Journal of Information Security and Applications 64 (2022) 103065

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

A novel method to generate key-dependent s-boxes with identical algebraic properties

Ahmad Y. Al-Dweik [a], Iqtadar Hussain [a], Moutaz Saleh [b,*], M.T. Mustafa [a]

[a] Department of Mathematics, Statistics and Physics, College of Arts and Science, Qatar University, Doha, 2713, State of Qatar
[b] Department of Computer Science and Engineering, College of Engineering, Qatar University, Doha, 2713, State of Qatar

ARTICLE INFO

Keywords:
Cryptography
Key-dependent s-boxes
Permutation
Bijection
Nonlinearity
Strict avalanche criterion
Bits independence criterion
Invariant

ABSTRACT

The s-box plays the vital role of creating confusion between the ciphertext and secret key in any cryptosystem, and is the only nonlinear component in many block ciphers. Dynamic s-boxes, as compared to static, improve entropy of the system, hence leading to better resistance against linear and differential attacks. It was shown in Easttom (2018) that while incorporating dynamic s-boxes in cryptosystems is sufficiently secure, they do not keep non-linearity invariant. This work provides an algorithmic scheme to generate key-dependent dynamic $n \times n$ clone s-boxes having the same algebraic properties namely bijection, nonlinearity, the strict avalanche criterion (SAC), the output bits independence criterion (BIC) as of the initial seed s-box. The method is based on group action of symmetric group S_n and a subgroup S_{2^n} respectively on columns and rows of Boolean functions ($GF(2^n) \rightarrow GF(2)$) of s-box. Invariance of the bijection, nonlinearity, SAC, and BIC for the generated clone copies is proved. As illustration, examples are provided for $n = 8$ and $n = 4$ along with comparison of the algebraic properties of the clone and initial seed s-box. The proposed method is an extension of Hussain et al. (2012); Hussain et al. (2012); Hussain et al. (2018); Anees and Chen (2020) which involved group action of S_8 only on columns of Boolean functions ($GF(2^8) \rightarrow GF(2)$) of s-box. For $n = 4$, we have used an initial 4×4 s-box constructed by Carlisle Adams and Stafford Tavares (Adams and Tavares, 1990) to generated $(4!)^2$ clone copies. For $n = 8$, it can be seen (Hussain et al. (2012); Hussain et al. (2012); Hussain et al. (2018); Anees and Chen (2020)) that the number of clone copies that can be constructed by permuting the columns is $8!$. For each column permutation, the proposed method enables to generate $8!$ clone copies by permuting the rows.

1. Introduction

Cryptography has emerged as a key solution for protecting information and securing data transmission against passive and active attacks. Substitution box or s-box is a vital component of symmetric block encryption schemes such as Data Encryption Standard (DES), Advanced Encryption Standard (AES) and International Data Encryption Algorithm (IDEA). The cryptographic strength of these encryption systems mainly depends upon the efficiency of their substitution boxes being the only components capable of inducing the nonlinearity in the cryptosystem [1]. This attracted the attentions of many researches to design cryptographically potent s-boxes for the sake of developing robust encryption schemes. Theoretically, there are several properties that can evaluate the performance of a proposed s-box [2]. The most commonly applied properties are the bijective property, nonlinearity, strict avalanche criteria (SAC) and bits independence criterion (BIC).

Depending on the design nature of s-box, it can be classified into either static or dynamic. The static s-box is one whose values are key-independent and once defined by the designer it is maintained during the whole encryption process. This means that the same s-box will be used in every round, and so it might be vulnerable to cryptanalysis. On the other hand, dynamic s-boxes do not suffer from fixed structure block ciphers since the s-boxes itself are changed in every encryption round and it is considered key-dependent. Hence, the adoption of dynamic s-boxes improves the security of the system and better resists against various differential and cryptanalysis attacks [3].

Many researchers have explored several ideas for s-box design such as randomness, dynamicity, and key-dependency. For instance, Krishnamurthy and Ramaswamy employed s-box rotation and used it as an additional component in the traditional AES algorithm to design a dynamic s-box [4]. The process consists of three steps in which the s-boxes are rotated based on fixed, partial and whole key values to increase

* Corresponding author.
E-mail addresses: aydweik@qu.edu.qa (A.Y. Al-Dweik), iqtadarqau@qu.edu.qa (I. Hussain), moutaz.saleh@qu.edu.qa (M. Saleh), tahir.mustafa@qu.edu.qa (M.T. Mustafa).

https://doi.org/10.1016/j.jisa.2021.103065

Available online 1 December 2021

their security. In [5], Piotr Mroczkowski proposed an algorithm to replace the available s-boxes through using pseudo-randomly generators to design similar s-boxes in both encryption and decryption processes. The work claims that changing of s-boxes could prevent intruders from receiving enough information to execute effective cryptanalysis attack. Stoianov [6] proposed a novel approach for changing the s-boxes used in the AES algorithm through introducing two new s-boxes known as SBOXLeft and S-BOXRight that employs the left and right diagonals as the axis of symmetry.

The practice of using key-dependent generated s-boxes in cryptography has been also extensively studied in the literature. For instance, the work in [7] used RC4 algorithm to generate key dependent s-boxes based on the input key. The authors showed that their generated dynamic s-boxes have increased the AES complexity and also make the differential and linear cryptanalysis more difficult. In [8], Kazlauskas et al. proposed an approach to randomly generate key dependent s-box that rely on changing only one bit of the secret key. Their approach is claimed to solve the problem of the fixed structure s-boxes, and increase the security level of the AES block cipher system due to its resistance of linear and differential cryptanalysis attacks. Ghada Zaibi et al. presented dynamic s-boxes based on one dimensional chaotic maps and evaluated its efficiency compared to the static s-box [9]. Their findings showed that AES using dynamic chaotic s-box is more secure and efficient than AES with static s-box. In their work [10], Jie Cui et al. proposed to increase the complexity and security of AES s-box by modifying the affine transformation cycle. The evaluation results suggested that the improved AES s-box has better performance and can readily be applied to AES. Likewise, Anna Grocholewska-Czurylo [11] described an AES-like dynamic s-boxes generated using finite field inversion. The significant remark for this work indicates that removing the affine equivalence cycles from s-boxes does not influence on their cryptographic properties. Julia Juremi et al. [12] proposed a key-dependent s-box to enhance the security of AES algorithm through employing a key expansion algorithm together with s-box rotation. The obtained results showed that the enhancement on the original AES does not violate the security of the cipher. Similar to the work in [8], Razi Hosseinkhani et al. [13] introduced an algorithm to generate dynamic s-box from cipher key. The quality of this algorithm was tested by changing only two bits of cipher key to generate new s-boxes. The authors claim that the key advantage of this algorithm is that various s-boxes can be generated by changing cipher key. Iqtadar Hussain et al. [14] presented a method for constructing 8×8 s-boxes using the Liu J substitution box as a seed during the creation process. The proposed design relies on the symmetric group permutation operation which is embedded in the algebraic structure of the new s-box. An extension of the above work was conducted by the same authors in [15]. They proposed a novel method that uses the symmetric group permutation based on the characteristics of affine–power–affine structure to generate nonlinear s-box component with the possibility to incorporate 40320 unique instances. The work presented a deep analysis to evaluate the properties of these new s-boxes and determine its suitability to various encryption applications.

In the middle of the last decade, several attempts were made to design robust dynamic s-boxes for symmetric cryptography systems. For instance, Oleksandr Kazymyrov et al. [16] described an improved method based on the analysis of vectorial Boolean functions properties for selection of s-boxes with optimal cryptographic properties that would lead to provide high level of robustness against various types of attacks. Mona Dara et al. [17] used chaotic logistic maps with cipher key to construct key dependent s-boxes for AES algorithm. The proposed s-box was tested against equiprobable input/output XOR distribution, key sensitivity, nonlinearity, SAC and BIC properties. In [18], Eman Mahmoud et al. designed and implemented a dynamic AES-128 with key dependent s-boxes using pseudo random sequence generator with linear feedback shift Register. The quality of the implemented s-boxes is experimentally investigated, and compared with original AES in terms of security analysis and simulation time. In their work [19], Sliman Arrag et al. improved s-box complexity through using nonlinear transformation algorithm. Further, they also adjusted key expansion schedule and use s-box lookup table to make it dynamic. Fatma Ahmed et al. [20] proposed s-boxes by using dynamic key and employed it as a repository for randomly selecting s-boxes in AES algorithm.

Using pseudo-random generators have also been broadly employed to design dynamic key-dependent s-boxes. Following the approaches in [5,8,18], Adi Reddy et al. [21] enhanced the AES security by designing s-boxes using random number generator for sub keys in key expansion module of their algorithm. The work showed that the proposed s-boxes are free from linear and differential cryptanalysis attack, and also it required less memory with high processing speed compared to other existing improvements. In [22], Kazlauskas et al. modified the existing AES algorithm by generating key-dependent s-boxes using random sequences. The authors claim that the new generated algorithm outperform the traditional AES. Balajee Maram et al. [23] generated key-dependent s-boxes by using Pseudo-Random generator. Their statistical analysis shows that the proposed algorithm could generate s-boxes faster than other available algorithms.

Recently, Shishir Katiyar et al. [24] generated dynamic s-boxes by using logistic maps. The efficiency of the proposed dynamic s-box was reviewed and analyzed over static s-box. The carried out experiments have shown that the key-dependent s-box satisfies all the cryptographic properties of good s-box and can enhance the security due to its dynamic nature. In [25], Tianyong Ao et al. made affine transformation key-dependent to generate dynamic s-boxes for their algorithm. The authors investigations revealed that the algebraic degree of an s-box is conditional invariant under affine transformation. Unal C. et al. [26] proposed a secure image encryption algorithm design using dynamic chaos-based s-box. The work showed that the developed s-box based image encryption algorithm is secure and speedy. In [27], Agarwal P. et al. developed a key-dependent dynamic s-boxes using dynamic irreducible polynomial and affine constant. This latter algorithm was used by Amandeep Singh et al. to [28] develop a new dynamic AES in which s-boxes are made completely key-dependent. In [29], Iqtadar Hussain et al. proposed an encryption algorithm based on the substitution–permutation performed by the S8 Substitution boxes and also incorporates three different chaotic maps. The presented simulation and statistical results showed that the proposed encryption scheme is secure against different attacks and resistant to the channel noise.

Despite the extensive works by many researcher towards designing key-dependent s-boxes, Chuck Easttom [30] showed that while key-dependent variations of Rijndael are sufficiently secure, they do not demonstrate improved non-linearity over the standard Rijndael s-box, instead they do introduce additional processing overhead. To address this claim, Amir Anees et al. [31] proposed a new method for creating multiple substitution boxes with the same algebraic properties using permutation of symmetric group on a set of size 8 and bitwise XOR operation. Their analysis demonstrated that the proposed substitution boxes can resist differential and linear cryptanalysis and sustain algebraic attacks. Ultimately to further extend the latter work, we propose a novel method to generate key dependent s-boxes with identical algebraic properties by applying two permutations on both of the inputs and outputs vectors of an initial s-box. A rigorous analysis is also presented to evaluate the properties of the newly created s-boxes particularly the bijection, nonlinearity, SAC, and BIC invariant.

The remainder of this paper is organized into following sections: Section 2 discusses in details the common algebraic properties of the s-box, Section 3 presents main theorem and describes the proposed key-dependent dynamic s-box generation algorithm. The conclusion is provided in Section 4.

2. Preliminaries

A Boolean function of n inputs, $f(x_1, x_2, \ldots, x_n)$, is a function of the form $f : \{0,1\}^n \to \{0,1\}$. It can be regarded as a binary vector $\mathbf{f}$ of length 2^n, where $\mathbf{f}$ is the rightmost column of the truth table describing this function. We denote the set of all Boolean functions of n inputs B_n.

The Boolean functions can serve as the n output bits of the s-box. Let $f_1, f_2 \ldots f_n$ be the n Boolean functions, where each function f_i corresponds to a binary vector $\mathbf{f}_i$ of length 2^n. Then the s-box $S = [\mathbf{f}_1, \mathbf{f}_2 \ldots \mathbf{f}_n]$ is a $2^n \times n$ bit matrix with the $\mathbf{f}_i$ as column vectors. Any given input vector $x = x_1, x_2, \ldots, x_n$, maps to an output vector $y = y_1, y_2, \ldots, y_n$, by the assignment $y_i = f_i(x_1, x_2, \ldots, x_n)$.

The main purpose of this paper is providing a key-dependent algorithm that generates a set of Boolean functions $f_1, f_2, \ldots, f_n$, such that the corresponding s-box is bijective, nonlinear, and fulfills SAC and BIC. Before introducing this algorithm, let us first revise these four algebraic properties.

2.1. Bijection

It ensures that all possible 2^n n-bit input vectors will map to distinct output vectors (i.e., the s-box is a permutation of the integers from 0 to $2^n - 1$).

Proposition 2.1 ([32]). *The necessary and sufficient condition for the s-box S to be bijective is that any linear combination of the columns of S has Hamming weight 2^{n-1}. (i.e., $wt(a_1\mathbf{f}_1 \oplus a_2\mathbf{f}_2 \oplus \cdots \oplus a_n\mathbf{f}_n) = 2^{n-1}$, where the $a_i \in \{0,1\}$ and the a_i are not all simultaneously zero).*

2.2. Nonlinearity

It ensures that the s-box is not a linear mapping from input vectors to output vectors (since this would render the entire cryptosystem easily breakable).

The nonlinearity N_f of a function f is defined [33] as the minimum Hamming distance between that function and every linear function. (i.e., $N_f = min_{l \in L_n} d_H(f,l)$, where L_n is a set of the whole linear and affine functions and $d_H(f,l)$ denotes the Hamming distance between f and l)

Remark 2.2. Pieprzyk and Finkelstein [34] claim that the highest nonlinearity achievable with 0-1 balanced functions can be calculated by the following equation

$$N_f = \begin{cases} \sum_{\frac{1}{2}(n-3) \le i \le n-3} 2^{i+1} & for \quad n = 3,5,7,\ldots, \\ \sum_{\frac{1}{2}(n-4) \le i \le n-4} 2^{i+2} & for \quad n = 4,6,8,\ldots. \end{cases} \tag{2.1}$$

Remark 2.3. Carlisle Adams and Stafford Tavares [32] stated that if the n Boolean functions of an s-box S are nonlinear, then S is guaranteed to be nonlinear at the bit level and at the integer level.

Lemma 2.4 ([35]). *Let f be a Boolean function over $\{0,1\}^n$, B be an $n \times n$ nonsingular matrix, and β a constant vector from $\{0,1\}^n$. Then the function $g(x) = f(xB \oplus \beta)$ has the same nonlinearity as the function f so $N_g = N_f$.*

2.3. Strict avalanche criterion

SAC was introduced by Webster and Tavares [36]. Informally, an s-box satisfies SAC if a single bit change on the input results in changes on a half of output bits. More formally, a function $f : \{0,1\}^n \to GF(2)$ satisfies the SAC if $f(x) \oplus f(x \oplus \gamma)$ is balanced for all γ whose weight is 1, (i.e., $wt(\gamma) = 1$). In other words, the SAC characterizes the output when there is a single bit change on the input.

Theorem 2.5 ([35]). *Let $f : \{0,1\}^n \to GF(2)$ be a Boolean function and A be an $n \times n$ nonsingular matrix with entries from $GF(2)$. If $f(x) \oplus f(x \oplus \gamma)$ is balanced for each row γ of A, then the function $\psi(x) = f(xA)$ satisfies the SAC.*

2.4. Bit independent criterion

Given two Boolean functions f_j, f_k in an s-box, if $f_j \oplus f_k$ is highly nonlinear and meets the SAC, then the correlation coefficient of each output bit pair may be close to 0 when one input bit is flipped. Thus, we can check the BIC of the s-box by verifying whether $f_j \oplus f_k$ $(j \neq k)$ of any two output bits of the s-box meets the nonlinearity and SAC [36].

3. The main theorem and proposed key-dependent dynamic s-box algorithm

Definition 3.1. A permutation matrix is a matrix obtained by permuting the rows of an $n \times n$ identity matrix according to some permutation of the numbers 0 to $n-1$. Every row and column therefore contains precisely a single 1 with 0s everywhere else, and every permutation corresponds to a unique permutation matrix.

Lemma 3.2. *Let X be the identity s-box with the $2^n \times n$ bit matrix $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$ and P_1 be a permutation matrix of size $n \times n$. There exist a permutation matrix Q_1 of size $2^n \times 2^n$ such that $Q_1 X = X P_1$.*

Proof. The post-multiplying of the permutation matrix P_1 with the matrix X to form the matrix $W_1 = X P_1$ results in permuting columns of the matrix X. Using Proposition 2.1, the $2^n \times n$ bit matrix W_1 is bijective s-box. Therefore, converting the rows of the matrix W_1 to the decimal representation provides a permutation P_3 of size 2^n.

Let Q_1 be the permutation matrix of size $2^n \times 2^n$ corresponding to the permutation P_3. Since the pre-multiplying of the permutation matrix Q_1 with the matrix X to form the matrix $Q_1 X$ results in permuting rows of the matrix X, then $Q_1 X = X P_1$. □

Theorem 3.3. *Let X be the identity s-box with the $2^n \times n$ bit matrix $[\mathbf{x}_1, \mathbf{x}_2, \ldots, \mathbf{x}_n]$, Y be an s-box with the $2^n \times n$ bit matrix $[\mathbf{y}_1, \mathbf{y}_2, \ldots, \mathbf{y}_n]$, P_1, P_2 be permutation matrices of size $n \times n$, and Q_1 be the permutation matrix of size $2^n \times 2^n$ satisfying $Q_1 X = X P_1$. The $2^n \times n$ bit matrix $Q_1 Y P_2$ is new s-box with the same algebraic properties: bijection, nonlinearity, SAC, BIC as the initial s-box Y.*

Proof. Let the n Boolean functions $f_1, f_2 \ldots f_n$ correspond to the column vectors $\mathbf{y}_i = \mathbf{f}_i$ of the matrix Y.

Since, the post-multiplying of the permutation matrix P_2 with the matrix Y to form the matrix $Y P_2$ results in permuting columns of the matrix $Y = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n]$, then it is clear that the $2^n \times n$ bit matrix $Y P_2$ is new s-box with the same algebraic properties: bijection, nonlinearity, SAC, BIC as the initial s-box Y. Therefore, it is enough to prove that the $2^n \times n$ bit matrix $Q_1 Y = [Q_1\mathbf{f}_1, Q_1\mathbf{f}_2, \ldots, Q_1\mathbf{f}_n]$ is new s-box with the same algebraic properties: bijection, nonlinearity, SAC, BIC as the initial s-box Y.

(a) Bijection:

The pre-multiplying of the permutation matrix Q_1 with the matrix Y to form the matrix $Q_1 Y$ results in permuting rows of the matrix Y. Using Proposition 2.1, $Q_1 Y$ is bijective s-box.

(b) Nonlinearity and SAC:

Introduce $g_1(x), g_2(x), \ldots, g_n(x)$ to be the Boolean functions defined by $g_i(x) = f_i(x P_1)$ for $i = 1, \ldots, n$. Using Lemma 2.4, $N_{f_i} = N_{g_i}$ for $i = 1, \ldots, n$. Also, if the functions $f_i(x)$ satisfy the SAC for $i = 1, \ldots, n$, then $f_i(x) \oplus f_i(x \oplus \gamma)$ is balanced for each row γ of P_1 for $i = 1, \ldots, n$. Therefore, using Theorem 2.5, the functions $g_i(x)$ satisfy the SAC for $i = 1, \ldots, n$. Hence, the s-box $[\mathbf{g}_1, \mathbf{g}_2, \ldots, \mathbf{g}_n]$ satisfies the SAC and has the same nonlinearity as the initial s-box $Y = [\mathbf{f}_1, \mathbf{f}_2, \ldots, \mathbf{f}_n]$.

Finally, since $Y = [\mathbf{f}_1(X), \mathbf{f}_2(X), \dots, \mathbf{f}_n(X)]$ and $Q_1 X = X P_1$, then the s-box

$$\begin{aligned}[\mathbf{g}_1(X), \mathbf{g}_2(X), \dots, \mathbf{g}_n(X)] &= [\mathbf{f}_1(XP_1), \mathbf{f}_2(XP_1), \dots, \mathbf{f}_n(XP_1)] \\ &= [\mathbf{f}_1(Q_1X), \mathbf{f}_2(Q_1X), \dots, \mathbf{f}_n(Q_1X)] \\ &= [Q_1\mathbf{f}_1(X), Q_1\mathbf{f}_2(X), \dots, Q_1\mathbf{f}_n(X)] = Q_1Y.\end{aligned} \tag{3.2}$$

(c) BIC:

Assume that the function $h_{ij}(x) = f_j(x) \oplus f_k(x)$ of any two different output bits f_j and f_k of the s-box Y meets the nonlinearity and SAC. Introduce $k_{ij}(x)$ to be the Boolean functions defined by the function $k_{ij}(x) = h_{ij}(xP_1)$. Using Lemma 2.4, $N_{h_{ij}} = N_{k_{ij}}$ for $i \neq j$. Similarly, using Theorem 2.5, the functions $k_{ij}(x)$ satisfy the SAC for $i \neq j$. Therefore, the function $k_{ij}(x) = h_{ij}(xP_1) = f_j(xP_1) \oplus f_k(xP_1) = g_j(x) \oplus g_k(x)$ meets the nonlinearity and SAC. Hence, the s-box $[\mathbf{g}_1, \mathbf{g}_2, \dots, \mathbf{g}_n]$ satisfies BIC. □

The proposed method for key-dependent dynamic s-boxes consists of permutations of the inputs and outputs vectors of an initial s-box. The following algorithm provides s-boxes with identical algebraic properties. The steps are summarized as follows.

1. Express the initial $n \times n$ s-box as a vector IS consisting of different 2^n values ranging between 0 and $2^n - 1$.
2. Compute the matrix Y of size $2^n \times n$ by evaluating the binary representation of the initial s-box. Similarly, compute the matrix X of size $2^n \times n$ by evaluating the binary representation of the identity s-box.
3. Use the key to construct two permutations σ_1 and σ_2 of different n values ranging between 0 and $n - 1$.
4. Construct the corresponding permutation matrices P_1 and P_2 of size $n \times n$ for the two permutations σ_1 and σ_2.
5. Compute the matrix $W_1 = XP_1$ of size $2^n \times n$.
6. Construct the corresponding permutation P_3 of size 2^n by getting back the decimal representation of the matrix W_1 as a vector.
7. Construct the permutation matrix Q_1 of size $2^n \times 2^n$ corresponding to the permutation P_3.
8. Compute the matrix $W_2 = Q_1 Y P_2$ of size $2^n \times n$.
9. Construct the dynamic key-dependent s-box by getting back the decimal representation of the matrix W_2 as the vector NS.
10. Detect the possible fixed point and reverse fixed point of the constructed vector NS.
11. In case there are fixed points or reverse fixed points, update the initial permutations using $\sigma_1 = \bar{\sigma}_1\sigma_1, \sigma_2 = \bar{\sigma}_2\sigma_2$ such that $\bar{\sigma}_1, \bar{\sigma}_2 \in S_n$ where S_n is the symmetric group and then return back to step 4. Otherwise, end the algorithm and produce the vector NS as clone dynamic key-dependent s-box (see Fig. 1).

An efficient Maple code implementation of the method described in this section is presented in Appendix.

Remark 3.4. The two permutations σ_1 and σ_2 of size n are extracted from a key which can be of any bits size resistant to the used cryptosystem. The two permutations could be extracted from the key by the factorial number system and Lehmer code using $2\lceil \log_2(n! - 1) \rceil$ bits of the key where there is one to one correspondence between the set of all permutations of size n and the set of integers numbers $\{0, 1, \dots, n! - 1\}$.

4. Performance analysis

This section provides demonstration of how our algorithm can be applied to construct clone copies for a given s-box while preserving its cryptographic features and strength. The application is independent of the method used for construction of the given s-box. In case of the initial given s-box having fixed points or reverse fixed points, the algorithm can be applied to obtain improved clone versions where all the fixed points and reverse fixed points are removed but the specifications like bijection, nonlinearity, SAC, and BIC are conserved with same strength as the initial given s-box. This adds particular significance and increases the scope of applications of the algorithm in the context of the recent analysis [37] of the exploitable weakness of fixed point and reverse fixed point contained in s-boxes.

The performance of our method is illustrated through the following two examples:

Example 4.1. Demonstration of algorithm for $n = 4$

We use the initial 4×4 s-box given as a vector IS

$$IS = [9, 13, 10, 15, 11, 14, 7, 3, 12, 8, 6, 2, 4, 1, 0, 5]^t \tag{4.3}$$

constructed by Carlisle Adams and Stafford Tavares [32]. Let us assume that the key gives the two permutations $\sigma_1 = (1, 2, 0, 3)$, $\sigma_2 = (3, 2, 0, 1)$ of size 4.

The corresponding $X, Y, P_1, P_2, W_1, P_3, Q_1, W_2$ are found as

$$X = \begin{bmatrix} 0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \\ 0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1 \\ 0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1 \\ 0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \end{bmatrix}^t$$

$$Y = \begin{bmatrix} 1&1&0&1&1&0&1&1&0&0&0&0&0&1&0&1 \\ 0&0&1&1&1&1&1&1&0&0&1&1&0&0&0&0 \\ 0&1&0&1&0&1&1&0&1&0&1&0&1&0&0&1 \\ 1&1&1&1&1&1&0&0&1&1&0&0&0&0&0&0 \end{bmatrix}^t$$

$$P_1 = \begin{bmatrix} 0&1&0&0 \\ 0&0&1&0 \\ 1&0&0&0 \\ 0&0&0&1 \end{bmatrix}, P_2 = \begin{bmatrix} 0&0&0&1 \\ 0&0&1&0 \\ 1&0&0&0 \\ 0&1&0&0 \end{bmatrix}$$

$$W_1 = \begin{bmatrix} 0&0&0&0&1&1&1&1&0&0&0&0&1&1&1&1 \\ 0&1&0&1&0&1&0&1&0&1&0&1&0&1&0&1 \\ 0&0&1&1&0&0&1&1&0&0&1&1&0&0&1&1 \\ 0&0&0&0&0&0&0&0&1&1&1&1&1&1&1&1 \end{bmatrix}^t$$

$$P_3 = (0, 2, 4, 6, 1, 3, 5, 7, 8, 10, 12, 14, 9, 11, 13, 15)$$

$$Q_1 = \begin{bmatrix} 1&0&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&1&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&1&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&1&0&0&0&0&0&0&0&0&0 \\ 0&1&0&0&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&1&0&0&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&1&0&0&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&1&0&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&1&0&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&1&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&1&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&1&0 \\ 0&0&0&0&0&0&0&0&0&1&0&0&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&1&0&0&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&1&0&0 \\ 0&0&0&0&0&0&0&0&0&0&0&0&0&0&0&1 \end{bmatrix}$$

$$W_2 = \begin{bmatrix} 0&0&0&1&1&1&1&0&1&1&1&0&0&0&0&1 \\ 1&1&1&0&1&1&1&0&1&0&0&0&1&0&0&0 \\ 0&1&1&1&0&1&1&1&0&1&0&0&0&1&0&0 \\ 1&0&1&1&1&1&0&1&0&0&0&0&0&0&1&1 \end{bmatrix}^t$$

Finally, the vector NS

$$NS = [10, 6, 14, 13, 11, 15, 7, 12, 3, 5, 1, 0, 2, 4, 8, 9]^t \tag{4.4}$$

provides the dynamic key-dependent s-box, having the same four algebraic properties as the initial vector IS as shown in the following table (see Table 1).

Remark 4.2. The algorithm was applied using the initial 4×4 s-box and all the 4! permutations of size 4. As a result, $(4!)^2 = 576$ different

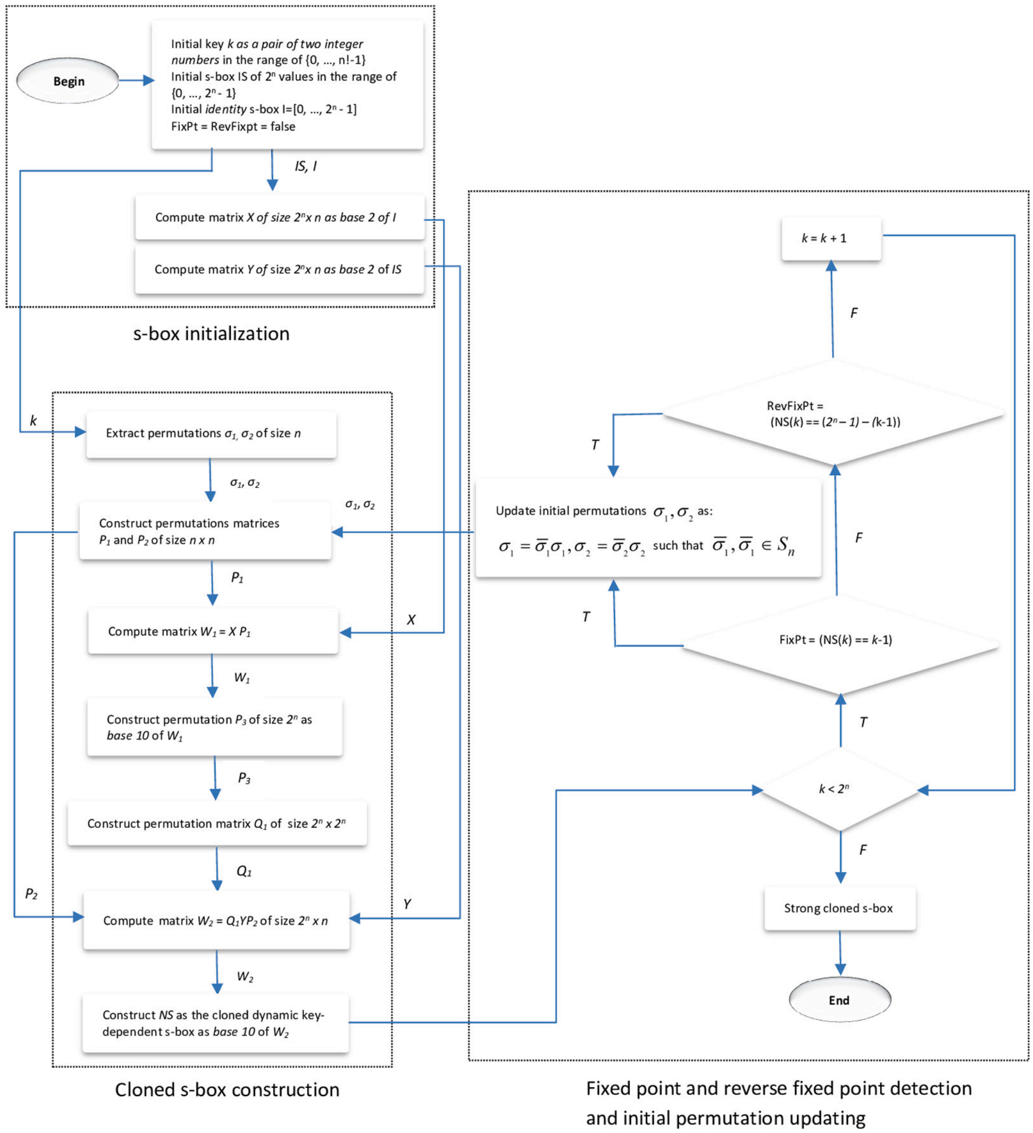

Fig. 1. Flowchart of constructing cloned key-dependent s-box.

Table 1
Comparison of the algebraic properties of the initial s-box IS (Eq. (4.3)) and its clone copy NS (Eq. (4.4)) resulting from applying the two permutations $\sigma_1 = (1,2,0,3)$, $\sigma_2 = (3,2,0,1)$ on IS.

	Nonlinearity			SAC				BIC of nonlinearity				BIC of SAC			
	min	max	avg	min	max	avg	SD	min	max	avg	SD	min	max	avg	SD
Initial s-box IS	4	4	4	0	1	0.5	0.132583	4	4	4	0	0.4375	0.75	0.552083	0.104686
Clone copy s-box NS	4	4	4	0	1	0.5	0.132583	4	4	4	0	0.4375	0.75	0.552083	0.104686

s-boxes were generated and it was verified that all have the same four algebraic properties as the initial s-box.

Example 4.3. Demonstration of algorithm for $n = 8$

We use the initial 8×8 AES s-box constructed by Joan Daemen and Vincent Rijmen [38] given in Table 2. Let us assume that the key gives the two permutations $\sigma_1 = (1,2,0,6,5,7,3,4)$, $\sigma_2 = (5,7,3,4,1,2,0,6)$ of size 8. Similarly, applying the new method provides new s-box given in Table 3 with the same four algebraic properties as the initial AES s-box (see Table 4).

5. Conclusion

This work investigates the question of generating key-dependent dynamic $n \times n$ clone s-boxes having the same algebraic properties. Using initial s-box, we provide an algorithmic approach to generate clone s-boxes which have the same genetic traits like bijection, nonlinearity, SAC, and BIC. Invariance of the bijection, nonlinearity, SAC, and BIC for the generated clone copies is proved. The flow chart and Maple code of the presented algorithm are also given. The efficiency of the algorithm is tested through examples. In conclusion, instead of focusing on finding ways to generate strong s-boxes, it may be enough to start

Table 2
Presentation of AES s-box [38] in 16 × 16 matrix form.

R/C	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
	99	124	119	123	242	107	111	197	48	1	103	43	254	215	171	118
	202	130	201	125	250	89	71	240	173	212	162	175	156	164	114	192
	183	253	147	38	54	63	247	204	52	165	229	241	113	216	49	21
	4	199	35	195	24	150	5	154	7	18	128	226	235	39	178	117
	9	131	44	26	27	110	90	160	82	59	214	179	41	227	47	132
	83	209	0	237	32	252	177	91	106	203	190	57	74	76	88	207
	208	239	170	251	67	77	51	133	69	249	2	127	80	60	159	168
	81	163	64	143	146	157	56	245	188	182	218	33	16	255	243	210
	205	12	19	236	95	151	68	23	196	167	126	61	100	93	25	115
	96	129	79	220	34	42	144	136	70	238	184	20	222	94	11	219
	224	50	58	10	73	6	36	92	194	211	172	98	145	149	228	121
	231	200	55	109	141	213	78	169	108	86	244	234	101	122	174	8
	186	120	37	46	28	166	180	198	232	221	116	31	75	189	139	138
	112	62	181	102	72	3	246	14	97	53	87	185	134	193	29	158
	225	248	152	17	105	217	142	148	155	30	135	233	206	85	40	223
	140	161	137	13	191	230	66	104	65	153	45	15	176	84	187	22

Table 3
Presentation of clone copy s-box as 16 × 16 matrix resulting from applying the two permutations $\sigma_1 = (1,2,0,6,5,7,3,4)$, $\sigma_2 = (5,7,3,4,1,2,0,6)$ on AES s-box shown in Table 2.

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
165	175	199	189	31	183	181	105	48	28	178	147	224	146	157	68
238	226	142	239	127	140	190	89	67	212	161	166	253	247	57	104
121	162	187	9	24	93	234	170	214	44	26	78	23	156	204	201
69	150	49	12	134	144	136	27	101	82	53	216	87	34	115	74
6	173	223	244	32	180	235	143	131	203	52	188	182	230	229	72
14	109	39	38	108	103	83	42	41	128	3	250	119	191	30	84
73	159	13	50	236	62	59	167	85	15	177	240	123	186	126	208
193	92	98	77	227	133	106	55	242	232	217	20	154	117	43	251
209	113	215	169	192	63	51	71	163	0	4	102	99	125	95	179
8	164	18	40	233	225	202	210	35	1	194	22	228	248	122	111
5	185	132	66	96	91	148	80	7	110	17	207	158	141	160	152
237	174	120	153	81	61	107	116	88	112	254	129	100	56	205	21
124	196	90	135	75	252	76	65	149	222	145	19	241	54	25	249
168	64	245	198	130	197	172	47	94	211	2	231	206	36	255	195
137	86	219	176	221	10	155	243	37	171	200	58	46	118	97	218
29	79	45	220	139	213	151	16	33	60	70	246	114	184	11	138

Table 4
Comparison of the algebraic properties of AES s-box (Table 2) and its clone copy (Table 3).

	Nonlinearity			SAC				BIC of nonlinearity				BIC of SAC			
	min	max	avg	min	max	avg	SD	min	max	avg	SD	min	max	avg	SD
AES s-box	112	112	112	0.453125	0.5625	0.504883	0.015678	112	112	112	0	0.480469	0.525391	0.504604	0.011271
Clone copy of AES s-box	112	112	112	0.453125	0.5625	0.504883	0.015678	112	112	112	0	0.480469	0.525391	0.504604	0.011271

with one strong s-box such as AES s-box, S8 AES s-box, APA s-box, and Gray s-box and then get its clone copies.

CRediT authorship contribution statement

Ahmad Y. Al-Dweik: Conceptualization, Software, Writing – original draft. **Iqtadar Hussain:** Writing – review & editing, Supervision, Software. **Moutaz Saleh:** Methodology, Software, Writing – original draft, Writing – review & editing. **M.T. Mustafa:** Writing – review & editing, Methodology, Investigation, Software.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

The authors would like to thank Qatar University for its support and excellent research facilities.

Appendix. Maple code for the proposed algorithm

Listing 1: Maple Procedure for converting a permutation σ to a permutation matrix R

```
permatrix:=proc(sigma)
local R,i;
R:=Matrix(nops(sigma),nops(sigma),[0]);
for i from 1 to nops(sigma) do R[i,sigma[i]+1]:=1; end do;
R;
end proc:
```

Listing 2: Maple Procedure for evaluating the binary representation of an s-box S as a Boolean matrix M of size $2^n \times n$

```
BLmatrix := proc(S,n)
local M;
M:=Matrix([seq(convert([S[i]],base,10,2),i=1..nops(S))]);
end proc:
```

Listing 3: Maple Procedure for evaluating the decimal representation of a Boolean matrix M of size $2^n \times n$ as an s-box S

```
Sbox := proc(M,n)
local S;
S:=[seq(add(convert(Row(M,i),list)[j]*2^(j-1),j=1..n),i=1..RowDimension(M))];
end proc:
```

Listing 4: Maple code for the proposed algorithm for constructing new s-box NS using the initial s-box IS and the two permutations σ_1 and σ_2 which extracted from the key

```
NS := proc(IS,sigma1,sigma2)
local n,Y,IDSBox,X,P1,P2,W1,P3,Q1,W2,NS;
n:=log[2](nops(IS));
Y:=BLmatrix(IS,n);
IDSBox:=[seq(i,i=0..2^n-1)]; X:=BLmatrix(IDSBox,n);
P1:=permatrix(sigma1); P2:=permatrix(sigma2);
W1:=X.P1; P3:=Sbox(W1,n); Q1:=permatrix(P3);
W2:=Q1.Y.P2;
NS:=Sbox(W2,n)
end proc:
```

Listing 5: Maple code for deduction of fixed point and reverse fixed point for s-box S

```
FIXP :=proc(S)
local n,i,FPS,RFPS;
n:=nops(S);
FPS:={};
RFPS:={};
for i from 1  to n do
if S[i]=i-1 then
FPS:={S[i]}union  FPS;
elif  S[i]=255-(i-1) then
RFPS:={S[i]}union  RFPS;
end if;
end do;
[FPS,RFPS];
end proc:
```

References

[1] Ahmad Musheer, Khan Parvez, Ansari Mohd. A simple and efficient key-dependent sbox design using Fisher–Yates Shuffle technique. In: International conference on security in computer networks and distributed systems SNDS. Recent trends in computer networks and distributed systems security, Springer; 2014, p. 540–50.

[2] Dawson M, Tavares S. An expanded set of design criteria for substitution boxes and their use in strengthening DES-like cryptosystems. In: IEEE Pacific rim conference on communications, computers and signal processing, 9 May 1991, pp. 191-195.

[3] Kwok-WoWong YongWang, Liao Xiaofeng, Xiang Tao. A block cipher with dynamic Sboxes based on tent map. Commun Nonlinear Sci Numer Simul 2009;(14):3089–99.

[4] Krishnamurthy GN, Ramaswamy V. Making AES stronger: AES with key dependent S-Box. Int J Comput Sci Netw Secur 2008;8:388–98.

[5] Piotr M. Generating pseudorandom S-Boxes a method of improving the security of cryptosystems based on block ciphers. J Telecommun Inf Technol 2009.

[6] Stoianov N. One approach of using key-dependent S-BOXes in AES. In: Multimedia communications, services and security. Springer; 2011, p. 317–23.

[7] ElGhafar A, Rohiem A, Diaa A, Mohammed F. Generation of AES key dependent SBoxes using RC4 algorithm. In: 13th international conference on aerospace sciences aviation technology, ASAT- 13. 2009, p. 26–8.

[8] Kazys K, Jaunius K. Key-dependent S-Box generation in AES block cipher system. Informatica 2009;23–34.

[9] Ghada Z, Abdennaceur K, Fabrice P, Daniele F. On dynamic chaotic S-BOX. IEEE; 2009.

[10] Cui J, Huang L, Zhong H, Chang C, Yang W. An improved AES S-Box and its performance analysis. Int J Innovative Comput Inf Control 2011.

[11] Anna G. Cryptographic properties of modified AES-like S-boxes. Ann UMCS Inf AI XI 2011;2:37–48.

[12] Julia J, Ramlan M, Salasiah S, Jazrin R. Enhancing advanced encryption standard SBox generation based on round key. Int J Cyber-Security Digit Forensics 2012;3:183–8.

[13] Hosseinkhani R, Haj Seyyed Javadi H. Using cipher key to generate dynamic S-Box in AES cipher system. Int J Comput Sci Secur 2012;6:19–28.

[14] Hussain Iqtadar, Mahmood Tariq Shah Hasan, Gondal Muhammad Asif. Construction of S8 Liu J S-boxes and their applications. Comput Math Appl 2012;64:2450–8.

[15] Hussain Iqtadar, Shah Tariq, Gondal Muhammad Asif, Mahmood Hasan. S8 affine-power-affine S-boxes and their applications. Neural Comput Appl 2012;21(Suppl 1):S377–83.

[16] Kazymyrov O, Kazymyrova V, Oliynykov R. A method for generation of highnonlinear S-Boxes based on gradient descent. IACR Cryptol 2013.

[17] Dara M, Manochehri K. A novel method for designing S-Boxes based on chaotic logistic maps using cipher key. World Appl Sci J 2013;28:2003–9.

[18] Mohammed Mahmoud E, Abd El Hafez A, Talaat A, Zekry A. Dynamic AES-128 with key-dependent s-box. Int J Eng Res Appl 2013;3:1662–70.

[19] Arrag S, Hamdoun A, Tragha A, Eddine Khamlich S. Implementation of stronger AES by using dynamic S-Box dependent of master key. J Theor Appl Inf Technol 2013.

[20] Ahmed F, Elkamchouchi D. Strongest AES with S-Boxes bank and dynamic key MDS matrix (SDK-AES). Int J Comput Commun Eng 2013.

[21] Adi Narayana Reddy K, Vishnuvardhan B. Secure linear transformation based cryptosystem using dynamic byte substitution. Int J Secur 2014.

[22] Kazys K, Gytis V, Robertas S. An algorithm for key-dependent S-Box generation in block cipher system. Informatica 2015;26:51–65.

[23] Balajee Maram K, Gnanasekar JM. Evaluation of key dependent S-Box based data security algorithm using hamming distance and balanced output. TEM J 2016.

[24] Katiyar S, Jeyanthi N. Pure dynamic S-box construction. Int J Comput 2016.

[25] Tianyong A, Jinli R, Kui D, Xuecheng Z. Construction of high quality keydependent S-Box. IAENG Int J Comput Sci 2017.

[26] Unal C, Sezgin K, Ihsan P, Ahmet Z. Secure image encryption algorithm design using a novel chaos based S-Box. Chaos Solitons Fractals 2017;95:92–101.

[27] Agarwal P, Singh A, Kilicman A. Development of key dependent dynamic S-Boxes with dynamic irreducible polynomial and affine constant. Adv Mech Eng 2018.

[28] Singh A, Agarwal P, Chand M. Image encryption and analysis using dynamic AES. In: 5th international conference on optimization and applications. 2019.

[29] Hussain Iqtadar, Anees Amir, Aslam Muhammad, Ahmed Rehan, Siddiqui Nasir. A noise resistant symmetric key cryptosystem based on S8 S-boxes and chaotic maps. Eur Phys J Plus 2018;133:167.

[30] Easttom C. An examination of inefficiencies in key dependent variations of the Rijndael S-Box. In: Electrical engineering (ICEE), Iranian conference on 2018 may 8. IEEE; 2018, p. 1658–63.

[31] Anees Amir, Chen Yi-Ping Phoebe. Designing secure substitution boxes based on permutation of symmetric group. Neural Comput. Appl. 2020;32(11):7045–56.

[32] Adams CM, Tavares SE. The structured design of cryptographically good S-Boxes. J Cryptol 1990;3(1):27–41.

[33] Shannon CE. Communication theory of secrecy system. Bell Syst Tech J 1949;28:656715.

[34] Pieprzyk J, Finkelstein G. Towards effective nonlinear cryptosystem design. IEE Proc, Part E: Comput Digit Tech 1988;135:325–35.

[35] Pieprzyk J, Hardjono T, Seberry J. Fundamentals of computer security. Springer Science and Business Media; 2013.

[36] Webster AF, Tavares SE. On the design of S-Boxes. In: Proceedings of advances in cryptology. Lecture notes in computer science, vol. 218, Springer-Verlag; 1986, p. 523–34.

[37] Liu Hongjun, Kadir Abdurahman, Xu Chengbo. Cryptanalysis and constructing S-Box based on chaotic map and backtracking. Appl Math Comput 2020;376:125153.

[38] Daemen J, Rijmen V. AES proposal: Rijndael. 1999, http://csrc.nist.gov/archive/aes/rijndael/Rijndael-ammended.pdf.