Enhancing the Mypher Lightweight Cipher Using an LTT-Based Dynamic S-Box for Secure IoT Devices

Eriek Cahyo Purwanto
Cybersecurity and Digital Forensics School of Computing, Telkom University, Bandung, Indonesia
eriekcp@student.telkomuniversity.ac.id

Farah Afianti
Cybersecurity and Digital Forensics School of Computing, Telkom University, Bandung, Indonesia
farahafi@telkomuniversity.ac.id (corresponding author)

Received: 16 December 2025 | Revised: 22 January 2026 | Accepted: 28 January 2026

 DOI: https://doi.org/10.48084/etasr.16985

ABSTRACT

The rapid expansion of Internet of Things (IoT) systems has significantly increased the demand for lightweight cryptographic mechanisms capable of ensuring robust data confidentiality under strict resource constraints, including limited memory, processing power, and energy consumption. Lightweight block ciphers are widely adopted in such environments; however, many existing designs still rely on static Substitution boxes (S-boxes), which introduce fixed substitution patterns that can be analyzed and exploited through linear cryptanalysis. Mypher is a lightweight block cipher designed for efficient operation on resource-constrained devices, yet its security is constrained by the use of a static 4x4 Substitution box (S-box). To address this limitation, this study proposes an enhancement to the Mypher algorithm by integrating a dynamic 4x4 S-box generated using the Linear Trigonometric Transformation (LTT) method. The proposed approach employs key-derived parameters and uses an optimized constant parameter *X*, selected through a nonlinearity-driven search, to generate highly nonlinear and bijective S-boxes. This optimized *X* value is consistently applied during both encryption and decryption to ensure correct and efficient implementation. A comprehensive security evaluation of the generated dynamic S-boxes is conducted using standard cryptographic criteria, including bijectivity, Strict Avalanche Criterion (SAC), Bit Independence Criterion (BIC), and nonlinearity analysis. The experimental results demonstrate significant improvements in cryptographic strength compared to the original Mypher and UL-AES S-boxes, indicating increased resistance to linear attacks. Furthermore, the proposed scheme is implemented and evaluated on an ESP32 microcontroller to assess its practical feasibility in real Internet of Things (IoT) environments. Performance measurements show that the integration of the LTT-based dynamic S-box introduces only a modest increase in computational cost, with encryption and decryption times increasing by approximately 7% and 6%, respectively. These results confirm that the proposed LTT-based dynamic S-box significantly enhances the security of Mypher while preserving its lightweight characteristics, making it a practical and effective cryptographic solution for securing data on resource-constrained IoT devices.

Keywords-dynamic s-box; ESP32; IoT security; lightweight cryptography; linear trigonometric transformation; mypher block cipher

I. INTRODUCTION

The advancement of IoT devices has significantly transformed modern digital ecosystems by enabling autonomous data acquisition, wireless communication, and remote control across diverse applications, including smart homes, industrial automation, and smart cities. As these technologies further mature, their deployment is expected to expand across industrial and governmental sectors, reinforcing IoT's critical role in shaping future digital ecosystems [1]

Cryptography is essential for securing communications, providing confidentiality, integrity, and authentication. A block cipher is a symmetric cryptographic technique used to encrypt and decrypt blocks of messages using the same key [2]. The key challenges in implementing conventional cryptographic techniques on IoT devices include several inherent limitations of the devices. IoT devices typically have restricted memory resources, reduced computational power, and low battery capacity, while IoT systems often demand real-time responsiveness. The emergence of lightweight cryptography provides an efficient solution for securing data on devices with

limited resources. Lightweight cryptography has several characteristics, including the use of relatively small keys, simple computation, simple key generation, and provision of adequate security. Lightweight cryptography applies not only to IoT devices but also to resource-rich devices that interact directly or indirectly with IoT devices, such as servers, PCs, tablets, and smartphones. Lightweight cryptography is an active research area, with many lightweight block ciphers having been proposed [3]. Lightweight cryptography is a mechanism used to provide security on devices with limited resources such as wireless network sensors, RFID, and embedded systems. Several lightweight block cipher algorithms have been proposed to provide security on devices with limited resources. These algorithms are designed using various architectures, including Feistel, Substitution-Permutation Network (SPN), and hybrid, which is a combination of the two architectures. The proposed algorithms not only focus on efficiency, but security is also a priority, and they must comply with certain standards [4]. Advanced Encryption Standard (AES/Rijndael) [5], PRESENT [6], and GIFT [7] are algorithms with SPN structures that are widely used and developed.

PRESENT is a popular lightweight block cipher. It has been widely used and acts as a reference for various other lightweight block ciphers. This algorithm is built using the SPN architecture and consists of 31 rounds with a message block size of 64 bits. PRESENT has two key variants, namely 80bit and 128bit, but the use of 80bit is more proposed by its inventor. With the emergence of PRESENT, various lightweight block cipher algorithms were developed, which provided advantages in terms of security or efficiency.

GIFT is a development of PRESENT, offering better efficiency. It has two versions: GIFT-64, which uses 28 rounds with a 64-bit block size, and GIFT-128, which uses 40 rounds with a 128-bit block size. Both versions have the same key size of 128 bits. This algorithm is considered to offer better efficiency because it implements smaller subkey additions, fewer rounds, and a faster and simpler key schedule.

Mypher [8] is a combination of PRESENT, PRINCE [9], and TWINE [10]. This algorithm is a lightweight block cipher with a 64-bit block size and a key length of 80 bits. Mypher aims to provide moderate security with relatively low computational costs. This algorithm is designed with the SPN architecture and is suitable for use on IoT devices such as sensors. Like PRESENT, GIFT, and AES, Mypher relies on a static S-box. The use of static S-boxes renders ciphers susceptible to linear cryptanalysis and differential characteristic searches [11].

AES is a cryptographic algorithm established by the U.S. National Institute of Standards and Technology (NIST). This algorithm is based on an SPN structure with a 128-bit block size input. There are three key options available: 128-bit with 10 rounds, 192-bit with 12 rounds, and 256-bit with 14 rounds. AES is resistant to classic attacks such as linear and differential cryptanalysis. However, for encryption on devices with limited resources, various lightweight algorithms have been developed, providing high efficiency. AES aims at reducing computational and energy overhead to better support IoT environments, with ongoing design enhancements expected to contribute to a secure and lightweight IoT ecosystem [12].

Authors in [13] proposed an innovative lightweight S-box for optimizing the AES aimed at enhancing performance and reducing memory consumption in IoT applications. Their design features a compact S-box containing only 16 entries, which significantly lowers storage requirements from the standard 512 bytes to just 16 bytes, all while ensuring effective encryption and decryption. This optimization not only streamlines memory usage but also results in substantial improvements in execution time, showcasing its potential as an efficient cryptographic solution for resource-constrained devices. Research on Lightweight Dynamic Crypto (LWDC) concentrated on refining the algorithm was introduced in 2008. The core components of LWDC, namely the index generation, encryption, and decryption processes, were enhanced to improve its stability and resistance against various attacks. Focus was also placed on strengthening the dynamic key generation mechanism and optimizing internal operations to ensure that the algorithm remained lightweight while achieving higher security. However, despite these enhancements, LWDC still relied on a static S-box, which remained a significant limitation in terms of resistance to advanced cryptanalytic techniques. These cumulative contributions positioned LWDC as a relevant lightweight symmetric encryption scheme for high-speed and secure communication in modern network environments [14].

Authors in [15] modified the AES, a commonly employed standard encryption algorithm with the purpose improving its security. Three modifications were made, one of which was to replace the static S-box with a dynamic S-box. The results were then tested using the Basic Five Statistical Tests and the NIST Tests Suite. The findings of the test showed that the modified algorithm passed the test. Authors in [16] modified the PRESENT algorithm by also replacing the static S-box with a dynamic S-box. The modified algorithm was resistant to various cryptanalysis methods. In addition, the dynamic nature of the S-box made the algorithm resistant to linear attacks because attackers cannot identify linear probabilities in the algorithm.

The application of dynamic S-boxes in LWDC, AES, and PRESENT demonstrates that key-dependent dynamic S-boxes can significantly enhance cipher security without introducing a substantial increase in computational cost. Consequently, dynamic S-box designs are well-suited for lightweight and resource-constrained environments. This comparison highlights the advantage of dynamic S-box approaches in maintaining strong security while satisfying the efficiency requirements of modern lightweight cryptographic systems. Various methods have been proposed for generating S-boxes in cryptographic systems, including approaches based on Linear Trigonometric Transformations (LTT) [17], chaotic maps [18], and elliptic curves [19].

LTT [17] aims to provide a mechanism for using a dynamic S-box that is robust and efficient. The parameters used to generate the dynamic S-box involve utilizing part of the key. Consequently, any variation in the key results in a different S-box. In this study, there are two stages in generating the S-box,

namely a preliminary S-box generation and an improvisation plan. Preliminary S-box generation is used to generate S-boxes in a straightforward manner using the LTT method. While the improvisation plan is used to improve nonlinearity in the preliminary S-box. The S-box generated using the LTT method is then tested using several tests. According to the test findings, the resulting S-box has good quality.

Mypher has a high level of efficiency, making it suitable for devices with limited resources. To enhance its security, this study modified the algorithm by replacing the static S-box with a dynamic S-box generated by the LTT method. The method can generate S-boxes with high cryptographic strength [17]. A contribution of this study is the replacement of a floating-point parameter used in the S-box generation process with a fixed constant, to simplify computation and improve efficiency, particularly for resource-constrained devices. The purpose of this study is to enhance the security of Mypher and provide an alternative lightweight cryptography solution for securing resource-constrained devices. With dynamic S-boxes, the substitution value can be varied, thereby improving the security of the Mypher algorithm.

II. PROPOSED METHOD

To ensure compatibility with the Mypher structure, the LTT parameters are adjusted to generate a 4×4 S-box using variables (A, B, C) derived directly from the secret key, in conjunction with an optimized constant parameter X. Unlike conventional static designs, this approach generates unique substitution patterns for every encryption process, thereby increasing resistance to linear cryptanalysis without altering the core operations of the cipher.

A. Modified Mypher Using Dynamic S-Box LTT

The use of dynamic S-boxes makes it more difficult to identify patterns that could be exploited in linear attacks, thereby enhancing the resistance of the Mypher-LTT algorithm to such attacks. The encryption process in Mypher-LTT is shown in Figure 1.

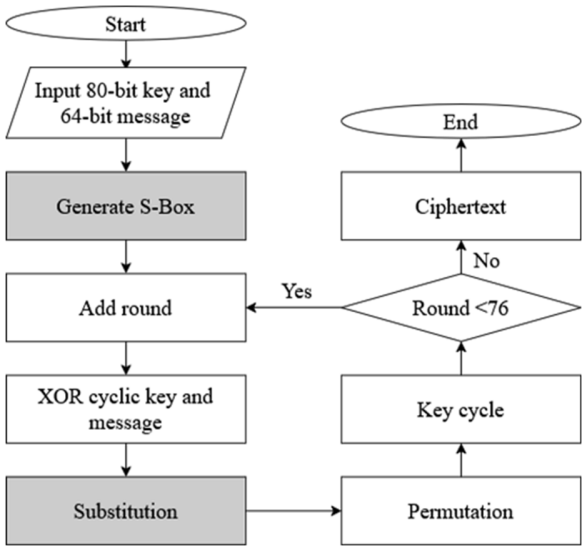

Fig. 1. Flowchart of encryption process in Mypher-LTT.

1) Input

The input of Mypher-LTT is a 64-bit plaintext and an 80-bit Key. The plaintext is processed to produce ciphertext, while the key is used to generate the S-box by deploying the LTT method and performing the add round key process.

2) S-Box Generation

To generate S-boxes in Mypher-LTT, the LTT method was employed once at the initiation of the encryption process. The S-boxes were then used for each round in the encryption process, aiming to reduce the computational cost of Mypher-LTT. Size adjustment of parameters A, B, and C from 8-bit to 4-bit was performed. The 8-bit parameter was used to generate 8×8 S-boxes, while the 4-bit sized parameter was utilized to generate 4×4 S-boxes. The process begins by initializing variables A, B, C, and X, which are set to a constant value of 0.18. The values of A, B, and C are derived from the 10 Least Significant Bits (LSBs) of the 80-bit key. The value of A is taken from the 8^{th} to the 10^{th} bit, and the value of C is taken from the 1^{st} to the 3^{rd} bit. Then, bit 1 is added to the values of A and C to ensure that the resulting values are odd numbers. The value of B is extracted from the 4^{th} to the 7^{th} bit. These values are then used in the first loop to compute the elements of the array B. Each element $B[i]$ is calculated using the cosine function based on A, B, C, and X according to:

$$B[i] = \cos(A + B) * X * i + C \quad (1)$$

The dynamically adjusted value of X is used as input to the next process. This loop runs 16 times, filling up the array B with values based on the changing X and the cosine computation. In the second part of the algorithm, the process enters another loop that searches for the minimum values in the array B. During this loop, the algorithm compares the current minimum value with each element in B, and if a smaller value is found, it updates the minimum and its corresponding index. The index of the minimum value is stored in the S-box array, and the value in B at that index is marked as selected by setting it to 111. This process is repeated until all positions in the S-box are filled. The result is a 4×4 S-box, where each entry corresponds to the index of the smallest element found during each iteration.

The process of generating S-boxes using the LTT method is depicted in Figure 2. The LTT method is detailed in Algorithm 1. The generated S-boxes are then used to perform an encryption or decryption process.

```
Algorithm 1. Mypher-LTT S-box generation

Input parameters:
K ← initial key
X
A ← ((K ≫ 6) & 15) | 1
B ← ((K ≫ 3) & 15)
C ← ((K ≪ 1) & 15) | 1

Output:
Sbox
```

```
Initializations:
i = 0
j = -1
Loc = 0
while ( i < 16 ) do
R = (A+B) · X
B[i] = cos (R · i + C)
If (X > 0.5) then
X = X · X
else
X = X · 1.75
endif
i = i + 1
endwhile
j = j + 1
while ( j < 16 ) do
Mn = B[ 0 ]
Lc = 0
i = 0
while ( i < 16 ) do
if (Mn > B[i]) then
Mn = B[i]
Lc = i
endif
i = i + 1
endwhile
Sbox[j] = Lc
B[Lc] = 111
j = j + 1
endwhile
return Sbox
```

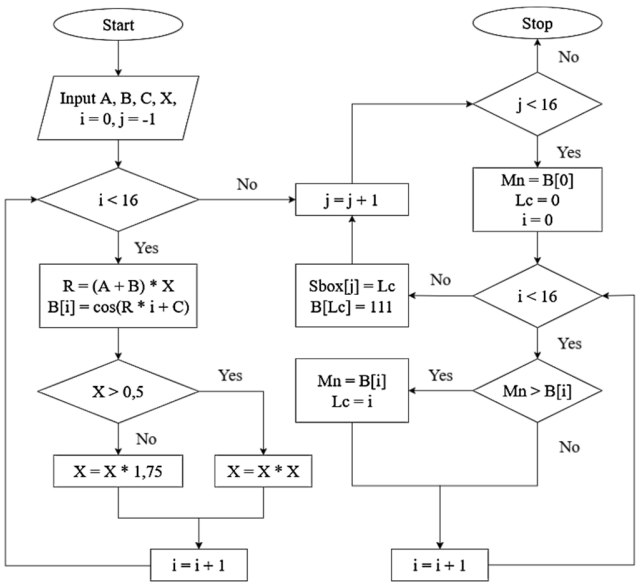

Fig. 2. LTT-based s-box generation in Mypher-LTT.

3) Adding a Round Key

This process involves an XOR operation between the message and the cyclic key. In the first round, the cyclic key is the 64th Most Significant Bit (MSB) of the 80-bit key. In subsequent rounds, the cyclic key consists of the first 64 bits of the output from the key cycle process.

4) Permutation

This process involves changing the positions of the message bits according to the permutation box in Table I.

TABLE I. P-BOX MYPHER-LTT

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
9	C	5	0	F	B	1	A	E	3	6	2	8	7	D	4

5) Key Cycle

This process is used to generate a cyclic key by shifting the key to the right by 16 bits and performing substitutions on the entire 80-bit key.

B. Determining the Value of X

The S-box used in Mypher-LTT is a 4x4 dynamic S-box. It is generated using the LTT method, and the parameters are defined as:

$$X \in \{0 < X < 1\}$$
$$A, C \in \{1, 3, \ldots, 15\} \quad (2)$$
$$B \in \{0, 1, \ldots, 15\}$$

In this study, a search for the value of X as input to generate a dynamic S-box using the LTT method was conducted. The value of X was defined at the beginning with the aim of reducing the parameters or computations needed to generate the S-box. This search was conducted by determining the best nonlinearity for the value $X = b/100$, where $b \in \{1, 2, \ldots, 99\}$. The calculation of nonlinearity in the dynamic S-box LTT was carried out on all possible S-boxes that can be generated, namely, as many as 2^{10}. The results of the nonlinearity calculations on the dynamic S-box LTT showed that 0.18 was the best nonlinearity value, and was selected as the optimal configuration, as illustrated in Figure 3.

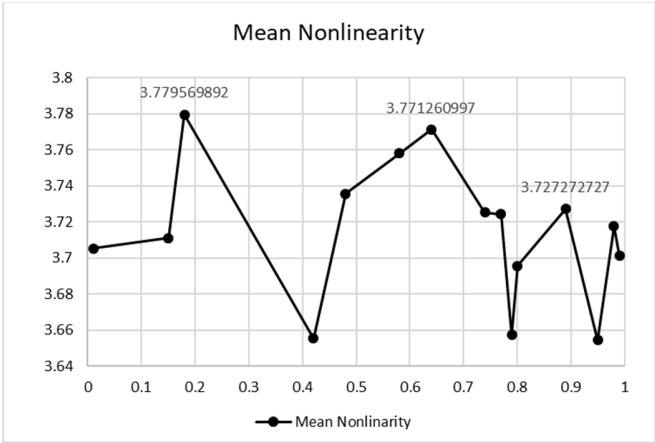

Fig. 3. Nonlinearity of LTT's s-boxes.

III. IMPLEMENTATION AND ANALYSIS

Security analysis was carried out using S-box analysis. An S-box is deemed secure if it satisfies several criteria, including bijectivity, nonlinearity, BIC, and SAC [20]. Thus, the bijectivity, SAC, BIC, and nonlinearity tests were conducted. The S-boxes evaluated include Mypher's S-box [8], UL-AES's S-box [13], and the proposed S-box. Performance evaluation was also carried out by testing the execution speed of Mypher [8], UL-AES [13], and the proposed method in the same test environment. The performance evaluation aims to find the best trade-off between the cryptographic strength and performance. The results of these evaluations were then compared to identify the trade-off between security and efficiency.

A. Security Analysis

Four tests were conducted, namely Bijectivity, SAC, BIC, and the nonlinearity test. The analysis was carried out using S-box samples generated using the LTT algorithm in Table II. Several constants used in the study were based on preliminary experiments to generate the S-box using the values $X = 0.18$, $A = 13$, $B = 9$, and $C = 6$.

TABLE II. S-BOX MYPHER-LTT

0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
D	C	0	8	6	1	E	F	5	9	7	2	3	B	4	A

1) Bijectivity

An n×n S-box satisfies the bijective property if all 2^n of the n input bits are mapped to a unique output vector [20]. This property will guarantee that the S-box has an inverse for the decryption process and ensures that the output vector only appears once. The test results show that the Mypher S-box and the LTT S-box in Mypher have bijective properties, as presented in Table III. This means that each input value will be mapped to a unique output value.

TABLE III. BIJECTIVE RESULTS

S-box	**Results**
Mypher	Bijective
Mypher-LTT	Bijective
UL-AES	Bijective

2) SAC

SAC is used to evaluate the function by examining the effect of replacing one input bit on each output bit [20]. The change in the *i*-th input bit is expected to change the *j*-th bit with a probability of ½, so that it can be said to have/to obtain good diffusion properties. The SAC is calculated as:

$$\frac{1}{2^n}\sum_{j=1}^{n} w(d_j^{e_i}) \quad (3)$$

The SAC value at (*i, j*) is interpreted as the change in the *j* output bit when the i input changes. If the SAC value approaches ½ for each (*i, j*), then the S-box satisfies the SAC with a small error.

SAC testing revealed that 14 output bits changed by 50% in the LTT S-box and 11 in the Mypher S-box was. This shows that the LTT S-box on Mypher has a better SAC value than the Mypher S-box and UL-AES. The results of the SAC test are outlined in Tables IV-VI.

3) BIC

BIC is used to evaluate a function by ensuring that a change in the i-th input bit will cause the two output bits (*i, j*) to change independently [20]. This is intended so that there is no pattern that can be used to guess the relationship between input and output based on small changes in the input.

TABLE IV. MYPHER S-BOX SAC TEST RESULTS

	e_1	e_2	e_3	e_4
Bit-1	50%	50%	50%	75%
Bit-2	50%	50%	50%	50%
Bit-3	75%	25%	75%	75%
Bit-4	50%	50%	50%	50%

TABLE V. MYPHER-LTT S-BOX SAC TEST RESULTS

	e_1	e_2	e_3	e_4
Bit-1	50%	50%	50%	50%
Bit-2	25%	50%	75%	50%
Bit-3	50%	50%	50%	50%
Bit-4	50%	50%	50%	50%

TABLE VI. UL-AES S-BOX SAC TEST RESULTS

	e_1	e_2	e_3	e_4
Bit-1	100%	0%	0%	0%
Bit-2	0%	100%	0%	0%
Bit-3	0%	0%	100%	0%
Bit-4	0%	0%	0%	100%

The BIC requirement for a 4×4 S-box is satisfied if there are exactly 6-bit pairs (*i, j*) that satisfy:

$$\sum_{x=0}^{15} BF[i][x] \oplus BF[j][x] = 8 \quad (4)$$

The test results of BIC on S-box Mypher, S-box LTT, and S-box UL-AES revealed that all three S-boxes passed the BIC test. This shows that a change in one bit on the input will affect the output bit independently so that there is no correlation between changes in the input bit and the output bit. The BIC results are summarized in Table VII. The BIC test results are shown in Tables VIII-X. It is evident that each change in the i-th input produces two independent output bits. It can be observed that for every two output bits, there are 8 out of 16 different bit pairs, so the probability of dependence is $\frac{1}{2}$. This indicates that a change in one bit of the input will affect the output bits independently, with no correlation between the changes in the input bit and the output bit. The BIC test results for Mypher S-box, LTT S-box, and UL-AES S-box show these passed the BIC test.

TABLE VII. BIC RESULT SUMMARY

S-box	**Count**	**Results**
Mypher	6	Passed
Mypher-LTT	6	Passed
UL-AES	6	Passed

TABLE VIII. MYPHER S-BOX BIC TEST RESULTS

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	**Results**
S-BOX	D	9	5	F	0	3	1	C	2	4	B	E	7	A	6	8	
OPBF4	1	1	0	1	0	0	0	1	0	0	1	1	0	1	0	1	
OPBF3	1	0	1	1	0	0	0	1	0	1	0	1	1	0	1	0	
OPBF2	0	0	0	1	0	1	0	0	1	0	1	1	1	1	1	0	
OPBF1	1	1	1	1	0	1	1	0	0	0	1	0	1	0	0	0	
DPBF 3,4	0	1	1	0	0	0	0	0	0	1	1	0	1	1	1	1	8
DPBF 2,4	1	1	0	0	0	1	0	1	1	0	0	0	1	0	1	1	8
DPBF 2,3	0	0	1	0	0	1	1	1	0	0	0	1	1	1	0	1	8
DPBF 1,4	1	0	1	0	0	1	0	1	1	1	1	0	0	1	0	0	8
DPBF 1,3	0	1	0	0	0	1	1	1	0	1	1	1	0	0	1	0	8
DPBF 1,2	1	1	1	0	0	0	1	0	1	0	0	1	0	1	1	0	8

TABLE IX. MYPHER-LTT S-BOX BIC TEST RESULTS

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	**Results**
S-BOX	D	C	0	8	6	1	E	F	5	9	7	2	3	B	4	A	
OPBF4	1	1	0	1	0	0	1	1	0	1	0	0	0	1	0	1	
OPBF3	1	1	0	0	1	0	1	1	1	0	1	0	0	0	1	0	
OPBF2	0	0	0	0	1	0	1	1	0	0	1	1	1	1	0	1	
OPBF1	1	0	0	0	0	1	0	1	1	1	1	0	1	1	0	0	
DPBF 3,4	0	0	0	1	1	0	0	0	1	1	1	0	0	1	1	1	8
DPBF 2,4	1	1	0	1	1	0	0	0	0	1	1	1	1	0	0	0	8
DPBF 2,3	0	1	0	1	0	1	1	0	1	0	1	0	1	0	0	1	8
DPBF 1,4	1	1	0	0	0	0	0	0	1	0	0	1	1	1	1	1	8
DPBF 1,3	0	1	0	0	1	1	1	0	0	1	0	0	1	1	1	0	8
DPBF 1,2	1	0	0	0	1	1	1	0	1	1	0	1	0	0	0	1	8

TABLE X. UL-AES S-BOX BIC TEST RESULTS

	0	**1**	**2**	**3**	**4**	**5**	**6**	**7**	**8**	**9**	**A**	**B**	**C**	**D**	**E**	**F**	**Results**
S-BOX	F	E	D	C	B	A	9	8	7	6	5	4	3	2	1	0	
OPBF4	1	1	1	1	1	1	1	1	0	0	0	0	0	0	0	0	
OPBF3	1	1	1	1	1	0	0	0	1	1	1	1	0	0	0	0	
OPBF2	1	1	0	0	1	1	0	0	1	1	0	0	1	1	0	0	
OPBF1	1	0	1	0	1	0	1	0	1	0	1	0	1	0	1	0	
DPBF 3,4	0	0	0	0	1	1	1	1	1	1	1	1	0	0	0	0	8
DPBF 2,4	0	0	1	1	0	0	1	1	1	1	0	0	1	1	0	0	8
DPBF 2,3	0	1	0	1	0	1	0	1	1	0	1	0	1	0	1	0	8
DPBF 1,4	0	0	1	1	1	1	0	0	0	0	1	1	1	1	0	0	8
DPBF 1,3	0	1	0	1	1	0	1	0	0	1	0	1	1	0	1	0	8
DPBF 1,2	0	0	1	0	0	1	1	0	0	1	1	0	0	1	1	0	8

4) Nonlinearity Test

Nonlinearity is the Hamming distance between a function and a linear or affine function [20]. In the context of cryptography, a function with higher nonlinearity is more secure and difficult to predict. The maximum nonlinear value of a Boolean function with *k* bits is calculated as:

$$2^{k-1} - 2^{\frac{k}{2}-1} \quad (5)$$

The closer the S-box is to the maximum value, the more nonlinear it is, which can be an indicator of better security. Conversely, a low value indicates that a function is approaching linear or affine properties, so it is easy to exploit. After testing the nonlinearity on S-box Mypher and S-box LTT, the results indicate that the LTT S-box in Mypher exhibits higher nonlinearity than Mypher S-box and UL-AES S-box. This is evidenced by the value of 2 in the nonlinearity of S-box Mypher, indicating that the S-box is closer to a linear approximation (Table XI).

TABLE XI. NONLINEARITY TEST RESULTS

S-box	$N(f_1)$	$N(f_2)$	$N(f_3)$	$N(f_4)$
Mypher	4	2	4	4
Mypher-LTT	4	4	4	4
UL-AES	0	0	0	0
Expectation	6	6	6	6

B. Performance Evaluation

Following the security evaluation, the efficiency of the proposed construction [21] was assessed. The performance evaluation was carried out by measuring the execution time (s) of Mypher, Mypher-LTT, and UL-AES under the same environment. Input sizes of 8 bytes, 16 bytes, 32 bytes, 64 bytes, 128 bytes, 256 bytes, 512 bytes, and 1024 bytes were used for testing these algorithms. The testing was performed by calculating the mean value obtained from 30 individual test executions.

The modified Mypher (Mypher-LTT) was implemented and evaluated on an ESP32 platform, as shown in Figure 4, representing a realistic IoT environment. The ESP32 is a System-on-Chip (SoC) microcontroller integrated with Bluetooth and Wi-Fi. The hardware specifications of the ESP32 used include a 240 MHz dual-core processor and 520 KB of SRAM. According the test results, the average encryption and decryption times of Mypher-LTT increased by 7% and 6%, respectively, compared to Mypher. In a separate test, the average encryption and decryption times of Mypher-LTT increased by 54% and 38%, respectively, compared to UL-AES. The results of the encryption and decryption time overhead evaluation are presented in Table XII.

TABLE XII. ENCRYPTION AND DECRYPTION TIME IMPROVEMENT EVALUATION RESULTS

Algorithm	Operation	Increase (%)
Mypher-LTT and Mypher	Encryption	7%
	Decryption	6%
Mypher-LTT and UL-AES	Encryption	54%
	Decryption	38%

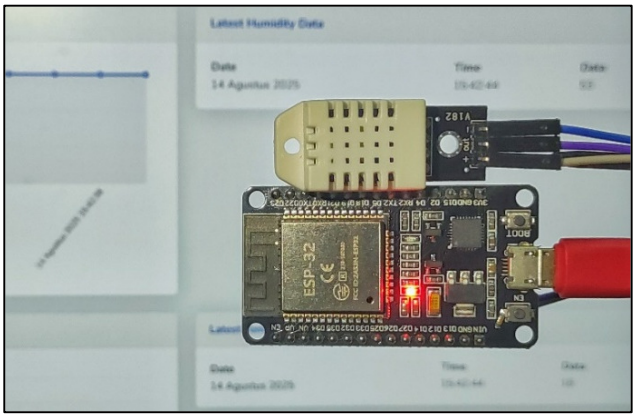

Fig. 4. The testing environment utilizing an ESP32.

TABLE XIII. PERFORMANCE EVALUATION RESULTS

Message	Operation	Mypher	Mypher-LTT	UL-AES
8 byte	Encryption	0.520	0.559	0.363
	Decryption	0.578	0.616	0.446
16 byte	Encryption	1.039	1.112	0.723
	Decryption	1.154	1.226	0.889
32 byte	Encryption	2.077	2.225	1.445
	Decryption	2.308	2.451	1.778
64 byte	Encryption	4.154	4.448	2.889
	Decryption	4.616	4.902	3.554
128 byte	Encryption	8.308	8.896	5.777
	Decryption	9.232	9.802	7.108
256 byte	Encryption	16.614	17.791	11.554
	Decryption	18.463	19.604	14.216
512 byte	Encryption	33.228	35.581	23.106
	Decryption	36.925	39.207	28.432
1024 byte	Encryption	66.454	71.161	46.211
	Decryption	73.848	78.414	56.862

It was observed that the increase in encryption and decryption time in Mypher-LTT is not significant when compared to the original Mypher algorithm. However, the increase becomes significant when compared to UL-AES due to the substantial difference in the number of rounds. It is possible that the increase would not be significant if the round number in Mypher-LTT was equal to that of UL-AES. The performance evaluation results are presented in Table XIII.

IV. CONCLUSION

The widespread adoption of IoT technology has increased the demand for secure and efficient cryptographic mechanisms suitable for resource-constrained environments. This study enhances the Mypher lightweight block cipher by replacing its static S-box with a dynamic S-box generated using the LTT method. The proposed modification, Mypher-LTT, improves cryptographic strength while preserving the lightweight characteristics required for practical deployment on IoT devices.

Security and performance evaluations confirm that the proposed LTT-based dynamic S-box satisfies essential cryptographic criteria, including bijectivity, SAC, BIC, and nonlinearity. Compared to the original Mypher and UL-AES S-boxes, the proposed S-box exhibits improved nonlinearity, indicating increased resistance to linear cryptanalysis. Implementation on an ESP32 platform shows that the enhanced security is achieved with only a modest increase in computational cost, with encryption and decryption times increasing by approximately 7% and 6%, respectively, relative to the original Mypher cipher.

The novelty of this study lies in the integration of an LTT-based dynamic S-box into the Mypher lightweight block cipher using a fixed, optimized parameter X, which reduces computational complexity during S-box generation while maintaining strong cryptographic properties. By establishing a practical balance between enhanced security and performance efficiency, this work demonstrates that improved cryptographic strength can be achieved without compromising deployability on resource-constrained IoT platforms. Consequently, the proposed Mypher-LTT scheme provides an effective and scalable security solution for lightweight IoT cryptographic systems.

ACKNOWLEDGMENT

This research was supported by the Internal Research Grant Fund under the Non-Cooperation C3 Scheme of PPM Telkom University (Grant No. 454) and by the Ministry of Communication and Digital Affairs of the Republic of Indonesia.

REFERENCES

[1] K. Elgazzar *et al.*, "Revisiting the internet of things: New trends, opportunities and grand challenges," *Frontiers in the Internet of Things*, vol. 1, Nov. 2022, https://doi.org/10.3389/friot.2022.1073780.

[2] W. Stallings, *Cryptography and Network Security: Principles and Practice*. Boston Munich: Pearson, 2017.

[3] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities," *IEEE Access*, vol. 9, pp. 28177–28193, 2021, https://doi.org/10.1109/ACCESS.2021.3052867.

[4] S. L. Rana, "A Survey Paper of Lightweight Block Ciphers Based on Their Different Design Architectures and Performance Metrics," *International Journal of Computer Engineering and Information Technology*, vol. 11, no., 6, pp. 119–129, June 2019.

[5] J. Daemen and V. Rijmen, "The Block Cipher Rijndael," in *Smart Card Research and Applications*, 2000, pp. 277–284, https://doi.org/10.1007/10721064_26.

[6] A. Bogdanov *et al.*, "PRESENT: An Ultra-Lightweight Block Cipher," in *Cryptographic Hardware and Embedded Systems - CHES 2007*, 2007, pp. 450–466, https://doi.org/10.1007/978-3-540-74735-2_31.

[7] S. Banik, S. K. Pandey, T. Peyrin, Y. Sasaki, S. M. Sim, and Y. Todo, "GIFT: A Small Present," in *Cryptographic Hardware and Embedded Systems – CHES 2017*, 2017, pp. 321–345, https://doi.org/10.1007/978-3-319-66787-4_16.

[8] Lo'ai, Tawalbeh, M. Alicea, and I. Alsmadi, "New and Efficient Lightweight Cryptography Algorithm for Mobile and Web Applications," *Procedia Computer Science*, vol. 203, pp. 111–118, Jan. 2022, https://doi.org/10.1016/j.procs.2022.07.016.

[9] J. Borghoff *et al.*, "PRINCE – A Low-Latency Block Cipher for Pervasive Computing Applications," in *Advances in Cryptology – ASIACRYPT 2012*, 2012, pp. 208–225, https://doi.org/10.1007/978-3-642-34961-4_14.

[10] T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, "TWINE: A Lightweight Block Cipher for Multiple Platforms," in *Selected Areas in Cryptography*, 2013, pp. 339–354, https://doi.org/10.1007/978-3-642-35999-6_22.

[11] N. Zakaria, A. Ahmad, A. Halim, and F. Ridzuan, "SECURITY ANALYSIS BETWEEN STATIC AND DYNAMIC S-BOXES IN BLOCK CIPHERS," *Journal of Information System and Technology Management*, vol. 6, pp. 10–16, Mar. 2021, https://doi.org/10.35631/JISTM.620002.

[12] Amrita, C. P. Ekwueme, I. H. Adam, and A. Dwivedi, "Lightweight Cryptography for Internet of Things: A Review," *EAI Endorsed Transactions on Internet of Things*, vol. 10, Mar. 2024, https://doi.org/10.4108/eetiot.5565.

[13] Y. S. Vaz, J. C. B. Mattos, and R. I. Soares, "Improving an Ultra Lightweight AES for IoT Applications," in *2023 IEEE 9th World Forum on Internet of Things (WF-IoT)*, Oct. 2023, pp. 01–06, https://doi.org/10.1109/WF-IoT58464.2023.10539597.

[14] A. Al-Omari, "Lightweight Dynamic Crypto Algorithm for Next Internet Generation," *Engineering, Technology & Applied Science Research*, vol. 9, pp. 4203–4208, June 2019, https://doi.org/10.48084/etasr.2743.

[15] A. Maolood and Y. Ali, "Modifying Advanced Encryption Standard (AES) Algorithm," *Journal of Al-Rafidain University College For Sciences*, pp. 259–285, Oct. 2021, https://doi.org/10.55562/jrucs.v41i3.187.

[16] Y. C. A. and S. S. R. K., "Key-based dynamic S-Box approach for PRESENT lightweight block cipher," *KSII Transactions on Internet & Information Systems*, vol. 17, no. 12, Dec. 2023, Art.no. 3398, https://doi.org/10.3837/tiis.2023.12.010.

[17] A. H. Zahid *et al.*, "Efficient Dynamic S-Box Generation Using Linear Trigonometric Transformation for Security Applications," *IEEE Access*, vol. 9, pp. 98460–98475, 2021, https://doi.org/10.1109/ACCESS.2021.3095618.

[18] A. Zahid, M. Arshad, M. Ahmad, N. Soliman, and W. El-Shafai, "Dynamic S-Box Generation Using Novel Chaotic Map with Nonlinearity Tweaking," *Computers, Materials and Continua*, vol. 75, pp. 3011–3026, Mar. 2023, https://doi.org/10.32604/cmc.2023.037516.

[19] G. Murtaza, N. A. Azam, and U. Hayat, "Designing an Efficient and Highly Dynamic Substitution-Box Generator for Block Ciphers Based on Finite Elliptic Curves," *Security and Communication Networks*, vol. 2021, no. 1, 2021, Art. no. 3367521, https://doi.org/10.1155/2021/3367521.

[20] C. Adams and S. Tavares, "The structured design of cryptographically good s-boxes," *Journal of Cryptology*, vol. 3, no. 1, pp. 27–41, Jan. 1990, https://doi.org/10.1007/BF00203967.

[21] S. Ibrahim and A. M. Abbas, "A Novel Optimization Method for Constructing Cryptographically Strong Dynamic S-Boxes," *IEEE Access*, vol. 8, pp. 225004–225017, 2020, https://doi.org/10.1109/ACCESS.2020.3045260.