Key-based dynamic S-Box approach for PRESENT lightweight block cipher

Yogaraja C A[1*], and Sheela Shobana Rani K[2]
[1] Department of Computer Science and Engineering,
Ramco Institute of Technology, Rajapalayam,
Tamilnadu, India.
[e-mail: raja2dlas@gmail.com]
[2] Karpagam College of Engineering, Coimbatore,
Tamilnadu, India.
[e-mail: yuvsheka@gmail.com]
[*]Corresponding author: Yogaraja C A

Received April 3, 2023; revised October 2, 2023; accepted November 19, 2023; published December 31, 2023

Abstract

Internet-of-Things (IoT) is an emerging technology that interconnects millions of small devices to enable communication between the devices. It is heavily deployed across small scale to large scale industries because of its wide range of applications. These devices are very capable of transferring data over the internet including critical data in few applications. Such data is exposed to various security threats and thereby raises privacy-related concerns. Even devices can be compromised by the attacker. Modern cryptographic algorithms running on traditional machines provide authentication, confidentiality, integrity, and non-repudiation in an easy manner. IoT devices have numerous constraints related to memory, storage, processors, operating systems and power. Researchers have proposed several hardware and software implementations for addressing security attacks in lightweight encryption mechanism. Several works have made on lightweight block ciphers for improving the confidentiality by means of providing security level against cryptanalysis techniques. With the advances in the cipher breaking techniques, it is important to increase the security level to much higher. This paper, focuses on securing the critical data that is being transmitted over the internet by PRESENT using key-based dynamic S-Box. Security analysis of the proposed algorithm against other lightweight block cipher shows a significant improvement against linear and differential attacks, biclique attack and avalanche effect. A novel key-based dynamic S-Box approach for PRESENT strongly withstands cryptanalytic attacks in the IoT Network.

Keywords: Cryptanalysis, critical data, dynamic S-Box, lightweight encryption, security threats

1. Introduction

One of the notable revolutions in this modern technological era is IoT Technology and it has become part of day-to-day routine activities. It also contributes to the economic growth of a country. Interconnection of devices has been in practice for the last five decades, the term IoT was coined in 1999 [1][2] and the real boom of IoT has been realized only in the last five years. With the exponential growth of IoT, its applications have also expanded into agriculture, vehicles, transportation, healthcare, manufacturing, civil structures and even lifestyle. IoT connects devices such as smart gadgets or sensors that are located in separate networks with different access mechanisms. These devices are also capable of interacting with one another through the internet. Simultaneously, since billions of IoT gadgets, applications, and networks are currently being used, the potential to increase its effectiveness is high. In recent days, devices with edge computing technology have moved ahead by performing computing near the source of data. Many smart devices require personal information of the user to provide personalized and customized services. To secure data, few basic security principles of IoT devices are to be familiarized [3].

Confidentiality: Confidentiality and privacy are significantly identical concepts. Measures for maintaining confidentiality aim to provide protection against unauthorized access to sensitive data. Data is frequently categorized based on the scope and nature of the attack that could result from it getting into the attacker.

Integrity: Integrity refers to maintaining the data throughout its full lifecycle in terms of consistency, accuracy, and dependability. Data cannot be altered while in transmission, and precautions need to be taken to prevent unauthorized parties from changing the data.

Availability: Data should be available consistently and easily to authorized persons for ease of access.

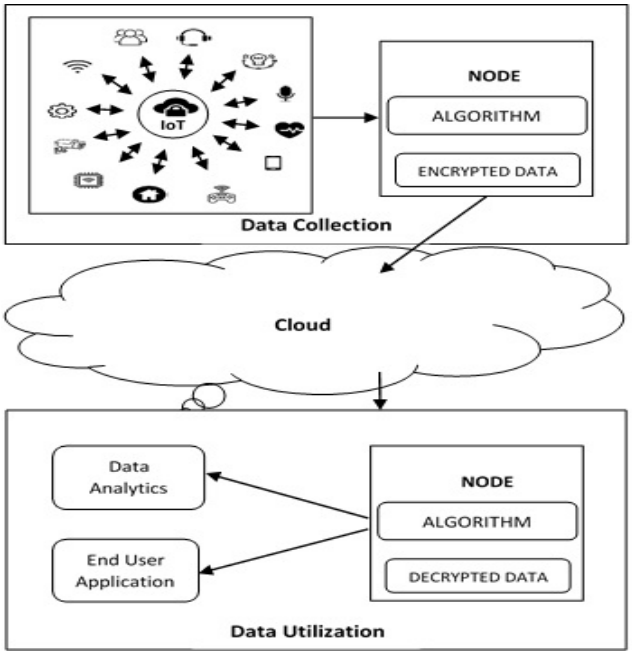

Fig. 1. Secured IoT Architecture.

In near future, IoT paradigm has the scope to take autonomous decisions even in a critical infrastructure. The security of IoT devices has been a concern for a long time because of chances of both minor and major attacks. A large portion of these attacks originate from direct security flaws, such as the use of default passwords on the network. Majority of the IoT network relies on resource-constrained sensors that frequently transmit sensitive and private data. A more common secured IoT architecture is shown in **Fig. 1**, It takes researcher's focus to secure the transmitting data between sensors and the server rather than focusing on reducing complexity [4]. The traditional asymmetric encryption algorithms like RSA, Elliptic Curve Cryptography(ECC) and Diffie-Hellman algorithms are used largely to encrypt small data because of the time taken to process the cipher text with the two keys (namely private and public key) of larger size. Moreover, these algorithms require high resources in terms of power, memory and processing. On the other hand, symmetric algorithms like DES, AES and RC4 are faster than asymmetric algorithms and allow the encryption of larger data [5].

The symmetric algorithms have two distinct styles of processing the plaintext, namely, stream and block ciphers. A key used by stream ciphers has the same size as the data. Plaintext is processed in a bit fashion to create the encrypted text. Only few lightweight stream ciphers with high throughput have been proposed by researchers. Some examples are Salsa20/12 [6], Rabbit [7], HC-128 [8] and SOSEMANUK [9] which are part of software oriented stream ciphers. Trivium [10], MICKEY 2.0 [11] and Grain [12] are hardware oriented stream ciphers. Despite having a low throughput, Grain 128 [12] is one of the most popular and appropriate lightweight cyphers for restricted devices. They utilize LFSR and a non-linear filtering function.

Symmetric block ciphers are predominant in practice than Symmetric stream ciphers. This is because symmetric block ciphers use key as block. whereas in stream cipher, a proper pseudorandom generator is necessary to generate keys. Block ciphers are popularly implemented with three different structures, namely: Feistel Structure, Substitution Permutation Network(SPN) and Add-Rotate-XOR(ARX) [13]. Few common algorithms like DES [14], TEA [15], CAST [16] and GOST [17] fall under Fiestel Structure and algorithms like AES [18], Kalyna [19], SHARK [20], PRESENT [21][22], TWINE [23], PICARO [24], SKINNY [25], GIFT [26] and Serpent [27] come under Substitution Permutation Network. Likewise, Chaskey [28], IDEA [29], LEA [30], HIGHT [31] are algorithms of Add-Rotate-XOR structure.

Like any cryptographic algorithm, the key issue for the PRESENT algorithm is to keep its security from being compromised by several types of advanced attacks. It is critical to maintain PRESENT's resistance against differential and linear cryptanalysis as well as other well-known assaults as cryptanalysis techniques progress. The proposed work focuses the confidentiality in the security triad for securing the data against improvised cryptanalytic attacks. Results are compared with the relevant implementations like ANU, LED, L-Block and FEW [43] for the evaluation of the proposed algorithm.

2. Related Work

2.1 Substitution Permutation Network

An SPN is a recursive construction of substitution and permutation [32], in which a different key is used in each round for XOR operation to produce a cipher text from a plain text. The functional operations used in the first, last and intermediate rounds are varied in SPN. The number of S-Boxes needed for SPN is determined from the block size and the word size.

Substitution adds confusion and permutation adds diffusion properties to the SPN structure. A well-designed S-Box holds the property of avalanche effect, and a minor change in the plaintext or key affects the cipher text largely. AES [33], PRESENT [21] and GIFT [26] algorithms are commonly deployed with the SPN structure.

2.2 PRESENT: A Lightweight method

With the evolution of quantum computers, the use of block sizes of over 128-bit has diminished because of the limited resources on the devices. This brings the PRESENT into the implementation on resource constrained devices "Ultra-Lightweight Block Cipher". PRESENT is a Substitution Permutation Network based cipher which consists of 31 rounds with a block size of 64 bits and key sizes of 80 bits and 128 bits [21]. The algorithm can withstand differential and linear cryptanalysis attacks to a reasonable extent. Algebraic attacks are possible with minimal S-boxes and when seven S-boxes (block size of 28 bits) are used, attack is not possible.

Linear and differential cryptanalysis is used to choose the sixteen numbers of good S-Boxes from SERPNET, Finite Fields and Hummingbird. Among sixteen S-Boxes, a random S-Box is chosen for each round of the PRESENT algorithm. Round key value of 64-bit is of 16 digits, where each digit is a 4-bit value. All 16 digits are XORed and a single 4-bit value is obtained to select the S-Box from the chosen sixteen good S-Boxes. The Linear and differential cryptanalysis of this approach is considerably better than the original PRESENT [34].

A lightweight mechanism for resource constrained environment is proposed in [35], in which a Key scheduling algorithm is used to generate 30 subkeys of each 64-bit length to cater to I-PRESENT-80 and I-PRESENT-128. It performs 15 round functions, an involutive function and another 15 inverse round functions to produce ciphertext. Encryption and decryption of this approach use the same circuit. [36] The confusion property is added to S-Box and fused with GRP (group operation) for a hybrid cryptosystem. Implementations are tested in LPC2129 Processor against fused S-boxes of PRESENT, TEA, CLEFIA with GRP. It is found that PRESENT consumes very less memory when compared to other lightweight algorithms. To withstand the property of avoiding fixed points in S-Box, [37] genetic algorithm based S-Boxes (S1 and S2) are proposed. Dynamic S-Box is chosen by splitting block into 8-bits along with key selection function. It enhances the diffusion rate and solves the anti-fixed point problem of S-Box in PRESENT.

PRESENT algorithm is proposed with two design approaches to overcome few security issues in resource limited IoT environment. Both approaches aim to reduce the implementation size and the energy required for the key generation phase. The basic idea was to divide the 80-bit key into four 20-bit words and 128-bit key into four 32-bit words. Dividing the key of size 16-bit words requires five clock cycles and eight clock cycles respectively. Experimental results show a significant reduction in area, and a subsequent downfall is also observed in the performance of both 80-bit and 128-bit key size.

To maintain the higher level of complexity in S-Box design, several S-Box generator schemes are proposed by researchers. An ordered elliptic curve and binary sequence based S-Box generator scheme proposed [38] with strong security mechanism against modern security attacks with nonlinearity range between 0.117 and 0.172.

In practical applications, most of the IoT devices transmit only few bits of data which on average is less than 64 bits. An improved version of PRESENT cipher is GIFT [26], which comes in two different block sizes of 64 and 128 bits with a common key size of 128 bits. It provides a higher throughput with lighter S-Box. It also comes with reduced rounds of operation and a simple key schedule. RECTANGLE [39] is another cipher derived from

PRESENT with rounds reduced to 25 for meeting the lightweight property. Chatterjee et. al reduced the encryption round from 31 to 25 by adding a new layer in between substitution and permutation [40]. Key generation for each round is updated by encrypting the round key with delta value function of Tiny Encryption Algorithm (TEA) for 80-bit key.

Several approaches including dynamic S-Box on the PRESENT algorithm have been proposed to solve anti-fixed point in [37]. To improve the performance of PRESENT, the number of logic gates and critical path delay is dramatically reduced by using Karnaugh map in [41]. ANU [43] is a very light-weight cipher built on the Feistel network that is intended for use in the Internet of Things (IoT). The ANU cipher seeks to offer secure communication while reducing memory and power usage. Compared to other lightweight ciphers, it has a far more compact structure for a 128-bit key length.

3. Methodology

This section explains in detail about the proposed substitution using key-based dynamic S-Box, Permutation and Adding Round key process for encryption and decryption. The simplified diagram of the operations is shown in **Fig. 2**.

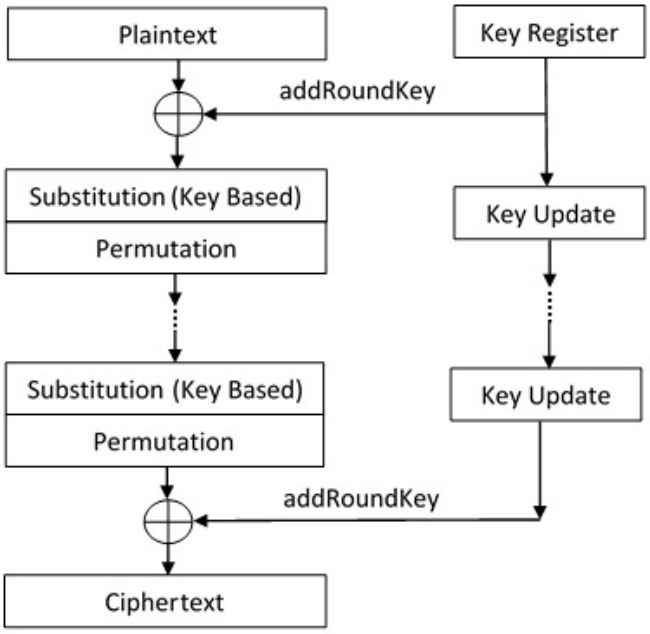

Fig. 2. Key-based dynamic S-Box approach for PRESENT.

An algorithmic representation of each operation is presented in **Algorithm 1**. The detailed explanation of the implementation of the three operations is provided alongside the key generation for each round.

3.1 Substitution using key-based dynamic S-Box

S-Box in PRESENT cipher in **Table 1** is a 4-bit substitution technique, where a 4-bit (nibble) input is mapped to the 4-bit output. It is initially loaded with a 64-bit value (4-bit X 16). For each round, a new S-Box is computed from the key.

Table 1. S-Box of PRESENT algorithm

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

Algorithm 1. PRESENT algorithm using Key-based dynamic S-Box

Input: X is a 64-bit plaintext, K is a Key, S-Box of b4 bit and N is a number of Rounds
1: Construct S_0 to S_{15} from S
2: Derive N round keys K_i from K
3: STATE ← X
4: **for i = 1 to N do**
5: Construct STATE as $STATE_0 \cdots STATE_{15}$, where $STATE_j$ is a 4-bit nibble
6: **for j = 0 to 15**
7: $STATE_j \leftarrow S_{i,j}(STATE_j)$
8: **end for**
9: Reassemble STATE
10: Bit permutation P_i to STATE
11: Add round key K_i to STATE
12: XOR ($STATE_0$, K_i)// 4 bits
13: **end for**
14: C ← STATE

A traditional S-Box technique uses the same S-Box for all 31 rounds of encryption. With the evolution, key based S-Box proposed with the fixed key for each round. All the previous proposed schemes are having the linear property in the S-Box generation techniques, which enables the attacker to leverage the algorithm and get chance of compromising the system with advanced cryptanalytic attacks. In this approach, we proposed a novel technique for S-Box generation that completely relies on the dynamic key used for encryption rather than the fixed S-Box and fixed key based S-Box techniques.

In this approach, LSB 4-bit (nibble) selected from the round key as round constant. S-Box is divided into 16 4-bits and the round constant (4-bit) is selected from the round key. It is then XORed with the 4-bit of S-Box value to key-based dynamic S-Box for the next round. The same is represented in **Fig. 3**. The values in the S-Box are varied regularly to keep attackers away from the cryptanalysis. Performance of the S-box largely relies on factors such as the avalanche effect, nonlinearity, diffusion property, anti-fixed point and fixed point. This approach addresses almost all aforementioned performance factors.

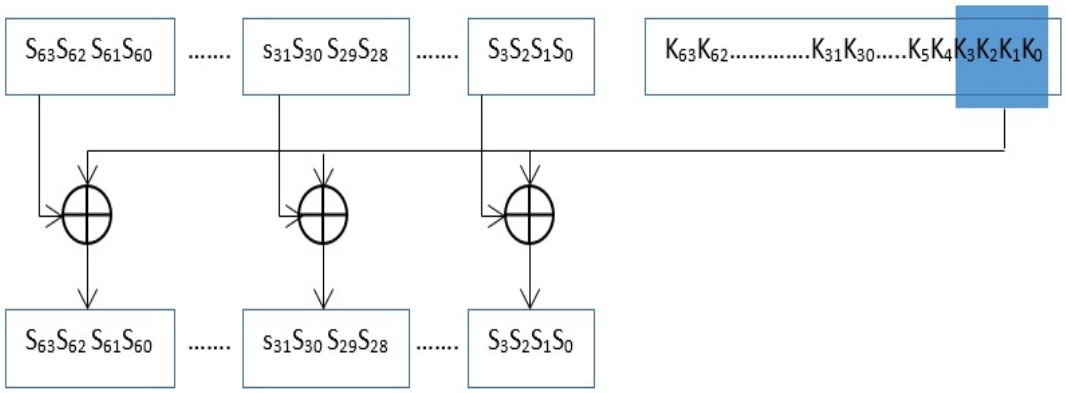

Fig. 3. Key-based dynamic S-Box.

3.2 Permutation

Permutation involves a large number of operations. It changes the original bit order and produces the shuffled output for the next round input state. In this permutation layer, i^{th} input bit of a state is moved to P(i) bit of the output state. Permutation simply swaps the bit within a state to produce output state. Initially 64 bits are arranged in a sequence fashion; after the first permutation, the state changes as shown in **Table 2**.

Table 2. Permutation function

I	0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
P(i)	0	16	32	48	1	17	33	49	2	18	34	50	3	19	35	51
I	16	17	18	19	20	21	22	23	24	25	26	27	28	29	30	31
P(i)	4	20	36	52	5	21	37	53	6	22	38	54	7	23	39	55
I	32	33	34	35	36	37	38	39	40	41	42	43	44	45	46	47
P(i)	8	24	40	56	9	25	41	57	10	26	42	58	11	27	43	59
I	48	49	50	51	52	53	54	55	56	57	58	59	60	61	62	63
P(i)	12	28	44	60	13	29	45	61	14	30	47	62	15	31	47	63

From the **Table 2**, the bits 0, 21, 42 and 63 remain unchanged in their initial positions. The remaining 60 bits are swapped by grouping them into 20 3-bit groups to exchange their positions. Bit b1 is swapped with bit b4 and again bit b1 is swapped with bit b16. From the group of 20, 1 group completes the permutation and the remaining 19 groups follow the same procedure. This can be represented as follows:

$$\text{Swap } (b_1, b_4) \text{ and Swap } (b_1, b_{16})$$

3.3 Key Generation

PRESENT Cipher has 80-bit and 128-bit keys that run key schedule algorithm to generate a 64-bit key for the add round key operation. For the 128-bit key, the input size recommended for NIST statistical tests is 100-bit. Key generation for 128-bit architecture is shown in **Fig. 4** Initially, the 128 bit key is stored as $K_{127}K_{126}K_{125}\ldots..K_2K_1K_0$ in the Key Register K. For the first round, the leftmost 64-bit of the original 128-bit secret key is chosen as K_1, and for the next 31 rounds, the key is generated as below:

- 61 bits Left Shift of Key K, i.e.., $K_i=[K_{66}K_{65}K_{64}\ldots\ldots K_1K_0K_{127}K_{126}\ldots..K_{68}K_{67}]$ for any i^{th} round
- Update the Key Register $K=K_i$ for $K= [K_{127}K_{126}K_{125}\ldots..K_2K_1K_0]$
- Substitute leftmost four bits of K $[K_{127}K_{126}K_{125}K_{124}]$ with S-Box $[K_{127}K_{126}K_{125}K_{124}]$
- Substitute next four leftmost bits of K $[K_{123}K_{122}K_{121}K_{120}]$ with S-Box $[K_{123}K_{122}K_{121}K_{120}]$
- Perform XOR operation for 5-bits of Least Significant Bit in round counter i with $K[K_{66}K_{65}K_{64}\ K_{63}K_{62}]$

For the 80-bit key, initially the key is stored as $K_{79}K_{78}K_{77}\ldots..K_2K_1K_0$ in the Key Register K. The leftmost 64-bits in key K is used as key K_1 in the first round. For further rounds, the key is generated as follows:

- 61 bits Left Shift of Key K, i.e.., $K_i=[K_{18}K_{17}K_{16}\ldots\ldots K_1K_0K_{79}K_{78}\ldots..K_{20}K_{19}]$ for any i^{th} round
- Update the Key Register $K=K_i$ for $K= [K_{79}K_{78}K_{77}\ldots..K_2K_1K_0]$
- Substitute leftmost four bits of K $[K_{79}K_{78}K_{77}K_{76}]$ with S-Box $[K_{79}K_{78}K_{77}K_{76}]$
- Substitute next four leftmost bits of K $[K_{75}K_{74}K_{73}K_{72}]$ with S-Box $[K_{75}K_{74}K_{73}K_{72}]$
- Perform XOR operation for 5-bits of Least Significant Bit in round counter i with $K[K_{19}K_{18}K_{17}\ K_{16}K_{15}]$

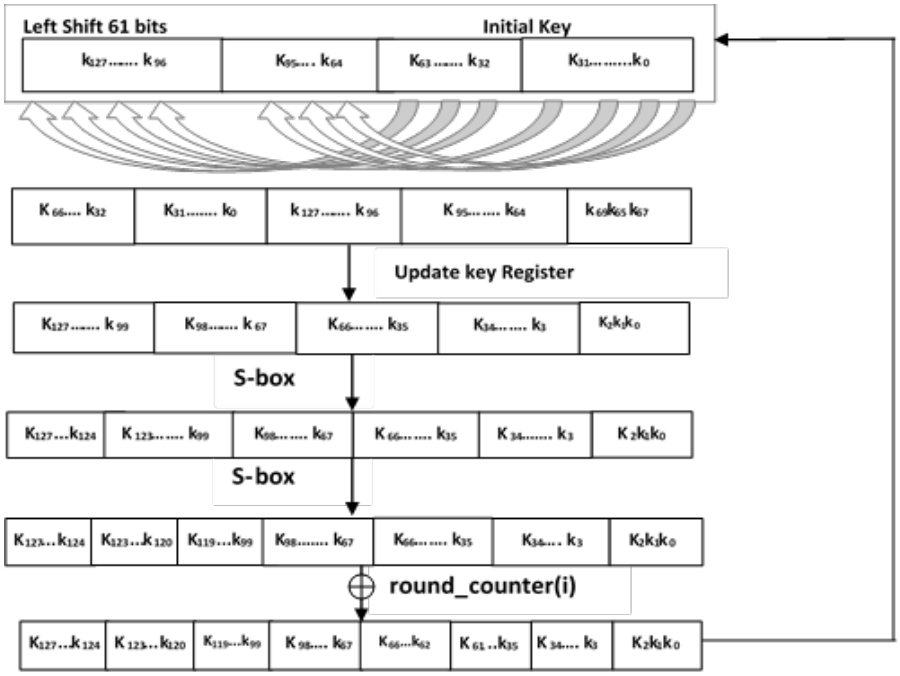

Fig. 4. PRESENT Key generation for 128-bit.

3.4 Add Round Key

In this step, the 64-bit state is XORed with the 64-bit round key that is generated from the key schedule. A bitwise XOR operation is performed with equal size of bits from the input state and the key at this stage. These all operations are repeated for 31 rounds to acquire the ciphertext.

4. Results

The increasing popularity of Lightweight cryptography algorithm is promising for both new conveniences and new privacy concerns. This section provides the performance analysis of key-based dynamic S-Box technique against the PRESENT-80, ANU-128, L-Block, FEW and LED-80 algorithms. Implementations of the algorithm are purely based on the specifications provided. The results discussed and provided are implemented using Python 3.10.2 on Intel i5 windows 10 machine. Initially, the CPU performance for testing is maintained below 10%. The values provided in the results are obtained from the average of runs at each stage. Security parameters like linear cryptanalysis, differential cryptanalysis, biclique attacks and hamming distances are analyzed. Additionally, efficiency parameters like memory utilization, computational cost and encryption time are analyzed in this section.

4.1 Linear cryptanalysis

Linear cryptanalysis is a kind of traditional security attack targeted mostly on symmetric block cipher to identify the linear relationships between the plaintext, ciphertext, and the key used. Linear relationships of each known plaintext and ciphertext can be tabulated as Linear Approximation Table (LAT) against each input and output bits for the corresponding key, also it is known as correlation probability. Bias for the linear probability can be denoted as $|P_L\text{-}1/2|$ for the proposed key based dynamic S-Box. Bias probability for n rounds are calculated using Matsui's Piling-up lemma [43]. **Table 3** shows the comparisons of linear cryptanalysis attack for various lightweight algorithms.

Table 3. Linear attack comparisons

Algorithm/Cipher	Rounds	Number of known plaintext	Reference
Proposed(80 bit)	18	2^{122}	This paper
PRESENT-80	25	2^{102}	[43]
ANU-128	18	2^{110}	[43]
L-Block	15	2^{66}	[43]
FEW	27	2^{90}	[43]

For three rounds of proposed algorithm, it was found ten S-Boxes are active. Bias(ε) can be calculated as

$$\varepsilon = 2^{9} \times (2^{-2})^{\wedge 10}$$
$$\varepsilon = 2^{-11}$$

Extending the above to 18 rounds by 6 times, the bias(ε) given as

$$\varepsilon = 2^5 \times (2^{-11})^{\wedge 6}$$
$$\varepsilon = 2^5 \times (2^{-66})$$
$$\varepsilon = 2^{-61}$$

Linear attack complexity can be determined as follows

$$N_L = 1/(\varepsilon)^2$$

For 18 rounds,

$$N_L = 1/(2^{-61})^{\wedge 2}$$
$$N_L = 2^{122}$$

N_L Indicates the number of known plaintext and ciphertext pairs

4.2 Differential cryptanalysis

Block cipher analysis using differential cryptanalysis is a form of selective plaintext attack. It entails researching the likelihood that particular plaintext differential values may differentially propagate during the encryption process. The block cipher's key can be obtained by locating a high probability differential trail. The difference distribution table is used to investigate the S-box, a non-linear element of our design. The difference between the high probability input and output for each round is taken into account while forming the differential trails. A non-zero input or output difference characterizes an active S-box. As like linear cryptanalysis, amount of chosen plaintext (N_D) is required to perform the differential attack on the algorithm.

$$N_D \approx c / P_D$$

Where c is a constant and P_D is the probability of differential characteristics of R^{th} round for previous R-1 rounds.

For three rounds of proposed algorithm, it was found nine S-Boxes are active. Similarly, 54 S-Boxes are active for 18 rounds. **Table 4** shows the comparisons of differential cryptanalysis attack for various lightweight algorithms. Total differential probability represented as

$$P_D = (2^{-2})^{54}$$
$$= 2^{-108}$$

Differential attack complexity can be determined as follows

$$N_D \approx c / P_D$$
$$N_D = 1/2^{-108}$$
$$N_D = 2^{108}$$

Table 4. Differential attack comparisons

Algorithm/Cipher	Rounds	Number of chosen plaintext	Reference
Proposed(80 bit)	18	2^{108}	This paper
PRESENT-80	25	2^{100}	[43]
ANU-128	18	2^{94}	[43]
L-Block	15	2^{64}	[43]
FEW	27	2^{90}	[43]

Comparisons of linear and differential cryptanalytic attacks for the proposed scheme with the other lightweight algorithms shown in **Fig. 5**, which indicates the significant improvement of the proposed scheme to withstand against linear and differential cryptanalysis.

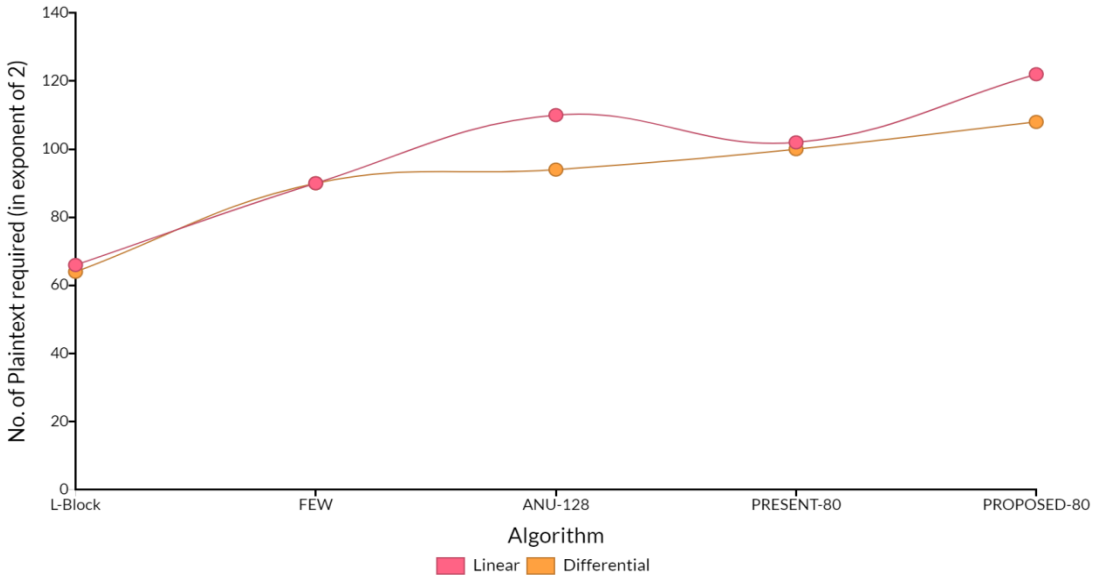

Fig. 5. Analysis of linear and differential cryptanalytic attacks

4.3 Biclique Attack

Biclique attack helps to enhance the cryptanalysis of block ciphers, which is based on meet in the middle (MITM) attack. Biclique attack is the process of building two sets of plaintext-ciphertext pairings. These bicliques selected with care to have certain characteristics that might provide insight into the encryption key. This attack's main strategy is to create bicliques on the target subcipher and utilize them to increase computation efficiency. **Table 5** shows the comparison of biclique attack with other lightweight ciphers.

The computational complexity of this attack computed as follows.

$$C_{total} = 2^{k-2d}(C_{biclique} + C_{precomp} + C_{recomp} + C_{falsepos}).$$

where,

– k = 80 and d = 4.
– $C_{biclique}$ is a computational single biclique
– $C_{precomp}$ is a computational precomputation
– C_{recomp} is a computational recomputing
– $C_{falsepos}$ is a computational false positive

Table 5. Biclique attack comparison

Target algorithm	Rounds	Data	Computational	Reference
Proposed(80 bit)	31	2^{23}	$2^{79.81}$	This paper
PRESENT-80	31	2^{23}	$2^{79.76}$	[42]

PRESENT-80	31	2^{25}	$2^{79.49}$	[44]
PRESENT-128	31	2^{23}	$2^{127.32}$	[44]
PRESENT-128	31	2^{19}	$2^{127.81}$	[42]
ANU-128	31	2^{64}	$2^{127.75}$	[43]
LED-80	48	2^{64}	$2^{79.37}$	[42]

4.4 Hamming Distance

Avalanche effect of the proposed algorithm is calculated against the inner round operation and against the ciphertext's of 1-bit, 2-bit and 4-bit change in plaintext. Hamming distance method is used for calculating the avalanche effect, while the number of bit positions at which two ciphertext or intermediate ciphertext differs are also taken into account. The avalanche effect of a bit variation in plaintext with same key in the round operation is shown in **Table 6**.

Avalanche effect over bit(s) variation in plaintext with same key against ciphertext is shown in the **Table 7** and Avalanche effect on ciphertext over different message size is shown in the **Table 8** and **Fig. 6** shows the significant improvement of the number of bit variations in the ciphertext with respect to the input bit variation.

Table 6. Avalanche effect of inner round operation

Round #	PRESENT	PRESENT Using Key-based dynamic S-Box	Round #	PRESENT	PRESENT Using Key-based dynamic S-Box
1	3	3	**17**	30	34
2	11	11	**18**	30	33
3	28	35	**19**	27	48
4	26	33	**20**	38	37
5	30	32	**21**	39	26
6	31	33	**22**	35	28
7	28	30	**23**	40	35
8	34	36	**24**	31	35
9	38	35	**25**	33	39
10	29	34	**26**	34	28
11	33	37	**27**	35	30
12	35	35	**28**	36	33
13	31	28	**29**	36	33
14	27	31	**30**	31	32
15	40	35	**31**	31	31
16	37	31	**Average**	**31.19**	**31.65**

Table 7. Avalanche effect over bit(s) variation

No. of Bit change	PRESENT	PRESENT Using Key-based dynamic S-Box
1-bit	31.67	31.91
2-bit	31.58	31.83
4-bit	31.53	31.97

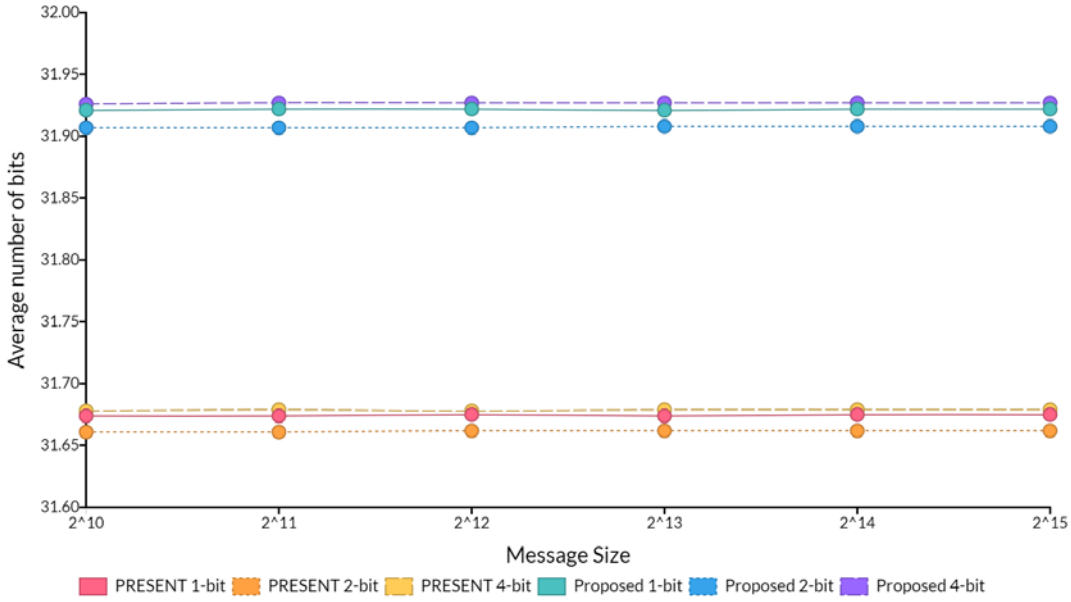

Fig. 6. Analysis of avalanche effect on ciphertext

Table 8. Avalanche effect on Ciphertext over different message size

Message Size	PRESENT			PRESENT Using Key-based dynamic S-Box		
	1-bit	2-bit	4-bit	1-bit	2-bit	4-bit
2^{10}	31.674	31.661	31.678	31.921	31.907	31.926
2^{11}	31.674	31.661	31.679	31.922	31.907	31.927
2^{12}	31.675	31.662	31.678	31.922	31.907	31.927
2^{13}	31.674	31.662	31.679	31.921	31.908	31.927
2^{14}	31.675	31.662	31.679	31.922	31.908	31.927
2^{15}	31.675	31.662	31.679	31.922	31.908	31.927

4.5 Memory utilization

Memory utilization is an important performance factor which can be used to choose an algorithm with minimal memory. This metric focuses particularly on memory utilization during execution of the algorithm to store the intermediate results. Current memory and peak memory for each size of messages are considered for memory comparison between algorithms. The additional XOR operation in the round function of Key-based dynamic S-Box slightly requires higher memory when compared to the traditional algorithm. The memory utilization

of the proposed algorithm with Key-based dynamic S-Box against the PRESENT algorithm is show in **Table 9**, and **Table 10** represents the constant utilization of memory over the PRESENT algorithm with respect to all sizes of the messages.

Table 9. Current Memory Utilization

Messages #	PRESENT	PRESENT Using Key-based dynamic S-Box	Difference	% of difference with Message size
2^{10}	58619	58803	184	0.31
2^{11}	95475	95659	184	0.19
2^{12}	169211	169395	184	0.11
2^{13}	316659	316843	184	0.06
2^{14}	611579	611763	184	0.03
2^{15}	1332467	1332651	184	0.01

Table 10. Peak Memory Utilization

Messages #	PRESENT	PRESENT Using Key-based dynamic S-Box	Difference	% of difference with Message size
2^{10}	69323	69507	184	0.27
2^{11}	111200	111384	184	0.17
2^{12}	221792	221976	184	0.08
2^{13}	442976	443160	184	0.04
2^{14}	885344	885528	184	0.02
2^{15}	1901152	1901336	184	0.01

Memory utilization analysis proposed scheme shown in **Fig. 7** shows the very marginal amount of difference around 0.01% is required over the PRESENT algorithm.

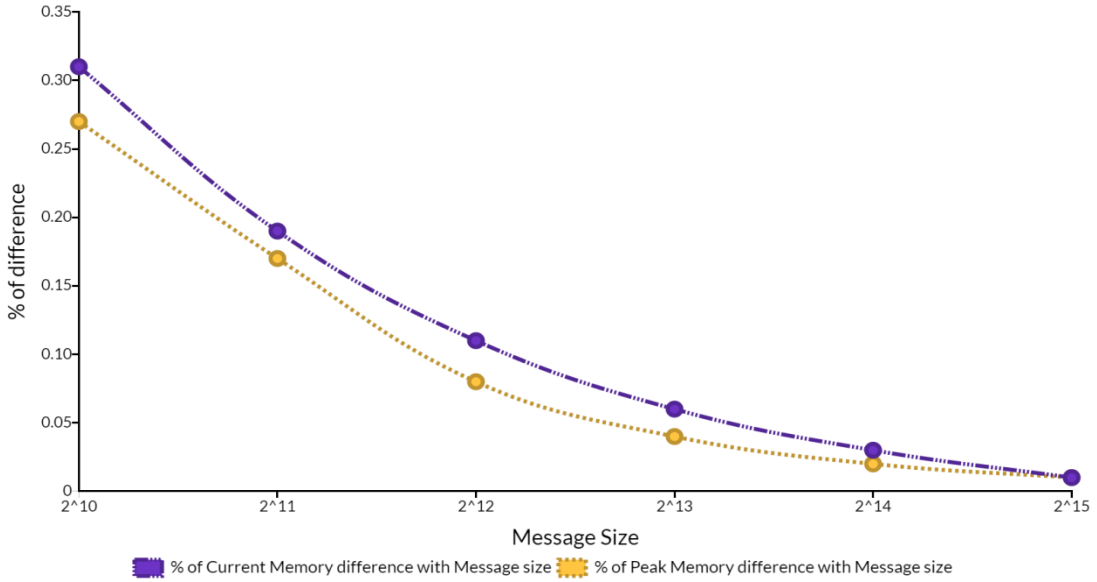

Fig. 7. Current memory and peak memory utilization comparison

4.6 Computational Cost

The energy consumed for executing the algorithm is measured from the implementation of the schemes discussed. Asymptotic analysis and implementation results are provided against the algorithms. Asymptotic analysis provides elegant result and it is abstract, so the implementation result gives the real computational cost. Measuring computational cost of an algorithm running on a typical computer is a bit tricky because of the noises from processes, other threads and caching strategies. To avoid noise, algorithm runs K times and the average CPU cycle is determined for each of the message size. The results obtained are provided in the **Table 11** which indicates the constant among all different size of messages.

Table 11. Percentage of computational cost over PRESENT

Messages #	% of Computational cost of Proposed algorithm over PRESENT
2^{10}	5.4372
2^{11}	5.5783
2^{12}	5.4905
2^{13}	5.3934
2^{14}	5.5697
2^{15}	5.4439

4.7 Encryption Time

Time consumed by the algorithm for converting plaintext to ciphertext is termed as encryption time. Encryption time is calculated against different input sizes and the average time is calculated for the algorithm. Measuring the actual CPU time spent for the encryption process is quite difficult because the CPU is frequently utilized by other processes, which naturally interrupts the memory transfer. The comparison results with the average values are provided in the **Table 12**. A tradeoff between security and performance always holds the impact in encryption algorithms. An increase in encryption time is noticed in the table trade-off with the security provided by the proposed algorithm.

Table 12. Analysis of encryption time

Messages #	PRESENT	PRESENT Using Key-based dynamic S-Box
2^{10}	1.416518	1.494072
2^{11}	2.829841	2.971066
2^{12}	5.740541	6.028379
2^{13}	11.36865	12.18960
2^{14}	22.52197	24.01752
2^{15}	45.31340	47.97777

5. Conclusion

In this research work, the key-based dynamic S-Box is implemented for PRESENT lightweight block cipher algorithm. The S-Box of PRESENT is used for the first round of the operation and for subsequent rounds, S-Box is dynamically constructed based on the round

key generated from the key schedule algorithm. Linear and differential cryptanalysis requires 2^{122} known plaintext and 2^{108} chosen plaintexts respectively. Biclique attack requires $2^{79.81}$ computational complexity and 2^{23} as data complexity to break the algorithm, which is comparatively higher than previous schemes. The inner round hamming distance of 31.65 is achieved against the 31.19 of previous implementations. Even a better avalanche effect for 1-bit, 2-bit and 4-bit with average hamming distance of 31.91, 31.83 and 31.97 respectively is achieved. The constant rate of increase in memory utilization, computation time and encryption time is also achieved even as the of message size increases. The dynamism of the S-Box breaks the myth of linearity in the proposed algorithm and helps hold the attackers from identifying the linear probability. This proposed algorithm can also be tested with the hardware implementations. In future work, it is planned to reduce the memory utilization, computation time and encryption time of the algorithm.

References

[1] W. Iqbal, H. Abbas, M. Daneshmand, B. Rauf, and Y. A. Bangash, "An In-Depth Analysis of IoT Security Requirements, Challenges, and Their Countermeasures via Software-Defined Security," *IEEE Internet Things J.*, vol. 7, no. 10, pp. 10250–10276, 2020. Article (CrossRef Link)

[2] K. Asthon, "That ' Internet of Things ' Thing," *RFiD J.*, vol. 22, no. 7, pp. 97–114, 2010, [Online]. Available: http://www.itrco.jp/libraries/RFIDjournal-That Internet of Things Thing.pdf

[3] I. K. Dutta, B. Ghosh, and M. Bayoumi, "Lightweight cryptography for internet of insecure things: A survey," in *Proc. of 2019 IEEE 9th Annu. Comput. Commun. Work. Conf. CCWC 2019*, pp. 475–481, 2019. Article (CrossRef Link)

[4] P. P., M. M., S. K.P., and M. S. Sayeed, "An Enhanced Energy Efficient Lightweight Cryptography Method for various IoT devices," *ICT Express*, vol. 7, no. 4, pp. 487–492, 2021. Article (CrossRef Link)

[5] M. Mahbub, "Comparative link-level analysis and performance estimation of channel models for IIoT (Industrial-IoT) wireless communications," *Internet of Things (Netherlands)*, vol. 12, p. 100315, 2020. Article (CrossRef Link)

[6] D. J. Bernstein, "The salsa20 family of stream ciphers," in *New Stream Cipher Designs*, pp. 84–97, 2008. Article (CrossRef Link)

[7] M. Boesgaard, M. Vesterager, T. Christensen, and E. Zenner, "The Stream Cipher Rabbit," *Fast Softw. Encryption, 10th Int. Work. FSE 2003*, vol. 4986 LNCS, pp. 69–83, 2003.

[8] H. Wu, "The stream cipher HC-128," in *New Stream Cipher Designs*, pp. 39–47, 2008. Article (CrossRef Link)

[9] C. Berbain et al., "Sosemanuk, a fast software-oriented stream cipher," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 4986 LNCS, pp. 98–118, 2008. Article (CrossRef Link)

[10] C. De Cannière, "TRIVIUM: A stream cipher construction inspired by block cipher design principles," in *Proc. of ISC 2006: Information Security*, pp. 171–186, 2006. Article (CrossRef Link)

[11] S. Babbage and M. Dodd, "The stream cipher MICKEY 2.0," *ECRYPT Stream Cipher Proj. Rep.*, pp. 1–12, 2006.

[12] M. Hell, T. Johansson, and W. Meier, "Grain: A stream cipher for constrained environments," *Int. J. Wirel. Mob. Comput.*, vol. 2, no. 1, pp. 86–93, 2007. Article (CrossRef Link)

[13] G. Hatzivasilis, K. Fysarakis, I. Papaefstathiou, and C. Manifavas, "A review of lightweight block ciphers," *J. Cryptogr. Eng.*, vol. 8, no. 2, pp. 141–184, 2018. Article (CrossRef Link)

[14] S. J. Han, H. S. Oh, and J. Park, "Improved Data Encryption Standard (DES) algorithm," *IEEE Int. Symp. Spread Spectr. Tech. Appl.*, vol. 3, pp. 1310–1314, 1996.

[15] D. J. Wheeler and R. M. Needham, "TEA, a tiny encryption algorithm BT - Fast Software Encryption," *Lect. Notes Comput. Sci.*, pp. 363–366, 1995.

[16] C. Rekha and G. N. Krishnamurthy, "An optimized key scheduling algorithm for CAST-128 using dynamic substitution S-box," *Int. J. Recent Technol. Eng.*, vol. 8, no. 3, pp. 2585–2590, 2019. Article (CrossRef Link)
[17] N. T. Courtois, "Security Evaluation of GOST 28147-89 in View of International Standardisation," *Cryptologia*, vol. 36, no. 1, pp. 2–13, 2012. Article (CrossRef Link)
[18] B. Rothke, "A look at the Advanced Encryption Standard (AES)," *Inf. Secur. Manag. Handbook, Sixth Ed.*, pp. 1151–1158, 2007.
[19] R. Oliynykov *et al.*, "A New Encryption Standard of Ukraine : The Kalyna Block Cipher," *IACR Cryptol. ePrint Arch. 2015*, pp. 1–113, 2015.
[20] O. Rabie, J. Ahmad, and D. Alghazzawi, "Modified SHARK Cipher and Duffing Map-Based Cryptosystem," *Mathematics*, vol. 10, no. 12, 2022. Article (CrossRef Link)
[21] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, and A. Poschmann, "PRESENT : An Ultra-Lightweight Block Cipher,".
[22] N. Bagheri, R. Ebrahimpour, and N. Ghaedi, "New differential fault analysis on PRESENT," *EURASIP J. Adv. Signal Process.*, vol. 2013, no. 1, 2013. Article (CrossRef Link)
[23] T. Suzaki, K. Minematsu, S. Morioka, and E. Kobayashi, "Twine: A lightweight, versatile block cipher," *ECRYPT Work. pn Light. Cryptogr. LC11*, pp. 146–169, 2011.
[24] G. Piret, T. Roche, and C. Carlet, "PICARO – A Block Cipher Allowing Efficient Higher-Order Side-Channel Resistance," in *Proc. of ACNS 2012: Applied Cryptography and Network Security*, pp. 311–328, 2012. Article (CrossRef Link)
[25] P. Zhang and W. Zhang, "Differential Cryptanalysis on Block Cipher Skinny with MILP Program," *Secur. Commun. Networks*, vol. 2018, 2018. Article (CrossRef Link)
[26] K. Jang, G. Song, H. Kim, H. Kwon, H. Kim, and H. Seo, "Efficient implementation of present and gift on quantum computers," *Appl. Sci.*, vol. 11, no. 11, 2021. Article (CrossRef Link)
[27] T. Shah, T. U. Haq, and G. Farooq, "Improved SERPENT Algorithm: Design to RGB Image Encryption Implementation," *IEEE Access*, vol. 8, pp. 52609–52621, 2020. Article (CrossRef Link)
[28] A. D. Dwivedi, "Security analysis of lightweight iot cipher: Chaskey," *Cryptography*, vol. 4, no. 3, pp. 1–10, 2020. Article (CrossRef Link)
[29] J. Chen, D. Xue, and X. Lai, "An analysis of international data encryption algorithm(IDEA) security against differential cryptanalysis," *Wuhan Univ. J. Nat. Sci.*, vol. 13, no. 6, pp. 697–701, 2008. Article (CrossRef Link)
[30] D. Lee, D. C. Kim, D. Kwon, and H. Kim, "Efficient hardware implementation of the lightweight block encryption algorithm LEA," *Sensors (Switzerland)*, vol. 14, no. 1, pp. 975–994, 2014. Article (CrossRef Link)
[31] D. Hong et al., "HIGHT: A new block cipher suitable for low-resource device," in *Proc. of International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 46–59, 2006. Article (CrossRef Link)
[32] Y. Dodis, J. Katz, J. P. Steinberger, A. Thiruvengadam, and Z. Zhang, "Provable Security of Substitution-Permutation Networks," *IACR Cryptol. ePrint Arch.*, vol. 2017, p. 16, 2017.
[33] G. Bertoni, L. Breveglieri, P. Fragneto, M. Macchetti, and S. Marchesin, "Efficient Software Implementation of AES on 32-Bit Platforms," in *Proc. of International Workshop on Cryptographic Hardware and Embedded Systems*, pp. 159–171, 2003. Article (CrossRef Link)
[34] S. S. M. Aldabbagh and I. F. T. Al Shaikhli, "Improving PRESENT lightweight algorithm," in *Proc. of 2013 Int. Conf. Adv. Comput. Sci. Appl. Technol. ACSAT 2013*, no. 1, pp. 254–258, 2013. Article (CrossRef Link)
[35] M. R. Z'aba, N. Jamil, M. E. Rusli, M. Z. Jamaludin, and A. A. M. Yasir, "I-PRESENT^{TM}: An Involutive Lightweight Block Cipher," *J. Inf. Secur.*, vol. 05, no. 03, pp. 114–122, 2014. Article (CrossRef Link)
[36] G. Bansod, N. Raval, and N. Pisharoty, "Implementation of a new lightweight encryption design for embedded security," *IEEE Trans. Inf. Forensics Secur.*, vol. 10, no. 1, pp. 142–151, 2015. Article (CrossRef Link)
[37] Z. Tang, J. Cui, H. Zhong, and M. Yu, "A random PRESENT encryption algorithm based on dynamic s-box," *Int. J. Secur. its Appl.*, vol. 10, no. 3, pp. 383–392, 2016. Article (CrossRef Link)

[38] 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,” *Secur. Commun. Networks*, vol. 2021, 2021. Article (CrossRef Link)
[39] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, “RECTANGLE: A bit-slice lightweight block cipher suitable for multiple platforms,” *Sci. China Inf. Sci.*, vol. 58, no. 12, pp. 1-15, 2015. Article (CrossRef Link)
[40] R. Chatterjee and R. Chakraborty, “A Modified Lightweight PRESENT Cipher For IoT Security,” in *Proc. of 2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, pp. 1–6, 2020. Article (CrossRef Link)
[41] B. Rashidi, “Efficient and high-throughput applicationspecific integrated circuit implementations of HIGHT and PRESENT block ciphers,” *IET Circuits, Devices Syst.*, vol. 13, no. 6, pp. 731–740, 2019. Article (CrossRef Link)
[42] K. Jeong, H. Kang, C. Lee, J. Sung, and S. Hong, “Biclique cryptanalysis of lightweight block ciphers PRESENT, piccolo and LED,” Tech. Rep. 2012/621, Cryptology ePrint Archive, 2012. Article (CrossRef Link)
[43] Bansod, G., Patil, A., Sutar, S., and Pisharoty, N., "ANU: an ultra lightweight cipher design for security in IoT," *Security Comm. Networks*, 9, 5238–5251, 2016. Article (CrossRef Link)
[44] F. Abed, C. Forler, E. List, S. Lucks and J. Wenzel, "Biclique Cryptanalysis of the PRESENT and LED Lightweight Ciphers," *Cryptology ePrint Archive, Report 2012/591*, 2012. Article (CrossRef Link)

C. A. Yogaraja, is working as an Assistant Professor(SG) in the Department of Computer Science and Engineering, Ramco Institute of Technology. He has completed his B.Tech (IT) & M.E (CSE) from Kalaignar Karunanidhi Institute of Technology, Coimbatore affiliated to Anna University. His areas of interest are network security and IoT. He has around 10 years of academic experience and published 11 papers in journal & International conferences. He is a Life Member of ISTE

Dr.K.Sheela Sobana Rani received her Ph.D degree in the year 2013 in the Faculty of Information and Communication Engineering from Anna University, Chennai. She received her M.E degree in Applied Electronics from Karunya University, Coimbatore in the year 2006 and her B.E degree in Electrical and Electronics Engineering from Sri Ramakrishna Engineering College under Bharathiar University in the year 2003. She is having 16 years of teaching experience. Currently she is working as Professor in ECE department at Karpagam College of Engineering, Coimbatore. Her research interests are Wireless Sensor Networks, Artificial Intelligence and VLSI. She has published 44 research work in international journals and 30 papers in international conferences. She is a life member of ISTE, IAENG, IACSIT, IEEE and SDIWC.