Ibn Al Haitham Journal for Pure and Applied Science

Journal homepage: http://jih.uobaghdad.edu.iq/index.php/j/index

Proposed A Permutation and Substitution Methods of Serpent Block Cipher

Intisar Abid Yousif

Department of Computer, College of Education, University of Mustansiriyah

int_abd@uomustansiriyah.edu.iq

Article history: Received 19 February 2019, Accepted 21 April 2019, Publish May 2019

Doi: 10.30526/32.2.2120

Abstract

Block cipher technique is one of cryptography techniques to encrypt data block by block. The Serpent is one of AES candidates. It encrypts a 128-bit block by using 32 rounds of a similar calculation utilizing permutations and substitutions. Since the permutations and substitutions of it are static. Then this paper proposes dynamic methods for permutation, substitution and key generation based on chaotic maps to get more security. The proposed methods are analyzed and the results showed that they were able to exceed the weakness resulting from the use of static permutations and substitutions boxes in the original algorithm and also can reduce number of rounds and time usage compared with a classical Serpent block cipher algorithm.

Keywords: Serpent block cipher, initial and final permutation, substitutions boxes, key generation, logistic chaos map, standard chaos map.

1. Introduction

Chaos theory is an awesome dynamic part in current cryptography. The principle highlights of utilizing chaos theory in numerous cryptosystems can be finished up on its affectability to starting conditions parameter and irregular conduct settings that accomplish the essential Shannon prerequisites of disarray and dissemination. Among them, a great deal of calculations in view of disorganized hypothesis endorsed its quality in many concerned angles with respect to security, execution speed, intricacy, control utilization and computational overhead, and so forth. These days, some turmoil based picture encryption cryptosystems and irregular number age calculations in view of discrete bedlam are proposed, however its security is for the most part not affirmed [1,2]. For some applications, the Data Encryption Standard (DES) algorithm endorsed its shortcoming against a great deal of cryptanalytic assault; therefore require a safe calculation has been issued by the US National Institute of Standards and Technology, to be known as the Advanced Encryption Standard (AES). The primary highlights of the proposed calculation that it must be both quicker and more secure than triple DES, it should likewise bolster a 128 bit block length and a 128, 192, and 256 bit key length [3]. The block cipher Serpent was outlined by Anderson, Biham and

Knudsen to be contender for AES [4]. while, Daemen and Rijmen outlined the block cipher Rijndael. Serpent and Rijndael cipher were in an opposition between the last 5 finalist algorithms. The block cipher Rijndael standardized by NIST as the AES algorithm [5]. The security of AES has been submitted to present [6]. since its acknowledgment as an AES, closed equation for AES has been found by Ferguson, Shroeppel and Whiting which can be considered as a promotion of proceeded divisions [7,8]. Courtois and Pieprzyck see that the byte substitution change which utilized in the AES calculation can be determined by various certain quadratic Boolean conditions. Recently, there are numerous viable extensions in the cryptanalysis of AES. The AES applies a wide-trail configuration arranging [4, 9]. This was at first proposed on account of its quality against ground-breaking cryptanalytic systems.

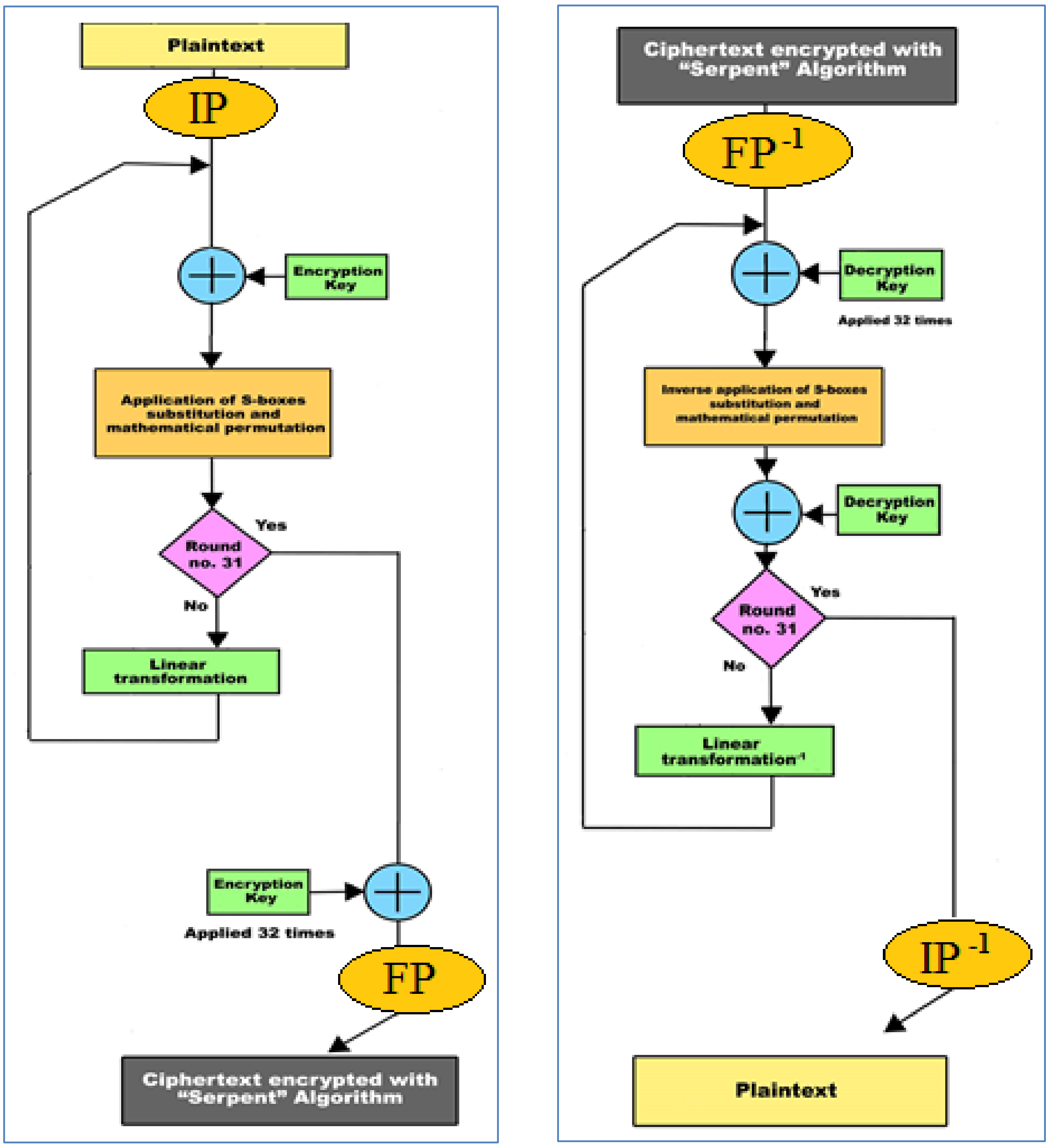
This paper presents a proposed modification of Serpent algorithm by replacing the static permutation and substitution with dynamical properties using logistic chaos map and standard map. The organization of this paper is chaos based cryptography in section 2,serpent block cipher in section 3,overview of logistic map and standard map in section 4,the proposed dynamic permutation and substitution methods in section 5,Security and Statistical analysis in section 6 and the rest sections are conclusion.

2. Chaos and cryptography

It is a modern study that many researchers towards it [10-12]. The cryptography techniques are stream cipher and block cipher. Stream cipher algorithms are developed with using chaos maps as (Pseudo Random Bit Generator) PRBG to generate keystream.[13-17]. S-box of block cipher algorithms are developed with using chaos maps [18-22].This paper study and developed permutation box (IP & FP), S-box of serpent block cipher using chaos map, and key round generation.

3. Serpent block cipher

Serpent is a symmetric block cipher and has Substitution - Permutation network (SPN) design. It has 32 rounds and each round requires its special 128 bit round key which is generated by a key schedule. The encrypting and decrypting phase have the same level of complexity. The decryption operations are exactly the inverted transformations used to encrypt the message but in opposite order Serpent uses different mathematical substitutions "S-boxes" with a 4 bit entrance and a 4 bit exit. Every encryption phase uses an S-box that work collaterally for the 32 times [23]. **Figure 1,** shows the encryption and decryption algorithms for Serpent block cipher.


Plaintext
IP
Encryption Key
Application of S-boxes substitution and mathematical permutation
Round no. 31
Yes
No
Linear transformation
Encryption Key
Applied 32 times
FP
Ciphertext encrypted with "Serpent" Algorithm
Ciphertext encrypted with "Serpent" Algorithm
FP^-1
Decryption Key
Applied 32 times
Inverse application of S-boxes substitution and mathematical permutation
Decryption Key
Round no. 31
Yes
No
Linear transformation^-1
IP^-1
Plaintext

Encryption process Decryption process

Figure 1. Serpent block cipher: Encryption and Decryption process.

At first, it accepts a 128 bit of plaintext and permutes these bits by fixed initial permutation (IP) which is shown in **Table 1.** The resulted bits input to 32 rounds, each round has three processes: - XOR with round key, substitution process by one of eight S-boxes as shown in **Table 2,** depend on round number (i) mod 8, linear transformation process as shown in **Figure 2,** with its algorithm. After last round there is XOR process with round key k_{32} and do permutation of the result 128 bit by fixed final permutation (FP) as shown in **Table 3,** to get a ciphertext [24]. In decryption algorithm reverses an encryption block diagram processes to recover a plaintext [25].

Table 1. Initial permutation (IP).

0	32	64	96	1	33	65	97	2	34	66	98	3	35	67	99
4	36	68	100	5	37	69	101	6	38	70	102	7	39	71	103
8	40	72	104	9	41	73	105	10	42	74	106	11	43	75	107
12	44	76	108	13	45	77	109	14	46	78	110	15	47	79	111
16	48	80	112	17	49	81	113	18	50	82	114	19	51	83	115
20	52	84	116	21	53	85	117	22	54	86	118	23	55	87	119
24	56	88	120	25	57	89	121	26	58	90	122	27	59	91	123
28	60	92	124	29	61	93	125	30	62	94	126	31	63	95	127

Table 2. S-box.p.

S0:	3	8	15	1	10	6	5	11	14	13	4	2	7	0	9	12
S1:	15	12	2	7	9	0	5	10	1	11	14	8	6	13	3	4
S2:	8	6	7	9	3	12	10	15	13	1	14	4	0	11	5	2
S3:	0	15	11	8	12	9	6	3	13	1	2	4	10	7	5	14
S4:	1	15	8	3	12	0	11	6	2	5	4	10	9	14	7	13
S5:	15	5	2	11	4	10	9	12	0	3	14	8	13	6	7	1
S6:	7	2	12	5	8	4	6	11	14	9	1	15	13	3	10	0
S7:	1	13	15	0	14	8	2	11	7	4	12	10	9	3	5	6

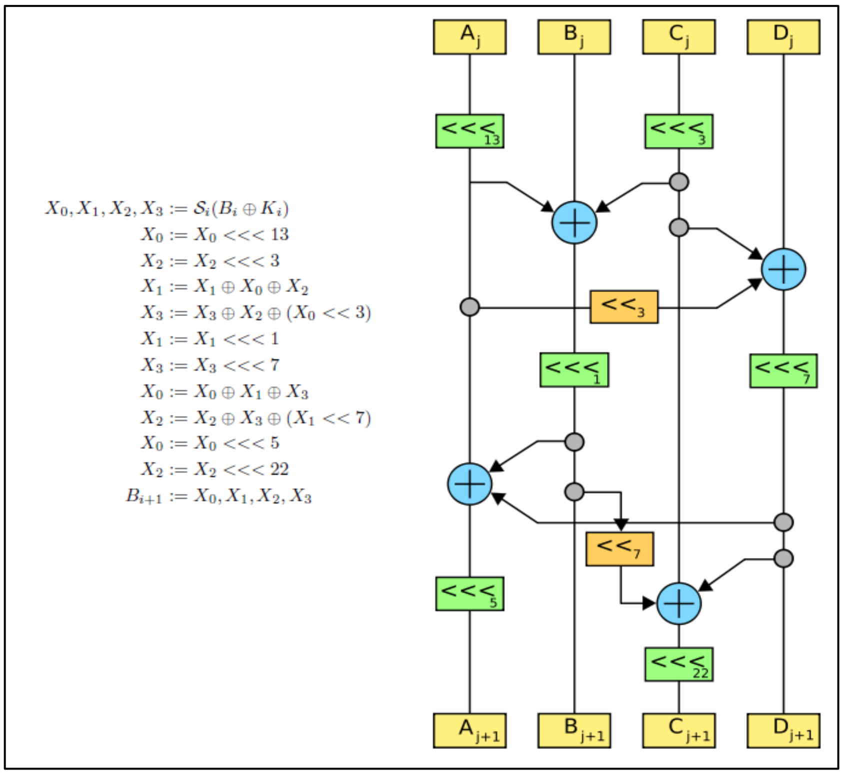

$$
\begin{aligned}
X_0, X_1, X_2, X_3 &:= \mathcal{S}_i(B_i \oplus K_i)\\
X_0 &:= X_0 <<< 13\\
X_2 &:= X_2 <<< 3\\
X_1 &:= X_1 \oplus X_0 \oplus X_2\\
X_3 &:= X_3 \oplus X_2 \oplus (X_0 << 3)\\
X_1 &:= X_1 <<< 1\\
X_3 &:= X_3 <<< 7\\
X_0 &:= X_0 \oplus X_1 \oplus X_3\\
X_2 &:= X_2 \oplus X_3 \oplus (X_1 << 7)\\
X_0 &:= X_0 <<< 5\\
X_2 &:= X_2 <<< 22\\
B_{i+1} &:= X_0, X_1, X_2, X_3
\end{aligned}
$$

Figure 2. Linear transform (LT).

Table 3. Final permutation (FP).

0	4	8	12	16	20	24	28	32	36	40	44	48	52	56	60
64	68	72	76	80	84	88	92	96	100	104	108	112	116	120	124
1	5	9	13	17	21	25	29	33	37	41	45	49	53	57	61
65	69	73	77	81	85	89	93	97	101	105	109	113	117	121	125
2	6	10	14	18	22	26	30	34	38	42	46	50	54	58	62
66	70	74	78	82	86	90	94	98	102	106	110	114	118	122	126
3	7	11	15	19	23	27	31	35	39	43	47	51	55	59	63
67	71	75	79	83	87	91	95	99	103	107	111	115	119	123	127

4. Overview of logistic map and standard map

- **Logistic map:** This paper used logistic chaotic map as the following equation for developed IP and FP permutation:

$$X_{n+1} = rx_n(1 - x_n) \tag{1}$$

Where r range is ($3.57 < r \leq 4$) that become chaotic. For example, using x_0 = 0.2, and r = 3.738 as **Figure 3.**

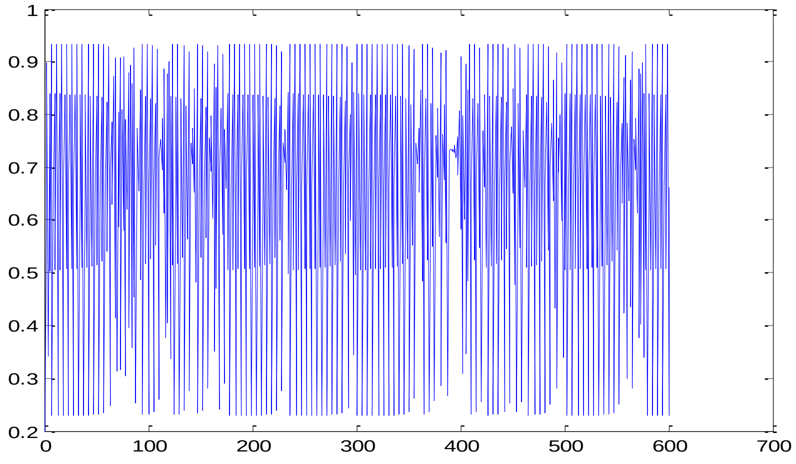


Figure 3. Logistic map signal.

- **Standard map:** This paper used Standard chaotic map as the following equation for developed S-box:

$$X_{n+1} = X_n + \text{k} \sin y_n \mod 2\pi \tag{2}$$
$$y_{n+1} = y_n + X_{n+1} \mod 2\pi \tag{3}$$

Where X_n and $y_n \in [0,2\pi]$, and ($k \geq 18.9$).

5. The proposed dynamic permutation and substitution methods

A. The proposed Dynamic Initial Permutation (DIP):

This section shows a distribution of fixed IP serpent as **Figure 4.** This figure shows that no random in distribution and the attacker can re-permute in easy. So that this paper suggests use random chaotic map to permute IP with a special key for each encryption process.

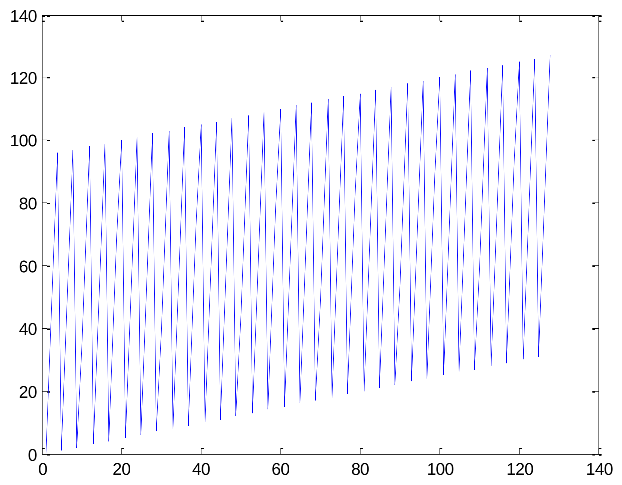


Figure 4. The distribution of IP.

To permute IP using logistic map by ascending or descending the generated values of map to get IP with a new random distribution as **Table 4.** and **Figure 5.**

The following algorithm shows the method of proposed Initial permutation (DIP):

Algorithm 1

Input:

- The table of Initial permutation (IP)
- The initial values of logistic map (r, n , x(1))

Output:

- A new random distribution of IP (DIP)

Begin

1. Save the table of IP in vector with length 128.
2. Generate a sequence of random values from logistic map with length 128 as index of IP vector.
3. Sort in ascending of these values to get a change of IP vector locations.

Table 4. Permute IP with logistic map.

0	97	3	36	69	102	8	41	74	107	87	31
13	120	46	26	79	117	59	51	49	50	30	96
116	29	52	93	112	53	23	33	66	99	5	38
71	104	10	43	94	76	56	127	109	89	15	122
114	113	32	92	115	48	86	2	35	68	101	7
40	73	106	12	119	63	45	25	78	58	28	111
22	91	123	54	16	83	60	81	82	90	110	57
77	95	24	44	62	11	105	72	39	6	100	67
34	1	118	21	80	61	20	124	84	64	125	18
17	19	27	85	47	121	14	88	108	126	55	75
42	9	103	70	37	4	98	65				

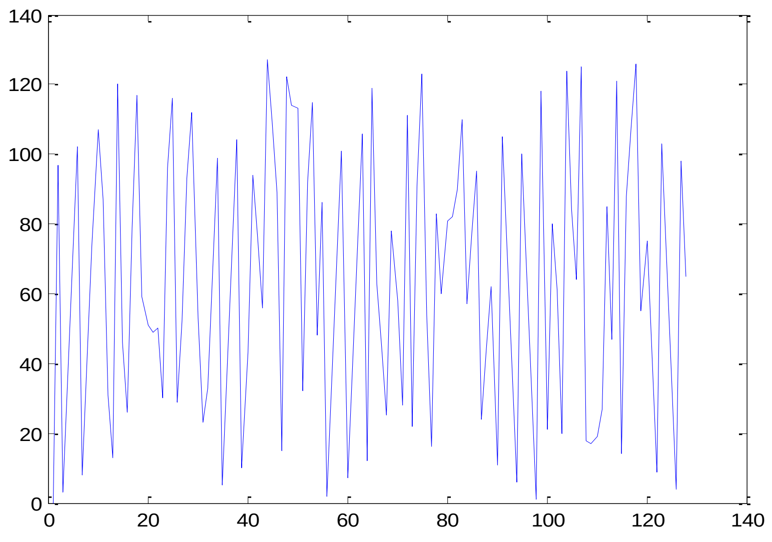


Figure 5. Distribution of proposed dynamic IP.

B. The proposed dynamic Final Permutation (DFP):

This section shows a distribution of fixed FP serpent as **Figure 6.** This figure shows that no random in distribution and the attacker can re-permute in easy. So that this paper suggests use random chaotic map to permute FP with a special key for each encryption process.

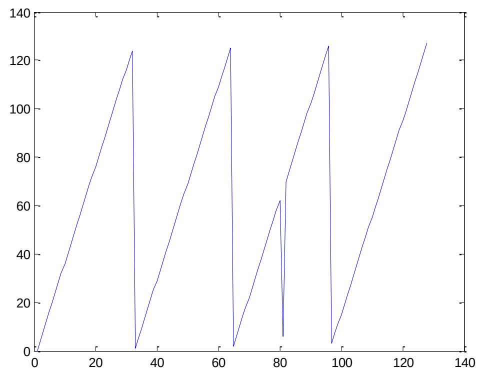


Figure 6. Distribution of FP.

To permute FP using logistic map by ascending or descending the generated values of map to get FP with a new random distribution as **Table 5,** and **Figure 7.** The following algorithm shows the method of proposed final permutation (DFP):

Algorithm 2

```
Input:
    - The table of final permutation (FP)
    - The initial values of logistic map (r, n , x(1))
Output:
    - A new random distribution of FP (DFP)
Begin
        1. Save the table of FP in vector with length 128.
        2. Generate a sequence of random values from logistic map with length 128 as index
           of FP vector.
```

3. Sort in ascending of these values to get a change of FP vector locations.

Table 5. Permute FP with logistic map.

0	28	48	68	88	108	1	21	41	61	122	115
81	15	101	35	121	94	55	54	22	38	99	12
78	83	70	91	14	86	114	20	40	60	80	100
120	13	33	53	107	73	7	127	93	27	113	47
46	30	4	75	62	6	106	32	52	72	92	112
5	25	45	65	126	119	85	19	105	39	67	125
98	59	63	102	2	58	71	26	42	43	109	23
89	123	3	69	103	49	29	9	116	96	76	56
36	16	110	82	10	87	6	79	74	8	95	34
18	50	51	90	117	31	97	11	77	111	118	57
37	17	124	104	84	64	44	24				

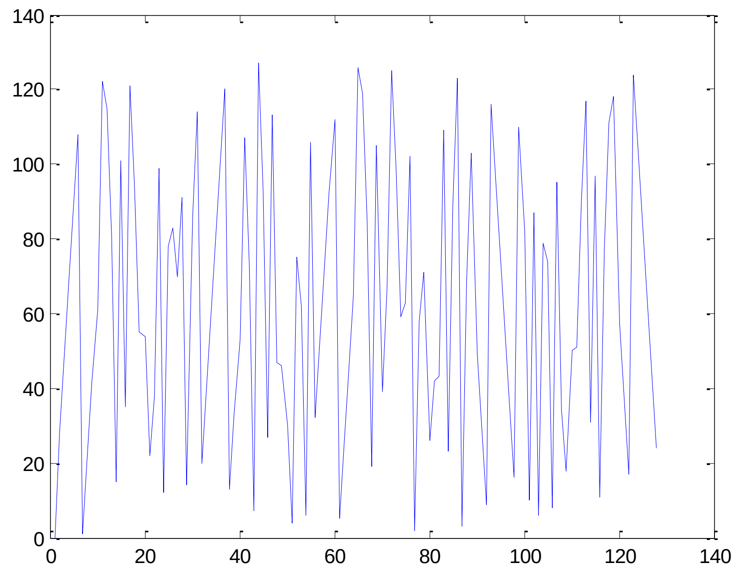


Figure 7. Distribution of proposed dynamic FP.

• The Implementation Example

A plaintext is “*Mary had a little lamb*”. A 128 bit of first 16 characters is

```
0100110101100001011100100111100100100000011010000110000101100100001000000110000100100000001
1011000110100101110100011101000110110001100101001000000110110001100001011011010101100010
```

IP of these bits is

```
0000100101110000100110000000100100001111111100010100000100001010000011011111100100000000110
00010000001111111110001011011100001000
```

With a simple statistical test to know the features of the result bits:
Autocorrelation test = -2.75 pass
Poker test= 17.4878306878307 pass
Serial test= fail
Run test=0.0060, h=1 (no random)

Permute the result IP with chaotic logistic map as following binary string:

0100000111010001110011000001000010100010000011111101100000010110000101101000000001101000100
10001011000101001111100100001011100110

With a simple statistical test to know the features of the result bits compared with previous IP bits. The test shows the proposed IP has best features:
Autocorrelation test = -0.25 pass
Poker= 7.87089947089947 pass
Serial = 5.51402559055117 pass
Run test=0.5974, h=0 (random)

Example of result original FP as following binary string:

0010110010011100001010001001001010101100000110010100100011011011000000000110
10001010000110011011101110010000110101000110001100000

With a simple statistical test to know the features of the result bits:
Autocorrelation test = 0.25 pass
Poker= 7.73544973544973 pass
Serial =4.75812007874015 pass
Run test=0.4463, h=0 (random)

Permute the result FP with chaotic logistic map as following binary string:

0010011101101110110001000110000100101000101001111000101111000010100000110100000000010000111
01010001100100010100110011001001000010

With a simple statistical test to know the features of the result bits compared with previous FP bits. The test shows the proposed FP has best features:
Autocorrelation test = - 0.25 pass
Poker= 10.4444444444444 pass
Serial = 4.56914370078741 pass
Run = 0.6962, h =0 (more random)

C. The proposed dynamic substitution box

The following algorithm shows the method of proposed dynamic substitution box (S-box) rather than fixed table.

Algorithm 3

```
Input:
    -   Eight S-boxes table ($S_0$ to $S_7$)
    -   The initial values of standard map (n, k, x(1), y(1))
Output:
    -   A new random distribution of eight S-boxes
Begin
        1. Save the table of each of eight S-boxes into eight vectors each with length 8.
        2. Generate a sequence of random values from standard map with length 8 as
           index of each S-box vector. Where apply $X_{n+1}$ with even S-boxes ($S_0$, $S_2$, $S_4$,
           $S_6$) and $Y_{n+1}$ with odd S-boxes ($S_1$, $S_3$, $S_5$, $S_7$).
        3. Sort in ascending of the even S-box values to get a change of vector locations.
        4. Sort in descending of the odd S-box values to get a change of vector locations.
```

D. The proposed round keys

The following algorithm shows the method of proposed dynamic two round keys for each round based on chaotic standard map.

Algorithm 4

```
Input:
    -   The initial values of standard map (n, k, x(1), y(1))
Output:
    -   A new two round keys for each round
Begin
        1. Implement While Loop (no. round <n), where n =10 in the proposed method
           rather than 32 of classical serpent.
           a. Generate a sequence of random values from standard map. Where apply
              $X_{n+1}$ to generate 1st round key and $Y_{n+1}$ to generate 2nd round key.
           b. Rounding these values to the nearest integer.
           c. Convert the integer values into binary sequence.
           d. Pick 1st 128 bits as 1st round key
           e. Pick 2nd 128 bits as 2nd round key
           f. Change the value of initial values of chaotic map to generate new sequence
              for next round (i.e.) X(n+1)=x(n)+0.1 and y(n+1)=y(n)+0.1
    End of While Loop
```

Flowchart of the proposed modification Serpent

Figure 8, shows the proposed of encryption method that describe by previous sections of proposed algorithms.

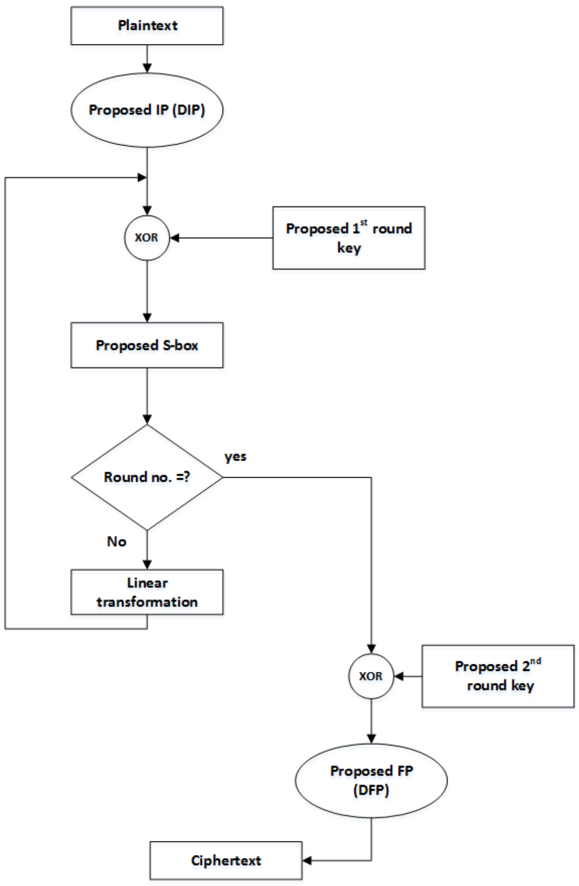


Figure 8. Flowchart of proposed encryption method.

6. Security and Statistical Analysis

- **Brute Force Attack**

Key space size is the total number of different keys that can be used for the encryption algorithm. It should be large enough to resist the brute force attack. The precision of initial values and parameter of the proposed method is (10^{16}). The key space size of secret keys is $(10^{16})^{K}$. The number of initial values and parameters is k = 3 of logistic map (r, n , x(1)) and k= 4 of standard map (n, k, x(1), y(1)). So, the total k value is 7. Then the key space size is $(10^{16})^{7}$ so that the at least key space size of our proposed is (10^{112}). It is large enough against brute force attack.

- **Key Sensitivity**

An ideal encryption algorithm should be sensitive to secret key that means the encrypted method cannot be decrypted correctly, if there is only a slight change in the secret key. The proposed method has a key sensitivity when changing only one parameter of the secret keys and keeping all other keys unchanged. The secret keys are (r, n, x(1), y(1), and k).

- **NIST Tests**

The average of 100 NIST tests between a classical serpent and the proposed modified as following **Table 6,** of NIST statistical tests. This table shows that the proposed method has best randomness compared with a classical serpent. And also can reduce the number of round to 10 rounds rather than 32 so the result is the best randomness and decrease the time average.

Table 6. NIST statistical tests.

No.	NIST test	Classical Serpent with 32 round	Modified Serpent with 10 round	Modified Serpent with 32 round
1	Frequency	0.5022	0.8025	0.8814
2	Block frequency M=128	0.8818	0.9997	0.9335
3	Runs	0.0155	0.0169	0.0161
4	Longest Runs of Ones	0.0247	0.0437	0.0446
5	Rank	0.1576	0.1559	0.1916
6	FFT	0.2629	0.3018	0.5153
7	NonOverlapping Templates m=9	0.3260	0.4872	0.4301
8	Overlapping Templates all ones B='111111111'	0.0223	0.0351	0.0373
9	Universal statistical L=6 & Q=1000	0.6864	0.6532	0.6973
10	Linear complexity M=500	0.1842	0.4087	0.7816
11	Serial m=12	0.3310	0.3656	0.5969
		0.5001	0.6851	0.6142
12	Approximate Entropy m=8	0.1365	0.1515	0.1732
13	Cumulative Sums (Forward)	0.5219	0.6784	0.9161

- **Histogram Analysis and Other Tests**

Figure 9, shows the implement of proposed encryption for baboon image. Note that a good result of the flat histogram for encryption image. The result of other analysis tests is good such as (entropy =7.9989). The tests against differential attack (NPCR=99.8022, and UACI= 33.3382).

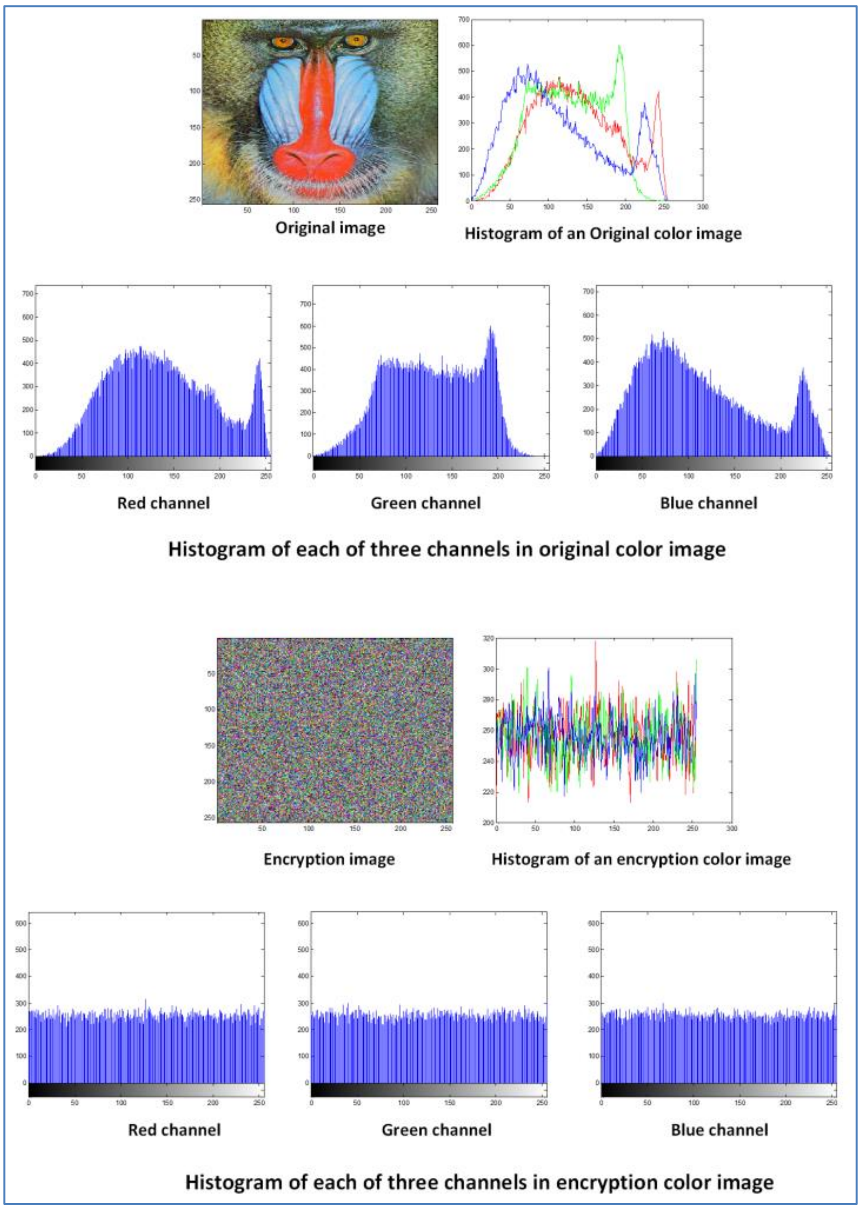


Figure 9. Histogram analysis of proposed encryption.

7. Conclusion

The proposed dynamic initial permutation (IP), final permutation (FP), Substitution box (S-box), and the generated round keys give the best randomness compared with classical serpent algorithm and also can reduce the number of round and time usage. The proposed method has sensitivity to any change in the key since it uses chaotic map in the key round generation. And also, it more robust than classical method since it used another chaotic map to create dynamic permutation and substitution.

References

1. Baptista, M.S. Cryptography with chaos. Phys. Lett. A.**1998**, *240*, *1*, 50-54.
2. Xiaojun, T.; Minggen, C.A. Novel Image Encryption Scheme Based on Feedback and 3D Baker. 4th International Conference on Wireless Communications. **2008**
3. Ross, A.; Eli, B.; Lars, K. Serpent: A Proposal for the Advanced Encryption Standard. The first AES candidate conference.**1999**

4. Joan, D.; Vincent, R. The block cipher Rijindael. The Third International Conference on Smart Card Research and Applications, CARDIS'98. Lecture Notes in computer Science (LNCS).**1998**, *18, 20*, 277-284.
5. Morris, J.D.; Elaine, B.B.; James, R.N.; James, F.; Lawrence, E.B.; Roback, E.; James, F.D. Advanced Encryption Standard. Federal Information Processing Standards Publications (FIPS 197). **2001**, 8-18.
6. Eli, B.; Dunkelman, O.; Nathan, K. Related key impossible differential attacks on AES-192. CT-RSA'06, LNCS. **2005**, *3860*, 21-33.
7. Niels, F.; John, K.; Stefan, L.; Bruce, S.; Mike, S.; David, W.; Doug, W. Improved Cryptanalysis of Rijndael. proceedings of FSE, LNCS. **2002**, *1978*, 213-230.
8. Nicolas, T.C.; Josef, P. Cryptanalysis of block ciphers with over defined systems of equations. *ASIACRYPT'02, LNCS*. **2002**, *2501*, 267-287.
9. Joan, D.; Vincent, R. The Wide Trail Design Strategy. LNCS. **2001**, *2260*, 222-238.
10. AlSaffar, A. Cryptography with Dynamic DNA Depending on Edge Detection. *Ibn Al-Haitham J. for Pure & Appl. Sci.* **2018**, *31*, *2*,186-192.
11. Ghada, S.M. Text Encryption Algorithm Based on Chaotic Neural Network and Random Key Generator. *Ibn Al-Haitham J. for Pure & Appl. Sci.* **2016**, *29, 3*, 222-233.
12. Enas, W.A. Combining a Hill Encryption Algorithm and LSB Technique With Dispersed Way for Securing Arabic and English Text Messages Hidden in Cover Image. *Ibn Al-Haitham J. for Pure & Appl. Sci.* **2017**, *30*, *2*, 214-223.
13. Hassan, M.E.; Ali, E.T.; Mahmoud, A.S.A. Modified Serpent Based Algorithm for Image Encryption. *35th National Radio Science Conference (NRSC 2018)*. International University (MIU), Cairo, Egypt. **2018**
14. Kanso, A.; Smaoui, N. Logistic chaotic maps for binary numbers generations. *Chaos, Solitons, and Fractals*.**2009**, *40*, 5, 2557‑2568.
15. Badar, S.; Saad, R. Proposed random unified chaotic map as PRBG for voice encryption in wireless communication. Elsevier. Procedia computer science.**2015**, *65*, 314-323.
16. Patidar, V.; Sud, K.; Pareek, N.A. Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing. *Informatica*.**2009**, *33*, 441‑452.
17. Patidar, V.; Sud, K.A. Novel Pseudo Random Bit Generator Based on Chaotic Standard Map and its Testing. *Electronic Journal of Theoretical Physics*.**2009**, *6*, *20*, 327-344.
18. Xingyuan, W.; Xiaojuan, W.; Jianfeng, Z.; Zhenfeng, Z. Chaotic encryption algorithm based on alternant of stream cipher and block cipher. Springer. *Nonlinear Dynamics*. **2011**, *63*, *4*, 587‑597.
19. Asim, M.; Jeoti, V. Efficient and Simple Method for Designing Chaotic S-Boxes. *ETRI Journal*.**2008**, *30*, *1*, 170‑172.
20. Tawfiq, L. N. M.; Eqhaar, Q.H. On Multilayer Neural Networks and Its Application for Approximation Problem. *3rd scientific conference of the College of Science. University of Baghdad*. **2009**.
21. Zaibi, G.; Peyrard, F.; Kachouri, A.; Fournier, D.; Samet, M. A new design of dynamic S-Box based on two chaotic maps. *ACS/IEEE International Conference on Computer Systems and Applications – AICCSA*. **2010**.

22. Jeyamala, C.; Subramanyan, B.; Raman, G. Ensembles of blowfish with chaos based S-box design for text and image encryption. [IJNSA]. *International Journal of Network Security & Its Applications (IJNSA)*.**2011**, *3*, *4*, 165-173.
23. Ashtiyani, M.; Birgani, P.; Madahi, S. Speech Signal Encryption Using Chaotic Symmetric Cryptography". *J. Basic. Appl. Sci. Res.***2012**, *2*, *2*, 1678‑1684.
24. Tawfiq, L. N. M.. Using Collocation Neural Network to Solve Eignvalue Problems. *MJ Journal on Applied Mathematics*. **2016**, *1*, *1*, 1–8.
25. Sugier, J. Implementing Serpent Cipher in Field Programmable Gate Arrays. PolandICIT *The 5th International Conference on Information Technology*. **2011**.