Journal of Engineering Science and Technology
Vol. 21, No. 2 (2026) 326 - 353

SIMPLE YET SECURE: CONSTRUCTION OF SECURE LOW-DEGREE S-BOXES WITH EFFICIENT MASKING VIA THRESHOLD IMPLEMENTATION

KOK-AN PANG[1], ABDUL-LATIP S. FAISAL[2,3,*], NORZIANA JAMIL[4], HAZLIN A. RANI[5]

[1]Fakulti Teknologi Maklumat dan Komunikasi, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, Durian Tunggal, 76100, Melaka, Malaysia
[2]Fakulti Kecerdasan Buatan dan Keselamatan Siber, Universiti Teknikal Malaysia Melaka, Hang Tuah Jaya, Durian Tunggal, 76100, Melaka, Malaysia
[3]Symmetric Division, Malaysia Cryptology Technology and Management Centre, c/o Universiti Putra Malaysia, 43400 UPM, Serdang, Selangor, Malaysia
[4]College of IT, United Arab Emirates University, Al Ain, 15551, United Arab Emirates
[5]Cryptography Development Department, Cybersecurity Malaysia, Menara Cyber Axis, Cyberjaya, 63000, Selangor, Malaysia
*Corresponding Author: shekhfaisal@utem.edu.my

Abstract

Cryptographic S-boxes are essential components of cryptographic primitives that induce confusion by making the relationship between the secret key and the ciphertext complex and obscure. An S-box must satisfy the Strict Avalanche Criterion (SAC), meaning that flipping a single input bit should alter each output bit with a probability of 0.5, ensuring that the output appears random and uncorrelated with the input. However, designing an S-box to meet this ideal is challenging, and the difficulty increases when it must also be efficiently masked using Threshold Implementation (TI). In this paper, we identify S-boxes that not only exhibit perfect SAC but also maintain uniformity when masked with TI. By constraining the algebraic degree of the S-boxes to 2, we achieve uniform TI implementations through degree-count classification of TI shares. Furthermore, our threshold implementation method applies to other degree-2 5-bit S-boxes while preserving uniformity. From the generated S-boxes, we select an S-box namely $\mathcal{P}$, that scores exceptionally well in nonlinearity, SAC, BIC, differential uniformity and linear probability. Our results contribute to the field of cryptographic security by providing a practical solution for constructing S-boxes that meet SAC requirements and integrate seamlessly with TI, thereby enhancing the security and effectiveness of side-channel countermeasures. Our work provides new insights for studies that aim to achieve uniformity in TI implementations of algorithms whose algebraic degree exceeds 2.

Keywords: Cryptography, S-box, Side channel, Strict avalanche criterion, Threshold implementation.

1. Introduction

In an increasingly interconnected world, where vast amounts of data are exchanged and stored online, the security of digital information has become more critical than ever. Ensuring digital security is paramount in today's world. Cryptographic methods play a crucial role in protecting data and maintaining privacy. The advancements and innovative approaches in this field are vital for enhancing security measures and safeguarding against cyber threats. Understanding the significance and application of these methods is essential for developing robust cryptographic systems that can effectively protect sensitive information.

One of the key areas in cryptography is symmetric-key cryptography, in which a sender and a receiver share a single secret key. The security of their communication depends entirely on keeping this key confidential, because an adversary who does not possess it cannot decrypt any messages exchanged between them. A symmetric-key primitive that has seen near-universal adoption is the Advanced Encryption Standard (AES), which is implemented in software and hardware worldwide to protect classified government information, secures financial transactions and healthcare data through its AES-256 variant, and underpins modern wireless-network protocols such as Wi-Fi Protected Access WPA2 and WPA3. With the rise of IoT-embedded technology, demand for more lightweight cryptographic primitives has increased. Modern cryptography therefore focuses not only on the security of its algorithms but also on the efficiency they deliver when deployed on resource-constrained embedded platforms.

A fundamental component within symmetric-key cryptographic algorithms is the cryptographic substitution box, commonly known as an S-box. These nonlinear transformations are crucial for obscuring the relationship between plaintext and ciphertext, thereby enhancing the algorithm's resistance to various forms of cryptanalytic attacks, such as linear [1] and differential cryptanalysis [2]. The effectiveness of an S-box is often measured by several criteria, one of which is the Strict Avalanche Criterion (SAC). The SAC [3-7] requires that a small change in the input results in a significant and unpredictable change in the output. By achieving perfect SAC, every output bit changes with a probability of 0.5 whenever a single input bit is flipped. This property is essential for maintaining the security of the cryptographic system, as it ensures that the output behaves randomly and is uncorrelated with the input.

In addition to mathematical security properties like SAC, practical implementations require resilience against physical attacks, such as side-channel analysis. These attacks exploit leakage from hardware (e.g., power consumption, electromagnetic emissions) to extract secret keys. To mitigate such threats, Threshold Implementation (TI) [8] has emerged as a provably secure countermeasure. TI is a specialized form of masking that combines secret sharing with multiparty computation principles. Its core innovation lies in distributing sensitive data (e.g., S-box inputs) into t shares, where computations are performed independently on each share.

A notable milestone in the realm of hash function cryptography was marked by the selection of KECCAK [9] as the new Secure Hash Algorithm 3 (SHA-3) standard [10]. More recently, the National Institute of Standards and Technology (NIST) has chosen ASCON [11], a cipher employing a 5-bit S-box, as the new standard for lightweight cryptography [12]. The introduction of SHA-3 and the

selection of the ASCON cipher highlight the critical importance of studying 5-bit S-boxes. Furthermore, candidates from NIST competitions such as PRIMATEs [13], ICEPOLE [14], and Shamash [15] also incorporate 5-bit S-boxes. Notably, S-boxes with odd sizes and high differential uniformity, including 5-bit S-boxes, can be constructed using Almost Perfect Nonlinear (APN) functions, which are predominantly found in odd dimensions [16-18]. These developments suggest that 5-bit S-boxes may play a pivotal role in the future of cryptography.

Designing all-around S-boxes that are both secure and lightweight is non-trivial. The S-boxes of ASCON and KECCAK [9, 11] utilize an identical bit sliced implementation, which enhances cost-efficiency. Additionally, their algebraic degree, being only 2, renders the cost of implementing TI low. However, these S-boxes exhibit limited nonlinearity. In comparison, the S-boxes of PRIMATEs and Shamash [13, 15] boast excellent nonlinearity, differential, and linear properties, despite the presence of inverse fixed points and short period rings. ICEPOLE, a lightweight authenticated encryption scheme designed for hardware environments [14], outperforms AES-GCM in hardware performance. Nevertheless, its S-box possesses a high algebraic degree, and its SAC is uneven and significantly deviates from 0.5. Kim et al.'s S-box [19] demonstrates perfect SAC, though its nonlinearity is inferior compared to other 5-bit S-boxes.

Constructing TI was often an ad-hoc process, relying on manual adjustments and heuristics to achieve the desired security properties. This changed with Baksi et al.'s method [20], which introduced a systematic and automated approach to finding TI of S-boxes. This method significantly streamlined the process by providing a more structured way to generate TI. However, Baksi's method does not guarantee uniformity, a property in TI that we will introduce in Section 6.2. Ensuring uniformity is critical for the security of the cryptographic system, as it helps prevent potential vulnerabilities that could be exploited by side-channel attacks. Nevertheless, achieving uniformity in TI remains a challenging issue [20-23].

Our contribution. In this paper, we improve Baksi's method by constructing TI with shares categorized using degree-count classification. Additionally, we develop separate TI for degree-1 and degree-2 monomials, which can be applied to create TI for any degree-2 polynomials. Beyond TI, we present a 5-bit S-box that not only achieves perfect SAC but is also well-suited for TI implementation. Our method consistently ensures uniformity and can be extended to other 5-bit S-boxes with degree-2 monomials without compromising this property.

Paper outline. The structure of the paper is organized as follows. Section 2 outlines the preliminaries of SAC, TI, and degree-count classification. Section 3 addresses the construction of TI for algebraic polynomials. Section 4 describes the construction of the S-box, while Section 5 details the construction of TI on a selected S-box based on the process outlined in Section 4. Section 6 presents an analysis of the S-box using various well-known security parameters. The applicability of our method to other S-boxes is demonstrated in Section 7. Finally, Section 8 concludes the paper.

2. Preliminaries

In this work, we define a sequence as follows. Let S be an n-bit S-box. A sequence is defined as a vector of size 2^n storing the r-th bit of $S[i]$, where i is the input of S, such that $(S_r[0], S_r[1], \dots, S_r[2^n - 1])$.

In this section, we explore the fundamental principles of SAC and offer a comprehensive overview of TI.

2.1. Strict avalanche criterion

The SAC [19, 24, 25] defines that a minor modification in the input, such as flipping a single bit, should result in substantial changes in the output. This characteristic is essential for introducing confusion in the ciphertext, thereby enhancing the security of the cryptographic system. Specifically, an S-box conforms to the SAC if, for each output bit, an alteration in each input bit affects the output bit with a probability of 0.5. As a result, each output bit must be a complex and non-linear function of the input bits, making it computationally challenging for an attacker to predict the output based on the input or vice versa. In cryptography, it is widely recognized that an S-box with an SAC value approaching 0.5 demonstrates robustness [26].

The avalanche property of S is evaluated by determining the probability of a change in S_r for $0 \leq r < n$, given an input variation $\Delta \boldsymbol{x}$ with $HW(\Delta \boldsymbol{x}) = 1$. Ideally, the SAC value is 0.5, indicating that $P(S_r(\boldsymbol{x}) + S_r(\boldsymbol{x} + \Delta \boldsymbol{x}) = 1) = 0.5$. The SAC of S_r can be computed using Eq. (1) [24, 27]. The procedure for calculating the SAC is described in Algorithm 1.

$$\delta(S, \boldsymbol{x}, \Delta \boldsymbol{x}) = \frac{\#\{\boldsymbol{x} \in \mathbb{F}_2^n | S_r(\boldsymbol{x}) + S_r(\boldsymbol{x} + \Delta \boldsymbol{x}) = 1\}}{2^n} \tag{1}$$

The SAC of an S-box can be independently evaluated for each output bit, simplifying the process of generating sequences with perfect SAC. The ability to measure the SAC of each output bit of an S-box independently allows us to identify sequences with perfect SAC without first considering the entire S-box. This significantly reduces the search time.

Algorithm 1: Calculating SAC of S-box $\mathcal{P}$

Input: S-box $\mathcal{P}$, input bit index k and output bit index ℓ of $\mathcal{P}$ such that $k, \ell \in \{0,1,2,3,4\}$.

Output: SAC value of $\mathcal{P}_\ell$ with input difference 2^k.

Start
Initialize $d = 0$
For each $0 \leq \boldsymbol{x} < 2^5$:
    Accumulate d by $\frac{\mathcal{P}_\ell(x) + \mathcal{P}_\ell(x + 2^k)}{2^5}$.
End for
Return d
End

2.2. Threshold implementation

TI [8] is a cryptographic technique designed to enhance security against side-channel attacks, particularly Differential Power Analysis (DPA). Side-channel attacks can be devastating, as they bypass the mathematical security of the

algorithm by exploiting physical leakages such as power consumption, electromagnetic emissions, or timing information. TI leverages principles from multi-party computation and secret sharing to distribute cryptographic operations across multiple shares, ensuring that no single share reveals sensitive information. The core idea of TI is to split the secret key into multiple shares. Each share is processed independently, and the final cryptographic operation is performed in such a way that the secret key is never fully reconstructed during the computation. This approach leverages the principles of secret sharing, where the security of the system relies on the distribution of the secret across multiple components. Figure 1 illustrates an example of TI using three shares implemented on an S-box.

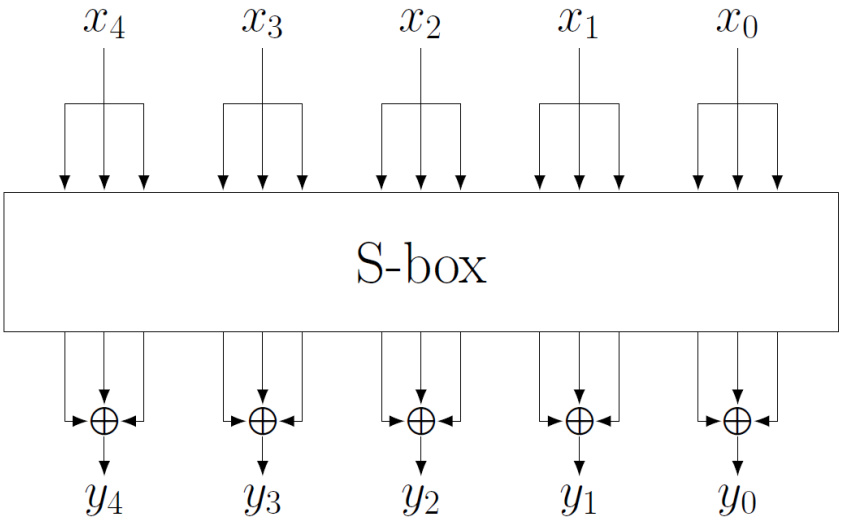

Fig. 1. A three-share TI applied to an S-box.

The minimum number of shares, denoted as d, necessary for computing the product of n variables is $d \geq n + 1$, as established in Theorem 1.

Theorem 1. The minimum number of shares, denoted as d, necessary for computing the product of n variables is $d \geq n + 1$.

The three essential characteristics of TI are non-completeness, correctness, and uniformity. Let $f: \mathbb{F}_2^n \rightarrow \mathbb{F}_2$, and each j-th share of f in TI is denoted by f_j. Definitions 1, 2, and 3 outline the rules required for each corresponding characteristic of an ideal TI.

Definition 1 (Non-completeness). Each f_j has no j-th share of x_i.

Non-completeness ensures that no single share of the computation can independently complete the cryptographic function. This means that each share only contains partial information, and only when combined with other shares can the full computation be completed. An attacker would need to access all shares simultaneously to gain any meaningful information.

Definition 2 (Correctness). TI of f is said to be correct if $\bigoplus_{j=0}^{d-1} f_j = f$.

Correctness guarantees that the threshold implementation correctly computes the cryptographic function when all shares are combined. Despite the splitting of the computation into multiple shares, the final result must be identical to the result of the original, unshared computation.

Definition 3 (Uniformity). All possible values of S-box TI share the same count.

Uniformity ensures that the distribution of the shares is consistent and indistinguishable from random noise. This characteristic is vital for preventing statistical analysis attacks, as it ensures that the shares do not reveal any patterns or biases that could be exploited by an attacker. Uniformity maintains the unpredictability and security of the cryptographic function.

2.3. Degree-count classification

Ensuring uniformity in TI continues to pose significant challenges [20-23]. To address this, we classify monomials in a polynomial using a new method termed *degree-count classification*. The definition of *degree-count classification* is provided in Definition 4.

Definition 4: Degree-count classification categorizes polynomials over $\mathbb{F}_2^n$ based on the degree of each monomial and the number of monomials at each degree level. Let $\mathbb{A}_n$ denote the set of all polynomials of degree n. In this method, every polynomial in $\mathbb{A}_n$ with a_i monomials of degree i for every $1 \leq i \leq n$ is grouped together in the degree-count class $A_{a_1,a_2,\ldots,a_n}$, where $A_{a_1,a_2,\ldots,a_n} \in \mathbb{A}_n$.

We illustrate how various polynomials are compared and grouped with degree-count classification in Examples 1, 2 and 3.

Example 1: Let $f_0 = x_a + x_b x_c$ and $f_1 = x_d + x_e x_f$. Since both f_0 and f_1 each contain one linear monomial (x_a and x_d) and one quadratic monomial ($x_b x_c$ and $x_e x_f$), f_0 and f_1 are categorised in the same degree-count class such that $f_0, f_1 \in A_{1,1}$.

Example 2: Let $f_2 = x_d + x_e + x_f x_g$. Both f_0 and f_2 contain one quadratic monomial. However, they differ in the number of linear monomials; f_0 has one linear monomial (x_a), while f_2 has two linear monomials (x_d and x_e). Therefore, f_0 and f_2 are not grouped in the same degree-count class: $f_0 \in A_{1,1}$ and $f_2 \in A_{2,1}$.

Example 3: Let $f_3 = x_d + x_e x_f + 1$. Both f_0 and f_3 contain one quadratic monomial and one linear monomial. While f_3 includes a constant term of 1, f_0 lacks a constant term. Despite this difference, both f_0 and f_3 belong to the same degree-count class, i.e., $f_0, f_3 \in A_{1,1}$.

The presence of the constant term 1 in the base polynomial does not introduce any noticeable differences to its threshold implementation. Therefore, we do not consider its existence a distinction in the degree-count classification. For this reason, we do not specifically address the threshold implementation of constants in this paper.

3. Construction of TI

For monomials of degree at most 2 with 3 shares of TI, there are two variants each for monomials of degree one and two that consist of shares from the same degree-count class. The first variant is presented in Rule 1 and Rule 2, while the second variant is shown in Rule 3 and Rule 4.

Rule 1. All monomials of degree 1 are added in the TI as $f_{(j+1) \bmod 2} = x_{a,j}$.

Rule 2. All monomials of degree 2 are included in the TI as shown in Eq. (2).

$$\begin{aligned} f_0 &= x_{0,1}x_{1,2} + x_{0,2}x_{1,1} + x_{0,2}x_{1,2} \\ f_1 &= x_{0,0}x_{1,0} + x_{0,0}x_{1,2} + x_{0,2}x_{1,0} \\ f_2 &= x_{0,0}x_{1,1} + x_{0,1}x_{1,0} + x_{0,1}x_{1,1} \end{aligned} \tag{2}$$

Rule 3. All monomials of degree 1 are added in the TI as $f_j = x_{a,(j+1) \bmod 2}$.

Rule 4. All monomials of degree 2 are included in the TI as demonstrated in Eq. (3).

$$\begin{aligned} f_0 &= x_{0,1}x_{1,1} + x_{0,1}x_{1,2} + x_{0,2}x_{1,1} \\ f_1 &= x_{0,0}x_{1,2} + x_{0,2}x_{1,0} + x_{0,2}x_{1,2} \\ f_2 &= x_{0,0}x_{1,0} + x_{0,0}x_{1,1} + x_{0,1}x_{1,0} \end{aligned} \tag{3}$$

Rule 1 can be applied in conjunction with either Rule 2 or Rule 4, and the same holds for Rule 3. Based on our findings, no additional rules beyond Rule 1, Rule 2, Rule 3, and Rule 4 can construct a TI with three shares within the same degree-count class.

We tested the combination of rules against all 2^{16} polynomials of degree at most two. There are $\binom{5}{1} = 5$ possible degree-1 monomials, $\binom{5}{2} = 10$ possible degree-2 monomials, and one constant term, giving a total of 16 distinct monomials and therefore 2^{16} distinct polynomials since each monomial may be present or absent. Threshold counts were computed using Algorithm 2. Empirical results from Algorithm 2 show that when every possible combination of rules from the two variants is applied exhaustively to all 2^{16} polynomials, uniformity is achieved with probability 1.

Having proven that TI with shares from the same degree class preserves uniformity, we now investigate the opposite case: TI constructed with shares from different degree classes. We evaluate the uniformity of such a TI scheme, as described in Eq. (4).

$$\begin{aligned} f_0 &= x_{0,1}x_{1,1} + x_{0,1}x_{1,2} + x_{0,2}x_{1,1} + x_{0,2}x_{1,2} \\ f_1 &= x_{0,0}x_{1,0} + x_{0,0}x_{1,2} + x_{0,2}x_{1,0} \\ f_2 &= x_{0,0}x_{1,1} + x_{0,1}x_{1,0} \end{aligned} \tag{4}$$

We observed that the above TI lacks uniformity for certain polynomials. This result demonstrates that TI with polynomials from various degree-count classes does not guarantee uniformity with probability 1. Therefore, TI with shares from different degree-count classes is not employed in the construction of the S-box in this paper.

Algorithm 2: Calculate the threshold count of TI

Input: A function f with degree at most 2, input $\boldsymbol{x}$, output shares of f_0, f_1, f_2
Output: Threshold count of the case $(x_4, x_3, x_2, x_1, x_0, f_0, f_1, f_2)$

Start
Initialise b_0, b_1, b_2.
Initialise $count = 0$.
Initialise a_0, a_1, and a_2.
For each $(x_{4,0}, x_{4,1}, x_{4,2}, x_{3,0}, x_{3,1}, x_{3,2}, x_{2,0}, x_{2,1}, x_{2,2}, x_{1,0}, x_{1,1}, x_{1,2}, x_{0,0}, x_{0,1}, x_{0,2})$ that satisfies $\boldsymbol{x}$:
    Initialise b_0, b_1, b_2.
    For each monomial t in f:
        If $deg(t) = 1$:
            Create TI of t based on Rule 1 or 3 and assign the shares to a_0, a_1, and a_2.
        Else if $deg(t) = 2$:
            Create TI of t based on Rule 2 or 4 and assign the shares to a_0, a_1, and a_2.
        End if
        $b_0 \leftarrow b_0 + a_0$
        $b_1 \leftarrow b_1 + a_1$
        $b_2 \leftarrow b_2 + a_2$
    End for
    If $(b_0, b_1, b_2) = (f_0, f_1, f_2)$:
        Increment $count$ by 1.
    End if
End for
Return $count$.
End

4. Construction of the Proposed S-box $\mathcal{P}$

As established in Section 3, TI shares for monomials with a degree upper-bounded by 2 ensure uniformity. Accordingly, we construct our S-box using degree-2 polynomials. Note that, in order to ensure bijectivity (see Section 6.1.1), we must guarantee that the polynomials we employ are equivalent to sequences with a Hamming weight of $2^{5-1} = 16$.

Experimental observations reveal that 36,517 degree-2 polynomials in 5 variables correspond to sequences with a Hamming weight of 16, of which 25,536 satisfy the perfect SAC criterion. Given that enumerating all $\binom{25536}{5} \approx 2^{66.29}$ possible S-boxes is computationally infeasible on a standard PC, a random selection process is employed to choose 5 polynomials from the available set. As detailed in Section 6, this approach effectively yields S-boxes with superior security properties within a practical timeframe. From this set, the S-box $\mathcal{P}$ was selected. Table 1 presents $\mathcal{P}$, and its Algebraic Normal Form is provided in Eq. (5).

$$\begin{aligned}
\mathcal{P}_0 &= x_2 + x_4 + x_2x_1 + x_3x_0 + x_4x_0 + 1 \\
\mathcal{P}_1 &= x_1 + x_4 + x_2x_0 + x_2x_1 + x_4x_3 \\
\mathcal{P}_2 &= x_1 + x_3 + x_1x_0 + x_4x_2 + x_4x_3 \\
\mathcal{P}_3 &= x_0 + x_2 + x_3x_1 + x_3x_2 + x_4x_0 \\
\mathcal{P}_4 &= x_0 + x_3 + x_1x_0 + x_3x_2 + x_4x_1 + 1
\end{aligned} \tag{5}$$

The security of the S-box against various security parameters, is presented in **Error! Reference source not found.**.

Table 1. The S-box $\mathcal{P}$.

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Output	17	9	23	27	24	2	29	19	5	28	11	6	20	15	25	22

Input	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Output	18	3	4	1	31	12	10	13	0	16	30	26	21	7	8	14

Despite our best efforts, we have not yet found a definitive solution for constructing TI for monomials of degree greater than 2 that can preserve uniformity. However, the aim of this paper to generate S-boxes with an algebraic degree of 2 remains relevant, as this type of S-box is still widely used, as demonstrated by their application in ASCON [11], KECCAK [9], PRIMATEs [13], and Shamash [15].

5. TI Construction of $\mathcal{P}$

Since all $\mathcal{P}_r$ belong to the same degree-count class, we present the threshold implementation (TI) for only one representative polynomial in this paper. Without loss of generality, we select $\mathcal{P}_1$, whose corresponding polynomial is provided in Eq. (5).

For any degree-one monomial x_i, its TI shares can be uniformly distributed among the shares of $\mathcal{P}_1$ according to either Rule 1 or Rule 3. In this context, the symbol ← denotes the assignment of a monomial share to a share of $\mathcal{P}_1$. For demonstration purposes, consider x_1, one of the monomials in $\mathcal{P}_1$. Equations (6) and (7) illustrate the assignment of the shares of x_1 to those of $\mathcal{P}_1$ according to Rule 1 and Rule 3, respectively.

$$\begin{aligned} \mathcal{P}_{1,0} &\leftarrow x_{1,2} \\ \mathcal{P}_{1,1} &\leftarrow x_{1,0} \\ \mathcal{P}_{1,2} &\leftarrow x_{1,1} \end{aligned} \tag{6}$$

$$\begin{aligned} \mathcal{P}_{1,0} &\leftarrow x_{1,1} \\ \mathcal{P}_{1,1} &\leftarrow x_{1,2} \\ \mathcal{P}_{1,2} &\leftarrow x_{1,0} \end{aligned} \tag{7}$$

Both Rules 1 and 3 similarly apply to the degree-one monomial x_4 in $\mathcal{P}_1$.

For degree-2 monomials, such as $x_i x_j$, the shares can be distributed across the shares of a function $\mathcal{P}_1$ according to specific sharing rules to ensure correctness and security. In particular, the TI shares of a monomial like $x_2 x_0$ in $\mathcal{P}_1$ can be uniformly allocated using either Rules 2 or 4. The application of these rules is illustrated in Eqs. (8) and (9), which show the share distribution of the monomial $x_2 x_0$ in $\mathcal{P}_1$ following Rule 2 and Rule 4, respectively.

$$\begin{aligned} \mathcal{P}_{1,0} &\leftarrow x_{2,1}x_{0,2} + x_{2,2}x_{0,1} + x_{2,2}x_{0,2} \\ \mathcal{P}_{1,1} &\leftarrow x_{2,0}x_{0,0} + x_{2,0}x_{0,2} + x_{2,2}x_{0,0} \\ \mathcal{P}_{1,2} &\leftarrow x_{2,0}x_{0,1} + x_{2,1}x_{0,0} + x_{2,1}x_{0,1} \end{aligned} \tag{8}$$

$$\mathcal{P}_{1,0} \leftarrow x_{2,1}x_{0,1} + x_{2,1}x_{0,2} + x_{2,2}x_{0,1}$$

$$\begin{aligned}\mathcal{P}_{1,1} &\leftarrow x_{2,0}x_{0,2} + x_{2,2}x_{0,0} + x_{2,2}x_{0,2} \\ \mathcal{P}_{1,2} &\leftarrow x_{2,0}x_{0,0} + x_{2,0}x_{0,1} + x_{2,1}x_{0,0}\end{aligned} \tag{9}$$

Both Rules 2 and 4 likewise apply to the other degree-two monomials x_2x_0, x_2x_1, and x_4x_3 in $\mathcal{P}_1$.

When TI is applied to both degree-1 and degree-2 monomials in $\mathcal{P}_1$, one of the resulting TI constructions for $\mathcal{P}_1$ is presented in Eq. (10).

$$\begin{aligned}\mathcal{P}_{1,0} &= x_{1,2} + x_{4,2} + x_{2,1}x_{0,2} + x_{2,1}x_{1,2} + x_{2,2}x_{0,1} + x_{2,2}x_{0,2} \\ &\quad + x_{2,2}x_{1,1} + x_{2,2}x_{1,2} + x_{4,1}x_{3,2} + x_{4,2}x_{3,1} + x_{4,2}x_{3,2} \\ \mathcal{P}_{1,1} &= x_{1,0} + x_{4,0} + x_{2,0}x_{0,0} + x_{2,0}x_{0,2} + x_{2,0}x_{1,0} + x_{2,0}x_{1,2} \\ &\quad + x_{2,2}x_{0,0} + x_{2,2}x_{1,0} + x_{4,0}x_{3,0} + x_{4,0}x_{3,2} + x_{4,2}x_{3,0} \\ \mathcal{P}_{1,2} &= x_{1,1} + x_{4,1} + x_{2,0}x_{0,1} + x_{2,0}x_{1,1} + x_{2,1}x_{0,0} + x_{2,1}x_{0,1} \\ &\quad + x_{2,1}x_{1,0} + x_{2,1}x_{1,1} + x_{4,0}x_{3,1} + x_{4,1}x_{3,0} + x_{4,1}x_{3,1}\end{aligned} \tag{10}$$

Since $\mathcal{P}_1$ comprises 2 degree-1 monomials and 3 degree-2 monomials, with two implementation methods for each degree-1 monomial (Rules 1 and 3) and two alternatives for each degree-2 monomial (Rules 2 and 4), the total number of distinct TI constructions for $\mathcal{P}_1$ is $2^{2+3} = 32$.

The TI construction approach for $\mathcal{P}_1$ applies analogously to $\mathcal{P}_2$ and $\mathcal{P}_3$, as all reside within the same degree-count class. For $\mathcal{P}_0$ and $\mathcal{P}_4$, which include the constant 1, the construction necessitates only the addition of 1 to a single share within their respective sets, as outlined in Section 2.3. Equation (11) illustrates one of the valid TI constructions for $\mathcal{P}_0$, where the constant 1 is incorporated into $\mathcal{P}_{0,0}$.

$$\begin{aligned}\mathcal{P}_{0,0} &= x_{2,2} + x_{4,2} + x_{2,1}x_{1,2} + x_{2,2}x_{1,1} + x_{2,2}x_{1,2} + x_{3,1}x_{0,2} \\ &\quad + x_{3,2}x_{0,1} + x_{3,2}x_{0,2} + x_{4,1}x_{0,2} + x_{4,2}x_{0,1} + x_{4,2}x_{0,2} + 1 \\ \mathcal{P}_{0,1} &= x_{2,0} + x_{4,0} + x_{2,0}x_{1,0} + x_{2,0}x_{1,2} + x_{2,2}x_{1,0} + x_{3,0}x_{0,0} \\ &\quad + x_{3,0}x_{0,2} + x_{3,2}x_{0,0} + x_{4,0}x_{0,0} + x_{4,0}x_{0,2} + x_{4,2}x_{0,0} \\ \mathcal{P}_{0,2} &= x_{2,1} + x_{4,1} + x_{2,1}x_{1,2} + x_{2,2}x_{1,1} + x_{2,2}x_{1,2} + x_{3,1}x_{0,2} \\ &\quad + x_{3,2}x_{0,1} + x_{3,2}x_{0,2} + x_{4,1}x_{0,2} + x_{4,2}x_{0,1} + x_{4,2}x_{0,2}\end{aligned} \tag{11}$$

6. Results

In this section, we first review the security characteristics of the proposed S-box. Subsequently, we examine the TI of the S-box and its uniformity property.

Algorithm 3: Calculating nonlinearity of S-box $\mathcal{P}$

Input: S-box $\mathcal{P}$, output bit index $b \in \{0,1,2,3,4\}$ of $\mathcal{P}$.
Output: The nonlinearity values of all output bits of $\mathcal{P}$

```
Start
Initialize W_P(a) = 0.
Initialize max = 0.
For each 0 ≤ a < 5:
    For each x:
        If P(x) + ax = 1:
            Decrement W_P(a) by 1.
        Else if P(x) + ax = 0:
            Increment W_P(a) by 1.
```

```
        End if
    End for
    If max < W_P(a):
        max = W_P(a)
    End if
End for
Return 2^(n-1)( 1 − 2^(−n) · max)
End
```

6.1. Security properties

The assessment of a nonlinear component includes various essential aspects, such as bijectivity, nonlinearity, the strict avalanche criterion, the bit-independence criterion, differential uniformity, and linear approximation. These aspects have been thoroughly explored in numerous studies, including those by [19, 28-33].

We compare the security results of our S-box with other published 5-bit S-boxes, including those of ASCON [11] and KECCAK [9], as well as the S-boxes used in PRIMATEs [13], ICEPOLE [14], and Shamash [15]. Additionally, we evaluate our generated S-box against a recently proposed 5-bit S-box by Thakor et al. [34].

The method proposed by Kim et al. [19] can be used to generate S-boxes of various sizes, including 5-bit S-boxes. For a fair comparison with $\mathcal{P}$ and other published S-boxes discussed in this paper, we identify the most optimal S-box generated using Kim et al.'s method, denoted as $\mathcal{K}$, which is characterized by the highest achievable nonlinearity. **Table 2** presents the generated S-box $\mathcal{K}$.

Table 2. Input and output pairs of $\mathcal{K}$.

Input	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
Output	24	26	23	27	9	6	29	28	18	0	3	31	30	1	20	5

Input	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
Output	10	8	11	7	22	25	12	13	16	2	15	19	17	14	21	4

6.1.1. Bijectivity

An S-box is considered bijective when each input corresponds to a distinct output. This property ensures that no two different inputs produce the same output and that every possible output is generated by some input. The bijectivity of an S-box can be evaluated using Eq. (12) [35, 36]. For an n-bit S-box,

$$\#\{\boldsymbol{x}|Sn(\boldsymbol{x}) = \boldsymbol{y}\} = 1 \tag{12}$$

The bijectivity of $\mathcal{P}$ is evident from **Table 1** because there are neither one-to-many nor many-to-one mappings, and every input value corresponds to a unique output value with no input or output left unmapped.

6.1.2. Nonlinearity

The nonlinearity of a function quantifies its capability to withstand linear attacks [37]. The nonlinearity, NL, of a function f is defined by Eq. (13) [38, 39], where $W_f(\boldsymbol{a})$ denotes the Walsh Hadamard Transform of $f(\boldsymbol{x})$. This transform is expressed as $W_f(\boldsymbol{a}) = \sum_{\boldsymbol{x}\in\mathbb{F}_2^n}(-1)^{f(\boldsymbol{x})+\boldsymbol{a}\cdot\boldsymbol{x}}$ for all $\boldsymbol{a} \in \mathbb{F}_2^n$. The steps for computing it are shown in Algorithm 3.

$$NL(f) = 2^{n-1}\left(1 - 2^{-n}\, max\left|W_f(\boldsymbol{a})\right|\right) \tag{13}$$

Table 3 presents the nonlinearity of the five output bits of $\mathcal{P}$. The average nonlinearity comparison for each S-box is detailed in **Error! Reference source not found.**.

Table 3. Nonlinearity of $\mathcal{P}$.

$\boldsymbol{NL(\mathcal{P}_0)}$	12
$\boldsymbol{NL(\mathcal{P}_1)}$	12
$\boldsymbol{NL(\mathcal{P}_2)}$	12
$\boldsymbol{NL(\mathcal{P}_3)}$	12
$\boldsymbol{NL(\mathcal{P}_4)}$	12

6.1.3. Bit-independence criterion

The bit-independence criterion (BIC) [24, 31, 34, 40] requires that each input bit affects every output bit in such a way that the changes in the output bits remain independent of one another. The independence of these alterations can be assessed by measuring the nonlinearity [41] (see Section 6.1.2) and the SAC [42] (see Section 2) of $Sn_r + Sn_s$ for all $0 \leq r, s < n$ where $r \neq s$. Tables 4 and 5 present the BIC-nonlinearity and BIC-SAC of $\mathcal{P}$, respectively. BIC-SAC of $\mathcal{P}$ can be calculated using Algorithm 4, while the algorithm for calculating BIC-nonlinearity can be derived from Algorithm 3.

Table 4. BIC-nonlinearity of $\mathcal{P}$.

$\boldsymbol{S_{k+\ell}}$	$\boldsymbol{\ell = 0}$	$\boldsymbol{\ell = 1}$	$\boldsymbol{\ell = 2}$	$\boldsymbol{\ell = 3}$	$\boldsymbol{\ell = 4}$
$\boldsymbol{k = 0}$	-	12	12	12	8
$\boldsymbol{k = 1}$	12	-	12	12	12
$\boldsymbol{k = 2}$	12	12	-	12	8
$\boldsymbol{k = 3}$	12	12	12	-	12
$\boldsymbol{k = 4}$	8	12	8	12	-

Table 5. BIC-SAC of $\mathcal{P}$.

$\boldsymbol{S_{k+\ell}}$	$\boldsymbol{\ell = 0}$	$\boldsymbol{\ell = 1}$	$\boldsymbol{\ell = 2}$	$\boldsymbol{\ell = 3}$	$\boldsymbol{\ell = 4}$
$\boldsymbol{k = 0}$	-	0.5	0.5	0.5	0.5
$\boldsymbol{k = 1}$	0.5	-	0.5	0.5	0.5
$\boldsymbol{k = 2}$	0.5	0.5	-	0.5	0.5
$\boldsymbol{k = 3}$	0.5	0.5	0.5	-	0.5
$\boldsymbol{k = 4}$	0.5	0.5	0.5	0.5	-

Algorithm 4: Calculation of BIC-SAC of S-box $\mathcal{P}$

Input: S-box $\mathcal{P}$, two output bit index k and ℓ such that $k, \ell \in \{0,1,2,3,4\}$.
Output: BIC-SAC of $\mathcal{P}_k + \mathcal{P}_\ell$

1: **Start**
2: Initialise an empty array Q
3: **For each** $0 \leq m < 5$:
4: **For each** $0 \leq i < 2^5$:
5: $Q(i) \leftarrow \mathcal{P}(i) + \mathcal{P}(i + 2^m)$

```
        Initialise counter c = 0.
        For each 0 ≤ x < 2^5:
          If Q_k(x) + Q_ℓ(x):
              Increment c by 1
          End if
        End for
      End for
    End for
    Return c/2^5.
    End
```

6.1.4. Differential uniformity

Differential cryptanalysis [2] is a prominent cryptanalytic technique. The resilience of Sn to this attack is quantified by Eq. (14).

Differential cryptanalysis [2] is a prominent cryptanalytic attack. This method exploits the probabilistic propagation of input differences through non-linear components such as S-boxes using carefully chosen plaintext pairs. An adversary analyses corresponding output differences to identify high-probability difference propagation patterns. The resilience of Sn against this attack is quantified by Eq. (14).

$$D = max_{\Delta x \in \mathbb{F}_2^n, \Delta y \in \mathbb{F}_2^n} \left(\#\left(\Delta \boldsymbol{y} = Sn(\boldsymbol{x}) + Sn(\boldsymbol{x} + \Delta \boldsymbol{x})\right)\right) \quad (14)$$

where $\Delta \boldsymbol{x}, \Delta \boldsymbol{y} \neq 0$. A lower value of differential uniformity, D, indicates a stronger resistance of the S-box to differential cryptanalysis [32, 43, 44]. Moreover, D represents the maximum differential value in the Difference Distribution Table [45], whose entries can be calculated using Algorithm 5. The difference distribution tables of $\mathcal{P}$ are outlined in Appendix A. Differential uniformity should not be confused with the uniformity property of TI, as defined in Definition 3.

Algorithm 5: Calculating differential uniformity of S-box $\mathcal{P}$.

Input: S-box $\mathcal{P}$, input difference $\Delta \boldsymbol{x}$, output difference $\Delta \boldsymbol{y}$.
Output: Differential uniformity of S-box $\mathcal{P}$ with Δx and Δy.

```
Start
Initialize sum = 0.
For each 0 ≤ x < 2^5:
    If P(x) + P(x + Δx) = Δy:
        Increment sum by 1.
    End if
End for
Return sum
End
```

Algorithm 6: Calculating linear approximation of S-box $\mathcal{P}$.

Input: S-box $\mathcal{P}$, input mask Γ_x and output mask Γ_y.
Output: Linear approximation of S-box $\mathcal{P}$ with Γ_x and Γ_y.

```
Start
```

2: Initialize $sum = 0$.
3: **For each** $0 \leq \boldsymbol{x} < 2^5$:
4: **If** $\boldsymbol{x} \cdot \Gamma_{\boldsymbol{x}} = \mathcal{P}(\boldsymbol{x}) \cdot \Gamma_{\boldsymbol{y}}$:
5: Increment sum by 1.
6: **End if**
7: **End for**
8: **Return** $sum - 2^4$.
9: **End**

6.1.5. Linear probability

Linear cryptanalysis [1] is another prominent cryptanalytic attack. This method exploits statistical biases in linear approximations between subsets of plaintext, ciphertext, and key bits using known plaintext-ciphertext pairs. It specifically targets non-linear components such as S-boxes by identifying high-probability linear relations. The resilience of Sn against this attack is quantified by Eq. (15).

$$L = max_{\Gamma_{\boldsymbol{x}} \in \mathbb{F}_2^n, \Gamma_{\boldsymbol{y}} \in \mathbb{F}_2^n} \left| \frac{\#\left(\boldsymbol{y} \cdot \Gamma_{\boldsymbol{y}} = Sn(\boldsymbol{x} \cdot \Gamma_{\boldsymbol{x}})\right)}{2^n} - \frac{1}{2} \right| \tag{15}$$

where the linear masks $\Gamma_{\boldsymbol{x}}, \Gamma_{\boldsymbol{y}} \neq 0$. An S-box exhibits stronger resistance to linear cryptanalysis when it has a lower linear probability, L. Furthermore, L represents the maximum linear value in the Linear Approximation Table [28, 45], which can be computed using Algorithm 6. The linear approximation tables of $\mathcal{P}$ are shown in Appendix B.

6.1.6. Fixed points, reverse fixed points and short period ring

To construct a secure S-box, it is crucial to identify and eliminate the presence of fixed points, reverse fixed points, and short period rings [46-48]. A fixed point, as defined in [48, 49], occurs when an input value in an S-box maps directly to itself, as shown in Eq. (16). Conversely, a reverse fixed point is when the input of an n-bit S-box maps to its own bitwise complement, as illustrated in Eq. (17).

$$Sn(\boldsymbol{x}) = \boldsymbol{x} \tag{16}$$

$$Sn(\boldsymbol{x}) = 2^n - 1 - \boldsymbol{x} \tag{17}$$

The presence of fixed points within an S-box can potentially expose secret data to attackers through intercepted ciphertext [50]. Therefore, it is essential to ensure that the finalized S-box is free from any fixed points [51]. Additionally, a robust S-box should avoid short iterative periods [48, 49, 52, 53], a condition where $f^m(\boldsymbol{x}) = \boldsymbol{x}$ holds for a small value of m. **Error! Reference source not found.** presents the number of fixed points and reverse fixed points, as well as the iterative periods of all S-boxes analysed in this paper.

6.1.7. SAC comparison with other S-boxes

The S-boxes of ASCON and KECCAK exhibit SAC values of either an absolute 0 or 1. In contrast, the S-boxes of PRIMATEs and Shamash show improved SAC conditions, though with an absolute SAC value of 1. Thakor's S-box does not exhibit absolute SAC values of 0 or 1, instead presenting a minimum SAC value of

0.25 and a maximum of 0.75. The S-box of ICEPOLE displays a broader range of SAC values, with a minimum of 0.125 and a maximum of 0.875. Both $\mathcal{K}$ and $\mathcal{P}$ do not exhibit SAC values of absolute 0 or 1, but they demonstrate a consistent set of values at 0.5. **Error! Reference source not found.** presents the SAC values for five output bits of $\mathcal{P}$. As illustrated in **Error! Reference source not found.**, $\mathcal{P}$ adheres strictly to a perfect SAC value of 0.5, distinguishing it from other S-boxes.

Table 6. SAC of $\mathcal{P}$.

	$\ell = 0$	$\ell = 1$	$\ell = 2$	$\ell = 3$	$\ell = 4$
$k = 0$	-	0.5	0.5	0.5	0.5
$k = 1$	0.5	-	0.5	0.5	0.5
$k = 2$	0.5	0.5	-	0.5	0.5
$k = 3$	0.5	0.5	0.5	-	0.5
$k = 4$	0.5	0.5	0.5	0.5	-

6.2. Threshold implementation

The resultant TI of the S-box indicates that every input and output combination has 256 counts. **Error! Reference source not found.** presents the threshold count of $\mathcal{P}_0$'s shares. The threshold counts for the shares of $\mathcal{P}_1$, $\mathcal{P}_2$, $\mathcal{P}_3$, and $\mathcal{P}_4$ are shown in Appendix C.

Table 7. Threshold count of $\mathcal{P}_0$'s shares.

	$(\mathcal{P}_{0,0}, \mathcal{P}_{0,1}, \mathcal{P}_{0,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	000	001	010	011	100	101	110	111
00000	256	0	0	256	0	256	256	0
00001	256	0	0	256	0	256	256	0
00010	256	0	0	256	0	256	256	0
00011	256	0	0	256	0	256	256	0
00100	256	0	0	256	0	256	256	0
00101	0	256	256	0	256	0	0	256
00110	0	256	256	0	256	0	0	256
00111	256	0	0	256	0	256	256	0
01000	0	256	256	0	256	0	0	256
01001	0	256	256	0	256	0	0	256
01010	0	256	256	0	256	0	0	256
01011	0	256	256	0	256	0	0	256
01100	0	256	256	0	256	0	0	256
01101	256	0	0	256	0	256	256	0
01110	256	0	0	256	0	256	256	0
01111	0	256	256	0	256	0	0	256
10000	256	0	0	256	0	256	256	0
10001	256	0	0	256	0	256	256	0
10010	0	256	256	0	256	0	0	256
10011	0	256	256	0	256	0	0	256
10100	256	0	0	256	0	256	256	0
10101	0	256	256	0	256	0	0	256
10110	256	0	0	256	0	256	256	0
10111	0	256	256	0	256	0	0	256
11000	256	0	0	256	0	256	256	0
11001	256	0	0	256	0	256	256	0
11010	0	256	256	0	256	0	0	256
11011	0	256	256	0	256	0	0	256
11100	256	0	0	256	0	256	256	0

11101	0	256	256	0	256	0	0	256
11110	256	0	0	256	0	256	256	0
11111	0	256	256	0	256	0	0	256

Table 8. Comparison of S-boxes.

		Ascon	Keccak	PRIMATEs	ICEPOLE	Shamash	Thakor's S-box	$\mathcal{K}$	Our work ($\mathcal{P}$)
Nonlinearity	**Lowest**	8	8	12	8	12	8	8	**12**
	Highest	8	8	12	8	12	10	12	**12**
	Average	8	8	12	8	12	8.4	8.8	**12**
SAC	**Lowest**	0	0	0.5	0.125	0.5	0.25	0.5	**0.5**
	Highest	1	1	1	0.875	1	0.75	0.5	**0.5**
	Average	0.62	0.4	0.54	0.425	0.6	0.54	0.5	**0.5**
BIC-Nonlinearity	**Lowest**	8	8	12	8	12	8	8	**12**
	Highest	12	12	12	12	12	10	12	**12**
	Average	11.2	10	12	10	12	9	9.2	**12**
BIC-SAC	**Lowest**	0.3	0.5	0.5	0.5	0.5	0.45	0.4	**0.5**
	Highest	0.6	0.6	0.6	0.6	0.5	0.575	0.6	**0.6**
	Average	0.52	0.55	0.51	0.55	0.5	0.5075	0.49	**0.55**
Algebraic degree		2	2	2	4	2	4	2	**2**
Differential uniformity, *D*		8	8	2	8	2	8	32	**2**
Linear probability, *L*		0.25	0.25	0.125	0.25	0.125	0.25	0.5	**0.125**
Number of fixed points		0	2	0	0	1	0	0	**0**
Number of inverse fixed points		0	0	2	2	0	1	0	**0**
Shortest iterative period		6	1	2	2	1	4	16	**32**
References		[11]	[9]	[13]	[14]	[15]	[34]	[19]	**This paper**

7. Application on other S-boxes

Besides $\mathcal{P}$, our TI method can also be employed to implement TI for other S-boxes with algebraic degree 2, since Rule 1, Rule 2, Rule 3 and Rule 4 are general to other S-boxes of degree 2.

8. Conclusions

We have developed a TI method capable of preserving uniformity by employing polynomials from the same degree-count class across all output bits of the S-box. This innovative approach not only ensures the uniformity of S-boxes but also provides a versatile framework that can be extended to the construction of TI for other S-boxes while upholding the principle of uniformity. The broader applicability of this method signifies a substantial advancement in the design and analysis of S-boxes within cryptographic systems, contributing to their robustness and security. This methodology is poised to enhance the reliability of cryptographic implementations, offering a uniform approach resilient to various cryptanalytic attacks. Consequently, our work lays the groundwork for future research and development in the field of secure S-box constructions, paving the way for more effective and secure cryptographic solutions.

8.1. Open problem

Several studies have proposed implementing TI on component S-boxes obtained through the decomposition of the main S-box. Although TI with decomposition is a widely adopted technique [20, 22, 54-56], it does not inherently guarantee uniformity. Extending our method to decomposed S-boxes remains an open problem.

Furthermore, our method can preserve uniformity only when applied to S-boxes with an algebraic degree bounded by 2. Extending this method to S-boxes with an algebraic degree greater than 2 presents an interesting avenue for future research.

Acknowledgement

This research was supported by Malaysia Cryptology Technology and Management Centre, CyberSecurity Malaysia, and Fundamental Research Grant Scheme (FRGS) of Universiti Teknikal Malaysia Melaka (FRGS/1/2022/FTMK/F00526) funded by the Ministry of Higher Education, Malaysia.

Nomenclatures

$\mathbb{A}_n$	A set of all polynomials of degree n.
$\mathbb{F}_p$	Finite field with p elements.
$\mathbb{F}_p^n$	n-dimensional vector space over $\mathbb{F}_p$.
W_f	Walsh Hadamard Transform of f

Abbreviations

AES	Advanced Encryption Standard
BIC	Bit-independence criterion
DPA	Differential power analysis
GCM	Galois/Counter mode
NIST	National Institute of Standards and Technology
SAC	Strict avalanche criterion
SHA	Secure Hash Algorithm
TI	Threshold implementation
WPA	Wi-Fi Protected Access

References

1. Matsui, M. (1994). *Linear cryptanalysis method for DES cipher*. In Helleseth T. (Ed.), *Advances in Cryptology-Eurocrypt*'93. Springer Berlin Heidelberg, 386-397.
2. Biham, E.; and Shamir, A. (1991). Differential cryptanalysis of DES-like cryptosystems. *Journal of Cryptology*, 4, 3-72.
3. Vaughn, R.; and Borowczak, M. (2024). Strict avalanche criterion of SHA-256 and sub-function-removed variants. *Cryptography*, 8(3), 40.
4. Zhu, B.; Jiang, X.; Huang, K.; and Yu, M. (2024). A response-feedback-based strong PUF with improved strict avalanche criterion and reliability. *Sensors*, 24(1), 93.
5. Siddhanti, A.A.; Bodapati, S.; Chattopadhyay, A.; Maitra, S.; Roy, D.; and Stănică, P. (2019). *Analysis of the strict avalanche criterion in variants of arbiter-based physically unclonable functions*. In Hao, F.; Ruj, S.; Sen Gupta, S. (Eds.), *Progress in Cryptology-INDOCRYPT* 2019. Springer International Publishing, 556-577.
6. Patil, P.; Karoshi, A.; Marje, A.; and Desai, V. (2023). *Enhancing S-box nonlinearity in AES for improved security using key-dependent dynamic S-box*.

In Bindhu, V.; Tavares, J.M.R.S.; and Vuppalapati, C. (Eds.), *Proceedings of the fourth international conference on communication*, *computing and electronics systems*. Springer Nature Singapore, 91-102.

7. Mahboob, A.; Nadeem, M.; and Rasheed, M.W. (2023). A study of text-theoretical approach to S-box construction with image encryption applications. *Scientific Reports*, 13(1), 21081.
8. Nikova, S.; Rechberger, C.; and Rijmen, V. (2006). *Threshold implementations against side-channel attacks and glitches*. In Ning, P.; Qing, S.; and Li, N. (Eds.), *International Conference on Information and Communications Security*. Springer Berlin Heidelberg, 529-545.
9. Bertoni, G.; Daemen, J.; Peeters, M.; and Van A., G. (2013). Keccak. *Advances in Cryptology-Eurocrypt* 2013, 313-314.
10. National Institute of Standards and Technology (2012). NIST selects winner of secure hash algorithm (SHA-3) Competition. Retrieved July 10, 2025, from https://www.nist.gov/news-events/news/2012/10/nist-selects-winner-secure-hash-algorithm-sha-3-competition.
11. Dobraunig, C.; Eichlseder, M.; Mendel, F.; and Schläffer, M. (2021). Ascon v1.2: Lightweight authenticated encryption and hashing. *Journal of Cryptology*, 34(3), 33.
12. National Institute of Standards and Technology (2023). Lightweight cryptography standardization process: NIST selects Ascon. Retrieved July 10, 2025, from https://csrc.nist.gov/News/2023/lightweight-cryptography-nist-selects-ascon.
13. Andreeva, E.; Bilgin, B.; Bogdanov, A.; Luykx, A.; Mendel, F.; Mennink, B.; Mouha, N.; Wang, Q.; and Yasuda, K. (2014). PRIMATEs v1: Submission to the CAESAR competition. Retrieved July 10, 2025, from https://competitions.cr.yp.to/round1/primatesv1.pdf.
14. Morawiecki, P.; Gaj, K.; Homsirikamol, E.; Matusiewicz, K.; Pieprzyk, J.; Rogawski, M.; Srebrny, M.; and Wójcik, M. (2014). *ICEPOLE*: *High-speed*, *hardware-oriented authenticated encryption*. In Salinesi, C.; Norrie, M.C.; and Pastor, Ó. (Eds.), *Advanced Information Systems Engineering*. Springer Berlin Heidelberg, 392-413.
15. Penazzi, D.; and Montes, M. (2019). Shamash (and Shamashash) (version 1). Retrieved July 10, 2025, from https://csrc.nist.gov/CSRC/media/Projects/Lightweight-Cryptography/documents/round-1/spec-doc/ShamashAndShamashash-spec.pdf
16. Bartoli, D.; and Timpanella, M. (2022). On a conjecture on APN permutations. *Cryptography and Communications*, 14(4), 925-931.
17. Gérard, B.; Grosso, V.; Naya-Plasencia, M.; and Standaert, F.-X. (2013). *Block ciphers that are easier to mask*: *How far can we go*? In Bertoni, G.; and Coron, J.-S. (Eds.), *Cryptographic Hardware and Embedded Systems-CHES* 2013. Springer Berlin Heidelberg, 383-399.
18. Canteaut, A.; Duval, S.; and Perrin, L. (2017). A generalisation of Dillon's APN permutation with the best known differential and nonlinear properties for all fields of size 2^{4k+2}. *IEEE Transactions on Information Theory*, 63(11), 7575-7591.

19. Kim, K.; Matsumoto, T.; and Imai, H. (1990). *A recursive construction method of S-boxes satisfying strict avalanche criterion*. In Menezes, A.J.; and Vanstone, S.A. (Eds.), *Advances in Cryptology-Crypto*'90. Springer Berlin Heidelberg, 564-574.
20. Baksi, A.; Guilley, S.; Shrivastwa, R.-r.; and Takarabt, S. (2024). *From substitution box to threshold*. In Chattopadhyay, A.; Bhasin, S.; Picek, S.; and Rebeiro, C. (Eds.), *Progress in Cryptology-INDOCRYPT* 2023. Springer Nature Switzerland, 48-67.
21. Božilov, D.; Knežević, M.; and Nikov, V. (2022). Optimized threshold implementations: Securing cryptographic accelerators for low-energy and low-latency applications. *Journal of Cryptographic Engineering*, 12(1), 15-51.
22. Bilgin, B.; Nikova, S.; Nikov, V.; Rijmen, V.; Tokareva, N.; and Vitkup, V. (2015). Threshold implementations of small S-boxes. *Cryptography and Communications*, 7(1), 3-33.
23. Yao, F.; Chen, H.; Wei, Y.; Pasalic, E.; Zhou, F.; and Fan, L. (2024). Optimizing AES threshold implementation under the glitch-extended probing model. *IEEE Transactions on Computer-Aided Design of Integrated Circuits and Systems*, 43(7), 1984-1997.
24. Webster, A.F.; and Tavares, S.E. (1985). *On the design of S-boxes*. In Williams, H.C. (Ed.), *Advances in Cryptology-CRYPTO'*85. Springer Berlin Heidelberg, 523-534.
25. Forrié, R. (1990). *The strict avalanche criterion*: *Spectral properties of boolean functions and an extended definition*. In Goldwasser, S. (Ed.), *Advances in Cryptology-CRYPTO*'88. Springer-Verlag Berlin Heidelberg, 450-468.
26. Topanto, D.; and Alamsyah, A. (2022). Security improvement of AES algorithm using S-box modification based on strict avalanche criterion on image encryption. *Journal of Soft Computing Exploration*, 3(1), 55-61.
27. Mahboob, A.; Asif, M.; Siddique, I.; Saleem, A.; Nadeem, M.; Grzelczyk, D.; and Awrejcewicz, J. (2022). A novel construction of substitution box based on polynomial mapped and finite field with image encryption application. *IEEE Access*, 10, 119244-119258.
28. Lu, Z.; Mesnager, S.; Cui, T.; Fan, Y.; and Wang, M. (2022). An STP-based model toward designing S-boxes with good cryptographic properties. *Designs, Codes and Cryptography*, 90(5), 1179-1202.
29. Carlet, C.; Djurasevic, M.; Jakobovic, D.; and Picek, S. (2020). *A search for additional structure*: *The case of cryptographic S-boxes*. In Bäck, T.; Preuss, M.; Deutz, A.; Wang, H.; Doerr, C.; Emmerich, M.; and Trautmann, H. (Eds.), *Parallel Problem Solving from Nature-PPSN XVI*. Springer International Publishing, 343-356.
30. Seberry, J.; Zhang, X.-M.; and Zheng, Y. (1994). Improving the strict avalanche characteristics of cryptographic functions. *Information Processing Letters*, 50(1), 37-41.
31. Madarro-Capó, E.J.; Legón-Pérez, C.M.; Rojas, O.; Sosa-Gómez, G.; and Socorro-Llanes, R. (2020). Bit independence criterion extended to stream ciphers. *Applied Sciences*, 10(21), 7668.

32. Durasevic, M.; Jakobovic, D.; Mariot, L.; Mesnager, S.; and Picek, S. (2023). *On the evolution of boomerang uniformity in cryptographic S-boxes*. In Correia, J.; Smith, S.; and Qaddoura, R. (Eds.), *Applications of Evolutionary Computation*. Springer Nature Switzerland, 237-252.
33. Saarinen, M.-J. (2011). *Cryptographic analysis of all* 4 × 4-*bit S-boxes*. In Miri, A.; and Vaudenay, S. (Eds.), *Selected Areas in Cryptography*. Springer Berlin Heidelberg, 118-133.
34. Thakor, V.A.; Razzaque, M.A.; Darji, A.D.; and Patel, A.R. (2023). A novel 5-bit S-box design for lightweight cryptography algorithms. *Journal of Information Security and Applications*, 73, 103444.
35. Ibrahim, S.; and Abbas, A.M. (2020). A novel optimization method for constructing cryptographically strong dynamic S-boxes. *IEEE Access*, 8, 225004-225017.
36. Wang, Y.; Zhang, Z.; Zhang, L.Y.; Feng, J.; Gao, J.; and Lei, P. (2020). A genetic algorithm for constructing bijective substitution boxes with high nonlinearity. *Information Sciences*, 523, 152-166.
37. Liu, J.; Mesnager, S.; and Chen, L. (2017). On the nonlinearity of S-boxes and linear codes. *Cryptography and Communications*, 9(3), 345-361.
38. Cusick, T.W.; and Stănică, P. (2017). *Cryptographic Boolean functions and applications*. Academic Press.
39. Mahboob, A.; Asif, M.; Nadeem, M.; Saleem, A.; Eldin, S.M.; and Siddique, I. (2022). A cryptographic scheme for construction of substitution boxes using quantic fractional transformation. *IEEE Access*, 10, 132908-132916.
40. Levinskas, M.; and Mihalkovich, A. (2021). Avalanche effect and bit independence criterion of perfectly secure Shannon cipher based on matrix power. *Mathematical Models in Engineering*, 7(3), 50-53.
41. Ahmad, M.; Alkanhel, R.; El-Shafai, W.; Algarni, A.D.; El-Samie, F.E.A.; and Soliman, N.F. (2022). Multi-objective evolution of strong S-Boxes using Non-Dominated Sorting Genetic Algorithm-II and chaos for secure telemedicine. *IEEE Access*, 10, 112757-112775.
42. Artuğer, F. (2023). A new S-box generator algorithm based on 3D chaotic maps and whale optimization algorithm. *Wireless Personal Communications*, 131(2), 835-853.
43. Tian, S.; Boura, C.; and Perrin, L. (2020). Boomerang uniformity of popular S-box constructions. *Designs*, *Codes and Cryptography*, 88(9), 1959-1989.
44. Mesnager, S.; Tang, C.; and Xiong, M. (2020). On the boomerang uniformity of quadratic permutations. *Designs*, *Codes and Cryptography*, 88(10), 2233-2246.
45. Kim, G.; Kim, H.; Heo, Y.; Jeon, Y.; and Kim, J. (2021). Generating cryptographic S-boxes using the reinforcement learning. *IEEE Access*, 9, 83092-83104.
46. Aboytes-González, J.A.; Soubervielle-Montalvo, C.; Campos-Cantón, I.; Perez-Cham, O.E.; and Ramírez-Torres, M.T. (2023). Method to improve the cryptographic properties of S-boxes. *IEEE Access*, 11, 99546-99557.
47. Si, Y.; Liu, H.; and Chen, Y. (2021). Constructing keyed strong S-box using an enhanced quadratic map. *International Journal of Bifurcation and Chaos*, 31(10), 2150146.

48. Wang, M.; Liu, H.; and Zhao, M. (2023). Construction of a non-degeneracy 3D chaotic map and application to image encryption with keyed S-box. *Multimedia Tools and Applications*, 82(22), 34541-34563.
49. Liu, H.; Liu, J.; and Ma, C. (2023). Constructing dynamic strong S-box using 3D chaotic map and application to image encryption. *Multimedia Tools and Applications*, 82(16), 23899-23914.
50. Manzoor, A.; Zahid, A.H.; and Hassan, M.T. (2022). A new dynamic substitution box for data security using an innovative chaotic map. *IEEE Access*, 10, 74164-74174.
51. Zhang, Y.-Q.; Hao, J.-L.; and Wang, X.-Y. (2020). An efficient image encryption scheme based on S-boxes and fractional-order differential logistic map. *IEEE Access*, 8, 54175-54188.
52. Cui, J.; Huang, L.; Zhong, H.; Chang, C.; and Yang, W. (2011). An improved AES S-box and its performance analysis. *International Journal of Innovative Computing, Information and Control*, 7(5), 2291-2302.
53. Nitaj, A.; Susilo, W.; and Tonien, J. (2020). *A new improved AES S-box with enhanced properties*. In Liu, J.K.; and Cui, H. (Eds.), *Information Security and Privacy*. Springer International Publishing, 125-141.
54. Liu, B.; Gong, Z.; and Qiu, W. (2017). Automatic search of threshold implementations of 4-bit S-boxes resisting DPA. *Chinese Journal of Electronics*, 26(1), 93-100.
55. Piccione, E.; Andreoli, S.; Budaghyan, L.; Carlet, C.; Dhooghe, S.; Nikova, S.; Petrides, G.; and Rijmen, V. (2023). An optimal universal construction for the threshold implementation of bijective S-boxes. *IEEE Transactions on Information Theory*, 69(10), 6700-6710.
56. Liu, B.; and Tang, M. (2024). MS-LW-TI: Primitive-based first-order threshold implementation for 4×4 S-boxes. *IET Information Security*, 12024(1), 8851878.

Appendix A

Differential distribution tables

Table A-1. DDT Table (Part 1).

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	32	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
2	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2	0
3	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
4	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
5	0	0	0	0	2	2	2	2	2	2	2	2	0	0	0	0
6	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
7	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
8	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0	0
9	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0	0
10	0	2	2	0	2	0	0	2	0	2	2	0	2	0	0	2
11	0	2	2	0	2	0	0	2	2	0	0	2	0	2	2	0
12	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
13	0	2	2	0	2	0	0	2	0	2	2	0	2	0	0	2
14	0	2	0	2	2	0	2	0	2	0	2	0	0	2	0	2
15	0	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2
16	0	2	0	2	0	2	0	2	2	0	2	0	2	0	2	0
17	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
18	0	2	2	0	0	2	2	0	2	0	0	2	2	0	0	2
19	0	0	2	2	0	0	2	2	2	2	0	0	2	2	0	0
20	0	2	0	2	0	2	0	2	2	0	2	0	2	0	2	0
21	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
22	0	2	0	2	2	0	2	0	2	0	2	0	0	2	0	2
23	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
24	0	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2
25	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2	0
26	0	2	2	0	0	2	2	0	2	0	0	2	2	0	0	2
27	0	0	0	0	2	2	2	2	2	2	2	2	0	0	0	0
28	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
29	0	0	2	2	0	0	2	2	2	2	0	0	2	2	0	0
30	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
31	0	2	2	0	2	0	0	2	2	0	0	2	0	2	2	0

Table A-2. DDT Table (Part 2).

	16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	2	2	2	2	0	0	0	0	2	2	2	2	0	0	0	0
2	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2	0
3	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2	2
4	2	2	2	2	2	2	2	2	0	0	0	0	0	0	0	0
5	2	2	2	2	0	0	0	0	0	0	0	0	2	2	2	2
6	0	0	0	0	0	0	0	0	2	2	2	2	2	2	2	2
7	0	0	2	2	2	2	0	0	2	2	0	0	0	0	2	2
8	0	0	2	2	2	2	0	0	0	0	2	2	2	2	0	0
9	2	2	0	0	0	0	2	2	2	2	0	0	0	0	2	2
10	0	2	2	0	2	0	0	2	0	2	2	0	2	0	0	2
11	0	2	2	0	2	0	0	2	2	0	0	2	0	2	2	0
12	0	0	0	0	2	2	2	2	0	0	0	0	2	2	2	2
13	2	0	0	2	0	2	2	0	2	0	0	2	0	2	2	0
14	0	2	0	2	2	0	2	0	2	0	2	0	0	2	0	2
15	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2	0
16	0	2	0	2	0	2	0	2	2	0	2	0	2	0	2	0

Table A-2 (Continue). DDT Table (Part 2).

	16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
17	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2	2
18	0	2	2	0	0	2	2	0	2	0	0	2	2	0	0	2
19	2	2	0	0	2	2	0	0	0	0	2	2	0	0	2	2
20	2	0	2	0	2	0	2	0	0	2	0	2	0	2	0	2
21	2	2	0	0	0	0	2	2	0	0	2	2	2	2	0	0
22	2	0	2	0	0	2	0	2	0	2	0	2	2	0	2	0
23	0	2	0	2	2	0	2	0	0	2	0	2	2	0	2	0
24	0	2	0	2	0	2	0	2	0	2	0	2	0	2	0	2
25	2	0	0	2	2	0	0	2	2	0	0	2	2	0	0	2
26	2	0	0	2	2	0	0	2	0	2	2	0	0	2	2	0
27	0	0	0	0	2	2	2	2	2	2	2	2	0	0	0	0
28	2	0	2	0	0	2	0	2	2	0	2	0	0	2	0	2
29	0	0	2	2	0	0	2	2	2	2	0	0	2	2	0	0
30	2	2	0	0	2	2	0	0	2	2	0	0	2	2	0	0
31	2	0	0	2	0	2	2	0	0	2	2	0	2	0	0	2

Appendix B

Linear approximation tables

Table B-1. LAT Table (Part 1).

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**10**	**11**	**12**	**13**	**14**	**15**
0	16	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	0	0	4	0	0	4	0	0	0	0	4	0	-4	0	-4	-4
2	0	0	4	4	0	4	0	-4	4	0	0	-4	4	4	-4	4
3	0	0	0	-4	0	0	0	4	4	0	4	-4	0	-4	0	0
4	0	0	0	0	4	-4	0	0	4	0	4	0	0	4	4	0
5	0	0	-4	0	-4	0	0	0	4	0	0	0	4	4	0	-4
6	0	0	4	-4	4	0	0	4	0	0	-4	4	4	0	0	4
7	0	0	0	-4	4	4	0	-4	0	0	0	-4	0	0	4	0
8	0	-4	0	0	4	0	-4	-4	0	0	-4	0	4	-4	0	-4
9	0	4	-4	0	-4	4	4	-4	0	0	0	0	0	-4	4	0
10	0	4	-4	-4	4	-4	4	0	-4	0	-4	-4	0	0	-4	0
11	0	-4	0	4	4	0	4	0	-4	0	0	-4	4	0	0	-4
12	0	-4	0	0	0	-4	4	-4	4	0	0	0	-4	0	-4	-4
13	0	4	4	0	0	0	-4	-4	4	0	-4	0	0	0	0	0
14	0	-4	-4	-4	0	0	-4	0	0	0	0	4	0	-4	0	0
15	0	4	0	-4	0	-4	-4	0	0	0	4	-4	4	4	4	-4
16	0	-4	0	0	0	-4	0	0	0	4	-4	-4	-4	0	0	0
17	0	-4	4	0	0	0	0	0	0	-4	0	4	0	0	4	-4
18	0	4	4	-4	0	0	0	-4	-4	-4	4	0	0	-4	-4	-4
19	0	4	0	4	0	-4	0	4	4	-4	0	0	-4	-4	0	0
20	0	-4	0	0	-4	0	0	0	-4	-4	0	-4	-4	4	4	0
21	0	-4	-4	0	4	4	0	0	4	-4	4	-4	0	-4	0	4
22	0	-4	4	-4	-4	-4	0	-4	0	-4	0	0	0	0	0	-4
23	0	-4	0	-4	-4	0	0	4	0	4	4	0	4	0	-4	0
24	0	0	0	0	4	4	4	4	0	-4	0	4	0	4	0	-4
25	0	0	4	0	-4	0	4	4	0	-4	-4	-4	4	-4	4	0
26	0	0	4	-4	4	0	4	0	4	4	0	0	-4	0	4	0
27	0	0	0	4	4	-4	-4	0	-4	-4	4	0	0	0	0	4
28	0	0	0	0	0	0	4	-4	-4	4	4	4	0	0	4	4
29	0	0	-4	0	0	-4	4	-4	4	-4	0	4	4	0	0	0
30	0	0	4	4	0	-4	4	0	0	4	4	0	4	-4	0	0
31	0	0	0	4	0	0	-4	0	0	4	0	0	0	-4	4	-4

Table B-2. LAT Table (Part 2).

	16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0
1	-4	4	0	4	4	0	-4	4	0	0	4	0	-4	0	4	4
2	0	-4	-4	0	0	0	0	0	0	0	-4	4	0	4	0	4
3	-4	0	4	-4	-4	0	4	4	0	0	0	4	4	4	4	0
4	0	-4	4	0	0	-4	0	4	4	4	0	0	-4	-4	-4	4
5	4	0	4	4	-4	4	-4	0	-4	4	4	0	0	4	0	0
6	0	0	0	0	0	4	0	4	-4	-4	4	4	-4	0	-4	0
7	4	4	0	4	-4	4	4	0	4	-4	0	-4	0	0	0	4
8	0	4	4	-4	0	-4	-4	-4	0	0	0	4	0	0	0	4
9	-4	0	-4	0	-4	-4	0	0	0	0	4	4	-4	0	-4	0
10	0	0	0	4	0	-4	-4	4	0	0	-4	0	0	4	0	0
11	-4	-4	0	0	-4	4	0	0	0	0	0	0	-4	-4	4	-4
12	0	0	0	4	0	0	4	0	-4	-4	0	4	4	-4	-4	0
13	-4	4	0	0	-4	0	0	4	-4	4	-4	-4	0	-4	0	-4
14	0	-4	-4	4	0	0	4	0	-4	4	-4	0	-4	0	4	4
15	-4	0	-4	0	4	0	0	-4	-4	-4	0	0	0	0	0	0
16	-4	0	-4	-4	0	4	0	0	0	4	4	-4	0	4	-4	4

Table B-2 (Continue). LAT Table (Part 2).

	16	**17**	**18**	**19**	**20**	**21**	**22**	**23**	**24**	**25**	**26**	**27**	**28**	**29**	**30**	**31**
17	0	-4	-4	0	-4	-4	-4	4	0	-4	0	-4	4	4	0	0
18	4	-4	0	-4	0	4	0	0	0	4	0	0	0	0	-4	0
19	0	0	0	0	-4	4	-4	-4	0	-4	-4	0	-4	0	0	5
20	4	4	0	-4	0	0	0	4	-4	0	-4	4	-4	0	0	0
21	0	0	0	0	4	0	-4	0	-4	0	0	-4	0	0	-4	-4
22	-4	0	4	4	0	0	0	-4	4	0	0	0	-4	4	0	4
23	0	4	-4	0	-4	0	-4	0	4	0	-4	0	0	-4	-4	0
24	-4	4	0	0	0	0	4	-4	0	4	-4	0	0	4	-4	0
25	0	0	0	4	4	0	0	0	0	4	0	0	4	-4	0	4
26	4	0	-4	0	0	0	-4	-4	0	4	0	4	0	0	4	-4
27	0	4	-4	4	-4	0	0	0	0	4	4	4	4	0	0	0
28	-4	0	4	0	0	4	-4	0	-4	0	-4	0	4	0	0	4
29	0	4	-4	-4	4	4	0	4	4	0	0	0	0	0	4	0
30	4	4	0	0	0	-4	4	0	-4	0	0	-4	-4	4	0	0
31	0	0	0	4	4	4	0	4	4	0	-4	4	0	4	-4	-4

Appendix C

Output count of Threshold

Table C-1. Threshold counts of $\mathcal{P}_1$'s shares.

	$(\mathcal{P}_{1,0}, \mathcal{P}_{1,1}, \mathcal{P}_{1,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
00000	256	0	0	256	0	256	256	0
00001	256	0	0	256	0	256	256	0
00010	256	0	0	256	0	256	256	0
00011	0	256	256	0	256	0	0	256
00100	256	0	0	256	0	256	256	0
00101	256	0	0	256	0	256	256	0
00110	256	0	0	256	0	256	256	0
00111	0	256	256	0	256	0	0	256
01000	256	0	0	256	0	256	256	0
01001	256	0	0	256	0	256	256	0
01010	256	0	0	256	0	256	256	0
01011	0	256	256	0	256	0	0	256
01100	0	256	256	0	256	0	0	256
01101	0	256	256	0	256	0	0	256
01110	0	256	256	0	256	0	0	256
01111	256	0	0	256	0	256	256	0
10000	0	256	256	0	256	0	0	256
10001	256	0	0	256	0	256	256	0
10010	0	256	256	0	256	0	0	256
10011	0	256	256	0	256	0	0	256
10100	256	0	0	256	0	256	256	0
10101	0	256	256	0	256	0	0	256
10110	256	0	0	256	0	256	256	0
10111	256	0	0	256	0	256	256	0
11000	0	256	256	0	256	0	0	256
11001	256	0	0	256	0	256	256	0
11010	0	256	256	0	256	0	0	256
11011	0	256	256	0	256	0	0	256
11100	0	256	256	0	256	0	0	256
11101	256	0	0	256	0	256	256	0
11110	0	256	256	0	256	0	0	256
11111	0	256	256	0	256	0	0	256

Table C-2. Threshold counts of $\mathcal{P}_2$'s shares.

	$(\mathcal{P}_{2,0}, \mathcal{P}_{2,1}, \mathcal{P}_{2,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
00000	256	0	0	256	0	256	256	0
00001	256	0	0	256	0	256	256	0
00010	256	0	0	256	0	256	256	0
00011	256	0	0	256	0	256	256	0
00100	256	0	0	256	0	256	256	0
00101	0	256	256	0	256	0	0	256

Table C-2 (Continue). Threshold counts of $\mathcal{P}_2$'s shares.

	$(\mathcal{P}_{2,0}, \mathcal{P}_{2,1}, \mathcal{P}_{2,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
00110	0	256	256	0	256	0	0	256
00111	256	0	0	256	0	256	256	0
01000	0	256	256	0	256	0	0	256
01001	0	256	256	0	256	0	0	256
01010	0	256	256	0	256	0	0	256
01011	0	256	256	0	256	0	0	256
01100	0	256	256	0	256	0	0	256
01101	256	0	0	256	0	256	256	0
01110	256	0	0	256	0	256	256	0
01111	0	256	256	0	256	0	0	256
10000	256	0	0	256	0	256	256	0
10001	256	0	0	256	0	256	256	0
10010	0	256	256	0	256	0	0	256
10011	0	256	256	0	256	0	0	256
10100	256	0	0	256	0	256	256	0
10101	0	256	256	0	256	0	0	256
10110	256	0	0	256	0	256	256	0
10111	0	256	256	0	256	0	0	256
11000	256	0	0	256	0	256	256	0
11001	256	0	0	256	0	256	256	0
11010	0	256	256	0	256	0	0	256
11011	0	256	256	0	256	0	0	256
11100	256	0	0	256	0	256	256	0
11101	0	256	256	0	256	0	0	256
11110	256	0	0	256	0	256	256	0
11111	0	256	256	0	256	0	0	256

Table C-3. Threshold counts of $\mathcal{P}_3$'s shares.

	$(\mathcal{P}_{3,0}, \mathcal{P}_{3,1}, \mathcal{P}_{3,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
00000	0	256	256	0	256	0	0	256
00001	256	0	0	256	0	256	256	0
00010	0	256	256	0	256	0	0	256
00011	0	256	256	0	256	0	0	256
00100	0	256	256	0	256	0	0	256
00101	256	0	0	256	0	256	256	0
00110	256	0	0	256	0	256	256	0
00111	256	0	0	256	0	256	256	0
01000	0	256	256	0	256	0	0	256
01001	0	256	256	0	256	0	0	256
01010	0	256	256	0	256	0	0	256
01011	256	0	0	256	0	256	256	0
01100	0	256	256	0	256	0	0	256
01101	0	256	256	0	256	0	0	256
01110	256	0	0	256	0	256	256	0
01111	0	256	256	0	256	0	0	256
10000	0	256	256	0	256	0	0	256
10001	256	0	0	256	0	256	256	0
10010	0	256	256	0	256	0	0	256
10011	0	256	256	0	256	0	0	256
10100	0	256	256	0	256	0	0	256

Table C-3 (Continue). Threshold counts of $\mathcal{P}_3$'s shares.

	$(\mathcal{P}_{3,0}, \mathcal{P}_{3,1}, \mathcal{P}_{3,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
10101	256	0	0	256	0	256	256	0
10110	256	0	0	256	0	256	256	0
10111	256	0	0	256	0	256	256	0
11000	256	0	0	256	0	256	256	0
11001	256	0	0	256	0	256	256	0
11010	256	0	0	256	0	256	256	0
11011	0	256	256	0	256	0	0	256
11100	256	0	0	256	0	256	256	0
11101	256	0	0	256	0	256	256	0
11110	0	256	256	0	256	0	0	256
11111	256	0	0	256	0	256	256	0

Table C-4. Threshold counts of $\mathcal{P}_4$'s shares.

	$(\mathcal{P}_{4,0}, \mathcal{P}_{4,1}, \mathcal{P}_{4,2})$							
$(x_4, x_3, x_2, x_1, x_0)$	**000**	**001**	**010**	**011**	**100**	**101**	**110**	**111**
00000	256	0	0	256	0	256	256	0
00001	256	0	0	256	0	256	256	0
00010	0	256	256	0	256	0	0	256
00011	0	256	256	0	256	0	0	256
00100	256	0	0	256	0	256	256	0
00101	0	256	256	0	256	0	0	256
00110	256	0	0	256	0	256	256	0
00111	0	256	256	0	256	0	0	256
01000	256	0	0	256	0	256	256	0
01001	256	0	0	256	0	256	256	0
01010	0	256	256	0	256	0	0	256
01011	0	256	256	0	256	0	0	256
01100	256	0	0	256	0	256	256	0
01101	0	256	256	0	256	0	0	256
01110	256	0	0	256	0	256	256	0
01111	0	256	256	0	256	0	0	256
10000	256	0	0	256	0	256	256	0
10001	256	0	0	256	0	256	256	0
10010	0	256	256	0	256	0	0	256
10011	0	256	256	0	256	0	0	256
10100	0	256	256	0	256	0	0	256
10101	256	0	0	256	0	256	256	0
10110	0	256	256	0	256	0	0	256
10111	256	0	0	256	0	256	256	0
11000	0	256	256	0	256	0	0	256
11001	0	256	256	0	256	0	0	256
11010	256	0	0	256	0	256	256	0
11011	256	0	0	256	0	256	256	0
11100	256	0	0	256	0	256	256	0
11101	0	256	256	0	256	0	0	256
11110	256	0	0	256	0	256	256	0
11111	0	256	256	0	256	0	0	256