ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

Encryption Method Including Byte Rotation Method and Finite Field Arithmetic

Manish Shrivastava[1]; Dr. S. S. Shrivastava[2]

[1,2]Institute for Excellence in Higher Education, Bhopal, (M.P.)

Publication Date: 2026/11/20

Abstract: The purpose of this research is to offer a newly developed encryption method including Byte Rotation-Based Mechanism and Finite Field. On different blocks of plaintext, the Byte Rotation-Based Mechanism is applied. It follows a parallel execution approach through multithreading concept of single processor system. Therefore, this encryption approach technique formed an innovative data protection model and generates a dual-layer security mechanism that ensures rapid execution with enhanced data protection.

Keywords: *Encryption, Decryption, Finite-Field Arithmetic and Byte Rotation.*

How to Cite: Manish Shrivastava; Dr. S. S. Shrivastava (2026) Encryption Method Including Byte Rotation Method and Finite Field Arithmetic. *International Journal of Innovative Science and Research Technology*, 11(11), 855-861. https://doi.org/10.38124/ijisrt/26nov696

I. INTRODUCTION

Cryptography refers to the practice of transforming information into an encoded format to ensure secure communication and prevent unauthorized access. A cipher is a mathematical procedure or algorithm employed to convert plaintext into ciphertext and recover the original information through decryption. Cryptographic systems are broadly classified into two major categories: symmetric-key cryptography and asymmetric-key cryptography. Symmetric-key cryptography, traditionally associated with classical cryptographic techniques, uses a shared secret key for both encryption and decryption. This approach involves an encryption algorithm, a secret encryption key, a decryption algorithm, and the corresponding key required to retrieve the original message. In symmetric-key cryptography, the same secret key is employed for both the encryption and decryption of information. In contrast, asymmetric-key cryptography operates using a pair of mathematically related keys, where the public key is utilized to encrypt the data, while the corresponding private key is required to decrypt it. Thus, symmetric encryption relies on the secure sharing of a common secret key between communicating parties, whereas asymmetric encryption separates the encryption and decryption keys.[1]

➢ *Finite Field:*

A field is a fundamental algebraic structure in mathematics that consists of a set of elements equipped with two operations: addition and multiplication. Under addition, the elements of a field form an Abelian group, while the non-zero elements constitute an Abelian group under multiplication. Furthermore, multiplication satisfies the distributive property over addition. A field containing a limited and countable number of elements is referred to as a finite field or Galois field.

Every finite field contains p^nelements, where $p$represents a prime number and $n$is a positive integer. Such a field is commonly referred to as a Galois Field and is denoted by GF (p^n). When $n = 1$, the field GF (p) corresponds to the integers modulo p. For $n > 1$, the elements of GF (p^n) can be represented using polynomials of degree at most $n - 1$, with coefficients belonging to GF (p), under suitable irreducible-polynomial arithmetic. Finite fields have significant applications in various areas of computer science and engineering, particularly in coding theory and cryptography. For instance, elliptic curve cryptographic systems utilize elliptic curves defined over finite fields.

➢ *Byte Rotation-Based Encryption Technique*

The Byte Rotation-Based Encryption Technique is a symmetric-key block cipher approach that processes plaintext in multiple fixed-size blocks. Each block consists of 16 or more bytes, depending on the selected block configuration. Initially, the input plaintext is divided into individual blocks, and each block is organized into a two-dimensional array structure. Subsequently, byte-level rotational operations are performed across the rows and columns of the array to transform the data into an encrypted form. These rotation operations modify the arrangement of bytes, contributing to the overall encryption process.

II. RELATED WORK

Several researchers, including Sharma [2], Lakshami [3], Sharma [4], and Bhati [5], have made significant

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

contributions to establishing this work some techniques employed for encrypting and decrypting messages including Byte-Level Rotation Method and Finite Fields.

Observing usefulness and importance of employing various encryption and decryption techniques for secure message transmission, in accordance with the aforementioned framework of above authors, we offer to establish a new technique for encryption and decryption for processing a message using the Byte Rotation Technique, finite-field operations, and logical operators to ensure secure data transformation using XOR.

III. METHODOLOGY

In this study, a combination of two techniques was utilized to develop the proposed framework for encrypting and decrypting of a message, the first approach utilizes elements from finite fields and second one is 'Byte Rotation Technique'. On distinct sections of the plaintext data, the Byte Rotation Technique is incorporated into the framework and executed using a parallel processing approach.

In the proposed encryption scheme, the encryption process consists of two successive stages. Initially, the Byte Rotation Technique is applied to the input plaintext to generate an intermediate cipher. Subsequently, elements of finite fields are employed to transform this intermediate output into the final ciphertext. The decryption process follows the reverse sequence of operations. First, finite field elements are utilized to recover the intermediate plaintext, followed by the application of the Byte Rotation Technique to reconstruct the original plaintext.

The proposed algorithm incorporates two distinct cryptographic keys to perform the encryption and decryption operations. The first key is represented by a randomly generated matrix, whereas the second key is constructed using finite field elements. During both encryption and decryption, finite field elements are represented and manipulated in binary and polynomial forms to facilitate the required mathematical operations.

A. Algorithm:

➢ *Encryption:*

Following are the steps of encryption procedure:

- Partition the plaintext into fixed-size units, each consisting of n bytes. Represent these units in the form of square matrix P.
- Rotate the first four columns of matrix P vertically as follows: shift the four bytes in the first column, three bytes in the second column, two bytes in the third column, and one byte in the fourth column. Leave the fifth column unchanged. Denote the resulting matrix as P_{vr}.
- Apply horizontal rotation to the first four rows of matrix Pvr by shifting four bytes in the first row, three bytes in the second row, two bytes in the third row, and one byte in the fourth row, while keeping the fifth row unchanged. Denote the resulting matrix as P_{hr}.
- Obtained the transpose for matrix P_{hr} and represent it by P_{hr}^T or M.
- To compute the intermediate cipher using ASCII code, represent each element of matrix Mrow-wise by its equivalent alphabet or character.
- Convert each element of matrix Minto its corresponding 7-bit binary representation and store the resulting values in a new matrix, denoted as M_1
- Randomly generate an $n \times n$matrix A to serve as the first secret key. Convert each element of matrix A into its corresponding 7-bit binary representation and denote the resulting matrix as A_1.
- Randomly select the rows or columns of matrix $A_1$and perform the XOR operation on the corresponding rows of matrix M_1. Store the resulting matrix and denote it as N_{XOR} All entries of N_{XOR} alter into the elements of GF(2^m) and find out another matrix (say N'_α).
- Generate another matrix K', in which each element is assigned a value of 1 if the corresponding power of α is greater than $2m-1$, and 0 otherwise. Then, multiply each element of $N'_\alpha$by $\alpha^m$and transmit the resulting matrix to the receiver. Reduces the powers of the entries of N'_α to mod ($2^m - 1$) and get another matrix (say N''_α).
- Convert each element of N''_αinto its binary representation. Next, transform the binary values into their corresponding numerical equivalents and subsequently encode them in ASCII hexadecimal format to obtain the final ciphertext.
- Reduce the exponents of each element in N'_αmodulo $(2^m - 1)$to obtain a new matrix, denoted by N''_α.
- Represent each element of N''_αin binary form, then convert the resulting binary sequences into their equivalent numerical values. Subsequently, encode these values in ASCII hexadecimal format to generate the final ciphertext.

➢ *Decryption*

- Initially, convert the received ciphertext into its corresponding ASCII decimal values.

Then, represent each decimal value as an m-bit binary sequence and map these binary sequences to their corresponding elements in the finite field $GF(2^m)$. Finally, arrange the obtained field elements into a square matrix, denoted by N''_α.

- Multiply the elements of $N''_\alpha$by $\alpha^{(2^m-1)}$at the positions corresponding to entries with a value of 1 in the key matrix K. The resulting matrix is then obtained and denoted by N'_α.
- Multiply every element of $N'_\alpha$by α^{-m}, then convert the resulting elements into their corresponding m-bit binary representations. Arrange these binary values into a new matrix, denoted by N.
- Identify the rows or columns of matrix Athat were randomly selected by the sender and perform the corresponding XOR operations with each row of matrix N. Arrange the resulting values into a new matrix, denoted by M_1.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

- To obtain the intermediate plaintext, convert each element of matrix M_1, in row-wise order, into its corresponding ASCII character. Next, transform these ASCII characters into their equivalent ASCII decimal values and arrange the resulting values row-wise in an $m \times m$square matrix, denoted by M.
- Compute the transpose of matrix M (say M^T).
- Perform a horizontal rotation on the first four rows of the transposed matrix M^T. Rotate the first row by four bytes, the second row by three bytes, the third row by two bytes, and the fourth row by one byte, while keeping the fifth row unchanged. Denote the resulting matrix as M_{hr}^T.
- Perform a vertical rotation on the first four columns of matrix M_{hr}^T. Rotate the first column by four bytes, the second column by three bytes, the third column by two bytes, and the fourth column by one byte, while leaving the fifth column unchanged. Denote the resulting matrix by P, which represents the recovered plaintext block of the required size.
- Obtained original plain text after combining such type of blocks.

B. Illustration:

➢ *Encryption Steps:*

- Begin by taking the input plaintext as the initial message for the encryption process as follows:

➢ *Encryption Decryption Model*

- Using ASCII code, represent each alphabet/character of plain text into their numeric value and arrange them in the form of a matrix of order 5 × 5 of block of 25 bytes, we get

$$\begin{bmatrix} 69 & 78 & 67 & 82 & 89 \\ 80 & 84 & 73 & 79 & 78 \\ 68 & 69 & 67 & 82 & 89 \\ 80 & 84 & 73 & 79 & 78 \\ 77 & 79 & 68 & 69 & 76 \end{bmatrix} = P \text{ (say)}$$

- Now vertically rotate the bytes of first four columns of P (for which sender and receiver are agreed) as given below:

$$\begin{bmatrix} 77 & 84 & 67 & 79 & 89 \\ 69 & 79 & 73 & 82 & 78 \\ 80 & 78 & 68 & 79 & 89 \\ 68 & 84 & 67 & 69 & 78 \\ 80 & 69 & 73 & 82 & 76 \end{bmatrix} = P_{vr} \text{ (say)}$$

- Similarly, horizontally rotate the bytes of first four rows of P_{vr} (for which sender and receiver are agreed) as given below:

$$\begin{bmatrix} 89 & 77 & 84 & 67 & 79 \\ 82 & 78 & 69 & 79 & 73 \\ 68 & 79 & 89 & 80 & 78 \\ 84 & 67 & 69 & 78 & 68 \\ 80 & 69 & 73 & 82 & 76 \end{bmatrix} = P_{hr} \text{ (say)}$$

- Compute the transpose of P_{hr}, we get as follows:

$$P_{hr}^T = \begin{bmatrix} 89 & 82 & 68 & 84 & 80 \\ 77 & 78 & 79 & 67 & 69 \\ 84 & 69 & 89 & 69 & 73 \\ 67 & 79 & 80 & 78 & 82 \\ 79 & 73 & 78 & 68 & 76 \end{bmatrix} = M \text{ (say)}$$

- To get intermediate cipher text, using ASCII-code and represents every element (row wise) of matrix M into their equivalent alphabet/character as follows:

YRDTPMNOCETEYEICOPNROINDL

- Alters the elements of matrix M into 7-bit binary string as follows:

$$\begin{bmatrix} 1011001 & 1010010 & 1000100 & 1010100 & 1010000 \\ 1001101 & 1001110 & 1001111 & 1000011 & 1000101 \\ 1010100 & 1000101 & 1011001 & 1000101 & 1001001 \\ 1000011 & 1001111 & 1010000 & 1001110 & 1010010 \\ 1001111 & 1001001 & 1001110 & 1000100 & 1001100 \end{bmatrix} = M_1 \text{ (say)}$$

- Now randomly chosen a secret key in the form of 5 × 5 matrix as follows:

$$A = \begin{bmatrix} 85 & 68 & 90 & 83 & 68 \\ 80 & 87 & 87 & 65 & 67 \\ 79 & 69 & 90 & 86 & 85 \\ 69 & 76 & 88 & 76 & 74 \\ 79 & 69 & 77 & 67 & 79 \end{bmatrix}$$

Alters the elements of matrix A into 7-bit binary string:

$$A_1 = \begin{bmatrix} 1010101 & 1000100 & 1011010 & 1010011 & 1000100 \\ 1010000 & 1010111 & 1010111 & 1000001 & 1000011 \\ 1001111 & 1000101 & 1011010 & 1010110 & 1010101 \\ 1000101 & 1001100 & 1011000 & 1001100 & 1001010 \\ 1001111 & 1000101 & 1001101 & 1000011 & 1001111 \end{bmatrix}$$

- To perform logical, XOR operation with each row of matrix M_1, randomly choose the rows/columns R_1, C_2, R_3, C_4, R_5 from the matrix A_1.

✓ Chose elements of R_1 of A_1 and apply logical operator XOR with the elements of first row of matrix M_1 as follows:

10101011000100101101010100111000100

XOR

10110011010010100010010101001010000

We get first row of another matrix N_{XOR} (say), as given below:

00011000010110001111000001110010100

✓ Chose elements of C_2 of A_1 and apply logical operator XOR with the elements of second row of matrix M_1 as follows:

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

10001001010111100010110011001000101

XOR

10011011001110100111110000111000101

We get second row of matrix N_{XOR} (say), as given below:

00010010011001000101000011110000000

✓ Chose elements of R_3 of A_1 and apply logical operator XOR with the elements of third row of matrix M_1 as follows:

10011111000101101101010101101010101

XOR

10101001000101101100110001011001001

We get third row of matrix N_{XOR} (say), as given below:

00110110000000000001100100110011100

✓ Chose elements of C_4 of A_1 and apply logical operator XOR with the elements of fourth row of matrix M_1 as follows:

10100111000001101011010011001000011

XOR

10000111001111101000010011101010010

We get fourth row of matrix N_{XOR} (say), as given below:

00100000001110000011000000100010001

✓ Chose elements of R_5 of A_1 and apply logical operator XOR with the elements of fifth row of matrix M_1 as follows:

10011111000101100110110000111001111

XOR

10011111001001100111010001001001100

We get fifth row of matrix N_{XOR} (say), as given below:

00000000001100000001100001110000011

Therefore, matrix N_{XOR} will be shows as follows:

$$N_{XOR}=\begin{bmatrix}0001100 & 0010110 & 0011110 & 0000111 & 0010100\\ 0001001 & 0011001 & 0001010 & 0001111 & 0000000\\ 0011011 & 0000000 & 0000011 & 0010011 & 0011100\\ 0010000 & 0001110 & 0000110 & 0000010 & 0010001\\ 1001111 & 1001001 & 1001110 & 1000100 & 1001100\end{bmatrix}$$

- Now alters each entry of N_{XOR} into the elements of GF (2^7) in their basis form s.t. ($\alpha^7+\alpha^2+1)=0$, we get as follows:

$$N_\alpha=\begin{bmatrix}\alpha^{9} & \alpha^{32} & \alpha^{22} & \alpha^{56} & \alpha^{16}\\ \alpha^{63} & \alpha^{108} & \alpha^{15} & \alpha^{21} & \alpha^{128}\\ \alpha^{70} & \alpha^{128} & \alpha^{7} & \alpha^{67} & \alpha^{58}\\ \alpha^{4} & \alpha^{57} & \alpha^{8} & \alpha^{1} & \alpha^{28}\\ \alpha^{37} & \alpha^{27} & \alpha^{46} & \alpha^{30} & \alpha^{69}\end{bmatrix}$$

Multiply each element of N_α by α^7, we get

$$\begin{bmatrix}\alpha^{16} & \alpha^{39} & \alpha^{29} & \alpha^{63} & \alpha^{23}\\ \alpha^{70} & \alpha^{115} & \alpha^{22} & \alpha^{28} & \alpha^{135}\\ \alpha^{77} & \alpha^{135} & \alpha^{14} & \alpha^{74} & \alpha^{65}\\ \alpha^{11} & \alpha^{64} & \alpha^{15} & \alpha^{8} & \alpha^{35}\\ \alpha^{44} & \alpha^{34} & \alpha^{53} & \alpha^{37} & \alpha^{76}\end{bmatrix}=N'_\alpha \text{ (say)}$$

Now choose the key (second key) matrix

$$K=\begin{bmatrix}0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 1\\ 0 & 1 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\\ 0 & 0 & 0 & 0 & 0\end{bmatrix}$$

s.t. the entry in the key matrix is taken 0 if power of α in $N_{\alpha'}$ is less than 127, otherwise 1.

- Now reduce $N_{\alpha'}$ to mod 127, as follows:

$$\begin{bmatrix}\alpha^{16} & \alpha^{39} & \alpha^{29} & \alpha^{63} & \alpha^{23}\\ \alpha^{70} & \alpha^{115} & \alpha^{22} & \alpha^{28} & \alpha^{8}\\ \alpha^{77} & \alpha^{8} & \alpha^{14} & \alpha^{74} & \alpha^{65}\\ \alpha^{11} & \alpha^{64} & \alpha^{15} & \alpha^{8} & \alpha^{35}\\ \alpha^{44} & \alpha^{34} & \alpha^{53} & \alpha^{37} & \alpha^{76}\end{bmatrix}=N''_\alpha \text{ (say)}$$

- Convert each entry of matrixN''_α into the corresponding binary form, we get as follows:

$$\begin{bmatrix}0010100 & 0111010 & 0100010 & 0001001 & 0111100\\ 0011011 & 0101011 & 0011110 & 0010001 & 0000110\\ 0101101 & 0000110 & 0000101 & 0110101 & 0100100\\ 0110000 & 0010010 & 0001010 & 0000110 & 0110011\\ 1010010 & 1011000 & 1010001 & 1001111 & 1010111\end{bmatrix}=S \text{ (say)}$$

Alter each entry of matrix S into equivalent numerical value, we get

$$\begin{bmatrix}20 & 58 & 34 & 9 & 60\\ 27 & 43 & 30 & 17 & 6\\ 45 & 6 & 5 & 53 & 36\\ 48 & 18 & 10 & 6 & 51\\ 82 & 88 & 81 & 79 & 87\end{bmatrix}=S' \text{ (say)}$$

- Convert each entry of matrix S' into equivalent ASCII-HEX code, we get the final cipher text as given below:

143A22093C1B2B1E11062D0605352430120A06335258514F57

This Cipher text will be send to the receiver via public channel.

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

➢ *Decryption Steps:*

- Take Cipher text message

143A22093C1B2B1E11062D0605352430120A06335258514F57

Using ASCII-Code, alters the message into equivalent numerical values, we get

$$\begin{bmatrix}20 & 58 & 34 & 9 & 60\\ 27 & 43 & 30 & 17 & 6\\ 45 & 6 & 5 & 53 & 36\\ 48 & 18 & 10 & 6 & 51\\ 82 & 88 & 81 & 79 & 87\end{bmatrix} = S' \text{ (say)}$$

- Now change these numerical values into equivalent binary form of 7-bit string, as follows:

$$\begin{bmatrix}0010100 & 0111010 & 0100010 & 0001001 & 0111100\\ 0011011 & 0101011 & 0011110 & 0010001 & 0000110\\ 0101101 & 0000110 & 0000101 & 0110101 & 0100100\\ 0110000 & 0010010 & 0001010 & 0000110 & 0110011\\ 1010010 & 1011000 & 1010001 & 1001111 & 1010111\end{bmatrix} = S \text{ (say)}$$

Again convert each elements of matrix S into the elements of GF(2^7), we get

$$\begin{bmatrix}\alpha^{16} & \alpha^{39} & \alpha^{29} & \alpha^{63} & \alpha^{23}\\ \alpha^{70} & \alpha^{115} & \alpha^{22} & \alpha^{28} & \alpha^{8}\\ \alpha^{77} & \alpha^{8} & \alpha^{14} & \alpha^{74} & \alpha^{65}\\ \alpha^{11} & \alpha^{64} & \alpha^{15} & \alpha^{8} & \alpha^{35}\\ \alpha^{44} & \alpha^{34} & \alpha^{53} & \alpha^{37} & \alpha^{76}\end{bmatrix} = N''_{\alpha} \text{ (say)}$$

- Multiply those entries of N''_{α} by α^{127} that represents 1 in the equivalent key matrix K, we get another matrix as follows:

$$\begin{bmatrix}\alpha^{16} & \alpha^{39} & \alpha^{29} & \alpha^{63} & \alpha^{23}\\ \alpha^{70} & \alpha^{115} & \alpha^{22} & \alpha^{28} & \alpha^{135}\\ \alpha^{77} & \alpha^{135} & \alpha^{14} & \alpha^{74} & \alpha^{65}\\ \alpha^{11} & \alpha^{64} & \alpha^{15} & \alpha^{8} & \alpha^{35}\\ \alpha^{44} & \alpha^{34} & \alpha^{53} & \alpha^{37} & \alpha^{76}\end{bmatrix} = N'_{\alpha} \text{ (say)}$$

Now multiply matrix $N_{\alpha'}$ by α^{-7}, we get

$$\begin{bmatrix}\alpha^{9} & \alpha^{32} & \alpha^{22} & \alpha^{56} & \alpha^{16}\\ \alpha^{63} & \alpha^{108} & \alpha^{15} & \alpha^{21} & \alpha^{128}\\ \alpha^{70} & \alpha^{128} & \alpha^{7} & \alpha^{67} & \alpha^{58}\\ \alpha^{4} & \alpha^{57} & \alpha^{8} & \alpha^{1} & \alpha^{28}\\ \alpha^{37} & \alpha^{27} & \alpha^{46} & \alpha^{30} & \alpha^{69}\end{bmatrix} = N_{\alpha} \text{ (say)}$$

- The binary representation of N_{α} is as follows:

$$\begin{bmatrix}0001100 & 0010110 & 0011110 & 0000111 & 0010100\\ 0001001 & 0011001 & 0001010 & 0001111 & 0000000\\ 0011011 & 0000000 & 0000011 & 0010011 & 0011100\\ 0010000 & 0001110 & 0000110 & 0000010 & 0010001\\ 1001111 & 1001001 & 1001110 & 1000100 & 1001100\end{bmatrix} = N \text{ (say)}$$

- Now identifies the rows/columns R_1, C_2, R_3, C_4, R_5 of the matrix B, which is arbitrarily chosen by the sender, as follows:

$$B = \begin{bmatrix}85 & 68 & 90 & 83 & 68\\ 80 & 87 & 87 & 65 & 67\\ 79 & 69 & 90 & 86 & 85\\ 69 & 76 & 88 & 76 & 74\\ 79 & 69 & 77 & 67 & 79\end{bmatrix}$$

Convert each entry of matrix B into equivalent 7-bit binary form, we get

$$\Rightarrow \begin{bmatrix}1010101 & 1000100 & 1011010 & 1010011 & 1000100\\ 1010000 & 1010111 & 1010111 & 1000001 & 1000011\\ 1001111 & 1000101 & 1011010 & 1010110 & 1010101\\ 1000101 & 1001100 & 1011000 & 1001100 & 1001010\\ 1001111 & 1000101 & 1001101 & 1000011 & 1001111\end{bmatrix} = B_1 \text{ (say)}$$

✓ Take the elements of row R_1 of matrix B_1 and compute logical operator XOR with first row of matrix N, we get

10101011000100101101010100111000100

XOR

00011000010110001111000001110010100

We get 1st row of another matrix M_1 (say), as follows:

10110011010010100010010101001010000

✓ Take the elements of column C_2 of matrix B_1 and compute logical operator XOR with 2nd row of matrix N, we get

10001001010111100010110011001000101

XOR

00010010011001000101000011110000000

We get 2nd row of matrix M_1 (say), as follows:

10011011001110100111110000111000101

✓ Take elements of row R_3 of matrix B_1 and compute logical operator XOR with 3rd row of matrix N, we get

10011111000101101101010101101010101

XOR

00110110000000000001100100110011100

We get the 3rd row of matrix M_1 (say), as follows:

10101001000101101100110001011001001

✓ Take elements of column C_4 of matrix B_1 and compute logical operator XOR with 4th row of matrix N, we get

10100111000001101011010011001000011

XOR

00100000001110000011000000100010001

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

We get the 4th row of matrix M_1 (say), as follows:

10000111001111101000010011101010010

✓ Take elements of row R_5 of matrix B_1 and compute logical operator XOR with 5th row of matrix N, we get

10011111000101100110110000111001111

XOR

00000000001100000001100001110000011

We get the 5th row of matrix M_1 (say), as follows:

10011111001001100111010001001001100

Thus, the matrix M_1 is as follows:

$$M_1 = \begin{bmatrix} 1011001 & 1010010 & 1000100 & 1010100 & 1010000 \\ 1001101 & 1001110 & 1001111 & 1000011 & 1000101 \\ 1010100 & 1000101 & 1011001 & 1000101 & 1001001 \\ 1000011 & 1001111 & 1010000 & 1001110 & 1010010 \\ 1001111 & 1001001 & 1001110 & 1000100 & 1001100 \end{bmatrix}$$

- Now represent each element (row wise) of matrix M_1 into their equivalent ASCII-code, we get intermediate plain text as follows:

YRDTPMNOCETEYEICOPNROINDL

- Now above ASCII character of intermediate plain text convert into their equivalent ASCII-DEC form. Arrange them row-wise in a square matrix of order 5 in a block of 25 bytes, as follows:

$$\begin{bmatrix} 89 & 82 & 68 & 84 & 80 \\ 77 & 78 & 79 & 67 & 69 \\ 84 & 69 & 89 & 69 & 73 \\ 67 & 79 & 80 & 78 & 82 \\ 79 & 73 & 78 & 68 & 76 \end{bmatrix} = M(say)$$

- Compute the transpose of M, we get

$$\begin{bmatrix} 89 & 77 & 84 & 67 & 79 \\ 82 & 78 & 69 & 79 & 73 \\ 68 & 79 & 89 & 80 & 78 \\ 84 & 67 & 69 & 78 & 68 \\ 80 & 69 & 73 & 82 & 76 \end{bmatrix} = M^T \text{ (say)}$$

- Now rotated horizontally the bytes of first four rows of M^T (for which sender and receiver are agreed) as follows:

$$\begin{bmatrix} 77 & 84 & 67 & 79 & 89 \\ 69 & 79 & 73 & 82 & 78 \\ 80 & 78 & 68 & 79 & 89 \\ 68 & 84 & 67 & 69 & 78 \\ 80 & 69 & 73 & 82 & 76 \end{bmatrix} = M_{hr}^T \text{ (say)}$$

- Now rotated vertically bytes of first four columns of M_{hr}^T (for which sender and receiver are agreed) as follows:

$$\begin{bmatrix} 69 & 78 & 67 & 82 & 89 \\ 80 & 84 & 73 & 79 & 78 \\ 68 & 69 & 67 & 82 & 89 \\ 80 & 84 & 73 & 79 & 78 \\ 77 & 79 & 68 & 69 & 76 \end{bmatrix} = P \text{ (say)}$$

- Convert each element of matrix P row-wise into their equivalent alphabet/character by using ASCII code, we get original plain text as follows:

➢ *Encryption Decryption Model*

IV. RESULT AND DISCUSSIONS

No single encryption algorithm can address all security requirements or provide complete protection against potential cryptographic attacks. Therefore, researchers continue to explore and develop improved encryption and decryption techniques to overcome the limitations of existing methods. In this paper, we propose a new encryption algorithm that combines finite field operations with a Byte Rotation Technique. The proposed approach consists of two main stages. In the first stage, the plaintext is divided into multiple blocks of a predefined size, consisting of 16 or more bytes. A byte rotation operation is then applied to each block to generate an intermediate ciphertext. In the second stage, finite field arithmetic and XOR operations are employed to further transform the intermediate ciphertext. The combination of these two techniques introduces an additional layer of security through a dual-stage encryption mechanism. Consequently, the proposed block cipher aims to enhance the complexity of cryptanalysis and improve resistance against ciphertext-only attacks.

V. CONCLUSION

An algorithm has several advantages and disadvantages. The proposed algorithm combines finite field operations with the Byte Rotation Technique, providing an effective approach to strengthening the encryption process and enhancing data security. The use of finite field elements in the proposed approach supports secure symmetric key generation, thereby enhancing the confidentiality and protection of messages during encryption and decryption; consequently, for cipher text attack and plaintext attack, the proposed cryptosystem gives better protection. Therefore, we proposed a new encryption algorithm that provides sufficient security system.

REFERENCES

[1.] Stallings W.: Cryptography and Network Security: Principles and Practices, Prentice Hall, 1999.

[2.] Sharma P. L., Rehan M.: On Security of Hill Cipher using Finite Fields, International Journal of Computer Applications, Volume 71, No.4, 2013, pp. 30-33.

[3.] Lakshami, G. N., Kumar, B. R., Suneetha, Ch., and Chandra Shekhar: A Cryptographic Scheme of Finite Fields Using Logical Operators, International

ISSN No:-2456-2165 https://doi.org/10.38124/ijisrt/26nov696

Journal of Computer Applications, 31(4), 2011, pp.1- 4.

[4.] Sharma P. L. and Sharma S.: An Application of Finite Field in Hill Cipher, Int. J. Tech., 4(1), 2014, pp. 248-251.

[5.] Bhati Sunita, Bhati Anita, Sharma S. K.: A New Approach towards Encryption Schemes: Byte – Rotation Encryption Algorithm, Proceedings of the World Congress on Engineering and Computer Science, Vol II WCECS, 2012, pp. 1-4.