IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received October 26, 2020, accepted November 3, 2020, date of publication November 6, 2020, date of current version November 19, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.3036589

SLIM: A Lightweight Block Cipher for Internet of Health Things

BASSAM ABOUSHOSHA[1], RABIE A. RAMADAN[2,3], (Member, IEEE), ASHUTOSH DHAR DWIVEDI[4], AYMAN EL-SAYED[5], (Senior Member, IEEE), AND MOHAMED M. DESSOUKY[5,6]

[1]Department of Communication and Computer Engineering, Higher Institute of Engineering, El Shorouk Academy, Cairo 11837, Egypt
[2]Department of Computer Science and Software Engineering, University of Hail, Hail 81451, Saudi Arabia
[3]Department of Computer Engineering, Cairo University, Cairo 12678, Egypt
[4]Department of Applied Mathematics and Computer Science, Technical University of Denmark, 2800 Kongens Lyngby, Denmark
[5]Department of Computer Science and Engineering, Faculty of Electronic Engineering, Menoufia University, Menouf 32952, Egypt
[6]Department of Computer Science and Artificial Intelligence, College of Computer Science and Engineering, University of Jeddah, Jeddah 23218, Saudi Arabia

Corresponding author: Rabie A. Ramadan (rabie@rabieramadan.org)

The work of Ashutosh Dhar Dwivedi was supported by the Independent Research Fund Denmark for Technology and Production under Grant 8022-00348A.

ABSTRACT Nowadays, there is a strong demand for increasing the protection of resource-constrained devices such as Radio frequency identification (RFID) systems. Current cryptographic algorithms are sufficient for high-resource desktop computers. RFID systems are commonly used in high-security applications such as access control systems, transaction banking systems, and payment systems. The attacker attempts to mislead RFIDs for unauthorized access to services without payment or to circumvent security mechanisms by detecting a secret password. The biggest challenge in RFID systems is how to ensure successful protection against such infringements. Lightweight cryptography can provide security assurance for protecting RFID systems. This article presents a new ultra-lightweight cryptography algorithm for RFID systems called SLIM. SLIM is a 32-bit block cipher based on the Feistel structure since block ciphers are the most used cryptographic and provide very tight protection for IoT devices. The key challenge in designing a lightweight block cipher is to cope with performance, cost, and security. SLIM, like all symmetric block cipher, uses the same key for encryption and decryption. The proposed algorithm has an excellent performance in both hardware and software environments, with a limited implementation area, an acceptable cost/security for RFID systems, and an energy-efficient behaviour. SLIM has demonstrated high immunity against the most effective linear and differential cryptanalysis attacks and has a sufficient margin of defence against these attacks.

INDEX TERMS RFID, block ciphers, lightweight cryptography, feistel ciphers, cryptanalysis.

I. INTRODUCTION

Saving human life is considered the most important demands in the world. Therefore, the real-time healthcare monitoring system of patients' physical conditions is one of the significant requirements of hospital authorities. Remote Patient Monitoring System (RPMS) plays an important role in today's healthcare system and it uses different technologies, mainly IoHT, where it allows storing patient's data on the cloud. Generally, in a smart healthcare system, sensing devices for biomedical instrumentation transfer medical data of patients to a Central healthcare server Control Room (CCR) in normal cases. In the critical case, medical data is transferred remotely to the doctor using the Global System for Mobile Communication (GSM) module. The carried personal medical data may be moved through an untrusted network or may be stored in an untrusted cloud service, confidentiality of sensitive data will be exposed to cyber-attacks. Moreover, a vital issue must be considered when it transfers through resource constrained IoHT devices. Traditional algorithms aren't suitable for the nature of these devices. Existing cryptographic algorithms are mainly suitable and designed for the desktop computing era. In today's era, tiny computing devices are much popular, but on the other hand, these resource-constrained devices bring security risks. Standard cryptographic algorithms are not well suited for very

The associate editor coordinating the review of this manuscript and approving it for publication was Gautam Srivastava.

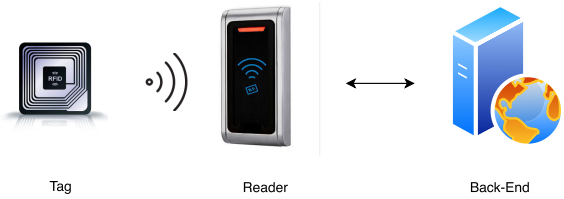

FIGURE 1. RFID components.

constrained software and hardware devices such as RFID systems. RFID systems (see Figure 1) consist of three main components; RFID tag, reader, and back-end database (server). RFID systems have several real applications in healthcare such as patient identification and tracking, equipment and asset tracking, reducing blood and drug administration errors, etc. Apart from this it has several other applications in real life such as contactless payments, electronic passports, product tracking, etc. An RFID tag is added to these products that contain important information about the product. The RFID tag or transponder is an identification technology that consists of an integrated circuit (IC) connected to an antenna. The integrated circuit is used for computation and storage purposes. The antenna provides communication between the transponder and the reader. Due to their low production costs, they can be attached to any object in the surroundings. An RFID tag can be classified based on the source of power into three categories, active tag, passive tag, and semi-passive tag. Active tags have a power source that could be a battery through which the internal circuit and antenna get the power. Passive tags do not contain any power source, and the RFID reader or transceiver provides energy for the RFID tags. There are now several different commercially available passive RFID transponders. Semi-Active tags have a battery, but that provides power only to the internal circuit, and this power is not used by the antenna. Each tag can also be divided into two types: Read-Write tags and Read-only tags. Read-write tags allow to store information as well as to modify them while on the other hand, Read-only tags allow only to read the saved information. The RFID system can be seen as the future of the barcode and soon will replace the barcode. In the context of this article, passive RFID tags are important because passive tags have very limited memory, power, and computing capability and therefore applying standard security algorithms is not feasible.

This article proposes a new ultra-lightweight cryptographic algorithm suitable for RFID restricted tags. The proposed algorithm is intended to protect the RFID data as well as the transmission. With simple operations on the RFID tag, it will be enabled to defend against different types of attacks, which make it very attractive to low-cost RFIDs. The most widely used cryptographic primitive is block cipher nowadays that provides very tight security to IoHT devices and can be applied to encryption, hashing, authentication, and random bit generations. A block cipher has versatile primitives, and a stream cipher can be achieved by running block cipher in counter mode. Also, it is easy to understand the block cipher design rather than the stream cipher. The biggest challenge to design lightweight block cipher is to cope with trade-offs between performance, cost, and security. This is not possible to provide all these three properties together to resource-constrained devices. The cipher in this article is designed carefully with respect to power and area constraints and tried to avoid any compromise with security. The cipher has the following properties:

- SLIM is a symmetric block cipher that is based on the Feistel structure. This implies that the same key is used for encryption as well as decryption.
- SLIM uses four 4×4 S-boxes that works as a non-linear component of the cipher and perform a non-linear operation on a 16-bit word.
- SLIM has very simple implementation and design, however; it has a rigidity profile against the most effective malicious cryptanalyses ‘‘linear and differential attacks".
- The cipher can be easily implemented with resource-constrained devices such as RFID and suitable for the Internet of Health Things.

The paper is organized as follows: Section II presents the literature review and the contemporary lightweight cryptographic algorithms. The proposed algorithm architecture and its functions are discussed in Section III. Implementation considerations of the SLIM algorithm are given in Section IV. The performance evaluation is discussed in Section V. Section VI presents the cryptanalysis results against linear and differential cryptanalysis and shows that cipher has enough security margin against these two powerful attacks. Finally, the paper ended with a conclusion and references.

II. RELATED WORK

In RFID devices, the storage area available for the cryptographic feature is restricted, and allowable power consumption is severely constrained. Therefore, one of the most suitable solutions for information protection in these environments is the lightweight cryptographic algorithm. This section discusses the most current state of the art in lightweight primitives or algorithms that are designed to meet the limitations of low-cost RFID. Consequentially; studying block ciphers, stream ciphers, hash functions, and random number generators.

RFID systems suffer from several malicious attacks. The risks are evolving too [24] by developing RFID technology day by day. Considering the nature of RFID Tags, it is important to implement different cryptographic protocols to reduce the impact of security and privacy issues. Modern encryption algorithms designed for high-end devices are not suitable for RFID tags because the implementation of these schemes would require ample computational power, memory, and resources [25].

Block ciphers are one of the most fundamental cryptographic primitives. The first lightweight block ciphers go

TABLE 1. Lightweight block ciphers for RFID systems.

Algorithm	Technology(μm)	Key Size	Block Size	Area(GE)	Speed(Kbps)
LBlock [1]	0.18	80	64	1320	200
PRESENT [2]	0.18	80	64	1570	200
Mcrypton [3]	0.13	64	96	2500	492.3
Piccolo [4]	0.13	80	64	616	432
LED [5]	0.18	80	64	1872	3.4
AES [6]	0.13	128	128	3100	80
TEA [7]	0.35	128	64	1984	22
HIGHT [8]	0.25	128	64	3048	188.2
KATAN [9]	0.13	80	64	1054	25.1
CLEFIA [10]	0.09	128	128	4950	355.6
KLEIN [11]	0.18	80	64	1220	207
PRINT [12]	0.18	80	48	402	3.2
KATANTAN [9]	0.13	80	64	688	25.1
SEA [13]	0.13	96	96	3758	103
DESXL [14]	0.18	184	64	2168	44.4

TABLE 2. Hash functions.

Algorithm	Technology(μm)	Output Size	Block Size	Area(GE)	Speed(Kbps)
DQuark [15]	0.18	176	160	1702	2.27
Spongent [16]	0.18	176	160	2190	17.78
Keccak [17]	0.13	200	160	1300	1.86
DM-Present [18]	0.18	64	64	1600	14.63
Armadilo-C [19]	0.18	160	160	5406	25
H-Present [20]	0.18	128	128	2330	1.45
Photon [21]	0.18	160	160	1396	2.7

TABLE 3. Stream Ciphers.

Algorithm	Technology(μm)	Interface Bits (bit/cycle)	Area(GE)	Speed(Kbps)
Enocoro [22]	0.18	8	2700	800
Grain [23]	0.13	8	800	2200
	0.13	1	100	1294
Trivium [23]	0.13	8	800	2800
	0.13	1	100	2599

back to 1997 by the appearance of the TEA block cipher [26]. Table 1 shows a comparison between some of the recent proposals for lightweight block ciphers concerning their specifications. Ciphers are compared concerning the hardware implementation area required in gate equivalent (GE), their speed in Kbps at the frequency of 100 kHz, and their CMOS technology in μm. The list includes two standard lightweight block ciphers, PRESENT [2] and CLEFIA [10].

The use of hash functions is also common in proposals for RFID systems regarding security and privacy protocols. Some of the recent lightweight hash functions and their specifications are compared in Table 2. It should be noted that the algorithms listed in the table may have different variants regarding block size, output size, or internal state size, but as a representative, only one of the variants is mentioned. There are not many proposals for lightweight stream ciphers comparing to block ciphers. Table 3 displays some of the recent proposals for lightweight stream ciphers, Grain v1 [23], Trivium [23], Enocoro [22]. The most popular of these are two hardware-oriented ciphers listed in the portfolio of the eStream project, Grain v1, and Trivium. However, the Enocoro and Trivium algorithms are the standard lightweight stream ciphers.

III. STRUCTURE OF THE PROPOSED ALGORITHM

This section presents the structure of the proposed algorithm (SLIM). It is a symmetric encryption algorithm in which both the encryption and decryption processes use the same

key (encryption and decryption keys are identical). The only difference between the two processes is that the decryption sub-keys are applied in reverse order. In SLIM, two essential design issues are taken into consideration, security, and simplicity. It achieves immunity against the exhaustive search attack by using the NIST recommendations report for key length (key length ≥ 80). SLIM accomplishes both confusion and diffusion concepts. A compact 4-bit S-box with high non-linearity properties are used to fulfil confusion. A combination of operations besides the nature of the Feistel structure is used to diffuse the data. On the other hand, simplicity is achieved in terms of the compact size of the S-box as well as the used internal simple operations. SLIM block cipher operates with 32-bit plaintext and ciphertext blocks and is controlled by an 80-bit key. The fundamental feature in the design of this algorithm is to have the smallest footprint area suitable for RFID applications. The cryptosystem structure was designed to be easily implemented in both software and hardware. Besides, SLIM consists of 32 rounds using 32 sub-keys each of 16-bit that are generated from the 80-bit key. The Basic architecture of the SLIM encryption algorithm is shown in Figure 2. As can be seen in this figure, SLIM architecture is based on a Feistel structure. The input is divided into right and left parts that go through several rounds (32-round) along with the generated sub-keys. The input could be 32-bits as well as the key size is 80-bits.

A. SINGLE ROUND PROCESSING

The more detailed architecture of SLIM can be obtained by investigating the internal structure of a single round. The first thing is that the 32-bit input is divided into two equal sixteen-bit halves known as L_i and R_i. The overall processing at each round can be summarized by Equation 1 and 2 where the right half of the input R_i and the sub-key K_i are manipulated using XOR operation. The output of the XORing operation is forwarded to a substitution box and the output of the S-boxes is forwarded to a permutation process. Finally, the output is XORed with the left half to become the right half input of the next round. The right half of the input R_i became the left half input of the next round, see Equations 1 and 2.

$$L_i = R_{i-1} \tag{1}$$

$$R_i = L_{i-1} \oplus P\left(S\left(K_i \oplus R_{i-1}\right)\right) \tag{2}$$

SLIM has been designed to mitigate the previous work limitations and their constraints such as S-boxes trapdoors, memory space, speed, lookup tables, P-boxes, key size, complexity, software, and hardware implementations.

B. SUBSTITUTION LAYER

S-boxes design is one of the most challenging tasks in cryptography. They can be considered the cornerstone of all cryptosystems because they are the only non-linear component in the most modern algorithms. Therefore, the feeble design of S-boxes produces a weakness in the overall algorithm. Biham introduced the differential cryptanalysis in 1991 [27],

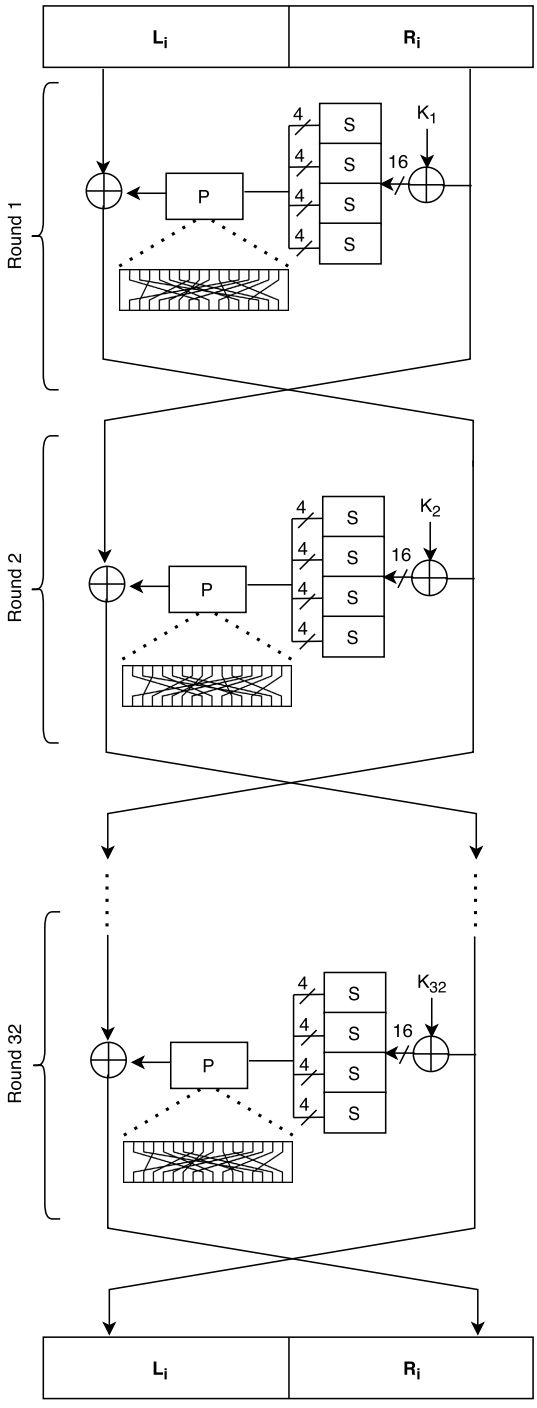

FIGURE 2. SLIM encryption.

depends on the existence of DES S-boxes trapdoors vulnerabilities. Moreover, Matsui succeeded in attacking DES again and reduced the time needed to break it using a linear cryptanalysis attack through the same trapdoors in the S-boxes. Therefore, S-boxes should be carefully designed or chosen. Thus, with a small area footprint in mind, SLIM uses S-box (see Table 4) that is used four times in parallel. S-box must be chosen to be strong enough to prevent linear and differential attacks, and at the same time, it has one of the lowest area

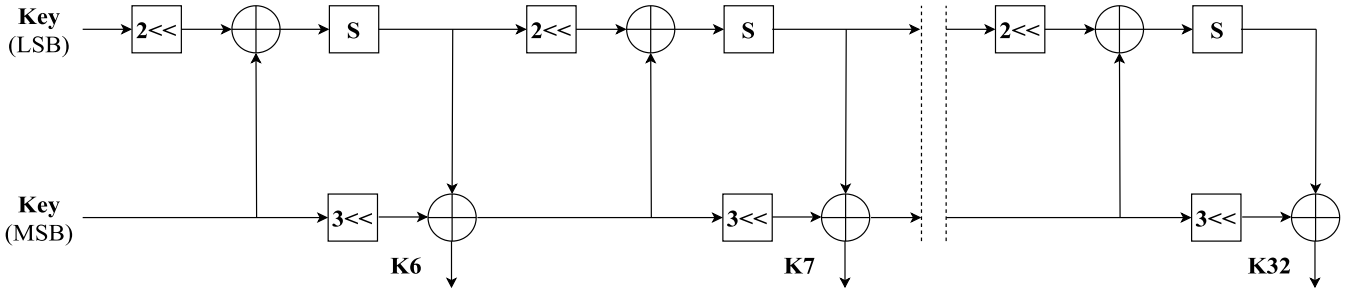

FIGURE 3. Key generation diagram of SLIM.

TABLE 4. Substitution layer of SLIM.

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

TABLE 5. Permutation layer of SLIM.

x	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
P(x)	7	13	1	8	11	14	2	5	4	10	15	0	3	6	9	12

footprints of 4-bit S-boxes [2]. Substitution layer is chosen based on linear and differential cryptanalysis performed on cipher.

C. PERMUTATION LAYER

In general, the permutation is a rearrangement process. Here, the permutation is the last phase of the SLIM function. The permutation box accepts 16-bit and permutes them using a certain rule producing a 16-bit output. In Table 5, the permutation process is given considering that there is no fixed point to avoid linearity analysis. Permutation layer is chosen based on linear and differential cryptanalysis performed on cipher.

D. KEY GENERATION

For 32 rounds and a block of 32-bit, it is required 32 sub-keys (16-bit), which are generated from the 80-bit encryption key (see Figure 3). The scheme for generation is as follows:

- The first five sub-keys, labelled K_1, K_2 . . . ., and K_5 are taken directly from the original key, with K_1 is equal to the first (least significant) 16-bits, K_2 corresponding to the next 16-bits, and so on. Then, the 80-bit key passes through a divider those results in two 40-bit quantities, labelled KeyMSB and KeyLSB; each half is thereafter treated separately.
- At each round, KeyLSB goes through a circular left shift by two bits, and then the produced output is XORed by the KeyMSB. The output of the XORing operation is forwarded to a substitution layer. The output of the S- boxes and the rotated KeyMSB (KeyMSB $\leq$ 3) are manipulated using XOR operation to produce the round sub-key.

E. SLIM DECRYPTION STRUCTURE

The process of decryption is the same as that of encryption. SLIM decryption is done using ciphertext as an input to the same SLIM structure, but the decryption sub-key is applied in the reverse order with another selection of sub-key. The reverse process can be seen in Figure 4 for both encryption and decryption of round i. The decryption round structure is similar to that of encryption, like any symmetric encryption algorithm. In Figure 4 the left-hand side shows the encryption process with the K_i subkey. The round output consists of n-bits representing the coded message. On the other hand, the decryption procedure with the same sub-key K_i is shown on the right side of Figure 4. The round function is in the opposite order. The round output is made of n-bits representing the original message.

IV. IMPLEMENTATION CONSIDERATIONS OF THE PROPOSED SLIM ALGORITHM

As shown before, SLIM is designed to facilitate both software and hardware implementation. Hardware implementation achieves high speed while the software has lower cost of implementation and more flexible with several platforms. The software implementation design principles are outlined as follows:

- **Utilization of sub-blocks:** For perfect cipher operations, they have to operate on sub-blocks that are suitable and nature for software in the form of 4, 8, 16, or 32-bits. SLIM can achieve that easily because it can be adapted to use 4 or 16 -bit sub-blocks.
- **Use simple operations:** Cipher operations should be easily programmed using addition, subtraction, shifting, complement, XORing, and so on. The elements of SLIM meet this requirement as well.

The design principles are also stated for hardware implementation as follow:

- **The similarity of encryption and decryption:** Encryption and decryption should only be different in the way that the key is used to enable the same device to use both encryption and decryption. SLIM has a structure that satisfies this requirement.
- **Using compact S-boxes:** it can save the memory space and the implementation costs. SLIM uses four 4×4 S-box.
- **Regular structure:** A regular structure for VLSI implementation should be in place for the cipher. SLIM is constructed based on only one simple modular building block that is repeated multiple times.

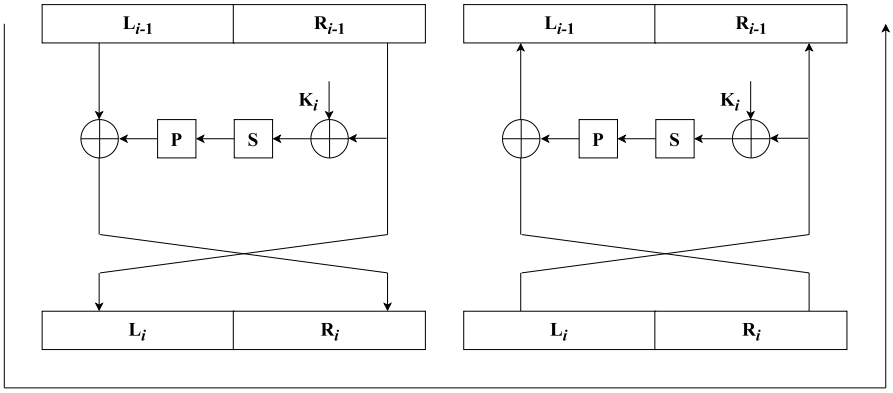

FIGURE 4. The reversing process of the SLIM algorithm for any round *i*.

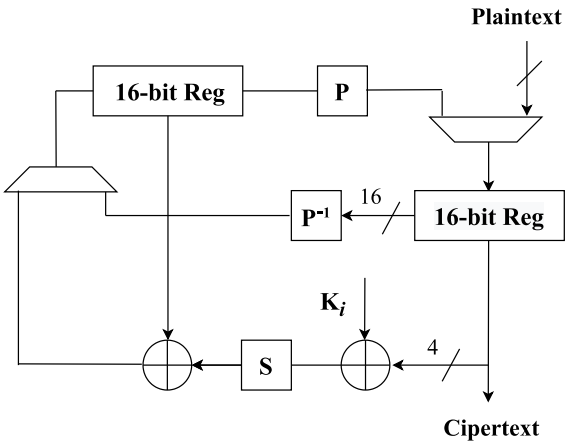

FIGURE 5. The data-path of an area-optimized version of SLIM −80.

A. HARDWARE IMPLEMENTATION OF THE PROPOSED SLIM ALGORITHM

A round-based implementation of SLIM can be done straightforwardly, while a serialized implementation poses some challenges for a hardware designer. Thus, spare the details of the former architecture and focus on the latter with a data path width of 4 bits. Most challenging is the permutation step since it permutes the whole state. Thus, it is not possible to operate on 4-bit chunks, but instead, the operation on the whole state (16-bit) had been used. In the proposed architecture illustrated in Figure 5, it takes 4 clock cycles to process all chunks of the state and to perform one round of SLIM. Then swapping the content of the registers as it is required by the Feistel structure within one clock cycle, i.e. operating on the whole state. This clock cycle had been used to perform the permutation process, but in the proposed architecture, both halves of the state had been already XORed. Thus, permuting the right halve in the previous clock cycles inversely before storing it as the new left halve. Then when the XOR sum of both halves is permuted, the final step of one round of SLIM is performed. In short, the following operations are carried out when the content of the registers is swapped:

The circuit area in GE required for the SLIM lightweight cipher is calculated using the standard ASIC library IBM 8RF (0.130 ms). Standard library values for different gates, in the above implementation, the area requirement is occupied by D-flip-flops for storing the key and the data state. To store the 80-bit key requires about 340 GE, and to store the 32-bit data state requires about 136 GE [28]. For the round structure, it consists of the following three sub-functions (XOR, S-box, and permutation).

- **Add Round Key:** In SLIM, the key addition procedure is a 4-bit XOR operation used to achieve this mixing process, which requires about (7.5 GE).
- **Substitution-layer:** In SLIM, the non-linear S-box layer in the serial implementation consists of a single 4-bit S-box which requires about (27 GE).
- **Diffusion-layer:** At the end of each SLIM round function, a permutation process is executed, which can be implemented by simple wiring and costs no area.

Finally, the additional modulo 2 of both halves requires a 4-bit XOR operation, which requires (7.5 GE). Besides, the standard round of SLIM requires two a 2-to-1 MUXes to select between the input data (plaintext) and to let the registers latch onto the result that appears at the bottom of the data-path of the previous round; a single 2-to-1 MUX cost 2.25 GE. Consequentially, the multiplexing (selecting) process requires ($2 \times 2.25 = 4.5$ GE). Therefore, Table 6 gives the entire encryption process area in gate equivalents.

The architecture of key scheduling is shown in Figure 6. For the key generator structure, it consists of the following three sub-functions (XOR, S-box, and shifting).

- **Shifting Process:** In SLIM, the right-hand side (LSBs) of the key segments is passed through the left circular shifting process by 2 steps, which not requires any gates. The left-hand side (MSBs) of the key segments is passed through the left circular shifting process by 3 steps, which not requires any gates as well.
- **Substitution-layer:** In SLIM, Single S-box is used for key scheduling to reduce the overhead in the implementation of the datapath. The non-linear S-box layer

TABLE 6. The area estimation of the hardware implementation of SLIM in GE.

Component	Gate Count
Registers	
Left Shift Register (16-bit)	$16 \times 4.25 = 68$ GE
Right Shift Register (16-bit)	$16 \times 4.25 = 68$ GE
Round Function	
XOR	$2 \times 7.5 = 15$ GE
MUX	$2 \times 2.25 = 4.5$ GE
Substitution Layer	27 GE
Total	
182.5 GE	

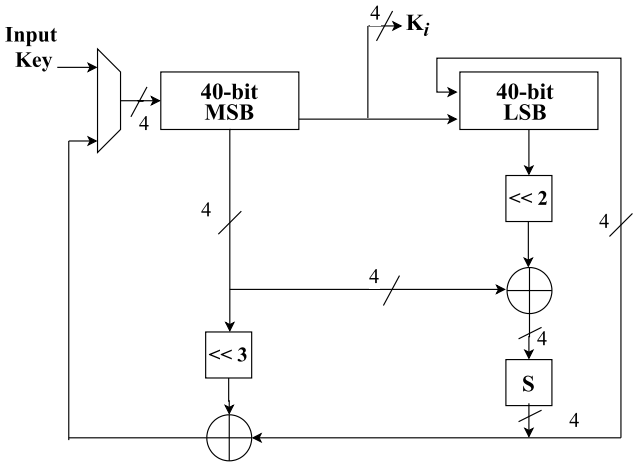

FIGURE 6. Key scheduling architecture of SLIM.

consists of a single 4-bit S-box (4-AND, 4-XOR), which requires about (13 GE) as shown in Figure 7.

- **XORing Process:** In SLIM, the key generator requires two 4-bit XOR operation used to manipulate the different inputs, which requires about ($2 \times 7.5 = 13$ GE).

Finally, a single 2-to-1 MUX is required to select between the input key and the result that appears at the bottom of the data-path of the previous round; a single 2-to-1 MUX cost 2.25 GE. Consequentially, GEs calculations for key scheduling of this architecture for its hardware implementation are shown in Table 7.

V. ANALYSIS OF THE PROPOSED SLIM ALGORITHM

In this section, the characteristics of the SLIM related to its cryptographic strength are illustrated as follow,

- **Block length:** The block length should be long enough to deter statistical analysis. This is valid for SLIM as well, where a 32-bit block is used.

TABLE 7. GE calculations for key scheduling.

Component	Gate Count
Shift Register	
Left Register (40-bit)	$40 \times 4.25 = 170$ GE
Right Register (40-bit)	$40 \times 4.25 = 170$ GE
Key Function	
XOR	$2 \times 7.5 = 15$GE
MUX	$1 \times 2..25 = 2.25$ GE
Substitution Layer	13 GE
Total	
370.25 GE	

- **Key Length:** The key length should be long enough to prevent exhaustive key searches effectively. With a length 80-bits, SLIM is secure in this area far into the future.
- **Confusion:** The ciphertext in a complicated and associated way should rely on the plaintext and key. The goal is to confuse the determination of how ciphertext statistics depend on plaintext statistics. SLIM achieves this goal by using a strong 4-bit S-box.
- **Diffusion:** Each plaintext bit should affect each ciphertext bit, and each key bit should affect each ciphertext, spreading the plaintext statistical structure over several bits of ciphertext. Through SLIM, a permutation procedure is carried out in addition to interchanging each round the two halves of the plaintext. This structure takes two (n/2)-bits values derived from the plaintext as input and two (n/2)-bits sub-keys derived from the key and produces (n/2)-bits output. Each output bit in the first round depends on each input bit from the plaintext and each bit of the sub-keys. This basic structure is repeated m times in the algorithm.

The hardware implementation result of SLIM is shown in Table 8 with a comparison of other algorithms. The figures must be taken with cares as suggested by because they depend on the type of FF, technology, library, etc. [9]. The lightweight block ciphers implemented on $0.13\mu m$ technology have been listed. Besides, Gates/Memory Bit in the table, which denotes the size (in GE) of 1-bit memory device used for the key and states, is listed as well. SPECK32-64 has a smaller implementation area, but it use only 64 bit key.

VI. CRYPTANALYSIS OF SLIM

$$\epsilon_{1,2,3..n} = 2^{n-1} \prod_{i=0}^{i=n} \epsilon_i \tag{3}$$

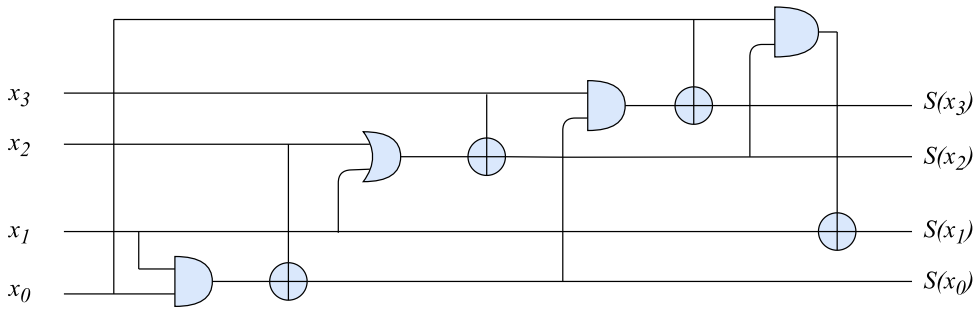

FIGURE 7. 4-bit S-box with optimal bit-slice representation [29].

TABLE 8. Hardware implementations comparison of lightweight block cipher on 0.13 mm technology.

Algorithm	Technology(μm)	Key Size	Block Size	Area(GE)	Network Structure
AES-128 [30]	0.13	128	128	3100	SPN
CLEFIA [31]	0.13	128	128	2488	GFN
PRINCE [32]	0.13	128	64	2953	SPN
LED [33]	0.13	128	64	3194	SPN
PRESENT-80 [2]	0.13	80	64	2195	SPN
NOEKEON [34]	0.13	128	128	2880	SPN
KLEIN [11]	0.13	64	64	1432	SPN
RECTANGLE [35]	0.13	80	64	1111	SPN
SPECK [28]	0.13	64	32	549	Feistel
SPECK [28]	0.13	96	48	778	Feistel
SPECK [28]	0.13	128	64	1005	Feistel
SIMON [28]	0.13	64	32	562	Feistel
SIMON [28]	0.13	96	48	796	Feistel
SIMON [28]	0.13	128	64	1026	Feistel
XTEA [26]	0.13	128	64	2521	Feistel
SEA [34]	0.13	96	96	2562	Feistel
mCrypton-64 [3]	0.13	64	128	2420	GFN
mCrypton-96 [3]	0.13	96	128	2681	GFN
mCrypton-128 [3]	0.13	128	128	2949	SPN
Piccolo-80 [4]	0.13	80	64	683	GFN
Piccolo-128 [4]	0.13	128	64	758	GFN
SLIM	0.13	80	32	553	Feistel

The most important cryptanalytic tools are linear and differential cryptanalysis. In both the cryptanalysis techniques, the attacker tries to find a linear or differential path through the various round of cipher. Finding an optimal path is the most challenging task of such attacks due to the large search space of possibilities. Dhar *et al.* [36] proposed a Nested search based heuristic technique to find the differential path in several block ciphers. However, applying such a method directly to any cipher is not like plug and play but requires a strong coding skill to modify cipher in the required form. Such methods can be applied to find the linear or differential path in most of the block ciphers. The code is written in python language and publicly available at GitHub [37] click here.

A. LINEAR CRYPTANALYSIS

Matsui [38] in 1993 introduced the concept of Linear cryptanalysis and since that it becomes the most powerful technique to analyse any cipher. The basic idea is to generate a linear approximation that describes the relation between plaintext (input) and ciphertext (output) bits. Such a relationship should hold with probability 0.5 (bias $\epsilon = 0$) for s secure cipher. If an attacker can find a relationship where bias $\epsilon \neq 0$, then it could be converted to some attack. To find linear paths in cipher, firstly we generate the linear approximation table of given S-box. In each round, the Nested algorithm tried to find the best bias from the approximation table and return the optimal path. Total bias is calculated using a formula introduced by Matsui [38]. We could only find the differential

TABLE 9. Linear trails for SLIM Cipher.

Round	Block1	Block2	Bias	Active S-box
1	0x0000	0x8000	2	1
2	0x8000	0x0420	4	2
3	0x0420	0x6000	1	1
4	0x6000	0x8400	2	2
5	0x8400	0x8639	5	4
6	0x8639	0xec0f	5	3
7	0xec0f	0x5c01	4	3
8	0x5c01	0xd070	2	2
9	0xd070	0x0005	2	1
10	0x0005	0x0002	2	1
11	0x0002	0x0040	2	1
Total Bias:			32	21

TABLE 10. Differential trails for SLIM Cipher.

Round	Block1	Block2	$\log_2 p$	Active S-box
1	0x0000	0x8000	3	1
2	0x8000	0x8001	5	2
3	0x8001	0x1003	5	2
4	0x1003	0x1000	2	1
5	0x1000	0x9002	5	2
6	0x9002	0x0060	3	1
7	0x0060	0x0006	4	2
Total Probability:			27	11

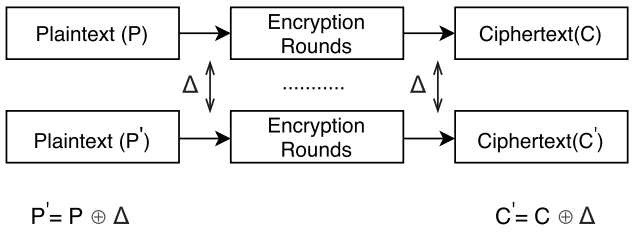

FIGURE 8. Difference propagation of plaintext pair.

path (see Table 9) up to 11 rounds and therefore cipher is secured for the full number of rounds.

B. DIFFERENTIAL CRYPTANALYSIS

Differential cryptanalysis is one of the most powerful tools to analyse any cipher that was introduced by Biham and Shamir [27]. Differential cryptanalysis works in a chosen plaintext-ciphertext scenario where an intruder can access the encrypted text after selecting any plaintext as input to the cipher. To do the differential cryptanalysis, the attacker can take a pair of plaintexts, and these two plaintexts are related to each other by a constant difference. This difference could be 2^n modular addition or XOR operation (see Figure 8).

A differential path (also trail or characteristic) is a sequence of differences through several rounds of cipher encryption. When searching for the differential path, attackers mainly care about non-linear components (S-box in this case). There could be many output differences (through S-box) for a particular input difference. Therefore, the attacker creates a difference distribution table of S-box that gives the probability of certain output over an output. In this article, to find a differential path in SLIM, we used the Nested tree search heuristic approach proposed by Dwivedi and Srivastava [39] and Dwivedi [40]. Firstly, we draw a difference distribution table based on S-box specifications and using that table we tried to generate a differential path. We could only find the differential path (see Table 10) up to 7 rounds and therefore cipher is secured for the full number of rounds.

VII. CONCLUSION

This article proposed a new ultra-lightweight cryptography algorithm for RFID systems called SLIM. RFID systems are suffering from a variety of malicious attacks. The key challenge in RFID systems is how to ensure successful defense against these attacks. Modern encryption algorithms designed for high-end devices are not appropriate for RFID systems, because the implementation of these schemes will require high computational power, memory, and resources. The usable cryptographic storage area is limited in RFID systems and the permissible power consumption is severely constrained. Lightweight cryptographic algorithms are one of the most appropriate solutions for securing information in these environments. The proposed ultra-lightweight cryptography SLIM algorithm is suitable for use in constraint RFID

systems. SLIM is a block cipher based on a 32-bit block size Feistel structure. SLIM uses a long key length equal to 80-bits to avoid exhaustive key searches. SLIM uses a strong four 4 4 substitution boxes to determine how ciphertext statistics depend on plaintext statistics. The proposed algorithm has an excellent performance in both hardware and software environments, with a limited implementation area of only 553 GE, an acceptable cost/security for RFID systems, and an energy-efficient behavior. SLIM has proved high immunity against the most effective linear and differential cryptanalysis attacks and has a sufficient protection margin against these attacks. The proposed algorithm is suitable for wireless networks, especially Wireless Sensor Networks (WSNs) and IoT applications, where data messages are typically within a few bytes range. SLIM proved to be highly efficient compared to other existing and implemented algorithms. The future work involves the implantation of SLIM on a healthcare IoT framework to analyze it for sensitive applications.

REFERENCES

[1] W. Wu and L. Zhang, "Lblock: A lightweight block cipher," in *Proc. 9th Int. Conf. Appl. Cryptogr. Netw. Secur.*, in Lecture Notes in Computer Science, Nerja, Spain, vol. 6715, J. López and G. Tsudik, Eds., Jun. 2011, pp. 327–344.

[2] A. Bogdanov, L. R. Knudsen, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, Y. Seurin, and C. Vikkelsoe, "PRESENT: An ultra-lightweight block cipher," in *Proc. 9th Int. Workshop*, in Lecture Notes in Computer Science, Vienna, Austria, vol. 4727, P. Paillier and I. Verbauwhede, Eds. Vienna, Austria: Springer, Sep. 2007, pp. 450–466.

[3] C. H. Lim and T. Korkishko, "Mcrypton - A lightweight block cipher for security of low-cost RFID tags and sensors," in *Proc. 6th Int. Workshop*, in Lecture Notes in Computer Science, Jeju Island, South Korea, vol. 3786, J. Song, T. Kwon, and M. Yung, Eds. Springer, Aug. 2005, pp. 243–258,

[4] K. Shibutani, T. Isobe, H. Hiwatari, A. Mitsuda, T. Akishita, and T. Shirai, "Piccolo: An ultra-lightweight blockcipher," in *Proc. 13th Int. Workshop*, in Lecture Notes in Computer Science, Nara, Japan, vol. 6917, B. Preneel and T. Takagi, Eds. Springer, Sep. 2011, pp. 342–357.

[5] L. Batina, A. Das, B. Ege, E. B. Kavun, N. Mentens, C. Paar, I. Verbauwhede, and T. Yalçin, "Dietary recommendations for lightweight block ciphers: Power, energy and area analysis of recently developed architectures," in *Proc. 9th Int. Workshop*, in Lecture Notes in Computer Science, Graz, Austria, vol. 8262, M. Hutter and J. Schmidt, Eds. Graz, Austria: Springer, Jul. 2013, pp. 103–112.

[6] J. Daemen and V. Rijmen, *The Design Rijndael: AES—The Advanced Encryption Standard* (Information Security and Cryptography). Cham, Switzerland: Springer, 2002.

[7] D. J. Wheeler and R. M. Needham, "Tea, A tiny encryption algorithm," in *Proc. 2nd Int. Workshop*, in Lecture Notes in Computer Science, Leuven, Belgium, vol. 1008, B. Preneel, Ed. Leuven, Belgium: Springer, Dec. 1994, pp. 363–366.

[8] D. Hong, J. Sung, S. Hong, J. Lim, S. Lee, B. Koo, C. Lee, D. Chang, J. Lee, K. Jeong, H. Kim, J. Kim, and S. Chee, "HIGHT: A new block cipher suitable for low-resource device," in *Proc. 8th Int. Workshop*, in Lecture Notes in Computer Science, Yokohama, Japan, vol. 4249, L. Goubin and M. Matsui, Eds. Yokohama, Japan: Springer, Oct. 2006, pp. 46–59.

[9] C. D. Cannière, O. Dunkelman, and M. Knezevic, "KATAN and KTANTAN - A family of small and efficient hardware-oriented block ciphers," in *Proc. 11th Int. Workshop*, in Lecture Notes in Computer Science, vol. 5747, C. Clavier and K. Gaj, Eds. Lausanne, Switzerland: Springer, Sep. 2009, pp. 272–288.

[10] T. Sugawara, N. Homma, T. Aoki, and A. Satoh, "High-performance ASIC implementations of the 128-bit block cipher CLEFIA," in *Proc. IEEE Int. Symp. Circuits Syst.*, Seattle, WA, USA, May 2008, pp. 2925–2928.

[11] Z. Gong, S. Nikova, and Y. W. Law, "KLEIN: A new family of lightweight block ciphers," in *Proc. 7th Int. Workshop*, in Lecture Notes in Computer Science, Amherst, MA, USA, vol. 7055, A. Juels and C. Paar, Eds. Amherst, MA, USA: Springer, 2011, pp. 1–18.

[12] L. R. Knudsen, G. Leander, A. Poschmann, and M. J. B. Robshaw, "Printcipher: A block cipher for ic-printing," in *Proc. 12th Int. Workshop* in Lecture Notes in Computer Science, Santa Barbara, CA, USA, vol. 6225, S. Mangard and F. Standaert, Eds. Santa Barbara, CA, USA,: Springer, Aug. 2010, pp. 16–32.

[13] F. Standaert, G. Piret, N. Gershenfeld, and J. Quisquater, "SEA: A scalable encryption algorithm for small embedded applications," in *Proc. Workshop RFIP Light Weight Crypto*, in Lecture Notes in Computer Science, Tarragona, Spain, vol. 3928, J. Domingo-Ferrer, J. Posegga, and D. Schreckling, Eds. Graz, Austria: Springer. Apr. 2006, pp. 222–236.

[14] G. Leander, C. Paar, A. Poschmann, and K. Schramm, "New lightweight DES variants," in *Proc. 14th Int. Workshop*, in Lecture Notes in Computer Science, Luxembourg, Luxembourg, vol. 4593, A. Biryukov, Ed. Luxembourg City, Luxembourg: Springer, Mar. 2007, pp. 196–210.

[15] J.-P. Aumasson, L. Henzen, W. Meier, and M. Naya-Plasencia, "Quark: A lightweight hash," *J. Cryptol.*, vol. 26, no. 2, pp. 313–339, Apr. 2013.

[16] A. Bogdanov, M. Knezevic, G. Leander, D. Toz, K. Varici, and I. Verbauwhede, "Spongent: A lightweight hash function," in *Proc. 13th Int. Workshop Cryptograph. Hardw. Embedded Syst.*, in Lecture Notes in Computer Science, vol. 6917, B. Preneel and T. Takagi, Eds. Nara, Japan: Springer, Sep. 2011, pp. 312–325.

[17] G. Bertoni, J. Daemen, M. Peeters, and G. Van Assche. (2012). *Keccak Sponge Function Family Main Document*. [Online]. Available: http://keccak.noekeon.org/Keccak-main-2.1.pdf

[18] A. Y. Poschmann, "Lightweight cryptography: Cryptographic engineering for a pervasive world," Ph.D. dissertation, Dept. Elect. Eng. Inf. Technol., Ruhr Univ., Bochum, Germany, 2009.

[19] S. Badel, N. Dagtekin, K. Ouafi, N. Reffé, P. Sepehrdad, P. Susil, and S. Vaudenay, "ARMADILLO: A multi-purpose cryptographic primitive dedicated to hardware," in *Proc. 12th Int. Workshop*, in Lecture Notes in Computer Science, vol. 6225, S. Mangard and F. Standaert, Eds. Santa Barbara, CA, USA: Springer, Aug. 2010, pp. 398–412.

[20] A. Bogdanov, G. Leander, C. Paar, A. Poschmann, M. J. B. Robshaw, and Y. Seurin, "Hash functions and RFID tags: Mind the gap," in *Proc. 10th Int. Workshop*, in Lecture Notes in Computer Science, vol. 5154, E. Oswald and P. Rohatgi, Eds. Washington, DC, USA: Springer, Aug. 2008, pp. 283–299.

[21] J. Guo, T. Peyrin, and A. Poschmann, "The PHOTON family of lightweight hash functions," in *Proc. 31st Annu. Int. Cryptol. Conf. (CRYPTO)*, in Lecture Notes in Computer Science, vol. 6841, P. Rogaway, Ed. Santa Barbara, CA, USA: Springer, Aug. 2011, pp. 222–239.

[22] D. Watanabe, T. Owada, K. Okamoto, Y. Igarashi, and T. Kaneko, "Update on enocoro stream cipher," in *Proc. Int. Symp. Inf. Theory Its Appl.*, Taichung, Taiwan, Oct. 2010, pp. 778–783.

[23] M. Feldhofer and J. Wolkerstorfer, *Hardware Implementation of Symmetric Algorithms for RFID Security*. Cham, Switzerland: Springer, 2008, pp. 373–415.

[24] S. L. Garfinkel, A. Juels, and R. Pappu, "RFID privacy: An overview of problems and proposed solutions," *IEEE Security Privacy*, vol. 3, no. 3, pp. 34–43, May 2005.

[25] A. Poschmann, G. Leander, K. Schramm, and C. Paar, "New light-weight crypto algorithms for RFID," in *Proc. IEEE Int. Symp. Circuits Syst.*, New Orleans, LO, USA, May 2007, pp. 1843–1846.

[26] R. M. Needham and D. J. Wheeler. *Tea Extensions*. Accessed: Aug. 30, 2020. [Online]. Available: http://www.cix.co.uk/~klockstone/xtea.pdf

[27] E. Biham and A. Shamir, "Differential cryptanalysis of Des.like cryptosystems," *J. Cryptol.*, vol. 4, no. 1, pp. 3–72, Jan. 1991.

[28] R. Beaulieu, D. Shors, J. Smith, S. Treatman-Clark, B. Weeks, and L. Wingers, "The SIMON and SPECK lightweight block ciphers," in *Proc. 52nd Annu. Design Autom. Conf.*, 2015, p. 175.

[29] V. Grosso, G. Leurent, F. Standaert, and K. Varici, "Ls-designs: Bitslice encryption for efficient masked software implementations," in *Proc. 21st Int. Workshop*, in Lecture Notes in Computer Science, vol. 8540, C. Cid and C. Rechberger, Eds. London, U.K.: Springer, Mar. 2014 pp. 18–37.

[30] P. Hämäläinen, T. Alho, M. Hännikäinen, and T. D. Hämäläinen, "Design and implementation of low-area and low-power AES encryption hardware core," in *Proc. 9th Euromicro Conf. Digit. System Des.*, Dubrovnik, Croatia, Aug./Sep. 2006, pp. 577–583.

[31] T. Akishita and H. Hiwatari, "Very compact hardware implementations of the blockcipher CLEFIA," in *Proc. 18th Int. Workshop*, in Lecture Notes in Computer Science, vol. 7118, A. Miri and S. Vaudenay, Eds. Toronto, ON, Canada: Springer, Aug. 2011, pp. 278–292.

[32] J. Borghoff, A. Canteaut, T. Güneysu, E. B. Kavun, M. Knezevic, L. R. Knudsen, G. Leander, V. Nikov, C. Paar, C. Rechberger, P. Rombouts, S. S. Thomsen, and T. Yalçin, "PRINCE—A low-latency block cipher for pervasive computing applications (full version)," *Proc. IACR*, 2012, p. 529.
[33] J. Guo, T. Peyrin, A. Poschmann, and M. J. B. Robshaw, "The LED block cipher," in *Proc. 13th Int. Workshop*, in Lecture Notes in Computer Science, vol. 6917, B. Preneel and T. Takagi, Eds. Nara, Japan: Springer, Sep./Oct.2011, pp. 326–341,.
[34] T. Plos, C. Dobraunig, M. Hofinger, A. Oprisnik, C. Wiesmeier, and J. Wiesmeier, "Compact hardware implementations of the block ciphers mcrypton, noekeon, and SEA," in *Proc. 13th Int. Conf. Cryptol.* in Lecture Notes in Computer Science, Kolkata, India, vol. 7668, S. D. Galbraith and M. Nandi, Eds. New Delhi, India: Springer, Dec. 2012, pp. 358–377.
[35] W. Zhang, Z. Bao, D. Lin, V. Rijmen, B. Yang, and I. Verbauwhede, "RECTANGLE: A bit-slice ultra-lightweight block cipher suitable for multiple platforms," in *Proc. IACR*, 2014, p. 84.
[36] D. A. Dhar, P. Morawiecki, and S. Wājtowicz, "Finding differential paths in arx ciphers through nested monte-carlo search," *Int. J. Electron. Telecommun.*, vol. 64, no. 2, p. 15, Dec. 2018.
[37] A. D. Dwivedi. (2020). *Slim: An Ultra-Lightweight Block Cipher Algorithm Suitable for RFID Systems*. [Online]. Available: https://github.com/ashudhar7/Agile
[38] M. Matsui, "Linear cryptanalysis method for DES cipher," in *Proc. Workshop Theory Appl. Cryptograph. Techn.*, Lofthus, Norway, May 1993, pp. 386–397.
[39] A. D. Dwivedi and G. Srivastava, "Differential cryptanalysis of round-reduced LEA," *IEEE Access*, vol. 6, pp. 79105–79113, 2018.
[40] A. D. Dwivedi, "Security analysis of lightweight IoT cipher: Chaskey," *Cryptography*, vol. 4, no. 3, p. 22, Aug. 2020.

BASSAM ABOUSHOSHA received the bachelor's degree (Hons.) in computer engineering from the Higher Institute of Engineering, El-Shorouk Academy, Cairo, Egypt, in 2009, and the Master of Science degree in computer engineering from the Arab Academy for Science, Technology, and Maritime Transport (AASTMT), Cairo, in 2015. He is currently pursuing the Ph.D. degree in computer engineering with Menoufia University, Menouf, Egypt. His research interests include cyber security, cryptology, the Internet of Things (IoT), wireless sensor networks (WSNs), control systems, digital design, and smart city design and planning.

RABIE A. RAMADAN (Member, IEEE) received the Ph.D. degree in computer engineering from Cairo University and Southern Methodist University (SMU), Dallas, TX, USA, in 2007. He is currently the Chair of Computer Engineering Department, Hail University, Hail, Saudi Arabia. He is also an Associate Professor with Cairo University, Cairo, Egypt, and Hail University. He is the author of more than 125 articles in the field of big data, cloud computing, the IoT, and computational intelligence. He is also the Director of the Industrial Partnership Program (CISCO, Oracle, and Microsoft), College of Computer Science and Engineering, Hail University. He is also the Founder of FabLab and the Center of Programming and Applications, Hail University, sponsored by the Saudi Arabia Ministry of Education. He is also the Founder of TripleTech Expert House sponsored by Hail University. He co-led the Second and First Place Team in the RoboCup 2009 and 2010, respectively. He has served as the General Chair, the Program Committee Chair, and a TPC for many of the conferences and journals, including the Web of Science and IEEE TRANSACTIONS journals. He has served as the Co-Chair of the International Conference on Recent Advances in Computer Systems (RACS-2015) and the 2nd National Computing Colleges Conference (NC3 2017) held at Hail University. He is also the General Chair of the International Conference on New Computer Science and Engineering Trends (NCSET2020). He was the Chair of the ACM Programming Competition in Saudi Arabia associated with NC3 2017 conference. He is also a Co-Founder of IEEE Computational Intelligence, Egypt Chapter. The chapter was awarded by the IEEE Computational Intelligence Society (CIS), the 2011 IEEE CIS Outstanding Chapter Award. He is also serving as the Chapter Secretary.

ASHUTOSH DHAR DWIVEDI received the Ph.D. degree from the Polish Academy of Sciences, Warsaw, Poland. He has worked as a full-time Visiting Researcher with the University of Waterloo, ON, Canada, a Research Associate with Brandon University, MB, Canada, a Research Employee with the Polish Academy of Sciences, and a Research Scholar with the Military University of Technology, Warsaw. He is currently a Postdoctoral Researcher with the Department of Applied Mathematics and Computer Science, Cyber Security Section, Technical University of Denmark. His research interests include symmetric key cryptography and blockchain. He has made contributions to multiple journal and conference articles. He is also a member of reviewer board of three journals and reviewed articles from more than 24 different journals and several other conferences and workshops. Heidelberg Laureate Forum, Germany, has selected him among top 100 young researchers all over the world in Computer Science to participate in the HLF-2019 event.

AYMAN EL-SAYED (Senior Member, IEEE) received the B.Sc. degree in computer science and engineering and the master's degree in computer networks from Menoufia University, Egypt, in 1994 and 2000, respectively, and the Ph.D. degree in computer network from the Institute National De Polytechnique De Grenoble (INPG), France, in 2004. He is currently a Professor and the Dean of the Faculty of Electronic Engineering. He is an approved Supervisor for M.Sc. and Ph.D. programs in various Universities. He has completed various projects in government and private organization. He has published more than 105 research articles in international journals and two books about OSPF protocol and multicast protocols. He is also serving as an Editorial Board Member in various international journals and conferences.

MOHAMED M. DESSOUKY received the B.Sc. (Hons.), M.Sc., and Ph.D. degrees from the Faculty of Electronic Engineering (FEE), Menuofia University, Egypt, in 2006, 2011, and 2016, respectively. Since 2016, he has been a Teaching Staff Member with the Department of Computer Science and Engineering, FEE. He is currently an Assistant Professor. He is a Supervisor for M.Sc. and Ph.D. students. He has published more than 30 research papers in international journals or conferences, and he has published a book chapter about Alzheimer's disease. He is the CISCO Academy Curriculum Lead for more than ten years.