TEM Journal. Volume 9, Issue 3, Pages 1277-1284, ISSN 2217-8309, DOI: 10.18421/TEM93-57, August 2020.

Maintaining the Integrity of Encrypted Data by Using the Improving Hash Function Based on GF (2^8)

Sahab Dheyaa Mohammed [1], Abdul Monem S. Rahma [2,]
Taha Mohammed Hasan [3]

[1] *University of Information Technology and Communications, Baghdad, Iraq*
[2] *Imam Ja'afar Al-Sadiq University,Computer Engineering Tech., Baghdad, Iraq*
[3] *University of Diyala, College of Science, Diyala, Iraq*

***Abstract* - The issue of protecting the information from penetration has become an important issue. The system that depends on the encryption ensures the confidentiality of the information non-disclosure of sensitive information but does not ensure the integrity of data from destruction and change. In this paper, a proposed system is designed to protect the confidentiality and integrity of data from penetration, disclosure, and destruction. The proposed system based on the polynomial numbers of GF (2^8) is achieved by improving the encryption approach using the idea of the magic square and the linear equation system also uses improving the digital signature method for ensuring that data is not changed or modified. The system has higher encryption and decryption throughput (548.924Kb /sec), (548.924 Kb /sec) and acceptable value 0.759294 of the randomness data according to the NIST randomness tests as well as a high confusion and diffusion in cipher text based on the ratio of Avalanche effect test.**

***Keywords* - Gaussian Elimination, hash function, integrity, Magic Square, Finite Field.**

DOI: 10.18421/TEM93-57
https://doi.org/10.18421/TEM93-57

Corresponding author: Sahab Dheyaa Mohammed,
University of Information Technology and Communications, Baghdad, Iraq
Email: sahab7dia@gmail.com

Received: 29 April 2020.
Revised: 11 August 2020.
Accepted: 17 August 2020.
Published: 28 August 2020.

1. Introduction

The cryptographic systems are a suitable solution for protecting data through the transition. Such systems have ability to guarantee both confidentiality and integrity of sensitive data, and also against unauthorized modifying data. So, it is needed to be careful when building a cryptographic system on top of outsourced cloud storage services [1].

Several methods have been designed for remote data integrity verifying protocols, and these methods are dealing with the encrypted text integrity or plain text integrity. The problem of encrypted data is when performed the computing and manipulation of this encrypted data. Alternatively, data must be anonymized for enhancing privacy. Anonymization indicates to a privacy reservation technique to make the data valueless to unauthorized access except for the data owner.

Hiding, hashing, permutation, shift, substitution, and truncation are some of the traditional techniques to obscure data [2].

Modern cryptography methods depend on several subjects of mathematics, such as number theory, rings, fields and groups, information theory, probability theory, and statistics [3].

The algorithms of cryptographic are categorized into several forms: Symmetric, Asymmetric encryption, and Data integrity algorithms. Symmetric encryption is utilized for encrypting the plaintext of streams or blocks for any data size. Asymmetric encryption is utilized for encrypting small data unit, such as the values of hash function. Data integrity algorithms are utilized for preserving the data blocks from the protocols of authentication and alteration [4].

1.1. Gaussian Elimination

System of linear equations is a set of equations that are linear depending on the same set of variables. Speed and accuracy are important factors for exact solution of linear equations. There are two categories of linear equation methods: direct and indirect [5].

The two categories include several methods for solving equations, such as elimination approaches. One of these approaches is the Gaussian elimination which is a direct method [6].

Indirect approaches are basically repeated ways that are useful because they require a little number of multiplication steps against large computation operations. When the linear equation method is studied, one of the requirements to be understood are the matrices operations [7].

One of the standard methods for solving linear equations is the Gaussian elimination. The Gaussian elimination is useds to get the solution of equations, gain the determinant, concluding the rank of the coefficient matrix. The Gaussian elimination is categorised by two stages, forward and backward.

Forward stage: Unknowns are eliminated by changing the elements of equations and put up as an echelon form. The backward stage is dependent on the back substitution process by a reduced upper triangular method resulting in a solution of the equation [8].

1.2. Hash Function

The technique of Hashing utilizes a variable-length message as Input and produces a fixed-length and unique string. It is extremely hard to hold a reverse process to the original message. Where, the hashing mechanism that represents one of secure ways in cryptography uses a hash function and generates the hash values [9].

Generally, the hash functions include several steps: Firstly, divide the input message into blocks; Secondly, for the first block, calculate the hash value of a fixed size. Thirdly, obtain the hash for the second block, and add it to the previous output; Finally, repeat these steps until all blocks are calculated [10].

The hash algorithm is the algorithm that changes the text or message into a set of symbols and letters to protect it from access to the plaintext or retrieve it by the attackers of the system. The one-way property is used by hashing algorithms which makes it difficult or even impossible to retrieve the original text from the value generated by this algorithm. You can use this type of algorithm to create an electronic signature that achieves the authenticity of the data, that is, it is from a reliable source. Hash algorithms consist of a procedure that converts big data into small data or from data of variable size to data of a fixed size as shown in Figure 1. The data generated by this type of algorithm are called hash codes [11].

One-way hash algorithms is one of the most important algorithms used in cryptography where it is used to solve the problems of integrity and it is considered an alternative to the encryption algorithm authentication code for the message that does not need to have a key in the encryption process where the result is a hash value that cannot retrieve the original data [12].

2. Related Work

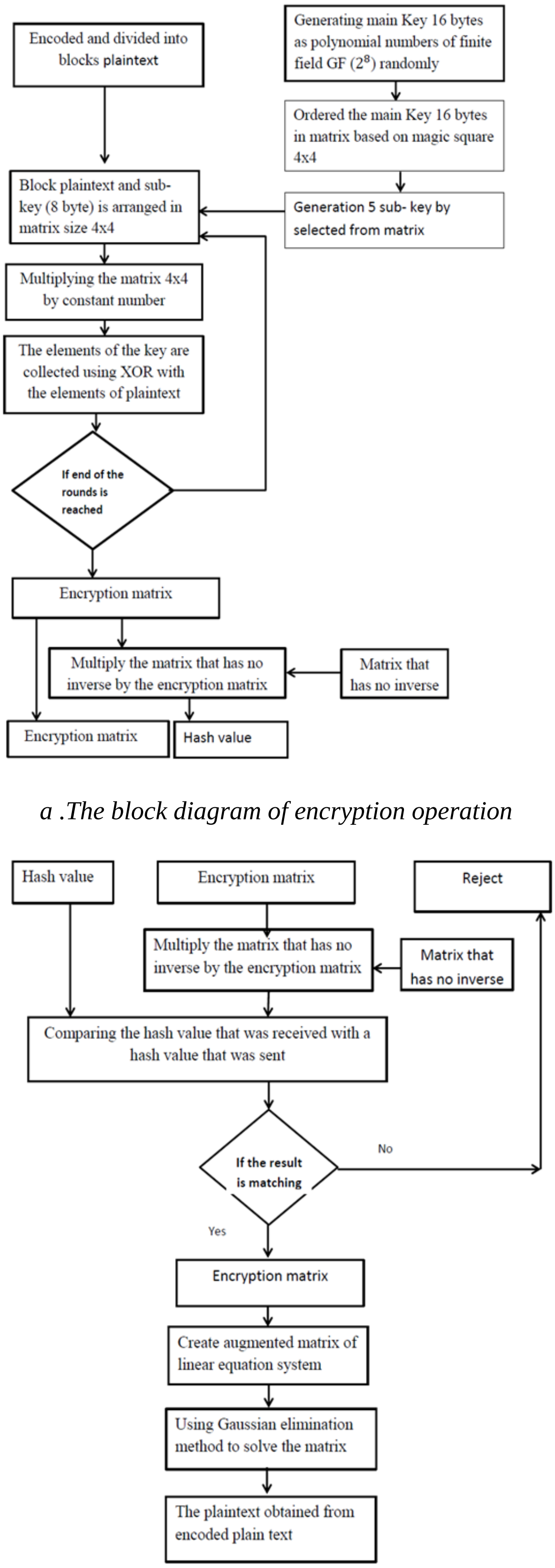
The authors (2006) proposed a work dealing with the vector space dimension of regular -diagonal magic squares. They proved the result of the even-order regular magic squares were singular. Odd-order regular magic squares that have singular and nonsingular are generated from matrix theoretic constructions [13]. The authors (2015) suggest a work that presents an approach for linear equations. The purpose of this work was analyzing several methods of elimination of linear equations and computing the efficiency of the Gauss elimination and the Gauss Jordan approach, to find their relative significance and importance in the range of symbolic and numeric computations [14].

In (2012), Nitin Pandey proposed an approach of encryption and decryption depending on the construction of magic rectangles and RSA encryption. The proposed approach built different single even magic of even ordered rectangles that don't accept division by 4, where the summation of every row and column values is equal. This developed approach works on maximizing the complexity and randomness of the cipher-text, additionally, it requires more time for the implementation process [15]. In the same year, M.A.Noaman uses a hash function (MD5) by generating an authentication key form as an authentication code (MAC). The aims of the performance of this algorithm is to provide privacy in the application to execute the MD5 for the data integrity checking method and to make sure of the valid state of the data in the information system [16]. E. Noroozi, S.Daud and A. Sabouhi (2013) this paper presents a new method in generating small-sized digital signature by using a hash algorithm, where the reducing 4% of the size of the original file in plaintext mean 1600 bytes. This method contributes to enhancing the verification of the signature and improving all the speed and time of the operation [17].

3. Proposed Methodology

The proposed cryptographic system contributes to improving the privacy and data integrity received from trusted authorities using an encryption method based on the linear equation system in addition to creating a method for digital signature based on the hash function to ensure that data is not changed or modified, which provides additional security features.

The proposed system includes two operations; the first is the plain-text encryption process. The second process is sending the encrypted text associated with the digital signature based on the hash function. Figure 1, illustrates the encryption and decryption operations of the proposed system.


Encoded and divided into blocks plaintext
Generating main Key 16 bytes as polynomial numbers of finite field GF (2^8) randomly
Ordered the main Key 16 bytes in matrix based on magic square 4x4
Generation 5 sub- key by selected from matrix
Block plaintext and sub-key (8 byte) is arranged in matrix size 4x4
Multiplying the matrix 4x4 by constant number
The elements of the key are collected using XOR with the elements of plaintext
If end of the rounds is reached
Encryption matrix
Multiply the matrix that has no inverse by the encryption matrix
Matrix that has no inverse
Encryption matrix
Hash value

a .The block diagram of encryption operation

Hash value
Encryption matrix
Reject
Multiply the matrix that has no inverse by the encryption matrix
Matrix that has no inverse
Comparing the hash value that was received with a hash value that was sent
If the result is matching
No
Yes
Encryption matrix
Create augmented matrix of linear equation system
Using Gaussian elimination method to solve the matrix
The plaintext obtained from encoded plain text

b. the block diagram of decryption operation

Figure 1. The encryption and decryption operations of the proposed system.

The implementation of the proposed system is executed in two phases as the following:

3.1. Encryption Phase

Before entering the encryption process, the plain text is encoded and divided into blocks, each block has a size of 8 bytes. The encoding process is done by encoding all characters using the ASCII code for plain text between 1 and 255 bytes where each character represents a byte of the polynomial number as a finite field of GF (2^8). Figure 2. illustrates the encoding process of a sample of the characters.

plaintext	***p1***	***p2***	***p3***	***p4***	***p5***	***p5***	***p6***	***p7***	***p8***
ASCII COD	**P1**	**P2**	**P3**	**P4**	**P5**	**P5**	**P6**	**P7**	**P8**

Figure 2. The encoding sample of the characters

The characters are arranged and numbered in a single array.

Then, the steps of the encryption process are performed in five-round where each round contents the following steps:

Step 1: The encoded block (in the first round) or previous encryption block and sub-key element (8 bytes) are arranged in matrix $[\boldsymbol{L_i}]$ size 4x4. Figure 2. shows the matrix including elements of encoded plain text and sub-key elements.

$$[\boldsymbol{L_i}] = \begin{array}{|c|c|c|c|}\hline K1 & K2 & K3 & P1 \\ \hline K4 & K5 & K6 & P2 \\ \hline K7 & K8 & P3 & P4 \\ \hline P5 & P6 & P7 & P8 \\ \hline\end{array}$$

Step 2: multiplying the matrix (L_i) by constant number 245 based on Equation 1. using an irreducible polynomial $(\boldsymbol{x^8+x^4+x^3+x+1})$ .The results represent new elements of matrix $(\boldsymbol{C_i})$ 4 × 4 shown in Figure 3.

$$\boldsymbol{C_i= (L_i \,.\, x)\ Mod\ M} \qquad (1)$$

Where:
$\boldsymbol{C_i}$ = the elements of new encoded matrix 4 × 4.
$\boldsymbol{L_i}$ = the elements of encoded matrix 4 × 4.
$\boldsymbol{x}$ = constant number (245).
$\boldsymbol{M}$ = irreducible polynomial $(\boldsymbol{x^8+x^4+x^3+x+1})$.

$$[\boldsymbol{C_i}] = \begin{array}{|c|c|c|c|}\hline Y1 & Y2 & Y3 & B1 \\ \hline Y4 & Y5 & Y6 & B2 \\ \hline Y7 & Y8 & B3 & B4 \\ \hline B5 & B6 & B7 & B8 \\ \hline\end{array}$$

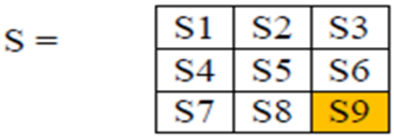
Figure 3. Shows the new matrix of plain text and key

Step 3: The matrix (L_i) 4 × 4 includes 8 bytes of encoded plaintext and 8 bytes of the sub-key (that significant from the main key of size 40 bytes had been selected randomly from 1-255). To be performed the encryption process, must the elements of the sub-key are collected using XOR with the elements of plaintext for each row and column of the matrix in the form of linear equations. So, the following equations will be calculated based on performed experiments, that only are used (first, second, third and fourth) rows and (first, second and the fourth) columns to be summed up .Also, the main diagonals are used in addition to rows and columns.

Y1+ Y2+ Y 3+B1 =S1 [2]
Y4+ Y5+ Y6+B2 =S2 [3]
Y7+ Y8+B3+B4 = S3 [4]
B5+B6+BP7+B8 = S4 [5]
Y1+ Y4+ Y7+B5 =S5 [6]
Y2+ Y5+Y8+B6 = S6 [7]
B1+B2+B4+B8 =S7 [8]
Y1+ Y5+B3+B8 =S8 [9]

The summations of the 2-9 equations represent the results of the 8 bytes of cipher text [S1, S2, S3, S4, S5,S6,S7,S8] .

Step 3: The previous steps repeat five rounds, the output of each step being input for the next step. So, the output of the last round is arranged in matrix (S) 3x3. The last element (S9) is added to the matrix as a salt to fill in the matrix as shown in Figure 4.:

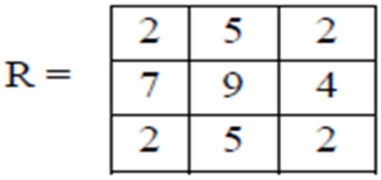


S =

S1	S2	S3
S4	S5	S6
S7	S8	S9

Figure 4. Shows the encrypted matrix of the last round

Step 4: in this step, any matrix that has no inverse is chosen to generating a one-way hash algorithm. Figure 5, shows the matrix has no inverse because the determinant of this matrix is equal to 0.



R =

2	5	2
7	9	4
2	5	2

Figure 5. Shows the matrix has no inverse

So we multiply the matrix that has no inverse by the encryption matrix and the remainder of the division is a hash value.

The sender of the message calculates the hash value of the encryption text using the equation (10) that all operation are using polynomial operation and irreducible polynomial $(\boldsymbol{x^8 + x^4 + x^3 + x + 1})$ in the following.

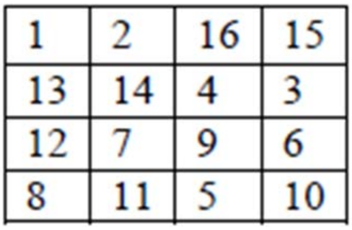
$$\mathbf{[V] = [S]. [R]} \qquad \mathbf{(10)}$$

Where:

[V] = the matrix of hash value
[S] = the matrix of encrypted data
[R] = matrix has no inverse
Then, hash value [V] and Encryption matrix [S] have been gathered and sent to the receiver.

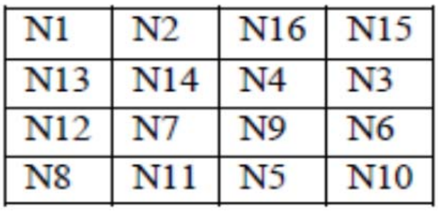
3.2. Keys Generation

This system creates of 16-bytes main key N_i by one of the random number generation algorithms as non-zero elements. Any byte is a set in the finite field GF (2^8). Then, it is ordered in the matrix in size 4×4 bytes based on the form of the magic square fourth order 4×4 is shown in Figure 6. Figure 7. show the order of key values in the matrix (4×4).



1	2	16	15
13	14	4	3
12	7	9	6
8	11	5	10

Figure 6. Magic square fourth order 4×4



N1	N2	N16	N15
N13	N14	N4	N3
N12	N7	N9	N6
N8	N11	N5	N10

Figure 7. Show the order of key values in the matrix 4×4

Each round has 8 bytes of different Sub-Key that derived from the main key; where the sub-key of each round is assigned of the one row and one column of elements except the fifth round which is assigned the two diagonals of the elements. The sub-keys of the five rounds are calculated in the following.

1. Sub-Key of first round = N1, N2, N16, N15, N1, N13, N12, N8.
2. Sub-Key of second round = N13, N14, N4, N3, N2, N14, N7, N11.
3. Sub-Key of third round = N12, N7, N9, N6, N16, N4, N9, N5.
4. Sub-Key of fourth round= N8 + N11 +N5 + N10, N15, N3, N6, N16.
5. Sub-Key of fifth round = N1, N14, N9, N10, N15, N4, N7, N8.

The Encryption algorithm of proposed method is:

Input : block (8 bytes) of encoded original data **Output : block (9 bytes) of cipher text**	
Step 1	**The encoded block and sub-key element are arranged in matrix** [$\boldsymbol{L_i}$] size 4x4. [$\boldsymbol{L_i}$]=$\boldsymbol{K_j}$, [$\boldsymbol{L_i}$]=$\boldsymbol{P_j}$ **where:** **i=(1,2…16) ,** $\boldsymbol{j}$ **= (1,2….8)**
Step 2	multiplying the matrix (L_i) by constant number 245 $\boldsymbol{C_i}$**= (**$\boldsymbol{L_i}$ **. x) Mod M**
Step 3	the sub-key elements and the elements of plaintext of each row and column of the matrix[$\boldsymbol{C_i}$] are collected using XOR in the form of linear equations: Y1+ Y2+ Y 3+B1 =S1 [2] Y4+ Y5+ Y6+B2 =S2 [3] Y7+ Y8+B3+B4 = S3 [4] B5+B6+BP7+B8 = S4 [5] Y1+ Y4+ Y7+B5 =S5 [6] Y2+ Y5+Y8+B6 = S6 [7] B1+B2+B4+B8 =S7 [8] Y1+ Y5+B3+B8 =S8
Step 4	repeat the three steps above five-times
Step 5	the output of the last round is arranged in matrix [S] 3x3 [S_i]= {S1,S2,S3,S4,S5,S6,S7,S8,S9} , where : S9 = is a salt
Step 6	Generating a one-way hash algorithm by chosen not inverse matrix[R] and multiplying by the encryption matrix [S]. [V] =[S]. [R]

3.3. Decryption Phase

The decryption phase is performed by reversing the steps of the encryption phase. The last round of decryption phase is illustrated as the following.

Step 1: Upon receiving the encrypted message and the hash value by the recipient, the receiver ensures that the message is integrated and has not been changed. The same procedures of equation (10) are used in the calculation of the hash value performed on the sender side and comparing it with the hash value that is received.

$$[V] = [S] . [R] \qquad (10)$$

Then the recipient compares the hash value that was received with a hash value that was sent. If the result has corresponded, the encrypted message is valid and did not expose to change and update.

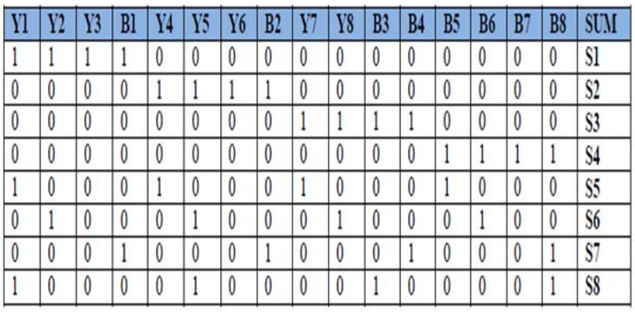
Step 2: Before the decryption process, we need to know the last values ($\boldsymbol{Y_i}$) of the sub-key by using the following equation 1:

$$\boldsymbol{Y_i = (K_i \ . P) \ \ \text{mod M}} \qquad \textbf{(1)}$$

Where:

Y_i = last values of sub-key
K_i= last original sub-key
P= the constant number
M = irreducible polynomial ($x^8 + x^4 + x^3 + x + 1$).

Then, create Augmented matrix of linear equation system of encryption matrix [S] based on equations 2,….,9 and arrange the summation column of the Augmented matrix based on the elements of encryption matrix [S] as follows:



Y1	Y2	Y3	B1	Y4	Y5	Y6	B2	Y7	Y8	B3	B4	B5	B6	B7	B8	SUM
1	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	S1
0	0	0	0	1	1	1	1	0	0	0	0	0	0	0	0	S2
0	0	0	0	0	0	0	0	1	1	1	1	0	0	0	0	S3
0	0	0	0	0	0	0	0	0	0	0	0	1	1	1	1	S4
1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	S5
0	1	0	0	0	1	0	0	0	1	0	0	0	1	0	0	S6
0	0	0	1	0	0	0	1	0	0	0	1	0	0	0	1	S7
1	0	0	0	0	1	0	0	0	0	1	0	0	0	0	1	S8

Step 3: removing Key columns {1, 2, 3, 4, 5, 6, 7, and 8} and rearranging of summation column as follows:

B1	B2	B3	B4	B5	B6	B7	B8	SUM
1	0	0	0	0	0	0	0	**S1**
0	**1**	0	0	0	0	0	0	**S2**
0	0	**1**	**1**	0	0	0	0	**S3**
0	0	0	0	**1**	**1**	**1**	**1**	**S4**
0	0	0	0	**1**	0	0	0	**S5**
0	0	0	0	0	**1**	0	0	**S6**
1	**1**	0	**1**	0	0	0	**1**	**S7**
0	0	**1**	0	0	0	0	**1**	**S8**

Step 4: using the Gaussian elimination method to solve the matrix and depending on the operation of the finite field.

B1	B2	B3	B4	B5	B6	B7	B8	
1	0	0	0	0	0	0	0	**B1**
0	**1**	0	0	0	0	0	0	**B2**
0	0	**1**	0	0	0	0	0	**B3**
0	0	0	**1**	0	0	0	0	**B4**
0	0	0	0	**1**	0	0	0	**B5**
0	0	0	0	0	**1**	0	0	**B6**
0	0	0	0	0	0	**1**	0	**B7**
0	0	0	0	0	0	0	**1**	**B8**

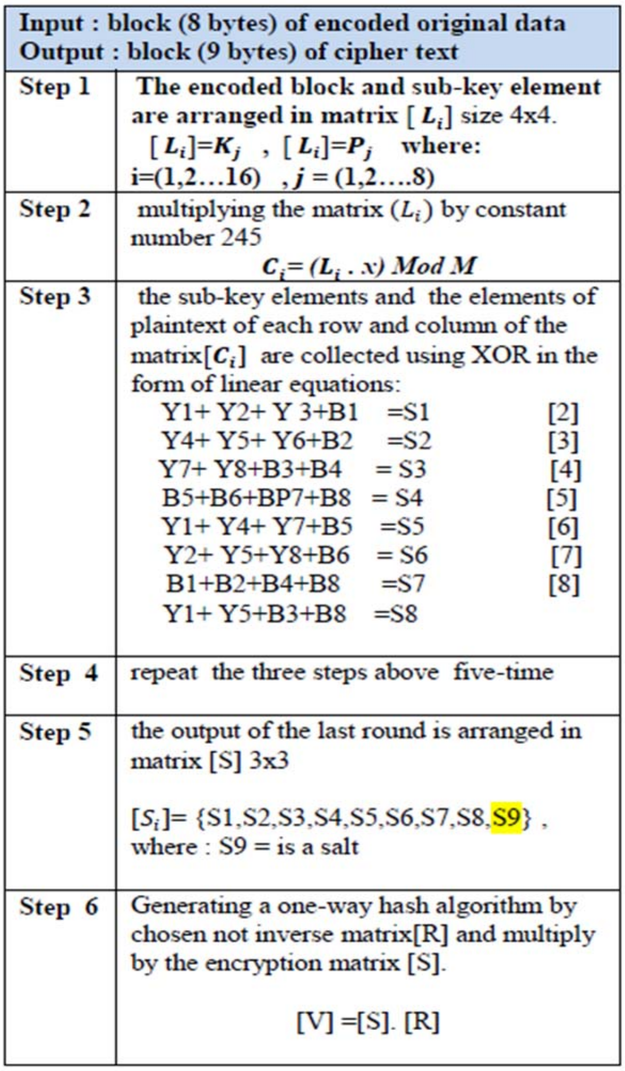
Then, the encryption block of the next round can be obtained from the equation (12) as follow:

$$\boldsymbol{P_i = B_i \ \ . x^{-1} \ \ \text{Mod m}} \qquad \textbf{(12)}$$

Where
$\boldsymbol{P_i}$= encryption block ($\boldsymbol{S_i}$) of next round
$\boldsymbol{B_i}$= new encoded plain text
$\boldsymbol{x^{-1}}$= **reverse the constant number**
m = irreducible **polynomial** ($\boldsymbol{x^8 + x^4 + x^3 + x + 1}$**).**

Finally, the encoded plaintext of the last round is obtained by the same decryption way of the previous rounds and then, decoded to gain plaintext by using the ASCII code table.

The Decryption algorithm of proposed method is:



Input : block (8 bytes) of encoded original data Output : block (9 bytes) of cipher text	
Step 1	The encoded block and sub-key element are arranged in matrix [L_i] size 4x4. [L_i]=K_j , [L_i]=P_j where: i=(1,2…16) ,j = (1,2….8)
Step 2	multiplying the matrix (L_i) by constant number 245 $C_i = (L_i \, . \, x) \; Mod \; M$
Step 3	the sub-key elements and the elements of plaintext of each row and column of the matrix[C_i] are collected using XOR in the form of linear equations: Y1+ Y2+ Y 3+B1 =S1 [2] Y4+ Y5+ Y6+B2 =S2 [3] Y7+ Y8+B3+B4 = S3 [4] B5+B6+BP7+B8 = S4 [5] Y1+ Y4+ Y7+B5 =S5 [6] Y2+ Y5+Y8+B6 = S6 [7] B1+B2+B4+B8 =S7 [8] Y1+ Y5+B3+B8 =S8
Step 4	repeat the three steps above five-time
Step 5	the output of the last round is arranged in matrix [S] 3x3 [S_i]= {S1,S2,S3,S4,S5,S6,S7,S8,S9} , where : S9 = is a salt
Step 6	Generating a one-way hash algorithm by chosen not inverse matrix[R] and multiply by the encryption matrix [S]. [V] =[S]. [R]

4. Results and Analysis

The proposed system aims to implement a new encryption method and achieve the integrity and confidentiality of sensitive data that is sent away, which is tested through various techniques related to the encryption data to prove the effectiveness of the proposed algorithm

The measures of system results such as the Time criteria, Randomness test and Avalanche effect test are illustrated in the following.

Various file sizes of encrypted data and original data were used for testing. Where a comparison algorithms DES, original AES (Rijandar) with the proposed system was performed by testing the execution time of the algorithms on the same device that uses to the proposed system. Table 1. and 2. show the comparison between the other algorithms of execution times (in seconds) with various samples size in (Kb). The execution time is used to computing the throughput of the encrypted and decryption algorithms.

Table 1. The Comparison of encryption throughput (Kb/sec)

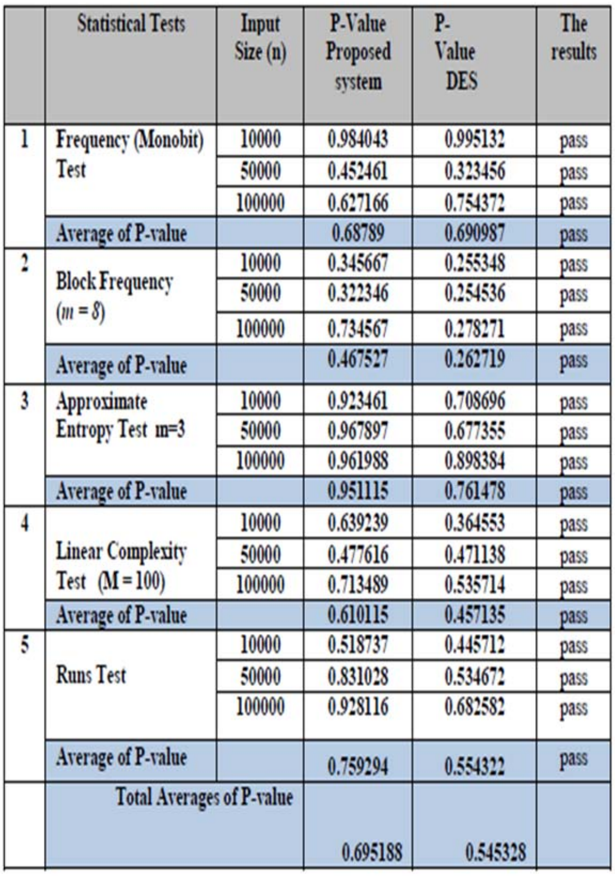
File Size (Kb)	DES 56	AES (Rijandar)	proposed system
255	1.612	0.973	0.710
2802.01	8.239	6.953	4.139
6591.52	16.048	13.281	10.613
9963.97	22.177	18.753	15.113
∑ 19612.5	48.076	39.960	30.575
Average 4903.125	12.019	9.99	7.64375
Encryption throughput (Kb /sec)	407.948	490.804	641.456

Table 2. The Comparative of decryption throughput (Kb/sec)

File Size (Kb)	DES 56	AES (Rijandar)	proposed system
255	1.946	1.365	0.953
2802.01	8.743	7.371	4.893
6591.52	16.487	13.642	12.625
9963.97	22.598	19.189	17.258
∑ 19612.5	49.774	41.567	35.729
Average 4903.125	12.443	10.391	8.93225
decryption throughput (Kb /sec)	394.046	471.862	548.924

In the tables above it can be noticed that the proposed system requires less time than DES, AES algorithm in encryption and decryption operations. Because the number of rounds in proposed system are less than the numbers rounds in the other algorithms. The randomness tests are very important to know if there is a deviation between plaintext/cipher text bits and define the degree of random form for the cipher text. The proposed system results are shown an accepted and suitable degree according to the NIST randomness test. Where a set of experimental results was tested for all parts of the system displayed in the following Table 3.

Table 3. The NIST Statistical tests for encrypted results



	Statistical Tests	Input Size (n)	P-Value Proposed system	P-Value DES	The results
1	Frequency (Monobit) Test	10000	0.984043	0.995132	pass
		50000	0.452461	0.323456	pass
		100000	0.627166	0.754372	pass
	Average of P-value		0.68789	0.690987	pass
2	Block Frequency (*m* = 8)	10000	0.345667	0.255348	pass
		50000	0.322346	0.254536	pass
		100000	0.734567	0.278271	pass
	Average of P-value		0.467527	0.262719	pass
3	Approximate Entropy Test m=3	10000	0.923461	0.708696	pass
		50000	0.967897	0.677355	pass
		100000	0.961988	0.898384	pass
	Average of P-value		0.951115	0.761478	pass
4	Linear Complexity Test (M = 100)	10000	0.639239	0.364553	pass
		50000	0.477616	0.471138	pass
		100000	0.713489	0.535714	pass
	Average of P-value		0.610115	0.457135	pass
5	Runs Test	10000	0.518737	0.445712	pass
		50000	0.831028	0.534672	pass
		100000	0.928116	0.682582	pass
	Average of P-value		0.759294	0.554322	pass
	Total Averages of P-value		0.695188	0.545328	

Where the randomness test results of the proposed system are acceptable and have high averages of P-values, where a P-value for a test is equal to one, and then the value will be ideal randomness. A P-value of zero refers to that the value is completely non-random.

The Avalanche Effect test is an important test of the cipher algorithm. This test is based on a little change in the plain text, and then we calculate the outputs of the bits in the cipher text.

In other words, any change in one bit of the plain text or key must be a large change in the bits of the cipher text. Avalanche effect is calculated by using the following equation:

$$\text{AE} = \frac{\text{Number of Changed Bits in Cipher text}}{\text{Total number of bits in cipher text}} * 100$$

To evaluate the proposed system on a variety of bits that change by one-bit in plaintext with remains the key bits constant or change one-bit of the key bits with remains plaintext bits constant. Table 4. shows the test of two samples that represent the outputs of the proposed system.

No.	Keys	input	Output	AE	AE Test %
1	32,161,24, 111,255,158,62,21	65,98,115 ,116,114, 97,99,116	99,105,112,104,101 ,60,182, 238,87	0.55	55%
	32,161,24, 111,255,158,126,21	65,98,115 ,116,114, 97,99,116	31,24,13, 54,77,39, 74,40,14		
2	1,107,225, 60,182,238 ,87,227	71,97,117 ,115,115, 97,110	60,182,238,87,227, 239,174, 24,161	0.51	51%
	1,107,225, 60,182,238 ,87,227	71,97,117 ,115,77,97,110	97,136,242,55,187, 209,146, 106,157		
Main percentage avalanche value				0.53	53%

$$\textbf{\textit{AE}}\ \textbf{Sample 1} = \frac{\mathbf{5+4+6+5+2+4+6+4+4}}{\mathbf{72}} = \frac{\mathbf{40}}{\mathbf{72}} = \mathbf{0.55}$$

$$\textbf{\textit{AE}}\ \textbf{Sample 2} = \frac{\mathbf{5+5+3+2+3+5+4+6+4}}{\mathbf{72}} = \frac{\mathbf{37}}{\mathbf{72}} = \mathbf{0.51}$$

The XOR operation is calculated between the two results of each sample to getting the summation of the numbers of different bits between these two results. The percentage average of the Avalanche value of the proposed system is approximately 53% of the different bits of cipher text. Where the acceptable ratio is 50%. So our algorithm has good confusion and diffusion between the cipher text and plaintext and also between the key and the cipher text.

5. Conclusion

In this paper, a proposed system is designed to protect the confidentiality and integrity of data from penetration, disclosure, and destruction. The proposed system provides increasing of the substitution, permutation techniques in the encryption stage and increasing the difficulty of breaking the cipher text and gives higher encryption and decryption throughput than DES, AES algorithm . The proposed system is achieved by improving the encryption approach and digital signature method based on improving the one-way hash algorithm by using the idea of the magic square and the linear equation system in encryption and decryption stages. In addition, the initial hash value was created by the inverse matrix for ensuring that encrypted data is not changed or modified.It is recommended for the future to improve the research by increasing the size magic square 8x8 in the encryption stage in order to accommodate a larger block size of plain text and a longer key length.

References

[1].Yun, A., Shi, C., & Kim, Y. (2009, November). On protecting integrity and confidentiality of cryptographic file system for outsourced storage. In *Proceedings of the 2009 ACM workshop on Cloud computing security* (pp. 67-76).

[2].Tikore, S. V., Pradeep, K. D., & Prakash, B. D. (2015). Ensuring the data integrity and confidentiality in cloud storage using hash function and TPA. *International Journal on Recent and Innovation Trends in Computing and Communication*, *3*(5), 2736-2740.

[3].Perlman, R., Kaufman, C., & Speciner, M. (2016). *Network security: private communication in a public world*. Pearson Education India.

[4]. Stallings, W. (2012). Cryptography and network security, principles and practice (international edition). International Journal of Engineering & Computer Science, 1(01), 121-136.

[5].Hsu, C. W., & Lin, C. J. (2002). A comparison of methods for multiclass support vector machines. *IEEE transactions on Neural Networks*, *13*(2), 415-425.

[6].Issa, R. I. (1986). Solution of the implicitly discretised fluid flow equations by operator-splitting. *Journal of computational physics*, *62*(1), 40-65.

[7].Grcar, J. F. (2011). How ordinary elimination became Gaussian elimination. *Historia Mathematica*, *38*(2), 163-218.

[8].Winkler, F. (2012). *Polynomial algorithms in computer algebra*. Springer Science & Business Media.

[9].P. V. Rao, S. G. Rao, P. C. Reddy, G. R. Sakthidharan, and Y. M. Kumar,(2019). Improve the integrity of data using hashing algorithms.*Int. J. Innov. Technol. Explor. Eng., 8*(7), 2018–2023.

[10]. European Data Protection Supervisor (EDPS). (2019). Introduction to the hash function as a personal data pseudonymisation technique. Retrieved from: https://edps.europa.eu/sites/edp/files/publication/19-10-30_aepd-edps_paper_hash_final_en.pdf [accessed: 10 March 2020].

[11]. Zheng, Y., Pieprzyk, J., & Seberry, J. (1992, December). HAVAL—a one-way hashing algorithm with variable length of output. In *International workshop on the theory and application of cryptographic techniques* (pp. 81-104). Springer, Berlin, Heidelberg.

[12]. Mittmann, J. (2005). One-Way Encryption and Message Authentication. Retrieved from: http://wwwmayr.in.tum.de/konferenzen/Jass05/courses/1/papers/mittmann_paper.pdf [accessed: 20 January 2020].

[13]. Lee, M., Love, E., & Wascher, E. (2006). Linear Algebra of Magic Squares. *Undergraduate Research, Central Michigan University, Mount Pleasant, Mich, USA*.

[14]. Gharib, S., Ali, S. R., Khan, R., & Munir, N. (2015). System of Linear Equations, Guassian Elimination. *Global Journal of Computer Science and Technology*.

[15]. Pandey, N. (2012). Secure Communication Scheme with Magic Square. *Journal of Global Research in Computer Science*, *3*(12), 12-14.

[16]. Noaman, M. A. (2013). A VHDL model for implementation of MD5 hash algorithm. *Engineering and Technology Journal*, *31*(6 Part (A) Engineering), 1107-1116.

[17]. Noroozi, E., Daud, S. M., & Sabouhi, A. (2013). Secure digital signature schemes based on hash functions. *International Journal of Innovative Technology and Exploring Engineering (IJITEE)*, *2*(4), 321-325.