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

jagpreet kaur
Chitkara University
Dr. Ramkumar K.R. (k.ramkumar@chitkara.edu.in)
Chitkara University

Research Article

Keywords: Polynomial interpolation, Lagrange interpolation, Root-finding equations, security, cyber attacks

Posted Date: September 14th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2050151/v1

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

Abstract: Confidentiality is one of the fundamental aspects to consider while securing wireless communication. Endeavouring current developments will catch up with the conventional cryptographic methods soon than expected; therefore, a new path is indispensable. In this context, this article presents an innovative idea of using polynomials to achieve a higher level of data privacy. Polynomials' strength lies in their irreducible property, which makes them plausibly appropriate for cryptography. Thenceforth, two novel schemes are proposed based on root-finding polynomial interpolations such as Bisection, Newton-Raphson, and Secant. Relying on these techniques, while the first scheme performs substitution, the BI-New scheme performs substitution, rotation, replacement, and substitution. Hence, both schemes provide confusion and diffusion, a fundamental security aspect. Besides, these algorithms utilise initial points to extract the exact root of the polynomial $p_1(y) = 0$. On the basis thereof, an algorithm is developed to retrieve the initial data points. Consequently, the decryption is accomplished in reverse order by evaluating a function. The proposed scheme has unique characteristics, including non-linear interpolating polynomials, roots of non-linear algebraic equations, and non-linear functions. Due to the amalgamation of various non-linear methods and randomised variables, the methods are one-way functions that can endure several assaults. Lastly, the algorithm's security is illustrated by multiple state-of-the-art quantitative metrics such as time complexity, accuracy, memory usage, key space analysis, key sensitivity analysis, robustness analysis, and statistical analysis.

KEYWORDS
Polynomial interpolation, Lagrange interpolation, Root-finding equations, security, cyber attacks

1. INTRODUCTION

The communication and retention of data are rising exponentially in this cutting-edge technological epoch. For decades, the privacy and security of these resources have been an obligatory focus of research. Data privacy is one of this era's most formidable questions [1]. Data is, however, stored as bits and is transmitted and accessible by several parties via the internet. Nevertheless, over the past decade, the way of fetching and forging data by intruders has increased, so the term that is immensely popular among practitioners is "Cyber Security [2]." Besides, the integration of wireless communication and computing led to several vulnerabilities in the system [3]. Wireless sensor networks are typically deployed in unscreened environments and as vital components; they employ in numerous sectors such as the economy, health department, banking, intelligence sector, etc. [4]. Therefore, flaws in cyber security profoundly impact these sectors. Hence, their security is paramount. Cryptography is, by far, the indispensable way to attain authentication, confidentiality, and data integrity. Encryption and decryption are two essential phases of cryptography to procure secrecy. Several standards (DES, AES, RSA, RC5, and others) are already in operation to achieve these phases. These standards count upon symmetric and asymmetric encryption, which are fundamentally different. The asymmetric keys commit high confidentiality; yet slow. Even though symmetric algorithms are fast, they use XOR functions to design keys, and the lengths are fixed. These defined lengths effectuate invaders to break the keys employing brute force attacks, compromising DES [5]. Fundamentally, encryption holds on to the principles of confusion and diffusion. In addition, there are some guidelines that every cryptosystem needs to follow [6]. These prescribed guidelines explain three significant concerns: implementation, management of keys, and security analysis.

A comprehensive technique outlined for cryptography was based on Polynomial Interpolation to factorise problems and address various attacks [7]. Most innovators used Lagrange Interpolation for key management [8, 9]. Due to its irreversible and confusing property, interpolation is quite popular in cryptography [10, 11]. The Interpolation method integrated with the ECC algorithm led to the modification of that algorithm [12]; its incorporation led to a tradeoff between time and space. Newton interpolation scheme has been deployed for multi-Factor authentication

[13]. The Bisection method was used to generate a random sequence for image encryption [14]; however, the exploit of the XOR operation made it insecure [15]. Similarly, much research has been done in image encryption utilising polynomials [16-18].The encryption cryptosystems are widely categorised as modern and classic systems. In modern systems, confusion and diffusion are done in parallel, whilst in the latter; confusion and diffusion are done separately. The algorithms can be broken if processed alone with confusion [19]. A study proposed in [20] was solely based on confusion and hence, easily decrypted in [19]. Additionally, researchers utilised Chebyshev Polynomials for designing Public-key Cryptosystem [21]. Authors in [22] proposed an encryption scheme based on Chebyshev maps in which the AES algorithm was amalgamated with chaotic maps to enhance security. However, the downside of maps has been outlined in [23]. Chaos-based Chebyshev polynomial was utilised for image encryption, too [24]. Broadcast encryption is a scheme that states that the enciphered content is delivered through broadcast channels so that only grouped members decrypt the content [25]. Polynomial interpolation methods have been embedded with a broadcast scheme to make them more secure and flexible; however, they dealt with the trade-off between security and computational complexity [26].

According to the ideology of Root-finding methods, this article states two novel techniques for encrypting messages. The encryption procedure implemented uses three prominent root-finding methods (Bisection, Newton-Raphson, and Secant) and their comparative analysis. The proposed scheme comprises three steps: In the first part, an exponential polynomial is employed. The polynomial is constructed utilising Lagrange Interpolation. Furthermore, interpolating data points are generated using a random sequence generator and this polynomial act as a secret key to encrypt data. In the second division, the polynomial and the plain text are used for encryption. To further apply the root of algebraic equations, initial data points are required, wherein an algorithm is proposed to guess these points. In the third step, the root finding operations are applied to find the root, which acts as a cipher text. For practical execution, diminishing computational complexity without compromising a cryptosystem's security is pivotal. 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 principal rationale behind the less encryption and decryption time is the utilisation of mathematical methods, which has far less computational complexity. Furthermore, these schemes can resist existing cyber-attacks. Additionally, a comparison with the standard algorithm was performed.

The rest of the manuscript is organised in the following sequence. Section 2 presents an overview of recent security flaws in standardised algorithms. Section 3 elucidates the preliminary description of root-finding methods and describes the proposed algorithm guessing the initial values to get the root. Section 4 presents the security architecture, explaining schemes through a flow diagram with the steps applied in the crypto process, followed by a detailed illustration of the algorithm using a worked-out example to give an explicit rationale of the same. Section 5 outlines the simulation requirements. Section 6.1 validates the proposed work results and discusses the results. Section 6.2 substantiates the resilience and invulnerability to different attacks, quantifying the scrutiny of security.
Section 7 presents the conclusion.

2. Literature Review

In the digital environment, there is an ever-rising demand for speed, which has led to application development processes to enhance performance by incorporating various methods during implementation. While the improvements increase efficacy, security flaws also get introduced, which are sometimes overlooked. Multifarious attacks are such exploits that benefit from such faults, which should have a detrimental impact in critical scenarios, resulting in security breaches. Hence, the classification of encryption techniques and the issues related to security are succinctly discussed below.

2.1 Data Encryption Standard

DES is one of the most widely accepted block ciphers utilising shared keys to encrypt and decrypt the data. DES was published in 1977 and standardised in 1979. In this, the key is 64-bit, which has 8 bytes, and there is one parity bit for each byte. Henceforth, the value of the key left was 56-bit regardless of the output, which was again 64-bit cypher text [27]. However, DES is no longer invulnerable to attacks like brute force attacks, and key attacks are already compromised [28].

2.2 Advanced Encryption Standard

When Data Encryption Standard (DES) started to become vulnerable to brute-force attacks, it was declared that there was a need for a successor algorithm. Henceforth, the NIST began the development of the most popular and widely accepted symmetric encryption algorithm AES in 1997. This is a symmetric key symmetric block cypher. Hence 128-bit data have 128/192/256-bit keys, with 10/12/14 rounds respectively [29].

However, AES is vulnerable to various attacks due to its widespread usage in real-world applications. The following are some of the most recent attacks delineated:

A new homological fault base attack was outlined for AES that used a Clock glitch injection [30]. The attack process demonstrated that minute bugs finally disclosed the key. The suggested cryptanalysis approach can crack the key in AES by combining fault attacks with coarse-grained clock flaws.

Another attack strategy is the differential fault attack using a single fault strategy applied to AES [31]. The strategy showed that employing a two-stage technique and a single random byte defect at the eighth round's input could determine the AES key. A more improvised differential attack against AES is presented in the paper [32]. This approach integrates the encryption process's fault models with the key scheduling process's fault models. The results reveal that the complete AES-128 key may be recovered by utilising erroneous two-line cypher texts.

A related-key attack outlined in paper [33] showed the improvised version in [34] that worked on the previous cryptanalysis by modifying the sequence of round transforms and key generation modules. It minimised the rounds from 10th to 7th, leaving AES insecure even for the early rounds. The attack's complexity decreased from 2^{192} to 2^{104}. A more improved version obtained the attack in the 5th round, reducing the complexity from 2^{32} to 2^{22}[35].

A study demonstrated a Cache template attack on T-table against AES [36]. The cache's behaviour disclosed the input in its entirety or part. The authors proposed a simple yet efficacious search method that speeded up the profiling phase at least 64 times. The notable aspects of the strategy are the dependency relationships between private key and cache memory behaviour. The experimental findings demonstrated that the attack took less time to execute than the actual attack.

Another group outlined the collision attack against masked AES that took advantage of both colliding and non-colliding traces [37]. The scheme could potentially increase effectiveness while reducing the total number of traces required. The technique is efficient in recovering keys having an 80% success rate.

A timing attack is another exploit that allows an adversary to identify vulnerabilities in a remote or local system and retrieve sensitive information by monitoring the system's turnaround time to multiple parameters. A cache-based timing attack against AES-GSM implemented the attack model based on the broadly used OpenSSL library [38]. In most cases, their attack methodology enabled them to retrieve and converge the secret key.

2.3 RSA

RSA is asymmetric cryptography that enables public-key encryption for secure data transmission. This algorithm generates two sets of keys: a public key and a private key. The security of this algorithm depends upon the practical difficulty of factorisation of the product of two enormously large prime numbers. Nevertheless, computers can factor large prime numbers and even above 200 digits, which leads to compromise RSA [39]. However, if large prime numbers are generated utilising two same-sized prime factors, no factorisation algorithm can solve this problem in a reasonable time [40]. Hence, the potency of RSA is highly reliant on the integer factorisation problem. It acts as a trapdoor for this algorithm since it is easy to get the product of two large prime numbers but stupendously hard to go the other way, and no attack has been found which can solve this problem in real-time. Besides, the security of RSA counts upon a range of factors like P, Q, e, and d. A variety of attacks persists if not taken appropriately and is broadly described below.

The foremost strategy is the mathematical attack performed to gain P, Q, and Ø values and achieved through the below-mentioned attacks:

- The first attack in which the user quickens the encryption process may exploit the small exponent e required for encryption. Let's suppose; the plain text is small, and the public exponent taken by the user is also low, then the RSA values may be compromised [41]. This attack is also known as a Low exponent attack.
- The second attack wherein the user in a surge of diminishing the workload may employ the value of d one-fourth times smaller than N (modulus). Therefore, the decryption exponent d can be computed effectively through a public key [42]. This attack is known to be a Wiener attack successful in certain circumstances.
- The third attack is a common modulus attack in which, for instance, two messages are encrypted with the same modulus N having different encryption exponent e. To exemplify, $C1=M^{e1}$ mod N, $C2=M^{e2}$ mod N where C1 and C2 are cypher-texts for message M. However, this attack works in a condition such that gcd (e1,e2)=1 [43]. Henceforth, it is easy to recover plain text.

Another strategy is the side-channel attack which aims to obtain information based upon hardware faults instead of the flaw in the algorithm itself. This attack includes a timing attack [44] and a fault analysis attack on RSA [44, 45].

The next strategy is the Brute Force attack. The RSA's security entirely depends upon Integer Factorisation Problem. If the intruder can launch an attack on the problem, in other words, if an opponent may potentially factorise the modulus N, he may efficiently compute the private key [46]. Nevertheless, the researchers design the way to solve this problem by utilising the following algorithms for factoring: Pollard's algorithm, Quadratic Sieve method, and Number field Sieve method.

Furthermore, the strategy is lattice-based reduction attacks. Lattice reduction aims to detect a basis of vectors with short and well almost orthogonal vectors. In the cryptology of key systems, lattice-based reductions are often utilised. Numerous attacks under this category are akin to the coppersmith attack [47], wiener's attack, and Boneh Durfee Attack [48]. Although these attacks are classical, they possess significant advancements. These attacks are currently used in much research. According to Boneh and Durfee's heuristic [49], the RSA encryption algorithm is unsafe if exponent d is too small. They demonstrated that, given the public key, an adversary could retrieve the private key when d< N0.292.

Another recent attack that exploits the security is Diophantine Attacks [50]. This attack describes that the n occurrences of RSA modulus (N_j=pq, e_j) where j= 1, 2, 3…, t can be factored concurrently in polynomial time utilising Diophantine equations ($e_j m^2 - n^2 \phi$ (N_j) = o_j parallel with lattice reduction schemes. Additionally, another flaw of RSA public exponents represented by the duo of N and e demonstrates that if e meets the criteria of the Diophantine equations for acceptable values of m,n, and o under certain stated circumstances, it is possible to factorise N.

3. Polynomial Interpolations

This section reviews the fundamental mathematical concepts and principles used in the proposed cryptographic algorithm to benefit the protection strategy.

3.1 Analysis of Root-Finding Methods

A mathematical equation is represented as $f(x) = b_n x^n + b_{n-1} x^{n-1} + \ldots + b_n + b_0$, where b is constant, $b \neq 0$ and n is a positive integer, and called a polynomial equation of degree n. However, if g(x) consists of several distinct functions such as exponential, trigonometric, logarithmic etc then, $f(x) = 0$ is called a transcendental equation. The value of γ of x which satisfies $g(x) = 0$ is called root or zero of $g(x) = 0$ where g(x) is a single variable function. Hence, this procedure of finding the root is known as root finding. Generally, the point at which the graph of $y=g(x)$ crosses or touches the x-axis indicates the real roots or zeros of the function [51].

On the one hand, where the root-finding problem is amongst the most pertinent computing problems in physics, chemistry, and mathematics [52], conversely, in recent years, this problem has become one of the prominent applications in the field of security. Therefore, to solve and find the appropriate roots of nonlinear, the quadratic equations iterative method plays a significant role [53]. The procedure is iterative since it involves repeating a process of numerous arithmetic operations to reach the desired goal. Each repeating cycle is known as iteration, and the result of the first iteration becomes the starting point of the next iteration. There are majorly two types of iterative methods:

1. Bracketing iteration methods
2. Open-end iteration methods

3.1.1 Bracketing method

In this method, two guesses: X_u and X_v are initially required for the zeros that bracketed the roots, as shown in Figure 1 and Figure 2. If a continuous function, f(X) is bounded to X_u, X_v values, then $g(X_u).g(X_v) < 0$. Bracket width reduces as the iterations step forward until the inexact solution to the preferred accuracy is not acquired. The general root-finding bracketing methods include the Bisection method, Regula Falsi method (or False Position), and Improved or modified Regula Falsi method.

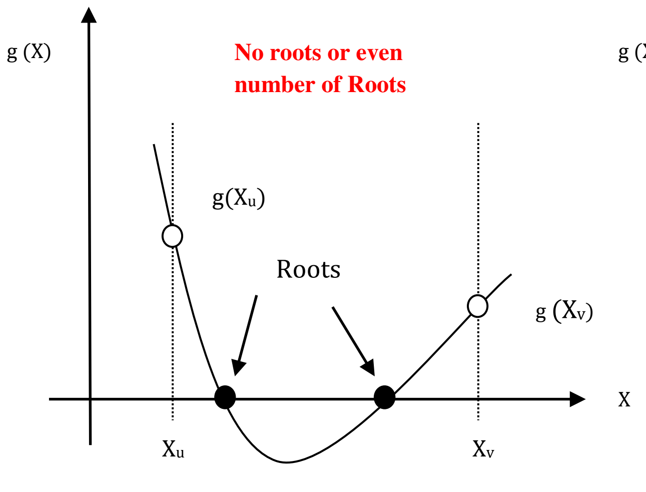

Figure 1: Even roots Graph [51]

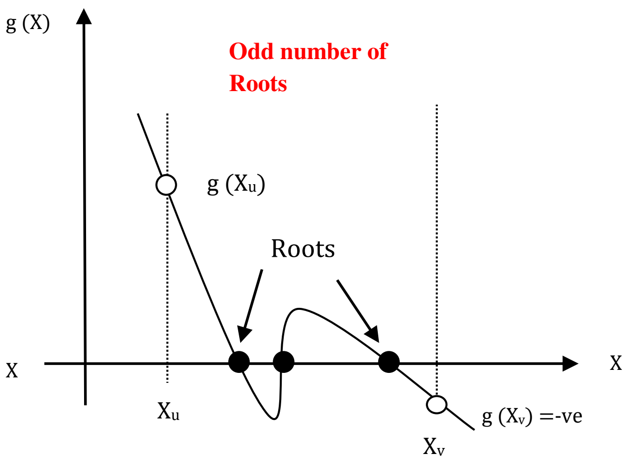

Figure 2: Odd roots Graph[51]

3.1.2 Open-end method

Unlike bracketing methods, open-end methods are based upon the fact that they require either one or two starting values that may not necessarily bracket the roots. These methods converge much faster than the bracketing methods; however, initial guesses may sometimes diverge. The general root-finding open-end methods include Simple Fixed-point iteration, Newton Raphson, and Secant.

3.2 Bisection Method

The bisection method is one of the primeval methods used to find the function's zeros. Generally, this method emerges from the following theorem: Let's assume a continuous function g(x) is positive at $x=x_u$ and is negative at $x=x_v$, then there exists at least one root between x_u and x_v. Henceforth, the initial approximation is computed by g $[(x_u + x_v)/2]$, the function value halfway between x_u and x_v [54]. If the computed value is zero, we have the root. Otherwise, the roots lie between the point and u or v, and the point accordingly as g(x) is negative or positive. This process is continued until the desired accuracy of the root is known. For this reason, this method is also known as the Binary Search method [55].

3.2.1 Working Principle of Bisection Method:

Step1: Select lower X_u and upper X_v initial guesses for the root such that:

$$g(X_u).\ g(X_v)<0$$

Step2: Root Estimation is achieved by finding the mid-point of lower and upper initial values as shown in Figure3:

$$X_r= (X_u + X_v)/2$$

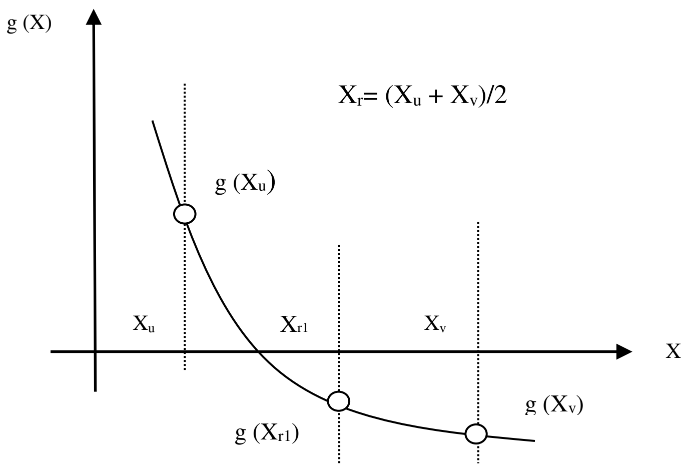

Figure 3: Root estimation Graph of Bisection[54]

Step3: The interval is subdivided according to the steps mentioned above:

Step 3.1: If (g (X_u).g (X_r) <0), then the root lies in the lower segment; $X_v= X_r$ as depicted in Figure 4 and return to step2.

Step 3.2: If (g (X_u).g (X_r) >0), then the root lies in the upper segment; $X_u= X_r$ and return to step2.

Step 3.3: If (g (X_u).g (X_r) =0), then the root is X_r and then stop.

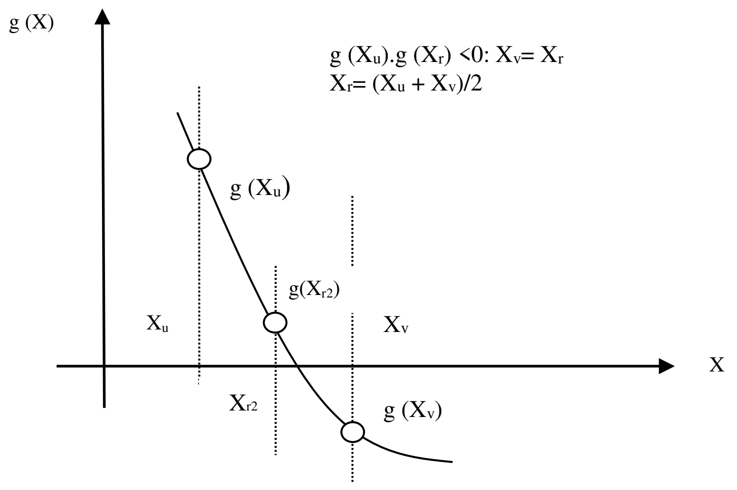

Figure 4: Root Finding Graph of Bisection[54]

This method is simple and prime in technique, but it does not have plenty of advantages. The significant advantage of the method is that it always converges to a root. However, it may sometimes fail under unusual circumstances akin to accretion of errors that would cause f(x) at some point calculated. That would further result in subdividing the wrong interval [56]. The main shortcoming of the bisection method is that it converges slowly in contrast to other methods; sometimes approximate values may not bracket the root because it is applied only when the g(x) is positive at one value and negative at another.

3.3 Newton-Raphson method

This method is extensively used for solving nonlinear equations [57-58]. It utilises the concept of slope. A tangent to g(x) at the current point Xi is elongated until it meets the x-axis and uses the improvised approximation of the root as the following reference, i.e., Xi+1 as shown in Figure 5. The iterations are continuous until the root is not found, or it reaches a specific tolerance value.

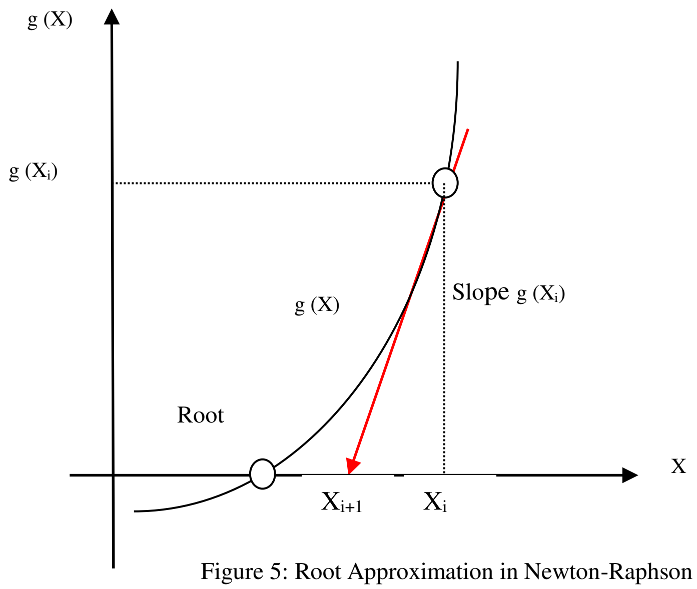

Figure 5: Root Approximation in Newton-Raphson [58]

The Newton-Raphson method is based upon Taylor series expansion as given in equation (1)

$$g(X_{i+1}) = g(X_i) + g'(X_i)\,\Delta X + g''(X_i)\,\frac{\Delta X^2}{2!} + \ldots\ldots\ldots \qquad (1)$$

Here, the value of (x_{i+1}) is the root, when $g(X_{i+1}) = 0$. Rearrange the equation (2),

$$X_{i+1} = X_i - \frac{g(X_i)}{g'(X_i)} \tag{2}$$

3.3.1 Working Principle of Newton-Raphson Method

Step1: Initial Entities: g(X): the function, X_0 = the initial guess, $\mathcal{E}$ = the tolerance factor , N= maximum iterations after which algorithm stops.

Step2: Evaluate g(X) and g′(X)

Step3: Evaluate $X_{i+1} = X_i - \frac{g(X_i)}{g'(X_i)}$ for i= 0, 1, 2…till given iterations or convergent.

Step4: If $|g(X_i)| < \mathcal{E}$ or $\left|\frac{X_{i+1} - X_i}{X_i}\right| < \mathcal{E}$ or i<N, then stop.

This method always converges fast and has the uniqueness of second-order convergence while the roots are simpler; however, this method converges linearly for multiple roots. Besides, it is sometimes damped if a wrong initial guess is taken and may not converge [59]. Moreover, this method rapidly finds that the roots of the graph are almost vertical when crossing the x-axis; nevertheless, evidently, it fails if the graph of g(X) is practically horizontal while crossing the x-axis or g′(X) approximately equals to zero in the neighbourhood of the root. Although the method rapidly converges, the performance rate declines with the following statement: g(X) and g′(X) must be calculated for the values of x till a tolerance is reached, which though factually systematic that each is evaluated from the previous value. In other words, it needs to evaluate two functions, i.e., g(X) and g′(X) per iteration, which makes it costly [60].

3.4 Secant Method

As aforementioned, with every function, there is a need to find the function's derivatives per iteration and since some functions are time-consuming, which turns out to be the foremost setback of the Newton-Raphson method. The solution design for this problem is by replacing the derivative $g'(X_i)$ [52,61], as shown in Figure 6, is obtained by equation (3):

$$g'(X_i) = \frac{g(X_i) - g(X_{i-1})}{X_i - X_{i-1}} \tag{3}$$

Hence, the formula for X_{i+1} is:

$$X_{i+1} = X_i - \frac{g(X_i)(X_{i-1} - X_i)}{g(X_{i-1}) - g(X_i)} \tag{4}$$

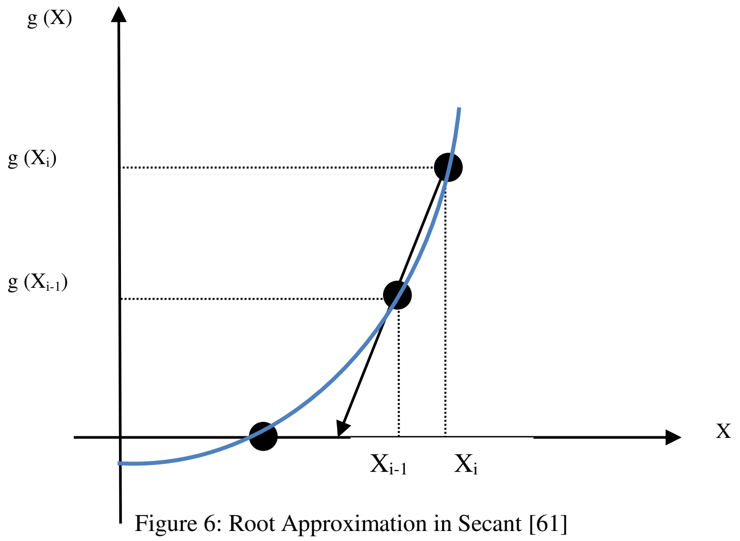

Figure 6: Root Approximation in Secant [61]

3.4.1 Working Principle of Secant Method

Step1: Initial Entities: g(X): the function, X_0 and X_1 = two initial guesses, $\mathcal{E}$ = the tolerance factor, N= maximum iterations after which the algorithm stops.

Step2: Evaluate the subsequent guess $X_{i+1} = X_i - \frac{g(X_i)(X_{i-1}-X_i)}{g(X_{i-1})-g(X_i)}$

Step3: If $|X_{i+1} - X_i| < \mathcal{E}$ or $i > N$, then stop.

The Secant method has a super linear convergence rate. Thenceforth, Secant converges relatively faster than Bisection, yet slower than Newton-Raphson. However, while evaluating the computational efficacy of the algorithm, it is not only the convergent rate that matters. Consideration should also be given to the number of flips required per iteration. As a result, Secant is faster than Newton-Raphson because the number of flops per iteration is higher in Newton-Raphson, which may take a long time to reach the accuracy level [55]. On the contrary, the Secant method is not robust, particularly when the initial assumptions are outlying the root.

3.5 Automation of Initial Values in Root-Finding 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, an algorithm is proposed and is defined below.

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

OUTPUT: Guessing Points on Graph for same Roots.

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

Step 2: 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
```

Step 3: Calculate the root value of constant term by following way to find the initial guess value as:

```
constant_nth_root = nth_root√constant
```

Step4: 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_secant = math.floor(constant_nth_root)
        initial_two_secant = math.ceil(constant_nth_root)
```

3.6 Worked out Example for proposed algorithm

Step1:Let's suppose function g(x): $\mathbf{k_n x^n + k_{n-1} x^{n-1} + k_1 x + k_0}$

Step2: Highest degree is n. Therefore, according to the above algorithm,

Constant_nth_root= $\mathbf{n/2\sqrt{\ } k_0}$

Constant_nth_root= **a**

Step3: Hence, the initial guess value depends upon the algorithm

```
If method= Bisection
    initial_one_bi = -a
    initial_two_bi = b
If method= Newton-Raphson
    initial_newton = a
If method= Secant
    initial_one_secant = a
    initial_two_secant = b
```

Step4: Therefore, based upon initial guess further solve the algorithms.

Step5: End.

4 Security architecture

Figure 7 delineate the operational architecture based upon langrange interpolation and root-finding algorithm. The proposed algorithm is segmented into two sections: first segmentGenerate Polynomial utilizing Lagrange Polynomial [62] which act as a Secret Key for both sender and recipient, in second segment plain text is fisrt feed 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 fundamental rationale for using UTF-8 is that it is distinct as it encodes characters in one-byte units. Besides, UTF-8 can support all of the readable ASCII characters, as well as the non-readable characters. After getting plain data Encryption is performed using Root-Finding algorithm (Bisection, Newton-Raphson, and Secant) and likewise, Decryption is done by Inverse Root-Finding algorithm where in after applying decryption again UTF-8 is run to get back the original plain text.

Hence, two algorithms have been proposed based on this architecture and are described in section4.

4.1 Interpolation-Based Cryptographic Algorithm (IBGA)

The proposed IBGA initially calls the polynomial generation function, randomly generating the polynomial using Lagrange Interpolation [62] and acting as a key. This is the polynomial shared with the receiver using the secure channel. The encryption and decryption entirely rely upon this polynomial. The plaintext is taken in the blocks of 80bits. 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 Figure 8:

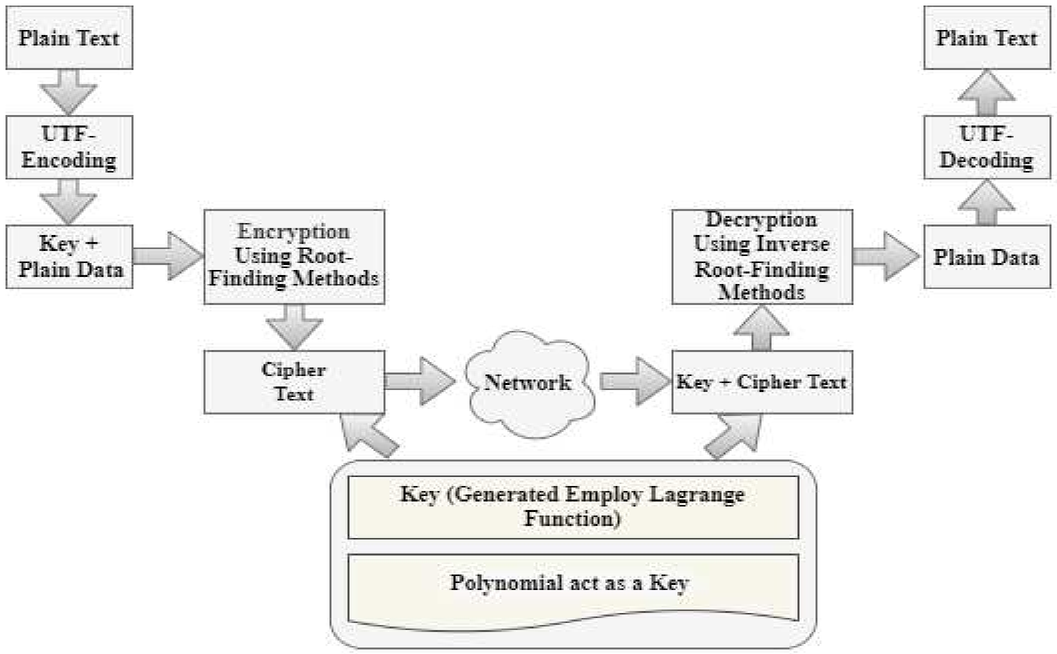

Figure 7: Security Architecture

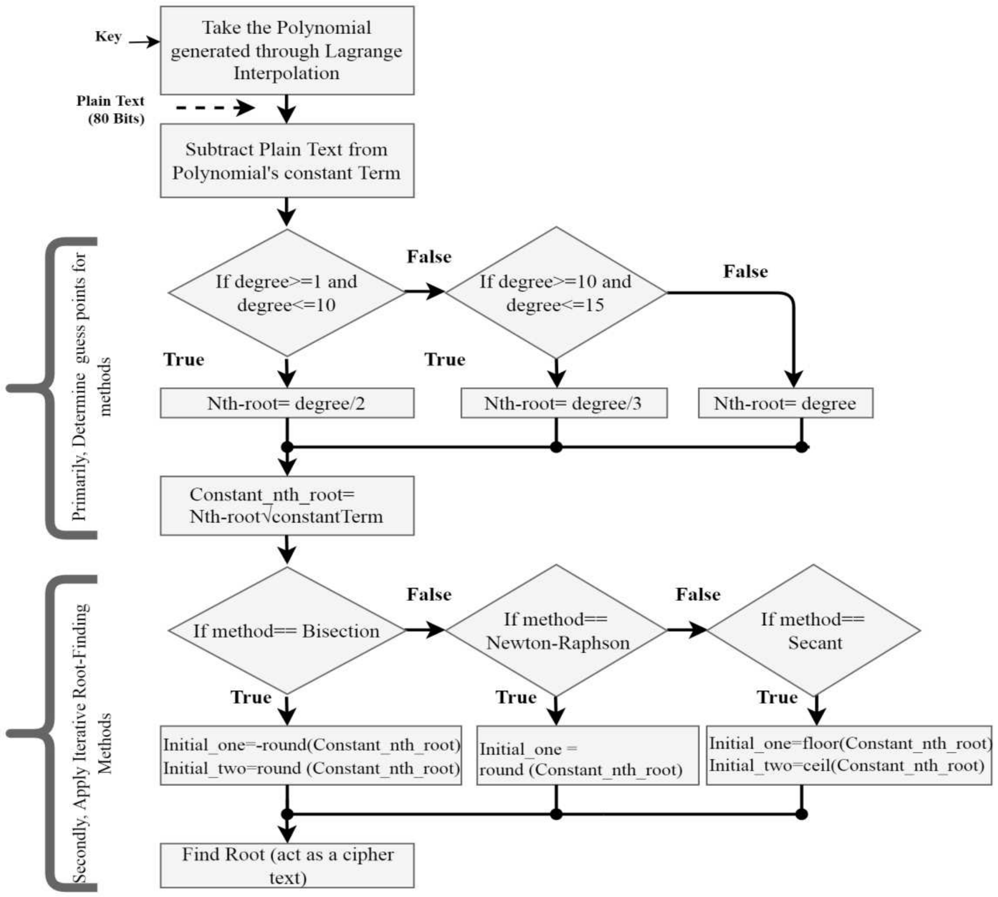
Key
Take the Polynomial generated through Lagrange Interpolation
Plain Text (80 Bits)
Subtract Plain Text from Polynomial's constant Term
Primarily, Determine guess points for methods
If degree>=1 and degree<=10
False
If degree>=10 and degree<=15
False
True
True
Nth-root= degree/2
Nth-root= degree/3
Nth-root= degree
Constant_nth_root= Nth-root√constantTerm
Secondly, Apply Iterative Root-Finding Methods
If method== Bisection
False
If method== Newton-Raphson
False
If method== Secant
True
True
True
Initial_one=-round(Constant_nth_root) Initial_two=round (Constant_nth_root)
Initial_one = round (Constant_nth_root)
Initial_one=floor(Constant_nth_root) Initial_two=ceil(Constant_nth_root)
Find Root (act as a cipher text)

Figure 8: Flowchart for IBGA

4.1.1 Interpolation Based Cryptographic Algorithm - Encryption

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

OUTPUT: Root Value

Step1: PT is taken to be encrypted in the form of digits.

Step2: Obtain the randomised polynomial generated through Lagrange Interpolation Method of odd degree till 13th order [62].

```
poly = lagrange(x, y)
```

Step3: Equate the PT with the obtained polynomial in step2. A new scrambled polynomial will be generated as follows:

```
f = poly-digit

poly_coefficient = Polynomial (f).coef

df = np.polyder(f)

constant = abs(f(0))
```

Step 4: Apply the proposed algorithm-3.5 for Guessing Initial values to be further employed while applying root methods.

Step5: Apply the Root-Finding method (i.e., Bisection, Newton-Raphson, and Secant) and generate the root, which acts as an encrypted text to be sent receiver.

```
if (method=Secant)

root = secant_algorithm(f, initial_one_secant, initial_two_secant, err, iteration) // call secant()

if (method=bisection)

root = bisection_algorithm(f, initial_one_bi, initial_two_bi, iterations) // call bisection()

if (method=newton-raphson)

root = newton_algorithm (f, df, initial_newton, error, iterations) // call newton-raphson
```

Step6: Additionally, compare all three Root-Finding approaches to examine which one works better in this situation.

4.1.2 Interpolation-Based Cryptographic Algorithm - 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)
```

4.1.3 Worked Out Example – IBGA:

A simplified example of nth order polynomial illustrates the working of the algorithm. Nevertheless, the scheme too works with exponential and floating polynomials, as delineated below.

Original Polynomial: $\mathbf{k_n x^n + k_{n-1} x^{n-1} + k_1 x + k_0}$

Digit to be encrypted: **k**

According to the method, subtract **k** from the constant of the original polynomial, i.e., $\mathbf{k_0}$

New polynomial: $\mathbf{k_n x^n + k_{n-1} x^{n-1} + k_1 x + k_0 - k}$

Find initial guess using nth root algorithm, i.e**.** $\mathbf{n/2\sqrt{n}}$

Constant_nth_root = **a**

Apply Bisection / Newton-Raphson method:

Root to be shared (encrypted value) : **b**

4.2 Bi-New

The proposed Bi-New scheme similarly employs a key as IBGA. This method also takes the plaintext in the blocks of 80bits. 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 supposes 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 Figure 9.

4.2.1 Bi – New: Encryption

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

OUTPUT: Root Value

Step1: PT is taken for encryption in the form of digits.

Step2: Obtain the randomised polynomial generated through Lagrange Interpolation Method of odd degree till 13th order [62].

```
        poly = lagrange(x, y)
```

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

```
        f = poly-digit
```

Step4: A new scrambled polynomial has been generated as follows:

```
        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-3.5 for guessing initial values for further use while applying Root-Finding methods.

Step6: 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:]
```

Step8: Afterwards, replace the constant and substitute the root_bisection_1 with negation.

```
poly_coefficient_ left_rotate = np.append(poly_coefficient_ left_rotate, -root_bisection_1)
```

This forms a new polynomial.

Step9: Constructing the new polynomial as:

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

and repeat Step4 and Step5.

Step10: Apply the Newton-Raphson Root-Finding method and generate the root, which acts as a ciphered text to be sent to the recipient.

4.2.2 Bi-New: Decryption

Decryption is performed by reversing all the steps performed while encrypting the text. Decryption is performed when the sender shares the encrypted text and original polynomial. The reverse steps of the encryption are as below:

INPUT: Root Value as an Encrypted Text

OUTPUT: 80 bits Plain Text (PT)

Step1: Take the polynomial as

```
f = poly

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

Step2: 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)
```

Step3: Substitute the root value received from the sender.

```
first_decryption_value = f_ left_rotate (root_final)
```

Step4: 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)

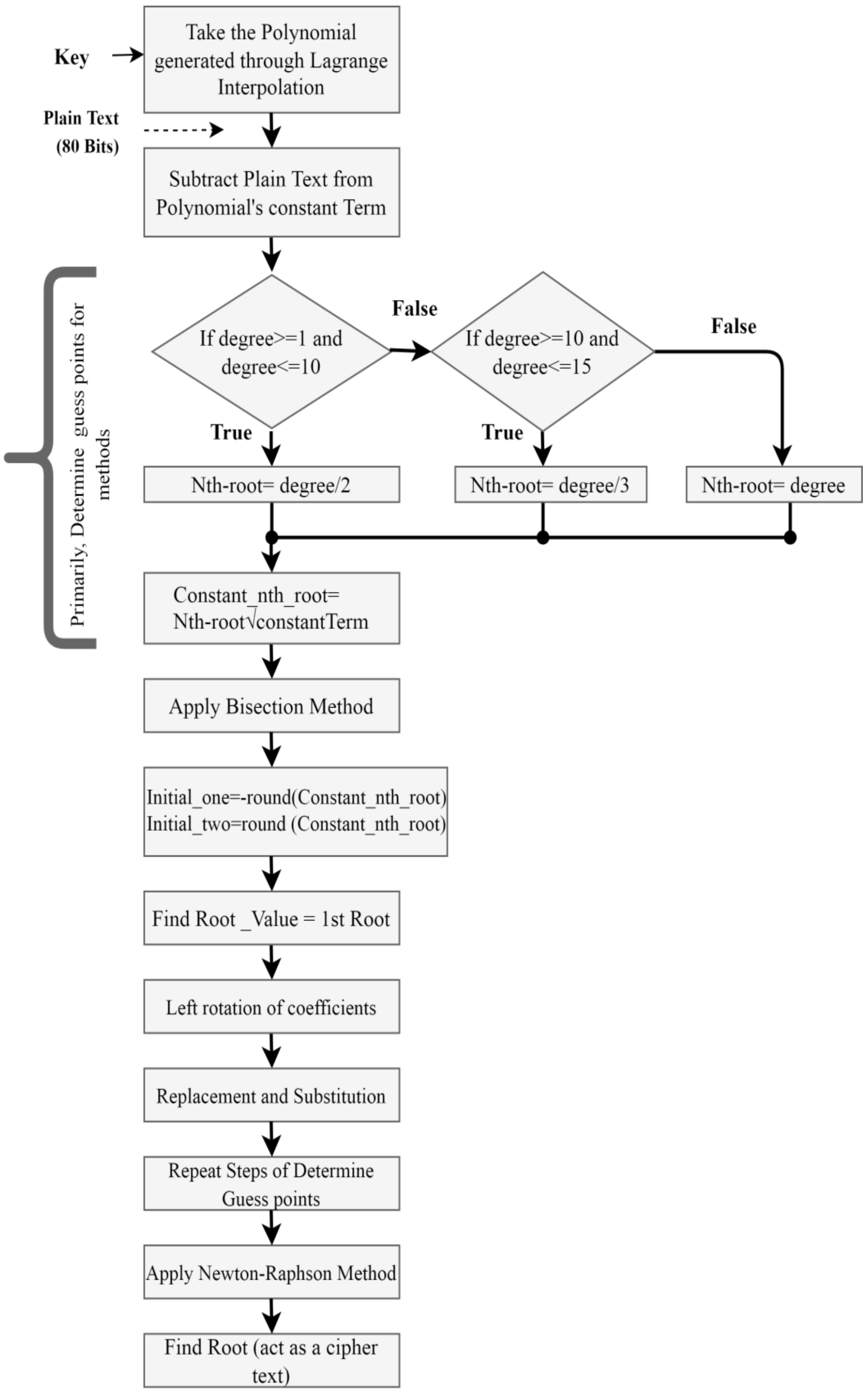
Key
Take the Polynomial generated through Lagrange Interpolation
Plain Text (80 Bits)
Subtract Plain Text from Polynomial's constant Term
Primarily, Determine guess points for methods
If degree>=1 and degree<=10
False
If degree>=10 and degree<=15
False
True
True
Nth-root= degree/2
Nth-root= degree/3
Nth-root= degree
Constant_nth_root= Nth-root√constantTerm
Apply Bisection Method
Initial_one=-round(Constant_nth_root)
Initial_two=round (Constant_nth_root)
Find Root _Value = 1st Root
Left rotation of coefficients
Replacement and Substitution
Repeat Steps of Determine Guess points
Apply Newton-Raphson Method
Find Root (act as a cipher text)

Figure 9: Flowchart for Bi-New

4.2.3 Worked Out Example – Bi-New:

Let's suppose:

Step1: Original polynomial be = $\mathbf{k_n x^n + k_{n-1} x^{n-1} - k_0}$

Digit to be encrypted, i.e. Plain Text = **k**

Step2: According to the method, subtract k from the constant of original polynomial (k_0) and the resultant new polynomial = $\mathbf{k_n x^n + k_{n-1} x^{n-1} - k_p}$

Step3**:** Find the initial guess using the nth root algorithm

Step4: Apply the Bisection Method further for finding the root. Hence, the **Root -1 is a.**

Step5**:** Now apply left rotate akin:

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

Step6: Now replace the new constant, i.e., k_n with the negative value of Root -1

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

Step7: Repeat step3 for the new coefficients of step6.

Step8: Apply the Newton-Raphson Method and find the final root to be shared as an encrypted value with the recipient.

Step9: Hence, the root to be shared = **b**

5. 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 schemes implementations are as delineate below:

(i) Windows 8 operating system
(ii) 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.
(iii) Microsoft office excel is used to plot graphs.

The hardware utilised for the proposed scheme was of the following configurations:

Speed: Intel i-3 processor with 2.50GHz

Memory: 4.00GB RAM

6. Performance and Security analysis

The IBGA and Bi-New performances of the proposed technique were scrutinised for security against various assaults and compared with conventional RSA (1024) and RSA (2048).

6.1 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 was derived from algorithm 3.5. 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 analysed using the following metrics, as delineated below. These metrics rationally convey the scheme's security and effectiveness.

6.1.1 Time Analysis

The computed time of any crypto system is the time taken to encrypt or decrypt the data. 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 gauged in seconds and is recorded as the average time after running the test 100 times. According to the data, the algorithm uses a small number of resources and has high-efficiency levels.

Figure 10 depicts the encryption and decryption results for IBGA, whilst Figure11 represents the running time of Bi-New. The results clearly show that there is no trade-off between key complexity and time complexity. In other words, with an increase in key size, there is a relatively minimal increase in time, making the proposed techniques reliable for real-time implementation.

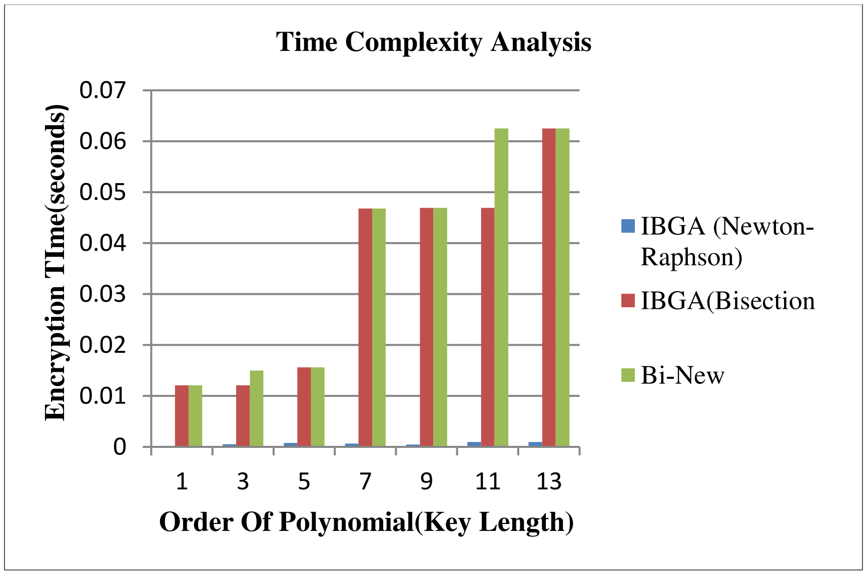

Figure 10: Encryption Time

For the experiment, 80-bit block data was used to encrypt. The graph shows the time analysis in seconds of polynomials of different degrees.

Besides, the execution time, including the Key generation time, Encryption Time and Decryption Time of proposed schemes, were compared with the conventional RSA algorithm of 1024 and 2048 bits. In Figure 12, the highest order of polynomial, the 13th order time analysis, was compared with RSA(1024) and RSA(2048). Overall research shows that the proposed algorithms performed better than RSA.

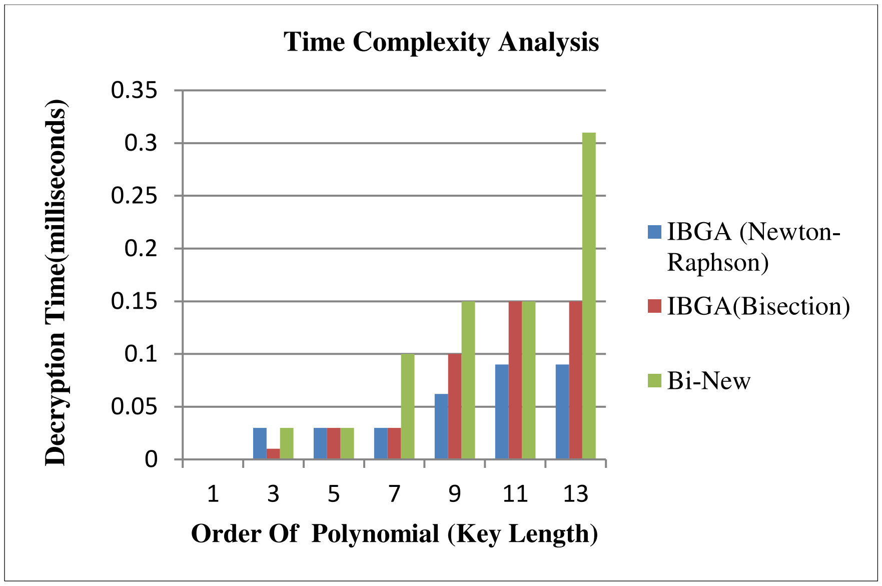

Figure 11: Decryption time

6.1.2 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 equation (5).

$$\text{Accuracy}=\frac{\text{Number of success}}{\text{Number of Success}+\text{Number of failure}} \tag{5}$$

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

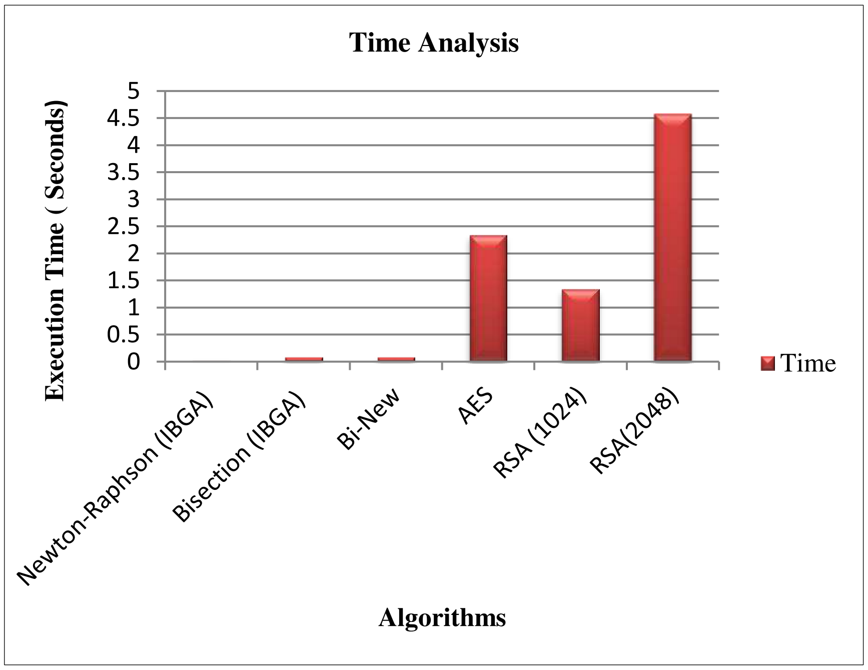

Figure 12: Execution Time

6.1.3 Memory Analysis

Memory allocation was accomplished through memory management of computer applications and services to use virtual or physical memory, wholly or partially, for the program and process execution.

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 IBGA and Bi-new are similar. Figure 13 depicts the memory requirement by the scheme till the 13th order polynomial and its comparison with RSA(1024) and RSA(2048).

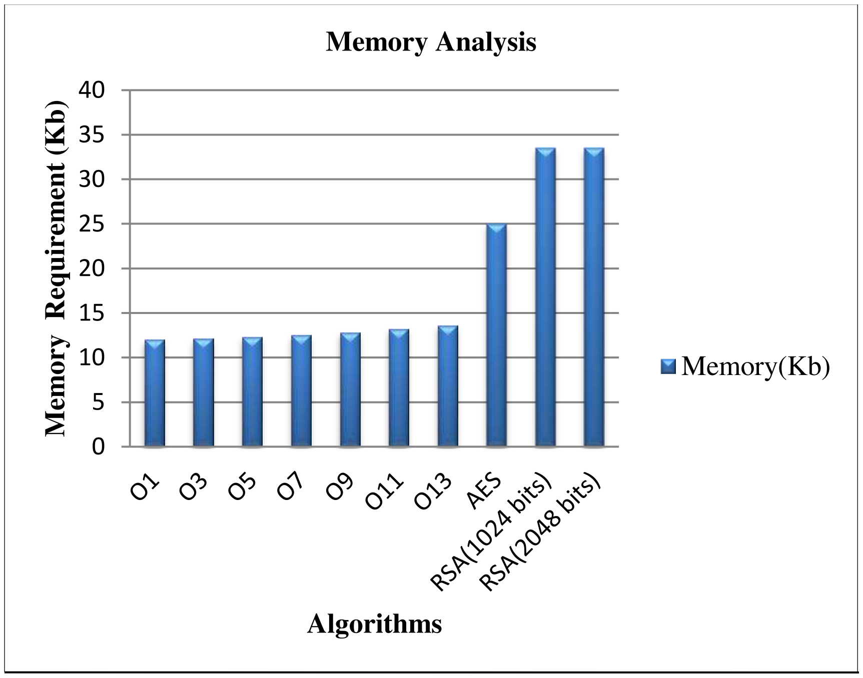

Figure13: Memory Requirement

The results show that the proposed schemes consumed less memory space than RSA. The highest order polynomial also took lesser time than RSA. Still, the cryptanalysis part needs to be analysed in detail to check the strength of the proposed algorithm.

6.2 Security Analysis

The evaluation of the security system is necessary to determine the potency of the cryptosystem in contrast to different encryption threads. On top of that, for a polynomial to be pertinent in a cryptosystem, it must possess specific security characteristics, which can be assessed using techniques presented below.

6.2.1 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 utilises 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).

6.2.2 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. Evidently, the more the algorithm is sensitive to keys, the higher the algorithm's security.

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. The results are tabulated in Table 1.

Table 1: Key Sensitivity

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

The findings have been obtained by executing a loop minimum of 100 times. As shown above, two cases are pictured. In Case I, the results were acquired by taking simple integer coefficients, and it evidently proved that changing the coefficient from 25 to 30 only resulted in an entirely different cypher text. Similarly, a modification of one digit from the initial and end coefficient in Case II and III, the floating polynomial acted as a key and resulted in an entirely different cypher text. Henceforth, this may infer that the proposed work can resist brute-force attacks.

6.2.3 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 unauthorised third party to guess the pattern of encryption and decryption. This is called differential cryptanalysis of attacks performed by attackers, wherein the changes in some chosen plaintexts and differences in the outputs resulting from encrypting each one, possibly recovering the keys, are analysed. The Number of Digit Change Rate (NDCR) was utilised to detect differential attacks.

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 digit change rate was analysed; for an ideal case, the value of NDCR should be >50%. The NDCR was calculated using the parameter in Equations (6) and (7).

$$\mathrm{NDCR}\ (\mathrm{O1}, \mathrm{O2}) = \frac{1}{n}\sum_{i=1}^{n} \mathrm{C}(i) * 100 \qquad (6)$$

$$\mathrm{C}\ (i) = \begin{cases} 0, \text{if } \mathrm{O1} = \mathrm{O2} \\ 1, \text{if } \mathrm{O1} \neq \mathrm{O2} \end{cases} \qquad (7)$$

Where, n is the number of bits; C(i) is calculated by comparing two outputs; O1 represents the output of original plain text; O2 represents the output of changed plaintext.

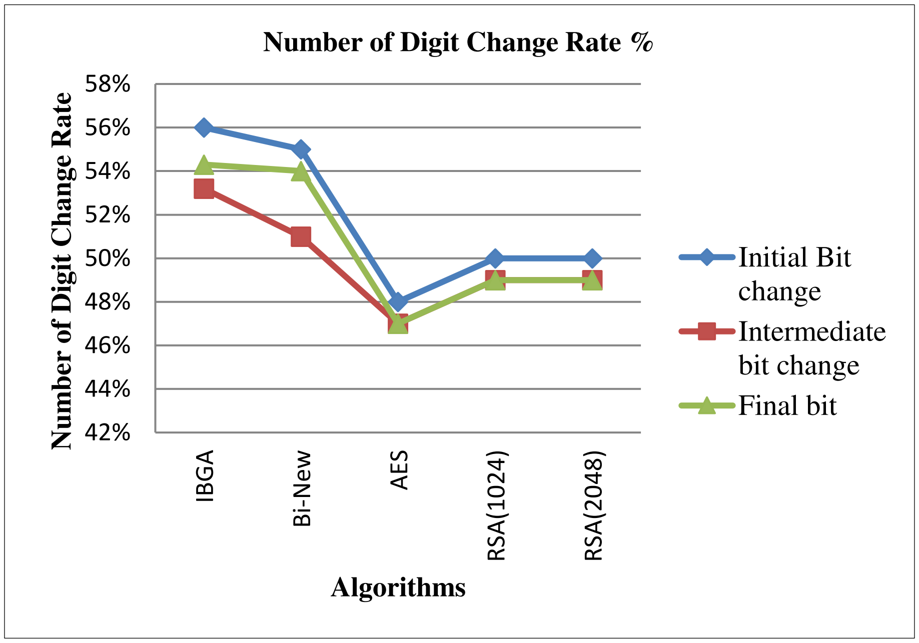

Figure14: NDCR

Figure 14 clearly shows that the proposed schemes have a better NDCR value for ten digits data while changing the final, intermediate, and first digit. Besides, the results compared with RSA-1024 bit and 2048-bit, and the proposed schemes showed improved performance. For all the cases, computed NDCR was close to the ideal Case and exhibited resilience towards differential cryptanalysis attacks.

6.2.4 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 entire relationship among the variables is either by - 1 or 1. The total absence of correlation is signified by 0. Pearson Correlation Coefficient (PCC) is widely accepted to compute correlation among several, and the relation is defined in equation 8 below. In this paper, CC was utilised to calculate the key sensitivity. The key was generated through Lagrange interpolation, using two data points to create a polynomial. Hence, the relation between the M and N data points was computed through correlation.

$$C_{M,N} = \frac{X(M,N)}{\sqrt{Y(M)}\sqrt{Z(N)}} \tag{8}$$

Where:

$$X(M,N) = \sum_{i=1}^{j}(M_i - \bar{M})(N_i - \bar{N}) \tag{9}$$

$$Y(M) = \sum_{i=1}^{j}(M_i - \overline{M})^2 \tag{10}$$

$$Z(N) = \sum_{i=1}^{j}(N_i - \overline{N})^2 \tag{11}$$

Where, M and N are two data points whose values are taken randomly through the generator. j is the number of data points taken. It depends on the degree of a polynomial. (j+1) are data points required for degree d polynomial. X (m,n) is the covariance between points, and Y(m) and Z(n) are the standard deviations of m and n.

Let's suppose the data points are:

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

In equation10:
Y (M) = (0, 0, 4, 4, 81, 81)
Y (M) = 170

In equation 11:
Z (N) = (9, 0, 36, 121, 49, 25)
Z (N) = 240
After getting the results and setting these values in equation 8, drive the correlation coefficient as:

$C_{M,N} = -0.02$

The above data points were generated randomly through the algorithm. A relation was solved by taking desired above values. The results showed that the data points were random as their dependency was null. Similarly, many more results were computed to check the efficiency and varied between 0.0 to 0.4 and -0.0 to 0.4, representing a weak dependency. Thenceforth, no relation among variables proved that data was random, and the proposed algorithm was invulnerable to correlation. Further, while encrypting, the proposed algorithm did not use any Binary functions. Thus correlation attacks were avoided.

7. Conclusions

In this paper, the non-linear polynomial-based schemes were thoroughly delineated based on roots of non-linear algebraic equations and non-linear functions, random generator, array rotation, etc. The Lagrange interpolated polynomial of odd degree till 13th order was used for further evaluation. Three root-finding methods (Bisection, Newton-Raphson, and Secant) were deployed on non-linear interpolating polynomials to find the roots of non-linear algebraic equations. In addition, an evaluation of methods was also carried out, which worked best under various conditions. It was observed that Bisection and Newton-Raphson worked well with nth degree floating-point polynomials, while the Secant method sometimes used arbitrary roots or output. Two schemes have been proposed using these methods. In the first scheme, substitution was performed, and Bisection and Newton-Raphson methods were merely applied to find the cypher text. In the other scheme, results were obtained by combining both methods. This method was executed using the approach of substitution->rotation->replacement->substitution, which provided more confusion and diffusion, thereby enhancing security. The substitution model was accomplished by integrating Lagrange's non-linear polynomial and plain text interpolation. Computing the root through the Bisection method and left rotation of the original polynomial was performed. Replacement and substitution were achieved on the rotated polynomial. Hence, it obscured the association of cyphertext and key.

Consequently, inverse operations carried out the decryption procedure in reverse order. The proposed scheme was compared with the conventional RSA(1024) and RSA(2048) bits on various metrics and depicted the improved performance of the proposed scheme. The experimental results demonstrated that the proposed schemes were efficacious and feasible. The security evaluation unveiled that the proposed scheme owned invulnerable against different assaults such as brute force attacks, differential attacks, and statistical attacks. Therefore, the proposed scheme is inferred to be applicable for practical implementation.

8. Declarations

- **Ethical Approval** : Not Applicable
- **Competing interests** : Not Applicable
- **Authors' contributions** : Not Applicable
- **Funding :** Not Applicable
- **Availability of data and materials**: Not Applicable

References

[1] Dave, Gaurav, Gaurav Choudhary, Vikas Sihag, Ilsun You, and Kim-Kwang Raymond Choo.2022. Cyber security challenges in aviation communication, navigation, and surveillance. Computers & Security. 112. 102516. DOI: https://doi.org/10.1016/j.cose.2021.102516

[2] Hasan, Shaikha, Mazen Ali, Sherah Kurnia, and Ramayah Thurasamy. 2021. Evaluating the cyber security readiness of organizations and its influence on performance. Journal of Information Security and Applications. 58. 102726.DOI: https://doi.org/10.1016/j.jisa.2020.102726.

[3] Jagpreet Kaur and Ramkumar, K.R.. 2021. The recent trends in cyber security: A review. Journal of King Saud University-Computer and Information Sciences.

[4] Parreño, Italo Fernando, and Diego Fernando Avila.2022. Analysis of the Cybersecurity in Wireless Sensor Networks (WSN): A Review Literature. Developments and Advances in Defense and Security. 83-102. DOI:https://doi.org/10.1007/978-981-16-4884-7_8

[5] Biham, Eli, and Adi Shamir. 2012. Differential cryptanalysis of the data encryption standard. Springer Science & Business Media, 2012. [6] Alvarez, Gonzalo, and Shujun Li.2006. Some basic cryptographic requirements for chaos-based cryptosystems. International journal of bifurcation and chaos. 16(8). 2129-2151.
DOI: https://doi.org/10.1142/S0218127406015970.

[7] Meletiou, G. C., D. K. Tasoulis, and M. N. Vrahatis.2003. Cryptography through interpolation approximation and computational intelligence methods. Bulletin of the Greek Mathematical Society. 48.61-75.

[8] Hsiao, Tsung-Chih, Zhen-Yu Wu, Tzer-Long Chen, Yu-Fang Chung, and Tzer-Shyong Chen.2018. A hierarchical access control scheme based on Lagrange interpolation for mobile agents. International Journal of Distributed Sensor Networks. 14(7). 1550147718790892. DOI: https://doi.org/10.1177%2F1550147718790892.

[9] Wang, Xiaogang, Weiren Shi, and Dan Liu.2019. A group key management scheme for WSN based on Lagrange interpolation polynomial characteristic. KSII Transactions on Internet and Information Systems (TIIS). 13(7). 3690-3713.DOI: https://doi.org/10.3837/tiis.2019.07.020.

[10] Hassen, Hani Ragab, Hatem Bettahar, Abdalmadjid Bouadbdallah, and Yacine Challal.2012. An efficient key management scheme for content access control for linear hierarchies. Computer Networks. 56(8). 2107-2118. DOI: https://doi.org/10.1016/j.comnet.2012.02.006.

[11] Wang, Xiaogang, Zhongfan Yang, Zhiqiang Feng, and Jun Zhao.2020.A WSN Layer-Cluster Key Management Scheme Based on Quadratic Polynomial and Lagrange Interpolation Polynomial. Sensors. 20(16). 4388. DOI: https://doi.org/10.3390/s20164388.

[12]Jie, Liew Khang, and Hailiza Kamarulhaili.2011. Polynomial interpolation in the elliptic curve cryptosystem. J. Math. Stat. 7. 326-331.

[13] Bezzateev, Sergey, Vadim Davydov, and Aleksandr Ometov.2020. On Secret Sharing with Newton's Polynomial for Multi-Factor Authentication. Cryptography. 4(4).34. DOI: https://doi.org/10.3390/cryptography4040034

[14] Biswas, Priyajit, Shyamalendu Kandar, and Bibhas Chandra Dhara.2020. An image encryption scheme using sequence--generated by interval bisection of polynomial function. Multimedia Tools and Applications. 79(43).31715-31738. DOI: https://doi.org/10.1007/s11042-020-09497-y.

[15] Sarna, Szymon, and Robert Czerwinski.2021. Small Prime Divisors Attack and Countermeasure against the RSA-OTP Algorithm. Electronics. 11(1).95. DOI: https://doi.org/10.3390/electronics11010095.

[16] Jamal, Sajjad Shaukat, Tariq Shah, Shabieh Farwa, and Muhammad Usman Khan.2019. A new technique of frequency domain watermarking based on a local ring. Wireless Networks. 25(4). 1491-1503. DOI: https://doi.org/10.1007/s11276-017-1606-y.

[17] Waqas, Umer Aziz, Majid Khan, and Syeda Iram Batool. 2020.A new watermarking scheme based on Daubechies wavelet and chaotic map for quick response code images. Multimedia tools and applications. 79(9). 6891-6914.DOI: https://doi.org/10.1007/s11042-019-08570-5

[18] Kavitha, P. K., and P. Vidhya Saraswathi.2018. Color image encryption: A new public key cryptosystem based on polynomial equation. Proceedings in International Conference on ISMAC in Computational Vision and Bio-Engineering, 69-78. Springer, Cham, 2018.DOI: https://doi.org/10.1007/978-3-030-00665-5_8.

[19] Munir, Noor, Majid Khan, Tariq Shah, Ammar S. Alanazi, and Iqtadar Hussain.2021. Cryptanalysis of nonlinear confusion component based encryption algorithm. Integration. 79. 41-47.DOI: https://doi.org/10.1016/j.vlsi.2021.03.004.

[20] Wang, Xingyuan, and Qian Wang.2014. A novel image encryption algorithm based on dynamic S-boxes constructed by chaos. Nonlinear Dynamics. 75(3). 567-576.DOI: https://doi.org/10.1007/s11071-013-1086-2.

[21] Bergamo, Pina, Paolo D'Arco, Alfredo De Santis, and Ljupco Kocarev.2005. Security of public-key cryptosystems based on Chebyshev polynomials. IEEE Transactions on Circuits and Systems I: Regular Papers. 52(7).1382-1393.DOI: 10.1109/TCSI.2005.851701.

[22] Sreedharan, Sujiya, and Chandra Eswaran.2021. A lightweight encryption scheme using Chebyshev polynomial maps. Optik. 240. 166786.DOI: https://doi.org/10.1016/j.ijleo.2021.166786.

[23] Kocarev, Ljupco, and Zarko Tasev. "Public-key encryption based on Chebyshev maps.2003. Proceedings of the 2003 International Symposium on Circuits and Systems. ISCAS'03., vol. 3, pp. III-III. IEEE, 2003. DOI: 10.1109/ISCAS.2003.1204947.
[24] Louzzani, Noura, Abdelkrim Boukabou, Halima Bahi, and Ali Boussayoud.2021. A novel chaos based generating function of the Chebyshev polynomials and its applications in image encryption. Chaos, Solitons & Fractals. 151. 111315. DOI: https://doi.org/10.1016/j.chaos.2021.111315.
[25] Deepika, M. P., and A. Sreekumar. 2017.A Key Distribution Scheme in Broadcast Encryption Using Polynomial Interpolation. International Journal of Applied Engineering Research. 12(24). 15475-15483..DOI:
[26] Yoo, E.S., Jho, N.S., Cheon, J.H. and Kim, M.H., 2004, December. Efficient broadcast encryption using multiple interpolation methods. Proceedings in International Conference on Information Security and Cryptology (pp. 87-103). Springer, Berlin, Heidelberg.
[27] Patil, Priyadarshini, Prashant Narayankar, D. G. Narayan, and S. Md Meena.2016. A comprehensive evaluation of cryptographic algorithms: DES, 3DES, AES, RSA and Blowfish. Procedia Computer Science. 78.617-624..DOI: https://doi.org/10.1016/j.procs.2016.02.108.
[28] Zodpe, Harshali, and Ashok Sapkal.2020. An efficient AES implementation using FPGA with enhanced security features. Journal of King Saud University-Engineering Sciences. 32(2).115-122.
DOI: https://doi.org/10.1016/j.jksues.2018.07.002.
[29] Samalkha, S. G. I.2013. Efficient Implementation of AES. International Journal.3(7).
[30] Wang, Qian, An Wang, Liji Wu, and Jiliang Zhang.2016. A new zero value attack combined fault sensitivity analysis on masked AES. Microprocessors and Microsystems. 45. 355-362.DOI: https://doi.org/10.1016/j.micpro.2016.06.014.
[31] Tunstall, Michael, Debdeep Mukhopadhyay, and Subidh Ali.2011. Differential fault analysis of the advanced encryption standard using a single fault. Proceedings in IFIP international workshop on information security theory and practices. 224-233. Springer, Berlin, Heidelberg..DOI: https://doi.org/10.1007/978-3-642-21040-2_15.
[32] Liu, Yixia, Xiaoxin Cui, Jian Cao, and Xing Zhang.2017. A hybrid fault model for differential fault attack on AES." In 2017 IEEE 12th International Conference on ASIC (ASICON), pp. 784-787. IEEE, 2017.DOI: https://doi.org/10.1109/ASICON.2017.8252593
[33] Jie, C., Liusheng, H., Hong, Z. and Wei, Y., 2010, August. Improved related-key attack on 7-round AES-128/256. Proceedings in 2010 5th International Conference on Computer Science & Education (pp. 462-466). IEEE.DOI: https://doi.org/10.1109/ICCSE.2010.5593579.
[34] Biryukov, A., Dunkelman, O., Keller, N., Khovratovich, D. and Shamir, A., 2010, May. Key recovery attacks of practical complexity on AES-256 variants with up to 10 rounds. Proceedings in Annual International Conference on the Theory and Applications of Cryptographic Techniques (pp. 299-319). Springer, Berlin, Heidelberg.DOI: https://doi.org/10.1007/978-3-642-13190-5_15.
[35] Bar-On, Achiya, Orr Dunkelman, Nathan Keller, Eyal Ronen, and Adi Shamir.2020. Improved key recovery attacks on reduced-round AES with practical data and memory complexities. Journal of Cryptology. 33(3).1003-1043.DOI: https://doi.org/10.1007/s00145-019-09336-w.
[36] Esfahani, Mahdi, Hadi Soleimany, and Mohammad Reza Aref. "Modified Cache Template Attack on AES." Cryptology ePrint Archive (2020).
[37] Yang, Xiaoya, Yongchuan Niu, Qingping Tang, Jiawei Zhang, Yaoling Ding, and An Wang.2020. Near and Far Collision Attack on Masked AES. In International Conference on Computer Engineering and Networks, pp. 810-817. Springer, Singapore, 2020.. DOI:https://doi.org/10.1007/978-981-15-8462-6_93
[38] Huang, James, and Xiaoming Li.2020. Cache-collision side-channel analysis and attacks against AES-GCM. International Journal of Big Data Intelligence. 7(4). 211-217.
[39] Rivest, Ronald L., Adi Shamir, and Leonard M. Adleman.2019. A method for obtaining digital signatures and public key cryptosystems. In Secure communications and asymmetric cryptosystems. 217-239.
[40] Kessler, G.C., 2012. Introduction to cryptography.
[41] Moghaddam, F. Fatemi, Maen T. Alrashdan, and Omidreza Karimi.2013. A hybrid encryption algorithm based on rsa small-e and efficient-rsa for cloud computing environments. Journal of advances in Computer Network.1(3).
[42] Susilo, Willy, and Joseph Tonien.2021. A Wiener-type attack on an RSA-like cryptosystem constructed from cubic Pell equations. Theoretical Computer Science. 885.125-130.DOI: https://doi.org/10.1016/j.tcs.2021.06.033.
[43] Kota, Chandra M., and Cherif Aissi.2022. Implementation of the RSA algorithm and its cryptanalysis. In 2002 GSW.DOI:. 10.18260/1-2-620-38785.
[44] Hirata, Tomonori, and Yuichi Kaji. "Information leakage through passive timing attacks on RSA decryption system." In 2020 International Symposium on Information Theory and Its Applications (ISITA), pp. 392-396. IEEE, 2020.DOI:

[45] Le, Duc-Phong, Rongxing Lu, and Ali A. Ghorbani.2021. Improved fault analysis on SIMECK ciphers. Journal of Cryptographic Engineering. 1-12. DOI: https://doi.org/10.1007/s13389-021-00263-w.
[46] Mumtaz, Majid, and Luo Ping.2019. Forty years of attacks on the RSA cryptosystem: A brief survey. Journal of Discrete Mathematical Sciences and Cryptography. 22(1). 9-29.DOI: https://doi.org/10.1080/09720529.2018.1564201.
[47] Miller, Stephen D., Bhargav Narayanan, and Ramarathnam Venkatesan.2021. Coppersmith's lattices and "focus groups. An attack on small-exponent RSA. Journal of Number Theory. 222.376-392.DOI:https://doi.org/10.1016/j.jnt.2021.01.002.
[48] Κουνής, K.E., 2021. Boneh-Durfee attack in RSA (No. GRI-2021-30033). Aristotle University of Thessaloniki.
[49] Nitaj, Abderahmanne, Muhammad Rezal Kamel Ariffin, Nurul Nur Hanisah Adenan, Domenica Stefania Merenda, and Ali Ahmadian. 2021. Exponential increment of RSA attack range via lattice based cryptanalysis. Multimedia Tools and Applications.1-16.DOI:https://doi.org/10.1007/s11042-021-11335-8 .
[50] Ruzai, Wan Nur Aqlili Wan Mohd, Abderrahmane Nitaj, Muhammad Rezal Kamel Ariffin, Zahari Mahad, and Muhammad Asyraf Asbullah.2022. Increment of insecure RSA private exponent bound through perfect square RSA diophantine parameters cryptanalysis. Computer Standards & Interfaces.80. 103584.
[51] Badr, Elsayed, Sultan Almotairi, and Abdallah El Ghamry.2021. A comparative study among new hybrid root finding algorithms and traditional methods. Mathematics. 9(11).1306.DOI: https://doi.org/10.3390/math9111306.
[52] Biswa Nath Datta.2012. Lecture Notes on Numerical Solution of root Finding Problems. Available on: www.math.niu.edu/dattab.
[53] Pratap, Rudra.2010. Getting started with MATLAB: a quick introduction for scientists and engineers. New York: Oxford University Press.
[54] M.G.Moazzam, A.Chakraborty and M.A.A.Bhuiyan.A. 2012. A robust method for solving transcendental equations. International Journal of Computer Science Issues (IJCSI). 9(6).413.
[55] Ehiwario, J. C., and S. O. Aghamie.2014. Comparative study of bisection, Newton-Raphson and secant methods of root-finding problems. IOSR Journal of Engineering. 4(4). 01-07.
[56] Chapra, Steven C.2008. Applied numerical methods with MATLAB for engineers and scientists. McGraw-Hill Higher Education.
[57] Ghitany, M. E., Dhaifalla K. Al-Mutairi, and Saralees Nadarajah.2008.Zero-truncated Poisson–Lindley distribution and its application. Mathematics and Computers in Simulation. 79(3). 279-287.DOI: https://doi.org/10.1016/j.matcom.2007.11.021.
[58] Sankaran, Munuswamy.1970. 275. note: The discrete poisson-lindley distribution." Biometrics (1970): 145-149..
[59] Shanker, Rama, F. Hagos, S. Sujatha, and Y. Abrehe.2015. On zero-truncation of Poisson and Poisson-Lindley distributions and their applications. Biometrics & Biostatistics International Journal. 2(6).1-14.
[60] Allen III, Myron B., and Eli L. Isaacson. 2011.Numerical analysis for applied science. Vol. 35. John wiley & sons.
[61] Burden, Richard L., and J. Douglas Faires.2010. Numerical analysis (nineth edition)." Thomson Brooks/Cole . 57-58.
[62] Kaur, J. and Kumar, K.R., 2022, Key Management using Lagrange Interpolation for Wireless Communications. In 2022 2nd International Conference on Advance Computing and Innovative Technologies in Engineering (ICACITE) (pp. 2084-2087). IEEE.