Encryption of DES Algorithm in Information Security

[1] Isnar Sumartono, [2] Andysah Putera Utama Siahaan
Faculty of Science and Technology, Universitas Pembangunan Panca Budi, Medan, Indonesia
Email: [1] isnar@pancabudi.ac.id, [2] andiesiahaan@gmail.com

Abstract: *Information security is the protection of personal and non-personal data from various threats to guarantee privacy. For business practices, data security can reduce business risk, and increase the return of investment and business opportunities. In designing information system security systems, there are information security aspects that need to be considered. Many threats will come before the information circulating. Information is a matter that will be targeted by wild parties. Cryptographic algorithms are needed to protect data from these threats. Data Encryption Standard (DES) belongs to the symmetry cryptography system and is classified as a block cipher type. DES operates on 64-bit block size. DES encrypts 64 plaintext bits into 64-bit ciphertext using 56 private key bits or subkeys. The internal key is generated from an external key that is 64 bits long. The DES method is an excellent cryptographic technique used to secure data. DES has 16 rounds to ensure safer data against unexpected attacks. Applying DES to data encryption will be very useful for protecting data.*

Keywords: *DES, encryption, decryption, algorithm.*

1. INTRODUCTION:

The digital age is an era where data is exchanged through bits on the internet network [1]–[3]. It causes security of the data is not guaranteed. Data can be intercepted during the shipping process. Networks that have been connected to a global network are networks that can be dismantled by anyone [4]. Information is one of the essential assets to be protected by its security. Users need to pay attention to the security of the information assets. Leakage of information and failure of the system can result in losses for users of that information. Data leakage can occur due to intentional or accidental [5]. It will be detrimental to both the financial side and one's productivity [6]. Data security is quality or state of being secure-to be free from danger. Some things that need to be considered to determine the security strategy are as follows:

- Physical Security, a strategy that focuses on securing organizational members, physical assets, unauthorized access and workplaces from various threats including fire hazards
- Personal Security, a more focused strategy to protect people in the organization
- Operation Security, a strategy to secure the ability of an organization or company to work without threats.
- Communications Security, a strategy that aims to secure information and information technology media.
- Network Security, a strategy that focuses on securing network equipment on organizational data.

Data security is a technique used to protect information from threats that might occur to guarantee information exchange [7]. It aims to reduce the level of risk and accelerate or maximize decision making. The level of security in information also depends on the level of sensitivity of the information in the database, information that is not too sensitive to the security system is not too tight, while for sensitive information, it is necessary to have a strict security level setting for access to that information [2], [8]–[11].

Cryptography is a technique used to protect data. Many techniques can be used to perform cryptographic processes. Cryptography involves the process of encryption and decryption [12]. The Data Encryption Standard (DES) algorithm is one technique that can be done to secure data [13]. This algorithm has a 64-bit encryption block. The key used is also 64-bit. The block length depends on the number of the plaintext used. This algorithm has a somewhat complicated process and stages. There is no complicated mathematical calculation process, but this algorithm has a substitution process that is complicated in several tables. By applying DES encryption, it is expected that the information sent will be safer and more secure [14]–[17].

2. THEORIES:

2.1 Substitution Cipher

Substitution ciphers are ciphers by substituting letters with other letters as specified. The main principle of substitution ciphers is to exchange each letter for plaintext with something [8], [18]–[20]. Substitution ciphers include classical cryptographic algorithms. The idea is to replace one or more letters in plaintext with one or more letters in plaintext with specific rules. These rules depend on the process of encryption and decryption. Substitution ciphers have several variants or types [21]. Types of substitution ciphers are:

- *Monoalphabetic Cipher*. This type of substitution cipher is often also called a simple substitution cipher. The idea of single-alphabet substitution ciphers is to replace one character in plaintext into one character in the ciphertext with specific rules. The ciphering function is one to one function.
- *Homophonic Substitution Cipher Cipher*. The idea of a homophonic substitution cipher is to replace one character in plaintext into one or more characters in the ciphertext. The ciphering function is one to many functions.
- *Alphalphabetic Substitution Cipher*. This type of substitution cipher can be called a double substitution cipher. Multiple-alphabet substitution ciphers are single-alphabet substitution ciphers that use different keys. Therefore, compound-substitution ciphers have periods m, m is the key length.
- *Polygram Substitution Cipher*. The idea of a polygram substitution cipher is to replace a character block with a ciphertext block. Blocks consist of one or more characters. For example, AAA was changed to BCD or PAP, and others.

2.2 Data Encryption Standard (DES)

Data Encryption Standard (DES) is a block cipher algorithm that is popular because it is used as a standard key-symmetry encryption algorithm, although at this time the standard has been replaced by a new algorithm, AES because DES is considered unsafe again. DES is the name of the symmetry encryption standard, the name of the encryption algorithm itself is DEA (Data Encryption Algorithm), but the DES name is more popular than DEA. The DES algorithm was developed at IBM under the leadership of W.L. Tuchman in 1972. This algorithm is based on the Lucifer algorithm created by Horst Feistel. This algorithm has been approved by the National Bureau of Standard (NBS) after its strength assessment by the United States National Security Agency (NSA) [22]–[24]

3. METHODOLOGY:

The DES algorithm requires several substitution tables. Tables are substitution, compression, expansion, inversion, and breakdown. These tables are used when performing encryption stages. The following are the tables used to perform DES encryption [25].

Table 1. Initial Permutation

58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

Table 2. Permutation Compression 1

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	45	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

Table 3. Left Shift

Round	Number of Shifts (Left Shift)
1	1
2	1
3	2

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

Table 4. Permutation Compression 2

14	17	11	24	1	5
3	28	15	6	21	10
23	19	12	4	26	8
16	7	27	20	13	2
41	52	31	37	47	55
30	40	51	45	33	48
44	49	39	56	34	53
46	42	50	36	29	32

Table 5. Permutation Compression 2

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

Table 6. S-Box

S1																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
01	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
10	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
11	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
S2																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
01	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
10	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
11	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
S3																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
01	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
10	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
11	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
S4																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**

00	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
01	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
10	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
11	3	15	0	6	10	1	13	18	9	4	5	11	12	7	2	14
S5																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
01	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	15
10	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
11	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3
S6																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
01	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
10	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
11	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13
S7																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
01	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
10	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
11	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12
S8																
	0000	**0001**	**0010**	**0011**	**0100**	**0101**	**0110**	**0111**	**1000**	**1001**	**1010**	**1011**	**1100**	**1101**	**1110**	**1111**
00	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
01	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
10	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
11	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

Table 7. Permutation

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

Table 8. Inverse Permutation

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

2. RESULT AND DISCUSSION

This section discusses testing the DES algorithm. The test is done by giving plaintext and keys each of eight characters. The following is a step illustration of the DES algorithm.

Step 1

==========

Plaintext =

	C	**O**	**M**	**P**	**U**	**T**	**E**	**R**
DEC	67	79	77	80	85	84	69	82
HEX	43	4F	4D	50	55	54	45	52

Key =

HEX	13	34	57	79	9B	BC	DF	F1

Step 2

==========

Plain Bit = 0100001101001111010011010101000001010101010101000100010101010010
= LR is the PLAIN BIT value based on the order in the Permutation Table
LR = 1111111110111000011101100101011100000000000000000000011010000011
= Separate LR into 2 parts L and R
L[0] = 11111111101110000111011001010111
R[0] = 00000000000000000000011010000011

Step 3: Generate Key

==========

Key Bit = 0001001100110100010101110111100110011011101111001101111111110001
= CD is the KEY BIT value based on the order in the PC1 Table
CD = 1111000011001100101010101111010101010110011001111000111 1
= Separate CD into 2 parts C and D
C[0] = 11110000110011001010101011 1
D[0] = 01010101011001100111100011 1

Step 4: Slide bits C and D according to the sliding distance in the LeftShift Table

==========

C[0]	= 11110000110011001010101011 1	D[0]	= 01010101011001100111100011 1
C[1]	= 11100001100110010101010111 1	D[1]	= 10101010110011001111000111 10
C[2]	= 11000011001100101010101111 1	D[2]	= 01010101100110011110001111 01
C[3]	= 00001100110010101010111111 1	D[3]	= 01010110011001111000111101 01
C[4]	= 00110011001010101011111111 00	D[4]	= 01011001100111100011110101 01
C[5]	= 11001100101010101111111100 00	D[5]	= 01100110011110001111010101 01
C[6]	= 00110010101010111111111000 11	D[6]	= 10011001111000111101010101 01
C[7]	= 11001010101011111111000011 00	D[7]	= 01100111100011110101010110 01
C[8]	= 00101010101111111110000110 11	D[8]	= 10011110001111010101011001 01
C[9]	= 01010101011111111100001100 110	D[9]	= 00111100011110101010110011 01
C[10]	= 01010101111111100001100 11001	D[10]	= 11110001111010101011001100 01
C[11]	= 01010111111110000110011001 01	D[11]	= 11000111101010101100110011 01
C[12]	= 01011111110000110011001 0101	D[12]	= 00011110101010110011001111 01
C[13]	= 01111111100001100110010101 01	D[13]	= 01111010101011001100111100 01
C[14]	= 11111110000110011001010101 01	D[14]	= 11101010101100110011110001 01
C[15]	= 11111000011001100101010111 01	D[15]	= 10101010110011001111000111 01
C[16]	= 11110000110011001010101111 1	D[16]	= 01010101011001100111100011 11

Step 5: K is a CD value based on the order in the PC2 table (64 to 48 bits)

==========

K[0] = 1100101100111101100010110000111000010111111110101
K[1] = 0001101100000010111011111111110001110000011100 10
K[2] = 0111100110101110110110011101101111001001111001 01
K[3] = 0101010111111100100010100100001011001111100110 01
K[4] = 0111001010101101110101101101101100110101001100 11 01
K[5] = 0111110011101100000001111110101101010011101010 00
K[6] = 0110001110100101001111100101000001111011001011 11
K[7] = 1110110010000100101101111111101100001100010111 100
K[8] = 1111011110001010001110101100000100111011111111 011

K[9] = 111000001101101111101011111011011110011110000001
K[10] = 101100011111001101000111101110100100011001001111
K[11] = 001000010101111111010011110111101101001110000110
K[12] = 011101010111000111110101100101000110011111101001
K[13] = 100101111100010111010001111110101011101001000001
K[14] = 010111110100001110110111111100101110011100111010
K[15] = 101111111001000110001101001111010011111100001010
K[16] = 110010110011110110001011000011100001011111110101

Step 6: Expansion

==========

===

= ER is the R value based on the order in the Expansion Table (32 to 48 bits)
= B is a meeting between rows (bits 1, 6) and columns (bits 2 to 5) in the S-BOX Table
= B produces 32 bits

The S-Box is a vital part of DES because it is the most challenging part to solve. It is because the S-Box is the only part of DES whose computation is not linear. Meanwhile, the design of the S-Box itself was not made public. Because of this, many suspect that the S-Box was designed in such a way as to provide a trapdoor.

Iteration 1

==========

= L is the previous R
L[0] = 111111111011100001110110010101 11
= R is PB xor previous L
R[0] = 00000000000000000000011010000011
ER[0] = 100000000000000000000000000000011010100000000110
K[1] = 000110110000001011101111111111100011100000111001 0
= LA is an XOR function on ER against K
= ---------- xor
A[1] = 100110110000001011101111111111100101001000111010 0
B[1] = 100001010100100000110010111010 10
= PB is B based on substitution in the Permutation table
PB[1] = 001010001011001101000100110100 01

Iteration 2

==========

= L is the previous R
L[1] = 00000000000000000000011010000011
= R is PB xor previous L
R[1] = 11010111000010110011001010000110
ER[1] = 011010101110100001010110100110100101010000001101
K[2] = 011110011010111011011001110110111100100111100101
= LA is an XOR function on ER against K
= ---------- xor
A[2] = 000100110100011010001111010000011001110111101000
B[2] = 110111000100001110000000111111 01
= PB is B based on substitution in the Permutation table
PB[2] = 100010111101100110001100000100 11

Iteration 3

==========

= L is the previous R
L[2] = 11010111000010110011001010000110
= R is PB xor previous L
R[2] = 10001011110110011000101010010000
ER[2] = 010001010111111011110011110001010101010010100001
K[3] = 010101011111110010001010010000101100111110011001

= LA is an XOR function on ER against K
= ---------- xor
A[3] = 000100001000001001111001100001111001101100111000
B[3] = 11010110001111001011011001111111
= PB is B based on substitution in the Permutation table
PB[3] = 01101111101100101001110011111110

Iteration 4
==========
= L is the previous R
L[3] = 10001011110110011000101010010000
= R is PB xor previous L
R[3] = 10111000101110011010111001111000
ER[3] = 010111110001010111110011110101011100001111110001
K[4] = 011100101010110111010110110110011010100011101
= LA is an XOR function on ER against K
= ---------- xor
A[4] = 001011011011100000100101000011101111011011101100
B[4] = 00101001110100001011101011111110
= PB is B based on substitution in the Permutation table
PB[4] = 00111111001110110100011110100001

Iteration 5
==========
= L is the previous R
L[4] = 10111000101110011010111001111000
= R is PB xor previous L
R[4] = 10110100111000101100110100110001
ER[4] = 11011010100101110000010101100101101010011010011
K[5] = 011111001110110000000111111010110101001110101000
= LA is an XOR function on ER against K
= ---------- xor
A[5] = 10100110011110110000001010001110111101000001011
B[5] = 01000001001111011000101011000011
= PB is B based on substitution in the Permutation table
PB[5] = 10010101001100101101100001000101

Iteration 6
==========
= L is the previous R
L[5] = 10110100111000101100110100110001
= R is PB xor previous L
R[5] = 00101101100010110111011000111101
ER[5] = 10010101101111000101011010111010110000011111010
K[6] = 011000111010010100111110010100000111101100101111
= LA is an XOR function on ER against K
= ---------- xor
A[6] = 11110110000110010110100011101010101110101101010101
B[6] = 01101101110111000011010101000110
= PB is B based on substitution in the Permutation table
PB[6] = 00100100000110111111001111111000

Iteration 7
==========
= L is the previous R
L[6] = 00101101100010110111011000111101
= R is PB xor previous L
R[6] = 10010000111110010011111011001001

ER[6] = 1100101000010111111100101001111111010110010100 11
K[7] = 1110110010000100101101111111011000011000101111 00
= LA is an XOR function on ER against K
= ---------- xor
A[7] = 0010011010010011010001010110100111001110111011 11
B[7] = 1110001101101011000001010010 1101
= PB is B based on substitution in the Permutation table
PB[7] = 1100100011000001111011001101100

Iteration 8

= L is the previous R
L[7] = 1001000011111001001111101100 1001
= R is PB xor previous L
R[7] = 1110010101001010100110000101000 1
ER[7] = 1111000010101010010101010100111100000010101000 11
K[8] = 1111011110001010001110101100000100111011111111 011
= LA is an XOR function on ER against K
= ---------- xor
A[8] = 0000011100100000011011111000111000111001010110 00
B[8] = 0000100011011000100000111101010 1
= PB is B based on substitution in the Permutation table
PB[8] = 0000011100111001001010010110000 1

Iteration 9

= L is the previous R
L[8] = 1110010101001010100110000101000 1
= R is PB xor previous L
R[8] = 1001011111000000000101111010100 0
ER[8] = 0100101011111110000000000000010101111110101010 0001
K[9] = 1110000011011011110101111101101111001111000000 1
= LA is an XOR function on ER against K
= ---------- xor
A[9] = 1010101000100101111010111110011100011010110100 00
B[9] = 0110111011100001101010110100 1010
= PB is B based on substitution in the Permutation table
PB[9] = 1101100100111011101000111001 0100

Iteration 10

= L is the previous R
L[9] = 1001011111000000000101111010100 0
= R is PB xor previous L
R[9] = 0011110001110001001110111100 0101
ER[9] = 1001111110000011101000101001111101111110000010 10
K[10] = 1011000111110011010001111011101001000110010011 11
= LA is an XOR function on ER against K
= ---------- xor
A[10] = 0010111001110000111001010010010100111000010001 01
B[10] = 0010000101110000010000010110 1101
= PB is B based on substitution in the Permutation table
PB[10] = 0000110000010101011011100010 0100

Iteration 11

= L is the previous R
L[10] = 0011110001110001001110111100 0101

= R is PB xor previous L
R[10] = 10011011110101010111100110001100
ER[10] = 01001111011111101010101010111111001111000101101 1001
K[11] = 00100001010111111101001111011110110100111000011 0
= LA is an XOR function on ER against K
= ---------- xor
A[11] = 01101110001000010111100101100001111011111101111 1
B[11] = 0101111000001100110110111100001 0
= PB is B based on substitution in the Permutation table
PB[11] = 01110001001111101011000001010011

Iteration 12
==========
= L is the previous R
L[11] = 10011011110101010111100110001100
= R is PB xor previous L
R[11] = 01001101010011111000101110010110
ER[11] = 00100101101010100101111111000101011111001010110 0
K[12] = 01110101011100011111010110010100011001111110100 1
= LA is an XOR function on ER against K
= ---------- xor
A[12] = 01010000110110111010101001010001000110110100010 1
B[12] = 01101000000010110011011010101101
= PB is B based on substitution in the Permutation table
PB[12] = 10101000011010001000111011101001

Iteration 13
==========
= L is the previous R
L[12] = 01001101010011111000101110010110
= R is PB xor previous L
R[12] = 00110011101111011111011101100101
ER[12] = 10011010011111011111101111111010111010110000101 0
K[13] = 10010111110001011101000111111010101110100100000 1
= LA is an XOR function on ER against K
= ---------- xor
A[13] = 00001101101110000010101000000000010100010100101 1
B[13] = 11111001110110110010010010110011
= PB is B based on substitution in the Permutation table
PB[13] = 10000110110010111100111111001011

Iteration 14
==========
= L is the previous R
L[13] = 00110011101111011111011101100101
= R is PB xor previous L
R[13] = 11001011100001000100010001011101
ER[13] = 11100101011111000000100000100000100000101111101 1
K[14] = 01011111010000111011011111110010111001110011101 0
= LA is an XOR function on ER against K
= ---------- xor
A[14] = 10111010001111111011111111010010011001011100000 1
B[14] = 10111000011111101100010111000001
= PB is B based on substitution in the Permutation table
PB[14] = 00000101110111010011101001001111

Iteration 15
==========

= L is the previous R
L[14] = 11001011100001000100010001011101
= R is PB xor previous L
R[14] = 00110110011000001100110100101010
ER[14] = 00011010110000110000000101100101101010010101010100
K[15] = 10111111100100011000110100111101001111110000 1010
= LA is an XOR function on ER against K
= ---------- xor
A[15] = 10100101010100101000110001011000100101100101111 0
B[15] = 01000001001110011111011100100111
= PB is B based on substitution in the Permutation table
PB[15] = 10100101001001101110110011101100

Iteration 16
==========
= L is the previous R
L[15] = 00110110011000001100110100101010
= R is PB xor previous L
R[15] = 01101110101000101010100010110001
ER[15] = 10110101110101010000010101010101000101011010 0010
K[16] = 11001011001111011000101100001110000101111111 0101
= LA is an XOR function on ER against K
= ---------- xor
A[16] = 01111110111010001000111001011011000000100101 0111
B[16] = 10000001011010101111011101001011
= PB is B based on substitution in the Permutation table
PB[16] = 00101001111101110110100011001100

= L is the previous R
L[16] = 01101110101000101010100010110001
= R is PB xor previous L
R[16] = 00011111100101111010010111100110

Step 7: Join LR
==========
RL = 0001111110010111101001011110011001101110101000101010100010110001
= CT is an RL value based on substitution in the Inverse Permutation table
CT = 0101011011110001110101011100100001010010101011111000000100111111
Hex = 56 F1 D5 C8 52 AF 81 3F

Ciphertext =

	13	34	57	79	9B	BC	DF	F1
DEC	19	52	87	121	155	188	223	241
CHAR	!!	4	W	y	›	¼	ß	ñ

4. CONCLUSION:

DES is a block cipher that operates with a 64-bit size. This algorithm is included in the substitution algorithm which is difficult to solve because it has 16 iterations during the encryption process. The DES algorithm can secure messages securely enough and give the Avalanche effect where each character that is secured using this DES algorithm is entirely unreadable and disappears on the ciphertext. The eight character key length used in the Data Encryption Standard algorithm can be said to be quite safe because when tested using the brute force method it takes more than 10 hours on average to find the password used. The use of this algorithm is beneficial for users in sending information on global networks. This algorithm does not require complicated mathematical calculations. This algorithm works by substituting the position of the plaintext matrix and the key to create a new character that is a ciphertext.

REFERENCES:

1. A. H. Lubis, S. Z. S. Idrus, and A. Sarji, "ICT Usage Amongst Lecturers and Its Impact Towards Learning Process Quality," vol. 34, no. 1, pp. 284–299, 2018.
2. A. P. U. Siahaan, "Rabin-Karp Elaboration in Comparing Pattern Based on Hash Data," *Int. J. Secur. Its Appl.*, vol. 12, no. 2, pp. 59–66, Mar. 2018.
3. M. D. T. P. Nasution, Y. Rossanty, A. P. U. Siahaan, and S. Aryza, "The Phenomenon of Cyber-Crime and Fraud Victimization in Online Shop," *Int. J. Civ. Eng. Technol.*, vol. 9, no. 6, pp. 1583–1592, 2018.
4. H. Ming and S. LiZhong, "A New System Design of Network Invasion Forensics," in *2009 Second International Conference on Computer and Electrical Engineering*, 2009, pp. 596–599.
5. B. Forouzan, *Cryptography and Network Security*. New York, NY, USA: McGraw-Hill, 2006.
6. A. Lubis and A. P. U. Siahaan, "Network Forensic Application in General Cases," *IOSR J. Comput. Eng.*, vol. 18, no. 6, pp. 41–44, 2016.
7. W. Stallings, *Cryptography and Network Security Principles and Practices*, 4th ed. Prentice Hall, 2005.
8. M. Syahrizal, M. Murdani, S. D. Nasution, M. Mesran, R. Rahim, and A. P. U. Siahaan, "Modified Playfair Cipher Using Random Key Linear Congruent Method," *J. Online Jar. COT POLIPD*, vol. 10, no. 2, pp. 45–49, 2017.
9. I. Sumartono, A. P. U. Siahaan, and Arpan, "Base64 Character Encoding and Decoding Modeling," *Int. J. Recent Trends Eng. Res.*, vol. 2, no. 12, pp. 63–68, 2016.
10. S. Ramadhani, Y. M. Saragih, R. Rahim, and A. P. U. Siahaan, "Post-Genesis Digital Forensics Investigation," *Int. J. Sci. Res. Sci. Technol.*, vol. 3, no. 6, pp. 164–166, 2017.
11. Khairul *et al.*, "Effect of Matrix Size in Affecting Noise Reduction Level of Filtering," *Int. J. Eng. Technol.*, vol. 7, no. 3, pp. 1272–1275, 2018.
12. A. P. U. Siahaan, *How to Code: Advanced Encryption Standard in C#*. Medan: Fakultas Ekonomi Universitas Panca Budi, 2018.
13. M. Sharma and R. B. Garg, "DES: The oldest symmetric block key encryption algorithm," in *2016 International Conference System Modeling & Advancement in Research Trends (SMART)*, 2016, pp. 53–58.
14. A. P. U. Siahaan, "Three-Pass Protocol Concept in Hill Cipher Encryption Technique," *Int. J. Sci. Res.*, vol. 5, no. 7, pp. 1149–1152, 2016.
15. A. P. U. Siahaan, "Factorization Hack of RSA Secret Numbers."
16. Hariyanto and A. P. U. Siahaan, "Intrusion Detection System in Network Forensic Analysis and," *IOSR J. Comput. Eng.*, vol. 18, no. 6, pp. 115–121, 2016.
17. V. Tasril, M. B. Ginting, Mardiana, and A. P. U. Siahaan, "Threats of Computer System and its Prevention," *Int. J. Sci. Res. Sci. Technol.*, vol. 3, no. 6, pp. 448–451, 2017.
18. B. Kapoor, P. Pandya, and J. S. Sherif, "Cryptography," *Kybernetes*, vol. 40, no. 9/10, pp. 1422–1439, Oct. 2011.
19. W. Fitriani, R. Rahim, B. Oktaviana, and A. P. U. Siahaan, "Vernam Encypted Text in End of File Hiding Steganography Technique," *Int. J. Recent Trends Eng. Res.*, vol. 3, no. 7, pp. 214–219, Jul. 2017.
20. I. Sumartono, A. P. U. Siahaan, and N. Mayasari, "An Overview of the RC4 Algorithm," *IOSR J. Comput. Eng.*, vol. 18, no. 6, pp. 67–73, 2016.
21. B. Parmaza, "Substitution Cipher," 2018. [Online]. Available: http://itjambi.com/cipher-substitusi-substitution-cipher/. [Accessed: 10-Oct-2018].
22. Zae, "Algoritma DES (Data Encryption Standard)," *2009*. [Online]. Available: http://ilmu-kriptografi.blogspot.com/2009/05/algoritma-des-data-encryption-standart.html.
23. Zae, "Tutorial Pemrograman Kriptografi C++ dan Java," *Belajar Kriptografi Sambil Ngopi*, 2015. [Online]. Available: http://ilmu-kriptografi.blogspot.com/2009/05/fungsi-hash.html.
24. A. Singh, M. Marwaha, B. Singh, and S. Singh, "Comparative Study of DES, 3DES, AES and RSA," *Int. J. Comput. Technol.*, vol. 9, no. 3, pp. 1162–1170, Dec. 2010.
25. Zacky, "Enkripsi Algoritma DES (Data Encryption Standard)," *KRIPTOGRAFI & JARINGAN KOMPUTER*, 2016. [Online]. Available: http://kriptografijaringan.blogspot.com/2016/03/enskripsi-algoritma-des-data-encryption.html.