A Cryptographic Algorithm using Polynomial Interpolations for Mitigating Key-Size Based Attacks

Jagpreet Kaur[1] and K. R. Ramkumar[2,*]

ARTICLE INFO

Article history:
Received: 14 July 2024
Revised: 8 February 2025
Accepted: 23 April 2025
Online: 20 July 2026

Keywords:
Lagrange interpolation equations
Polynomial Interpolation
Root-finding
Security-aware

ABSTRACT

One of the key factors to take into account when protecting wireless communication is confidentiality. A new route is necessary since current advancements will surpass traditional cryptographic techniques sooner than anticipated. Conventional security mechanisms suffer from various attacks. In this paper, we propose two novel approaches to enhancing data privacy in this scenario by utilizing polynomial interpolation. The first approach uses Bisection and Newton-Raphson methods separately, i.e., Interpolation-Based Cryptographic Algorithm (IBCA), and the second proposed work uses a hybrid of Bisection and Newton-Raphson methods, i.e., Bi-New, for determining the roots of the LaGrange polynomial. The proposed method includes unique elements such as non-linear interpolating polynomials, solutions to non-linear algebraic equations, and non-linear functions. Further, the proposed approach can withstand many attacks because it combines different non-linear methods and randomized variables. Additionally, a number of cutting-edge quantitative indicators, including time complexity, accuracy, memory use, key space analysis, key sensitivity analysis, etc., demonstrate the proposed algorithm's performance and security. It is evident from the simulation analysis that the proposed algorithm not only helps in ensuring data integrity but also outperforms AES and RSA algorithms on the benchmarks of memory consumption, number of digit change rate, time complexity analysis, etc.

1. INTRODUCTION

Cryptography plays a pivotal part in modern society by ensuring the confidentiality, integrity, and authenticity of digital communication and data [1][2]. It enables transactions security, protection of data, and confidentiality in an increasingly interconnected world. By employing encryption techniques, cryptography safeguards sensitive information from unauthorized access, mitigates cyber threats, and prevents eavesdropping or tampering during transmission. This is especially important for online banking, e-commerce, government communications, healthcare records, and more. Cryptography provides a foundation of trust in digital interactions, fostering secure communication and information sharing [3][4]. In a landscape rife with cyber attacks and digital vulnerabilities, cryptography serves as a fundamental tool to safeguard individuals, organizations, and critical infrastructure from malicious actors and breaches, maintaining the foundation of a secure digital environment [1].

Polynomial interpolation is a mathematical method employed to create a polynomial function that accurately fits a specified collection of data points [5]. The primary motivation behind polynomial interpolation is to approximate a continuous function or data set using a simpler polynomial equation that can be easily manipulated and analyzed. The goal of this process is to identify a polynomial that effectively represents the given data points. The fundamental idea is to create a polynomial equation that satisfies the condition of passing through each of the given data points [6].

Polynomial interpolation is increasingly favored in security, particularly in applications like secret sharing and cryptographic protocols, due to its distinctive advantages over alternative encryption methods. Its motivations are rooted in its mathematical robustness, founded in numerical analysis, offering a reliable basis for cryptographic operations. The use of information redundancy, as seen in Shamir's Secret Sharing, significantly boosts fault tolerance, allowing for data recovery even when some parts are compromised. Distributed security, as seen in threshold cryptography, spreads security across multiple entities, reducing single points of failure and central vulnerabilities. Polynomial interpolation often proves more efficient than traditional encryption, especially in identity-based encryption, simplifying key management. It also plays a crucial role in privacy preservation through zero-knowledge

[1]*Department of Computer Science and Engineering, Chitkara University Institute of Engineering and Technology, Chitkara University, Punjab, India*.
**Corresponding author:* K. R. Ramkumar; Email: k.ramkumar@chitkara.edu.in.

proofs, verifying data without revealing sensitive information. There are two primary types of polynomial interpolation: Newton's Interpolation and Lagrange Interpolation [7].

The significance of the Lagrange polynomial lies in its simplicity of implementation and calculation. It doesn't require solving a system of linear equations like some other interpolation methods, making it computationally efficient. Additionally, it provides a straightforward way to interpolate values between data points. Root-finding methods in cryptography utilize polynomial interpolation to address mathematical problems within finite fields. These methods involve translating cryptographic issues into systems of polynomial equations, where unknowns may represent secret keys.

To address the aforementioned concerns, the major contributions to the proposed work are stated as follows.

- This article states two novel techniques for encrypting messages. The encryption process incorporates three well-known root-finding techniques (Bisection, Newton-Raphson, and Secant), along with an analysis comparing their effectiveness.
- The proposed scheme comprises three steps: In the first part, an exponential polynomial is employed. The polynomial is constructed utilizing Lagrange Interpolation. Furthermore, interpolating data points are generated using a random sequence generator, and this polynomial acts as a secret key to encrypt data.
- In the second division, the polynomial, and the plain text are used for encryption. To effectively utilize the fundamentals of algebraic equations, we first need initial data points. A method has been proposed to calculate these values.
- In the third step, the root-finding operations are applied to find the root, which acts as a cipher text. Reducing computational complexity while maintaining the security of a cryptosystem is essential for effective implementation. Hence, owing to its paramount simplicity and low computational complexity, the simulation results showed a less computed time for encryption and decryption, making this scheme appropriate for deployment.

The above-mentioned steps lead the less encryption and decryption time due to the utilization of mathematical methods, which have far less computational complexity. Furthermore, these schemes can resist existing cyber-attacks. Additionally, a comparison with the standard algorithm was performed.

The remaining paper is structured in the following manner. Section 2 presents an overview of recently proposed security methods and describes the proposed algorithm for guessing the initial values to get the root. Section 3 presents the proposed security architecture. Section4 discusses the simulation results and finally, the conclusion is presented in Section 5.

2. LITERATURE REVIEW

In this section, we present the related work and the research gaps from them. We also aim to highlight the differences in polynomial interpolation when implemented in the contexts of AES and RSA. Further, we also give a comprehensive comparison of root-finding methods by discussing their advantages, challenges, and limitations in a polynomial interpolation-based approach.

2.1 Related Work

There is currently no workable method for implementing polynomial interpolation while maintaining user anonymity.

Huang et al. [7] presented an innovative access control method called LIDACM, which utilizes Lagrange interpolation to safeguard healthcare information in the cloud, thus guaranteeing secure access to personal health records (PHRs). The LIDACM limits access to protected health records (PHRs) by anyone who isn't authorized to see them, which makes it more difficult for hackers to steal sensitive data from the database. According to the results of the current study, the LIDACM successfully safeguards PHR data.

Chen et al. [8] offered a valuable solution to bridge this gap and enable this process. To achieve this, the authors first adapted the private polynomial interpolation into a private calculation of function values and then developed a compact private scalar product protocol. Afterwards Lagrange interpolation is used in conjunction with their scalar product technique to solve the initial issue.

Arora et al. [9] introduce an innovative key-sharing mechanism that employs Newton Interpolation. The proposed key-sharing method is meant to be used in situations where an explicit key-sharing mechanism is not feasible, such as when a secret symmetric key must be shared between communication nodes.

Dini and colleagues [10] investigated the detection of anomalies in communication networks. This paper details the development of a novel anomaly detection algorithm that makes use of polynomial interpolation and statistical evaluation. The unique approach is tested by applying it to a novel available dataset, EDGE-IIOTSET 2022, as well as on KDD99, UNSW-NB15, and CSE-CIC-IDS-2018, all of which are widely regarded as benchmarks in the scientific community. The experimental findings demonstrate that the suggested methodology is superior to state-of-the-art rule-based intrusion detection systems like SNORT and traditional one-class classifiers like the Extreme Learning Machine and Support Vector Machine models.

Protecting the privacy of data in the cloud relies heavily on secret sharing techniques. The authors in [11] develop new verification methods for Secret Sharing, which improves safety by identifying hostile individuals trying to recreate the secret through sham shares. Unlike conventional approaches, these algorithms have a small footprint. The Exponentiation Polynomial Root issue (EPRP) is a difficult

computing issue that is thought to be NP-Intermediate, and it is introduced in one method to further strengthen the security of the cryptographic protocol.

The research presented in the aforementioned studies highlights advancements in the use of polynomial interpolation-based cryptographic techniques for various applications. However, several research gaps and opportunities emerge:

i. Scalability in Healthcare Data Security: Huang et al.'s LIDACM is promising for healthcare data, but research should address scalability challenges in real-world, large-scale healthcare systems.

ii. Anonymity-Preserving Polynomial Interpolation: While Chen et al. achieved user anonymity in polynomial interpolation, further exploration of privacy-preserving techniques and their scalability is needed.

iii. Key-Sharing Mechanisms: Arora et al.'s key-sharing approach is innovative, but its applicability to diverse communication scenarios and potential vulnerabilities require further investigation.

iv. Anomaly Detection: Dini et al.'s anomaly detection method shows promise, but its robustness in complex and evolving network environments needs validation.

v. Verification Methods in Secret Sharing: While verification methods are improving, research should explore their integration into broader cryptographic protocols and assess their efficiency.

Addressing these research gaps can contribute to the advancement and wider adoption of polynomial interpolation-based cryptographic techniques, enhancing data security and privacy in various domains.

2.2 Difference from AES and Rivest–Shamir–Adleman (RSA) algorithms

Cryptographic algorithms employing polynomial interpolation are valuable for specific security needs, offering advantages distinct from AES (Advanced Encryption Standard) and RSA. The necessity for these algorithms stems from their unique capabilities. Table1 provides a concise comparison of the strengths and applications of polynomial interpolation, AES, and RSA [12].

2.3 Rationale behind using Root-findingMethods

Root-finding methods, also known as numerical methods for root approximation or root isolation, are techniques used to find the values (roots) of equations where the function equals zero. These methods are particularly useful when analytical solutions for equations are not readily available or computationally complex. Here are three common root finding methods [13].

i. Bisection Method [13]: The bisection method is a easy and dependable root-finding technique. It works by iteratively narrowing down an interval in which the root is known to lie. The method assumes that the function is continuous and changes sign within the interval. It continuously divides the interval in half to determine where a sign change occurs within the subintervals. This method persists until the interval is reduced to a satisfactory size or until an acceptable level of accuracy is reached. The bisection method guarantees convergence but might be slower compared to more advanced methods.

ii. Newton-Raphson Method [14]: The method, also identified as the Newton method, is an iterative root-finding algorithm. It begins with a preliminary estimate and sharpens that estimate by utilizing the tangent line of the function's curve at the current point. The next guess is obtained by finding the point where the tangent line intersects the x-axis. The method converges rapidly for well-behaved functions, but it requires the availability of derivatives and might not always converge if the initial guess is not chosen properly.

Secant Method [14]: The secant method is an iterative technique that is similar to the Newton-Raphson method but doesn't require the computation of derivatives. Instead, it approximates the derivative using two points that lie close to each other. The approach involved determines the root by identifying the x-intercept of the line that links these two points. The secant method is more versatile than the Newton-Raphson method in cases where derivatives are difficult to compute or involve excessive computational effort.

These root-finding methods are essential tools in numerical analysis and play a significant role in various fields, including engineering, physics, economics, and computer science. However, the choice of method depends on the nature of the problem, the properties of the function, and the availability of information such as derivatives [14]

Bi-New: Using a hybrid approach that combines the bisection method and the Newton-Raphson method can provide the advantages of both methods while mitigating their individual limitations. The hybrid approach aims to merge the fast convergence of the Newton-Raphson method with the dependable stability of the bisection method. It adapts its behavior based on the function's behavior, ensuring both safety and speed during the root-finding process.

2.4 Automation of Initial-guess Values in Root-Fining Methods

The description of root-finding methods and their advantages and disadvantages are detailed above. One must guess initial values to solve any of the methods above. Hence, instead of manually guessing the initial values, the proposed initial guess algorithm is:

INPUT: Nth degree Polynomial (Integer, Floating or Exponential)

OUTPUT: Guessing Points on Graph for same Roots.

Step1: Enter the function g(X) of any nth degree.

Step2: Based upon degree, perform the following function:

```
if degree >= 1 and degree <= 10:
    nth_root = degree / 2
elif degree > 10 and degree <= 15:
    nth_root = degree / 3
else:
    nth_root = degree
```

Step3: Calculate the root value of constant term by following way to find the initial guess value as:

constant_nth_root = nth_root√constant

Step 4: Formulated on the algorithm utilize the above guess value as:

```
if method= bisection
    initial_one_bi=-round (constant_nth_root)
    initial_two_bi = round (constant_nth_root)
if method= newton-raphson
    initial_newton = round (constant_nth_root)
if method= secant
    initial_one_sec= math.floor (constant_nth_root)
    initial_two_sec= math.ceil (constant_nth_root)
```

3. PROPOSED SECURITY ARCHITECTURE

Figure 1 illustrates the operational framework that utilizes Lagrange interpolation along with a root-finding algorithm. The proposed algorithm is segmented into two sections: the first segment is Generate Polynomial utilizing Lagrange Polynomial which acts as a Secret Key for both sender and recipient, in the second segment plain text is first fed into UTF-8 encoder to get the plain data. UTF-8 encoder is used to Unicode. A Unicode character can be translated into a specific binary string, and then binary string can be translated into decimal number. The main reason to use UTF-8 is its unique ability to encode characters in one-byte units. Besides, UTF-8 can support all of the readable ASCII characters, as well as the non-readable characters. Once the plain data is obtained, encryption is carried out using a Root-Finding algorithm, including methods like Bisection, Newton-Raphson, and Secant. For decryption, the Inverse Root-Finding algorithm is utilized, followed by running UTF-8 to retrieve the original plain text. Hence, two algorithms i.e., IBCA and Bi-New have been proposed based on this architecture and are described below:

3.1 Interpolation-Based Cryptographic Algorithm (IBCA)

The proposed IBCA initially calls the polynomial generation function, randomly generating the polynomial using Lagrange Interpolation and acting as a key. This is the polynomial that we have communicated to the receiver through a secure channel. The entire process of encryption and decryption is based on this polynomial. The constant coefficient of the polynomial is then subtracted from the plain text being taken; henceforth, a new scrambled polynomial is obtained. An initial value supposes X is computed by taking the nth root of the newly generated constant value, where n is the degree of the polynomial. Approximation roots for the given polynomial are generated using that initial calculated value. The calculated roots are then stored in a file and shared with the receiver. The flowchart for the scheme has been delineated below in Figure2.

3.1.1 IBCA-Encryption

INPUT: Nth odd degree exponential g(X) + 80 bits Plain Text (PT)

OUTPUT: Root Value

Step 1: PT is taken to be encrypted in the form of digits.

***Step 2*:** Obtain the randomized polynomial generated through the Lagrange Interpolation Method of odd degree up to the 13th order [15].

poly = lagrange(x, y)

*Step 3***:** Equate the PT with the obtained polynomial in Step 2. A new scrambled polynomial will be generated as follows:

```
f = poly−digit
poly_coefficient = Polynomial (f).coef
df = np.polyder(f)
Constant = |f (0)|
```

Step 4: Apply the proposed algorithm in section2.4 for guessing Initial values to be further employed while applying root methods.

Step 5: Apply the Root-Finding method (i.e., Bisection, Newton-Raphson, and Secant) as below and generate the root, which acts as an encrypted text to be sent to the receiver:

```
if method = Secant then
    root = secant_algorithm (f , initial_one_secant,
    initial_two_secant, err, iteration) //Call secant()
End if
if method = Bisection then
    root = bisection_algorithm (f , initial_one_bi,
    initial_two_bi, iterations) //Call bisection()
End if
if method = Newton-Raphson then
root = newton_algorithm (f , df , initial_newton, error,
iterations) //Call Newton-Raphson()
End if
```

Step 6: Additionally, compare all three Root-Finding approaches to examine which one works better in this situation.

Table1. Comparison of Polynomial Interpolation, AES, and RSA Algorithms

Category	**Polynomial Interpolation**	**AES (Advanced Encryption Standard)**	**RSA (Rivest–Shamir–Adleman) Algorithm**
Use Cases	Secret sharing, privacy-preserving proofs, distributed security	Data encryption, confidentiality	Asymmetric encryption, digital signature
Mathematical Robustness	Mathematically rigorous, well-founded in numerical analysis	Strong mathematical foundation, widely adopted	Strong mathematical basis, security relies on factorization
Privacy-Preserving Properties	Supports zero-knowledge proofs for privacy preservation	Primarily focuses on encryption, not designed for privacy	Supports digital signatures but not zero-knowledge proof
Efficiency and Scalability	Efficient for specific scenarios, scalable for key distribution	Efficient data encryption, scalability depends on key management	Slower than symmetric encryption, key management can be challenging
Distributed Security	Facilitates secure data sharing, distributed security measures	Primarily designed for centralized encryption, not suitable for distributed security	Not designed for distributed security
Flexibility and Versatility	Adaptable to various cryptographic use cases signatures	Specialized for data encryption, limited versatility	Specialized for asymmetric encryption and digital
Interoperability	Can be designed for interoperability across platforms	Standardized encryption, good interoperability	Standardized for digital signatures and encryption, decent interoperability

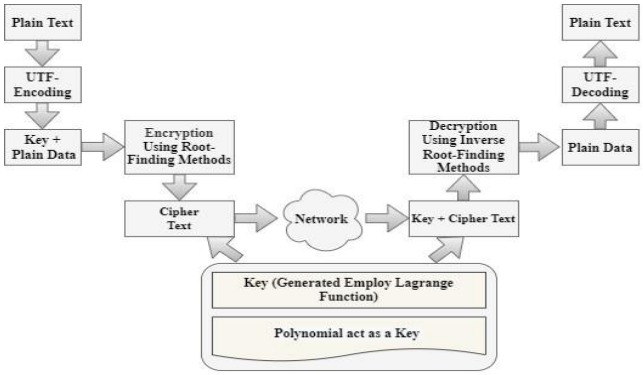

Fig.1. Proposed security architecture

3.1.2 IBCA- Decryption

When the recipient receives a root value in the form of encrypted text, decryption is entirely based upon the original polynomial and the encrypted text, where plain text is evaluated by assigning the received content in the polynomial function and is depicted as:

$$original\_digit = poly\ (root)$$

3.2 Bi-New

The proposed Bi-New scheme similarly employs a key as IBCA. The constant coefficient of the polynomial is subtracted from the selected plain text to derive a new scrambled polynomial. The encryption and decryption are entirely reliant upon this polynomial. An initial value sup poses X is computed by taking the nth root of the newly generated constant value, where n is the degree of the polynomial. The Bisection Method is applied using that initial calculated value to create an approximation root for the given polynomial. Furthermore, the left rotation is performed. After rotating, the same procedure is repeated, and the Newton-Raphson method is applied to find the root. This root will act as a final ciphered data to be shared. The calculated roots are then stored in a file and shared with the receiver. The flowchart for the scheme has been delineated below in Figure3.

3.2.1 Algorithm: Encryption of Bi-New

Input: Nth odd degree exponential g(X) + 80 bits Plain Text (PT)

Output: Root Value

Step 1: PT is taken for encryption in the form of digits.

Step2: Obtain the randomized polynomial generated through Lagrange Interpolation Method of odd degree up to 13th order [15].

$$poly = lagrange\ (x,y)$$

Step3: Subtract the plain text from the polynomial's constant term.

$$f = poly - digit$$

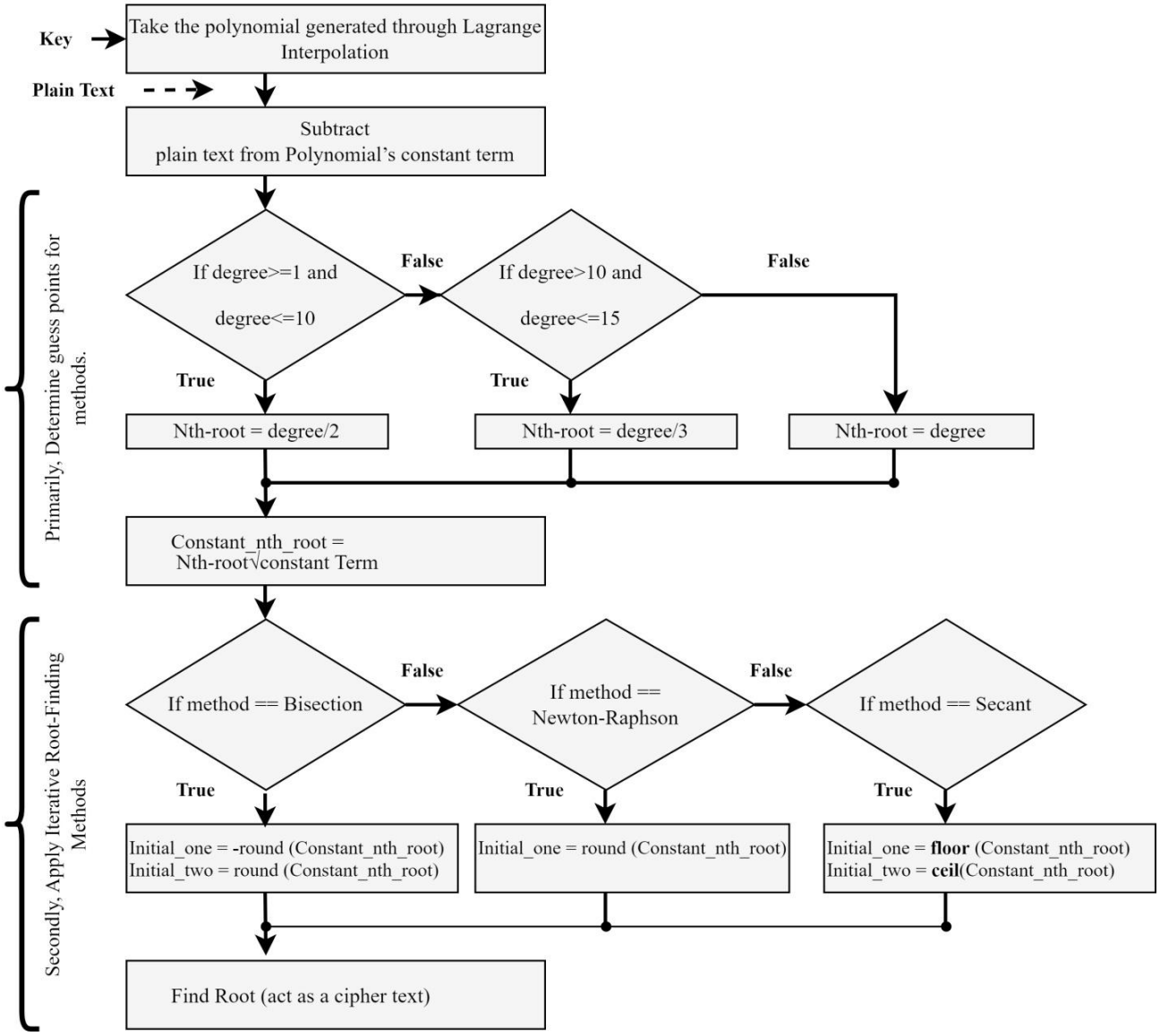

Fig. 2. Flowchart for IBCA.

Step 4: A novel scrambled polynomial has been created in the following way:

```
poly_coefficient = Polynomial(f).coef
df = np.polyder(f) // where np is numpy
constant = abs(f(0))
```

Step 5: Implement the above-mentioned proposed algorithm in section 2.4 for guessing initial values for further use while applying Root-Finding methods.

Step 6: Apply the Bisection Root-Finding method and generate the root: *root bisection 1*.

Step7: Rotate Left (ar[],d,n) that rotates arr[](the coefficients of the original polynomial) of size n by one element as:

```
poly_coefficient_left_rotate= poly_coefficient[1:]
```

Step 8: Afterwards, replace the constant and substitute the root_bisection_1 with negation.

```
poly_coefficient_left_rotate=np.append(poly_coefficie
nt_left_rotate,-root_bisection_1)
```

This forms a new polynomial.

Step 9: Constructing the new polynomial as:

```
f = np.poly1d (poly_coefficient_ left_rotate)
```

and repeat Step4 and Step5.

Step 10: Apply the Newton-Raphson Root-Finding method and generate the root, which acts as a ciphered text to be sent to the recipient.

3.2.2 Bi-New: Decryption

Input: Root Value as an Encrypted Text

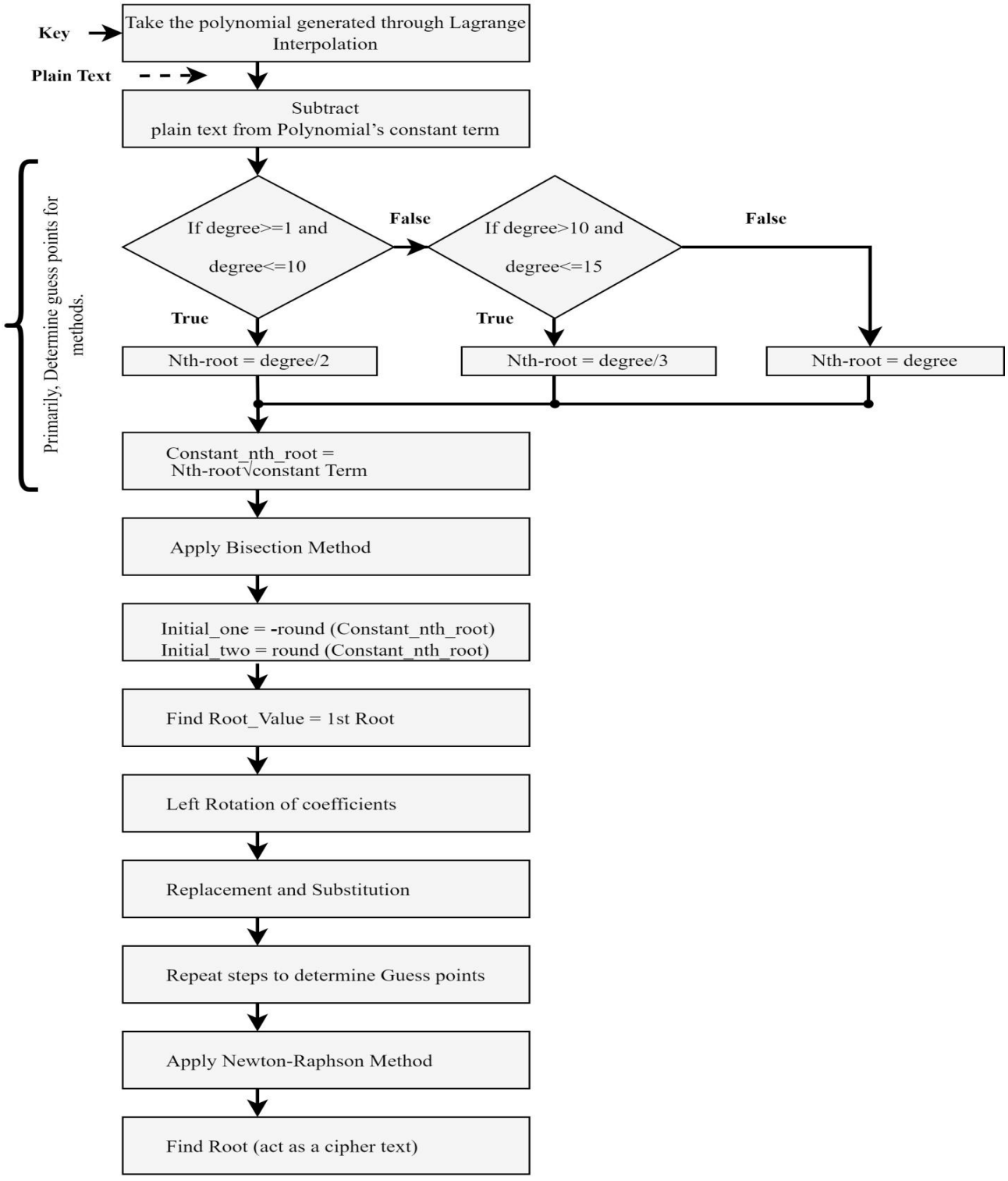

Fig. 3. Flowchart for Bi-New.

Output: 80 bits Plain Text (PT)

Step 1: Take the polynomial as

f = poly

poly_coefficient=Polynomial(f).coef //separating the coefficients in the array

Step 2: Rotate left the original polynomial and replace the last coefficient, i.e., the constant part, with zero.

poly_coefficient_left_rotate=poly_coefficient[1]

poly_coefficient_left_rotate=np.append(poly_coefficient_left_ rotate, 0)

f_left_rotate=np.poly1d(poly_coefficient_left_rotate)

Step 3: Substitute the root value received from the sender.

first_decryption_value = f_ left_rotate (root_final)

Step 4: After substituting, the receiver automatically obtains the first root value, which is known only to the sender and decrypts the text.

final_decryption_value=f(first_decryption_value)

3.2.3 Work out example for Bi-New

Step 1: Original polynomial be = $k_n x^n + k_{n-1} x^{n-1} - k_0$ Digit to be encrypted, i.e. Plain Text = k

Step 2: According to the method, subtract k from the constant of original polynomial (k_0) and the resultant new polynomial = $k_n x^n + k_{n-1} x^{n-1} - k_p$.

Step 3: Find the initial guess using the nth root algorithm

Step 4: Apply the Bisection Method further for finding the root. Hence, the Root1 is a.

Step 5: Now apply left rotate akin:

- Coefficients of Original Polynomial: [k_n , k_{n-1} , $-k_0$]
- Coefficients after left rotating: [k_{n-1} , $-k_0$, k_n]

Step 6: Now replace the new constant, i.e., k_n with the negative value of Root1.

Coefficients after replacing: [k_{n-1} , $-k_0$, -a]

Step 7: Repeat step3 for the new coefficients of Step 6.

Step 8: Apply the Newton-Raphson Method and find the final root to be shared as an encrypted value with the recipient.

Step 9: Hence, the root to be shared = b.

4. RESULTS AND DISCUSSION

In this section, we discuss simulation setting and 'security and performance analysis'.

4.1. Simulation Environment

Various metrics are required to be computed to test the efficacy of the proposed technique. Moreover, various security analysis threads are discussed to indicate that the proposed scheme is highly resistant to various attacks. The software requirements for the proposed scheme implementations are as delineated below:

- Windows 10 operating system
- Python 3.8.5, Pycharm 2020.2.3 programming language is deployed due to its reliability, flexibility, multiple libraries, and frameworks, ease of handling, and possessing topmost security.
- Microsoft Office Excel is used to plot graphs. The hardware utilized for the proposed scheme was of the following configurations:
- Speed: Intel i-7 processor with 2.50GHz
- Memory: 8 GB RAM

4.2. Performance Evaluation Parameters

An odd polynomial was considered till the 13th degree for experimental purposes. Initial guess points a and b for the bisection method and one initial point for the Newton-Raphson techniques were derived from algorithm in 2(D) section. Each encryption method had its own set of strengths and weaknesses. One should know the algorithms' performance, potency, and deficiencies to exert an appropriate cryptographic technique. As a result, these algorithms must be evaluated using various criteria. The proposed scheme was analyzed using the following metrics, as delineated below. These metrics rationally convey the scheme's security and effectiveness.

4.2.1. Time Analysis

The time it takes to encrypt or decrypt data in any cryptosystem is referred to as the computed time. The simulation results of time analysis depict that the proposed scheme was appropriate for real-time implementation since the time taken for it to encrypt or decrypt the data impacted the system's performance. Less time implies rapid response of the system. In the experiment, time is measured in seconds and documented as the average duration after conducting the test 100 times. According to the data, the algorithm uses a small number of resources and has high efficiency levels. Key generation analysis for different orders of polynomials is shown in Figure4. Figure5 depicts the encryption time for the techniques, and Figure6 represents the decryption time. The results show that there is no trade-off between key complexity and time complexity. To put it differently, as the key size increases, the corresponding rise in time is quite negligible.

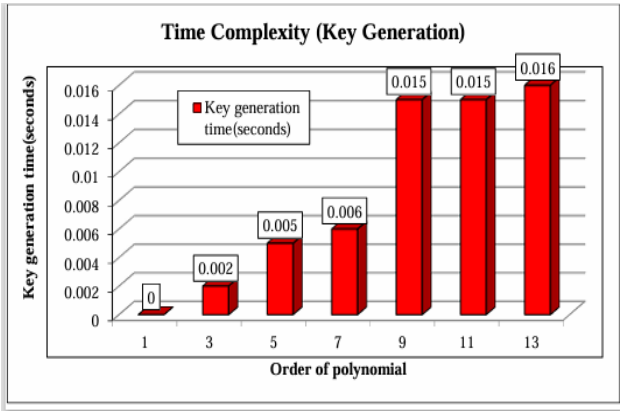

Fig. 4. Key generation analysis for different orders of polynomials.

The graph displays a time analysis in seconds for polynomials with degree range from 1 to 13.Besides, the execution time, including the Key generation time, Encryption Time, and Decryption Time of proposed schemes. In Figure 7, the execution duration of the 13th-degree polynomial, which is the highest degree polynomial,

was analyzed in relation to AES, RSA(1024), and RSA(2048). Overall research shows that the proposed algorithms performed better than AES and RSA. We have also provided the simulation snapshots of the performance investigation of Bi-New for time consumption in Figure 8.

4.2.2. Accuracy

The accuracy was checked by running both the methods with random non-linear polynomial (key) and random plain text in a loop for a minimum of 100 times. For examination, the relation followed as delineated in Equation1.

$$\text{Accuracy} = \frac{\text{Number of success}}{\text{Number of Success} + \text{Number of failure}} \quad (1)$$

The results show that both schemes are endowed with 100% accuracy till the 13th order; the performance beyond the 13th order needs to be analyzed which make the proposed schemes qualify for practical implementation.

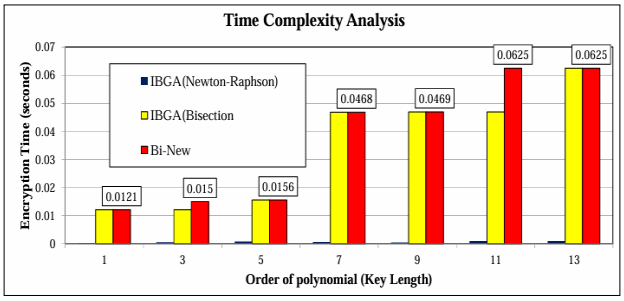

Fig. 5. Performance comparison of encryption time.

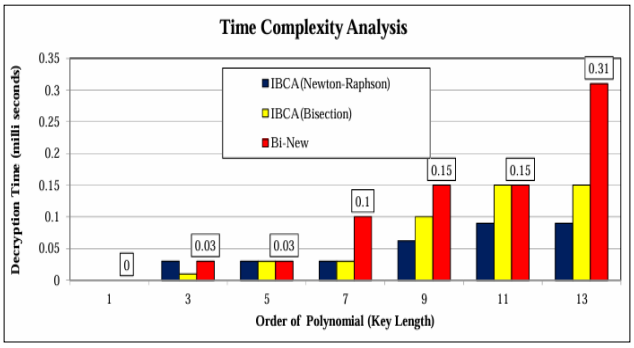

Fig. 6. Performance comparison of decryption time.

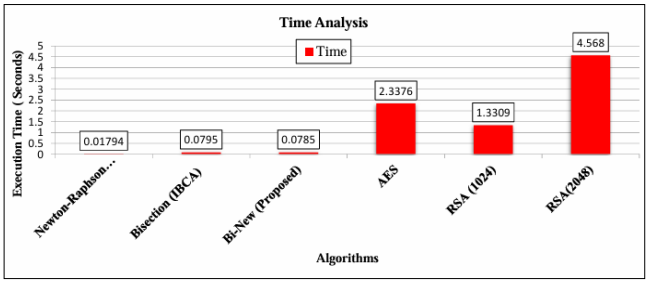

Fig. 7. Comparative analysis of execution time of proposed method against others.

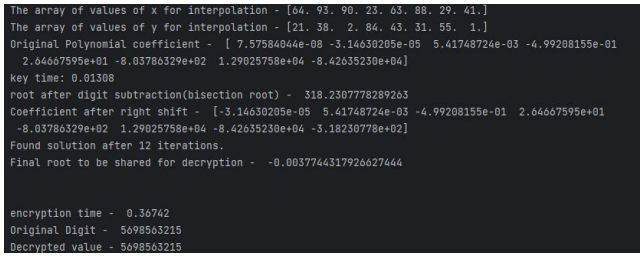

Fig. 8. Performance investigation of Bi-New for the time consumption.

4.2.3. Memory Analysis

Memory allocation is achieved by effectively managing the memory resources of computer applications and services, allowing them to utilize either virtual or physical memory-fully or in part—for executing programs and processes. A responsive and rapid system should consume less time. Different cryptographic techniques consume different-sized memory while encrypting or decrypting. The algorithm's number of operations determines the memory requirement since the programming language impacts the memory space. It is always preferred to have less memory consumption for cryptographic operations. The total memory consumption of IBCA and Bi-new are similar. Figure9 depicts the memory requirement by the scheme from 1st till the 13th order polynomial i.e., O1-O13 where O represents the order of polynomial. The results show that the proposed schemes consumed less memory space than RSA. The highest-order polynomial also took less time than RSA. Still, the cryptanalysis part needs to be analyzed in detail to check the strength of the proposed algorithm.

4.3. Security Analysis

Assessing the security system is essential to evaluate the effectiveness of the cryptosystem against various encryption threats. For a polynomial to be relevant in a cryptosystem, it needs to have certain security characteristics, which can be evaluated through the techniques outlined below.

4.3.1.KeySpace Analysis

The imperative thread of any encrypted algorithm is key/s. If the key is controlled, the cypher text can inevitably be obtained to breach security functionalities. The attacker utilizes a brute force approach to predict the keys; a confined keyspace makes the job of an attacker easier. For a polynomial system to be acceptable, the keyspace must be substantial enough to withstand brute-force attacks. The proposed method uses the polynomial functions and the random data points to specify the key. (e+1) coefficients are employed to represent a polynomial is of degree e. Certainly, 64-bit floating-point is well-chosen to describe the terms, then the keyspace is (e + 1) × 64 bits. Therefore, with considerable keyspace, it becomes difficult for the intruder to predict g(x).

4.3.2.Key Sensitivity Analysis

Keys sensitivity is yet another crucial metric for assessing an algorithm's ability to withstand all-out attacks. Generally, when two pairs of keys with slight variations are used to encode the same plain text, then two different ciphered texts with substantial variability should be obtained; the enciphered algorithm is then considered sensitive to the keys. Noticeably, the greater the algorithm's sensitivity to keys, the stronger its security level. As already stated, exponential polynomial till the 13th degree was taken as a key. Without loss of generality, the random polynomial with more nominal terms was used for tests, followed by a minor change in the polynomial for testing. Figure10 clearly shows that the proposed schemes have a better-altered value when the key's final and first digit has been changed and compared with the original key. Besides, the results compared with AES, RSA-1024 bit and 2048-bit, and the proposed schemes showed improved performance.

4.3.3. Robustness Analysis

A single-digit change in plaintext or cyphertext should produce a different result. Therefore, what should a proficient algorithm look like? Monitoring data changes could allow an unauthorized third party to guess the pattern of encryption and decryption. Attacker-performed differential cryptanalysis examines changes to some chosen plain texts and variations in the outputs of each one's encryption in order to potentially recover the keys. The Number of Digit Change Rate (NDCR) was utilized to detect differential attacks.

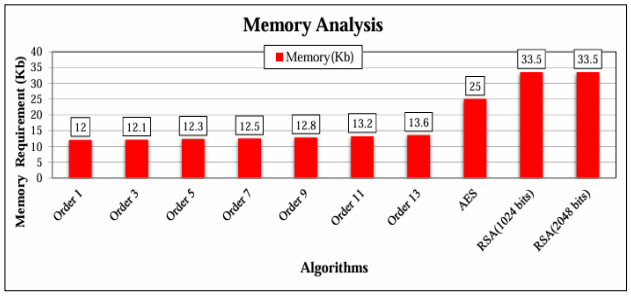

Fig. 9. Performance comparison in regard of memory consumption.

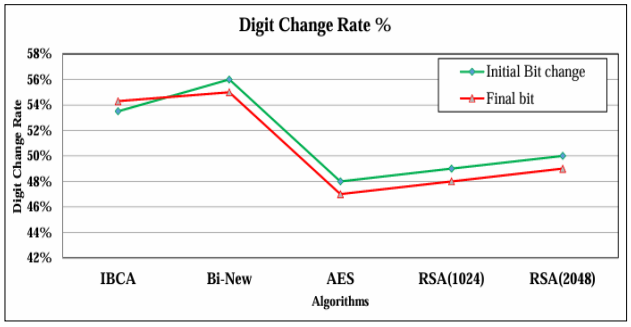

Fig. 10. Performance comparison of digit change rate %.

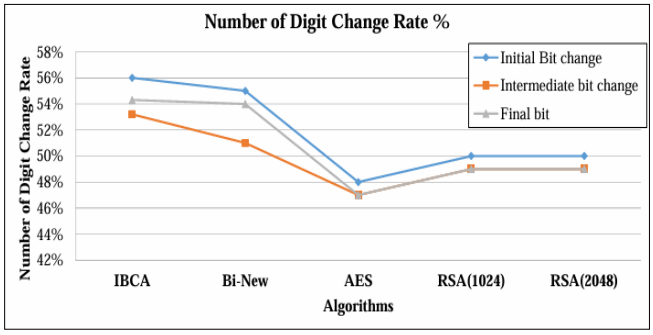

Fig. 11. Performance comparison of number of digit change rate %.

The comparison was conducted between the changed and original plain text encrypted via a similar polynomial. For the examination test, one digit in the plain text was altered. The percentage change in aspects of the digit change rate was analyzed; for an ideal case, the value of the NDCR should be>=50%. The NDCR was calculated using the parameters in Equations (2) and (3).

$$\text{NDCR (O1, O2)} = \frac{1}{n}\sum_{i=1}^{n} C(i) * 100 \quad (2)$$

$$C(i) = \begin{cases} 0, \text{if } O1 = O2 \\ 1, \text{if } O1 \neq O2 \end{cases} \quad (3)$$

In this context, n signifies the number of bits; C(i) is determined by evaluating two outputs; O1 denotes the output derived from the original plain text; while O2 signifies the output resulting from the modified plain text.

Figure 11 clearly indicates that the suggested schemes achieve a superior NDCR value for ten-digit data when altering the final, intermediate, and initial digits. Besides, the results compared with AES, RSA (1024 bits) and 2048 bits, and the proposed schemes showed improved performance In every instance, the calculated NDCR was near the optimal scenario and demonstrated robustness against differential cryptanalysis assaults. Hence, the findings have been obtained by executing a loop a minimum of 100 times. Cases are pictured with changes in coefficients mentioned in bold along with underline. Henceforth, this may infer that the proposed work can resist brute-force attacks. The results are tabulated in Table 2.

4.3.4.Statistical Analysis

Correlation is a statistical method for determining the association between two variables. In other words, the correlation coefficient (CC) formula aids in the calculation of the correlation coefficient, which evaluates the interdependence of one variable on another. CC is used to calculate the correlation. CC is calculated using a scale from- 1 to 0 to 1. The relationship between the variables is solely defined by either -1 or 1. The total absence of correlation is signified by 0. Pearson Correlation Coefficient (PCC) is defined in equation 4 below. In this paper, CC was utilized to calculate the key sensitivity. The key was

produced using Lagrange interpolation, which involved two data points to formulate a polynomial. Hence, the relation between the M and N data points was computed through correlation.

Equation (4) is defined as:

$$C_{M,N} = \frac{X(M,N)}{\sqrt{Y(M)}\sqrt{Z(N)}} \quad (4)$$

where:

$$X(M, N) = \sum_{i=1}^{j}(M_i - \bar{M})(N_i - \bar{N}) \quad (5)$$

$$Y(M) = \sum_{i=1}^{j}(M_i - \overline{M})^2 \quad (6)$$

$$Z(N) = \sum_{i=1}^{j}(N_i - \overline{N})^2 \quad (7)$$

Here, M and N are two data points with randomly generated values from a generator. j is the number of data points, which depends on the degree of a polynomial with j + 1data points required for a degree d polynomial). X(M, N)represents the covariance between points, while Y(M) and Z(N) represents standard deviations of M and N.

Let's consider the following data points:

M = [10, 10, 12, 8, 19, 1]

N = [14, 11, 5, 0, 18, 16]

Calculate the mean of M and N:

$\bar{M}$ = 10

$\bar{N}$ = 11

Now plug the above values in Equation 5:

X (M, N) = (0, 0, 2, -2, 9, -9) (3, 0, -6, -11, 7, 5) = -5

In Equation 6:

Y (M) = (0, 0, 4, 4, 81, 81) = 170

In Equation 7:

Z (N) = (9, 0, 36, 121, 49, 25) = 240

After obtaining these results and substituting them into Equation (4), calculate the correlation coefficient as:

$C_{M,N}$ = -0.02

The above data points were randomly generated through an algorithm. A relation was solved using the provided values, demonstrating that the data points were random as their dependency was null. Further calculations yielded correlation coefficients ranging between 0.0 to 0.4 and-0.0 to 0.4, indicating weak dependency. Therefore, no relation among variables was found, confirming that the data was random and the proposed algorithm was resilient to correlation attacks. Additionally, the proposed algorithm did not utilize any binary functions during encryption, further avoiding correlation attacks.

Table 2. Performance investigation of proposed work for different degrees of polynomial i.e., 3, 5 and 13

CASE I: Polynomial with integer coefficients of degree 3

Key Coefficients	Cipher Text	Key Altered	Cipher text
[23,1,0,**25**]	-0.7723457864376819	[23,1,0,**30**]	-0.8819413926396629

CASE II: Polynomial with floating coefficients of degree 5

Key Coefficients	Cipher Text	Key Altered	Cipher text
[-3.14414338e-05, 7.99715551e-03, -7.01100231e-01, 2.33444757e+01, -1.92818859e+02, **3**.87741669e+02]	8.048178151891836	[-3.14414338e-05, 7.99715551e-03, -7.01100231e-01, 2.33444757e+01, -1.92818859e+02, **7**.87741669e+02]	95.90980962165276

CASE III: Polynomial with floating coefficients of degree 13

Key Coefficients	Cipher Text	Key Altered	Cipher text
[**2**.79088071e-15, -2.10156003e-12, 7.14004920e-10, -1.44592359e-07, 1.94186842e-05, -1.82005008e-03, 1.21932990e-01, -5.87608175e+00, 2.01915834e+02, -4.81899340e+03, 7.57385393e+04, -7.03081888e+05, 2.91243607e+06, 5.40000000e+01]	- 139.9592909446094	[**9**.79088071e-15, -2.10156003e-12, 7.14004920e-10, -1.44592359e-07, 1.94186842e-05, -1.82005008e-03, 1.21932990e-01, -5.87608175e+00, 2.01915834e+02, -4.81899340e+03, 7.57385393e+04, -7.03081888e+05, 2.91243607e+06, 5.40000000e+01]	71.19100202522984

5. CONCLUSION

In this paper, the authors comprehensively detail non-linear polynomial-based encryption schemes, leveraging the roots of non-linear algebraic equations, non-linear functions, random generators, and array rotation, among other techniques. They employ Lagrange interpolated polynomials, extending up to the 13th degree, for further analysis. Three root-finding methods (Bisection, Newton-Raphson, and Secant) are used in conjunction with these interpolating polynomials to uncover the roots of non-linear algebraic equations. The paper introduces two novel encryption schemes based on these methods. In the initial approach, substitution is employed, using Bisection and Newton-Raphson methods to interpret the cipher text. In the second scheme, a combination of methods is employed, encompassing substitution, rotation, replacement, and further substitution. This multi-step approach enhances both confusion and diffusion, bolstering security. The proposed encryption scheme is tested against conventional algorithms such as AES, RSA (1024), and RSA (2048), demonstrating superior performance in various metrics. Furthermore, security assessments reveal its resilience against brute force attacks, differential attacks, and statistical attacks, affirming its practical suitability for implementation. Since polynomials are quantum safe, the proposed work may further be implemented by amalgamating it with homomorphic encryption, Internet of Things making it more reliable and secure.

ACKNOWLEDGEMENT

A preliminary version of this work has been deposited as a preprint on Research Square (DOI: 10.21203/rs.3.rs-2050151/v1).

REFERENCES

[1] Dave, G., Choudhary, G., Sihag, V., You, I. and Choo, K.K.R. 2022. Cyber security challenges in aviation communication, navigation, and surveillance. *Computers & Security*, *112*, 102516.

[2] Mondal, S., Shafi, M., Gupta, S. and Gupta, S.K. 2022. Blockchain based secure architecture for electronic healthcare record management. GMSARN Int. J, 16(4), 413-426.

[3] Charoenchit, S. and Thongchaisuratkrul, C. 2021. Safety inspection of electrical systems case study in paper and animal food factories. GMSARN International Journal, 15, 331-339.

[4] Kaur, J. and Ramkumar, K. 2022. The recent trends in cyber security:A review. Journal of King Saud University-Computer andInformation Sciences, 34(8),5766–5781.

[5] Jie, L. K. and Kamarulhaili, H. 2011. Polynomial interpolation in theelliptic curve cryptosystem. Journal of Mathematics andStatistics, 7(4), 326–331.

[6] Bezzateev, S. Davydov, V. and Ometov, A. 2020. On secret sharingwith newton's polynomial for multi-factor authentication. Cryptography, 4(4), 34.

[7] Huang, Y.T. Chiang, D.L. Chen, T.S. Wang, S.D. F. Lai, .P.and Lin, Y.D. 2022. Lagrange interpolation-driven access control mechanism: Towards secure and privacy-preserving fusion of personal health records. Knowledge-based systems, 23(6), 107679.

[8] Chen, Z. Huang, L. Shi, X. Huang, Q. Wang, H. and Liu, X. 2020. Privacy-preserving polynomial interpolation and its applications on predictive analysis. Information Sciences 54(1), 259–270.

[9] Arora, J. Ramkumar, K. Sathiyaraj, R. and Ghantasala, G. P. 2023. Securing web documents by using piggybacked frameworkbased on newton's forward interpolation method.Journal of Information Security and Applications, 75, 103498.

[10] Dini, P. Begni, A. Ciavarella, S. De Paoli, E. Fiorelli, G. Silvestro, C. and Saponara, S. 2022. Design and testing novel one-classclassifier based on polynomial interpolation with applicationto networking security. IEEE Access, 10, 67910–67924.

[11] Cafaro, M. and Pell`e, P. 2018. Space-efficient verifiable secret sharingusing polynomial interpolation. IEEE Transactions on CloudComputing, 6. (2), 453–463.

[12] Patil, P. Narayankar, P. Narayan, D. and Meena, S. M. 2016. A comprehensive evaluation of cryptographic algorithms: Des, 3des,aes, rsa and blowfish.Procedia Computer Science, 78,617–624.

[13] Ehiwario, J. and Aghamie, S. 2014. Comparative study of bisection,newton-raphson and secant methods of root-finding problems. IOSR Journal of Engineering, 4(4), 01–07.

[14] Badr, E. Almotairi, S. and Ghamry, A. E. 2021. A comparative studyamong new hybrid root finding algorithms and traditionalmethods. Mathematics, 9(11), 1306.

[15] Kaur, J. and Kumar, K. R. 2022. Key management using lagrangeinterpolation for wireless communications, in: 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE), IEEE, 2084–2087.