# A Robust Lightweight Algorithm for Securing Data in Internet of Things Networks

**Abdulrazzaq H. A. Al-Ahdal, Galal A. AL-Rummana, and Nilesh K. Deshmukh**

**Abstract** One of the modern technologies that link millions of different devices together, is the technology called Internet of things. The amount of data exchanged between these devices is very large. Therefore, it requires high protection for that data. Also, those devices are small size and limited resources. Therefore, conventional cryptography will take long process on the internet of things due to complex mathematical operations and an increase in the number of rounds, which leads to energy resource consumption for devices. In this paper, a lightweight cryptographic algorithm is suggested. The algorithm uses simple mathematical operations (XOR, XNOR, shifting, swapping). The algorithm works on a combination of Feistel and SP architectural methods to increase the complexity of the encryption. The algorithm has been compared to other algorithms in terms of structure, security, and flexibility.

**Keywords** IoT security · Khazad · Wireless sensor network WSN · Feistel · SP · Data encryption algorithm

## 1 Introduction

Internet of Things (IoT) technology has become an integral part of many activities in our daily life. Therefore, this modern technology has turned into a debate in the field of research and applications. Many different fields such as agriculture, industry,

A. H. A. Al-Ahdal (✉) · G. A. AL-Rummana · N. K. Deshmukh
School of Computational Sciences, S.R.T.M. University, Nanded, India
e-mail: alahdal201211@gmail.com

G. A. AL-Rummana
e-mail: Galal300z@gmail.com

N. K. Deshmukh
e-mail: nileshkd.srt@gmsail.com

A. H. A. Al-Ahdal
Faculty of Computer Science & Engineering, Hodeidah University, Al Hudaydah, Yemen

P. Karuppusamy et al. (eds.), *Sustainable Communication Networks and Application*,
Lecture Notes on Data Engineering and Communications Technologies 55,
https://doi.org/10.1007/978-981-15-8677-4_41

medicine, and smart cities use this modern technology on a large scale in artificial intelligence, sensors and RFID, depending on Internet technologies [1].

These networks use very limited resources, low-energy, and very small devices that communicate with each other to transfer information between them. Therefore, the amount of information exchanged between devices is large and needs to be secured the exchange of information between devices is called conventional cryptography.

Conventional cryptography algorithms are not suitable for devices with limited resources on the Internet of things because they need long operation (process) and do not make a trade-off among memory, security, cost, power, and performance. Therefore, lightweight cryptography is the new direction for the Internet of Things because it concerns memory, security, cost, power, and performance. There are strict limitations and requirements in applications using IoT. On that basis, the requirements must be met when designing a lightweight cryptography algorithm [2].

In this paper, a lightweight algorithm is proposed for devices with resource-limited in the Internet of Things. It is described in Sect. 2. It is compared to various aspects with many algorithms in Sect. 3.

## 2 Proposed Algorithm

The proposed algorithm for resource-limited devices is designed to implement lightweight encryption in the Internet of Things. Some encryption blocks have the advantages of replacing, switching, and confusing to change the text. As an established fact, SP network architecture is used by AES [3], PRESENT [4] and Square [5] blocks. In addition, some of the other advantages of the encryption process are the decryption itself. While SF [6], Blowfish [7] and DES [8] blocks are used by Feistel network architecture. The proposed algorithm is a combination of Feistel and SP networks to obtain the security required to develop lightweight encryption algorithms that work on the Internet of Things.

In any cipher algorithm, cryptography consists of several rounds, each round consisting of mathematical operations to create confusion and diffusion. An increase in the number of rounds increases security but leads to the energy consumption of devices [9]. The optimum number of rounds is from 10 to 12 rounds, to be a robust algorithm. But the proposed algorithm is used to improve power on devices that operate on the Internet of Things. So it works in six rounds, each round contains logical operations working with 4 bits of data. This creates confusion for attackers and further complicates the encryption process, and this will be further discussed in Sect. 3. In general, the proposed algorithm has three main blocks:

- Key Expansion Block
- Encryption Block
- Decryption Block.

In the following subsections, these blocks will be further clarified in detail, and some of the essential notes followed in the interpretation are presented in Table 1.

**Table 1** Notations

| Notation | Function |
|---|---|
| $\oplus$ | XOR |
| $\odot$ | XNOR |
| ‖,╫ | Concatenation |

## *2.1 Key Expansion Block*

The key is the main component of the algorithm (encryption/decryption). The size of the encryption key is very important for security. That is how (key size) becomes a major obstacle for being known by the attacker. Accordingly, confusion and diffusion (key generation) reduces the possibility of weakening the key, increase the security, increase the complexity of the encryption, and lack of knowledge of the key by the attackers.

The proposed algorithm uses 80 bits cipher key (Kc) to encryption a 64 block of data. From the Kc, two keys will be produced. The first will be sent to open the encryption while the second will become an entry key for the keys expansion process as explained in sections a and b. Moreover, the key will generate six unique keys. In order to encrypt the block with six keys. These operations (confusion and diffusion) improve the strength of security and decrease the opportunity of attacking.

a. **Before Encryption Process**
In this process, two keys will be created from the key entered by the user. The first key is the key that has been sent by the user for the process of opening the encryption as it will be explained in section b. The other key will be completely different from the key entered by the user and also will be the encrypted key for the data as in Fig. 1, and it will be discussed below:

- The 80-bit cipher key (Kc), It is divided into segment R0(40 bit) and segment L0(40 bit)
- For $i = 1$ to 40
  If Ri = 0 then.

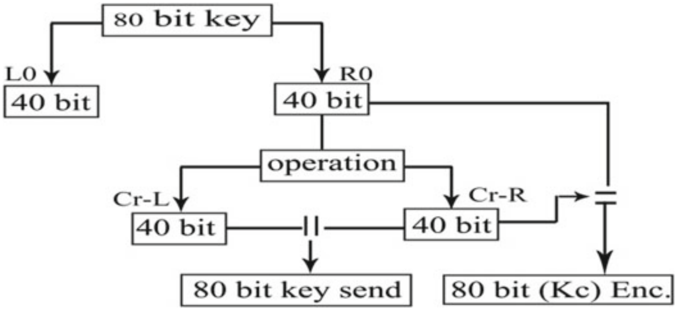

**Fig. 1** Before encryption process

Cr-R[i] = 1, Cr–L[i] = 1.
Else.
Cr-R[i] = 0, Cr–L[i] = 1.

- To combine Cr-L, Cr-R to key send(Send Key 80 bit)
- To combine R0, Cr-R to Kc(encryption Key 80 bit).

b. Key Expansion Process

Key expansion of the component key is done in Section a. Therefore, it will be explained as in Fig. 2.

- The 80-bit cipher key (Kc), It is divided into 20 segments, each segment of 4-bits.
- The f -function used 4 segments, each segment 4 bit (16 bit) as illustrated in Fig. 2. Substitution can generate for cipher key (Kc) by f-function as shown in Eq. (1).

$$Kb_i f = \|_{j=1}^{5} K_{c4(j-1)+i} \tag{1}$$

where $i = 5$;

- $Ka_iF$ is output from Eq. (2)

$$Ka_i f = f(b_i f) \tag{2}$$

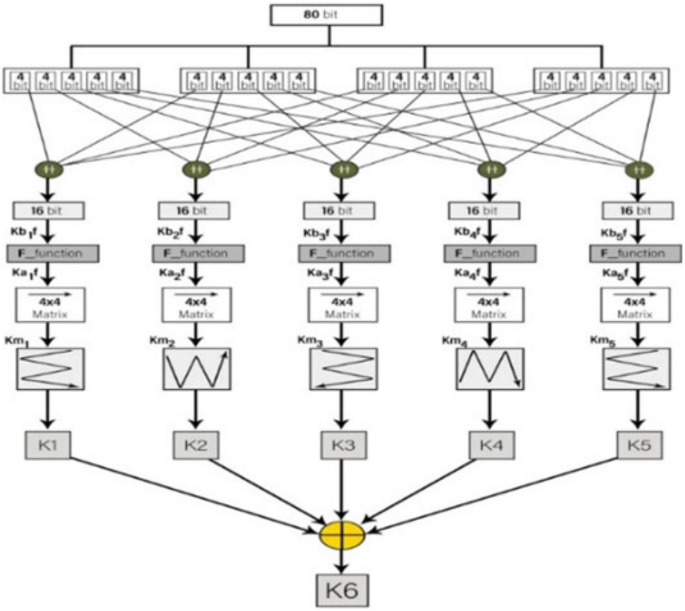

**Fig. 2** Key expansion process

- $f$ : $f$—function [10]. Is perform confusion and diffusion transformations linear and non-linear by comprised of P and Q tables are shown in the Tables 2 and 3 as illustrated in Fig. 3.
- Output, each f-function is generated one matrix 4*4 from (16 bit). Therefore it generates five arrays named (Km1, Km2, Km3, Km4, and Km5).
- The arrays generated K1, K2, k3, K4 and K5 are the round keys. The result of the rotation of the arrays is Km1, Km2, Km3, Km4, and Km5, respectively.
- To generate the sixth key, XOR works between the five keys generated by Eq. 3.

$$k6 = \bigoplus_{i=1}^{5} k_i \tag{3}$$

**Table 2** P Table

| Kci | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| P (Kci) | 3 | F | E | 0 | 5 | 4 | B | C | D | A | 9 | 6 | 7 | 8 | 2 | 1 |

**Table 3** Q Table

| Kci | 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| Q (Kci) | 9 | E | 5 | 6 | A | 2 | 3 | C | F | 0 | 4 | D | 7 | B | 1 | 8 |

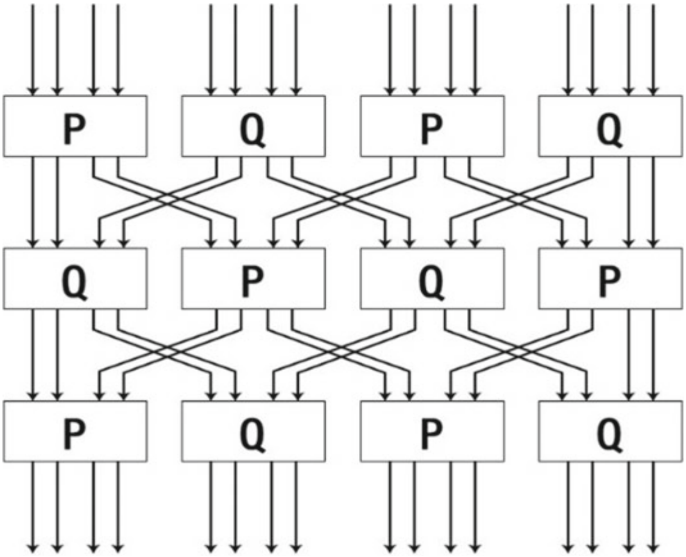

**Fig. 3** F-Function of SIT algorithm

## 2.2 Encryption Process

The encryption process takes place after generating the sub-keys (K1, K2… K6) from the key expansion and also the plaintext to be encrypted as in Fig. 4. The encryption process contains simple logical operations of XNOR, XOR, shifting (left, right), replacing and switching. These operations increase complexity and create confusion for the attackers.

In each round, the blocks are divided into four segments, each segment is 16 bits ($P_{0\text{-}15}$, $P_{16\text{-}32}$, $P_{33\text{-}47}$, $P_{48\text{-}63}$), respectively. This is to produce segments (Ro11, Ro12, Ro13, Ro14).

Ro11 is the output of XNOR between $P_{0\text{-}15}$ and K1, The product (Ro11) feeds F-Function to produce EfL1. Ro14 is the output of XNOR between $P_{48\text{-}63}$ and K1, The product (Ro14) feeds F-Function to produce EfR1.

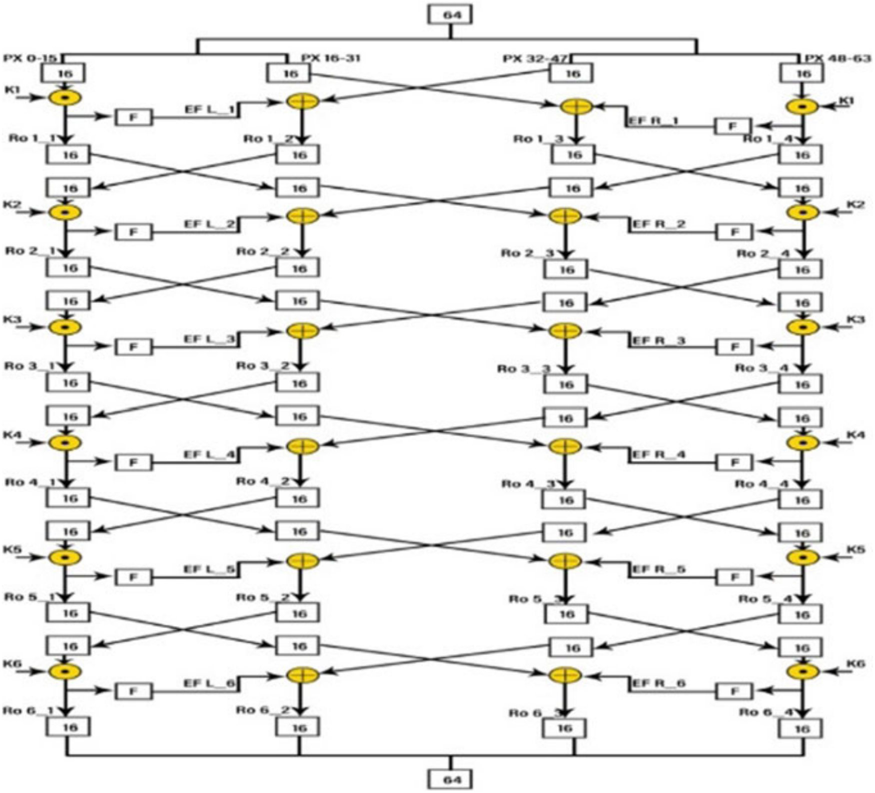

**Fig. 4** Encryption process

Ro12 is the output of XOR between $P_{32\text{-}47}$ and EfL1. Ro13 is the output of XOR between $P_{16\text{-}31}$ and EfR that, the process of switching takes place during the encryption process between the two internal halves. Then, the switches are between the parts (Ro11, Ro12) and (Ro13, Ro14).

All the previous processes are to increase the complexity of the coding as shown in Fig. 4. The same steps for the rounds are repeated by Eq. (4).

$$Ro_{i,j} = \begin{cases} Px_{i,j} \odot k_i; & j = 1.5 \\ Px_{i,j+1} \oplus Ef_{li}; & j = 2 \\ Px_{i,j-1} \oplus Ef_{ri}; & j = 3 \end{cases} \tag{4}$$

After that, the encoded text is obtained by Eq. (5).

$$Ct = R_{51} + \!\!\!+ R_{52} + \!\!\!+ R_{53} + \!\!\!+ R_{54} + \!\!\!+ R_{55} \tag{5}$$

## 2.3 When Decryption Process

Before opening the encryption process, the encrypted key will be extracted from the key sent by the user. The process will be explained in Fig. 5.

- The 80-bit cipher key (send key), is divided into segment R0(40 bit) and segment L0(40 bit).
- $K1 = R0 \oplus L0$
- To combine R0 and K1 (decryption Key 80 bit).

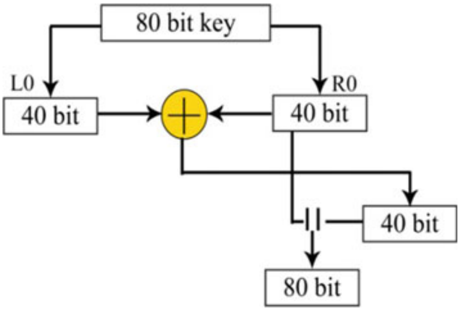

**Fig. 5** When decryption process

# 3 Analytical Comparison of Symmetric Key Algorithm for IoT Proposed Algorithm

In Internet of Things technologies, there are many main symmetric Key algorithms, some of which offer sufficient security and some of them offer better efficiencies in this area. In this section, the proposed algorithm will be discussed, compared, and analyzed with the main symmetric Key algorithms.

Table 4, it shows the comparison process in the various parameters (structure, flexibility, security and limitations) between the proposed algorithm and many of the main symmetric Key algorithms.

## *3.1 Result and Analysis*

The proposed algorithm for the Internet of Things contains a simple design to encrypt the data in six rounds; each round contains simple mathematical operations (free of mathematical complexity). Therefore, the algorithm reduces the process of processing and saving the energy of the devices. In contrast, the processing and power consumption of the algorithms increases AES, SKIPJACK, HIGHT, RC6 the number of turns 12, 32, 32 and 20, respectively, for block coding.

Moreover, the proposed algorithm is a lightweight algorithm for IoT not uses mathematically complex multiplication, small-sized S-boxes are used because most mathematical operations applied to 4-bit data and small-sized boxes. However, the proposed algorithm uses simple mathematical operations and small S-boxes are higher, the overall complexity is lower. Therefore AES and SKIPJACK use large size S-boxes, SEA use variable rounds.

The algorithm provides good flexibility compared to the rest of the algorithms to improve performance in the Internet of Things. Therefore, you use the 80-bit key for 64-bit encryption. The proposed algorithm's flexibility function aims to boost IoT's energy efficiency by reducing power usage, i.e. the energy needed to pad external bits to make the data block fit the size [2].

The proposed algorithm provides security because it merges the architectures SP and Feistel networks. As a result, it provides resistance to most attacks by attackers, as will be discussed below:

- **Linear and Differential Cryptanalysis**
  The proposed algorithm uses each bit in a similar way to maintain round transformation is uniform. Therefore, it provides resistance against this type.
- **Weak Keys**
  This condition occurs in [11] when the non-linear operations in the cipher blocks depend on the actual key value. However, the actual key in the proposed algorithm is not used and to increase the complexity, first uses XORed and then fed f-function. In the f–function all the non-linearity is fixed and there is no limitation on the selection of key.

**Table 4** Some Symmetric Key Algorithms' Comparison in Terms of Flexibility, Architecture, Security & Limitation

| | AES [16] | SKIPJACK [17] | HIGHT [18] | RC6 [19] | SEA [20] | Proposed algorithm |
|---|---|---|---|---|---|---|
| Overview | AES Rijndael developed by J. Daemen and V. Rijmen, was declared U.S.'s new encryption standard by the NIST. It uses variable key size making it extremely fast and compact cipher. Moreover, its symmetric and parallel structure provides great flexibility, with effective resistance against cryptanalytic attacks | The SKIPJACK algorithm was developed by NSA and is classified as SECRET. It was originally intended to be used with sensitive but unclassified information i.e. in the controversial clipper chip. Skipjack is a remarkably simple cipher, and consist of two different rounds A and B. The encryption with Skipjack consists of first applying 8 A-rounds, then 8 B-rounds | The HIGHT was developed by Korean researchers. The research was supported by MIC, Korea and was supervised by the IITA. HIGHT was especially designed for low resource computing device such as sensor nodes or RFID tag. HIGHT provides better security with simple operations to be energy efficient | RC6 a derivative of RC5, designed by R. Rivest, M. Robshaw, R. Sidney, and Y. Lisa Yin. It was design to congregate the requirements of the AES contest and was among the five finalists. It was also presented to the NESSIE and CRYPTREC projects. It is patented by RSA Security. RC6 offers good performance in terms of security and compatibility | SEA (Scalable Encryption Algorithm) was designed in 2006 by François Xavier and Mace. The design criterion of SEA was focused for low-cost embedded environments with limited resources (memory size, processor capacity). To meet the designing criteria, SEA algorithm makes use of basic bit operations such as XOR, bit/word rotations, modular addition, and s-box | The proposed algorithm is a symmetric key algorithm based on a mixture of the Festial and SP architecture that uses simple mathematical operations and fewer turns to provide better safety and lower energy consumption |

(continued)

**Table 4** (continued)

| | AES [16] | SKIPJACK [17] | HIGHT [18] | RC6 [19] | SEA [20] | Proposed algorithm |
|---|---|---|---|---|---|---|
| Architecture | | | | | | |
| • Structure | Substitution-Permutation | Unbalanced Feistal | Festial | Festial | Festial | Festial + SP |
| • Block size | 128 bit | 64 bits | 64 bits | 128 bits | 48, 96, 144 | 64 |
| • Key size | 128, 192, 256 bits | 80 bits | 128 bits | 128, 192, 256 bits | 48, 96, 144 | 80 |
| • No. of Round | 10, 12, 14 | 32 | 32 | 20 | Variable | 6 |
| • No. of operations/Rounds | 5 | 4 | 3 | 4 | 4 | 4 |
| • Mathematical Operations | XOR, Mixing, Substitution, Shifting, Multiplication, Addition. (16 bits) | Permutation, XOR, Shifting, Substitution. (16 bits) | Modular Addition, XOR, Modular subtraction, Shifting. (8 bits) | Variable Rotation, XOR, Modular Addition (2′s comp). (16 bits) | XOR, rotations, 2n mod addition, substitution (8 bits) | XOR, XNOR, Shifting, Substitution.(4 bits) |
| • S-P Structure | 1 S-Box | 1 S-Box | N/A | N/A | 1 S-Box | 4 S- Boxes |
| • S-Box Size | 16 * 16 (16 bits) | 16 * 16 (16 bits) | N/A | N/A | 3 bits | 4 × 4 (4 bits) |
| Flexibility | The layout can be extended to the 64-bit multiple, with the same amount of subkey as that of the main key | The framework does not allow modifications of any sort | The framework does not allow modifications of any sort | Can extend key length to 2048 bits | The size of the block and the key-length has to be in 6 bits and must not be independent of the processor bits | The framework does allow modifications of any sort |

(continued)

**Table 4** (continued)

| | AES [16] | SKIPJACK [17] | HIGHT [18] | RC6 [19] | SEA [20] | Proposed algorithm |
|---|---|---|---|---|---|---|
| Security and Limitations | Increasing the size of the key generates security. Therefore, it provides resistance against some attacks (collision attacks and possible quantum algorithms)[13] | The diffusion process provides security. The use of a smaller number in the rounds represents optimal uses of energy[14]. However, impossible differential and differential attacks are possible | Security depends on the number of rounds (mathematical operations are performed in each round) in order for the block to be strong and resistant to some different attacks such as differential, linear, and saturation. However, to implement the iteration it uses the lookup table which increases the memory requirement. Moreover, a larger number of rotors and a larger key size require additional execution time [13] | Security lies in their totally random sequence of output bits of 15 rounds or less[15] operating on 128-bit input blocks[13]. It takes more than 17 rounds for a single set of low keys to reach maximum arbitrariness[15]. Since of any procedure, such as vector rotation, it is computationally complex, so multiplication takes a longer processing time. Furthermore, it has a large key size and number of round [13] | Key size and variable number of operations represent security. Therefore, they are strong against linear and differential analysis attacks. However, (number of rounds, key size and table of rules used) require longer to implement [13] | The level of confusion and the publishing of data and mathematical operations represent an increase in the complexity of the encryption, and this all generates security, so the main expansion algorithm consists of different mathematical operations, so it is complex |
| | The algorithm uses a large block, complex calculations in each round, large s-boxes(16 × 16). This needs additional memory | It took advantage of 16 × 16 F-BOX which should be stored in either RAM or memory of program, which leads to additional memory usage | | | | |

- **Related Keys**
  Using different keys (unknown or partly unknown) to launch an attack during the encryption process with a selected relation. The attack depends on the symmetry and diffusion relationship in the expansion block. The proposed algorithm for the key expansion process is designed to be resistant to this type of attack because it possesses a high diffusion and non-linearity.
- **Interpolation Attacks**
  Simple coding structures (S-box) had polynomial or rational expressions controllable. It is subject to interpolation attacks that cannot be performed on the proposed algorithm when expressing an S-box along with a diffusion layer.
- **Square Attack**
  Square attack was presented in [12]. The attacker could get the last byte in the last round of the key. In addition, repeats the attack eight times to obtain the rest of the key. Therefore, to get one a byte requires key guessing from the attacker $2^8$ by $2^8$ chosen plaintexts $= 2^{16}$ S-box lookups.

## 4 Conclusion and Future Work

Many different devices and sensors in the Internet of things are interconnected to each other to exchange data. These devices are limited resources and require high data security. Therefore, the proposed algorithm is a lightweight which adopted on Internet of things networks with high flexibility and efficiency. Due to its light of computations, it providing low energy consumption of devices, providing high security, and low computation cost. The performance of the proposed algorithm was analyzed and compared to some symmetrical algorithms to various parameters.

The performance analysis resulting in the superiority of the proposed algorithm which can be represented in low complexity, high security, and saving energy. Briefly, balancing between security, performance, and complexity which leads to making it proper and suitable for devices and sensors in the IoT networks.

The future work is to implement the proposed algorithm in both hardware and software environments and comparing the performance in both environments, i.e., hardware and software.

## References

1. Gubbi, J., Buyya, R., Marusic, S., Palaniswami, M.: Internet of things (iot): a vision, architectural elements, and future directions. FutureGeneration Comput. Syst. **29**(7), 1645–1660 (2013)
2. Al-ahdal, H.A.A., Deshmukh, N.K.: A systematic technical survey of lightweight cryptography on Iot environment. Int. J. Sci. Technol., March 2020
3. Standard, A.E.: Federal Information Processing Standards Publication 197, FIPS PUB, pp. 46–3 (2001)

4. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M.J., Seurin, Y., Vikkelsoe, C.: Present: an ultra-lightweight block cipher. In: International Workshop on Cryptographic Hardware and Embedded Systems. Springer, pp. 450–466 (2007)
5. Daemen, J., Knudsen, L., Rijmen, V.: The block cipher square. In: International Workshop on Fast Software Encryption. Springer, pp. 149–165 (1997)
6. Ebrahim, M., Chong, C.W.: Secure force: a low-complexity cryptographic algorithm for wireless sensor network (wsn). In: 2013 IEEE International Conference on Control System, Computing and Engineering (ICCSCE). IEEE, pp. 557–562 (2013)
7. Schneier, B.: Description of a new variable-length key, 64-bit block cipher (blowfish). In: International Workshop on Fast Software Encryption. Springer, pp. 191–204 (1993)
8. Coppersmith, D.: The data encryption standard (des) and its strength against attacks. IBM J. Res. Dev. **38**(3), 243–250 (1994)
9. Chandramouli, R., Bapatla, S., Subbalakshmi, K.P.: Battery power-aware encryption. ACM Trans. Information Syst. Security **9**(2), 162–180 (2006)
10. Usman, M., Ahmed, I., Aslam, M.I., Khan, S., Shah, U.A.: SIT: a lightweight encryption algorithm for secure internet of things. arXiv preprint arXiv:1704.08688. 27 Apr 2017
11. Daemen, J.: Cipher and hash function design strategies based on linear and differential cryptanalysis. Ph.D. dissertation, Doctoral Dissertation, March 1995, KU Leuven (1995)
12. Barreto, P., Rijmen, V.: The khazad legacy-level block cipher. Primitive submitted to NESSIE, vol. 97 (2000)
13. Koo, W.K., Lee, H., Kim, Y.H., Lee, D.H.: Implementation and analysis of new lightweight cryptographic algorithm suitable for WSN. In: International Conference on Information Security and Assurance, IEEE (2008)
14. Lu, M.C., Bayilmi, C., Özcerit, A.T., Çetin, Ö.: Performance evaluation of scalable encryption algorithm for WSN. Sci. Res. Essays **5**(9), 856–861 (2010)
15. Stankovic, J.A.: Research challenges for WSN. ACM SIGBED Rev. **1**(2), 9–12 (2004)
16. National Institute of Standards and Technology (NIST): Advanced encryption standard (AES). Federal Information Processing Standard (FIPS) 197, Nov. 2001
17. National Institute of Standards and Technology: SkipJack and KEA algorithm specifications (Version 2.0), May 1998
18. Hong, D., Sung, J., Hong, S., Lim, J., Lee, S.: HIGHT: a new block cipher suitable for low-resource device. Cryptographic Hardware Embedded Syst. **4249**, 46–59 (2006)
19. Pavan, R.L., Robshaw, M.J.B., Sidney, R., Yin, Y.L.: The RC6 Block Cipher, Ver 1.1, August 1998
20. Huang, S.I., Shieh, S.: SEA: secure encrypted data aggregation in mobile WSNs. In: International Conference on Computational Intelligence and Security, IEEE (2007)