Simple Lightweight Cryptographic Algorithm to Secure Imbedded IoT Devices

https://doi.org/10.3991/ijim.v16i20.34505

Hakeem Imad Mhaibes[1](✉), May Hattim Abood[2], Alaa Kadhim Farhan[3]
[1] Technical Institute of Kut, Middle Technical University, Baghdad, Iraq
[2] College of Engineering, Al Iraqia University, Baghdad, Iraq
[3] Computer Science Dept., University of Technology, Baghdad, Iraq
hakeem.emade@mtu.edu.iq

Abstract—The internet of things (IoT) revolution has been sparked by the exponential increase in connected devices caused by recent advances in wireless technology. These embedded devices gather, analyze, and send vast data via the network. Data transmission security is a primary problem in IoT networks. Several lightweight encryption/decryption algorithms have been developed for such resource-constraint devices. One of most effective and fast lightweight encryption algorithms for IoT applications is the Tiny Encryption Algorithm (TEA). TEA has few lines source of codes to implement and based on Feistel structure to provide cryptographic primitive confusion and diffusion features in order to hide statistical aspects of plaintext. However, it is vulnerable to assaults using equivalent and related key attacks. This study suggested modifying TEA by employing a new proposed generating keys function using two Linear Feedback Shift Registers (LFSRs) as a combination to address the security flaw caused by utilizing different keys for each round function. The key sensitivity, Avalanche effect, and a completeness test were used to evaluate its security performance. The key sensitivity of the proposed modified TEA outperforms original TEA by 50.18 % to 44.88 %. The modified TEA avalanche effect outperforms TEA by 52.57 % to 47.69 %, and its completeness test outperforms TEA by 51.75 % to 48.36 %. Experimental results indicates that, the encryption performance of proposed modified TEA is better than original TEA.

Keywords—IoT, information security, block cipher, LFSR, lightweight cryptography, Feistel structure

1 Introduction

With the extensive development of the Internet of Things (IoT) and computer technology, the era of IoT has arrived; information technology has always influenced our work and daily lives. Due to the openness of the Internet's information system, the issue of information security is receiving increasing attention and must be resolved immediately [1], [2], [3], [4].

In many instances, the typical security protection based on software cannot provide sufficient system security. Hackers are capable of attacking the operating system and stealing crucial data.

IoT security technology is required for many application scenarios, such as industrial control security, innovative car safety, and intelligent home security [5].

Encryption and decryption technology is a best choice to safeguard our property security, secret data, and personal information, and along with their proper use, can effectively ensure the system's security.

But traditional cryptographic algorithms do not work well for IoT devices because of limited power, bandwidth, execution-time, and computation capabilities [6]. Therefore, lightweight cryptographic algorithm that uses less energy is important for devices with limited resources, such as those that run on batteries [7]. Hence, several lightweight encryption/decryption algorithms have been developed for such resource-constraint devices such as TEA [8], XTEA [9], PRESENT [10], RC5 [11], and HIGHT [12].

One of most effective and fast lightweight encryption algorithms for resource-constraint IoT applications is TEA. TEA has few lines source of codes to run, which is based on Feistel structure to provide cryptographic primitive such as confusion and diffusion features in order to hide statistical aspects of plaintext. TEA is suited for applications that demand low-power consumption, high-performance, low-battery. TEA is capable of meeting these conditions and resists differential cryptanalysis.

However, TEA's straightforward key scheduling is its greatest weakness [13]. Due to TEA's simple key scheduling, equivalent key attacks are possible. Numerous TEA modifications have been suggested over the last few years to solve the issue [14], [15], [16], [17].

This research intended to modify original TEA in order to overcome its shortcomings and vulnerabilities and improve its security. The proposed modified TEA suggests employing a new generating keys function using two Linear Feedback Shift Registers (LFSRs) as a combination to address the security flaw caused by utilizing different keys for each round function. The security and performance of proposed modified TEA are evaluated using three standard statistical analysis tests, the key sensitivity, avalanche effect, and completeness test and comparing with stat-of-art researches.

The structure of this paper is organized as follows: Section 2 presents current related works. Feistel structure is illustrated in section 3. In section 4, the original version of TEA was illustrated. The feedback shift register is stated in section 5. The description of proposed work is presented in section 6. Section 7, the results are given and security analysis is evaluated. Finally, conclusion and for future enhancements are presented in section 8.

2 Related works

The original developers of TEA implemented improvements to the method after discovering its flaws; however, the security performance of the original method was not evaluated using avalanche effect and completeness tests [8]. Numerous TEA modifications have been suggested over the last few years to enhance security of TEA.

In 2013, Abdelhalim et al. proposed MTEA as a modified for the original TEA using one LFSR to solve the problem of equivalent key. LFSR was used as a key generator instead of TEA original key to produce new key every round. Through avalanche effect, MTEA reveals better security performance result than TEA [14].

In 2018, another improvement for the TEA algorithm was using the new SBOX. SBOX is based on a nonlinear Boolean function which used to generate new keys every round. This improvement increases the TES's security to some level. Their proposed work was tested using a completeness test and outperformed TEA [16].

In 2018, the authors of [15] suggested that, instead of using one single function every round in original TEA, they used two with two keys. The author proved that, using related keys and same plaintext can produce different ciphertexts, resolving the related keys problem of TEA.

In 2019, a new modification to TEA was proposed by De Leon et al. in [17]. They improved the security of TEA using new key scheduling to rotate subkeys every round function. In this work, they adjusted the round function of original TEA by rotating the generated key before function integration. The security analysis for this work was test using avalanche effect and completeness test and benchmarking to previous works.

As of 2019, researchers in [13] have devised a new, small symmetric encryption algorithm (NTSA) that enhances the security of text file transfers over the Internet of Things (IoT) network by dynamically adding extra key confusions. K[0] is always the same, but K[1] changes for all odd rounds, and K[2] is always the same, but K[3] changes for all even rounds in this example. NTSA's avalanche effect, encryption and decryption times on an IoT network with embedded devices are studied in experiments. The findings reveal that the suggested NTSA algorithm is much more secure and efficient than the current state-of-the-art encryption techniques.

Hardi et al. suggested a new enhancement for TEA to overcome its weakness and increase encryption process. Thire work used Least Significant Bit (LSB) as a steganographic method to hide data into last bit of the other. The LSB was used Linear Congruential Generator (LCG) to generate the initial seeds as a PRNG. The flow of inserting data intended for encryption is used as a cover for safely sending confidential data. The proposed methods become modified LSB-PRNG [18].

In 2021, Suresh et al. proposed an Enhanced Tiny Symmetric Encryption Technique (ETSET). The suggested method dynamically modifies the key generation to generate new key every round function by dynamic bit shifting in each cycle. For simplicity and execution -time, their proposed method suggested minimize the number of TEA rounds.

This work was implemented using Java programming language and compared to evaluated using avalanche effect and completeness test and compared with TEA and XTEA [19].

3 Feistel structure

It is a process of hiding the statistical features of the plaintext (block cipher) by using two primary method substitution and permutation. The original concept of Feistel structure was first developed by Claud Shannon to replace the confusion and diffusion [20].

Shannon proved that, cryptoanalysis is depend on the frequency distribution of language aspects of the plaintexts such as letters, word, or phrases that being converted to ciphertexts. If the attacker has knowledge of these data, he may be able to decipher the plaintext encryption key.

Two methods of operations were Shannon provided to hide the statistical features of the plaintexts;

- **Confusion:** is to hide the relationship between ciphertext and encrypted key to make it difficult for an attacker to know the encrypted key.
- **Diffusion:** is to hide the relationship between the plaintext and ciphertext in order to be more difficult against an attacker to guess the plaintext.

The Feistel cipher uses substitution and permutation to achieve the concepts of confusion and diffusion, as seen in Figure 1. This structure is also referred to as the Feistel Network.

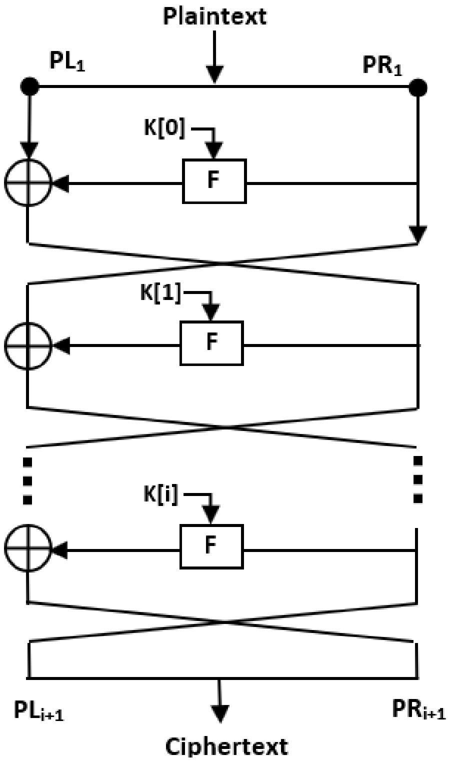


Fig. 1. Diagram of Feistel structure

As seen in Figure 1, Feistel network works as a series of mathematical linked operations for the data to be encrypted. It takes two input blocks; plaintext and the key and performed round function several times. In each round, the half of the plaintext are encrypted using the function F and the result is XORed with other half and repeated for a specific number of rounds.

The substitution function runs by XORing the left half with the result of function F. In permutation, the two halves of the plaintext were exchanging.

In the decryption process of Feistel network is same as encryption process but in reverse order.

There are many of symmetric block cipher algorithm used Feistel structure, such as Blowfish, DES, TEA, RC5, Simon, and Camellia [21].

4 Tiny Encryption Algorithm (TEA)

David Wheeler and Roger Needham developed original TEA at the Cambridge University in 1994 [8]. TEA method is recognized as one of the most efficient and fast block cipher algorithms.

Basically, TEA used basic operations of Feistel network (ADD, XOR, and SHIFT) to provide diffusion and confusion concepts.

TEA takes 64-bit plaintext as the input along with the 128-bit key and produces 64-bit ciphertext as the output. At first, it divides the input into two parts (PL_i and PR_i). These two halves are subjected to 32 rounds operations. Each round consists of 2 Feistel cycles and hence, in total there are 64 cycles.

As seen in Figure 2, the block diagram shows encryption process for only ith cycle of the algorithm.

The 128-bit key of the algorithm is splitting into K[0], K[1], K[2], and K[3], where each key is 32-bit. Basic operations are performed 32 rounds for plaintext to be encrypted, where each half used to encrypt other. Finally, the two halves are combined together to produce 64-bit ciphertext.

Following are the mathematical equation involved in each round;

$$PL_i = PL_{i-1} + F\big(PR_{i-1}, Delta_j, K[0,1]\big) \tag{1}$$

$$PR_i = PR_{i-1} + F\big(PL_{i-1}, Delta_j, K[2,3]\big) \tag{2}$$

Where F function is as follows;

$$F\big(P, Delta_j, K[a,b]\big) = (P \ll 4) + K[a]) \oplus \big(P + Delta_j\big) \oplus (P \gg 5) + K[b])) \tag{3}$$

TEA uses two bitwise shifts to mix bits of the data and the key repeatedly and uses XOR and ADD to provide nonlinearity.

Delta is a constant golden number ratio which derived using equation:

$$delta = \big(\sqrt{5} - 1\big) * 2^{31} = 9E3779B9h \tag{4}$$

$$delta[n] = (n + 1) * delta, n = 0, 1, 2, \ldots, 31 \tag{5}$$

The objective of delta is to ensure that the key is unique.

The decryption process is a reversed way, where the key of the final round in the encryption process will be used for the first round in the decryption process.

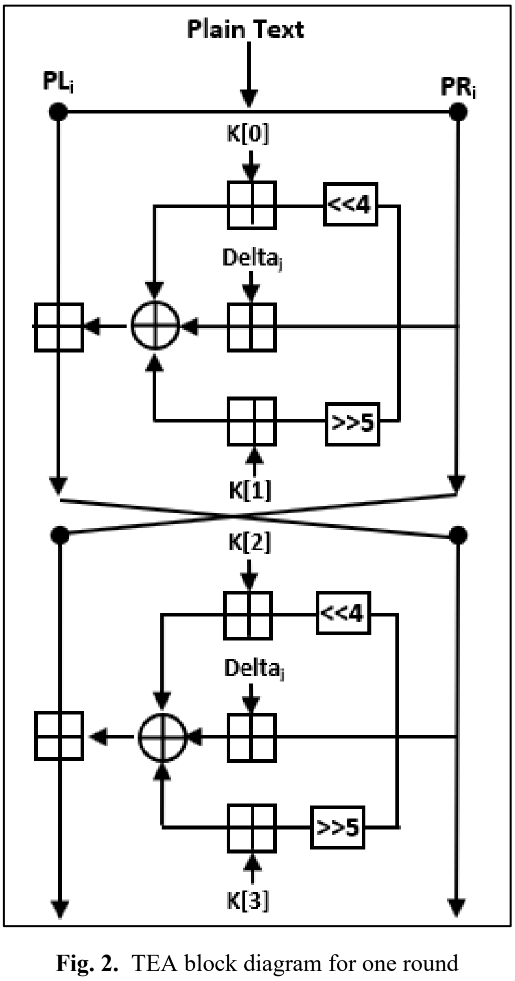


Fig. 2. TEA block diagram for one round

5 Feedback shift register

The feedback shift register consists of a shift register and a feedback function. The sequence of bits comprises a shift register, and a feedback function is nothing but XORing specific bits in the register (called tap sequence). The sequence of n-bit determined the length of the shift register.

LFSR is the simplest type of feedback shift register, where each time a bit is fed to the register, it causes a shift of all bits to the right by one. This bit is a left-most bit and is computed using a feedback function. The right-most bit is the output of LFSR.

Each LFSR has a specific length of the output of n-bit. This sequence can be repeated endlessly and is determined by the number of registers [22]. In other words, the maximum period of a LFSR can generate a bit-sequence and start repeating. The value of n determines the number of registers. Not every LFSR produces states, but only particular taps can create maximal length. Since LSFR is a deterministic approach, that means, the output sequence depends on the given initial seeds.
The statistical polynomial of maximum-length LFSR is primitive if it has the following:

- If the number of ones is n, then the number of zeros is equal to n+1.
- If the consecutive runs of ones are n, then the consecutive runs of zeros are n.
- Number of runs that have length of one are half for the output sequence.
- Number of runs that have length two are quart for the output sequence.

Researchers in [23] proved that, the best randomized performance achieved by LFSR if the polynomial uses one of the following equations:

$$l_1 = x^{128} + x^{123} + x^{78} + x^{64} + 1 \tag{6}$$

$$l_2 = x^{128} + x^{121} + x^{58} + x^{47} + x^{32} + 1 \tag{7}$$

$$l_3 = x^{128} + x^{7} + x^{2} + x + 1 \tag{8}$$

The above polynomials can produce 128-bit sequence key. The proposed method uses two LFSRs with polynomial in Eq. 6 to generate new keys each round. The illustration of this polynomial is depicted in Figure 3.

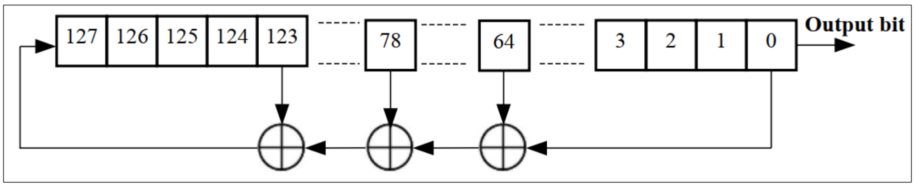


Fig. 3. 128-bit LFSR using Eq. 6

In cryptography, to propose a successful Pseudo-Random Numbers Generator (PRNG), the produced output sequence must be highly randomized and passed all the statistical test and the generator must be reliable, low-cost, unpredictable, high complexity, and high-speed. Therefore, LFSR may be used as deterministic cryptographic PRNG. In general, basic LFSR does not provide a high statistic random number [24], but it can be improved by picking LFSR with large number order and further using two or more LFSRs [25].

6 Proposed modification of TEA

The proposed modified TEA employs two LFSRs, L1 and L2. The LFSRs used in this proposal as PRNGs to generate keys each round to boost its security against crypt-analysis. To solve the problem of equivalent key attacks, 32-keys are used in the modified TEA $[K_0, K_1, K_2, \ldots K_{31}]$. This proposal increases the complexity of the algorithm. Keys are generated dynamically before execution; thus, they are unpredictable by an attacker.

To increase the characteristic performance of randomness. Each key is generated as follows; Since each LFSR generates a 128-bit sequence, then for each round, the output sequence of L1 is 128-bit, divided into four sub-sequences $(L1[0], L1[1], L1[2], and\ L1[3])$, each of which is 32-bit. Similarly, the generated sequence of L2 is 128-bit, divided into four sub-sequences $(L2[0], L2[1], L2[2], and\ L2[3])$. Now, for more confusion, all sub-sequences are XORed using the following equation;

$$K[0] = L1[0] \oplus L2[2] \tag{9}$$

$$K[1] = L1[1] \oplus L2[3] \tag{10}$$

$$K[2] = L1[2] \oplus L2[0] \tag{11}$$

$$K[3] = L1[3] \oplus L2[1] \tag{12}$$

Figure 4 shows the proposed modified new key K_i for round$_i$ in modified TEA.

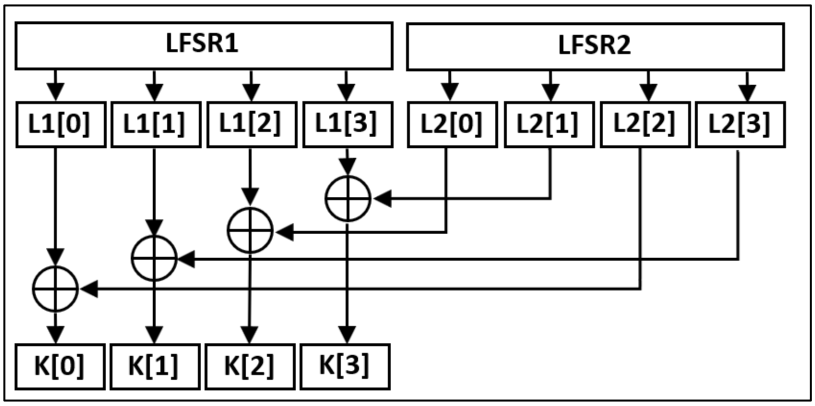


Fig. 4. New proposed keys generation

Generally, symmetric cryptography uses same key for encryption and decryption. Since the proposed modified TEA used 32 keys, that is one key for each round and numbered such as $[K_0, K_1, K_2, \ldots K_{31}]$. For decryption part, keys are in reverse order, such as, K_{31} is used for the first round and K_{30} for second round and so on.

This criterion is useful and has un advantages over original TEA, since the decryption keys are distributed to user in a secure channel, they are distinct from the encryption keys. These keys are strongly protected when an attacker tries to distinguish the original keys.

The block size of plaintext for the proposed modified TEA is same as original. The inputs are, 64-bit plaintext and 128-bit key, the output is 64-bit ciphertext. Figure 5 shows the ith round of modified TEA.

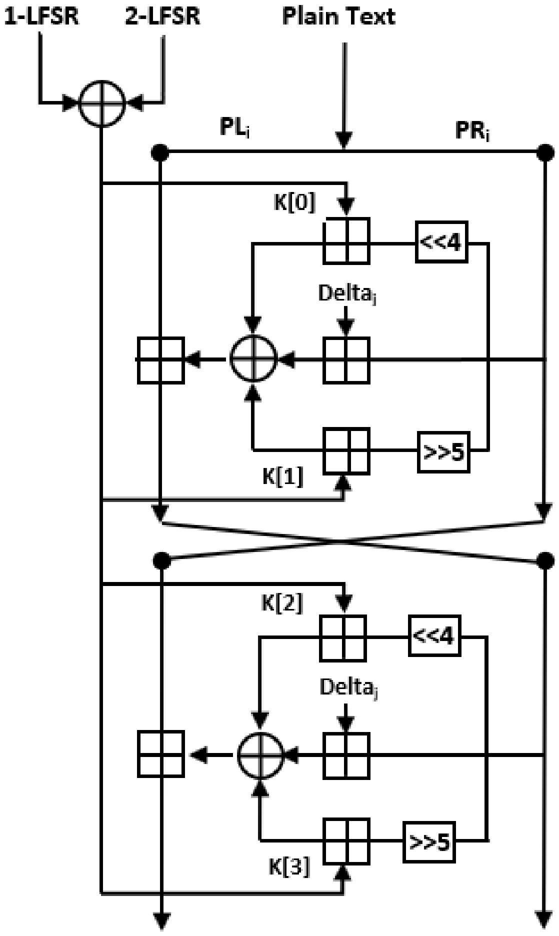


Fig. 5. The proposed modified TEA using new keys (one round, two cycles)

As seen in above figure, two LFSRs are XORing using proposed suggested method, where each round as follow.

For each cycle i;

$$PL_i = PL_{i-1} + F\left(PR_{i-1}, Delta_j, K[0,1]\right) \quad (1)$$

$$PR_i = PR_{i-1} + F\left(PL_{i-1}, Delta_j, K[2,3]\right) \quad (2)$$

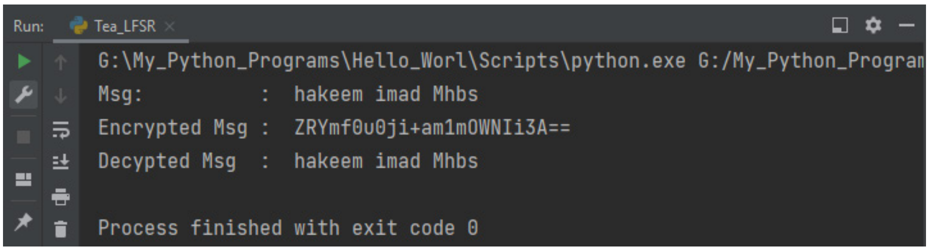
Where F function;

$$F\left(P, Delta_j, K[a,b]\right) = (P \ll 4) + K[a]) \oplus \left(P + Delta_j\right) \oplus (P \gg 5) + K[b])) \quad (3)$$

As stated before, the decryption of proposed TEA is same as encryption but in reverse order.

7 Result and security analysis

This work was implemented using Python programming language on Windows 10. The tests are performed using laptop with Core i3 CPU and 2.10 GHz, with 4 GB RAM. Authors in [26] proved that, the problem of equivalent keys produces same ciphertext using TEA algorithm. But the modified TEA produces different ciphertext. For example, using a plaintext "hakeem imad mhbs", with ones as LFSR initial state. As proposed modification, the key is changing dynamically each round. Therefore, thirty-two keys used in the encryption process. The output ciphertext is "ZRYmf0u0ji+am1mOWNIi3A==" and decrypted result is same as plaintext. As seen in Figure 6.



```
G:\My_Python_Programs\Hello_Worl\Scripts\python.exe G:/My_Python_Progran
Msg:          :  hakeem imad Mhbs
Encrypted Msg :  ZRYmf0u0ji+am1mOWNIi3A==
Decypted Msg  :  hakeem imad Mhbs

Process finished with exit code 0
```

Fig. 6. Encryption and decryption of plaintext using modified TEA in Python

Three crucial security analysis criteria for block ciphers—the key sensitivity, completeness, and avalanche effect tests are used to gauge the proposed algorithm's security strength. These experiments demonstrate that the modified TEA algorithm offers greater security strength when compared to the standard TEA method.

7.1 Key sensitivity analysis

To guarantee the security of an encryption algorithm, a ciphertext must be sensitive to encryption key. Key sensitivity measure how the encryption algorithm sensitive to the encryption key.
In this test, 10 plaintexts were encrypted using K_i, and encrypted same plaintexts again using same key except changing the last bit of the key.

In this example, key sensitivity is calculated by computing the differences between the obtained ciphertexts. Table 1 shows key sensitivity results in percent for original TEA and our proposed modified TEA. The proposed TEA was done using LFSR with polynomial in Eq. 6.

Table 1. Key sensitivity test

Block	TEA%	Modified TEA %
1	44.70	49.60
2	43.10	48.90
3	42.90	53.45
4	41.78	52.64
5	43.69	51.20
6	49.57	59.75
7	45.50	49.50
8	50.25	57.60
9	46.42	50.90
10	40.86	48.30
Avg.	44.88	52.18

The above table showed that the proposed modified TEA has large percentage value of key sensitivity than TEA.

7.2 Completeness test

When every bit of the ciphertext is sensitive to every bit of the plaintext, the cipher is said to be completed [27]. The desired probability of completeness test in a cryptographic algorithm is 50 %. Such that, if we change one bit in the plaintext, then each bit in the ciphertext has probability of change for an average of 50 %.

In this work, 65 plaintexts were used for this test. The size of ach plaintext is 64-bit, and each plaintext is varied in one bit to others. Then, we encrypt the plaintexts to obtained ciphertexts and each ciphertext is XORed with its previous version. The results are filled in a matrix of (64 x 64). Numbers of ones in each row and in the whole matrix were calculated. The percentage of ones in whole matrix were evaluated and presented in Table 2. The completeness test was evaluated and compared with original TEA and MTEA [14].

Table 2. Completeness test evaluation for proposed modified TEA

TEA% [8]	MTEA % [14]	Proposed Modified TEA%
48.36	50.46	51.75

The results indicate that, proposed modification TEA achieves highest completeness test, which means that, when one bit change in the plaintext, the output ciphertext changes significantly. This property is desirable for security analysis of the cryptographic algorithm.

7.3 Avalanche effect test

Feistel proposed the avalanche test on [28]. The purpose if this test is to evaluates if a slight change in the plaintext or the key leads to a significant change in the ciphertext. This can be calculated using an equation;

$$AE = \frac{No.of\ filled\ bit\ in\ ciphertext}{No.of\ bit\ in\ cipherdtext} \tag{13}$$

The performance of the avalanche effect was evaluated by encrypting 10 blocks of plaintexts, resulting in 10 ciphertexts, and then encrypting the same 10 plaintexts again but, one bit changed. Then each pair of generated ciphertext XORed together and evaluated by computing the number of ones in each result to obtained the percentage average (No. of ones / 64-bit). Finally, avalanche affect is evaluated by calculating percentage average for whole ciphertexts.

As seen in Table 3, the test was applied for original TEA and the proposed modified TEA. Compared to the standard TEA, the likelihood of bit reoccurrence in the ciphertext for each encrypted plaintext is extremely low. For instance, the avalanche effect for TEA was recorded 48.97 in block 10, while it was 53.95 for modified TEA. This implies that, probability of reoccurrence was 4.98 compared to original TEA. Figure 7 shows a considerable difference in margin for 10 blocks, which proves that, the modified TEA has higher performance for encryption than original TEA.

Table 3. Results of Avalanche effect for 10 Blocks using TEA and proposed modified TEA

No. of Blocks	TEA	Modified TEA	Differences
1	43.65	48.63	4.98
2	44.98	48.71	3.73
3	48.75	54.85	6.10
4	56.27	58.75	2.48
5	47.67	53.24	5.57
6	46.26	51.86	5.60
7	48.21	52.71	4.50
8	43.46	50.35	6.89
9	48.67	52.67	4.00
10	48.97	53.95	4.98
Average	47.69	52.57	4.88

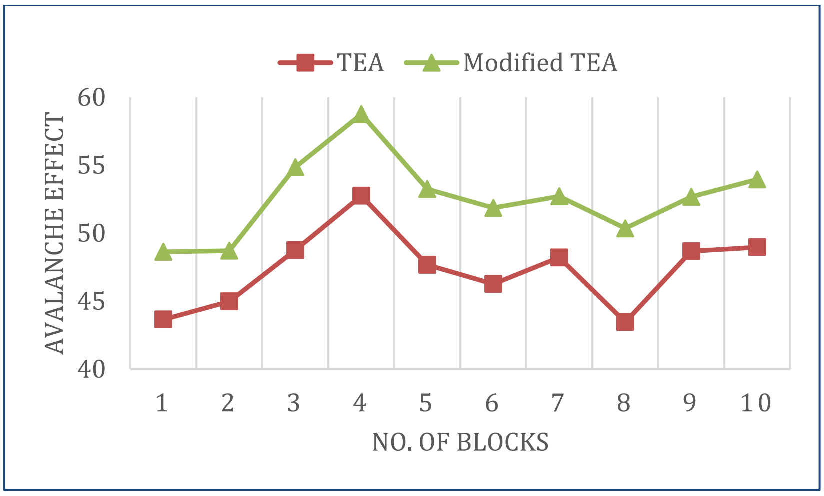


Fig. 7. Avalanche effect of modified TEA with respect to original TEA

Figure 8 shows high value of avalanche effect resulting from modified TEA as compared to original TEA. For 10 blocks, the minimum value of margin is 2.48 and the maximum is 6.89. This implies that, modified TEA has stronger encryption characteristic for all 10 generated ciphertexts than traditional TEA.

The benchmark created for differentiate the modified TEA to original TEA, as seem in Figure 8. Evidently, the devised method exceeded the typical threshold of 50 percent for avalanche effect processes.

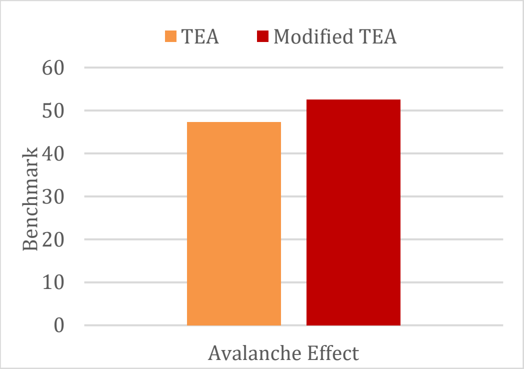


Fig. 8. Benchmarking for avalanche effect of modified TEA and TEA

8 Conclusion and future works

This paper intended to increase the security performance of the original TEA. The modification offers a simple and a new modified method of key generation using two LFSRs. The security vulnerability of the original TEA was solved.

The encryption and decryption performance of modified TEA was analyzed using key sensitivity, avalanche effect, and completeness test to be compared with the previous work of traditional TEA and MTEA.

- Through key sensitivity analysis evaluation, the proposed implementation of new key generation for modified TEA could generate keys that have higher randomness characteristic and more confusion than other.
- For completeness test, 65 plaintexts were tested, where there is only one-bit variation between each plaintext and the one after it. This test is run on both the modified and original TEA. The results indicate that, proposed method achieves highest completeness test, which means that, when one bit change in the plaintext, the output ciphertext changes significantly. This property is desirable for security analysis of the cryptographic algorithm.
- Moreover, modified TEA has higher value of avalanche effect. This test proved that, 50 % or higher of ciphertext is will be changed if one bit or few bits of the plaintext changed. For avalanche effect, the minimum value of margin is 2.48 and the maximum is 6.89 for 10 tested blocks. The benchmark created for differentiate the modified TEA to original TEA. Evidently, the proposed method exceeded the typical threshold of 50 percent for avalanche effect processes.

Since embedded IoT devices are resource constraint, the modified work is simple and has no complexity overhead. In addition, LFSR is a simple and an appropriate way to use in such devices because of its simplicity computation. Therefore, the proposed work is suitable for latest IoT applications to transfer data through network.

By encrypting the compressed file, the modified TEA would be used. Also, future plans include implementing and integrating this technique in fog net, sensor, or ad hoc for data transmission.

9 References

[1] Juels, A., “RFID security and privacy: A research survey,” *IEEE journal on selected areas in communications*, 2006. 24(2): p. 381-394. https://doi.org/10.1109/JSAC.2005.861395

[2] Hassan and W.H., “Current research on Internet of Things (IoT) security: A survey,” *Computer networks*, 2019. 148: p. 283-294. https://doi.org/10.1155/2021/8847099

[3] Hassija, V., et al., A survey on IoT security: application areas, security threats, and solution architectures,”.*IEEE Access*, 2019. 7: p. 82721-82743. https://doi.org/10.1109/ACCESS.2019.2924045

[4] Kadhim, A. and H.I. Mhaibes, “A new initial authentication scheme for kerberos 5 based on biometric data and virtual password,” *in 2018 International Conference on Advanced Sc ence and Engineering (ICOASE)*, 2018. IEEE. https://doi.org/10.1109/ICOASE.2018.8548852

[5] Zhang, Z.-K., et al. “IoT security: ongoing challenges and research opportunities,” *in 2014 IEEE 7th international conference on service-oriented computing and applications*. 2014. IEEE. https://doi.org/10.1109/SOCA.2014.58

[6] Ahmad, R. and I. Alsmadi, “Machine learning approaches to IoT security: A systematic literature review,” Internet of Things, 2021. 14: p. 100365. https://doi.org/10.1016/j.iot.2021.100365

[7] I Mhaibes, H. and S. Qadir, "A Lightweight Authentication Framework for Wireless Sensor Networks," *International journal of electrical and computer engineering systems, 2022.* 13(1): p. 19-27. https://doi.org/10.32985/ijeces.13.1.3

[8] Wheeler, D.J. and R.M. Needham, "TEA, a tiny encryption algorithm," *in international workshop on fast software encryption*. 1994. Springer. https://doi.org/10.1007/3-540-60590-8_29

[9] Needham, R.M. and D.J. Wheeler, "TEA Extensions," *Report (Cambridge University, Cambridge, UK, 1997),* 1997. http://www.club.cc.cmu.edu/~ajo/docs/xtea.pdf

[10] Bogdanov, A., et al. "PRESENT: An ultra-lightweight block cipher," *in international workshop on cryptographic hardware and embedded systems*. 2007. Springer. https://doi.org/10.1007/978-3-540-74735-2_31

[11] Rivest, R.L. "The RC5 encryption algorithm," *in International Workshop on Fast Software Encryption*. 1994. Springer. https://doi.org/10.1007/3-540-60590-8_7

[12] Hong, D., et al. "HIGHT: A new block cipher suitable for low-resource device," *in International workshop on cryptographic hardware and embedded systems*. 2006. Springer. https://doi.org/10.1007/11894063_4

[13] Rajesh, S., et al., "A Secure and Efficient Lightweight Symmetric Encryption Scheme for Transfer of Text Files between Embedded IoT Devices," *Symmetry,* 2019. 11(2). https://doi.org/10.3390/sym11020293

[14] Abdelhalim, M.B., M. El-Mahallawy, and M.A. A. Elhennawy, "Design and Implementation of an Encryption Algorithm for use in RFID System," *International Journal of RFID Security and Cryptography*, 2013. 2(1): p. 51-57. https://doi.org/10.20533/ijrfidsc.2046.3715.2013.0007

[15] Aradhyamath, S. and J. Paulose, "Multi-key Modified Tiny Encryption Algorithm for HealthCare," *International Journal of Engineering & Technology*, 2018. 7(2.14). https://doi.org/10.14419/ijet.v7i2.9894

[16] Rajak, C.K. and A. Mishra, "Implementation of Modified TEA to Enhance Security," *in Information and Communication Technology for Intelligent Systems (ICTIS 2017)* - Volume 1. 2018. p. 373-383. https://doi.org/10.1007/978-3-319-63673-3_46

[17] De Leon, R.M., A.M. Sison, and R.P. Medina, "A Modified Tiny Encryption Algorithm Using Key Rotation to Enhance Data Security for Internet of Things," *in 2019 International Conference on Information and Communications Technology (ICOIACT).* 2019. IEEE. https://doi.org/10.1109/ICOIACT46704.2019.8938456

[18] Hardi, S., et al. "Security of Image File with Tiny Encryption Algorithm and Modified Significant Bit Pseudo Random Number Generator," *in Journal of Physics: Conference Series*. 2020. IOP Publishing. https://doi.org/10.1088/1742-6596/1566/1/012108

[19] Suresh, S.A. and J. Priyadarsini, "ETSET: Enhanced Tiny Symmetric Encryption Techniques to Secure Data Transmission among IoT Devices," *Turkish Journal of Computer and Mathematics Education*, 2021. 12(10): p. 1094-1099.

[20] Shannon, C.E., "Communication theory of secrecy systems," *The Bell system technical journal,* 1949. 28(4): p. 656-715. https://doi.org/10.1002/j.1538-7305.1949.tb00928.x

[21] Paar, C., A. Poschmann, and M. Robshaw, "New designs in lightweight symmetric encryption, in RFID Security," *2008, Springer*. p. 349-371. https://doi.org/10.1007/978-0-387-76481-8_14

[22] Bhattacharjee, K., K. Maity, and S. Das, "A search for good pseudo-random number generators: Survey and empirical studies," *arXiv preprint arXiv*:1811.04035, 2018. https://doi.org/10.1016/j.cosrev.2022.100471

[23] Lauradoux, C. "From hardware to software synthesis of linear feedback shift registers," *in 2007 IEEE International Parallel and Distributed Processing Symposium*. 2007. IEEE. https://doi.org/10.1109/IPDPS.2007.370643

[24] Kadhim, A. and M.H. Emad, "Mouse movement with 3D chaotic logistic maps to generate random numbers," *Diyala Journal For Pure Science*, 2017. 13(3 -part 2): p. 24-39. https://www.iasj.net/iasj/article/128520

[25] Garcia-Bosque, M., C. Sánchez-Azqueta, and S. Celma, "Secure communication system based on a logistic map and a linear feedback shift register," *in 2016 IEEE International Symposium on Circuits and Systems (ISCAS).* 2016. IEEE. https://doi.org/10.1109/ISCAS.2016.7527454

[26] Andem, V.R., "A cryptanalysis of the tiny encryption algorithm," 2003, *University of Alabama Alabama*. https://www.tayloredge.com/reference/Mathematics/VRAndem.pdf

[27] Kam, J.B. and G.I. Davida, "Structured design of substitution-permutation encryption networks," *IEEE Transactions on Computers,* 1979. 28(10): p. 747-753. https://doi.org/10.1109/TC.1979.1675242

[28] Feistel, H., "Cryptography and computer privacy," S*cientific american*, 1973. 228(5): p. 15-23. https://doi.org/10.1038/scientificamerican0573-15

10 Authors

Hakeem Imad Mhaibes is member of the Middle Technical University, Baghdad, Iraq. He received the BSc degree in computer science from the Al-Mustansiriyah University, Baghdad, and the Msc degree in computer science (computer programming) from Jamia Hamdard University, New Delhi, India in 2011 and the Ph.D. degree in information security from the University of Technology, in 2019. His research interests include Cryptography, Information Security, Wireless Sensor Network, Biometric Techniques, Image Processing, and Pattern Recognition, Programming (Contact email: hakeem.emade@mtu.edu.iq).

May Hattim Abood is a Lecturer at Al-Iraqia University. She received the BSc. and MSc. in Information Engineering from Al-Nahrain University, College of Information Engineering, Baghdad, Iraq in 2005 and 2009 respectively. She has around 12 years of teaching experience and has a Certificate of Training Program for the Establishment of Computer Network System. Her areas of interests are Digital Image Processing, Video Processing, Communication and signal processing, Information Security, Network Security and Computer Networking (Contact email: may.hattim@aliraqia.edu.iq).

Alaa K. Farhan is Professor in the Department of Computer Sciences, University of Technology-Baghdad-Iraq. He completed his Bachelor of computer Science and Master of Science degrees in information security, from Department of computer Sciences-University of Technology, Baghdad, Iraq, in 2003, and 2005, respectively. He received his Ph.D. degree in information security from University of Technology, Baghdad, Iraq 2009. In 2005 he joined the Department of Computer Sciences, University of Technology, as an academic staff member. Assist. Prof. Dr. Alaa is the author of numerous technical papers since 2008, his research interests include: Cryptography, programming languages, Chaos theory, cloud computing (Contact email: 110030@uotechnology.edu.iq).

Article submitted 2022-08-06. Resubmitted 2022-09-21. Final acceptance 2022-09-22. Final version published as submitted by the authors.