Journal of Soft Computing and Computer Applications

Volume 1 | Issue 1 Article 1003

2024

A Novel Approach to Generate Dynamic S-Box for Lightweight Cryptography Based on the 3D Hindmarsh Rose Model

Ala'a Talib Khudhair
Computer Science Department-University of Technology - Iraq, Baghdad, Iraq,
cs.22.20@grad.uotechnology.edu.iq

Abeer Tariq Maolood
Computer Science Department-University of Technology - Iraq, Baghdad, Iraq,
abeer.t.maolood@uotechnology.edu.iq

Ekhlas Khalaf Gbashi
Computer Science Department-University of Technology - Iraq, Baghdad, Iraq,
Ekhlas.K.Gbashi@uotechnology.edu.iq

Recommended Citation

Khudhair, Ala'a Talib; Maolood, Abeer Tariq; and Gbashi, Ekhlas Khalaf (2024) "A Novel Approach to Generate Dynamic S-Box for Lightweight Cryptography Based on the 3D Hindmarsh Rose Model," *Journal of Soft Computing and Computer Applications*: Vol. 1: Iss. 1, Article 1003.
DOI: https://doi.org/10.70403/3008-1084.1003

JOURNAL OF SOFT COMPUTING AND COMPUTER APPLICATIONS 2024;1:1003

ORIGINAL STUDY

A Novel Approach to Generate Dynamic S-Box for Lightweight Cryptography Based on the 3D Hindmarsh Rose Model

Ala'a Talib Khudhair *, Abeer Tariq Maolood, Ekhlas Khalaf Gbashi

Computer Science Department-University of Technology – Iraq, Baghdad, Iraq

ABSTRACT

In lightweight cryptography, the absence of an S-Box in some algorithms like speck, Tiny Encryption Algorithm, or the presence of a fixed S-Box in others like Advanced Encryption Standard can make them more vulnerable to attacks. This study introduces an innovative method for creating a dynamic 6-bit S-Box (8×8) in octal format. The generating process of S-Box passes through two phases. The first is the number initialization phase. This phase involves generating sequence numbers 1, sequence numbers 2, and sequence numbers 3 depending on X_i, Y_i, and Z_i values generated using the 3D Hindmarsh Rose model. The second is the S-Box construction phase. This phase involves building S-Box values depending on sequence numbers 1, sequence numbers 2, and sequence numbers 3, which resulted from the number initialization phase. The effectiveness of the proposed S-Box was evaluated through various criteria, including the bijective property, balanced, fixed points, opposite fixed points, completeness criteria, avalanche criteria, and strict avalanche criteria. It was observed that S-Box achieved a linear and differential branch number of 3, non-linearity of 24, differential uniformity of 4, and algebraic degree of 3. In addition, reducing the number of linear and nonlinear operations makes it suitable for lightweight algorithms. The architecture of the proposed S-Box demonstrates robustness, with a total of 1.98×10^{87} possible S-Boxes against algebraic attacks. Moreover, the construction of the S-Box and its inverse take only 14.3542 milliseconds, making it suitable for use in many lightweight block ciphers. The new S-Box is the first to exhibit these characteristics.

Keywords: Dynamic S-Box, 3D Hindmarsh Rose model, Cryptography, Lightweight S-Box

1. Introduction

The fourth industrial revolution involves a wide array of advanced technologies, with the Internet of Things (IoT) being a crucial component that connects people, objects,

Received 25 March 2024; accepted 12 May 2024.
Available online 27 June 2024

* Corresponding author.
E-mail addresses: cs.22.20@grad.uotechnology.edu.iq (A. Talib Khudhair), abeer.t.maolood@uotechnology.edu.iq (A. Tariq Maolood), Ekhlas.K.Gbashi@uotechnology.edu.iq (E. Khalaf Gbashi).

https://doi.org/10.70403/3008-1084.1003

processes, data, applications, and services. However, trustworthy systems are essential to ensuring the security and reliability of IoT-based infrastructures, with cryptography being a fundamental building block. Due to the limited resources and small size of most devices in the IoT environment, lightweight cryptography, such as the standardized block ciphers PRESENT [1] and CLEFIA [2] by the International Organization for Standardization and International Electrotechnical Commission ISO/IEC, is necessary for their security. The threat of side-channel attacks, introduced by Paul Kocher in 1996, has highlighted the importance of implementing countermeasures to protect against such attacks, as they are not preventable through traditional cryptanalysis techniques [3]. Efforts are being made to develop efficient implementations of side-channel countermeasures, particularly focusing on reducing the number of nonlinear operations in masked implementations to minimize resource overhead. Several lightweight block ciphers have been proposed to achieve a low nonlinear operation count to enhance security against side-channel attacks [4–6]. Only three studies have attempted to build a 6-bit S-Box for encryption algorithms. In 2013, B. Bilgin et al. built a 6-bit S-Box, but it required a very large number of linear and nonlinear operations [7]. In 2019, S. Sarkar et al. presented a way to build S-Boxes with differential branch number (DBN) and linear branch number (LBN) of 3. They did this by employing resilient Boolean functions. They have good cryptographic qualities, but because their search algorithms are based on algebraic techniques, they are not bitslice efficient and include a large number of linear and nonlinear operations [8]. In 2021, H. Kim et al. exploited constructions of S-Boxes from smaller S-Boxes using a Feistel structure, but it is unsuitable for lightweight cryptography because it takes a long time for generation [9].

This paper introduces a new construction method for a dynamic lightweight 6-bit S-Box based on the 3D Hindmarsh Rose model. The S-Box generation process has two phases. The first is the number initialization phase. This phase involves generating sequence numbers 1, sequence numbers 2, and sequence numbers 3 depending on X_i, Y_i, and Z_i values. The user generates X_i, takes 13 values from the result to get sequence numbers 1, generates Y_i based on the new sequence numbers 1, takes 13 values from the result to get sequence numbers 2, generates Z_i based on the new sequence numbers 1, and takes 13 values from the result to get sequence numbers 3. The user repeats this process for as many rounds as they like. The second phase is the S-Box construction phase. This phase entails generating S-Box values based on sequence numbers 1, 2, and 3, derived from the number initialization phase. The process involves several steps. First, the user randomly selects a round. Second, the user randomly chooses to depend on either sequence numbers 1, 2, or 3. Third, a position is randomly selected, ensuring it does not exceed 10. Fourth, two digits are extracted from the chosen position modulo 64. Fifth, the two extracted digits are converted to 6-bit binary and then into octal format. Finally, the result is checked against the S-Box; if absent, it is added. Otherwise, the process is reiterated until an S-Box with a size of 8×8 is generated.

First, this proposal seeks to develop a new method for constructing a dynamic, lightweight 6-bit (8×8) S-Box in octal format, based on the 3D Hindmarsh-Rose model, with a linear and differential branch number of 3. Second, this proposal seeks to enhance security by reducing the number of linear and nonlinear operations, thereby also minimizing processing time. Third, this proposal aims to conduct performance evaluations to validate the effectiveness of the proposed method, particularly its high nonlinearity, low differential uniformity, and high algebraic degree.

The paper is organized as follows: Section 2 introduces the chaotic systems, Section 3 presents the proposed method of constructing a new S-Box, Section 4 assesses the proposed S-Box and provides a comparison of our proposed S-Box and existing S-Boxes. Finally, Section 5 presents the conclusions and directions for future work.

2. Chaotic systems

Mathematical behavior that is both nonlinear and deterministic is the foundation of chaos theory [10]. It has higher sensitivity to any change in initial conditions, including the control parameters and initial values. Consequently, a slight alteration to the beginning values or control settings causes a large alteration in the chaotic outputs [11]. These characteristics are linked to those of a good cipher in cryptography, such as confusion and diffusion, enabling researchers to use chaotic systems to raise the security level of numerous cryptographic systems [12]. In this study, the 3D Hindmarsh Rose model is used.

✓ The 3D Hindmarsh Rose model was developed by John Hindmarsh and Stuart Rose. Equation (1) shows the formal definition of the 3D Hindmarsh Rose iterator [13]:

$$\begin{cases} x_{n+1} = y - a * x^3 + b * x^2 - z + I \\ y_{n+1} = c - d * x^2 - y \\ z_{n+1} = r[s(x - xr) - z] \end{cases} \quad (1)$$

Eight parameters make up the model: a, b, c, d, r, s, xr, and I. It is customary to fix some of these and leave others as control parameters. The parameter I is typically regarded as a control parameter. The control parameters a, b, c, d, or r are also frequently employed in the literature. Typically, $s = 4$ and $xr = -8/5$ are fixed parameters. The values of a, b, c, and d, when fixed, are fixed are $a = 1$, $b = 3$, $c = 1$, and $d = 5$. The parameter r ranges from 0.001 to 0.003, while I has a range of -10 to 10 [13].

3. The proposed method

This paper presents a novel approach to constructing a dynamic 6-bit S-Box (8×8) in octal format based on the 3D Hindmarsh Rose model, with a 6-bit input and a 6-bit output. The generating process of S-Box passes through two phases. First, the number initialization phase involves generating sequence numbers 1, sequence numbers 2, and sequence numbers 3 depending on X_i, Y_i, and Z_i values. The user generates X_i, takes 13 values from the result to get sequence numbers 1, generates Y_i based on the new sequence numbers 1, takes 13 values from the result to get sequence numbers 2, generates Z_i based on the new sequence numbers 1, and takes 13 values from the result to get sequence numbers 3, as shown in Algorithm 1 and Fig. 1.

Algorithm 1. Number initialization phase.

Input: Initial conditions for the 3D Hindmarsh Rose model (a, b, c, d, r, s, xr, I, X_0, Y_0 and Z_0).
Output: sequence numbers 1, sequence numbers 2, and sequence numbers 3
Begin
1. Read the initial conditions.
2. Number initialization phase:
   - 2.1: For i = 1 to n // n is a dynamic value, defined by the user
   - 2.2: Generates X_i and takes 13 values from the result to get sequence numbers 1.
   - 2.3: Generates Y_i based on the new sequence numbers 1and takes 13 values from the result to get sequence numbers 2.
   - 2.4: Generates Z_i based on the new sequence numbers 1 and takes 13 values from the result to get sequence numbers 3.
   - 2.5: Next i.
3. Remove Sign and dot (.) for the generated sequence numbers 1, sequence numbers 2, and sequence numbers 3.

End

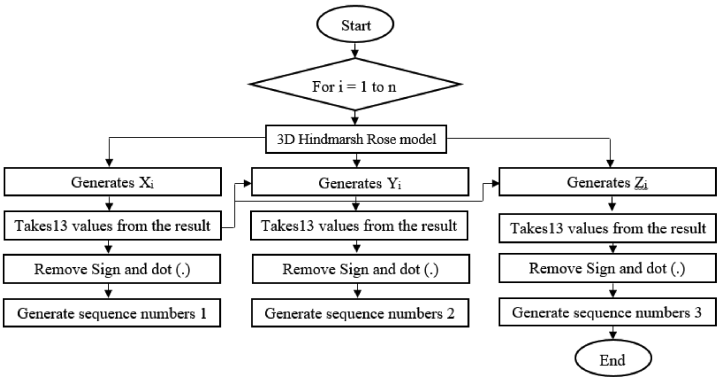

Fig. 1. Number initialization phase.

Algorithm 2. S-Box construction phase.

Input: Sequence numbers 1, sequence numbers 2, and sequence numbers 3
Output: S-Box (8×8) in octal format.
Begin
For i = 1 to n
Step 1: Select a round (i) randomly and read its value.
Step 2: Make a random selection depending on either sequence numbers 1, 2, or 3 and read its value.
Step 3: Select a position randomly. // the selected position must not exceed 10
Step 4: Cut two digits from the selected position, then takes mod 64.
Step 5: Convert the result of Step 4 into 6-bit binary format.
Step 6: Convert the 6-bit binary format into octal format.
Step 7: Add the result of Step 6 to the S-Box if the result does not exist inside it.
Next i.
End

Second, the S-Box construction phase entails generating S-Box values based on sequence numbers 1, 2, and 3 derived from the number initialization phase. The process involves several steps. First, the user randomly selects a round. Second, the user randomly chooses to depend on either sequence numbers 1, 2, or 3. Third, a position is randomly selected, ensuring it does not exceed 10. Fourth, two digits are extracted from the chosen position modulo 64. Fifth, the two extracted digits are converted to 6-bit binary and then into octal format. Finally, the result is checked against the S-Box; if absent, it is added. Otherwise, the process is reiterated until an S-Box with a size of 8×8 is generated, as illustrated in Algorithm 2 and Fig. 2.

3.1. Inverse S-Box

To generate each value in the inverse S-Box, the following process is followed. The user takes the first value of the S-Box, such as (43), and divides it into two digits, (4) and (3), to indicate the address of the value in the inverse S-Box. The extracted address of (43) is row 0 and column 0, which are then combined to determine the value that will be stored in the inverse S-Box. Hence, the value found in row (4) and column (3) of the inverse S-Box is (00), as shown in Algorithm 3.

A complete example of the generation of a dynamic S-Box employing the 3D Hindmarsh Rose model is shown below:

A. Number initialization phase

Implement the 3D Hindmarsh Rose model using $a = 1$, $b = 3$, $c = 1$, $d = 5$, $s = 4$, $I = 5$, $xr = -1.6$, $r = 0.001$, $X_0 = 0.1$, $Y_0 = 0.1$ and $Z_0 = 3$.

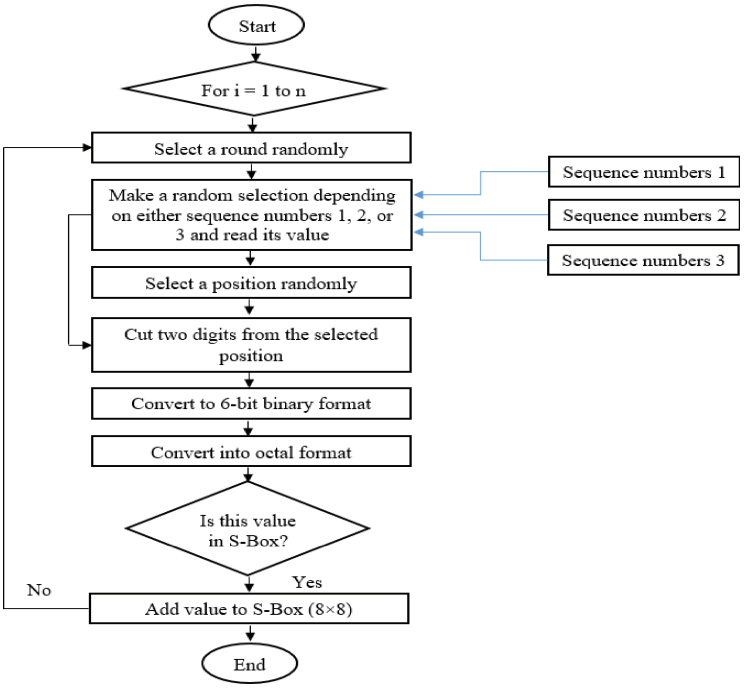

Fig. 2. S-Box construction phase.

Algorithm 3. Create the S-Box's inverse.

Input: S-Box (8 × 8).
Output: S-Box inverse (8 × 8).
Begin
Step 1: Loop with each of the S-Box (8 × 8) numbers.
Step 2: Split each number into two digits to determine the address of its corresponding value in the S-Box inverse.
Step 3: Combine the addresses of the two digits to determine the new value to be stored in the S-Box inverse.
Step 4: Repeat Steps 2–3 until the S-Box inverse is fully constructed.
End

When i = 1

$X_1 = 0.1 + (-1 \times (0.1)\ \hat{}3) + (3 \times (0.1)\ \hat{}2) - 3 + 5 = 2.129$, takes 13 digits; the result is 2.129

$Y_1 = 1 - (5 \times (2.129)\ \hat{}2) - 0.1 = -21.763205$, takes 13 digits; the result is −21.763205

$Z_1 = 0.001\ [4 \times (2.129 - (-1.6) - 3)] = 0.011916$, takes 13 digits; the result is 0.011916

When i = 2

$X_2 = -21.763205 + (-1 \times (2.129)\ \hat{}3) + (3 \times (2.129)\ \hat{}2) - 0.011916 + 5 = -12.827190689$, takes 13 values; the result is −12.827190689

$Y_2 = 1 - (5 \times (-12.\ 827190689)\ \hat{}2) - (-21.763205) = -799.920899859841473605$, takes 13 values; the result is −799.92089985

$Z_2 = 0.001\ [(4 \times (-12.827190689 - (-1.6))) - 0.011916] = -0.044920678756$, takes 13 values; the result is −0.0449206787

The rest of the results are obtained in the same way, as shown in Table 1.

The next step is to remove the sign and dot (.) to generate sequence numbers 1, 2, and 3, as shown in Table 2.

Table 1. Generate X_i, Y_i, and Z_i using the 3D Hindmarsh Rose model.

Number of i	X_i	Y_i	Z_i
1	2.129	−21.763205	0.011916
2	−12.827190689	−799.92089985	−0.0449206787
3	1809.27966171	−16366663.55	7.24356356751
4	−5929210303.4	−1.757776741	−23716841.214
5	2.08444559393	−18.966790429	23716.8559517
6	−23726.844737	−2814815785.9	−118.61783489
...	...	...	...

Table 2. Sequence numbers 1, 2, and 3 after removing sign and dot (.).

Number of i	Sequence numbers 1	Sequence numbers 2	Sequence numbers 3
1	2129	21763205	0011916
2	12827190689	79992089985	00449206787
3	180927966171	1636666355	724356356751
4	59292103034	1757776741	23716841214
5	208444559393	18966790429	237168559517
6	23726844737	28148157859	11861783489
...	...	...	...

Table 3. S-Box construction phase.

Select round	Select depending on either sequence numbers 1, 2, or 3	Value of round	Select position	Cut 2 digits from the chosen position mod 64	Convert 2 digits to 6-bit binary	Convert to octal
1	Sequence numbers 3	0011916	1	00 mod 64 = 0	000000	00
5	Sequence numbers 1	208444559393	6	45 mod 64 = 45	101101	55
6	Sequence numbers 2	28148157859	4	48 mod 64 = 48	110000	60
10	Sequence numbers 1	15248740239	1	15 mod 64 = 15	001111	17
18	Sequence numbers 1	27693244587	9	58 mod 64 = 58	111010	72
...	...	...	...	...	...	...

Table 4. S-Box.

00	55	60	17	72	40	16	61
15	07	51	14	03	66	67	32
52	31	26	42	74	46	65	37
25	63	04	30	33	34	53	41
47	23	36	77	20	01	73	10
71	76	35	62	06	54	44	21
27	12	70	45	11	57	05	24
50	64	43	02	22	75	56	13

B. S-Box construction phase

This phase entails generating S-Box values based on sequence numbers 1, 2, and 3, derived from the number initialization phase, as shown in Table 3.

Similarly, the generation process continues until the S-Box (8×8) is obtained, as shown in Table 4. Table 5 shows the S-Boxes in reverse.

4. Performance evaluation

This section demonstrates the security strength of the proposed 6-bit S-Box, measured over balanced, bijective property, fixed points (FP), opposite fixed points (OFP), completeness criteria (CC), avalanche criteria (AC), strict avalanche criteria (SAC), linear branch number (LBN), differential branch number (DBN), differential uniformity (DU), algebraic

Table 5. S-Box inverse.

00	45	73	14	32	66	54	11
47	64	61	77	13	10	06	03
44	57	74	41	67	30	22	60
33	21	17	34	35	52	42	27
05	37	23	72	56	63	25	40
70	12	20	36	55	01	76	65
02	07	53	31	71	26	15	16
62	50	04	46	24	75	51	43

degree, linear and nonlinear operations, algebraic attacks, and time. It also compares the cryptanalysis of the proposed 6-bit S-Box with other 6-bit S-Boxes from B. Bilgin et al. [7], Sakar [8], and H. Kim et al. [9].

4.1. Balanced

The obtained S-Box is balanced since it contains an equal number of 1s and 0s in the corresponding truth table of the Galois Field GF (2^6). The eighth and 16th columns have an equal number of 0s and 1s, representing the "XOR" function, as shown in Table 6.

Table 6. The balance of the S-Box.

S-Box value (octal)	Convert to 6-bit (octal)						XOR (x1, x2, x3, x4, x5, x6)	S-Box value (octal)	Convert to 6-bit (octal)						XOR (x1, x2, x3, x4, x5, x6)
	x1	x2	x3	x4	x5	x6			x1	x2	x3	x4	x5	x6	
00	0	0	0	0	0	0	0	47	1	0	0	1	1	1	0
55	1	0	1	1	0	1	0	23	0	1	0	0	1	1	1
60	1	1	0	0	0	0	0	36	0	1	1	1	1	0	0
17	0	0	1	1	1	1	0	77	1	1	1	1	1	1	0
72	1	1	1	0	1	0	0	20	0	1	0	0	0	0	1
40	1	0	0	0	0	0	1	01	0	0	0	0	0	1	1
16	0	0	1	1	1	0	1	73	1	1	1	0	1	1	1
61	1	1	0	0	0	1	1	10	0	0	1	0	0	0	1
15	0	0	1	1	0	1	1	71	1	1	1	0	0	1	0
07	0	0	0	1	1	1	1	76	1	1	1	1	1	0	1
51	1	0	1	0	0	1	1	35	0	1	1	1	0	1	0
14	0	0	1	1	0	0	0	62	1	1	0	0	1	0	1
03	0	0	0	0	1	1	0	06	0	0	0	1	1	0	0
66	1	1	0	1	1	0	0	54	1	0	1	1	0	0	1
67	1	1	0	1	1	1	1	44	1	0	0	1	0	0	0
32	0	1	1	0	1	0	1	21	0	1	0	0	0	1	0
52	1	0	1	0	1	0	1	27	0	1	0	1	1	1	0
31	0	1	1	0	0	1	1	12	0	0	1	0	1	0	0
26	0	1	0	1	1	0	1	70	1	1	1	0	0	0	1
42	1	0	0	0	1	0	0	45	1	0	0	1	0	1	1
74	1	1	1	1	0	0	0	11	0	0	1	0	0	1	0
46	1	0	0	1	1	0	1	57	1	0	1	1	1	1	1
65	1	1	0	1	0	1	0	05	0	0	0	1	0	1	0
37	0	1	1	1	1	1	1	24	0	1	0	1	0	0	0
25	0	1	0	1	0	1	1	50	1	0	1	0	0	0	0
63	1	1	0	0	1	1	0	64	1	1	0	1	0	0	1
04	0	0	0	1	0	0	1	43	1	0	0	0	1	1	1
30	0	1	1	0	0	0	0	02	0	0	0	0	1	0	1
33	0	1	1	0	1	1	0	22	0	1	0	0	1	0	0
34	0	1	1	1	0	0	1	75	1	1	1	1	0	1	1
53	1	0	1	0	1	1	0	56	1	0	1	1	1	0	0
41	1	0	0	0	0	1	0	13	0	0	1	0	1	1	1

4.2. Bijective property

The obtained S-Box is bijective by balancing 0s and 1s. The Hamming weight of all Boolean functions is [32 32 32 32 32 32 32 32], as shown in Table 7.

Table 7. The Hamming weight of the S-Box.

S-Box value (octal)	Convert to 6-bit (octal)						S-Box value (octal)	Convert to 6-bit (octal)					
	F1	F2	F3	F4	F5	F6		F1	F2	F3	F4	F5	F6
00	0	0	0	0	0	0	47	1	0	0	1	1	1
55	1	0	1	1	0	1	23	0	1	0	0	1	1
60	1	1	0	0	0	0	36	0	1	1	1	1	0
17	0	0	1	1	1	1	77	1	1	1	1	1	1
72	1	1	1	0	1	0	20	0	1	0	0	0	0
40	1	0	0	0	0	0	01	0	0	0	0	0	1
16	0	0	1	1	1	0	73	1	1	1	0	1	1
61	1	1	0	0	0	1	10	0	0	1	0	0	0
15	0	0	1	1	0	1	71	1	1	1	0	0	1
07	0	0	0	1	1	1	76	1	1	1	1	1	0
51	1	0	1	0	0	1	35	0	1	1	1	0	1
14	0	0	1	1	0	0	62	1	1	0	0	1	0
03	0	0	0	0	1	1	06	0	0	0	1	1	0
66	1	1	0	1	1	0	54	1	0	1	1	0	0
67	1	1	0	1	1	1	44	1	0	0	1	0	0
32	0	1	1	0	1	0	21	0	1	0	0	0	1
52	1	0	1	0	1	0	27	0	1	0	1	1	1
31	0	1	1	0	0	1	12	0	0	1	0	1	0
26	0	1	0	1	1	0	70	1	1	1	0	0	0
42	1	0	0	0	1	0	45	1	0	0	1	0	1
74	1	1	1	1	0	0	11	0	0	1	0	0	1
46	1	0	0	1	1	0	57	1	0	1	1	1	1
65	1	1	0	1	0	1	05	0	0	0	1	0	1
37	0	1	1	1	1	1	24	0	1	0	1	0	0
25	0	1	0	1	0	1	50	1	0	1	0	0	0
63	1	1	0	0	1	1	64	1	1	0	1	0	0
04	0	0	0	1	0	0	43	1	0	0	0	1	1
30	0	1	1	0	0	0	02	0	0	0	0	1	0
33	0	1	1	0	1	1	22	0	1	0	0	1	0
34	0	1	1	1	0	0	75	1	1	1	1	0	1
53	1	0	1	0	1	1	56	1	0	1	1	1	0
41	1	0	0	0	0	1	13	0	0	1	0	1	1

Therefore,

	F1	F2	F3	F4	F5	F6
Hamming weight	32	32	32	32	32	32

4.3. Fixed points (FP)

If an S-Box returns the same value as its input (S-Box (m) = m), it is said to contain an FP [14]. An S-Box with fewer FP is more resistant to attacks. The proposed S-Box possesses two FPs, as shown in Table 8. Table 12 compares the FP of the proposed S-Box with other S-Boxes in the literature.

Table 8. FP of the S-Box.

Input (octal)	Input (binary)	Output (octal)	Output (binary)	Result	Input (octal)	Input (binary)	Output (octal)	Output (binary)	Result
00	000000	00	000000	Fixed point	40	100000	47	111001	No fixed point
01	000001	55	101101	No fixed point	41	100001	23	010011	No fixed point
02	000010	60	110000	No fixed point	42	100010	36	011110	No fixed point
03	000011	17	001111	No fixed point	43	100011	77	111111	No fixed point
04	000100	72	111010	No fixed point	44	100100	20	100111	No fixed point
05	000101	40	100000	No fixed point	45	100101	01	010000	No fixed point
06	000110	16	001110	No fixed point	46	100110	73	000001	No fixed point
07	000111	61	110001	No fixed point	47	100111	10	111011	No fixed point
10	001000	15	001101	No fixed point	50	101000	71	001000	No fixed point
11	001001	07	000111	No fixed point	51	101001	76	111110	No fixed point
12	001010	51	101001	No fixed point	52	101010	35	011101	No fixed point
13	001011	14	001100	No fixed point	53	101011	62	110010	No fixed point
14	001100	03	000011	No fixed point	54	101100	06	000110	No fixed point
15	001101	66	110110	No fixed point	55	101101	54	101100	No fixed point
16	001110	67	110111	No fixed point	56	101110	44	100100	No fixed point
17	001111	32	011010	No fixed point	57	101111	21	010001	No fixed point
20	010000	52	101010	No fixed point	60	110000	27	010111	No fixed point
21	010001	31	011001	No fixed point	61	110001	12	001010	No fixed point
22	010010	26	010110	No fixed point	62	110010	70	111000	No fixed point
23	010011	42	100010	No fixed point	63	110011	45	100101	No fixed point
24	010100	74	111100	No fixed point	64	110100	11	001001	No fixed point
25	010101	46	100110	No fixed point	65	110101	57	101111	No fixed point
26	010110	65	110101	No fixed point	66	110110	05	000101	No fixed point
27	010111	37	011111	No fixed point	67	110111	24	010100	No fixed point
30	011000	25	010101	No fixed point	70	111000	50	101000	No fixed point
31	011001	63	110011	No fixed point	71	111001	64	110100	No fixed point
32	011010	04	000100	No fixed point	72	111010	43	100011	No fixed point
33	011011	30	011000	No fixed point	73	111011	02	000010	No fixed point
34	011100	33	011011	No fixed point	74	111100	22	010010	No fixed point
35	011101	34	011100	No fixed point	75	111101	75	111101	Fixed point
36	011110	53	101011	No fixed point	76	111110	56	101110	No fixed point
37	011111	41	100001	No fixed point	77	111111	13	001011	No fixed point

4.4. Opposite fixed points (OFP)

A point is called an OFP of S-Box if S (a) = à, where à is the complement of a [15]. There is no OFP in the new S-Box, as shown in Table 8. Any S-Box without an OFP performs better against differential cryptanalysis attacks than those with an OFP.

4.5. Completeness criteria (CC)

The newly generated S-Box meets the CC, as it depends on the input. If there is at least one bit of an input, such as X and X_i that only differs in one bit, then the output of f (X) and f (X_i) will differ at least in J bits. The output of the S-Box depends on the initial conditions of the 3D Hindmarsh Rose model (a, b, c, d, r, s, xr, I, X_0, Y_0, and Z_0). This means that even a slight alteration in these values can result in a significant change in the outputs of the S-Boxes. Tables 4 and 5 were created with the following initial conditions: $a = 1$, $b = 3$, $c = 1$, $d = 5$, $s = 4$, $I = 5$, $xr = -1.6$, $r = 0.001$, $X_0 = 0.1$, $Y_0 = 0.1$, and $Z_0 = 3$. Changing X_0 to be 0.11, compare with Tables 9 and 10, demonstrates how the values of the S-Box are completely altered by a slight modification in any of the parameters used to generate the S-Box.

Table 9. S-Box

00	45	01	44	76	02	35	56
26	63	62	43	10	41	03	70
32	72	52	20	17	27	14	30
16	47	46	13	61	25	06	24
60	22	21	36	42	55	05	50
37	04	34	40	54	33	07	75
15	23	74	73	31	11	66	65
77	53	64	71	57	67	12	51

Table 10. S-Box inverse

00	02	05	16	51	46	36	56
14	65	76	33	26	60	30	24
23	42	41	61	37	35	10	25
27	64	20	55	52	06	43	50
53	15	44	13	03	01	32	31
47	77	22	71	54	45	07	74
40	34	12	11	72	67	66	75
17	73	21	63	62	57	04	70

Table 11. AC test.

Method	Binary (input)	Replace in S-Box	Binary (output)	AC
Proposed S-Box	011011	30	011000	$2/6 = 0.333$
Change 1 bit	011111	41	100001	$5/6 = 0.833$

4.6. Avalanche criteria (AC)

A slight change in the input bits will result in a significant change in the output sequence, for example, flipping a single bit from 0 to 1 or vice versa. This criterion is highly desirable in block cipher methods due to its impact on diffusion computation. Table 11 illustrates the AC test.

4.7. The strict avalanche criterion (SAC)

This criterion is satisfied by the S-Box when modifying one bit in the input results in changing half of the output bits [16]. In other words, if both the CC and AC are met, the SAC is achieved [17]. The proposed technique fulfills these criteria, and, thus meets the SAC.

4.8. Linear branch number

The linear branch number (LBN) of an S-Box [18] is defined as

$$\min_{a,b,\Phi(a,b)\neq 0} \left(\mathrm{wt}\,(a) + \mathrm{wt}\,(b)\right). \tag{2}$$

The upper bound of LBN $\leq n - 1$, where n is input bits. A higher LBN enhances the system's resistance against linear attacks [19]. The proposed S-Box has a LBN of 3, as shown in Table 12.

Table 12. Comparison of the cryptographic characteristics and operation counts of 6-bit S-Boxes.

DBN	2	3	3	3
LBN	2	3	3	3
Differential uniformity	2	4	4	4
Non-linearity	24	24	24	24
Algebraic degree	4	2	4	3
Fixed points	0	4	2	2
nonlinear operations	35	36	9	0
linear operations	93	54	12	0
Construction method	Rijndael	Toeplitz matrix	Feistel	3D Hindmarsh Rose model
Reference	[7]	[8]	[9]	This paper

*The results were obtained using SageMath, a free, open-source mathematics tool.

4.9. Differential branch number

The differential branch number (DBN) of an S-Box [20] is defined as

$$\min_{a,b\neq a}\left(\mathrm{wt}\left(a\oplus b\right)+\mathrm{wt}\left(S\left(a\right)\oplus S\left(b\right)\right)\right). \tag{3}$$

The user calculates all possible XORs between the Galois Field GF (2^n) entries and then all possible XORs between the S-Box entries. Finally, the user calculates their Hamming weights and computes the minimum value. The upper bound of DBN is $[\frac{2n}{3}]$, where n is input bits. A higher DBN enhances the resistance of the system against differential attacks [21]. The proposed S-Box has a DBN of 3, as shown in Table 12.

4.10. Differential uniformity

The differential uniformity (DU) of an S-Box [22] is defined as

$$\max_{\Delta\alpha\neq 0,\Delta\beta}\#\left\{x\in F_n^2 S\left(x\right)\oplus S\left(x\oplus\Delta\alpha\right)=\Delta\beta\right\}. \tag{4}$$

The lower the DU, the better the resistance to differential attacks [23]. The lowest possible DU is 2 [24]. The proposed S-Box achieved a maximum DU score of 4. This low score affirms the potential of the proposed S-Box to resist differential cryptanalysis effectively. Furthermore, Table 12 compares the DU scores of the proposed S-Boxes with those found in the existing literature. This comparison underscores the efficacy of the S-Box in thwarting attempts at differential cryptanalysis.

4.11. Non-Linearity

The non-linearity (NL) of an S-Box [25] is defined as

$$2^{n-1}-2^{-1}\times\max_{\lambda\alpha,\,\lambda\beta\neq 0}\left|\Phi\left(\lambda\alpha,\,\lambda\beta\right)\right|,\ \text{where}\ \Phi\left(\lambda\alpha,\,\lambda\beta\right)=\sum_{x\,\in\,F_n^2}(-1)^{\lambda\beta\cdot S(x)\oplus\lambda\alpha\cdot x}. \tag{5}$$

where "·" is the inner product in the respective Galois field. A higher NL implies greater complexity and unpredictability in the relationship between inputs and outputs, which enhances the resistance of the cryptographic algorithm against various attacks, including differential cryptanalysis and linear cryptanalysis [25]. The maximum NL in GF (2^n) is (a)

for n = even is $2^{n-1} - 2^{\frac{n}{2}-1}$, (b) for n = odd is $2^{n-1} - 2^{\frac{n-1}{2}}$. The proposed S-Box attained an NL of 24, as demonstrated in Table 12.

4.12. Linear and nonlinear operations

The fewer linear (XOR, NOT) and nonlinear operations (AND, OR), the better. The proposed method for generating an S-Box does not utilize linear or nonlinear operations, resulting in a count of 0 for both types of operations, as indicated in Table 12.

4.13. Algebraic degree

Algebraic degree is the number of variables in the highest order term with non-zero coefficients. The maximum algebraic degree can be denoted as deg (f) = n − 1. A higher algebraic degree is preferable [26]. The proposed S-Box achieved a favorable algebraic degree of 3, as evidences in Table 12.

4.14. Algebraic attacks

The structure of the proposed S-Box is simple but robust. The total number of possible S-Boxes is 1.98×10^{87} which is massive and noticeably greater than the 5-bit S-Box ($31! \approx 8.22 \times 10^{33}$) and 4-bit S-Box ($15! \approx 1.3 \times 10^{12}$). Furthermore, the proposed 6-bit S-Box uses a dynamic, chaotic system to introduce randomness into the S-Box's elements, making it difficult to break through.

4.15. Time

It takes only 14.3542 milliseconds to generate the S-Box and its inverse, making it ideal for incorporation into numerous lightweight encryption algorithms.

5. Conclusions

This paper presents a novel method for building a dynamic lightweight S-Box based on the 3D Hindmarsh Rose model with a linear and differential branch number of 3, which reflects its strength against linear and differential attacks. Small changes in the initial conditions of the 3D Hindmarsh Rose model equations result in a new S-Box by exploiting the strong relationship between chaotic systems and cryptography. The S-Box's increased nonlinearity enhances the input-output relationship's complexity and unpredictability, making the cryptographic algorithm more resistant to various attacks, such as differential and linear cryptanalysis. Furthermore, reducing the number of linear and nonlinear operations enhances the security level. The evaluation of the new S-Box confirms its suitability for secure communication in lightweight cryptography algorithms. Future research could explore integrating other chaotic systems in the S-Box construction process to further bolster security.

References

1. B. Aboushosha, R. A. Ramadan, A. D. Dwivedi, A. El-Sayed, and M. M. Dessouky, "SLIM: a lightweight block cipher for Internet of health things," *IEEE Access*, vol. 8, pp. 203747–203757, 2020, doi: 10.1109/ACCESS.2020.3036589.

2. H. A. M. A. Basha, A. S. S. Mohra, T. O. M. Diab, and W. I. E. Sobky, "Efficient image encryption based on new substitution box using DNA coding and bent function," *IEEE Access*, vol. 10, pp. 66409–66429, 2022, doi: 10.1109/ACCESS.2022.3183990.
3. E. W. Affy, W. I. E. Sobky, A. Twakol, and R. A. Alez, "Algebraic construction of powerful substitution box," *International Journal of Recent Technology and Engineering*, vol. 8, no. 6, pp. 405–409, 2020.
4. M. M. Abd Zaid and S. Hassan, "Proposal framework to light weight cryptography primitives," *Engineering and Technology Journal*, vol. 40, no. 4, pp. 516–526, 2022.
5. X. Qian, Q. Yang, Q. Li, Q. Liu, Y. Wu, and W. Wang, "A novel color image encryption algorithm based on three dimensional chaotic maps and reconstruction techniques," *IEEE Access*, vol. 9, pp. 61334–61345, 2021, doi: 10.1109/ACCESS.2021.3073514.
6. V. Grosso, G. Leurent, F. Standaert, and K. Varici, "LS-designs: bitslice encryption for efficient masked software implementations," Presented at the 21st Int. Workshop on Fast Software Encryption (FSE), London, UK, March 3–5, 2014.
7. B. Bilgin, A. Bogdanov, M. Knezevic, F. Mendel, and O. Wang, "Fides: lightweight authenticated cipher with side-channel resistance for constrained hardware," Presented at the 15th Int. Workshop on Cryptographic Hardware and Embedded Systems (CHES), Aug. 20–23, 2013.
8. S. Sarkar, K. Mandal, and D. Saha, "On the relationship between resilient boolean functions and linear branch number of S-boxes," in *Proc. 20th Int. Conf. on Cryptology in India*, India, pp. 361–374, 2019, doi: 10.1007/978-3-030-35423-7_18.
9. H. Kim *et al.*, "A new method for designing lightweight S-boxes with high differential and linear branch numbers, and its application," *IEEE Access*, vol. 9, pp. 150592–150607, 2021, doi: 10.1109/ACCESS.2021.3126008.
10. F. Ishfaq, "A MATLAB tool for the analysis of cryptographic properties of S-boxes." Capital University, Master thesis, 2018.
11. S. Sarkar and H. Syed, "Bounds on differential and linear branch number of permutations," In *Proc. 23rd Australasian Conf. on Information Security and Privacy* (ACISP), Wollongong, Australia, pp. 207–224, 2018.
12. A. T. Maolood *et al.*, "Fast novel efficient S-boxes with expanded DNA codes," *Security and Communication Networks*, vol. 2023, Art. no. 5767102, 2023, doi: 10.1155/2023/5767102.
13. O. Camps *et al.*, "Implementation of the Hindmarsh–Rose model using stochastic computing", *Mathematics*, vol. 10, no. 23, Art. no. 4628, 2022, doi: 10.3390/math10234628.
14. S. A. Jassim and A. K. Farhan, "Designing a novel efficient substitution-box by using a flower pollination algorithm and chaos system," *International Journal of Intelligent Engineering and Systems*, vol. 15, no. 1, pp. 176–187, 2022, doi: 10.22266/ijies2022.0228.17.
15. A. K. Farhan, R. S. Ali, H. R. Yassein, N. M. G. Al-Saidi, and G. H. Abdul-Majeed, "A new approach to generate multi S-boxes based on RNA computing," *International Journal of Innovative Computing, Information and Control (IJICIC)*, vol. 16, no. 1, pp. 331–348, 2020, 10.24507/ijicic.16.01.331.
16. A. H. Zahid *et al.*, "Efficient dynamic S-box generation using linear trigonometric transformation for security applications," *IEEE Access*, vol. 9, pp. 98460–98475, 2021.
17. A. Razaq *et al.*, "A novel method for generation of strong substitution-boxes based on coset graphs and symmetric groups," *IEEE Access*, vol. 8, pp. 75473–75490, 2020, doi: 10.1109/ACCESS.2021.3095618.
18. A. Alhudhaif, M. Ahmad, A. Alkhayyat, N. Tsafack, A. K. Farhan, and R. Ahmed, "Block cipher nonlinear confusion components based on new 5-D hyperchaotic system," *IEEE Access*, vol. 9, pp. 87686–87696, 2021, doi: 10.1109/ACCESS.2021.
19. H. A. Abdulwahab, A. Noraziah, A. A. Alsewari, and S. Q. Salih, "An enhanced version of black hole algorithm via levy flight for optimization and data clustering problems," *IEEE Access*, vol. 7, pp. 142085–142096, 2019, doi: 10.1109/ACCESS.2019.2937021.
20. A. T. Maolood, E. K. Gbashi, and E. S. Mahmood, "Novel lightweight video encryption method based on ChaCha20 stream cipher and hybrid chaotic map," *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 12, no. 5, pp. 4988–5000, 2022.
21. D. Lambić, "A new discrete-space chaotic map based on the multiplication of integer numbers and its application in S-box design," *Nonlinear Dyn.*, vol. 100, no. 1, pp. 699–711, 2020.
22. F. Özkaynak, "On the effect of chaotic system in performance characteristics of chaos based S-box designs," *Phys. A Stat. Mech. its Appl.*, vol. 550, Art. no. 124072, 2020.
23. A. R. Alawi and N. F. Hassan, "A proposal video encryption using light stream algorithm," *Engineering and Technology Journal*, vol. 39, Part B, no. 1, pp. 184–196, 2021.
24. A. A. Abdallah and Alaa K. Farhan, "New S-box design for image encryption based on multi-chaotic system," *Engineering and Technology Journal*, vol. 41, no. 10, pp. 1211–1219, 2023.

25. C. A. Yogaraja and K. S. Rani, “Key-based dynamic S-box approach for PRESENT lightweight block cipher,” *KSII Transactions on Internet and Information Systems*, vol. 17, no. 12, pp. 3398–3415, 2023, doi: 10.3837/tiis.2023.12.010.
26. E. K. Gbashi, A. T. Maolood, and Y. N. Jurn, “Privacy security system for video data transmission in Edge-Fog-cloud environment,” *International Journal of Intelligent Engineering and Systems*, vol. 16, no. 2, pp. 307–318, 2023.