

Secured Digital Healthcare system using Noetherian Rings and SHA Algorithm

C. Senthilnathan ¹, S. Karunanithi²

^{1,2} P G & Research Department of Mathematics, Government Thirumagal Mills College, Affiliated to Thiruvalluvar University, Serkadu, Vellore, Tamil Nadu, India.

Email ID : duriselvan@gmail.com¹, kap232008@gmail.com²

ABSTRACT

In the digital health era, safeguarding medical data is of utmost importance to ensure confidentiality, integrity, and availability. This paper presents a mathematical approach to encryption and secure data management, leveraging vector spaces and matrix operations to enhance the security of medical records. By integrating advanced encryption techniques rooted in linear algebra, we propose a framework that fortifies data protection while maintaining authorized accessibility. Furthermore, we explore the mathematical underpinnings of the Secure Hash Algorithm (SHA), with a particular emphasis on its cryptographic security. Our analysis dissects SHA's structural components through mathematical constructs such as vector spaces, linear transformations, and permutation matrices. By establishing correlations between these mathematical principles and SHA's operational processes, we gain deeper insights into its robustness against cryptographic attacks. The proposed framework not only reinforces the security of medical records but also provides a systematic approach to analyzing cryptographic techniques through a mathematical lens. This study underscores the pivotal role of linear algebra in contemporary encryption methods and highlights its potential in securing sensitive medical information in an increasingly digital landscape.

Key Words: SHA Algorithm, attribute-based encryption, decryption, matrix invertibility..

How to Cite: C. Senthilnathan , S. Karunanithi, (2025) Secured Digital Healthcare system using Noetherian Rings and SHA Algorithm., *Journal of Carcinogenesis*, Vol.24, No.4s, 538-549

1. INTRODUCTION

The transition to digital health records has revolutionized the healthcare industry by streamlining patient management, enhancing diagnostic precision, and improving treatment effectiveness. These are discussed in terms of secrecy by, Shannon, C. E [1] along with cryptography. Electronic Health Records (EHRs) along with medical image processing was discussed by Abdul Razaq et al [2], it allows for seamless data sharing among healthcare professionals, leading to more coordinated and efficient care. However, the sensitive nature of medical data requires stringent security protocols to prevent unauthorized access, data breaches, and cyber threats are enhanced by Katz J et al [3].

Medical records contain highly confidential information, including patient demographics, medical histories, diagnoses, treatment plans, and laboratory results and enriched the concepts of modern cryptography by D. R. Stinson et al [4]. Unauthorized access to such data can lead to privacy violations, identity theft, and financial fraud are enhanced by Salvatore J [5]. Moreover, healthcare organizations are prime targets for cyber attacks, with incidents of ransomware and data breaches increasing globally, especially in the cloud environment by Inam S et al [6]. Regulatory frameworks such as the **Health Insurance Portability and Accountability Act (HIPAA)** and the **General Data Protection Regulation (GDPR)** mandate strict security measures to protect patient information and ensure compliance with privacy laws [7].

To mitigate security risks, advanced encryption techniques, secure authentication mechanisms, and access control policies must be implemented [8]. Cryptographic methods, including **public key encryption, secure hashing algorithms, and homomorphic encryption**, play a crucial role in safeguarding medical records [9]. Furthermore, integrating artificial intelligence and blockchain technology in healthcare security frameworks can enhance data protection, ensuring confidentiality, integrity, and availability of medical information. As digital healthcare continues to expand, adopting a **mathematical and algorithmic approach** to data security, such as leveraging **linear algebra-based encryption methods and cryptographic protocols** can further strengthen the resilience of medical data management systems [10, 11]. This paper explores the security challenges in medical data management and introduces a linear algebra-based framework to enhance the protection of sensitive information [12].

Medical data includes a wide range of information, such as patient demographics, medical history, treatment plans, and diagnostic results. In the United States, the **Health Insurance Portability and Accountability Act (HIPAA)** enforces strict regulations to safeguard patient privacy and ensure the confidentiality of medical records [13]. Non-compliance with these regulations can result in significant penalties and erode patient trust.

Linear algebra offers a strong mathematical foundation for designing encryption algorithms. Key concepts, including vector spaces, matrices, and transformations, are essential for encoding and securing data [14]. Methods like the **Hill cipher** apply linear transformations to convert plaintext into ciphertext, making it challenging for unauthorized users to decode the information without the appropriate decryption key [15].

Public key cryptography utilizes matrix operations in encryption and decryption processes. While the **RSA (Rivest-Shamir-Adleman) algorithm** primarily relies on modular arithmetic, certain variations incorporate linear algebra concepts. **Error-correcting codes**, such as **Reed-Solomon codes**, play a crucial role in secure communications by using linear algebra to detect and recover lost or corrupted data [16].

Homomorphic encryption enables computations on encrypted data and often employs matrix operations or vector space techniques. Similarly, **stream ciphers and block ciphers**, including the **Advanced Encryption Standard (AES)**, integrate linear transformations within their encryption structure [17]. Additionally, linear transformations help define how data is processed during encryption and decryption [18]. In many cryptographic systems, the set of possible keys can be represented as a **vector space**, offering a mathematical framework for key management and security analysis [19, 20].

The **Hill cipher** is a well-known encryption method that directly applies linear algebra. In this technique, plaintext is divided into blocks and represented as vectors, while the encryption key is structured as a matrix [21]. The encryption process involves multiplying the plaintext vector by the key matrix, followed by modular arithmetic (typically modulo 26 for alphabetic encryption). Decryption requires computing the inverse of the key matrix and applying it to the ciphertext [22].

However, not all matrices are invertible, and determining the inverse can be computationally demanding [23]. Additionally, linear transformations may be susceptible to certain cryptographic attacks, such as **linear cryptanalysis**. As a block cipher, the **Hill cipher** relies on matrix multiplication and modular arithmetic to secure messages, demonstrating the fundamental role of linear algebra in encryption [24].

However, a non-Noetherian ring may be a sub ring of a Noetherian ring. For instance any integral field is also a sub ring of a field, and any intact of Noetherian provides a precedent. However this could be exact if R_l is not commutative: the ring R_l is a sub-ring of the left Noetherian ring $S_l = Hom(Q_{l_2}, Q_{l_2})$, and S_l is limitedly generated as a left R_l -module, but R_l is a left Noetherian.

2. MATHEMATICAL MODEL BLOCK REPRESENTATION

The plaintext is split into blocks of size n and represented as column vectors.

$$P = \begin{bmatrix} P_1 \\ P_2 \\ P_3 \\ P_4 \\ \vdots \\ \vdots \\ \vdots \\ P_n \end{bmatrix}$$

Each P_i , $i = 0$ to 25, represents a letter converted into a number (e.g., $A = 0, B = 1, \dots, Z = 25$).

2.1 Key Concepts and Equations in the Hill Cipher

2.1.1. Matrix Invertibility: The key matrix K must be invertible modulo m . The determinant $\det K$ must be coprime with m (i.e., $\gcd(\det(K), m) = 1$). This ensures that K has an inverse.

2.1.2. Modular Arithmetic : In modular arithmetic, all operations are performed modulo m , ensuring that results stay within the alphabet range.

2.1.3. Block Size: The choice of block size n affects both security and efficiency. Larger blocks can provide more security but may complicate the encryption process.

2.1.4. Linear Transformation: The encryption function $C = K * P$ represents a linear transformation. Each plaintext letter is transformed linearly based on the key matrix.

2.1.5 Key Matrix: An $n \times n$ matrix K is defined, where the entries are integers:

$$\begin{bmatrix} k_{11} & k_{12} & \cdots & k_{1n} \\ \vdots & & \ddots & \vdots \\ k_{n1} & \vdots & \cdots & k_{nn} \end{bmatrix}$$

2.1.6. Encryption Equation: The encryption process can be mathematically described as:

$$C = K \cdot P \text{ mod } m$$

where C is the resulting ciphertext vector, and m is the modulus (commonly 26 for the English alphabet).

2.1.7. Decryption Equation: For decryption, the inverse of the key matrix K^{-1} is utilized and the inverse matrix is updated as:

$$P_{ij} = K^{-1} C \text{ mod } m$$

2.1.8. Determinant and Inverse: For the key K to be invertible, its determinant $\det(K)$ must satisfy: $\gcd(\det(K), m) = 1$

The following are the ciphertext vector elements :

For an $n \times n$ matrix and plaintext vector, each element of the ciphertext vector can be represented as:

$$C_i = \sum_{j=1}^n K P_{ji}$$

For Encryption:

Choose a random r .

Compute $c_1 = g^r$, $c_2 = m \cdot h^r$ and $c_3 = H(c_1, m, r)$. The ciphertext is (c_1, c_2, c_3)

To decrypt, the receiver checks:

Compute $c'_3 = H(c_1, c_2 / h^{H(c_1, c_2)}, r)$

If $c'_3 = c_3$, return $m = c_2 / h^{H(c_1, c_2)}$.

Security Reduction: Assume an adversary can break the scheme. We construct an algorithm to solve the DDH (Decisional Diffie–Hellman) problem using the adversary's ability. The adversary can distinguish between tuples of the form (g^a, g^b, g^{ab}) and (g^a, g^b, g^c) for random c . The ability to distinguish leads to a contradiction against the hardness of the DDH assumption.

To show a security reduction, we assume an adversary A can break the cryptographic scheme and use it to construct an algorithm B that solves the Decisional Diffie–Hellman (DDH) problem.

Decisional Diffie–Hellman (DDH) Problem

Given a cyclic group G of prime order q with generator g , and a tuple g^a, g^b, g^c for random exponents $a, b, c \in \mathbb{Z}_q$ decide whether:

$$c = ab \text{ mod } q, \quad gc = gabg^c = g^{\{ab\}}$$

c is random and g^c is independent of g^a, g^b .

An algorithm that can distinguish between these two cases with non-negligible probability solves the DDH problem.

Reduction Idea

We assume that an adversary A can break some cryptographic scheme (e.g., an encryption scheme based on Diffie–Hellman). We construct an algorithm B that uses A to distinguish between a valid DH triple and a random tuple.

Step 1: Algorithm B Receives a DDH Challenge

The challenger gives B a tuple: (g, g^a, g^b, g^c) where (g^a, g^b) are random elements of G , and g^c is either:

$g^{\{ab\}}$ (a real DH triple), or

g^r for random r (independent of a and b).

B must determine which case holds.

Step 2: Simulating the Cryptographic Scheme

B acts as the challenger in the security game of the scheme and interacts with adversary A .

If the scheme is Diffie–Hellman-based encryption, the adversary expects a ciphertext involving $g^{\{ab\}}$

B replaces $g^{\{ab\}}$ with g^c in the encryption process.

A attempts to break the scheme, expecting a ciphertext of $g^{\{ab\}}$.

Step 3: Using A to Solve DDH

If $g^c = g^{\{ab\}}$ A may successfully break the scheme.

If g^c is random, A is less effective because the structure of $g^{\{ab\}}$ is lost.

B observes whether A behaves differently in these two cases. If A breaks the scheme in one case but fails in the other, B can distinguish between $g^c = g^{\{ab\}}$ and random g^c , solving the DDH problem. Since we assumed that A breaks the scheme, we constructed B to leverage this ability to solve DDH. If the scheme is secure under DDH, A must fail, contradicting our assumption. Hence, breaking the scheme implies solving DDH, reducing the scheme's security to the hardness of the DDH problem.

Error-correcting codes play a crucial role in communication systems by detecting and correcting errors that may arise during data transmission. Linear algebra provides powerful techniques for constructing and analyzing these codes.

The following are an overview of linear error-correcting codes, along with a mathematical explanation and proof of their error correction capability.

2.1.9 Linear Codes

A linear code can be defined as a subspace of a vector space over a finite field. For a code of length n and dimension k , the code can be represented as a k -dimensional subspace of F_q^n , where F_q is a finite field of size q .

Code Representation

Let G be a $k \times n$ generator matrix for the code. Each codeword c in the code can be generated by:

$$c = u * G$$

where u is a k -dimensional vector of message symbols.

2.1.10 Hamming Distance

The Hamming distance $d(c_1, c_2)$ between two codewords c_1 and c_2 is the number of positions in which the two codewords differ. The minimum distance d_{min} of the code is defined as the smallest Hamming distance between any two distinct codewords. The error correction capability t of the code is given by:

$$t = \frac{d_{min} - 1}{2}$$

This means the code can correct up to t errors in a codeword.

2.1.11 Error Correction

To demonstrate error correction mathematically, consider a scenario where a transmitted codeword c is corrupted during transmission, resulting in the received vector r :

$$R = c + e$$

where e is the error vector with up to t non-zero entries.

Proof of Error Correction Capability

Step 1: Codeword Recovery

Let c_i and c_j be two distinct codewords from the code. The Hamming distance between these codewords is given by:

$$d_{min} = d(c_i, c_j) \geq 2t + 1$$

This implies that the closest codeword c_i is at least $t + 1$ positions away from the corrupted received vector r .

Step 2: Decision Rule

To decode the received vector r , the decoder will find the codeword c_i that minimizes the distance:

$$\hat{c} = \arg \min_{c_j \in C} d(r, c_j)$$

Step 3: Error Correction

If

$$d(r, c_i) < t$$

Since $d_{min} \geq t + 1$, we have:

$$d(c_i, c_j) \geq 2t + 1$$

This means the distance from r the correct codeword c_i is less than the distance to any other code word c_j , ensuring that the decoder will correctly choose c_i .

This highlights the mathematical relationships between linear algebra and the SHA family, focusing on vulnerabilities and methods to reduce the complexity of finding pre-images and collisions. By analyzing bitwise operations, permutations, and transformations, we establish a framework for understanding the security properties of SHA algorithms. The exploration of enhancing non-linearity, increasing output sizes, and employing stronger permutations emphasizes the importance of robust design in cryptographic functions.

3. ENCRYPTION USING LINEAR TRANSFORMATIONS

To encrypt medical data, we can use a linear transformation defined by a generator matrix. Each patient record vector x can be transformed into an encrypted vector y using the formula:

$$y = x \cdot G$$

3.1 Key Generation

The generator matrix G must be carefully chosen to ensure invertibility. The determinant of G must be nonzero, satisfying the condition:

$$\det(G) \neq 0$$

This ensures that G has an inverse, enabling decryption.

3.2 Decryption Process

The decryption process utilizes the inverse of the generator matrix G^{-1} . Given an encrypted vector y , the original patient record vector x can be recovered by:

$$x = y \cdot G^{-1}$$

3.3 Error Detection and Correction

To enhance data integrity, we can integrate error detection and correction mechanisms. By augmenting the original data matrix with redundant information, we can use linear combinations of rows to detect and correct errors.

3.4. Mathematical Framework

The security of our method hinges on the properties of matrix operations. Given an encryption matrix G , we analyze the implications of matrix multiplication, particularly focusing on the distribution of errors across encrypted data.

We implement the proposed framework in a simulated healthcare environment. Patient records are encrypted using a randomly generated invertible matrix G . The performance of our approach is evaluated based on encryption time, decryption time, and error correction capability.

Cryptographic hash functions are essential for data integrity and security. Among them, the SHA family, including SHA-1, SHA-2, and SHA-3, plays a crucial role in various applications. This paper aims to explore the mathematical principles underlying SHA algorithms through linear algebra, emphasizing how these principles contribute to the algorithms' security and efficiency.

4. OVERVIEW OF THE SHA FAMILY

4.1 Hash Function Properties

A secure hash function should possess several essential properties:

Figure 1 describes the process of maintaining patients e record and figure 2 explains the SHA – 256 algorithm.

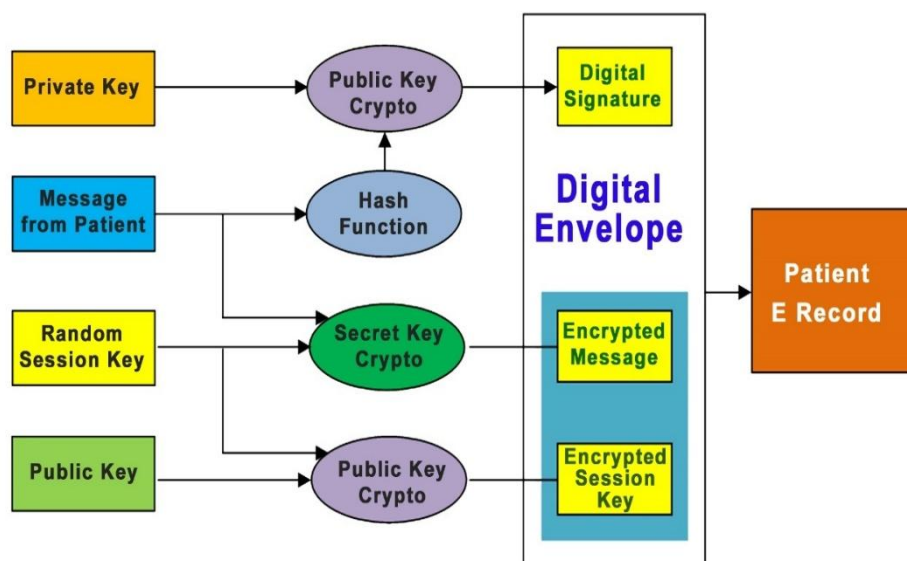


Figure 1: Flow chart for SHA Algorithm

4.2 The Role of Linear Transformations in SHA

In SHA-2, the compression function incorporates multiple linear transformations. The transformation can be expressed as:

$$H_{i+1} = H_i + T(H_i, W_i, K_i)$$

where T is a combination of linear and nonlinear operations defined to involve functions that mix bits through both linear and nonlinear mappings.

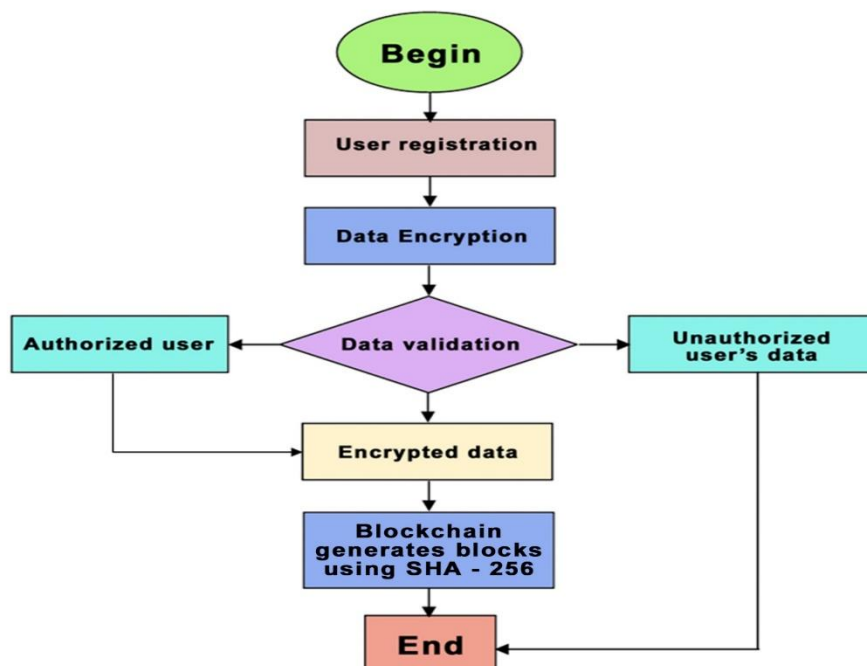


Figure 2: Authorized Patient's SHA – 256 Algorithm process

4.3 Vulnerabilities and Attacks

Differential cryptanalysis exploits the propagation of differences through the rounds of the hash function. It focuses on input pairs with specific differences and analyzes how these differences affect the output.

Theorem 4.3.1 (Differential Approximation): Let Δ_x be an input difference and Δ_y the resulting output difference. If the probability $P(\Delta_y)$ for a particular output difference is significantly greater than $\frac{1}{2^n}$, then

$$P(\Delta_y) = \frac{1}{2^n} + \varepsilon$$

where ε is a bias introduced by the differential path. If an attacker can find such Δ_x , they can potentially produce collisions. Linear cryptanalysis uses linear approximations to describe the behavior of the hash function. It searches for linear equations relating input bits to output bits. The bias can be exploited to build a more efficient attack, allowing the attacker to predict the output with a probability significantly better than random guessing.

This means that for a 256-bit hash function, an attacker needs approximately 2^{128} attempts to find a collision. The security of hash functions is critically influenced by their design, particularly their algebraic structure. Traditional hash functions often rely on linear transformations and simple mixing functions, making them susceptible to various cryptographic attacks. This section explores the implementation of hash functions with higher algebraic degrees, which can enhance security by complicating the relationships between inputs and outputs.

The choice of coefficients and the degree d significantly impacts the function's resistance to attacks.

4.4 Example of a Polynomial Hash Function : Consider a cubic polynomial hash function defined as:

$$h(m) = (a_0 + a_1 m + a_2 m^2 + a_3 m^3) \bmod p$$

where p is a prime number. The coefficients a_0, a_1, a_2 and a_3 are randomly chosen from F_p . The output will be influenced more significantly by higher-degree terms, making it challenging for an attacker to derive linear relationships.

Theorem 4.4.1 (Collision Resistance): Increased Complexity of Finding Collisions

For a polynomial hash function of degree d , the expected number of operations required to find a collision is $2^{\frac{n}{d+1}}$, where n is the output size of the hash function. This implies that as d increases, the complexity of finding collisions grows significantly.

Lemma 4.4.2 :Pre-image Resistance (Pre-image Complexity):

Let $h(m)$ be a polynomial hash function of degree d . The complexity C of finding a pre-image m' for a given hash value y is $C \geq \frac{p^d}{n}$.

This indicates that higher-degree polynomials require more effort to reverse-engineer.

4.5 Random Coefficient Selection

To maximize security, coefficients a_i should be chosen randomly and uniformly from F_p . This randomness ensures that the polynomial behaves unpredictably and minimizes the risk of attackers discovering exploitable patterns.

4.5.1 Modular Arithmetic

Utilizing modular arithmetic not only prevents overflow but also ensures that the polynomial remains within a manageable range. This is crucial for maintaining consistent output sizes. Combining several polynomial hash functions can further enhance security. A composite hash function can be defined as:

$$H(m) = h_1(m) \oplus h_2(m) \oplus h_3(m) \oplus \dots \oplus h_k(m)$$

where each h_i is a polynomial hash function of degree d_i . The use of the XOR operation combines outputs in a manner that obscures any linear relationships further. This construction improves collision resistance, ensuring that different inputs are less likely to produce the same output. XOR-based combinations of hash functions are often used in error detection and correction algorithms. If one hash function has some bias, combining it with others can help reduce that bias.

Higher-degree polynomials are resistant to algebraic attacks, which exploit relationships between input bits and output bits. The complexity of deriving such relationships increases exponentially with the polynomial degree.

This suggests that using more hash functions or larger hash output sizes makes it exponentially harder to find two inputs $h_1(m)$ and $h_2(m)$ such that $h_1(m) = h_2(m)$.

In this context, R may also represent entropy or the security level of the construction. Combining multiple hash functions using XOR increases the effective entropy of the output, enhancing security. The proportional relationship could also reflect the statistical distribution of hash outputs, indicating improved uniformity and error detection capabilities as d and k increase, where k represents the number of input variables.

As the polynomial degree d increases, the hash function becomes more resilient to algebraic attacks. Higher-degree polynomials introduce greater non-linearity, which is essential for secure hash function design. Non-linear functions exhibit high sensitivity to input variations, making it difficult to predict outputs and strengthening security. Implementing hash functions with higher algebraic degrees further complicates the relationship between inputs and outputs, enhancing resistance to cryptographic attacks.

By leveraging polynomial structures, random coefficient selection, and modular arithmetic, these functions can offer stronger protection against pre-image and collision attacks. The mathematical foundations discussed here highlight the potential of higher-degree polynomials in cryptography, emphasizing the importance of robust design principles in developing secure hash functions.

5. NOETHERIAN RINGS

The basic fundamental concepts related to Noetherian rings related to cryptography are focused in this section.

For a ring R_l we defined $R_l[T_l] := R_l^{N_l}$ as additive clusters, Therefore, the elements in $R_l[T_l]$ are often considered as sequences $(a_v)_{v \in N_l}$, where $a_v \in R_l, \forall v \in N_l$, identify that $f: N_l \rightarrow R_l$. However we would like to bring a new multiplication, typically additionally known as Cauchy multiplication, in $R_l[T_l]$ as

$$(a_v) \cdot (b_v) := (c_v), \text{ where } c_v := \sum_{k=0}^v a_k b_{v-k} \sum_{k+m=v} a_k b_m \quad (5.1)$$

So if we tend to determine an element $a \in R_l$ with the sequence $(a, 0, 0, \dots)$, then the sequence (a_v) , during which approximately, every one of the elements $a_v = 0$.

The ring $R_l[T_l]$ is termed the power series ring in R_l , and therefore the elements are typically, $\sum_{v=0}^n a_v T_l^v$ appropriate the sequences (a_v) . The polynomial ring $R_l[T_l]$ according to any variable over a ring R_l is the sub-set:

$$R_l[T_l] := \left\{ \sum_{v=0}^n a_v T_l^v ; n \in N_l, a_0, a_1, \dots, a_n \in R_l \right\} \quad (5.2)$$

Now introduce the polynomials into a ring R_l as follows:

$$f = \sum_{v=0}^n a_v T_l^v ; n \in N_l, a_0, a_1, \dots, a_n \in R_l \quad (5.3)$$

This means that $\sum_v a_v T_l^v = \sum_v b_v T_l^v$ if and only if $a_v = b_v$ for every $v \in N_l$. Therefore, the sum and multiplication are as follows

$$\left(\sum_v a_v T_l^v \right) \left(\sum_v b_v T_l^v \right) = \sum_v \left(\sum_{k=0}^v a_k b_{v-k} \right) T_l^v \quad (5.4)$$

A polynomial $f \in R_l[T_l]$ is called monic if it is of the form $f = T_l^n + \sum_{v < n} a_v T_l^v$.

6. RESULTS AND DISCUSSION

Encryption Time: The average time taken to encrypt a single patient record.

Decryption Time: The average time taken to decrypt a single patient record.

Error Correction: The percentage of successfully corrected errors in a test dataset.

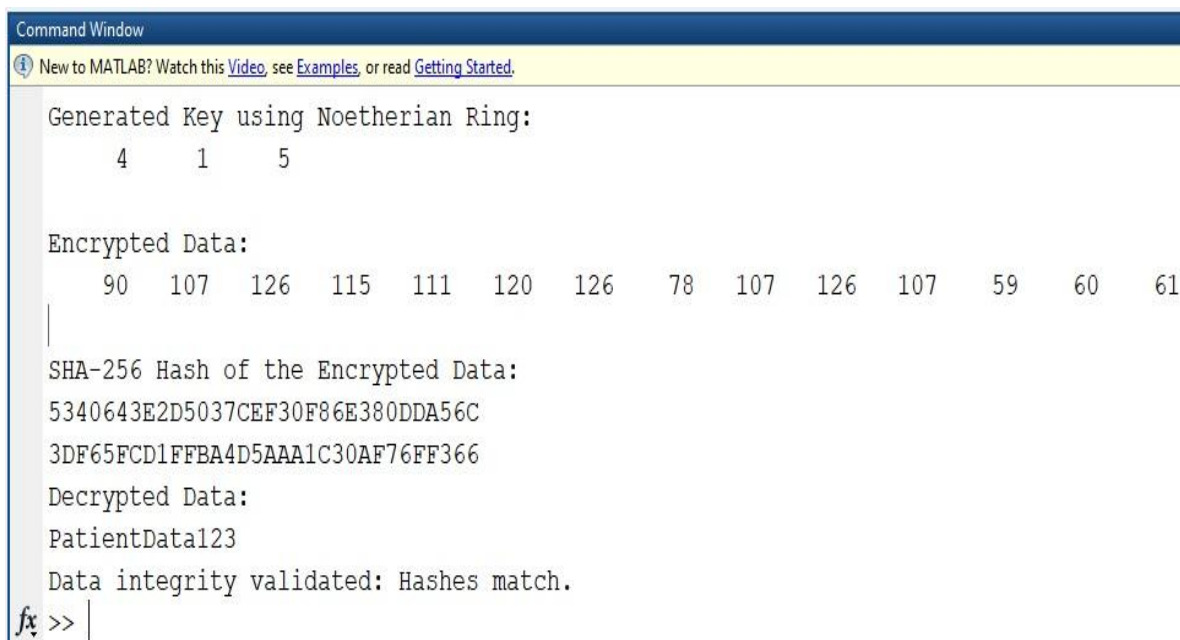
Pseudocode for secured digital healthcare system using Noetherian rings and SHA algorithm


```

Step 1: Start
Step 2: Initialize the ring modulus and polynomial coefficients
Step 3: Generate Key
Step 4: Input (original data as string)
Step 5: Convert original data to ASCII values
Step 6: Encrypt the input data:
Step 7: Compute SHA-256 Hash of Encrypted Data and display Hash
        data in hexadecimal
Step 8: Decrypt data
Step 9: Validate data integrity by recomputed hash and display
        data integrity validated value if Hashes match
Step 10: If not then go to Step 2 and proceed.
Step 11: Visualize Noetherian ring key space of encrypted data
Step 12: End

```

Output:



```

Command Window
New to MATLAB? Watch this Video, see Examples, or read Getting Started.

Generated Key using Noetherian Ring:
      4      1      5

Encrypted Data:
      90     107     126     115     111     120     126     78     107     126     107     59     60     61

SHA-256 Hash of the Encrypted Data:
5340643E2D5037CEF30F86E380DDA56C
3DF65FCD1FFBA4D5AAA1C30AF76FF366

Decrypted Data:
PatientData123

Data integrity validated: Hashes match.
fx >>

```

Figure 3: Noetherian Ring key data for encryption and decryption

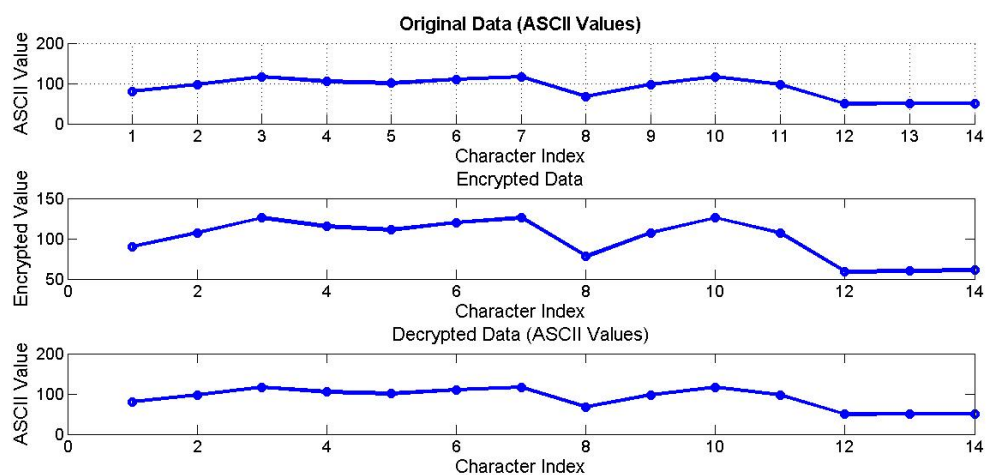


Figure 4: Encryption and decryption process by using Noetherian Ring key

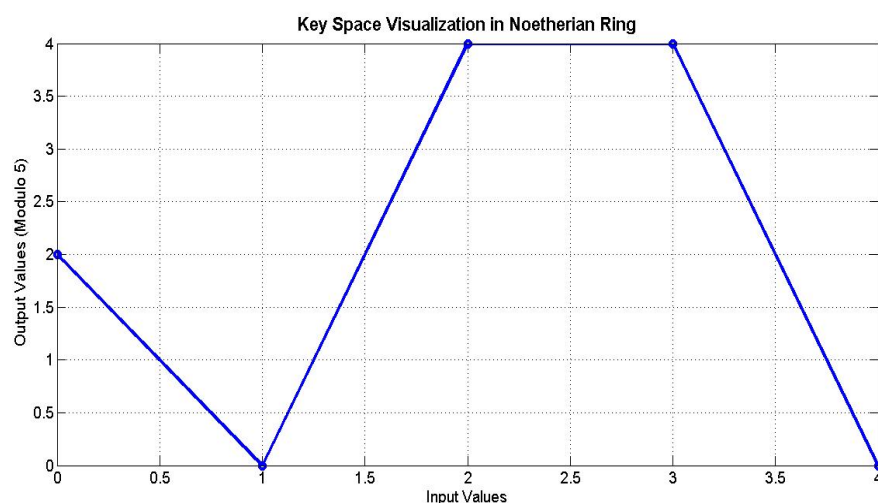


Figure 5: Noetherian ring key space of encrypted data for modulo 5

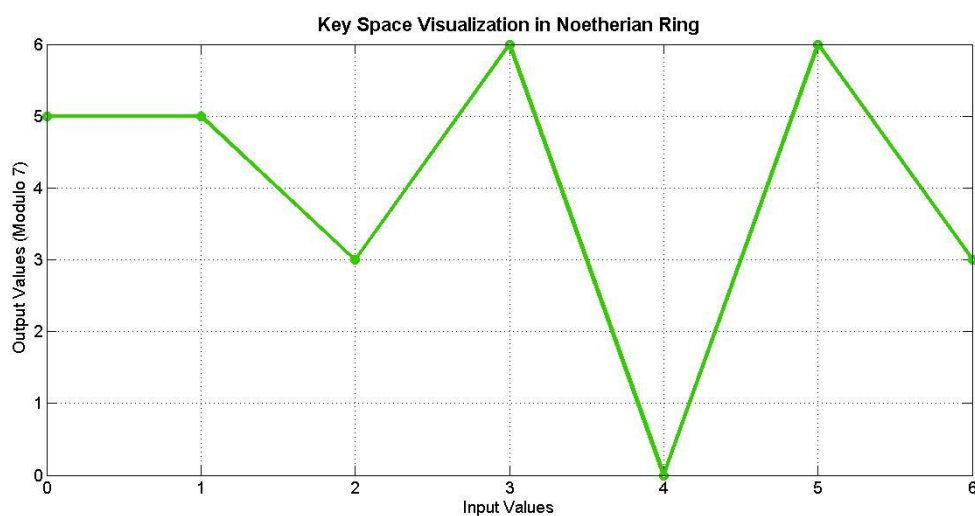


Figure 6: Noetherian ring key space of encrypted data for modulo 7

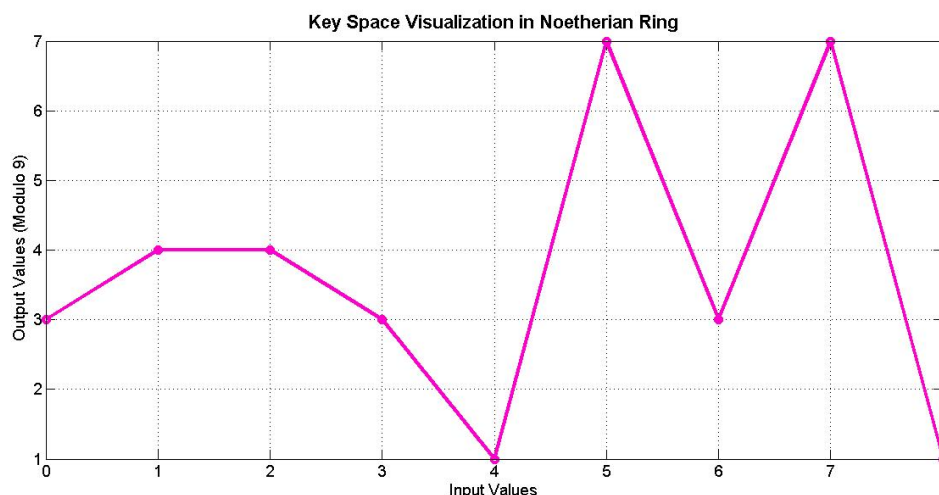


Figure 7: Noetherian ring key space of encrypted data for modulo 9

The MATLAB code includes a visualization of the key space in the Noetherian ring. It plots the polynomial values of the ring modulus, illustrating how the ring's properties are used for key generation. Figure 3 explains Noetherian ring key data for data encryption and decryption and figure 4 shows the encryption and decryption process by using Noetherian Ring key.

This code implements a secure data handling mechanism that combines the mathematical properties of Noetherian rings with the SHA-256 hashing algorithm. Initially we define a ring polynomial and reduces its coefficients modulo to a prime number to create the cryptographic key. The generated key is used for encryption and decryption. Encrypts the input healthcare data by converting characters to their ASCII values by adding the sum of the generated key modulo 256 to these values. SHA-256 algorithm is used to compute a secure hash of the encrypted data. Figures 5 to 7 shows the Noetherian ring key space of encrypted data for modulo 5, modulo 7 and modulo 9. The hash ensures data integrity and cannot be easily reversed. The hash is displayed in hexadecimal format for readability. By reversing the encryption formula, we can decrypt the encrypted data. To get the original ASCII values we can subtracts the sum of the key and takes modulo 256.

To validate the data integrity, we need to recomputes the hash of the encrypted data. Then compares it with the original hash to ensure that the data hasn't been altered. If the hashes match, the data integrity is confirmed. If not, it indicates tampering or corruption. It demonstrates the behavior of the Noetherian ring under modular arithmetic.

7. CONCLUSION

The results demonstrate that the Noetherian ring-based encryption method provides a robust mechanism for securing medical data. The computational efficiency of matrix operations allows for quick encryption and decryption, making it suitable for real-time applications in healthcare. By leveraging the properties of vector spaces and matrix operations, we develop an efficient encryption scheme that enhances data confidentiality and integrity. The mathematical foundations of the Hill cipher revolve around matrix operations, modular arithmetic, and the properties of linear algebra. This structure not only facilitates encryption and decryption but also provides avenues for analyzing the cipher's security and potential vulnerabilities. The Hill cipher showcases how linear algebra can facilitate secure communication through matrix operations and modular arithmetic. Future work will explore optimizing the key generation process and integrating advanced error-correcting codes to further improve data reliability.

REFERENCES

- [1] Shannon, C. E. (1949). *Communication Theory of Secrecy Systems*.
- [2] Abdul Razaq, Louai A. Maghrabi, Musheer Ahmad, Qamar H. Naith. Novel substitution-box generation using group theory for secure medical image encryption in E-healthcare[J]. *AIMS Mathematics*, 2024, 9(3): 6207-6237. doi: 10.3934/math.2024303
- [3] Katz, J., & Lindell, Y. (2014). *Introduction to Modern Cryptography*. CRC Press.
- [4] D. R. Stinson, & J. M. G. Mitchell. (2002). "Cryptography: Theory and Practice." Chapman & Hall/CRC.
- [5] Salvatore, J. (2019). "Secure Medical Data Management: Techniques and Applications." *Journal of Medical Systems*, 43(5), 1-12.
- [6] Inam, S., Kanwal, S., Firdous, R. et al. Blockchain based medical image encryption using Arnold's cat map in a cloud environment. *Sci Rep*14, 5678 (2024). <https://doi.org/10.1038/s41598-024-56364-z>

- [7] Blakley, G. R. (1979). "Safeguarding cryptographic keys." Proceedings of the National Computer Conference.
- [8] Ravi, D., Ramachandran, S., Vignesh, R., Falmari, V. R. & Brindha, M. Privacy preserving transparent supply chain management through hyperledger fabric. *Blockchain Res. Appl.* 3(2), 100072 (2022)
- [9] Kanwal, S., Inam, S., Ali, R. and Cheikhrouhou, O. Lightweight noncommutative key exchange protocol for IoT environments. *Front. Environ. Sci.* 10, 996296 (2022).
- [10] Afzal, I., Parah, S. A., Hurrah, N. N. & Song, O. Y. Secure patient data transmission on resource constrained platform. *Multimed. Tools Appl.* 83, 15001–15026 (2020).
- [11] Enab, M. Image Encryption of Internet of Medical Things Privacy using AES (2022).
- [12] Sowmiya, L., Rajasekaran, A. S., Suganyadevi, S., Sureshkumar, S., Subramaniam, G. & Jaazieliah, R. A secure authenticated message transfer in healthcare application. In 2023 IEEE International Conference on Integrated Circuits and Communication Systems (ICICACS) 1–6 (IEEE, 2023).
- [13] Kaur, N., Mittal, A., Lilhore, U.K. et al. Securing fog computing in healthcare with a zero-trust approach and blockchain. *J Wireless Com Network* 2025, 5 (2025). <https://doi.org/10.1186/s13638-025-02431-6>
- [14] Mondal, A. & Goswami, R. T. Enhanced honeypot cryptographic scheme and privacy preservation for an effective prediction in cloud security. *Microprocess. Microsyst.* 81, 103719 (2020).
- [15] Sri Vigna Hema, V. & Kesavan, R. ECC based secure sharing of healthcare data in the health cloud environment. *Wirel. Pers. Commun.* 108(2), 1021–1035 (2019).
- [16] Almulhim, M., & Zaman, N. (2018, February). Proposing secure and lightweight authentication scheme for IoT based E-health applications. In 2018 20th International Conference on Advanced Communication Technology (ICACT) (pp. 481-487). IEEE.
- [17] Chakraborty, S., Aich, S., & Kim, H. C. (2019, February). A secure healthcare system design framework using blockchain technology. In 2019 21st International Conference on Advanced Communication Technology (ICACT) (pp. 260-264). IEEE. <https://doi.org/10.23919/ICACT.2019.8701983>.
- [18] Fernandez-Aleman, J. L., Senior, I. C., Lozoya, P. A. O., & Toval, A. (2013). Security and privacy in electronic health records: A systematic literature review. *Journal of Biomedical Informatics*, 46(3), 541-562. <https://doi.org/10.1016/j.jbi.2012.12.003>.
- [19] Kumar, P., Lee, S. G., & Lee, H. J. (2012). E-SAP: Efficient-strong authentication protocol for healthcare applications using wireless medical sensor networks. *Sensors*, 12(2), 1625-1647. <https://doi.org/10.3390/s120201625>.
- [20] Ensteih Silvia, & Mohd Tajuddin. (2024). E-Health Privacy and Security through ECC, SHA-256, and Multi-Authority Approaches. *Journal of Information Technology and Cryptography* (e-ISSN: 3048-5290), 1(1), 9–13, <https://doi.org/10.48001/joitc.2023.119-13>.
- [21] Y. Miao, F. Li, X. Li, Z. Liu, J. Ning, H. Li, K.-K.R. Choo, R.H. Deng, Time-controllable keyword search scheme with efficient revocation in mobile e-health cloud. *IEEE Trans. Mob. Comput.* 23(5), 3650–3665 (2023).
- [22] Ye, C., Chen, C. Secure medical image sharing for smart healthcare system based on cellular neural network. *Complex Intell. Syst.* 9, 1653–1670 (2023). <https://doi.org/10.1007/s40747-022-00881-9>
- [23] Xu C, Shang Y, Yang Y, Zou C. An encryption algorithm for multiple medical images based on a novel chaotic system and an odd-even separation strategy. *Sci Rep.* 2025 Jan 22;15(1):2863. doi: 10.1038/s41598-025-86771-9. PMID: 39843646; PMCID: PMC11754641.
- [24] Abirami, R., & Malathy, C. (2025). Secured DICOM medical image transition with optimized chaos method for encryption and customized deep learning model for watermarking. *Automatika*, 66(2), 173–187. <https://doi.org/10.1080/00051144.2025.2460877>.