2018 International Conference on Computer, Control, Electrical, and Electronics Engineering (ICCCEEE)

An Enhancement of Data Encryption Standards Algorithm (DES)

Nadia Mustafa Mohammed Alhag
University of Gezira
Faculty of Mathematical Sciences and Computer
Medani, Sudan
fegaga7@gmail.com

Yasir Abdelgadir Mohamed
Karary University
Faculty of Computer Science and Information Technology
Khartoum, Sudan
Yasir_eym@yahoo.com

***Abstract*— The Data Encryption Standard (DES) algorithm has been considered as the most popular symmetric key, blocks ciphering cryptographic. Even though the DES algorithm had still been used in some applications, it was considered unsafe because of the short key length (64 Bits), besides, the Brute force attack has shown that the DES practically can be attacked. In January 1999, the DES key was cracked in only (22 hours) and (15 minutes). The aim of this research is to improve the DES algorithm by increasing the key length (1024 bits) that is to be divided into 16 keys (64 bits each), each key is independently generated for the different algorithm cycles. The results of the proposed algorithm were much better than the old algorithm for detecting the encryption key or the total number of keys that could be generated by following Blind search method (try all possible keys), Increasing the key length (degree of complexity) makes it difficult to search in a vast space of numbers and attempts.**

Keywords—Security, Data Encryption Standard, Key

I. INTRODUCTION

Information security and systems in the digital environment represent the protection of information in terms of availability, confidence and integrity. The availability reflects the characteristics of information systems that can be accessed and used on an immediate basis within a specific and desired pattern, the system can also be accessed when requested in an approved manner and according to the appropriate specifications of the system. Confidentiality is a property associated with the non-change, loss or loss of data and information only to persons and entities authorized and authorized to use it. The DES algorithm is considered unsafe as a result of the key length, which is considered too small (56 binary), which is one key from which the other keys are generated. The aim of this research is to improve the DES algorithm in terms of security and confidentiality by means of increasing the length of the key to be (1024 binary) and dividing it into sixteen keys each key independently of each other in each cycle of the algorithm. The importance of research is to use DES algorithm to provide security and confidentiality. The research has followed the same approach and method of the old one-key algorithm with some additions and changes. A 1024-bit key was used and divided into sixteen keys each key independent of the other. The PC1 (Permuted Choice-1) PC-1 selection table was used sixteen times instead of one time as in the old algorithm.

A shift table was also used sixteen times and the shift table was constant each time (only 1 bit shift to the left on all keys).

II. RELATED WORK

In [1], Sombir Singh improved the algorithm using the Simple Column Transposition Technique (Transposition Cryptography Techniques), which is to arrange the text in the form of rows within a square and then rearrange and read it vertically in a random way. According to the use of the text and to what extent it is intended to be complicated. The result of this arrangement is the text that is inserted into the DES algorithm to encrypt it. Therefore, the resulting encoded text is very complex, making it difficult to break the algorithm. The application of this algorithm requires additional external operations to execute and generate a number of columns randomly and then send them over the network affects network performance [2].

Payal Patel [3] has improved the algorithm by increasing the key length and increasing the complexity of the S-boxes as well as increasing the number of cases used to represent the given information. The aim of increasing the key length was not to make it easier for brute force attack. In this study, the researcher hasn't addressed the key.

Ayman E. Mohammed, Faisal M. Abdallah [4] worked on improving the algorithm by proposing a method for constructing the sub-keys in the algorithm using the genetic algorithm. The sub-keys generated by this technique depends on the genetic algorithm that gives a completely different set of semi-random sub-keys each time where the program is implemented [5]. Subkeys are generated from one basic key, unlike the proposed algorithm.

Arvind Kumar [6] proposed mechanisms to improve the DES algorithm by using two additional keys in addition to the 64-bit primary key, as well as an adjustment in some internal processes of DES using S-BOX for the AES algorithm.

Ramya G et.al in [7], has improved the performance of the DES algorithm by increasing the length of the key to 128 bits, a key from which the rest of the keys are generated, unlike the proposed algorithm where keys are inserted completely independently.

In [8] the researcher has worked on improving the DES algorithm by improving the key generation process using two random filters (each 8 digits), a left matrix and a left matrix and the random selection of numbers in these two arrays (0-27). The purpose of using the two arrays is to rearrange the bits in the keys. When the key is divided into two parts, the right part is rearranged according to the numbers in the right matrix and the left part then arranges the bits based on the numbers of the left matrix. Then the offset is then merged into a 56-bit key to be rearranged by inserting it into the PC-2 table. Any attempt to break this algorithm requires a key (56 bits) Randomized matrix [9].

III. METHODOLOGY

The following diagram illustrates the proposed methodology.

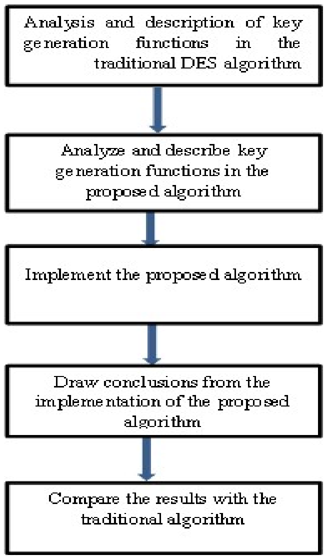

Analysis and description of key generation functions in the traditional DES algorithm
Analyze and describe key generation functions in the proposed algorithm
Implement the proposed algorithm
Draw conclusions from the implementation of the proposed algorithm
Compare the results with the traditional algorithm

Figure 1.Proposed methodology

A. Analysis and description of key generation functions in the traditional DES algorithm)

The key generation functions in the DES algorithm are completely independent of the cryptographic functions, so we only analyze key generation functions as the key is the key point in the algorithm. The sub-functions are:

i. *PC-1 Function*: This function is used only once, the input is the original 64-bit key, in which the bits are rearranged and the bits are deleted (8-16-24-32-44-56-64).

ii. *Shift function:* The output of the PC-1 function (56 bits) is divided into two equal parts (28 bits) (CD). Each of these sections is shifted to the left according to Table (3.2). This table is used 16 times to give 32 partition (16 keys) in different orders. After the offset is done, the two sections are merged into each other so that the output is a 56-bit key.

iii. *PC-2 Function:* In this function, the input is a 56 bits key. positions were switched and some bits in positions (9, 18, 22, 25, 35, 38, 43, 54) (3-3) were deleted.

The key is the input the PC-1 table. This table is used only once. The output is then divided into two equal parts. Each one is left shifted based on the SHIFT table which is used 16 times and the two sections are then merged and inserted into the PC-2 table that is also used 16 times.

B. Calculating time and storage complexity in traditional DES algorithm

The efficiency of the algorithm is determined based on two basic parameters:

i. The space complexity: The amount of memory required by a specific program operation until completion, where this type depends on two parts:

   a. A fixed part: independently of the input and output characteristics, this part includes the space instruction, variables - whether simple or variable- in addition to the vacuum constants.

   b. Variable part: consists of the space required by the program with complex variables, which depends on the size of the problem to be solved, in addition to the vacuum stack, and there is also dynamic storage can be explained by the variables entered by the program as well as the names of these variables. Hence the complexity of the vacuum S (p) of the program p is S(p) = const + Sp where

   Const: represents the fixed part of the algorithm or program.

   Sp: Represent the properties of the example.

ii. Time complexity measure: The amount of time required to form a program until its completion consists of: T (p) = Const + tp Where:

   Const: represents a constant for the time of translation or authorship.

   Tp: represents the runtime of the program.

C. Total time implementation algorithm

The execution time of each statement is related to the number of times that the statement was executed in addition to the single execution time of the term since the executable time of the term = number of times of implementation * single implementation time. To measure the run time of an algorithm executing program, there are many disadvantages to be found in this measurement, depends on the computer speed that runs the program, the quality of the interpreter or interpreter which produces the code in addition to the program quality, because we need to measure the algorithm efficiency, we need a method of measurement that is not affected by these external factors. This method calculates the number of times each algorithm is executed. This method is very difficult and is usually unnecessary. The most important process in the algorithm, called the "main process", which constitutes most of the execution time, counting the number of times executed on the input of the size of n.

The Big (O) notation is one of the best and most widely used formulas to measure the complexity of time and storage so it has been used here. The table below illustrates the algorithm complexity:

TABLE 1 TIME COMPLEXITY OF THE DES ALGORITHM

STEPS	EXCU-TIMES
Get (KEY)….	N2
FOR I := 1 TO 64 DO.	4N)2(
DO…PC-1	N2
FOR I := 1 TO 56 DO.	(3N+8) 2
FOR I := 1 TO 16 DO.	2N+2
DIVIDING (KEY) TO TWO+ITRATION.	N2
FOR I,J := 1 TO 56 DO.	(3N+8) 2*N
DO ….PC-2	N2
FOR I,J := 1 TO 56 DO.	(3N)2*N
PRINT THE RESULT TO THE ARRAY OF KEYS.	2N+2
FOR I.J := 1 TO 48 DO.	2(3N)+2*N
END FOR .	
THE END	
(COMPLEX)	$O(N^3)$

In the table above, the variable N was used but with a different weight as an attempt to use one variable that combined with some other specific numbers (16, 64, 56, and 48). The number 16 was chosen to be N and therefore 64 becomes 4N and 56 becomes 3N + 8) and 48 become 3N)). Using 2N + 2 indicates the loop complexity which is equals to 2N + 2 N2

The process of multiplication (* N) means that the phrase is followed or influenced by the loop repetition. To Measure the storage complexity we follow the algorithm below:

For (int i =0 ; i < 64 ; i++) Get K , " 64-bit key"
A variable for the loop that receives the loop key is repeated
64 times = 4N, A 64-bit matrix used once = 4N
= 4N+4N = 8N
For (int i =0 ; i < 56 ; i++)
K '= PC1 (K), applying permitted choice 1 and returning 56 bits
The frequency loop variable uses 3N + 8 times
A 56-bit matrix used once representing the temporary key =
3N + 8
A 56-year matrix representing the PC-1 table used once = 3N + 8
A 64-bit matrix used to represent the primary key = 4N -
The rearrangements are affected by the frequency loop = 3N + 8 = $(3N+8+3N+8+4N)*(3N+8)^2 = (10N+16)(9N^2+48N+16) = 90N^3 + 480N^2 +160N +144N^2 +768N +256 = 90N^3+620N2+ 928 N+256$ For (int i =1 ; i <= 16 ; i++)
The loop variable is used N times: For (Int i = 0; i <56; i ++)
{ The loop variable uses 3N + 8 times
K' =(C(i), D(i)) , dividing K' into two 28-bit parts
Three matrices with a length of 28 and one length of 56
3N + 8 + N + 12 + N + 12 = 5N + 32
K'= (r(i)C(i), r(i)D(i)) , shifting to the left
Three matrices with a length of 28 and one length of 56 = 3N + 8 + N + 12 + N + 12
We use one of the offset tables twice (for the two sections) = 2
A variable for the loop to which the temporary key receives a loop repeated 56 times = 3N + 8, the external loop (number of cycles) is repeated N times:
= 5N+34 = $(10N+66)(3N+8) + 3N+3 = (30 N^2 +281 N + 536)*N = 30N^3+281N^2+ 536N$ For (int i =0 ; i < 48 ; i++){
The frequency loop variable is used 3N times
K [i] = PC2 (K '), applying permitted choice 2 and returning 48 bits. A 48-bit matrix used once = 3N
A 48-table matrix represents a PC-2 table that is used once = 3N. A 56-bit matrix used once represents the temporary key = 3N + 8. A variable for the loop to which we receive the temporary key loop is repeated 48 times = 3N. The rearrangements (PC-2) are affected by the frequency loop = 3N
= (3N+3N+3N+8)*3N+3N
= $(27N^2+24N)+3N$
= $(272+27N)*N= 27N^3+27N^2$
Thus, the complexity of storage or space = S (P) = N^3
S (P): The amount of memory required to implementing a program till it completes.

IV. KEY GENERATION ANALYSIS

When you start analyzing the algorithm, there are a number of factors that can be ignored that affect the behavior of the algorithm when we move from one device to another, and focus on general rules that can be implemented on algorithms regardless of the structure or device you work on. Judging the effectiveness of that algorithm in general. The analysis is used to predict the size of the resources that the algorithm needs when executing the process for which it is built. Among those resources we need is the memory and time needed to execute. What is often Computational Time or the time it takes to accomplish its mission. In general, analyzing more than one algorithm to solve a particular problem helps us to know the best of those algorithms. We mean the "best" algorithm that takes the least time / consumption in implementation to solve that problem. In the course of studying these algorithms, we begin to abandon them One by one based on certain criteria, until the best Efficient Algorithm is reached.

A. *Key generation functions in the proposed DES algorithm*

i. *PC-1 Function*

This function is used 16 times as opposed to using the traditional DES only once, which is used with each cycle. In each cycle the 64-bit key is inserted and the bits are rearranged and deleted at the sites (8-16-24-32) -40-48-56 64) from the original key and the output from this function is 16 keys (56 bits) according to Table (2-1). PC-1 the use of this table 16 times may increase the complexity of the algorithm, more sources (time and storage).

ii. Shift function

The output from the PC-1 function for all keys (56 bits) is divided into two equal parts (28 bits) (CD). Each of these sections is moved to the left with only one bit. The output from this function Is 16 keys (56 bits) after the two sections are merged into each other.

iii. PC-2 Function

This function is also used as in the traditional DES algorithm where the keys are inserted after the offset function is applied. This function works to switch locations and delete the bits in the sites (9, 18, 22, 25, 35, 38, 43, 54) based on Table (2-3) PC-2 and the result of this function is 16 key (48 bit) Then one key is used with each session in the encryption process.

```
Algorithm   name: Key algorithm for (DES (16K)):
Input:
K(1),…,K(16): 64-bit keys
PC1: Permuted choice 1
r1, ... , r16: left shifts (rotations)
PC2: Permuted choice 2
Output:
K(1) , k(2), ..., k(16):  48-bit keys
Algorithm:
For (Int i =1 ; i <= 16 ;  i++)
{
Get   K(i) , " 64-bit key"
For (Int i =1 ; i <= 16 ;  i++)
K' = PC-1 (K(i))," applying permuted choice 1 and
returning 56 bits"
For (Int i =1 ; i <= 16 ;  i++)
K' =(C (i),D(i) , "dividing K' into two 28-bit parts"
K'= (r(i)C(i), r(i)D(i)) , "shifting to the left"
K"[i] = PC-2(K'), "applying permuted choice 2 and
returning 48 bits"
```

As shown in algorithm above, the loop is executed 16 times for all operations from entering the keys and then rearranging them by inserting them into the PC-1 table. The output from this table is divided into two equal parts, and each section is shifted to the left according to the SHIFT table and then The two sections are merged and the output from this table is inserted into the PC-2 table.

B. *Measurement of the time complexity of the key generation algorithm in the proposed DES*

TABLE 2. TIME COMPLEXITY IN THE PROPOSED ALGORITHM

STEPS	EXCU-TIMES
FOR I: = 1 TO 16 DO.	2N+2
Get (KEY)….	N^2
FOR I,J := 1 TO 64 DO.	$4N)^2$*N(
DOPC-1	N^2
FOR I, J: = 1 TO 56 DO.	$(3N+8)^2$*N
DIVIDING (KEY) TO TWO+ITRATION.	N^2
FOR I,J := 1 TO 56 DO.	$(3N+8)^2$*N
DO PC-2	N^2
FOR I,J := 1 TO 56 DO.	$(3N)^2$* N
PRINT THE RESULT TO THE ARRAY	2N+2
FOR I,J := 1 TO 48 DO.	2 (3N)+2*N
END FOR.	
THE END	
(COMPLEX)	$O(N^3)$

In the table above, the variable N was used, but with a different weight. This is due to the attempt to use one variable that is related to the other numbers (16, 64, 56, and 48). The number 16 is chosen to be N and therefore 64 becomes 4N and 56 becomes 3N + 8) and 48 become 3N)).

Using 2N + 2 This means that the loop complexity = 2N + 2

(For i: = 1 to n do) = i =: 1 is
executed once = 1 i <= n
executes n + 1 times = n + 1 i
++ executes n times = n
If the loop executes = 2n + 2

N2 is the complexity of the tables used is because it uses an order by the entry (we pass through each bit in the table once and add the bit position change)

The process of multiplication × N means that this phrase is followed or influenced by the loop repetition.

C. *Measuring storage complexity*

For (Int i =0 ; i < 64 ; i++)
Get K, " 64-bit key"
A variable for the loop that receives the loop key is repeated
64 times = 4N
A 64-bit matrix used once = 4N
= 4N + 4N = 8N
For (int i =0 ; i < 56 ; i++)
K' = PC1(K), applying permuted choice 1 and returning 56 bits
The loop variable uses 3N + 8 times
A 56-bit matrix used once representing the temporary key = 3N + 8. A 56-year matrix representing the PC-1 table used once = 3N + 8. A 64-bit matrix used to represent the primary key = 4N. The rearrangements are affected by the frequency loop = 3N + 8 (3N +8+3N+8+4N) * $(3N+8)^2$ = (10N+16) $(9N^2+48N+16)$ = $90N^3+480N^2+160N+144N^2+768N+256 = 90N^3 + 620N^2 + 928N + 256$ For (int i =1 ; i <= 16 ; i++), the loop variable uses N times
For (int i =0 ; i < 56 ; i++) {
The loop variable uses 3N + 8 times
K' =(C(i), D(i)) , dividing K' into two 28-bit parts
Three matrices; two with a length of 28 and one with a length of 56
= 5N+32
K'= (r(i)C(i), r(i)D(i)) , shifting to the left

Two matrices with a length of 28 and one length 56 = 3N+8+N+12+N+12; We use one box of the displacement table twice (for the two sections). A loop variable where the temporary key received is repeated 56 times
= 3N+8
The external loop (number of cycles) is repeated N times
= 5N+34
= (10N+66) (3N+8) + 3N+3
= $(30 N^2 +281 N +536)*N$
= $30N^3+281N^2+ 536N$
For (int i =0 ; i < 48 ; i++) {
= (3N+3N+3N+8)*3N+3N
= $(27N^2+24N)+3N$
= $(272+27N)*N= 27N^3+27N^2$

V. IMPLEMENTATION AND RESULTS

The application program contains key functions. The DES algorithm is of the Feistel structure. Therefore, the functions of encryption and decryption are similar with the use of inverted keys in decoding. It contains a time function to calculate the time it takes to generate the keys in each of the two algorithms so that we can compare them.

The calculation of time is as follows:

Taking the start time of the program Start Time At the start of the execution of the functions of generating keys we took the time Start K Time

The output is the time of generation of the keys. At the end of the execution of the program, a time was taken for the total execution time. The time of encryption and decoding was calculated from the output of the generation time of the keys. When implementing the program (implementation of the algorithms in order to obtain the results and then compare them with each other) Fig. 2 shows the program interface through which a user can access to the other interfaces.

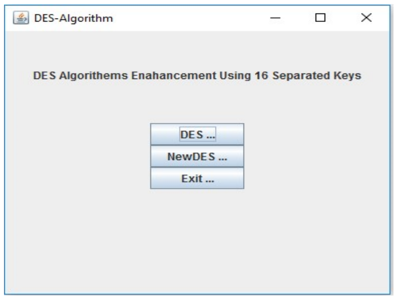
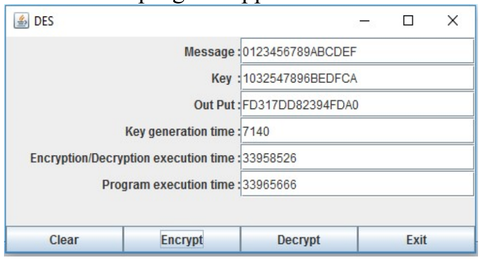


Figure 2. The main program execution interface

In Fig. 3, the interface shows the encryption process by inserting the encrypted text and key, then pressing the Decrypt button in addition to the time it takes to generate the keys, the decoding process and the total program time.

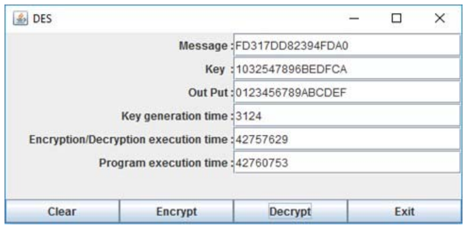
- Message: Shows the text to decrypt
- Key: is the same key that was used in encryption
- Output: The explicit text appears after pressing the Decrypt box
- Key generation time: shows the time it takes to generate keys, Encryption / Decryption time.
- Program Execution Time: The time taken to execute the overall program appears.



Message : 0123456789ABCDEF
Key : 1032547896BEDFCA
Out Put : FD317DD82394FDA0
Key generation time : 7140
Encryption/Decryption execution time : 33958526
Program execution time : 33965666

Figure 3. Implementation interface of the traditional algorithm (encryption phase)

On the other hand, the decryption process is illustrated an interface shows the decryption process in Fig.4.



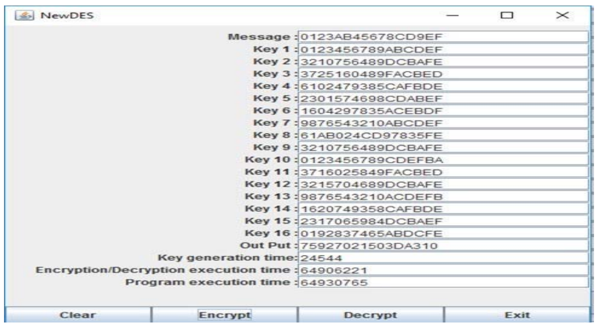
Message : FD317DD82394FDA0
Key : 1032547896BEDFCA
Out Put : 0123456789ABCDEF
Key generation time : 3124
Encryption/Decryption execution time : 42757629
Program execution time : 42760753

Figure 4. Implementation interface of the traditional algorithm (Decryption)

It starts with entering the text and key and then pressing the Decrypt button in addition to the time it takes to generate the keys, the decryption process and the total program time.

- Message: Shows the text to decrypt
- Key: is the same key that was used in encryption
- Output: The explicit text appears after pressing the Decrypt box
- Key generation time: shows the time it takes to generate keys, Encryption / Decryption time:
- Program Execution Time: The time taken to execute the overall program appears.

In Fig. 5, an interface shows the encryption process in the proposed algorithm where the text is inserted and 16 keys are pressed and the Encrypt button is pressed in addition to the time it takes to generate the keys, the encryption process and the total program time.

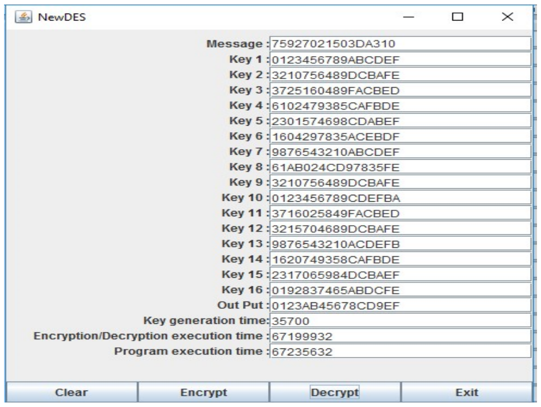


Message : 0123AB45678CD9EF
Key 1 : 0123456789ABCDEF
Key 2 : 3210756489DCBAFE
Key 3 : 3725160489FACBED
Key 4 : 6102479385CAFBDE
Key 5 : 2301574698CDABEF
Key 6 : 1604297835ACEBDF
Key 7 : 9876543210ABCDEF
Key 8 : 61AB024CD97835FE
Key 9 : 3210756489DCBAFE
Key 10 : 0123456789CDEFBA
Key 11 : 3716025849FACBED
Key 12 : 3215704689DCBAFE
Key 13 : 9876543210ACDEFB
Key 14 : 1620749358CAFBDE
Key 15 : 2317065984DCBAEF
Key 16 : 0192837465ABDCFE
Out Put : 75927021503DA310
Key generation time: 24544
Encryption/Decryption execution time : 64906221
Program execution time : 64930765

Figure 5. Implementing the proposed DES with different keys

In Fig. 6 an interface shows the decoding process in the proposed algorithm, where the encoded text is entered and 16 keys are pressed and the Decrypt button is pressed. In addition to the time it takes to generate the keys, the decryption process

(Fig.7) and the total program time, this interface is similar to the decryption interface in the traditional algorithm above with a difference in the appearance of the boxes from Key1 to Key16 in which 16 keys are inserted.



```
NewDES
                            Message :75927021503DA310
                              Key 1 :0123456789ABCDEF
                              Key 2 :3210756489DCBAFE
                              Key 3 :3725160489FACBED
                              Key 4 :6102479385CAFBDE
                              Key 5 :2301574698CDABEF
                              Key 6 :1604297835ACEBDF
                              Key 7 :9876543210ABCDEF
                              Key 8 :61AB024CD97835FE
                              Key 9 :3210756489DCBAFE
                             Key 10 :0123456789CDEFBA
                             Key 11 :3716025849FACBED
                             Key 12 :3215704689DCBAFE
                             Key 13 :9876543210ACDEFB
                             Key 14 :1620749358CAFBDE
                             Key 15 :2317065984DCBAEF
                             Key 16 :0192837465ABDCFE
                            Out Put :0123AB45678CD9EF
               Key generation time:35700
Encryption/Decryption execution time :67199932
             Program execution time :67235632
```

Figure 6. Implementing the proposed DES (Decryption)

In table 3, in the first case, the key attacker is supposed to know a lot about this algorithm. Using a 1024-bit key divided into 16 64-bit partitions per section and then deleting the bits, each section is 56 bits long and the length of each section is 896 bits. Known as well-known boxes are also used from the offset table, the number of keys that can be detected in this case is $10^{269} * 5.3 = 2^{896}$

TABLE 3. RESULTS OF KEY DETECTION COMPLEXITIES OF ALGORITHMS

Keys#		Algorithm
2nd Case	**1st Case**	
2^{56}	2^{56}	DES
$16 * x * y * 2^{869}$	2_{896}	New DES

In the second case, the key does not know much about this algorithm. Only the use of a 1024-bit key is divided into 16 64-bit partitions per section. The audit bit is deleted from each section. The length of each section is 56 bits and the length of each section is 896 bits. In an unknown sequence, the cells with anonymous values will also be used from the offset table.

As illustrated in Table 4, the difference is in the number of input times, the use of the PC-1 table, and the times a key is generated, the rest are similar with the conventional algorithm.

Table 4. COMPARISONS OF THE NUMBER OF APPROXIMATIONS OF THE TWO ALGORITHMS

# PC 2 used	**The number of times the key is divided and the offset**	**# PC 1 used**	**# times new a key is generated**	**Algorithm**
16	16	1	1	DES
16	16	16	16	New DES

In Table 5, the results of time processing (key generation - encryption - decryption of algorithms) is illustrated where in Table 6 the time and space complexity are declared.

TABLE 5 . RESULTS OF TIME COMPARISONS

Decryption (ɳs)	**Encryption (ɳs)**	**Key generation (ɳs)**	**Algorithm**
42757629	33958526	7140	DES
67199932	64906221	24544	New DES

TABLE 6 . TIME AND SPACE COMPLEXITY

S	**T**	**Algorithm**
N3	N3	DES
N3	N3	New DES

The basic idea of the proposed algorithm has been achieved; that is; independency of t generated sixteen keys independent.

VI. CONCLUSION

Data Encryption Standard (DES) is the most popular method of encrypting symmetric key blocks and although it is still used in some applications but is unsafe due to the length of the key, which is very short (56 bits) Brute force attack that DES can be attacked in practice. The aim of this research was to improve the DES algorithm by increasing the length of the key (1024 binary) and divided into 16 keys each key independent of the other in each of the algorithm cycles. Thus, the "Permuted Choice -1" (PC-1) table was used 16 times instead of once and the offset table was used 16 times. The offset value is constant at each time (only 1- bit shift to the left on all keys) and the PC-2 (Permuted Choice -2) table was used 16 times to implement the proposed algorithm. The Java Virtual Machine (JVM) environment: (TextPad), and the results are much better than the old algorithm for detecting the encryption key or the total number of keys that can be generated by blind search (all possible keys) as increasing the key length (complexity). The basic idea of the proposed improvement is the disconnection between the subkeys and to find sixteen keys independent of each other.

REFERENCES

[1] Singh, S., Maakar, S.K. and Kumar, D.S., 2013.Enhancing the security of DES algorithm using transposition cryptography techniques. International Journal of Advanced Research in Computer Science and Software Engineering, 3(6), pp.464-471.

[2] Patel, P., Shah, K. and Shah, K., 2014. Enhancement Of Des Algorithm With Multi State Logic. International Journal of Research in Computer Science, 4(3), p.13

[3] Modibada, Jabalpur, "Modification Of Des Algorithm", international journal of innovative research & development, pp. 1-11, 2012.

[4] Mohammed, A.E. and Abdalla, F.M., 2015. DES Security Enhancement using Genetic Algorithm.

[5] Sharma, A.K. and Sharma, H., 2015. New Approach To Des With Enhanced Key Management And Encryption/Decryption System (Des Ultimate). International Journal of Advances in Engineering & Technology, 8(3), p.368.

[6] Ms. G.Ramy, Ms.Madona Anita , 2015. Enhacing Des And Aes with 1024 Bits Key

[7] Bansal, D.R. and Thakur, P., 2016. Improved Key Generation Algorithm In Data Encryption Standard (DES).

[8] Stallings, W., 2006. Cryptography and network security: principles and practices. Pearson Education India

[9] Modibada, Jabalpur, "Modification Of Des Algorithm", international journal of innovative research & development, pp. 1-11, 2012.