Journal of Theoretical and Applied Information Technology
15th March 2019. Vol.97. No 5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

MODIFYING DES ALGORITHM BY USING DIAGONAL MATRIX BASED ON IRREDUCIBLE POLYNOMIAL

[1]SAHAB DHEYAA MOHAMMED, [2]ABDUL MONEM SALEH RAHMA,

1**MSC**,University of Information Technology and Communications, Baghdad, Iraq
2 **PROF. Dr.,** University of Technology, "Department of Computer Science, Baghdad, Iraq

E-mail : [1]sahab7dia@yahoo.com, [2] Monem.rahma@yahoo.com"

ABSTRACT

Risks of computer offenses and requirements for information confidentiality have led to increasing attention on high-security cryptosystems. Conventional encryption methods cannot provide enough security when executed on computer systems. Therefore, modern technology uses the principles of traditional encryption methods and mathematical principles applicable on computers. Data Encryption Standard (DES) must have more robust security than other cryptosystems. However, the process time necessary for cryptanalysis is less than usual. Moreover, as hardware techniques have quickly advanced, the DES may be attacked by several types of cryptanalysis using a parallel process. This study proposes changes in the operation of DES to ensure high security. Such changes include performing matrix multiplication operation instead of Exclusive OR (XOR) operation. Moreover, four keys are used for each round, two of which are derived from the main key and the remaining two are internally generated. The four keys are used in a special sequence with round numbers. The main key is generated from a random string of 64 bytes. Then, the key is expanded and distributed over 16 keys.

Keywords: *Data Encryption Standard (DES), Irreducible Polynomial, Diagonal Matrix, Polynomial*

1. INTRODUCTION

At present, any secure communication circumference cannot be perfect without cryptographic techniques. Cryptology can equip a high level of security to any sensitive data that needs to be preserved, such as stored information on hard disks..

Cryptology involves encryption and decryption. Original information is indicated as "plain text" and encryption information as "cipher text." To transform plain text to cipher text, an algorithm must implement a secret key to guarantee security. Two kinds of keys are available: symmetric and asymmetric [1]. In cryptography, a block cipher works on a fixed-length combination of bits. For instance, a block cipher accepts a 128-bit block of plain text as input when encrypting and outputs a similar 128-bit block of cipher text. When decrypting, the algorithm gets a 128-bit block of cipher text with the secret key and outputs a 128-Bit block of plain text. Stream ciphers can be compared with block ciphers. A stream ciphers runs on individual digits one at a time. However, the difference between them is unclear. A block cipher works effectively as a stream cipher when used in certain modes of processes, as shown in Fig. 1 **[2]**.

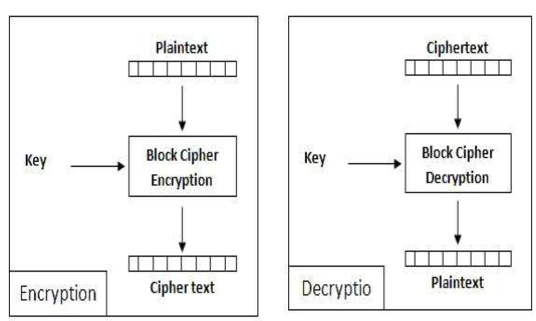

Figure1: Block cipher acts as a stream cipher ***[2]***

Journal of Theoretical and Applied Information Technology
15th March 2019. Vol.97. No 5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Data encryption standard (DES) is an advanced and effective block cipher design. DES was selected as a formal "Federal Information Processing Standard for the United States in 1976" and has thereafter enjoyed popular use worldwide. The algorithm was initially debatable because of its categorized styling elements, relatively small key length, and issues on a "National Security Agency" backdoor. Thus, DES was exposed under a heavy academic scanning and motivated on the modern conception of block ciphers and their cryptanalysis. DES is now considered an intimidation for several applications because its 56-bit key size is deemed small. In addition, DES keys can be cracked in fewer 24 hours. Several analytical consequences explain the theoretical weaknesses of ciphers. However, such consequences are infeasible to set in practice. Thus, the algorithm should be virtually secured in the form of Triple DES, considering that theoretical attacks may still happen. At present, the ciphers have been replaced by advanced encryption standard (AES)[2].

Traditional encryption is identified as single-key or symmetric encryption. For instance The Hill cipher algorithm of polynomial form in Galois field (GF) (2^8) is a symmetric key algorithms that serves a basis for data encryption. All symmetric and public keys algorithms include mathematical processes on integers. For suitability and efficiency, GF precisely fits, where a number of bits without losing bit types. The high complexity in these algorithms serves as an official verification [3].

In this study, a modified approach of DES is developed through the using a numbers of polynomial form in in GF (2^8) on plain text and utilizes the operations of matrix multiplication rather than using the XOR operation that is executed directly.

2. RELATED WORK

In [4], the security of S-DES algorithm is improved, and the transposition and shift row techniques are added before the S-DES algorithm performs its process. A developed S-DES algorithm can improve security, which is important in the communication and scope of the Internet. If transposition and shift row operations are used before the main S-DES algorithm, then an intruder first breaks the main S-DES algorithm and then transposition and shift row techniques are utilized. Therefore, security is approximately dual and contrasted with a simple S-DES algorithm. In [5], the software emulation result proves that the implication of the odd–even substitution to DES provides a more confusion technique to DES. The substitution also provides suitable security while providing firmness and rapidly treating encryption and decryption processes.

In [6], security is developed by modifying the standard keys and algorithmic steps of the DES algorithm. Key generation system creates two keys—one is simple and the other one is encrypted. The first round only uses simple key1, whereas other rounds use encrypted key2. In round 16, simple key1 is used again, and secured cipher text is gained. Thus, vulnerability increases and DES encryption develops. Furthermore, differential cryptanalysis cannot be executed on cipher text.

3. DES

DES algorithm was elaborated by IBM in the early 1970s. In this technique, plain text is agreeable, and the key order used for encryption and decryption defines the kind of cipher. DES is a symmetric, 64-bit block cipher and uses the same key for decryption and encryption. The two major components of the DES-based system are key and algorithm. DES algorithm is a complicated reactive process that involves permutations, substitutions, and mathematical operations. Figure 2 [6] illustrates that DES has constant algorithms and is a public information itself.

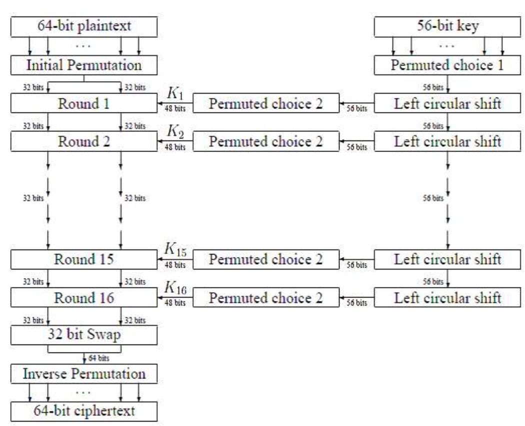

Figure 2. Working structure of DES [6]

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

3.1 Characterization of DES

DES is a kind of repeated cipher called a Feistel cipher. In this cipher, every entry is split into two parts of similar length. The round function has the following:

$$g(L^{i-1}, R^{i-1}, K^{i}) = (L^{i}, R^{i}),$$

Where

$$L^{i} = R^{i-1}$$
$$R^{i} = L^{i-1} \text{ XOR } f(R^{i-1}, K^{i}).$$

DES is a component of 16-round Feistel cipher that is 64 bit blocks long. It encrypts a plain-text bit-string X (of length 64 bit) using a 56-bit key to obtain a cipher-text bit-string (of length 64). Before to the last round of encryption, an initial permutation (IP) is performed to the plain text. We denote the following.

$$IP(X) = L^{0}\ R^{0}$$

When 16 rounds of encryption are finished, the inverse permutation is performed to the bit string, this lead to cipher-text y:

$$y = IP^{-1}\ (R^{16}\ L^{16})$$

Note ***y*** that is swapped before is applied.

IP application has no cryptographic indication and is often ignored when DES security is discussed. Figure 3 displays one round of DES encryption [7].

$$f = \{0,1\}^{32} \times \{0,1\}^{48} \longrightarrow \{0,1\}^{32}.$$

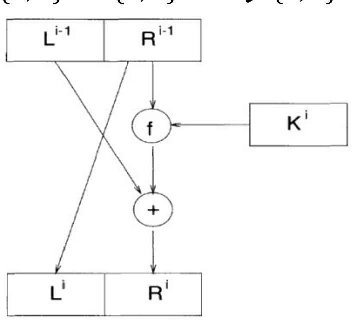

Figure 3 One round of DES [7]

Each L^{i} and R^{i} is 32 bits in length. In the right half of the present state, the function inputs a 32-bit string and a round key. The key stream $(K^{1}, K^{2}, \dots, K^{16})$ is included in the 48-bit round keys that are derived from the 56-bit key *K*. Each K^{i} is a certain permuted chosen of bits from *K*.

Figure 4 presents the ***f*** function. It includes a substitution (using an S-box), followed by a (fixed) permutation, indicated as *P*. Suppose we announce the first evidence of ***f*** by *A* and the second evidence by *J*. Then, the following steps are executed to compute *A* and *J*.

1. A is "expanded" to a bit-string of length 48 based on a fixed expansion function E. E(A) includes 32 bits from A, permuted in a particular way, with 16 bits showing twice.

2. Compute E(*A*) XOR *J* and write the result as the string of eight six-bit strings.

$$B = B_1\ B_2\ B_3 B_4 B_5 B_6 B_7 B_8.$$

3. S-boxes are used, denoted by $S_1, \dots, S_8$. Every S-box has the following value.

$$S_i : \{0,1\}^6 \longrightarrow \{0,1\}^8.$$

Six bits are mapped to four bits. These bits are conventionally depicted as a 4 × 16 array whose entries range from integers 0 to 15. Given a six-bit-long string, we obtain the following:

$$B_j = b_1 b_2 b_3 b_4 b_5 b_6\ .$$

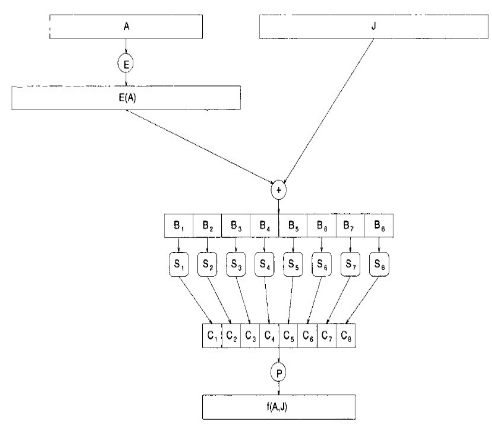

*Figure 3. Process of **f** function [7]*

We compute Sj (Bj) in the following manner. Two bits—b1 and b6—define the binary impersonation of row *r* of Sj ($0 \le r \le 3$). Four bits—b2, b3, b4, and b5—define the binary impersonation of column *c* of Sj ($0 \le c \le 15$). Then, Sj (Bj) is defined as entry Sj (*r*, *c*), written in binary as a four-bit-long string. (Hence, each Sj can be considered a function that inputs one two-bit string and one four-bit string. Sj also generates an output with a four-bit string length.)

In this format, we compute Cj = Sj (Bj), $1 \leq j \leq 8$.

4. String "C = C1, C2, C3, C4, C5, C6, C7", and C8 is 32-bit long and is permuted depending on fixed permutation *P*. The resulting bit-string P(C) is defined as f(*A*, *J*) **[7]**.

Since the adoption of DES in 1977, backdoor DES crackers have improved. Such crackers can decode DES messages in less than a week. For example, a "brute force" attack attempts as many keys as possible to decrypt cipher text into plain text. A special parallel computer is attached using million chips that attempt a million keys every per second. [8].

4. IRREDUCIBLE POLYNOMIAL OVER FINITE FIELDS

Finite fields are fields with finite elements. These fields are called GF, in honor of Evariste Galois (1811–1832). He conducted research on the roots of polynomials and discovered several essential properties. Several cryptographic algorithms are based on finite field arithmetic (such as 1976 ElGamal, Diffie, and Hellman, 1985; Miller, 1986; AES) [9]. All processes executed in the finite field result in an element into that field. Moreover, the arrangement of the finite field must be a power of a prime P^m, where p is a prime number and *m* is a positive integer. Multiplication, addition, exponentiation, inverse multiplication, and division are the most basic arithmetic operations in the finite field. Two polynomials are also either added or subtracted, thereby reducing the result module of the characteristic [10].

Let *p* be a prime number. Integers *mod p*, including integers {0, 1, 2, … , *p*−1} with multiplication and addition performed on *mod p*, is a finite field of order *p*.

In finite field, when prime (*p*) = 2, the elements of $GF(2^n)$ can be conventionally expressed as binary numbers. $GF(2^n)$ can be constructed by using a polynomial basis representation. Here, the elements of $GF(2^n)$ are the binary polynomials of degree at most n_{-1}. A polynomial f(x) in $GF(2^n)$ is presented as in Equation A1, which can be uniquely represented by its ***n*** binary coefficients $(an_{-1}an_{-2}. . . a_{-0})$ [11].

$$f(x) = a_{n-1}\, x^{n-1} + a_{n-2}\, x^{n-2} + \cdots + a_1\, x + a_0 = \sum_{i=0}^{n-1} a_i\, x^i \ldots\ldots \quad \text{A1}$$

"Polynomial f(x) is irreducible and is over a field GF(2^n) if and only if f(x) cannot be expressed as a product of two polynomials, both over GF(2^n) and both of degree lower than that of f(x)" **[11]**. Thus, every polynomial in GF (2^n) can be represented by an *n*-bit number. The irreducibility of *f*(x) with a degree less than *n* means that f(x) cannot be factored as a result of binary polynomials. Two finite field elements are accomplished by adding coefficients for identical powers in their polynomial representations. This addition is executed in GF (2), that is, mod 2. Thus, 1 + 1 = 0.

Consequently, addition and subtraction are equivalent to an exclusive-OR XOR operation of the *n*-bits that represent the field elements of *GF* (2^n) [12].

Finite field multiplication is more difficult than the addition that is completed by multiplying two polynomials for the two elements concerned. Both elements are combined similar to the powers of f(x) in the result.

If the multiplication result in a polynomial is a grade greater than n_{-1}, then the polynomial is reduced by the module of irreducible polynomial m(x) of grade *n*. That is, the polynomial is divided by m(x) and the remainder is kept [12].

4.1 Construction Multiplication Tables for GF (2^n)

The forms of numbers in the table are represented as the elements of GF (2^n) that stand for a polynomial. The table is put up by multiplying the polynomials. Each row and column is numbered. Then, the result is stored in a site represented by the cross of a row number with the column number (multiplied by every other). However, if the product of a polynomial degree is greater than n_{-1}, then the polynomial is split to the selected irreducible polynomial. The remainder is kept as the result. Table 2 represents the multiplication in GF (2^8) [13].

In (2^8), 256 elements exist, and an irreducible polynomial of degree eight is used for reduction. This polynomial is (x) =$x^8 + x^4 + x^3 + x + 1$, which in tuple representation is (1 0 0 0 1 1 0 1 1), corresponding to the hexadecimal number"011B." [11].

Journal of Theoretical and Applied Information Technology
15th March 2019. Vol.97. No 5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 1. List of irreducible polynomials [14]

No.	Irreducible Polynomials	Dec.	No.	Irreducible Polynomials	Dec.
1	$x^8+x^4+x^3+x+1$	283	16	$x^8+x^7+x^3+x+$	395
2	$x^8+x^4+x^3+x^2+1$	285	17	$x^8+x^7+x^3+x^2+$	397
3	$x^8+x^5+x^3+x+1$	299	18	$x^8+x^7+x^4+x^3+$	415
4	$x^8+x^5+x^3+x^2+1$	301	19	$x^8+x^7+x^5+x+$	419
5	$x^8+x^5+x^4+x^3+1$	313	20	$x^8+x^7+x^5+x^3+$	425
6	$x^8+x^5+x^4+x^3+x^2$	319	21	$x^8+x^7+x^5+x^4+$	433
7	$x^8+x^6+x^3+x^2+1$	333	22	$x^8+x^7+x^5+x^4+$	445
8	$x^8+x^6+x^4+x^3+x^2$	351	23	$x^8+x^7+x^6+x+$	451
9	$x^8+x^6+x^5+x+1$	355	24	$x^8+x^7+x^6+x^3+$	463
10	$x^8+x^6+x^5+x^2+1$	357	25	$x^8+x^7+x^6+x^4+$	471
11	$x^8+x^6+x^5+x^3+1$	361	26	$x^8+x^7+x^6+x^4+$	477
12	$x^8+x^6+x^5+x^4+1$	369	27	$x^8+x^7+x^6+x^5+$	487
13	$x^8+x^6+x^5+x^4+x^2$	375	28	$x^8+x^7+x^6+x^5+$	499
14	$x^8+x^6+x^5+x^4+x^3$	379	29	$x^8+x^7+x^6+x^5+$	501
15	$x^8+x^7+x^2+x+1$	391	30	$x^8+x^7+x^6+x^5+$	505

Table 2. Multiplication in GF (2^8) [15]

			00000001	00000010	00000011	...	11111101	11111110	11111111
Binary	Poly.	*	1	x	x+1	...	253	254	255
00000001	1	1	1	x	x+1	...	$x^7+x^6+x^5+x^4+x^3+x^2+1$	$x^7+x^6+x^5+x^4+x^3+x^2+x$	$x^7+x^6+x^5+x^4+x^3+x^2+x+1$
00000010	x	2	x	x^2	x^2+x	...	$x^7+x^6+x^5+1$	$x^7+x^6+x^5+x^2+x+1$	$x^7+x^6+x^5+x^2+1$
00000011	x+1	3	x+1	x^2+x	x^2+1	...	$x^4+x^3+x^2$	x^4+x^3+1	x^4+x^3+x
....	....	....	....	...	...	...	...	...	...
....	....	....	....	...	...	...	...	...	...
11111101	$x^7+x^6+x^5+x^4+x^3+x^2+1$	253	$x^7+x^6+x^5+x^4+x^3+x^2+1$	$x^7+x^6+x^5+1$	$x^4+x^3+x^2$	...	x^4+x^2+x+1	x^3+x+1	$x^7+x^6+x^5+x^4+x^2+x$
11111110	$x^7+x^6+x^5+x^4+x^3+x^2+x$	254	$x^7+x^6+x^5+x^4+x^3+x^2+x$	$x^7+x^6+x^5+x^2+x+1$	x^4+x^3+1	...	x^3+x+1	x^4+x	$x^7+x^6+x^5+x^3+x^2$
11111111	$x^7+x^6+x^5+x^4+x^3+x^2+x+1$	255	$x^7+x^6+x^5+x^4+x^3+x^2+x+1$	$x^7+x^6+x^5+x^2+1$	x^4+x^3+x	...	$x^7+x^6+x^5+x^4+x^2+x$	$x^7+x^6+x^5+x^3+x^2$	x^4+x+1

4.2 Multiplicative Inverse

The Euclidean algorithm can be suitable to detect the "Greatest Common Divisor" (GCD) of two polynomials. All elements of the finite field sets other than 0 have a multiplicative inverse. However, the extended Euclidean algorithm can be suitable to locate the multiplicative inverse of polynomials. Latter algorithm discovers the multiplicative inverse of b(x) mod m(x) if the degree of b(x) is less than the degree of m(x) and GCD [m(x), b(x)] = 1. If m(x) is an irreducible polynomial, then it has no factor other than itself or 1. Thus, GCD [m(x), b(x)] = 1. The multiplicative inverse table of GF (2^n) can be found in the multiplication table of GF(2^n) [11].

6. MODIFIED DES

This study proposes the design of modified DES algorithm called MODES. Changes are made in the DES structure in this algorithm, making it more secure than DES.

MODES perform all its computations on bytes rather than in bits. Hence, MODES treats the eight-byte plain text block as a non-zero polynomial element of finite field GF (2^8). The eight bytes are divided into two four matrix sizes (2 x 2), and MODES uses 16 rounds. Each round comprises four keys (two external keys and two internal keys) as matrix bytes (2 x 2). The main key size is 64 bytes generated by one of the random number generation keys as a non-zero polynomial element of finite field GF(2^8). MODES algorithm exploits multiplication of matrices to provide substitution and permutation techniques. External keys (2 x 2) represent a diagonal matrix form derived from the main keys for encrypting plain text data. Internal keys are dynamically generated from the product of external keys with data matrix (2 x 2). this algorithm are based on the mathematical theory of GF(2^8). Hence, all operations use irreducible polynomial "**x8 + *x4* + *x3* + *x* + *1***". Therefore, decryption requires the inverse of the four key matrices in each round. Figure 4 shows the schematic of MODES structure.

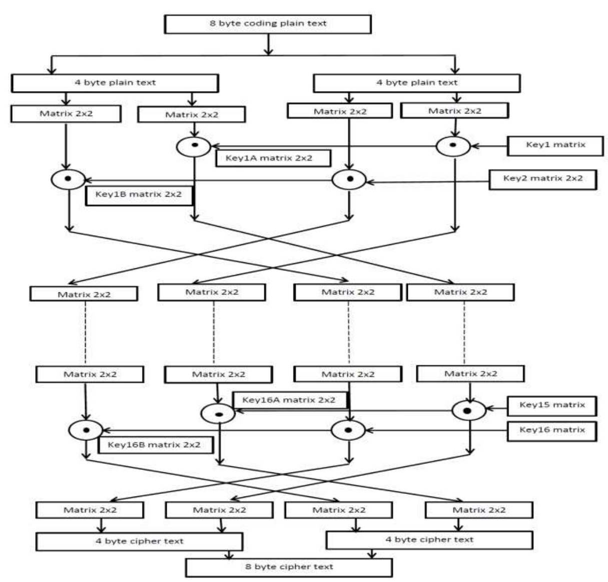

Figure 4. Schematic of MODES structure

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

6.1 Key Generation

This section presents the creation of 32 round keys from the 64-byte main key, in which each cycle takes four keys with a four-byte length for each key. Two keys are generated through the main key called external keys. The other two keys are dynamically generated during the encryption process. The following are the basic processes for key generation.

6.1.1 Main key generation

Initially, the 64-byte main key is generated by one of the random number generation algorithms as a non-zero polynomial element. Any byte of a set in finite field GF(2^8) is not repeated. Then, it is ordered in the matrix as an 8 x 8 byte.

6.1.2 Shift rows and shift columns

Permutation is performed in bytes instead of in bits. These operations execute by shifting operations on all bytes without changing bits into bytes.

Before rounds of key generation, shifting operation is performed to left and down three bytes by shifting the rows and columns selected previously. For example, shifting is executed in the selected column index (2, 5, 8) and row index (1, 4, 7). Figure 5 illustrates the shifting operation.

	1	2	3	4	5	6	7	8
1	102	55	111	200	6	17	22	54
2	44	59	245	212	190	170	2	133
3	62	216	263	231	41	122	246	68
4	99	234	89	25	231	40	90	64
5	23	233	234	267	175	217	97	222
6	11	62	13	60	214	180	251	220
7	67	70	88	110	165	150	52	199
8	91	98	23	1	241	203	50	7

a. shifting row and column operation (three bytes)

	1	2	3	4	5	6	7	8
1	17	22	220	102	98	111	200	214
2	44	67	245	212	165	170	2	199
3	62	62	263	231	241	122	246	7
4	40	90	54	99	55	89	25	6
5	23	59	234	267	190	217	97	133
6	11	216	13	60	41	180	251	68
7	150	52	64	70	234	88	110	231
8	91	233	23	1	175	203	50	222

b. after permutation
Figure 5. Shifting operation

6.1.3 Rounds of Key Generation (External and Internal Keys)

Create external keys After the direct shifting process, the main key is to split the main matrix into two matrices (2 x 16). Then, each matrix repeats itself twice. Thus, the matrix size should be 4 x 16. This matrix helps obtain rounds of external keys in a diagonal matrix format.

Diagonal matrices can be inverse matrices. Decryption is also possible. Figure 6 shows the operation of creating external keys.

17	22	220	102	98	111	200	214	44	67	245	212	165	170	2	199	44
62	62	263	231	241	122	246	7	62	40	90	54	99	55	89	25	6

23	59	234	267	190	217	97	133	11	216	13	60	41	180	251	68
150	52	64	70	234	88	110	231	91	233	23	1	175	203	50	222

a. The main key is split into two matrices (2x16),

K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12	K13	K14	K15	K16
17	22	220	102	98	111	200	214	44	67	245	212	165	170	2	199
62	62	263	231	241	122	246	7	40	90	54	99	55	89	25	6
17	22	220	102	98	111	200	214	44	67	245	212	165	170	2	199
62	62	263	231	241	122	246	7	40	90	54	99	55	89	25	6

b. Matrices repeat itself twice.

Figure 6. Operation used in creating external keys

Then, each key has a diagonal matrix.

$$K1 = \begin{pmatrix} 17 & 62 \\ 62 & 17 \end{pmatrix}, K9 = \begin{pmatrix} 44 & 40 \\ 40 & 44 \end{pmatrix} \ldots\ldots K32 = \begin{pmatrix} 68 & 222 \\ 222 & 68 \end{pmatrix}$$

Create internal keys Internal keys are diagonal matrices (2 × 2) dynamically created during the encryption operation. This operation is created by multiplying two diagonal matrices (external keys and data matrices).

These keys do not need to be known during construction in encryption or decryption operation. The decryption process is dynamically conducted by finding the key inverse performed by the proposed algorithm. Only one irreducible polynomial is used in the key inverse.

This technique assists in gaining high security, making keys difficult to guess.

6.2 Encryption Process

In the proposed MODES, encoded words are considered the plain text. Each word is converted into two bytes in a binary form saved in the encoding (dictionary) table. This table consists of all sensitive words and numbers (0–9). Each word or number is assigned as two integers (two

bytes) starting from 0 to 255 as polynomial formats in GF (2^8).
MODES is a symmetric approach, has 64-bit in block cipher , for encryption operation used two keys and two keys inverse for decryption. The major component of the block is then subsequently partitioned into two sub-block plain texts. Each sub-block is also partitioned into two bytes. Then, every sub-block is expanded by repeating itself to create diagonal matrices (2 x 2). Four input matrices are entered in the multiplication operation with four diagonal key matrices. Each expressed element of key matrices is a polynomial element in GF(2^8).
The two matrices in the right side are multiplied by two external key matrices, whereas the two matrices in the left side are multiplied by two internal key matrices. Internal keys are generated from the product of data and external key matrices.
Encryption function: Plain text blocks are partitioned into four blocks to be encrypted by four keys in each round. The phases consist of 16 rounds of the same tasks.
Consider that all operations should be done in GF using the degree of irreducible polynomial. Each multiplication operation of two diagonal matrices (2 x 2) performs eight (AND) and four (XOR) operations on bits, thus providing high substitution.
At the end of each round, the results are switched between them, thereby providing an additional level for permutation operation.
The output of the last round (16th) represents the end of the encryption operation.
Figure 7 shows the encryption process in a single round, where labeled L = (left), SL = (second left), SR = (second right), and R = (right).

6.2.1 The MDES Encryption Algorithm:

1. Find the coding of the plain text.
2. Convert the plain text to a binary polynomial code. The irreducible polynomial used is expressed as "m =$\boldsymbol{x^8 + x^4 + x^3 + x + 1}$"
3. The plain text block is then subsequently partitioned into four sub-block plain texts.
4. Multiply sub-block matrix (R_{i-1}) with K1 matrix and sub-block matrix (SR_{i-1}) with K2 matrix based on the following equations:

$$\boldsymbol{Li = (K2 \,.\, SR_{i-1}) \bmod m}$$
$$\boldsymbol{SL_i = (K1 \,.\, R_{i-1}) \bmod m}$$

Where the result of $\boldsymbol{Li}$ and $\boldsymbol{SL_i}$ is equal to internal keys KA1 and KB2.

$$\mathbf{KA1 = (K1 \,.\, \boldsymbol{R_{i-1}}) \bmod m}$$
$$\mathbf{KB2 = (K2 \,.\, \boldsymbol{SR_{i-1}}) \bmod m}$$

5. Multiply sub-block matrix (L_ (i-1)) with KB2 matrix and sub-block matrix (SL_(i-1)) with KA1 matrix based on the following equations:

$$\boldsymbol{SR_i = (KB2 \,.\, L_{i-1}) \bmod m}$$
$$\boldsymbol{R_i = (KA1 \,.\, SL_{i-1}) \bmod m}$$

6. Outputs are swapped to produce pre-outputs;
7. Repeat the encryption process for the 16 Rounds of the same task.
8. Finally, four matrix ciphers are gathered in a Block cipher text array.

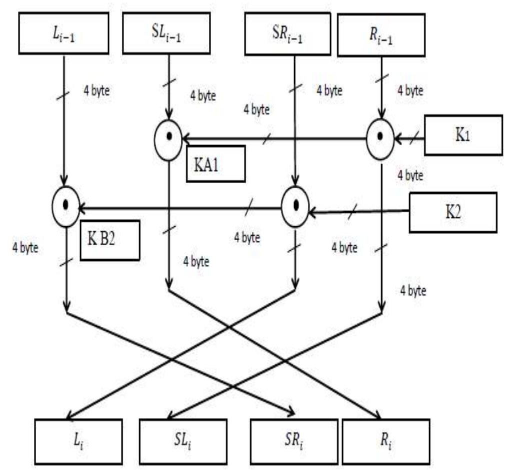

Figure 7. Encryption process in a single round

Example:

The following example illustrates our technique in Encryption process:

*Plain text = **sahab Dheyaa Mohammed jawad***

Where

Key One= **K1** = 99,111, 111, 99 = $\begin{pmatrix} 99 & 111 \\ 111 & 99 \end{pmatrix}$

*Key Two=**K2**= 109,112.112.109* = $\begin{pmatrix} 109 & 112 \\ 112 & 109 \end{pmatrix}$

Round Keys are selected from main Key that generated by one of the random number generation algorithms as a Polynomial non- zero elements.

"Irreducible polynomial = **m** = ($X^8+X^4+X^3+X+1$)"

To encryption of plaintext is performed in the following:

1- Find the encode of the words ***[sahab Dheyaa Mohammed jawad]*** *in the encoding table :* ***[117,116,101,114,88,102,120,46]*** *respectively.*

2- Convert plaintext (encoded words) to binary (polynomial).but in this example represent the numbers in decimal rather than the binary for simplify only.

K1, k2
$=99 = [01100011] = [X^6+ X^5+X+1]$.
$=111= [01101111] = [X^6+X^5+ X^3+ X^2+x+1]$.
$=109= [01101101] = [X^6+X^5+ X^3+ X^2+1]$.
$=112= [01110000] = [X^6+X^5+X^4]$".

Sahab
$=117= [01110101] = [X^6+X^5+ X^4+ X^2+1]$.
$=116= [01110100] = [X^6+X^5+ X^4+X^2]$.
Dheyaa
$=101= [01100101] = [X^6+X^5+ X^4+1]$.
$=114= [01110010] = [X^6+X^5+ X^4+ X]$.
Mohammed
$= 88 = [01011000] = [X^6+X^4+X^3]$
$= 102 = [01100110] = [X^6+X^5+X^2+X]$
jawad
$= 120 = [01111000] = [X^6+X^5 + X^4+X^3]$
$= 46 = [00101110] = [X^5+ X^3+ X^2+x]$

Plain text *= 117, 116, 101, 114, 88, 102, 120, 46*

3- *Partition to four diagonal matrices.*

$$SL_{i-1} = \begin{pmatrix}101 & 114\\ 114 & 101\end{pmatrix}\ SR_{i-1} = \begin{pmatrix}88 & 102\\ 102 & 88\end{pmatrix},$$
$$R_{i-1} = \begin{pmatrix}120 & 46\\ 46 & 120\end{pmatrix}\ L_{i-1} = \begin{pmatrix}117 & 116\\ 116 & 117\end{pmatrix}$$

4- *The Encryption Process:*

- *Perform the following equations to produce the Cipher text*

$$SL_i = (K1.\ R_{i-1})\ \mathbf{mod\ m}$$
$$Li = (K2.\ SR_{i-1})\ Mod\ m$$

Generation internal KA1, KB2 the equations above

$$KA1 = (K1 . R_{i-1})\ \text{mod m}$$
$$KB2 = (K2 . SR_{i-1})\ \text{mod m}$$

$$SL_i = \begin{pmatrix}99 & 111\\ 111 & 99\end{pmatrix} . \begin{pmatrix}120 & 46\\ 46 & 120\end{pmatrix}\ Mod\ m$$

= ((99 AND 120) XOR (111 AND. 46)) mod m = 91

= ((99 AND.46) XOR (111 AND 120)) mod m = 158

= ((111 AND 120) XOR (99 AND 46)) mod m =158
= ((111 AND 46) XOR (99 AND 120)) mod m = 91

$$SL_i = \begin{pmatrix}91 & 158\\ 158 & 91\end{pmatrix} \qquad KA1 = \begin{pmatrix}91 & 158\\ 158 & 91\end{pmatrix}$$

-So to gain the Li, KB2 must followed the same operation of SL_i:

$$Li = \begin{pmatrix}109 & 112\\ 112 & 109\end{pmatrix}. \begin{pmatrix}88 & 102\\ 102 & 88\end{pmatrix}\ Mod\ m = \begin{pmatrix}119 & 151\\ 151 & 119\end{pmatrix}$$

$$KB2 = \begin{pmatrix}119 & 151\\ 151 & 119\end{pmatrix}$$

-Then find **SRi, Ri** by Perform the following equations:

$$R_i = (KA1.\ SL_{i-1})\ \text{mod m}$$
$$SR_i = (KB2\ .\ L_{i-1})\ \text{mod m}$$

$$Ri = \begin{pmatrix}91 & 158\\ 158 & 91\end{pmatrix}.\begin{pmatrix}101 & 114\\ 114 & 101\end{pmatrix}\ Mod\ m$$

((91 AND 101) XOR (158 AND 114)) mod m = 70
((91 AND 114) XOR (158AND 101)) mod m= 207
((158 AND 101) XOR (91AND 114)) mod m= 207
((158 AND 114) XOR (91 AND 101)) mod m= 70

$$Ri = \begin{pmatrix}70 & 207\\ 207 & 70\end{pmatrix}$$

$$SRi = \begin{pmatrix}119 & 151\\ 151 & 119\end{pmatrix}. \begin{pmatrix}117 & 116\\ 116 & 117\end{pmatrix}\ mod\ m$$
$$SRi = \begin{pmatrix}121 & 153\\ 153 & 121\end{pmatrix}$$

- Gathering in one block cipher text array.

$$\begin{pmatrix}119 & 151\\ 151 & 119\end{pmatrix}\begin{pmatrix}91 & 158\\ 158 & 91\end{pmatrix}\begin{pmatrix}121 & 153\\ 153 & 121\end{pmatrix}\begin{pmatrix}70 & 207\\ 207 & 70\end{pmatrix}$$

Cipher text = 119, 151, 91, 158, 121, 153, 70, 207

6.3 Decryption Process

Decryption is the operation of retrieved the plain text from the cipher text. Decryption operation is the similar operation as the encryption process. The principle is as follows: use cipher text as the input to the proposed technique, and use the inverse of K_i in the reverse order. Thus, use $\boldsymbol{K_n}$ and $\boldsymbol{K_{n-2}}$ in the first round, $\boldsymbol{K_{n-2}}$ and $\boldsymbol{K_{n-3}}$ in the second round, and so on, until $\boldsymbol{K_1}$ and $\boldsymbol{K_2}$ are used in the last round. The inverse of internal keys $\boldsymbol{KB_2}$ and $\boldsymbol{KA_2}$ should also be used at the

Journal of Theoretical and Applied Information Technology
15th March 2019. Vol.97. No 5

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

same time. Figure 8 shows the decryption process in a single round. In decryption operation, converted the cipher text into the array of plain text using the same scenario in the encryption. The module inverse of key matrices must be calculated. The $\boldsymbol{K^{-1}}$ inverse of matrix *K* is defined by the following equation:

$$\mathbf{K}.\boldsymbol{K^{-1}} = \boldsymbol{K^{-1}}.\ \mathbf{K} = \boldsymbol{I},$$

Where

I : matrix that have all zeros except the main diagonal.

K **:** is the matrix.

$\boldsymbol{K^{-1}}$: Is the inverse of K matrix

Hence, decryption matrices are generated by multiplying inverse key and cipher text matrices. All operations in this process are performed in the modulus of the same degree *m*, which is an irreducible polynomial in GF. Then, they are converted into a character string using the encoded (dictionary) table.

6.3.1 The MDES Decryption Algorithm:

1. Find the inverse of key matrices (Ki) and apply them on cipher matrices related with irreducible polynomials. M= "$x^8 + x^4 + x^3 + x + 1$",) is used here.
2. Partition the eight-byte cipher text block into four cipher text matrices (2 x 2) (every two bytes are repeated to four bytes).
3. Multiply sub-block matrix (SL_i) with K_{n-1}^{-1} matrix and sub- block matrix (L_i) with K_{n-2}^{-1} matrix based on the following equations.

$$\boldsymbol{R_{i-1}} = (\ \boldsymbol{K1^{-1}}\ .\ \boldsymbol{SL_i})\ \textbf{mod m.}$$
$$\boldsymbol{SR_{i-1}} = (\ \boldsymbol{K2^{-1}}\ .\ \boldsymbol{L_i}\ )\ \textbf{mod m.}$$

4. Find the inverse of internal key matrices KA_{n-1}^{-1} and KB_{n-1}^{-1}.

5. Multiply sub-block matrix (SR_i) with KB_{n-1}^{-1} matrix and sub-block matrix (R_i) with KA_{n-1}^{-1}matrix based on the following equations:

$$\boldsymbol{SL_{i-1}} = (\boldsymbol{KA1^{-1}}\ .\ \boldsymbol{R_i})\ \textbf{mode m.}$$

$$\boldsymbol{L_{i-1}} = (\ \boldsymbol{KB2^{-1}}\ .\ \boldsymbol{SR_i})\ \textbf{mod m.}$$

6. Repeat the decryption process for 16 rounds of the same task.
7. Four plain text binary matrices (as polynomial matrices) are gathered in a block cipher array.
8. Convert the block cipher of the binary form to encoded text (eight bytes).

Decode the block cipher from the dictionary table.

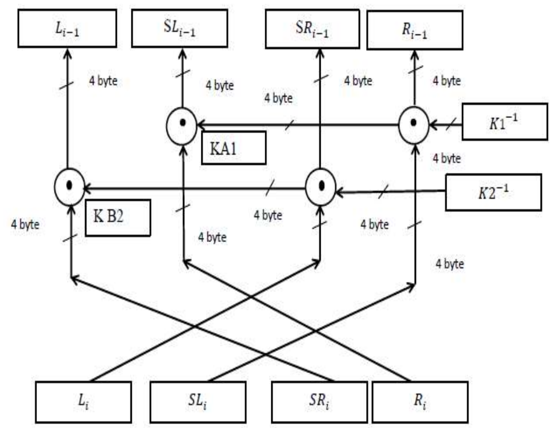

Figure 8. Decryption process in a single round

Example:

The following example illustrates our technique in decryption process:

Cipher text = 119, 151, 91, 158, 121, 153, 70, 207

Key One= $\boldsymbol{K_{n-15}}$ = **K1**

K1 = 99,111, 111, 99 = $\begin{pmatrix} 99 & 111 \\ 111 & 99 \end{pmatrix}$

Key Two= $\boldsymbol{K_{n-14}}$ = **K2**

K2= 109,112.112.109 =$\begin{pmatrix} 109 & 112 \\ 112 & 109 \end{pmatrix}$

1-Find the Inverse of the keys matrices Ki:

K1 = 99,111, 111, 99 = $\begin{pmatrix} 99 & 111 \\ 111 & 99 \end{pmatrix}$

- find the determinant of Key matrix.

Det. = $\begin{pmatrix} 99 & 111 \\ 111 & 99 \end{pmatrix}$ = ((99 And 99) XOR (111 AND 111)) mod ***m***

Det.=80

-find the Multiplicative Inverse of the determinate (80) from the Table. Multiplication Inverse in GF (2^8) with the ‘Irreducible Polynomial m(x) = (x^8 +x^4 +x^3 +x+1)”.

Multiplicative Inverse of **80** =**237**

- Creating the Adjugate Matrix to Find the Inverse Matrix.

Adg. = $\begin{pmatrix} 99 & -111 \\ -111 & 99 \end{pmatrix}$

$\boldsymbol{K1^{-1}} = \mathbf{237}.\begin{pmatrix} 99 & -111 \\ -111 & 99 \end{pmatrix}\ \boldsymbol{mod\ m}$

$\boldsymbol{K1^{-1}} = \begin{pmatrix} 219 & 107 \\ 107 & 219 \end{pmatrix}$

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Inverse of the keys matrices K2:

$\boldsymbol{K2^{-1}}= \begin{pmatrix} 193 & 129 \\ 129 & 193 \end{pmatrix}$

2- The cipher text block (8 byte) partition into 4 cipher text matrices (2x2)

$L_i = \begin{pmatrix} \mathbf{119} & \mathbf{151} \\ \mathbf{151} & \mathbf{119} \end{pmatrix}$ $SL_i = \begin{pmatrix} \mathbf{91} & \mathbf{158} \\ \mathbf{158} & \mathbf{91} \end{pmatrix}$ $SR_i = \begin{pmatrix} \mathbf{121} & \mathbf{153} \\ \mathbf{153} & \mathbf{121} \end{pmatrix}$ $R_i = \begin{pmatrix} \mathbf{70} & \mathbf{207} \\ \mathbf{207} & \mathbf{70} \end{pmatrix}$

3- Find the $\boldsymbol{R_{i-1}}$, $\boldsymbol{SR_{i-1}}$.

$\boldsymbol{R_{i-1}} = (\boldsymbol{K1^{-1}} \,.\, \boldsymbol{SL_i})$ **mod m**

$\boldsymbol{R_{i-1}} = \begin{pmatrix} \mathbf{219} & \mathbf{107} \\ \mathbf{107} & \mathbf{219} \end{pmatrix} . \begin{pmatrix} \mathbf{91} & \mathbf{158} \\ \mathbf{158} & \mathbf{91} \end{pmatrix}$ **mod m**

$\boldsymbol{R_{i-1}} = \begin{pmatrix} \mathbf{120} & \mathbf{46} \\ \mathbf{46} & \mathbf{120} \end{pmatrix}$

$\boldsymbol{SR_{i-1}} = (\boldsymbol{K2^{-1}} \,.\, \boldsymbol{L_i})$ **mod m**

$\boldsymbol{SR_{i-1}} = \begin{pmatrix} \mathbf{193} & \mathbf{129} \\ \mathbf{129} & \mathbf{193} \end{pmatrix} . \begin{pmatrix} \mathbf{119} & \mathbf{151} \\ \mathbf{151} & \mathbf{119} \end{pmatrix}$ **Mod m**

$\boldsymbol{SR_{i-1}} = \begin{pmatrix} \mathbf{88} & \mathbf{102} \\ \mathbf{102} & \mathbf{88} \end{pmatrix}$

$\boldsymbol{KA1} = \boldsymbol{SL_i}$ **KB2** $= \boldsymbol{L_i}$

4-Find the Inverse of the internal keys matrices KA1, KB2:

$\boldsymbol{KA1^{-1}} = \begin{pmatrix} \mathbf{10} & \mathbf{222} \\ \mathbf{222} & \mathbf{10} \end{pmatrix}$

$\boldsymbol{KB2^{-1}} = \begin{pmatrix} \mathbf{103} & \mathbf{214} \\ \mathbf{214} & \mathbf{103} \end{pmatrix}$

5. find the SL_{i-1}, L_{i-1}

$\boldsymbol{SL_{i-1}} = (\boldsymbol{KA1^{-1}} \,.\, \boldsymbol{R_i})$ **mode m**

$\boldsymbol{SL_{i-1}} = \begin{pmatrix} 10 & 222 \\ 222 & 10 \end{pmatrix} . \begin{pmatrix} 70 & 207 \\ 207 & 70 \end{pmatrix}$ Mod m

$SL_{i-1} = \begin{pmatrix} 101 & 114 \\ 114 & 101 \end{pmatrix}$

$\boldsymbol{L_{i-1}} = (\boldsymbol{KB2^{-1}} \,.\, \boldsymbol{SR_i})$ **mod m**

$L_{i-1} = \begin{pmatrix} 103 & 214 \\ 214 & 103 \end{pmatrix}) . \begin{pmatrix} 121 & 153 \\ 153 & 121 \end{pmatrix}$ Mod m

$\boldsymbol{L_{i-1}} = \begin{pmatrix} 117 & 116 \\ 116 & 117 \end{pmatrix}$

6. - gathering in one block plain-text array.

$\boldsymbol{L_{i-1}} = \begin{pmatrix} \mathbf{117} & \mathbf{116} \\ \mathbf{116} & \mathbf{117} \end{pmatrix}, \boldsymbol{SL_{i-1}} = \begin{pmatrix} \mathbf{101} & \mathbf{114} \\ \mathbf{114} & \mathbf{101} \end{pmatrix}$

$, \boldsymbol{SR_{i-1}} = \begin{pmatrix} \mathbf{88} & \mathbf{102} \\ \mathbf{102} & \mathbf{88} \end{pmatrix}, \quad \boldsymbol{R_{i-1}} = \begin{pmatrix} \mathbf{120} & \mathbf{46} \\ \mathbf{46} & \mathbf{120} \end{pmatrix}$

Plain-text= (117, 116, 101, 114, 88, 102, 120, 46)

7. RESULTS AND TESTING

The proposed approach allows the block encryption of cipher text pass through several steps to increase complexity and linearity. Therefore, immunity is improved, and resistance against malignant actions is enhanced.

This approach is built using various techniques, including modern internal operations, reversible operations of key matrices, multiplication operations of matrices that provide good substitutions, and permutation for cipher texts.

The number of possible keys that can decrypt one eight-bit block size in the proposed MODES algorithm is 2^8.

Without knowing inverse key matrices, an attacker cannot compute plain texts from cipher texts. The attacker should detected one of the 30 irreducible polynomials of degree (8) used in the proposed approach. Thus, the number of possible keys that can decrypt one eight-bit block size is $30*2^8$ = 7,680.

In this section, statistical tests are implemented on keys and cipher texts.

7.1 Randomness Tests and Statistical Analysis

Randomness tests and statistical analysis are important for this proposed work. The "National Institute of Standards and Technology (NIST)" supposes that these proceedings are useful in determining if any deviation or bias exists in the correlation of input/output bits and in revealing the performance of efficiency. The research provides an accepted and reasonable implication according to the NIST randomness tests.

NIST Test Suite is a statistical set, which includes 15 tests. Such tests are developed to examine the randomness of binary sequences produced by either hardware or software using cryptographic random or pseudorandom number generators. These tests works on various types of non-randomness that can performance in a sequence. Certain tests are decomposable into various subtests. The 15 tests are specified below.

Table 3 reveals the results of the randomness test for the encrypted data of modified MODES. The table contains flags of values (pass or fail .A inference concerning the quality of the sequences can be made on the basis of the P-values. In this section, we select the file sample of 1,000 encryption records as a set of 2,048,000 bits in length sequences of 0's and 1's for evaluation.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

Table 3. Results of the randomness test for the encrypted data

	Statistical Tests	Input Size (n)	P-Value >=0.01
1	**Frequency (Monobit) Test**	900000	0.439106
2	**Block Frequency (*m = 64*)**	10000	0.043657
3	**Overlapping Templates (m = 8)**	900000	0.906922
4	**Non- Overlapping Templates Test**	900000	0.175699
5	**Serial Test (m = 64, 2m∇Ψ)**	900000	
	Serial (1)		1.000000
	Serial (2)		1.000000
6	**Approximate Entropy Test (m = 10)**	1000	0.991246
7	**Linear Complexity Test (M = 64)**	900000	0.909498
8	**Cumulative Sums Test**	900000	
	Cumulative Sums (Forward(		0.676715
	Cumulative Sums (Reverse)		0.171146
9	**Runs Test**	900000	0.114184
10	**Longest Run of One's Test**	900000	0.788230
11	**Binary Matrix Rank Test**	4000	0.022886
12	**Spectral DFT Test**	850000	0.026145
13	**Random Excursions Test**	2000000	
	Random Excursions (1)		0.564849
	Random Excursions (2)		0.925536
	Random Excursions (3)		0.717109
	Random Excursions (4)		0.320232
	Random Excursions (5)		0.932431
	Random Excursions (6)		0.706715
	Random Excursions (7)		0.936945
	Random Excursions (8)		0.578396
14	**Random Excursions Variant Test**		
		2000000	
	Random Excursions Variant (1)		1.979414
	Random Excursions Variant (2)		1.126984
	Random Excursions Variant (3)		6.392400
	Random Excursions Variant (4)		1.108237
	Random Excursions Variant (5)		2.859941
	Random Excursions Variant (6)		4.345137
	Random Excursions Variant (7)		4.444900
	Random Excursions Variant (8)		5.353695
	Random Excursions Variant (9)		7.655383
	Random Excursions Variant (10)		8.114368
	Random Excursions Variant (11)		7.133901
	Random Excursions Variant (12)		7.022265
	Random Excursions Variant (13)		6.684202
	Random Excursions Variant (14)		7.655383
	Random Excursions Variant (15)		9.331191
	Random Excursions Variant (16)		6.042367
	Random Excursions Variant (17)		3.941459
	Random Excursions Variant (18)		4.178978

For every statistical test, a set of P-values (identical to the set of sequences) is created. For a detected significance level, a certain percentage of P-values are expected to point out failure. . A statistical test is pass whenever the P-value ≥ α and else is fails. For every statistical test, the ratio of sequences that pass is calculate and analyzed accordingly. A wide analysis should be executed using additional statistical procedures to Interpretation of Empirical Results.

8. CONCLUSION

The proposed algorithm has improved DES algorithm on the basis of the mathematical theory of GF (2^8). Matrix multiplication operation is used instead of XOR operation, which provides substitution and permutation to each multiplication process. XOR operation also assists the proposed algorithm to decrease the consumed time in encryption and decryption processes. The efficiency of this method depends on the use of two diagonal key matrices derived from the randomly generated main key. Operations of the two dynamic internal keys, matrix multiplication, and replacement of the old XOR make the known plain text difficult to attack.

The proposed algorithm is based on two main subjects. First, the time needful for encryption /decryption is sped up. Second, high security is increased to provide robustness to the algorithm. Any kind of intruding against the algorithm cannot easily obtain the key.

REFERENCES

[1] Aleisa, Noura. "A Comparison of the 3DES and AES Encryption Standards." International Journal of Security and Its Applications9.7 (2015): 241-246..

[2] Alanazi, Hamdan, et al. "New comparative study between DES, 3DES and AES within nine factors." *arXiv preprint arXiv:1003.4085* (2010).

[3] Majhi, Jyotirmayee. *Modified Hill-Cipher and CRT Methods in Galois Field GF (2^ M) for Cryptography*. Diss. 2009

[4] Akhtar, Adeem, Muhammad Zia Ullah Baig, and Waleej Haider. "Enhancing the Security of Simplified DES Algorithm Using Transposition and Shift Rows." *International Journal of Computer Science and Software Engineering* 6.5 (2017): 115.

[5] Sison, Ariel M., et al. "Implementation of Improved DES Algorithm in Securing Smart Card Data." *Computer Applications for Software Engineering, Disaster Recovery, and Business Continuity*. Springer, Berlin, Heidelberg, 2012. 252-263.

[6] S. R.kumar, S. A.Yadav, A. Singh, S. Sharma, "Design and Implementation of Improved Data Encryption Standard", Amity Journal of Computational Sciences,Amity University, Volume 1 Issue 1 2017 ISSN : 2456-6616 (Online).

[7] Stinson, Douglas R. Cryptography: theory and practice. CRC press, 2005.

[8] N. Aleisa." Comparison of the 3DES and AES Encryption Standards", International Journal of Security and Its Applications, Vol.9, No.7 (2015), pp.241-246 .

[9] Lidl, Rudolf, and Harald Niederreiter. *Introduction to finite fields and their applications*. Cambridge university press, 1994.

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

[10] Madhuri, O. B. B., E. Rambabu, and Malijeddi Murali. "Design and Implementation of Arithmetic Unit for GF (2m)." *International Journal of Advanced Research in Computer Engineering & Technology (IJARCET)* 1.9 (2012): pp-185.

[11] William, Stallings. "Cryptography and network security: principles and practice." *Prentice-Hall, Inc* (2012): 23-50.

[12] Yaco, Basima Z., and AbdulMonem S. Rahma. "Real-time Partial Encryption of Oigital Video using Symmetric Dynamic Dual Keys Algorithm (SDD)." *Engineering and Technology Journal* 30.5 (2012): 710-728.

[13] Yacob, B. M. "An Improved Algorithm for Partial Cryptography of Digital Video", University of Zakho, Ph.D Thesis,2012.

[14] Abduallah, Wafaa Mustafa, Abdul Monem S. Rahma, and Al-Sakib Khan Pathan. "A new data hiding technique based on irreducible polynomials." *Journal of Emerging Technologies in Web Intelligence* 5.1 (2013).

[15] Rahma, Abdul Monem Saleh, and Zainab Mohammed Hussein. "Modified Rc4 Dual Key algorithm based on Irreducible Polynomial." *International Journal of Emerging Trends & Technology in Computer Science (IJETTCS)* 4.2(2015).

[16] Kaipa, Adi Narayana Reddy, et al. "A hybrid cryptosystem using variable length sub key groups and byte substitution." *Journal of Computer Science* 10.2 (2014): 251

[17] Ismail, I. A., Mohammed Amin, and Hossam Diab. "How to repair the Hill cipher." *Journal of Zhejiang University-Science A* 7.12 (2006): 2022-2030.

[18] Panduranga, H. T., and Naveen Kumar SK. "Advanced partial image encryption using two-stage hill cipher technique." *International Journal of Computer Applications* 60.16 (2012).

[19] SelvaKumar, Thangarasu N. Dr Arul Lawrence. "ENCRYPTION USING LESTER HILL CIPHER ALGORITHM." (2014).