756
700*600	793 KB	1.9095	2.0183	2.0513	2.1122	2.3410

TABLE 11 Total execution time.

Dimensions of the image	Size of the image	Total execution time				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	5.375	6.178	6.624	7.0122	7.53124
256*320	25.8 KB	3.0127	3.1271	3.2741	3.4056	4.1098
700*600	793 KB	6.4695	6.6785	6.8031	7.1081	8.1221

and power consumption. Error analysis in cryptography typically involves assessing the reliability and robustness of cryptographic algorithms under various conditions. Encryption time refers to the amount of time it takes to convert plaintext into ciphertext using a cryptographic algorithm. Decryption time measures how long it takes to convert ciphertext back into plaintext. Execution time in cryptography refers to the total time required for any cryptographic operation, which might include key generation, encryption, and decryption. It is critical in scenarios where real-time or near-real-time processing is necessary. Power consumption is especially critical in mobile and embedded systems where battery life is limited.

Comparative results are obtained for the following metrics.

5.1.1 Accuracy

The capacity of a security model to properly decrypt the data is a test that determines the accuracy of the model. The computation of the percentage of true positives and true negatives was reviewed in each instance to get an estimate of the correctness of a security model. From a mathematical point of view, it may be expressed as the provided Equation (16).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{16}$$

Here, TP denotes true positive, TN represents true negative, FP denotes false positive, and FN is used to show false negative value.

5.1.2 Error analysis

The evaluation of correctness is one method for demonstrating the effectiveness of the system. Error analysis, on the other hand, is used to forecast mistakes that may occur during the process of decryption by using a security model. The systematic nature of errors, which leads to a reduction in the overall efficiency of the system, makes it imperative that they be reduced as much as possible.

5.1.3 Encryption time

Encryption time is defined as the time taken by a processor to completely encrypt the given plain text into cipher text. Encryption time is calculated for both public key encryption and private key encryption; these results are added to give total encryption time.

5.1.4 Decryption time

Data are pursued by the user only when it is in the form of plain text. The time taken by a technique to convert the obtained cipher text to binary value and then into plain text is defined as decryption time.

5.1.5 Execution time

The amount of time that it takes for an application to operate through its whole and produce relevant results is referred to as the execution time of a processor. The sum of the entire amount of time required for encryption and decryption by a method on the assessment platform is used to calculate this value.

Figure 3 shows a hyperelliptic curve mapped over a finite field. Graph generated for a random prime number 29 and random integer of 26 concerning coordinates (x_1, y_1) and (x_2, y_2) . By using the MFO optimization technique, the random prime number selection was made and assumed as a private key. Additionally, a random number was selected to encrypt the message data.

From Table 7, it is understandable that by varying key bit values to a certain extent key bit size gets increased. This shows the security level of the MO-HECC cryptography. Message is encrypted in the form of string and converted to ASCII code then to binary form in order to measure its length. Analysis was performed at the key side to measure them in the binary format from its primary number format.

Similarly, in error analysis, as shown in Figure 4, MO-HECC gives an error value of 12.5% which was comparatively lower than other previously existing methods. The HECC method gives an error value of 21% and SE-ENC produces 25% of error. TTEGKAP

TABLE 12 BER analysis result.

Dimensions of the image	Size of the image	BER value				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	0.003506	0.003694	0.003656	0.003757	0.003887
256*320	25.8 KB	0.003436	0.003531	0.003559	0.003648	0.003729
700*600	793 KB	0.003506	0.003566	0.003671	0.003679	0.003781

TABLE 13 MSE parameter results.

Dimensions of the image	Size of the image	Mean square error (MSE)				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	1.95E-24	1.95E-23	1.95E-22	1.95E-21	1.95E-20
256*320	25.8 KB	5.14E-25	5.14E-24	5.14E-23	5.14E-22	5.14E-21
700*600	793 KB	1.96E-24	1.96E-23	1.96E-22	1.96E-21	1.96E-20

technique gives an error value of 34%, and IBCSE produces a 42.5% error.

For evaluation purposes, let us consider some example words as plain text. It is given as shown in the text box and the key parameters are generated using a private key of value = 23. With the help of plain text and key values, cipher text is generated. Cipher text is shown in text box 2 of Figure 5.

During the analysis of encryption time, MO-HECC results with minimum time duration and thus compared with existing methods as shown in Figure 6. MO-HECC method gives 0.8×10^{-6} , and other existing algorithms such as HECC, SE-Enc, TTEGKAP, and IBCSE take an encryption time of 1.1×10^{-6} , 1.25×10^{-6} , 1.3×10^{-6} , and 1.45×10^{-6} , respectively. This was because of the optimal key selection procedure and efficient curve mapping procedure of the proposed scheme. Unless other methods follow the benefits of conventional methods and thus evaluate key using tree protocol and traditional random selection method, to improve the encryption security level optimization technique was introduced in this proposed scheme. As shown in Figure 7, the decryption time of the MO-HECC scheme was measured to be 3×10^{-7} s as it was found to be low due to its effective way of mapping the obtained message to the graph and decrypting it to get the original message. The HECC method takes a decryption time of 3.5×10^{-7} s and the SE-Enc method 3.9×10^{-7} s, respectively. This was due to its complexity in the decryption of messages via different algorithms. TTEGKAP method takes a highly long decryption time 4.5×10^{-7} , and the IBCSE method exhibits 5.1×10^{-7} s.

From Figure 8, the execution time for the MO-HECC algorithm 0.75×10^{-3} s and the HECC algorithm executes the cryptography in 0.9×10^{-3} s, which was a little higher than MO-HECC. Other conventional methods like the SE-Enc method take 1.15×10^{-3} s, the TTEGKAP method takes 1.3×10^{-3} s, and IBCSE takes a long period of 1.3×10^{-3} s that was comparatively higher than other methods.

From Figure 9, the power consumption of different techniques is plotted. MO-HECC scheme consumes less power compared to previous methods within the range of 0.5 mW and HECC scheme

consumed 0.65 mW power. SE-Enc scheme consumed a power value of 0.8 mW and TTEGKAP technique exhibited 1.1 mW and the IBCSE method exhibited 1.6 mW of power, which was comparatively higher than previous algorithms.

Memory consumption analysis results are plotted as a graph and presented in Figure 10. From this Figure 10, it is understandable that the proposed MOHECC model consumed less memory than the previous method. The HECC method consumed a comparatively higher volume of memory than the MOHECC method. The se-Enc method grants more memory for evaluation due to its security property and TTEGKAP method consumed more memory than all other methods and the IBCSE method consumed maximum memory due to its collaborative property.

Figure 11 shows that encryption time increases concerning key size, it is understandable that the proposed scheme takes minimum encryption time with increased key size. Key bit size varies between 1 and 5 bytes concerning encryption time that varies from 0 to 250 ms. In this selective range, the proposed model MO-HECC varies from 0 to 200 ms for 1.5–5 bytes of key size. Comparatively, for other algorithms the encryption time increased concerning key sizes, such as HECC, Se-Enc, TTEGKAP, and IBCSE techniques exhibit encryption times of 220, 230, 240s, and 250 ms, respectively, this happened due to the complexity property of previous methods in terms of cryptography. Figure 12 shows that security analysis for different techniques was done by presenting decryption time vs. key size level. The key size of the techniques increases which in turn increases decryption time. Comparing to other techniques, MO-HECC technique exhibits 10, 11, 48, 60, and 70 ms of decryption time for key sizes of 5-bit, 3.5-bit, 2-bit, 1.5-bit, and 1-bit, respectively.

In Figure 13, accuracy of MO-HECC is measured and plotted in the graph, accuracy was tested concerning different attacks such as brute-force attack (BFA), man-in-the-middle attack (MIMA), and timing attack (TA). While attacked by plain text attacks such as BFA and MMA, the MO-HECC scheme gives an accuracy of 87 and 79%, respectively. After a timing attack, MO-HECC generates 75% of accuracy. Compared to previous methods, the proposed

TABLE 14 SNR value analysis.

Dimensions of the image	Size of the image	Signal-to-noise ratio (SNR)				
		MO-HECC	HECC	Se-Enc	TTEGKAP	IBCSE
512*512	768 KB	4.13648	4.12790	4.12356	4.11788	4.10843
256*320	25.8 KB	3.023805	3.023802	3.023753	3.021765	3.02002
700*600	793 KB	2.696222	2.696102	2.695912	2.6958213	2.583941

scheme performs wisely for security analysis and thus results in high efficiency.

5.2 Performance analysis of MO-HECC for image encryption

The performance of the proposed algorithm for image encryption is presented in the Table 8. The image encryption and description results are presented in Tables 9, 10. The total execution time is presented in Table 11. The BER, MSE, and SER results are shown in Tables 12–14.

6 Conclusion

In this study, a cryptographic method was presented using the benefits of ECC and MFO. Previous methods secure data during communication via public and private keys. The presented method uses a hyperelliptic curve to secure data from various encryption attacks. MO-HECC starts by initializing required parameters, especially random integers for performing computations to find coordinates. The method then evaluates the fitness function of the MFO algorithm to predict the best solution for selecting a random prime number to be used as the private key. Private key selection was made optimal to increase the accuracy of the method. Then, the message is encrypted, and its coordinates are mapped in the curve along with key parameters. The key value and message are then encrypted, and this encryption was made hugely typical to make it secure communication. Thus, the other user receives this computed message and starts decrypting. During decryption, the points are again plotted to the curve and eliminated for key points. Key points are eliminated to obtain message bits; thus, message bits are decrypted to a binary value. Binary values are then converted to plain text for the ease of the user. Thus, this scheme was implemented in the MATLAB platform, and the results are evaluated and compared to previous methods. It has been clearly visible that the proposed MO-HECC scheme overwhelmed the existing methods and is suitable for real-time applications. Comparison with other optimization methods like PSO, GA, and DE on constrained real-parameter optimization tasks may reveal specific limitations or areas where the MA algorithm may not perform as effectively. Explore the integration of MO-HECC with other cryptographic systems to create hybrid encryption schemes for enhanced security. Other future direction is applying optimization techniques to improve the efficiency and speed of MO-HECC encryption and decryption processes.

Data availability statement

The original contributions presented in the study are included in the article/supplementary material, further inquiries can be directed to the corresponding authors.

Author contributions

RN: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Software, Writing – original draft, Writing – review & editing. MK: Conceptualization, Data curation, Investigation, Methodology, Software, Supervision, Writing – original draft, Writing – review & editing. GR: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Supervision, Validation, Writing – original draft, Writing – review & editing. AY: Conceptualization, Investigation, Software, Writing – original draft, Writing – review & editing. KA: Data curation, Formal analysis, Project administration, Validation, Writing – original draft, Writing – review & editing. AE: Conceptualization, Data curation, Formal analysis, Funding acquisition, Investigation, Methodology, Project administration, Resources, Software, Supervision, Validation, Visualization, Writing – original draft, Writing – review & editing. CR: Data curation, Methodology, Supervision, Writing – original draft, Writing – review & editing.

Funding

The author(s) declare that no financial support was received for the research, authorship, and/or publication of this article.

Conflict of interest

The authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

Publisher's note

All claims expressed in this article are solely those of the authors and do not necessarily represent those of their affiliated organizations, or those of the publisher, the editors and the reviewers. Any product that may be evaluated in this article, or claim that may be made by its manufacturer, is not guaranteed or endorsed by the publisher.

References

- Almajed, H. N., and Almogren, A. S. (2019). SE-ENC: a secure and efficient encoding scheme using elliptic curve cryptography. *IEEE Access* 7, 175865–175878. doi: 10.1109/ACCESS.2019.2957943
- Bhageerath Chakravarthy, G., Aditya Vardhan, R., Karthik Shetty, K., Mahesh, K., and Shitharth, S. (2021). “Handling tactful data in cloud using Pkg Encryption Technique,” in *4th Smart City Symposium* (Bahrain), 338–343.
- De Rango, F., Potrinio, G., Tropea, M., and Fazio, P. (2020). Energy-aware dynamic Internet of Things security system based on Elliptic Curve Cryptography and Message Queue Telemetry Transport protocol for mitigating Replay attacks. *Perv. Mob. Comput.* 61:101105. doi: 10.1016/j.pmcj.2019.101105
- Ding, S., Li, C., and Li, H. (2018). A novel efficient pairing-free CP-ABE based on elliptic curve cryptography for IoT. *IEEE Access* 6, 27336–27345. doi: 10.1109/ACCESS.2018.2836350
- Ellappan, V., Mahendran, A., Subramanian, M., Jotheeswaran, J., Khadidos, A. O., Khadidos, A. O., et al. (2023). Sliding principal component and dynamic reward reinforcement learning based IIoT attack detection. *Sci. Rep.* 13:20843. doi: 10.1038/s41598-023-46746-0
- Gueron, S., and Krasnov, V. (2015). Fast prime field elliptic-curve cryptography with 256-bit primes. *J. Cryptogr. Eng.* 5, 141–151. doi: 10.1007/s13389-014-0090-x
- He, D., and Zeadally, S. (2014). An analysis of RFID authentication schemes for internet of things in healthcare environment using elliptic curve cryptography. *IEEE Internet Things J.* 2, 72–83. doi: 10.1109/JIOT.2014.2360121
- Ibrahim, S., and Alharbi, A. (2020). Efficient image encryption scheme using henon map, dynamic S-boxes and elliptic curve cryptography. *IEEE Access* 8, 194289–194302. doi: 10.1109/ACCESS.2020.3032403
- Kapoor, V., Abraham, V. S., and Singh, R. (2008). Elliptic curve cryptography. *Ubiquity* 2008, 1–8. doi: 10.1145/1386853.1378356
- Khan, A. A., Kumar, V., and Ahmad, M. (2019). An elliptic curve cryptography based mutual authentication scheme for smart grid communications using biometric approach. *J. King Saud Univ. Comput. Inform. Sci.* 34, 698–705. doi: 10.1016/j.jksuci.2019.04.013
- Kumar, V., Ahmad, M., and Kumari, A. (2019). A secure elliptic curve cryptography based mutual authentication protocol for cloud-assisted TMIS. *Telemat. Informat.* 38, 100–117. doi: 10.1016/j.tele.2018.09.001
- Liu, Y., Chai, Y., Liu, B., and Wang, Y. (2021). Bearing fault diagnosis based on energy spectrum statistics and modified mayfly optimization algorithm. *Sensors* 21:2245. doi: 10.3390/s21062245
- Liu, Z., Seo, H., Großschädl, J., and Kim, H. (2015). Efficient implementation of NIST-compliant elliptic curve cryptography for 8-bit AVR-based sensor nodes. *IEEE Trans. Inform. For. Secur.* 11, 1385–1397. doi: 10.1109/TIFS.2015.2491261
- Mahmood, K., Chaudhry, S. A., Naqvi, H., Kumari, S., Li, X., and Sangaiyah, A. K. (2018). An elliptic curve cryptography based lightweight authentication scheme for smart grid communication. *Fut. Gener. Comput. Syst.* 81, 557–565. doi: 10.1016/j.future.2017.05.002
- Mahto, D., and Yadav, D. K. (2017). RSA and ECC: a comparative analysis. *Int. J. Appl. Eng. Res.* 12, 9053–9061.
- Mani, A. A., and Gopi, V. (2013). Secured broadband data access system in WiMAX. *Int. J. Manag. IT Eng.* 3:542.
- Mehrabi, M. A., Doche, C., and Jolfaei, A. (2020). Elliptic curve cryptography point multiplication core for hardware security module. *IEEE Trans. Comput.* 69, 1707–1718. doi: 10.1109/TC.2020.3013266
- Mrabet, A., El-Mrabet, N., Lashermes, R., Rigaud, J. B., Bouallegue, B., Mesnager, S., et al. (2016). “High-performance elliptic curve cryptography by using the CIOS method for modular multiplication,” in *International Conference on Risks and Security of Internet and Systems* (Cham: Springer), 185–198.
- Naresh, V. S., Sivaranjani, R., and Murthy, N. (2018). Provable secure lightweight hyper elliptic curve-based communication system for wireless sensor networks. *Int. J. Commun. Syst.* 31:e3763. doi: 10.1002/dac.3763
- Pan, W., Zheng, F., Zhao, Y., Zhu, W. T., and Jing, J. (2016). An efficient elliptic curve cryptography signature server with GPU acceleration. *IEEE Trans. Inform. For. Secur.* 12, 111–122. doi: 10.1109/TIFS.2016.2603974
- Rabie, O. B. J., Selvarajan, S., Hasanin, T., Alshareef, A. M., Yogesh, C. K., and Uddin, M. (2024). A novel IoT intrusion detection framework using Decisive Red Fox optimization and descriptive back propagated radial basis function models. *Sci. Rep.* 14:386. doi: 10.1038/s41598-024-51154-z
- Rawat, A., and Deshmukh, M. (2020). Tree and elliptic curve based efficient and secure group key agreement protocol. *J. Inform. Secur. Appl.* 55:102599. doi: 10.1016/j.jisa.2020.102599
- Sahoo, P. K., Jena, G., Chhotray, R. K., and Patnaik, S. (2013). An implementation of elliptic curve cryptography. *Int. J. Eng. Res. Technol.* 2, 1–8.
- Shitharth, S., Alshareef, A. M., Khadidos, A. O., and Alyoubi, K. H. (2023b). A conjugate self-organizing migration (CSOM) and reconcile multi-agent Markov learning (RMML) based cyborg intelligence mechanism for smart city security. *Sci. Rep.* 13:15681. doi: 10.1038/s41598-023-42257-0
- Shitharth, S., Manoharan, H., Shankar, A., Alswail, R. A., Pandiaraj, S., Edalatpanah, S. A., et al. (2023a). Federated learning optimization: a computational blockchain process with offloading analysis to enhance security. *Egypt. Informat. J.* 24:100406. doi: 10.1016/j.eij.2023.100406
- Sowjanya, K., Dasgupta, M., Ray, S., and Obaidat, M. S. (2019). An efficient elliptic curve cryptography-based without pairing KPABE for Internet of Things. *IEEE Syst. J.* 14, 2154–2163. doi: 10.1109/JSYST.2019.2944240
- Syed, S. A., Manickam, S., Uddin, M., Alsufyani, H., Shorfuzzaman, M., Selvarajan, S., et al. (2024). Dickson polynomial-based secure group authentication scheme for Internet of Things. *Sci. Rep.* 14:4947. doi: 10.1038/s41598-024-55044-2
- Wu, J., Liao, X., and Yang, B. (2017). Color image encryption based on chaotic systems and elliptic curve ElGamal scheme. *Sign. Process.* 141, 109–124. doi: 10.1016/j.sigpro.2017.04.006
- Zervoudakis, K., and Tsafarakis, S. (2020). A mayfly optimization algorithm. *Comput. Industr. Eng.* 145:106559. doi: 10.1016/j.cie.2020.106559
- Zhang, F., Zhang, Z., and Guan, P. (2020). ECC2: error correcting code and elliptic curve based cryptosystem. *Inform. Sci.* 526, 301–320. doi: 10.1016/j.ins.2020.03.069
- Zhou, Y., Li, Z., Hu, F., and Li, F. (2019). “Identity-based combined public key schemes for signature, encryption, and signcryption,” in *Information Technology and Applied Mathematics* (Singapore: Springer), 3–22.