Combined Multi-Encryption Techniques for Text Securing Using Block Cipher and Stream Cipher Crypto-Systems

http://dx.doi.org/10.3991/ijes.v1i1.2901

Hassene seddik
University of Tunis, Tunis, Tunisia

***Abstract*—Due to the growth of electronic intrusion techniques and personal data control, it is necessary for private or public organization to secures their data and prove their owner rights. Encryption techniques provide a useful mean to secure internet files exchange or numeric storage data. Old techniques present some disadvantages decreasing their robustness face to attacks. In this paper a new approach presenting a multi-encryption system for text and file text is presented. Dividing the file in different parts, each part is encrypted by a different algorithm to generate a poly-encrypted file.**

***Index Terms*—Block encryption, stream cipher encryption, poly-encryption, combined encryption, text and file encryption.**

I. INTRODUCTION

Many encryption software and applications are freely downloadable on the internet for text and file text securing. For example the software named "cryptage-text" available at this link: http://www.softonic.fr/s/cryptage-texte allows these tasks with the most absolute ease [1]. However, all these programs are based on algorithms whose attacks strategies are previously known [2]. It is in this context that we decided to develop an approach inspired on existing algorithms but by exploiting their advantages and improving their unseemly to increase their strength against different possible attacks. In this paper two encryption techniques are developed. They are combined alternately to encrypt pseudo-randomly a file. This file will be decomposed into sub-spaces based on the number of words that it contains, and each subset is assigned to a chosen encryption technique. Once all the text is encrypted it will be grouped again to present a secured file.

II. PROPOSED ENCRYPTION TECHNIQUES

In this paper two encryption techniques are proposed. Based on two families of symmetric crypto-Systems: combined block cipher and stream cipher. The development of these algorithms is inspired from in use techniques exploiting their improper to make them more robust and more resistant to preconceived attacks. The first technique is called Logic Stream Cipher Encryption "LSCE" which is based on the logic function "xor", and the second is called "DESM" which is a modified version of the famous algorithm "DES" [3].

A. Presentation of the LSCE Encryption technique

The first proposed technique is an algorithm that belongs to the family of encryption stream or "stream cipher" symmetric key based on XOR logical operator that has a mathematical characteristic [4], the plaintext can be deduced from the encrypted and vice versa through the flow key by logical sum as shown in the following equations:

$$x \oplus k = y \tag{1}$$

y and x are respectively the cipher text and the plaintext, the operator $\oplus$ is an exclusive OR (XOR), k the, key stream. Decryption is defined as follows:

$$y \oplus k = (x \oplus k) \oplus k = x \oplus (k \oplus k) = x \tag{2}$$

- The proposed approach differs from other approaches of the same types with the following advantages:

The generation of the encryption key as a random alphanumeric string drawn randomly with the same length as the original message. This key is generated from a short key belonging to the operator. The use of a short key is very important because it hides information about the actual size of the clear message.

- Increase the size of the dictionary based messages to 144 characters in extended ASCII noted $\Im$. Therefore extending the workspace is defined by the following equation: $\forall x \in \Im, e_z(x) = y / y \in \Im$

Stages of the creation of the key stream are achieved by the following method through a random short key input by the authorized user note K_{op} with length m:

$$K_{op} = \{k_1 \ldots k_m\} \tag{3}$$

K_{op} is repeated a-periodically by reversing the order of its bits at each step P up to the end of the clear message. The step P is determined by the order of the letter of lowest weight 'LSB' of the m key elements introduced to generate the key space β_i.

$$\beta_i = \left\{ \begin{matrix} [k_1 \dots k_m]_1, .., [k_1 \dots k_m]_p, [k_m \dots k_1]_1, [k_m \dots k_1]_2, \dots \\ , [k_m \dots k_1]_{2p}, \dots\dots, [k_1 \dots k_m]_{ip} \end{matrix} \right\}$$

Such that the value of the permutation P is not described as follows:

$$p = LSB(k_1 \dots k_m) \qquad (5)$$

$$\beta_i = \{K_{1op}, K_{2op}, \dots, k_{iop}\} \qquad (6)$$

A function noted C is used to order the elements and truncate the flow order to L elements respectively to the size of the clear message.

$$\beta_{i+1} = C(\beta_i) \qquad (7)$$

$$\beta_L = \{k_1 \dots k_m, k_1 \dots k_m, k_1 \dots k_m, \dots, k_1 \dots k_L\} \qquad (8)$$

A permutation function S allows a redistribution of bits of the key of length L by changing the position of its components depending on the position of the Kop vector elements beginning from the middle:

$$Z_i = S_i(\beta_L, K_{iop}) \qquad (9)$$

The randomness is introduced to break the correlated model of the function that models the periodicity of the key chain using the following equation

$$\forall k \in K_{op}, and\ Z_i = S_i(\beta_L) \nexists f \text{ such as } Z_i = f(K_{iop}) \qquad (10)$$

Finally we recover a stream of random keys Zi, which used to encrypt the original text.

$$Z_i = \{k_1, k_2, k_9, k_6, k_{L-2} \dots . k_L\} (11)$$

Our technique belongs to the families called one-time pad "OTP", since the key-stream z is generated randomly and independently of the text and the same length as the text to be encrypted. The encryption function $e_k(x)$ defines the cipher text by the following equation.

$$e_k(x) = y_i = x_i \oplus z_i \qquad (12)$$

As $Z_i \in \Im$, $\oplus$ is the logic operator (XOR), y_i the encrypted text with length i, x_i the plain text, Z_i the key stream. Since the encryption technique is reversible and lossless then:

$$\forall x \in \Im, \exists e_z(x) / h(z,x) = e_z(x) = y \text{ and } d_z(y) = h^{-1}(z,y) = x$$

Then the steps of the decryption techniques are similar to those of the encryption ones, by replacing $y_i = h(z_i, x_i)$ par $x_i = h^{-1}(z_i, y_i)$ as shown by the following equation (4)

$$d_k(x) = x_i = y_i \oplus z_i \qquad (13)$$

The key-stream is independent from the plaintext and expanded space from 26 letters to the 144 ASCII characters. Since there is an independence from the keys flow and the plaintexts and encrypted one, the proposed technique is considered synchronous flow, described by the triplet following equation:

$$\beta_{i+1} = C(\beta_i, K_{op}) \quad z_i = S(\beta_i, K_{iop}) \quad y_i = h(z_i, x_i) (14)$$

B. DESM-block cipher algorithm based on the technique modified DES

The second proposed algorithm is presented in the following section and participates in alternating the poly-encryption of a file is an evolution towards non-conventional approaches based on a synchronous block encryption. These algorithms derive their scrambling operation of data Feistels networks deemed by their permutations generating a statistical confusion processed text. The proposed algorithm is a Feistel network with 160 rounds, with key k 56-bit generating diversified 160 keys of 48-bit encoding blocks of 64 bits. It uses substitution boxes (S-box) fixed to introduce confusion. The proposed method is based on the structure of the DES algorithm modified to generate a new algorithm called DESM [5].

1) Steps of the proposed algorithm

The algorithm we propose, which is based on the structure of DES is a symmetric encryption algorithm block which can encrypt word of 64 bits from one key of 56 bits (56 bits used to encrypt, the $+8$ parity bits used to verify the integrity of the key). We keep the same structure but the permutation, transposition, the number of rounds and the number of generated random key will be changed [6].

2) Dynamic Partitioning permutation

We begin by splitting the message "plain text" which gives the partitioning into blocks of 64 bits followed by a dynamics transposition of each block unlike the DES algorithm whose implementation is static. A function of four successive rotations increment is applied to each block to change the position of its transposed elements. The first rotation is applied to the fourths central bits of the initial permutation table, the second rotation processes 16 bits excluding the central quad already permutated bits giving 12 bits processed. The third rotation rotates thirty six bits minus the central core 16 bits which gives 20 bits processed. While the fourth pivots all 64 bits of the initial permutation table excluding the table core of 36 bits which returns to pivot on the 28 bit located on the borders of the initial permutation table. The technique of permutation and quad successive rotations are shown by the following figures and equations. Let $T(x,y) / x \in [1:8]$ and $y \in [1:8]$ the initial permutation table and a function f composed of four sub-functions such as $f = \{f_1, \ldots, f_i\} / i \in [1:4]$. We consider the origin of the permutation table as the origin of the axes on the table while the rotations of the sub-matrices of the table will be made in the anti-clockwise as follows:

$$T_1(x,y) = \sum_{x=4}^{5} \sum_{y=4}^{5} T(x,y) \qquad (15)$$

$$T_2(x,y) = \sum_{x=3}^{6} \sum_{y=3}^{6} T(x,y) - \sum_{x=4}^{5} \sum_{y=4}^{5} T(x,y) \qquad (16)$$

$$T_3(x,y) = \sum_{x=2}^{7} \sum_{y=2}^{7} T(x,y) - \sum_{x=3}^{6} \sum_{y=3}^{6} T(x,y) \qquad (17)$$

$$T_4(x,y) = \sum_{x=1}^{8} \sum_{y=1}^{8} T(x,y) - \sum_{x=2}^{7} \sum_{y=2}^{7} T(x,y) \qquad (18)$$

The sub-functions of rotation is made by a step P. the following figures show a rotation with a step of P = 1. The transformation equations are as follows:

$$\left.\begin{aligned} T'_1(x,y) &= f_1^{P1}(T_1(x,y)) \\ T'_2(x,y) &= f_2^{P2}(T_2(x,y)) \\ T'_3(x,y) &= f_3^{P3}(T_3(x,y)) \\ T'_4(x,y) &= f_4^{P4}(T_4(x,y)) \end{aligned}\right\} \qquad (19)$$

For each block of 64 bits the dynamic permutation table changes. This change will continue until all the blocks composing the whole message are transposed as described by the following equations

$$T'_{Blck_1}(x,y) = f^{P1}(T(x,y)) \qquad (20)$$

$$T'_{Blck_2}(x,y) = f^{P2}\left(T'_{Blck_1}(x,y)\right) \qquad (21)$$

$$T'_{Blck_n}(x,y) = f^{P(n-4\times K+1)}\left(T'_{Blck_{n-1}}(x,y)\right) \qquad (22)$$

While $K \in \aleph$ is the last integer that divides n.

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

Figure 1. Initial permutation Table and its four cores

- blocks partition

Blocks are then divided into two blocks of 32 bits and noted G and D. We note by G_0 and D_0 the initial state of the two blocks.

- Expansion Function

The 32 bits of the D_0 block are extended to 48 bits with a table of expansion in which 32 bits are mixed together and 16 are duplicated. All bits are duplicated and dispersed to create a block of 48 bits that will be called D_0:

$$B_i = \left\{\begin{matrix} b_1, b_4, b_5, b_8, b_9, b_{12}, b_{13}, b16, b_{17}, b_{21}, b_{22}, \\ b_{24}, b_{25}, b_{28}, b_{29} \text{ and } b_{32} \end{matrix}\right\} \qquad (23)$$

- Logical Sum with the encryption key 48 bit

This is followed by applying XOR between D_0 and the first key K_1 generated from the key K (that must be shared transmitter and receiver) by the timing algorithm of the key which will be described below. We call D_0 the result of this operation.

- substitution box

Substitution is applied through eight "8" boxes substitutions used to reduce the size of blocks of 48 bits to 32 bits. Here's an example of an S-Box (S5). The output of 4 bits is obtained from the input of 6 bits. These 6 bits are divided in two parts: two bits at the ends and the four remaining bits (in the center). The two bits indicate the line and the central bits give the corresponding column. For example, with an input "011011" is divided into "0 1101 1" Which gives for the line «01» and column «1101». The output of the table is "1001".

		4 central bits of the inputs															
2 Bits Of the ends		0 0 0 0	0 0 0 1	0 0 1 0	0 0 1 1	0 1 0 0	0 1 0 1	0 1 1 0	0 1 1 1	1 0 0 0	1 0 0 1	1 0 1 0	1 0 1 1	1 1 0 0	1 1 0 1	1 1 1 0	1 1 1 1
	0 0	0 0 1 0	1 1 0 0	0 1 0 0	0 0 0 1	0 1 1 1	1 0 1 0	1 0 1 1	0 1 1 0	1 0 0 0	0 1 0 1	0 0 1 1	1 1 1 1	1 1 0 1	0 0 0 0	1 1 1 0	1 0 0 1
	0 1	1 1 1 0	1 0 1 1	0 0 1 0	1 1 0 0	0 1 0 0	0 1 1 1	1 1 0 1	0 0 0 1	0 1 0 1	0 0 0 0	1 1 1 1	1 0 1 0	0 0 1 1	1 0 0 1	1 0 0 0	0 1 1 0
	1 0	0 1 0 0	0 0 1 0	0 0 0 1	1 0 1 1	1 1 0 0	1 1 0 1	0 1 1 1	1 0 0 0	1 1 1 1	1 0 0 1	1 1 0 0	0 1 0 1	0 1 1 0	0 0 1 1	0 0 0 0	1 1 1 0
	1 1	1 0 1 1	1 0 0 0	1 1 0 0	0 1 1 1	0 0 0 1	1 1 1 0	0 0 1 0	1 1 0 1	0 1 1 0	1 1 1 1	0 0 0 0	1 0 0 1	1 0 1 0	0 1 0 0	0 1 0 1	0 0 1 1

Figure 2. Une s-box de substitution.

We apply, in parallel, 8 boxes (fixed function) of 6 bits to 4 bits. This reduces the internal state of 8 × 6 = 48 bits to 8 bits × 4 = 32 bits.

3) Permutations

Once the substitution is done a change in position of the result according to a permutation function causes a disorder of the "32" bit by the following table:

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

Figure 3. Permutation Table

- Logic sum using XOR

The obtained blocks composed by 32 bits are summed using the XOR logic functions with G_0 to give D_1 and the same think with the D_0 block that outputs G_1.

- Initial inverse transposition

The initial proposed algorithm at the end of the sixteen iterations, two blocks G_{16} and D_{16} are "glued" to reform a single block of 64 bits then undergoes initial inverse transposition. Our algorithm offers 64 continuous iterations, which generates at the end of the round two blocks G_{160} and D_{160}. The final text will be from gluing the two blocks. In the initial proposed algorithm and at the end of the sixteen iterations, two blocks G_{16} and D_{16} are "glued" to reform a single block of 64 bits then undergoes initial inverse transposition. Our algorithm offers 64 continuous iterations, which generates at the end of the round two blocks G_{160} and D_{160}. The final text will be from joining the two blocks.

- Algorithm of key generating

The key K of length 64 bits contains 8 bits for controls to ensure the integrity of information. So far its useful part is 56 bits. The key K provides 160 partial key $K_1,....,K_{160}$ 48 bits each one to be used in each of the 160 rounds that replace the 16 rounds used in the old algorithm. At first the key parity are eliminated to obtain key of 56 bits. This block then undergoes a permutation is divided into two 2 blocks of 28 bits. These two blocks are rotated to the left, that is to say that the bits in the second position take the first position, and those in the third position take the second, and so on. The blocks are grouped to a block of 56 bits that passes through a permutation providing a block of 48 bits representing the key k_i. These iterations provide the 160 keys used in the proposed algorithm.

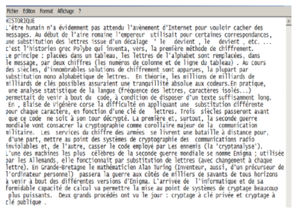
III. IMPLEMENTATION OF AN INTERFACE FOR MULTI-ENCRYPTION FILE

The application of these two encryption algorithms associated together to encrypt alternatively a plain text file increases security and minimizes the possibility of deciphering the secret text. The plaintext is partitioned into subsets of words. The collected subsets are swapped in the same text with a simple offset and for each one of the subsets one between the proposed techniques is applied. The choice of the technique is based on a key is composed of two elements $K=\{k_1,k_2\}/k_1\in[1:9]$ and $k_2\in[1:9]$. Each element of the key indicates the number of periods with which one of the two algorithms is applied. The following figure and equations illustrate the operation of partitioning the plaintext and the implementation of encryption algorithms. In the case where k_1=1 et k_2=1 the sub-sets of the plain text are alternatively encrypted with the same period by the algorithms noted AL_1 and AL_2. Considering M the file text, M_c the encrypted text, and M_i the sub-sets, the following equation resumes the encryption step:

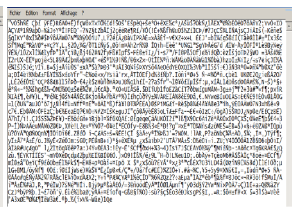
$$M_c=\{AL_1(M_1),..,AL_1(M_{2t}),AL_2(M_{2t+1})\} \quad (24)$$

such as $j=2t+1$

The following figures give an overview of poly-encrypting a plain text with an example of an original file and the result of its encryption.



```
HISTORIQUE
L'être humain n'a évidemment pas attendu l'avènement d'Internet pour vouloir cacher des
messages. Au début de l'aire romaine l'empereur  utilisait pour certaines correspondances,
une substitution des lettres issue d'un décalage  ' le   devient , le   devient , etc. ...
c'est l'historien grec Polybe qui inventa, vers, la première méthode de chiffrement.
Le principe : placées dans un tableau, les lettres de l'alphabet sont remplacées, dans
le message, par deux chiffres (les numéros de colonne et de ligne du tableau) . Au cours
des siècles, d'innombrables solutions de chiffrement sont apparues, la plupart par
substitution mono alphabétique de lettres.  En théorie, les millions de milliards de
milliards de clés possibles assuraient une tranquillité absolue aux codeurs.En pratique,
 une analyse statistique de la langue (fréquence des lettres, caractères isolés...)
permettait de venir à bout du  code, à condition de disposer d'un texte suffisamment long.
 En , Blaise de Vigénère corse la difficulté en appliquant une  substitution différente
pour chaque caractère, en fonction d'une clé de   lettres. Trois  siècles passeront avant
 que ce code  ne soit à son tour décrypté. La première et, surtout, la seconde guerre
mondiale vont consacrer la cryptographie comme corollaire majeur de la  communication
militaire.  Les  services du chiffre des armées  se livrent une bataille à distance pour,
 d'une part, mettre au point des systèmes de cryptographie des  communications radio
inviolables et, de l'autre, casser le code employé par Les ennemis (la 'cryptanalyse').
L'une des machines les plus  célèbres de la seconde guerre mondiale se nomme Enigma ; utilisée
par les Allemands, elle fonctionnait par substitution de lettres (avec changement à chaque
lettre). En Grande-Bretagne le mathématicien Alan Turing (Inventeur, aussi, d'un précurseur de
l'ordinateur personnel)  passera la guerre aux côtés de milliers de savants de tous horizons
à venir à bout des différentes versions d'Enigma. L'arrivée de  l'informatique et de sa
formidable capacité de calcul va permettre la mise au point de systèmes de cryptage beaucoup
 plus puissants.  Deux grands procédés ont vu le jour : cryptage à clé privée et cryptage à
clé publique .
```



```
[illegible]
```

Figure 4. Plain text and its poly-encryption by LSCE and DESM with k1=2 et k2=3

IV. CONCLUSION

In this paper, a new approach for text and text file poly-encryption is presented. This approach is composed by two techniques, the first is based on stream cipher symmetric encryption methods and the second uses a block cipher encryption structure. These two techniques are complex, the first named LSCE based on random key logic encryption independent of the text and the second named DESM inspired from the DES algorithm with a modification allowing it to be more complex and secure. These two techniques are combined to encrypt alternately a text partitioned into a set of words. Each set is encrypted by one of the two algorithms with a key that supervises the redundancy of these sets encryption. The resulting text is poly-encrypted by two different techniques which make the possibility of deciphering it more difficult.

REFERENCES

[1] Schlesinger, R., "A Cryptography Course for Non-Mathematicians", 1st annual conference on information security curriculum development, Kennesaw, Georgia, October 08-08,2004.

[2] Maurer, .U, "The Role of cryptography in Database Security", proceeding of the 2004ACM SIGMOD international conference on Management of data (SIGMOD'04), PP5-10, ISBN: 1-58113-859-8, Paris, France June 13- 18-2004.

[3] Anan, .T, Kuraki. K, Takahashi. J, "Paper Encryption Technology", Fujitsu Sci, Tech j, Vol 46, No.1, pp.87-94, Yan. 2010

[4] Y. Mao G. Chen, S. Lian, "A Novel Fast Image Encryption Scheme Based On 3D Chaotic Baker Maps", International Journal of Bifurcation and Chaos, Vol. 14, No. 10 (2004) 3613-3624. http://dx.doi.org/10.1142/S021812740401151X

[5] Ali Bani Younes, and Jantan, .A, "Image Encryption Using Block-Based Transformation Algorithm", IAENG International Journal of Computer Science, 35:1, IJCS_35_1_03.

[6] Xu, .X Dexter, S. and Eskicioglu, A. M., "A hybrid scheme for encryption and watermarking", IEEE Journal on Selected Areas in Communications 18 (2000) 850–. 860. 13. http://dx.doi.org/10.1109/49.848239

AUTHOR

Hassene seddik is with the ENSIT school of Tunisia, Unité de recherche CEREP, hassene.seddik@esstt.rnu.tn. He is born in 15 October 1970 in Tunisia, he has obtained the electromechanical engineer degree in 1995 and followed by the master degree in "signal processing: speaker recognition" and the thesis degree in image processing "watermarking using non conventional transformations". He has over

14 international journals papers and 65 conference papers. His domain of interest is: Audio-image and video processing applied in filtering, encryption and watermarking. He belongs to the CEREP research unit and supervises actually five thesis and 08 masters in the field.

Submitted 10 June 2013. Published as re-submitted by the author 23 July 2013.