International Journal of Engineering & Technology, 7 (2) (2018) 559-563

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i2.9894
Research paper

Multi-key Modified Tiny Encryption Algorithm for HealthCare

Sharath Aradhyamath [1]*, Joy Paulose [2]

[1] Department of Computer Science, Christ University, Bengaluru, India
[2] Department of Computer Science, Christ University, Bengaluru, India
**Corresponding author E-mail: sharath.aradhyamath@mca.christuniversity.in*

Abstract

Healthcare data is crucial in patient's surveillance. To give the finest treatment to patients the data should be of the best quality. Many security threats affect the integrity and ingenuity of the medical data. In the recent times, medical data can be accessed anytime and anywhere with the help of an overwhelming technology called the Internet of Things (IoT). IoT has numerous applications based on healthcare, however, it facilitates data misuse such as data breach and health care fraud. Sensitive and protected data is stolen and modified by an unauthorized person. Due to these data abuses, it degrades the quality of medical data and service. In this paper, secure data transfer is done in IoT based healthcare to enhance the security of the medical data. Data confidentiality can be achieved using encryption algorithms such as Tiny Encryption Algorithm (TEA) which is lightweight and suitable for resource constraint devices. The results of the proposed system show that the modified TEA overcomes the drawback of the equivalent key and also provides a better security to healthcare data.

Keywords: *IoT, Data confidentiality, HealthCare, lightweight, Equivalent key.*

1. Introduction

The Internet of Things (IoT) is a next-generation technology that can be considered as an interconnection of the individually identifiable smart device within today's internet infrastructure. Medical care represents one of the most promising application areas for IoT. The IoT has the ability to give rise to many applications in the field of medical care such as remote health monitoring, chronic disease, elderly care and many more. Therefore, various medical devices and health monitoring sensors can be considered as smart devices that constitute a fundamental part of IoT. These smart devices assist healthcare providers in monitoring their patients remotely, thereby enabling the doctors to respond quickly in the event of emergencies. It is also expected to increase the span of life, reduce the healthcare cost and enrich the user's experience [1] - [2].

Pervasive healthcare acquires the data from remotely monitored patients and this data is used to automate the clinical decision support system. So by remote monitoring, patients are more closely treated whereas financial and human resource can be minimized [3].

Privacy is one of the major concern in the field of Wireless Sensor Network (WSN) with respect to healthcare applications. The data associated with health are always private and confidential in nature. Transmitting the collected data from one patient through the wireless network can pose a severe threat to the confidentiality of an individual [4]. The attacks which can possibly occur in the healthcare system are addressed by some of the authors [5].

Monitoring and processing of healthcare data facilitate data abuse. There are several types of data abuses, among which data breach is a major issue. Data breach is nothing but a security incident where the protected data is viewed or transmitted by an unauthorized person. Healthcare sector is the most affected industry area when it comes to data breach [6]. Figure 1 shows that data theft is the major cause of data breaches [7].

Data abuse is another major threat, which modifies the health parameters and compromises the data integrity. Since the treatment of the patients is based on the collected medical data, data modification and erroneous information lead to the loss of life [7].

This study reveals the number of challenges associated with IoT. Medical sensors that use wireless communication are exposed to different types of attacks like eavesdropping, man-in-the-middle attack, Denial of Services (DoS) and many more. Since any device can be connected to the network, it causes unauthorized access to the network. IoT devices and other sensors are resources constrained in terms of bandwidth and processing power, therefore, providing the security solution can decrease the efficiency of the device [5].

In order to accomplish an end to end security, sensors and client/server should have an implementation of the cryptographic algorithm. For the devices with limited resources, e.g., battery-powered device, a cryptographic algorithm which has a lower energy consumption is important [8].

An encryption algorithm is called lightweight if it utilizes less memory and machine cycle. When Tiny Encryption Algorithm is compared to other algorithms such as HEIGHT, KATAN, and KLEIN it uses less number of machine cycles and less memory. Since some health monitoring devices are powered by batteries, it must be ensured that selected cryptographic algorithm uses less energy. TEA algorithm uses less energy compared to previously specified algorithms [9].

The proposed paper is structured as follows: section 2 and 3 provides a brief introduction to related work and original TEA algorithm. Section 4 provides a description of the system model. Finally in section 5 modified TEA algorithm is explained.

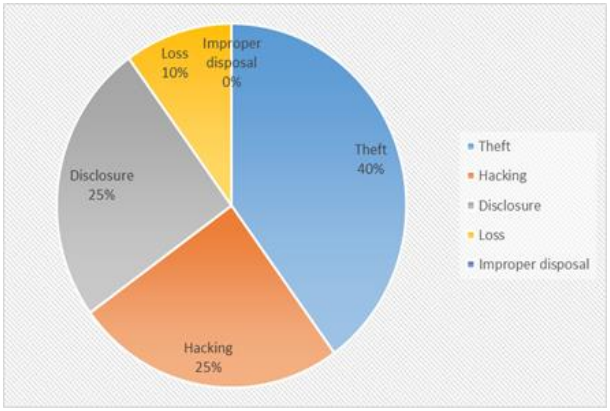

Fig. 1: Causes of a data breach.

2. Related work

Healthcare monitoring applications and devices deal with the private and confidential information such as healthcare data. Smart IoT devices are linked to the network to provide and access the collected data. IoT healthcare field is a major target for the attackers. To provide secure IoT healthcare environment, it is important to study the distinct features of IoT such as security, vulnerabilities, threats and possible countermeasures from the perspective of healthcare. An active attacker may plan different types of security threats to compromise current and future medical devices and network. Some attacks distort the information transmitted via the network. They may also perform interruption, interception and fabricate the original data [4].

Confidentiality is one of the main security requirement for the healthcare devices to maintain secrecy. Confidentiality ensures that all medical information's are inaccessible to unauthorized users. In addition, the confidential message does not reveal their true content or information to the eavesdropper. Symmetric encryption is generally used in the sensor network. Because when symmetric key is compared to asymmetric key, it is primarily less complicated, it needs a fewer number of operations and it provides the same level security as its asymmetric employing a smaller key size. Therefore, the symmetric key is more appropriate for low resources device [10].

TEA is used for the resource-constrained environments like sensor network. It is developed with few lines of code, and it does not make use complex operations [5].

In order to enhance the security of TEA against cryptanalysis, they propose to use Linear Feedback Shift Register (LFSR) as a PseudoRandom Number Generator (PRNG) for its ease of implementation using hardware and software [11].

RC5 is suitable for encryption in sensors and it has a lower computational power. It is most frequently used in WSN and it is susceptible to power analysis attack [12].

3. Tea algorithm

TEA algorithm is known to be one among the quickest and most effective cryptographic algorithms. TEA is a Feistel type cipher that uses simple operations such as ADD, XOR, and SHIFT to provide Shannon's properties of diffusion and confusion which is necessary to secure block cipher without the use of substitution boxes and permutation boxes. The source code of encryption module in TEA algorithm is shown in Figure 2. A double shift causes all the bits of the data and the key to be mixed constantly. TEA functions on 64-bit data using a 128-bit key and it can achieve complete diffusion after six rounds [10].

TEA algorithm takes plaintext and key as input. The given 64-bit plaintext is split into two halves, p= [Left (32), Right (32)], and the produced ciphertext is c = [Right (32), Left (32)]. In key schedule algorithm, the 128-bit key 'k' is fragmented into four 32-bit key blocks k0, k1, K2, k3. The keys k0 and k1 are used in the odd rounds and the keys K2 and k3 are used in even rounds. In each round, the data is processed by a round function which performs a set of basic operations to encrypt the data. Each half i.e. (left (32) or right (32)) is used to encrypt the other half of the 32 bit over 64 rounds and then combine to yield the 64-bit ciphertext block. The constant delta value is used to make sure that the subkeys are dissimilar and its value has no cryptographic significance. Delta value is derived from the golden number ratio [14] [16].

$$\text{Delta} = (\sqrt{5}-1)*231 = 9\text{E}3779\text{B}9\text{h} \quad (0)$$

```
def encrypt(v, k):
   p=c_uint32(v[0])
   q=c_uint32(v[1])
   sum=c_uint32(0)
   delta=0x9E3779B9 // golden number
   n=32
   F=[0,0]
   While (n>0):
   sum.value += delta
p.value += ( q.value << 4 ) + k[0] ^ q.value + sum.value ^
( q.value >> 5 ) + k[1]
q.value += ( p.value << 4 ) + k[2] ^ p.value +  sum.value ^
( p.value >> 5 ) + k[3]
   n -= 1
   F[0]=p.value
   F[1]=q.value
   return F
```

Fig. 2: Encryption module of TEA algorithm

The block diagram shows an abstract view of the ith cycle of TEA algorithm as shown in Figure 3[14]. There are two rounds in every cycle and TEA is considered to be secure after 32 cycle or 64 rounds [14]. Main advantages of TEA is that it is fast, lesser code size and uses less memory. Due to these advantages, TEA is suitable for the health monitoring devices which has low computational capabilities and requires fast throughput.

The most notable drawback that can be found in TEA is the equivalent key and associated-key attacks [13] [14] [15]. If a plain text 'p' is encrypted with two different keys K and K' and if it produces the same ciphertext then the two keys are called equivalent keys. It can be represented as

$$Ek(p) = Ek'(p)$$

The equivalent key in Table 1 shows that even though different keys are used to encrypt the same plaintext, it produces the same ciphertext[14]. In theory, when data is encrypted using different keys, for each key it should produce different ciphertext. Due to this drawback, it led to a technique of hacking Microsoft's Xbox gaming console [10]. TEA should still provide good security for remote health monitoring. In this paper, an equivalent key problem associated with TEA and a solution to overcome it is addressed.

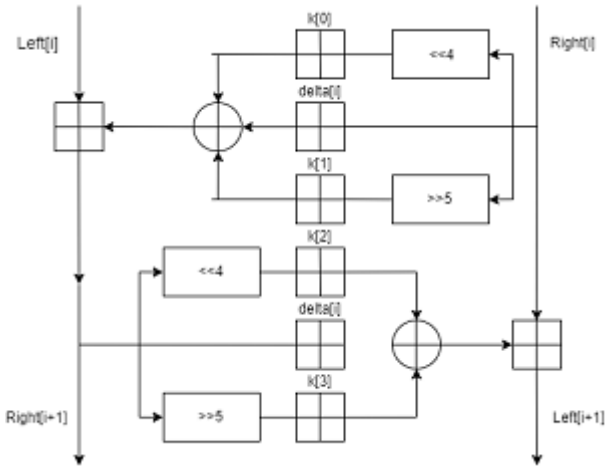

Fig. 3: i[th] Cycle of Tea Algorithm

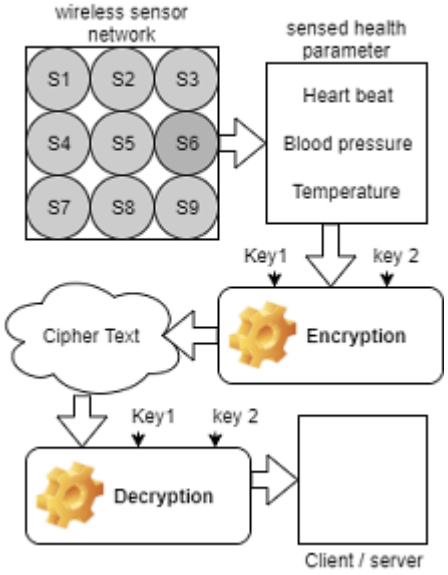

Fig. 4: System model

Table 1: Equivalent Key

Plain Test	Key	Cipher Text
0000000000000000	0000000080000000	246885698333653694
	0000000000000000	
0000000000000000	8000000000000000	246885698333653694
	0000000000000000	
0000000000000000	0000000000000000	246885698333653694
	8000000000000000	
0000000000000000	0000000000000000	246885698333653694
	0000000080000000	

4. System model

IoT uses several sensors that are used as a source of input and multiple clients that utilize these data. Wi-Fi and internet are used as a means for content sharing and distribution of data. An existing System model that is used in medical field to monitor patient's health is shown in Figure 4. Patients can use embedded or wearable sensors in their body to monitor their vitals and other parameters. Different types of sensors measure different health parameters such as body temperature, blood pressure, and heartbeat. These collected health parameters are transferred to the cloud or directly to the requested client. Cloud is where the collected data is stored, analyzed and transmitted to the client. During the transfer of the medical data, intruders may act as an intermediate to sniff and modify the health parameters. To provide data confidentiality and data integrity, several encryption algorithms are employed. Since the sensors are resource constraint, therefore symmetric encryption is used.

To provide a secure transmission between the sensor and the server data 'D' is encrypted using two keys 'Key 1' and 'Key 2' using a symmetric encryption algorithm 'E' and the equation is shown below (1). When data is being transmitted to the server from sensors, before transmitting, the data is encrypted using client key (Key 1) and server key (Key 2) to produce ciphertext 'C'. Using this method even if the intruder gets access to one of the keys, he needs both the keys to decrypt the data successfully. Since two keys are used to encrypt the data, it is hard for the intruder to use brute force and it is impossible to derive one key from another key.

$$C = E\ (D, \text{key}\ (\text{Key 1}, \text{Key 2})) \quad (1)$$

Initially, sensors convert the data into ciphertext with the help of keys and a symmetric algorithm. The server decrypts the received cipher with the same keys and same symmetric algorithm (2). The problem with this system is that server needs to distribute its key to all the sensors and the sensor needs to notify its key to the server before the data transfer. An encryption function can encrypt the same plaintext to different ciphertext depending on the key used. Therefore to get the right plain text from cipher text it is necessary to have the right keys.

$$D = E\ (\text{cipher text}, \text{key}\ (\text{Key 1}, \text{Key 2}) \quad (2)$$

In the initial configuration before any data transfer occurs following sequence of tasks are performed:

1. Key generation - When the wireless sensors and the server are started, both the parties will generate a private key.
2. Secure channel creation - A secure channel is established between the sensors and server so that secure communication can be achieved.
3. Key exchange - Server distributes its key to the all the sensors, and sensors, in turn, registers itself by sending its key to the server.

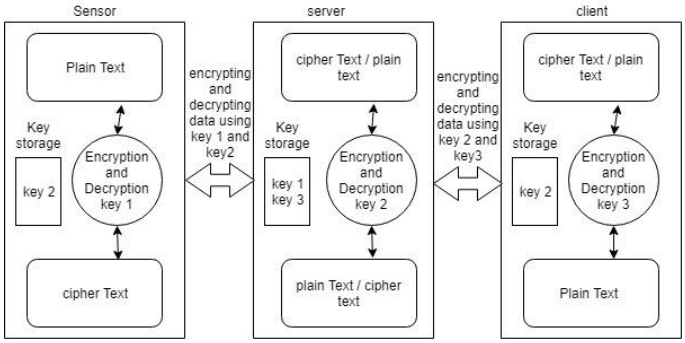

Fig. 5: Three level architecture model

The current model can be extended to three level architecture model shown in Figure 5. The data is sent from the sensors to sever, which can be housed locally or in the cloud and client access the data from the server. Before the data is exchanged, the sensors, server, and clients need to know each other's key. Each of the sensors, server, and the client will have key storage which contains the keys of the neighboring devices.

5. The proposed tea algorithm

In order to enhance the TEA security against the equivalent key and cryptanalysis, it is suggested to use two functions instead of one in each round. Each function uses a different key, Key-1 is used for the first function F and Key-2 for second function F'. The function consists of the bitwise left shift, right shift, addition and XOR operation.
Here the round function 'F' and 'F1' can be defined by

$$F(D, K[p, q], \text{delta}[i]) = ((D<<4) + k[p] \oplus (D + \text{delta}[i]) \oplus ((D >> 5) + k[q]) \quad (3)$$

where 'D' is data and 'K' is the key.
In the abstract structure of modified TEA show in Figure 6, it is see that 64-bit plaintext is divided into two 32-bit data i.e. Right (32) and Left (32). The right side of the 32-bit plain text is passed through the first round function F and the output is passed to the next round function F1. The output from the function F1 is ANDed with the left 32-bit plaintext to produce new left 32-bit data.
Again left 32-bit data is processed by round function F1 and the output is processed again by a function F, and output of function F is ANDed with the right 32-bit data. These two steps constitute one cycle. Each half of the 32-bit data is used to encrypt the other half of the 32-bit data over 64 rounds of processing to produce ciphertext. After the final iteration of the encryption process the two half of the resulting output is swapped, so that output of the final ciphertext will be right (32) || left (32). To produce final output we need to perform 32 cycles, which is 64 rounds. The general structure of the round function is same for every round but the key k[i] used will be different depending on the round. When using two functions with two different key, the chances of getting the equivalent key will be reduced.

Key schedule
In modified TEA key schedule is simple. A single 128-bit key is divided into four 32 bit key blocks. The Figure 7 shows the ith cycle of modified TEA algorithm where it uses two 128-bit key, and each 128-bit key will be divided to produce eight 32 bit key blocks. Key-1(128) will be divided into k1[0], k1[1], k1[2], k1[3], and Key-2(128) will be divided into k2[0], k2[1], k2[2], k2[3]. The keys k1[0], k1[1], k2[0], k2[1] will be used in the odd rounds and the keys k1[2], k1[3], k2[2], k2[3] will be used in the even rounds.

Decryption
Decryption is basically same as the encryption process. In deciphering routine the ciphertext is used as input to the algorithm. Subkeys k1[i] and k2[i] that are provided to the algorithm are used in the reverse order.

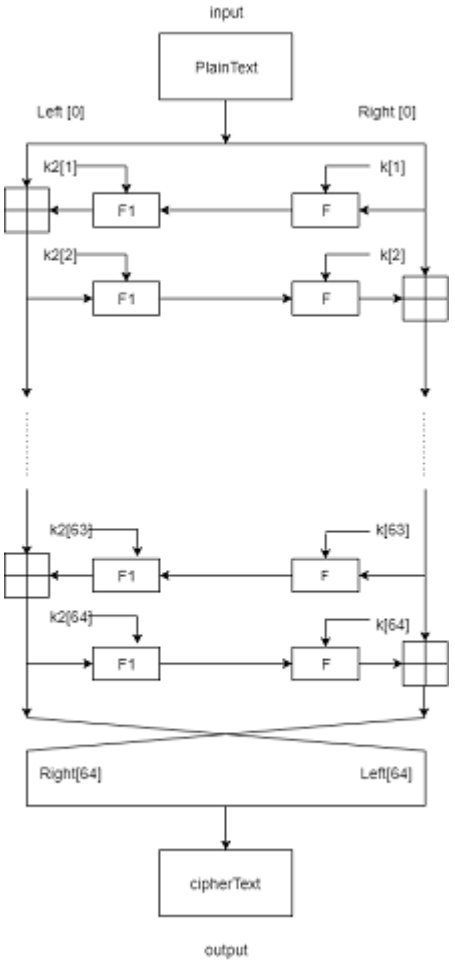

Fig. 6: Abstract structure of modified Tea algorithm

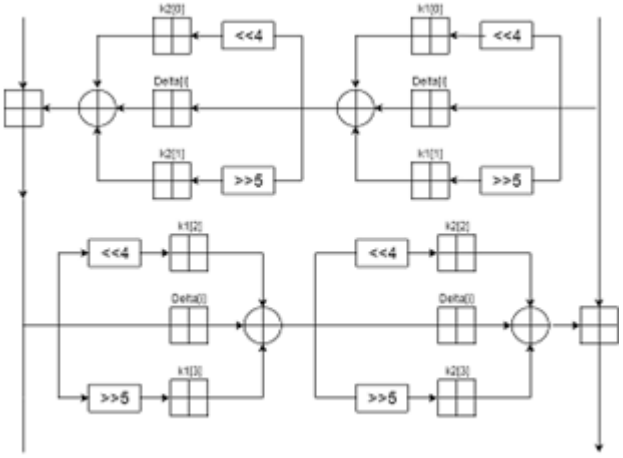

Fig. 7: i[th] cycle of modified TEA algorithm

6. Results

From the obtained results it is observed that modified TEA encryption algorithms eliminate the problem of the equivalent key by using two functions in each round instead of one and by using a different key for each function. When proposed TEA algorithm is compared to the original TEA algorithm it provides better security and a secure way to exchange data between the parties.
Original TEA algorithm produces the same ciphertext even if a different key is used to encrypt the plaintext shown in Table 1. This is due to the weakness of the round function in original TEA algorithm.

Table 2: Modified Tea with the same key for both the functions

PlainText	Key 1	Key 2	CipherText
0000000000000000	00000000 80000000 00000000 00000000	00000000 80000000 00000000 00000000	3993458279 1003555022
0000000000000000	80000000 00000000 00000000 00000000	80000000 00000000 00000000 00000000	3281545282 4173610317
0000000000000000	00000000 00000000 80000000 00000000	00000000 00000000 80000000 00000000	2472530398 1316777008
0000000000000000	00000000 00000000 00000000 80000000	00000000 00000000 00000000 80000000	2472530398, 1316777008

Table 3: Modified Tea with a different key for both the functions

PlainText	Key 1	Key 2	CipherText
0000000000000000	00000000 80000000 00000000 00000000	00000000 80000000 00000000 80000000	1777385544 1314195584
0000000000000000	80000000 00000000 00000000 00000000	00000000 80000000 00080000 80000000	602627954 3991741833
0000000000000000	00000000 00000000 80000000 00000000	00000000 80000000 56000000 80000000	3166073923 2773326125
0000000000000000	00000000 00000000 56080000 80000000	07300000 80000000 00000000 80000000	2406716869 1640107750

The proposed modified TEA encryption algorithm prevents the equivalent key condition. But the equivalent key condition cannot be evaded when same two keys are used for both the functions. This can be observed in Table 2. It is also shown that for some examples we still get the same ciphertext.

This can be overcome by using two keys. From the Table 3, it is seen that by using two different keys to encrypt a plaintext the problem of the equivalent key can be avoided. It also evident that modified TEA algorithm is secure against the chosen plaintext and ciphertext-only attack.

7. Conclusions

In this paper software implementation of modified TEA algorithm is proposed which uses two keys and two functions to overcome the security weakness and equivalent key drawback of a standard TEA algorithm. The modified TEA algorithm is compared with the standard TEA in the area of the equivalent key. The result shows that when using two functions in each round and each function utilizing one of two keys, chances of getting the same ciphertext for some plain text and key can be prevented.

In future, we plan to use AES key expansion algorithm to generate a different key, where each key generated can be used to encrypt the different data blocks.

Acknowledgement

The authors would like to thank the Christ University, Bengaluru, for supporting to complete the research.

References

[1] Islam, S. R., Kwak, D., Kabir, M. H., Hossain, M., & Kwak, K. S. (2015). The internet of things for health care: a comprehensive survey. *IEEE Access*, *3*, 678-708.

[2] Aledhari, M., Marhoon, A., Hamad, A., & Saeed, F. (2017, July). A New Cryptography Algorithm to Protect Cloud-Based Healthcare Services. In *Connected Health: Applications, Systems and Engineering Technologies (CHASE), 2017 IEEE/ACM International Conference on* (pp. 37-43). IEEE.

[3] Larburu, N., Bults, R., Van Sinderen, M., Widya, I. and Hermens, H. (2015). An Ontology for Telemedicine Systems Resiliency to Technological Context Variations in Pervasive Healthcare. IEEE Journal of Translational Engineering in Health and Medicine, [online] 3, pp.1-10. Available at: https://www.ncbi.nlm.nih.gov/pmc/articles/PMC4848059/

[4] Al Ameen, M., Liu, J., & Kwak, K. (2012). Security and privacy issues in wireless sensor networks for healthcare applications. Journal of medical systems, 36(1), 93-101.

[5] Luhach, A. K. (2016). Analysis of lightweight cryptographic solutions for Internet of Things. Indian Journal of Science and Technology, 9(28).

[6] Devi, K. N., & Muthuselvi, R. (2016). Secret Sharing of IoT Healthcare Data Using cryptographic algorithm. International Journal of Engineering Research Volume, (5).Y. Wang, G. Attebury, and B. Ramamurthy. (2006, Jun). A survey of security issues in wireless sensor networks. IEEE Commun. Surveys Tuts., vol. 8, no. 2, pp. 2-23.

[7] Healthcare Data Breaches (2015, Jun). Hippa Journal, Available: http://www.hipaajournal.com/2015-healthcare-data-breaches-pass-100-incident-milestone-7052/.

[8] Katagi, M., & Moriai, S. (2008). Lightweight cryptography for the internet of things. Sony Corporation, 7-10.

[9] Alizadeh, M., Hassan, W. H., Zamani, M., Karamizadeh, S., & Ghazizadeh, E. (2013). Implementation and evaluation of lightweight encryption algorithms suitable for RFID. Journal of Next Generation Information Technology, 4(1), 65

[10] Israsena, P. (2005, December). Design and implementation of low power hardware encryption for low cost secure RFID using TEA. In Information, Communications and Signal Processing, 2005 Fifth International Conference on (pp. 1402-1406). IEEE.

[11] Abdelhalim, M. B., El-Mahallawy, M., Ayyad, M., & ElHennawy, A. (2011, December). Implementation of a modified lightweight cryptographic TEA algorithm in RFID system. In Internet Technology and Secured Transactions (ICITST), 2011 International Conference for (pp. 509-513). IEEE

[12] Gong, T., Huang, H., Li, P., Zhang, K., & Jiang, H. (2015, December). A medical healthcare system for privacy protection based on IoT. In Parallel Architectures, Algorithms and Programming (PAAP), 2015 Seventh International Symposium on (pp. 217-222). IEEE.

[13] Kelsey, J., Schneier, B., & Wagner, D. (1996). Key-schedule cryptanalysis of idea, g-des, gost, safer, and triple-des. In Advances in Cryptology—CRYPTO'96 (pp. 237-251). Springer Berlin/Heidelberg.

[14] Andem, V. R. (2003). A cryptanalysis of the tiny encryption algorithm (Doctoral dissertation, University of Alabama).M. Young, *The Techincal Writers Handbook.* Mill Valley, CA: University Science, 1989

[15] Mala, H., Dakhilalian, M., & Shakiba, M. (2012). Cryptanalysis of mCrypton—A lightweight block cipher for security of RFID tags and sensors. International Journal of Communication Systems, 25(4), 415-426.

[16] Wheeler, D. J., & Needham, R. M. (1994, December). TEA, a tiny encryption algorithm. In International Workshop on Fast Software Encryption (pp. 363-366). Springer Berlin Heidelberg.