Research Article

Security Analysis of a Robust Lightweight Algorithm for Securing Data in Internet of Things Networks

Abdulrazzaq H. A. Al-Ahdal[a,b*], Galal A. AL-Rummana[a], Nilesh K. Deshmukh[c]

[a] School of Computational Sciences, S.R.T.M. University of organization, Nanded, India
[b]Assistant Lecturer Computer Science & Engineering, Hodeidah University, Yemen
[c]Assistant Professor, School of Computational Sciences, S.R.T.M. University of organization, Nanded, India

Article History: Received: 11 January 2021; Revised: 12 February 2021; Accepted: 27 March 2021; Published online: 23 May 2021

Abstract— Recently, the amount of important information and the tremendous sensitivity generated by the interconnectedness of millions of devices (embedded, wireless sensing, radio frequency identification) which are heterogeneous through promising such modern technologies as the IoT. In fact, these small computerized devices have limited capabilities of resources. Therefore, there is an extremely and dire need to protect and secure the sensitive data that generated by these devices taking into account their limited capabilities. However, the traditional coding algorithms like (AES and RSA) are not appropriate for these resource-limited devices due to their high computation cost and large memory consumption. In addition, the integrity and security of data should not be compromised by designing a simple encryption algorithm. Thus, a robust lightweight algorithm for encoding 64-bit data for an 80-bit key is suggested in this paper to provide high hardware security in just six rounds and work on a combination of Feistel and SP architectural methods to increase the encoding complexity. The suggested algorithm is simulated by FELICS and Matlab tools. Different data types such as text and images are used to implement this algorithm. The simulation results show the superiority of the proposed algorithm in different aspects as security, performance, and complexity.

Keywords—IoT, Lightweight Cryptography LWC; FELICS; MATLAB; RFID tags; feistel architecture; SP architecture

1. Introduction

Lightweight encryption is a branch of cryptography[1], [2]. It is uses to encrypt data on devices with limited resources used in the IoT. Cryptography implies' secret writing' [3]. Therefore, in computer communications, the data between receiver and the sender is encrypted and the message content is not known to anyone else. Security and lightweightness is the essence of LWC. The main problem is how to get a high level of security in small devices. The need to develop, improve traditional algorithms or to innovate new ones and implement them effectively on the basis of the lightweight has made the field of LWC of primary interest of the researchers.

The Internet allows users quick and easy access from anywhere and thus reduces the contact costs for users. Therefore, devices, sensors and smart devices are connected. Therefore, it provides a suitable environment for the IoT. In fact, there are major challenges to implementing the traditional coding standards for these resource-limited devices due to the increased correlation of small devicesand the amount of data produced. Many of the modern encryption algorithms have been improved for traditional desktop and server applications. The optimization of protection, efficiency and resources requirements make it difficult or impractical to incorporate such algorithms in resource-constrained devices.

Micro-controllers with a vast array of performance criteria are available. Although the most popular micro-controllers are 8-bit, 16-bit, and 32-bit, the sales of 4-bit micro-controllers are very low cost for some applications. There are other series of instructions that only produce a limited number of basic instructions. They carry in excess cycles when running common cryptographic algorithms. The expected design will get more energy-consuming and slower. RFID and sensors are sometimes used in systems where the timing and power specifications are rather strict [4].They are for devoted purposes only and their restrictions are tight. The algorithm they need has to meet their requirements as well.

We introduced a lightweight-complex symmetric key algorithm for IoT in our previous research work, entitled "A Robust Lightweight Algorithm for Securing Information in the Internet of Things Networks," and compared it to several current symmetric key algorithms based on design, usability and level of security [5].

2. Resource-constrained Device Security Challenges

There are several fields of implementation for resource-constrained devices: smart parking, vehicle applications, sensor networks, distributive control systems, disaster / weather monitoring, infrastructure, IoT, smart grid, cyber-physical services, smart cities, etc. To adopt modern technology and connect it with small computing devices, attention must be paid to privacy and security in order to establish trust between users [6]. Resource of restricted devices is inherently defenseless to several kinds of security threats.

IoT devices are more vulnerable to attacks because of the complexity of linking devices in the IoT, as well as being unsupervised for a long time [7]. In comparison, eavesdropping is easier with respect to the wireless communication medium, but such systems have minimal computing capacity and energy consumption capabilities [8]. Therefore, the use of traditional encryption algorithms will hinder the operation of devices (their computationally high costand energy consumption).

From a high-level perspective, there are three parts to creating the IoT: devices (sensors and drivers), middleware (storage and computing tools), and presentation (simultaneous translation tools) [10]. Collecting data from millionsof sensors (not all of them can be processed), solutions are proposed to handle only the important ones identified by the sensor [8]. IoT devices such as RFID are in confusion to achieve essential verification process requirements which include constant server communication and node exchange of messages.

Data security is retained in protected networks, where it is assured that the data maintains its originality throughout the process of transmission of messages,and the device does not see any modification. The IoT consists of several small sensors such as RFIDs that remain unattended for longer periods of time, making it possible for the competitor to access the data contained in the memory [9], [11]. Methodologies based on the meaning of obtained signal intensity (RSSI) are used in [12], [13], [14] and [15] to provide protection to attacks such as Sybil in RFID-tags.

Several approaches are proposed for WSN which regard the sensor as part of the internet linked through nodes [16]. Furthermore, the sensor nodes themselves are known as network nodes in IoT, making the task of authentication even more important. Information accuracy is also necessary to ensure the authenticity of the data.

3. Motivation

Communication technologies and data exchange in networks is one of the most important matters for researchers. Therefore, when exchanging messages in communication networks, it must be simple and secure. This means that the use of encryption is compulsory when sending / receiving messages. This is to ensure that the messages arrive without any interference or penetration from another person and the insurance of the integrity of the data. In this context, there is data processing and security by traditional encryption algorithms.These algorithms need large mathematical operations and need a large memory. This does not make them suitable for encryption on Internet of things devices. However, these devices are widely used in at the present time in the Internet of things that are using lightweight encryption for balance between performance and security. In addition to that, the study and survey provided by HP shows that 70% of these devices are liable to attackers [17].

The remaining of the paper is structured as; a brief literature review of past and cryptographic algorithms is given in Section 4. The introduction of (RSIT) algorithm is discussed in Section 5. Section 3 discusses implementation of the (RSIT) algorithm. The parameters for evaluating success are explained in this section, and all simulation findings are presented and analyzed, based on assessment criteria. The conclusion is ultimately drawn in Section 4.

4. Literature Review

Lightweight block ciphers in the form of the key size and block size have comparably smaller internal states. Feistel round features are quick, and require more rounds. An SP-network or SPN in cryptography is a sequence of mathematical operations that are related. In [17],[18], and [19],the need for LWC has been widely discussed, and the IoT weaknesses in terms of restricted devices have also been highlighted. In fact, there are some LWC algorithms that do not always take advantage of trade-offs on safety-effectiveness. The block ciphers have provided considerably stronger results than stream cipher, and hash functions.

In [20], symmetric block cipher using 64-bit key, over 64-bit data is proposed. In this proposal, block ciphers like AES uses the network of SP to merge Shannon's uncertainty properties with diffusion properties, and there are other ciphers like Blowfish and DES which use architecture of Feistel to gain the same mechanism for encryption and decryption.

Paper [21] examines the consistency and protection of various types of LWC algorithms used in resource-constrained applications in particular. On AVR Atmel ATtiny45 micro-controllers, HIGHT, TEA, KLEIN, KATAN algorithms are introduced to test performance analyzes on their processing ability and usage of resources, and to determine the degree of ambiguity and diffusion for security research.

In [22], Simon and Speck have suggested a lightweight block cipher to show, respectively, maximal hardware and software performance. Both ciphers have a variety of key size and width, but at least 22 round numbers are needed for adequate encryption. Since Simon focuses on low complexity of multiplication, the total number of mathematical operations needed is fairly high [23],[24].

Mica2 hardware platform has introduced TEA [25], Skipjack [26] and RC5 algorithms [27]. The Mica2 ciphers were designed in single mote to calculate the energy consumption and power usage. Many block ciphers have been introduced like AES [28], XXTEA [29], Skipjack and RC5 [30], calculating speed of execution and the energy consumption. The findings show that the size of the key in the AES algorithm greatly impacts the encryption, decryption and key setup phases I with the longer key size resulting in increased execution process. RC5 contains varying requirements for key type, number of rounds and word size may be changed. Authors have done a number of variations to figure out how the word size is expanded, it takes more time for implementation. Since the main setup phase does not concern itself with Skipjack and XXTEA algorithms, they become less secure than (RC5, AES) and use less energy.

SPN was suggested for the lightweight block cipher, PRESENT [31]. PRESENT is a 64-bit block cipher, with 80 or 128 key lengths. It features 31 rounds. The sheet of replacement adds 16 4×4 S-boxes. Both the S-box and the 64-bit permutation are however evolutionary. It yielded good hardware and software performance.

In 2010 Wu and Zhang [32] proposed a lightweight block (LBlock) which would use Feistel Structure. TheLBlock block has a block size of 64-bit and an 80-bit key capacity. The LBlock block has three functions: encryption, decryption, and scheduling. Within LBLOCK each round has two circular functions: uncertainty and diffusion (permutation of eight four-bit words). Hence, the Feistel layout is ideal for reducing S-box number and size. LBlock's one round splits data into two parts: the first half goes into circular process, and the second half applies basic rotation procedure, thus, the more circular iteration leads to margins of protection.

Piccolo is a modern lightweight block cipher, according to Shibutani et al.[33]. It has block size of 64-bit and the range of keys 80- and 128-bit. Because of its architecture is based on half-word round permutation and permutation for key expansion, Piccolo achieves low power consumption and high protection. Therefore, it suited for FRID applications.

TWINE is a 36-round lightweight block cipher. It was proposed with 64-bit block size and 80/128-bit key size [34]. In contrast, in [35], the author re-evaluated the protection of the TWINE-80 against impossible differential cryptanalysis in related-key model. Then, a single round assured the traditional impossible differential attack.

5. A Robust Lightweight Algorithm for Securing Data in Internet of Things Networks (RSIT)

The algorithm in [5] is a mixture of Feistel and SP networks to achieve the protection required for creating lightweight encryption algorithms. Proposed algorithm design offers architecture for implementation in IoT that is small in complexity and has 80 bit keys for 64 bit data. In the encryption process, the number of rounds was reduced to six cycles only in order to increase the energy efficiency, as suggested in [6]. For security improvement, each encryption round shall include four simple mathematical operations that operate on data of only four bit (Designed to be compliant with 8-bit IoT computing devices). This is to create a sufficient amount of confusion and data dissemination to be up against various types of attacks. The key is the algorithm's principal feature (encryption / decryption). For security reasons, the size of the encryption key is very important. Thus, the (key size) becomes a major obstacle for the attacker to be known. Consequently, confusion and diffusion (key generation) reduces the possibility of weakening the key, increasing security, the complexity of encryption, and the attackers' lack of knowledge of the key.

- **Key Expansion Block**

The primary method used to produce different encryption and decryption keys is defined as key expansion. Creating diffusion and confusion is very important in the key expansion process. Therefore, it increases the strength of the key and reduces the possibility of weakness. There are several processes used before deriving the round keys to increase the creation of very strong confusion and diffusion. The process is composed of two components: before encryption process, and key expansion process. In the before encryption process, the user generates two keys with simple mathematical and logical operations derived from the master key. The first key (Kt) is an introduction to the key expansion process. The other key (Ks) is sent to open the encryption process. This is illustrated in Figure (1) in the expansion process, the main expansion performs logical operations (XOR, XNOR), permutation of P-table and Q- table.

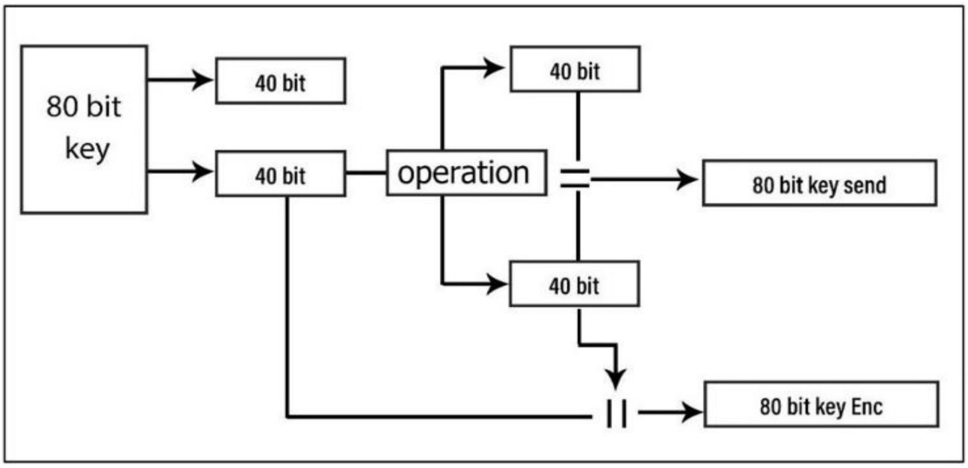


Fig.1: Before Encryption Process

- **Encryption Block**

Once the keys (Kt) are provided by the key expansion block, the encryption process begins. Basic operations to create confusion and diffusion are performed in the encryption process, namely substitution (S boxes), swapping, XOR, and XNOR procedures.

- **Decryption Block**

It is composed of two components: before decryption process, and decryption process. In before decryption process, the user receives the sent key; then, the user extracts the key, that the encryption process has mathematical and logical operations. After that, it is sent to open the encryption process as shown in the Figure (2) the decryption method is just the reverse of the above mentioned encryption technique.

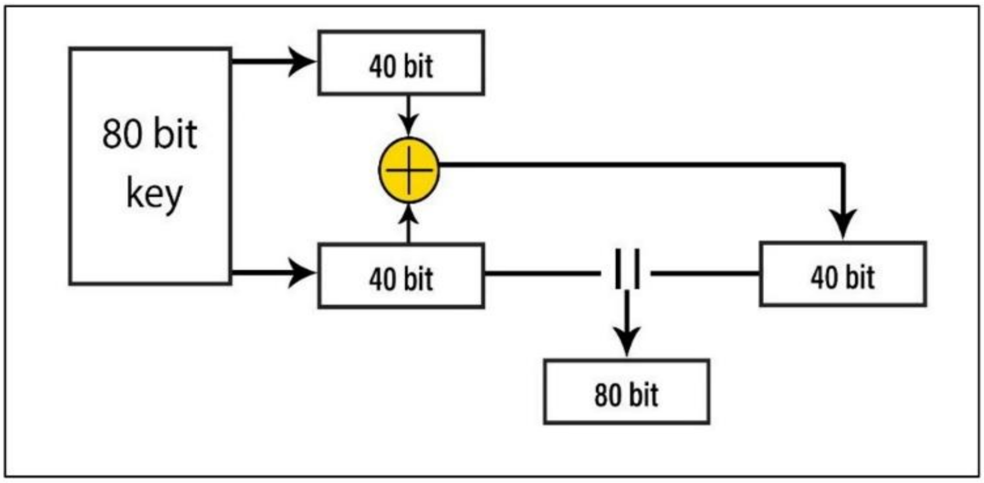


Fig.2: Before Decryption Process

6. Performance Evaluation Criteria

The suggested algorithm is written using Dev-C++.Dev-C++ which is a fully working, open, optimized software environment for C and C++ programming licensed under the GNU General Public License. The FELICS (Fair Evaluation of Lightweight Cryptographic Systems) tool is a simple used to test execution cycles (encryption/decryption), RAM usage, and program size [43].This tool implements LWC of various regular andcommon types like TWIN, PICCOL, HIGHT, AES and others and in any different versions.

FELICS offers interface in order to make the implementation as well as comparison of any new algorithms easier. It functions on Ubuntu Linux. Both Linux Ubuntu and FELICS work on a virtual machine program so that the user can install all possible requirements. In addition, the suggested algorithm is implemented in language of MATLAB, for evaluating security strength.

Since this RSIT is a mixture of Feistel and SP networks it benefits from security research in (weak keys, related keys, linear and differential cryptanalysis, SQUARE attack, and interpolation attacks). The security review of these algorithm is mentioned in [5], addressing their importance through the proposed algorithm. There are basic criteria for checking the quality of security in the proposed algorithm, as follows:

A. Key Space

When using a new technology, we must focus on determining the size of the key and all the possible keys used to generate it. This means that we should take into consideration the numbers of potential possibilities that may appear in hackers' minds to get the correct key. This process is called key space, which is the most important feature that supports the size and robustness of the encryption system. The large key space of the key in the cipher system plays an effective resistance against hacker's attacks and brute force attacks. In our proposed algorithm, we notice that the key space appears to be strong and this is what is observed during the key generation process. The key generation process takes start after inserting a key consisting of 80 bits and from this key six different keys are generated. These keys provide strong protection for the plaintext with process of rotations in Algorithm. The method of generating the keys is done by dividing the entry key into four blocks. Each block contains five blocks and each of these five contains four bits. Then, each of the first four bits of blocks is combined into one block and then the second bit until the fifth bit is reached so that we have five new blocks, every block is 16 bits.

Some functions are added to these five blocks. Regarding the sixth key, it is generated from the five keys with a simple process called XOR. Finally, we can say the proposed algorithm has the largest key space as $2^{\wedge 80}$ -$2^{\wedge 480}$, which means that there is no chance for the brute force attack to break the proposed algorithm [36].

B. Key Sensitive

A key-sensitive algorithm must be used for encryption. This means that if the key still has a minute deviation from the original version, the algorithm doesn't recover the original data. Avalanche check is used to modify a bit of the key or plain text in order to determine the number of modifications that happened in the cipher. The test is considered good in case fifty per-cent of the bits are changed as a result of one bit of change, according to the Strict Avalanche Criterion SAC [37]. To detect this result visually, we decode the picture using a key that varies by a single bit from the right one.

C. Execution Cycle

The most important metric for calculating the performance of the algorithms is the amount of time it takes to encrypt and decode the data supplied. With respect to resource-constraint systems, the proposed algorithm will consume minimal cycles and give the desired protection.

D. Memory Utilization

In IoT devices, memory use is a major concern that limits resources. The use of memory is a big problem for space constraint in IoT systems. An encryption algorithm consists of several cycles of computations that occupy a significant memory, which makes it inappropriate for IoT use. Therefore, the number of cycles is small, and uses simple mathematical operations in the proposed algorithm, which makes its efficiency high, and requires a small memory in IoT devices.

E. Histogram

Analysis of histogram of an image means visual portrait of the values of the pixel strength and shows the tonal distribution of an image [38], [39]. In short, it offers the image intensity statistical features from which image may be visible [40], [41]. The principal objective of the histogram analysis is to show the properties of the confusions and diffusions in the ciphered data. Nevertheless, when encrypting an image a histogram will quantify the randomness. Unless the measured histogram after encryption is consistent, a cryptographic algorithm refers to sufficiently security.

F. Entropy

The algorithm of encryption applies extra information to the data such as differentiating between the initial information and the one introduced by the algorithm, which is complicated for the attacker. The entropy of image is a quantity that is used to characterize the amount of data that an encryption algorithm needs to decrypt the extra information. Therefore, obtaining high entropy is considered important to have greater security of the algorithm for encryption. To measure entropy (H) on an image, equation (6) is based on intensity (x_i) values $P(x_i)$ which is the intensity value x_i probability.

$$\mathrm{Entropy(H)} = \sum_{i=0}^{255} P(x_i) \log_2 P(x_i) \qquad (1)$$

G. Correlation

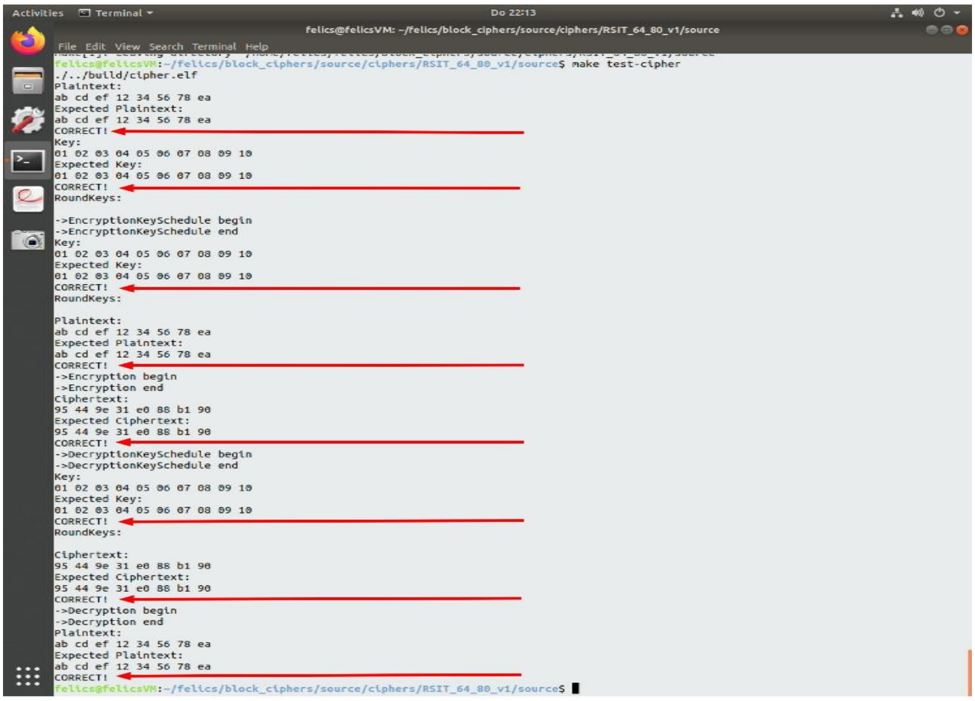
The relation between two neighboring pixels is calculated by the pixel correlation. In general, the image encryption scheme's excellence is expressed in its capacity to cover all the properties of a hidden image, and the encrypted image is absolutely random and extremely uncorrelated [42]. In the same image the correlation coefficient of any two adjacent pixels can be expressed in relation (2):

$$\mathrm{CorrCoeff} = \frac{\mathrm{cov}(x,y)}{\sqrt{\mathrm{Var}}(x) \times \sqrt{\mathrm{Var}(y)}} \qquad (2)$$

$$\mathrm{Var}(x) = \frac{1}{N}\sum_{i=1}^{N}[(x_i - E(x))^2] \text{ and } \mathrm{Cov} = \frac{1}{N}[(x_i - E(x)) \times (y_i - E(y))]$$

Where, the correlation coefficient is CorrCoeff and the covariance of $\mathrm{Cov}(x,y)$ is pixel x andy.$\mathrm{Var}(x)$ is the pixel value variance in an image, $E(x)$ is the value operator predicted and N is the total number of pixels in the matrix.

FELICES provides a command line interface to test and create any LWC code, like GCC (GNU Complier Collection). They have provided regular and common lightweight cipher already. The algorithm should be written in different format for the checking of some other cipher. They have provided documentation to make implementation easier. Whether or not the algorithm can be tested in FELICS, anyone can compile its implementation and check. It is a very useful tool which is highly recommendable. An example of a run is shown in Figure (3).


Activities
Terminal
Do 22:13
felics@felicsVM: ~/felics/block_ciphers/source/ciphers/RSIT_64_80_v1/source
File Edit View Search Terminal Help

```
felics@felicsVM:~/felics/block_ciphers/source/ciphers/RSIT_64_80_v1/source$ make test-cipher
./../build/cipher.elf
Plaintext:
ab cd ef 12 34 56 78 ea
Expected Plaintext:
ab cd ef 12 34 56 78 ea
CORRECT!
Key:
01 02 03 04 05 06 07 08 09 10
Expected Key:
01 02 03 04 05 06 07 08 09 10
CORRECT!
RoundKeys:

->EncryptionKeySchedule begin
->EncryptionKeySchedule end
Key:
01 02 03 04 05 06 07 08 09 10
Expected Key:
01 02 03 04 05 06 07 08 09 10
CORRECT!
RoundKeys:

Plaintext:
ab cd ef 12 34 56 78 ea
Expected Plaintext:
ab cd ef 12 34 56 78 ea
CORRECT!
->Encryption begin
->Encryption end
Ciphertext:
95 44 9e 31 e0 88 b1 90
Expected Ciphertext:
95 44 9e 31 e0 88 b1 90
CORRECT!
->DecryptionKeySchedule begin
->DecryptionKeySchedule end
Key:
01 02 03 04 05 06 07 08 09 10
Expected Key:
01 02 03 04 05 06 07 08 09 10
CORRECT!
RoundKeys:

Ciphertext:
95 44 9e 31 e0 88 b1 90
Expected Ciphertext:
95 44 9e 31 e0 88 b1 90
CORRECT!
->Decryption begin
->Decryption end
Plaintext:
ab cd ef 12 34 56 78 ea
Expected Plaintext:
ab cd ef 12 34 56 78 ea
CORRECT!
felics@felicsVM:~/felics/block_ciphers/source/ciphers/RSIT_64_80_v1/source$
```

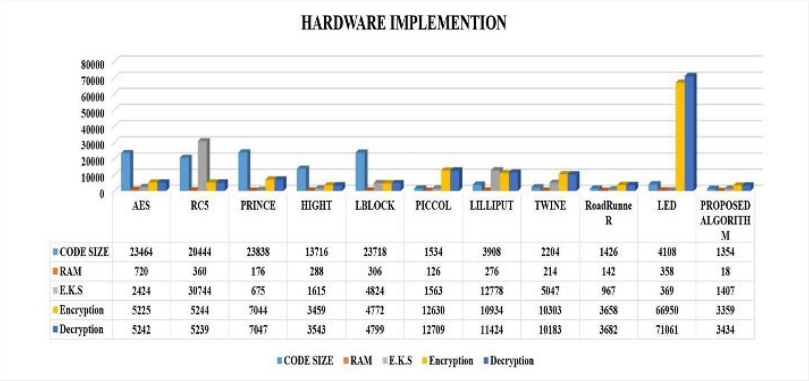
Fig. 3: Implementation Test of the Proposed Algorithm on FELICS

For LWC, the algorithm simulation is carried out by the famous FELICS open source benchmark tool. This uses various platforms for performance assessment (like AVR, ARM, and MSP) although typically under different conditions. It can also calculate the execution times (encryption/decryption), code size and RAM footprint. The new encoder can be compared to the previous one easily. TABLE 1 demonstrates contrast between various Lightweight Hardware Algorithms.

TABLE 1: Results for Cipher Implementations on AVR Architecture.

Cipher	Device	Block Size	Key Size	Code Size	RAM	Encry- Key Schedule	Encry- cycles	Decry-cycles
AES	AVR	128	128	23464	720	2424	5225	5242
RC5	AVR	64	128	20444	360	30744	5244	5239
PRINCE	AVR	64	128	23838	176	675	7044	7047
HIGHT	AVR	64	128	13716	288	1615	3459	3543
LBLOCK	AVR	64	80	23718	306	4824	4772	4799
PICCOL	AVR	64	80	1534	126	1563	12630	12709
LILLIPUT	AVR	64	80	3908	276	12778	10934	11424
TWINE	AVR	64	80	2204	214	5047	10303	10183
RoadRunneR	AVR	64	80	1426	142	967	3658	3682
LED	AVR	64	80	4108	358	369	66950	71061
PROPOSED ALGORITHM	**AVR**	**64**	**80**	**1354**	**18**	**1407**	**3359**	**3434**

Comparisons between different lightweight algorithms along with the proposed algorithm are shown in the bar



	AES	RC5	PRINCE	HIGHT	LBLOCK	PICCOL	LILLIPUT	TWINE	RoadRunneR	LED	PROPOSED ALGORITHM
■CODE SIZE	23464	20444	23838	13716	23718	1534	3908	2204	1426	4108	1354
■RAM	720	360	176	288	306	126	276	214	142	358	18
■E.K.S	2424	30744	675	1615	4824	1563	12778	5047	967	369	1407
■Encryption	5225	5244	7044	3459	4772	12630	10934	10303	3658	66950	3359
■Decryption	5242	5239	7047	3543	4799	12709	11424	10183	3682	71061	3434

Fig.4: Comparison for Hardware Implementation.

chart in Figure (4) the distinctions are made on the basis of number of cycle's encryption and decryption, as well as file size and RAM. The chart reveals that the suggested algorithm operates in less code size, RAM, fewer loops, gaining significantly over the others. The proposed algorithm takes the red color in Figure (5, 6, 7, and 8) respectively for the code size, RAM, and the number of encoding and decoding cycles.

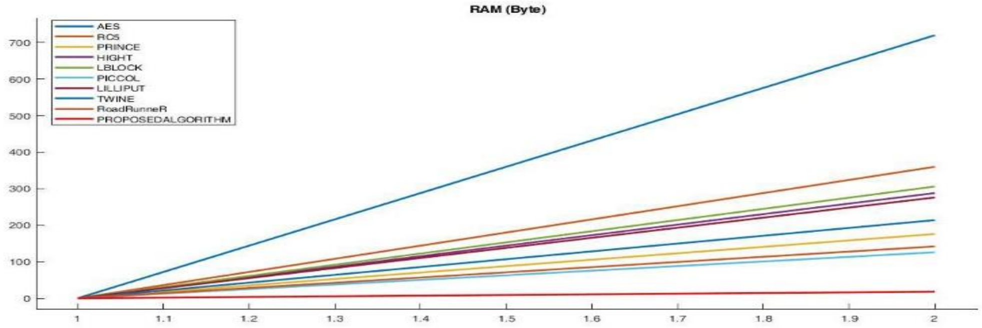


Fig.6: RAM Curve for Different Cipher.

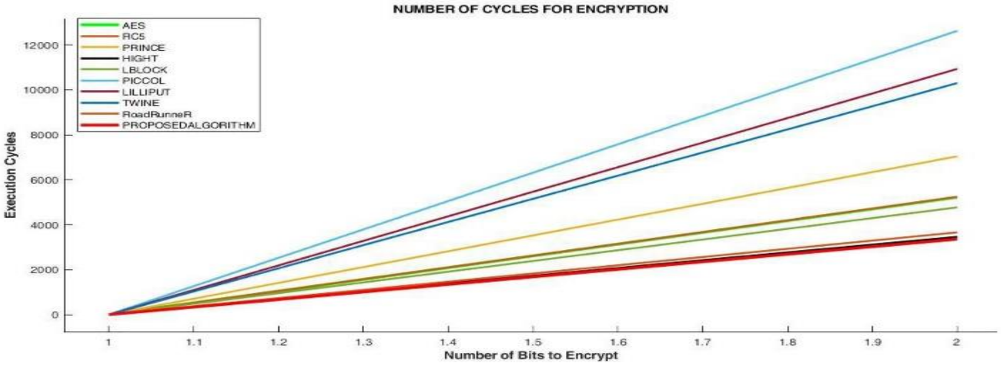


Fig.7: Cycle Curve Execution for various Cipher (Encryption).

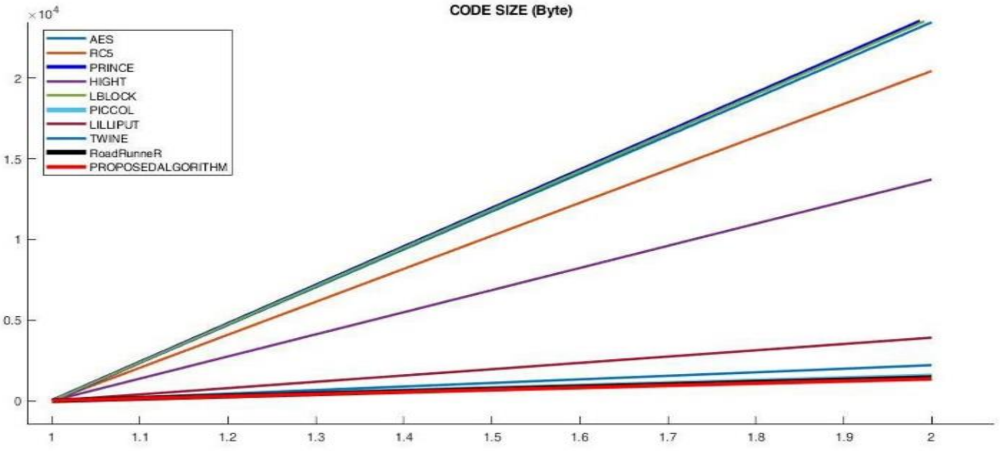


Fig.5: Code Size Curve for Different Cipher.

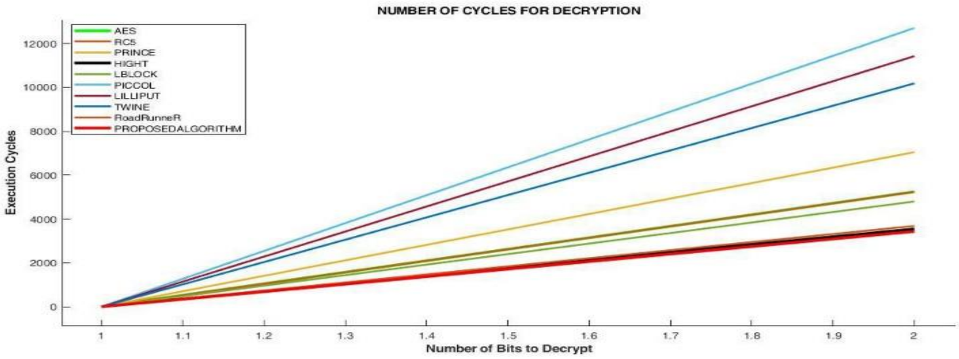


Fig.8: Cycle Curve Execution for various Cipher (Decryption).

Figure (9) we note the decrypted data with the correct key for visual encryption-decryption observation. The algorithm's avalanche check means that a single bit of key change, the plaintext brings about improvement of forty-nine per-cent in cipher bits. The decryption is unrecognizable if the original keys changed even one bit. From this result, the power of the algorithm can be interpreted.8-bit grey scale images is selected to perform entropy and histogram tests. Additionally, in the histogram appears in Figure (10) the uniform distribution of intensities after encoding, for the original and encoded file, indicates the desired security. An illustration with an 8-bit grey scale will achieve maximum entropy with 8 bits. It can be seen from the results in TABLE 2 that all encrypted images have an entropy close to maximum, reflecting an algorithm feature.

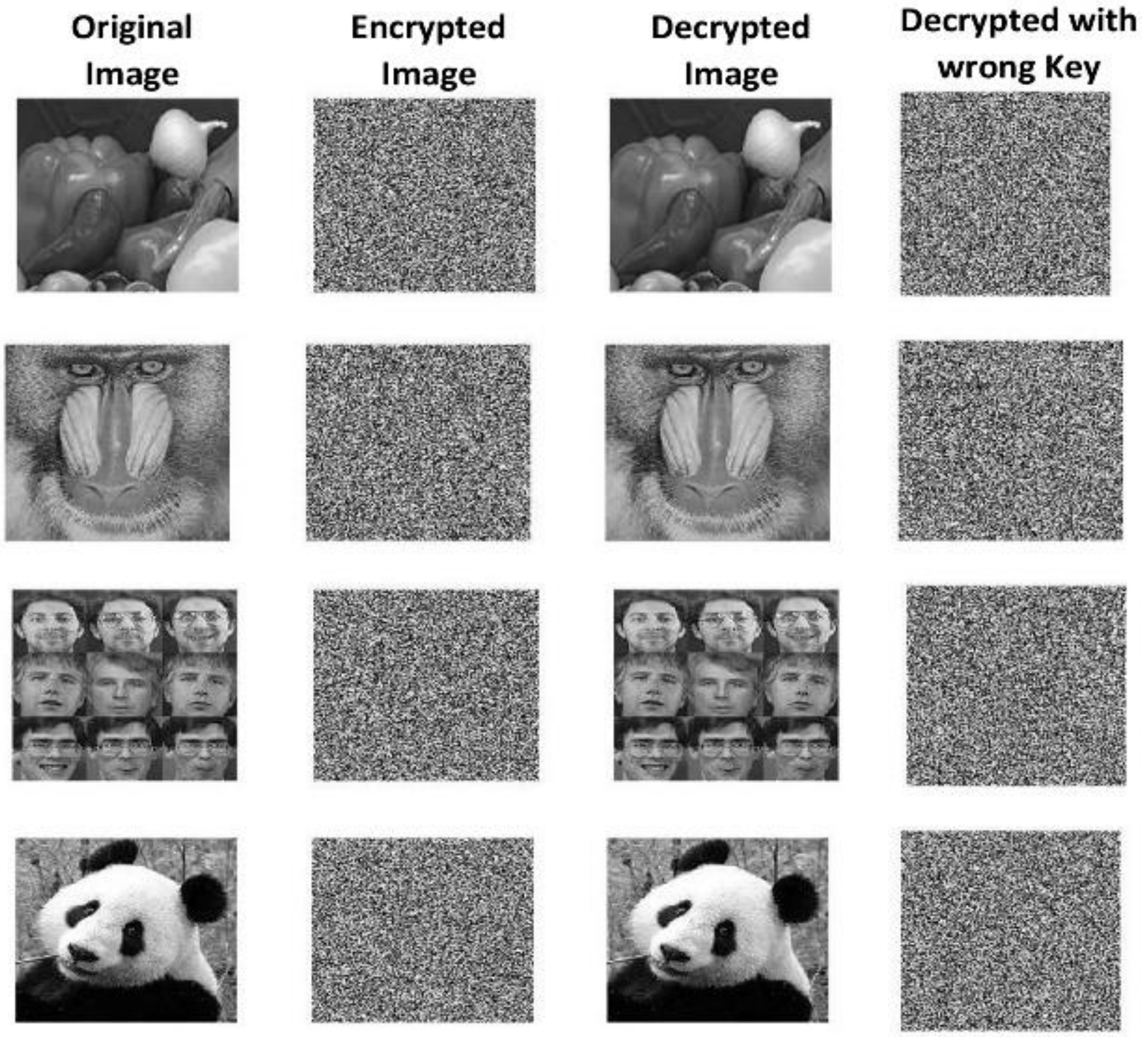


Fig. 9: Image Decryption and Key Sensitivity

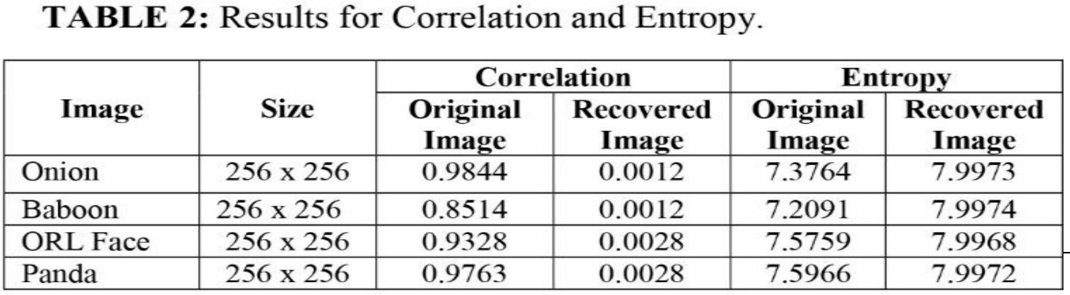


TABLE 2: Results for Correlation and Entropy.

Image	Size	Correlation		Entropy	
		Original Image	Recovered Image	Original Image	Recovered Image
Onion	256 x 256	0.9844	0.0012	7.3764	7.9973
Baboon	256 x 256	0.8514	0.0012	7.2091	7.9974
ORL Face	256 x 256	0.9328	0.0028	7.5759	7.9968
Panda	256 x 256	0.9763	0.0028	7.5966	7.9972

The correlation, the diagrams in Figure (11) display the contrast of the original images to the encrypted pictures. The initial image displays strongly correlated significance while the encrypted image tends to have minimal correlated importance. More connection gives the intended intent greater protection.

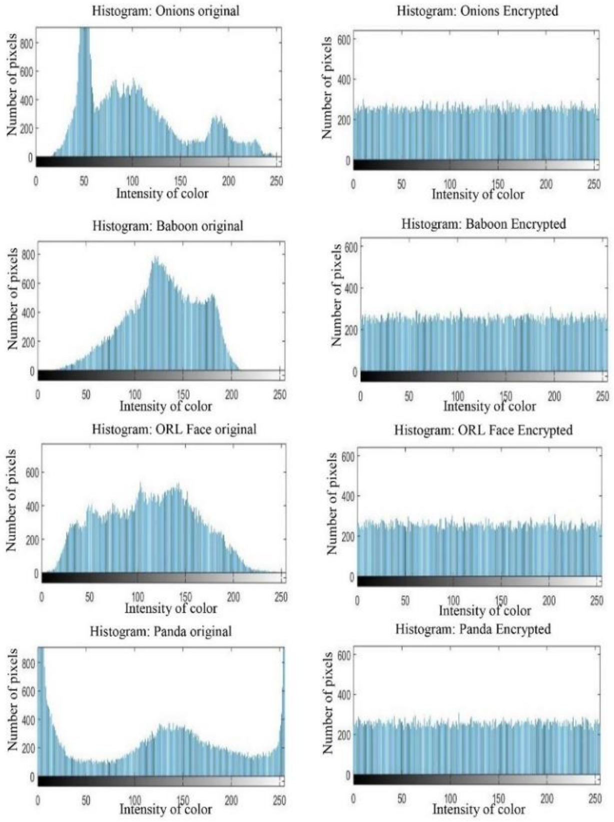


Fig. 10: Histogram comparison

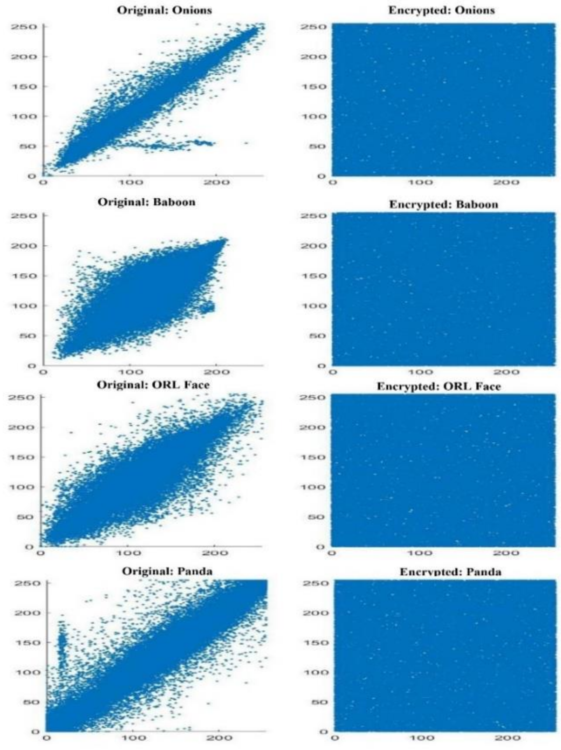


Fig. 11: Correlation comparison

7. Conclusion

The IoT will be important in our everyday lives in the near future. Numerous energy-constrained machines and sensors must communicate with each other constantly, the reliability of which should not be breached. This is requiring algorithms that can provide robust protection at an inexpensive mathematical cost. In this paper, we have implemented the proposed algorithm architectures. Furthermore, it shows results in the code size, RAM, and execution time, good results from other algorithms to be lightweight and candidate in Internet of Things devices. This work assists IoT Security researchers.

REFERENCES

1. K. A. McKay, L. Bassham, M. S. Turan, N. Mouha, “Report on Lightweight Cryptography”, National Institute of Standards and Technology, USA, March 2017.

2. Al-ahdal, Abdulrazzaq HA, and Nilesh K. Deshmukh. "A Systematic Technical Survey of Lightweight Cryptography On loT Environment."

3. B. Schneier, Applied Cryptography: protocols, algorithms, and source code in C, john wiley& sons, 2007.

4. L. Khelladi, Y. Challal, A. Bouabdallah, N. Badache, “On Security Issues in Embedded Systems: Challenges and Solutions”, International Journal of Information Security, Inderscience, 2008, 2 (2), pp.140-174.

5. Al-ahdal, Abdulrazzaq HA, and Nilesh K. Deshmukh, and Galal A. AL-Rummana. "A Robust Lightweight Algorithm for Securing Data in Internet of Things Networks."

6. H.J. Ban, J. Choi, and N. Kang, “Fine-grained support of security services for resource constrained internet of things”, International Journal of Distributed Sensor Networks, vol. 2016, 2016.

7. P. Wang, Professor S. Chaudhry, S. Li, T. Tryfonas and H. Li, “The internet of things: a security point of view”, Internet Research, vol. 26, no. 2, pp. 337-359, 2016.

8. S. Wang, Z. Zhang, Z. Ye, X. Wang, X. Lin, and S. Chen, “Application of environmental internet of things on water quality management of urban scenic river”, International Journal of Sustainable Development & World Ecology, vol. 20, no3, pp. 216-222, 2013.

9. F. Xie and H. Chen, “An efficient and robust data integrity verification algorithm based on context sensitive,” way, vol. 10, no. 4, 2016.

10. J. Gubbi, R. Buyya, S. Marusic, and M. Palaniswami, “Internet of things (iot): A vision, architectural elements, and future directions,” Future Generation Computer Systems, vol. 29, no. 7, pp. 1645–1660, 2013.

11. T. Karygiannis, B. Eydt, G. Barber, L. Bunn, and T. Phillips, “Guidelines for securing radio frequency identification (rfid) systems,” NIST Special publication, vol. 80, pp. 1–154, 2007.

12. J. Wang, G. Yang, Y. Sun, and S. Chen, “Sybil attack detection based on rssi for wireless sensor network,” in 2007 International Conference on Wireless Communications, Networking and Mobile Computing. IEEE, 2007, pp. 2684–2687.

13. S. Lv, X. Wang, X. Zhao, and X. Zhou, “Detecting the sybil attack cooperatively in wireless sensor networks,” in Computational Intelligence and Security, 2008. CIS’08. International Conference on, vol. 1. IEEE,2008, pp. 442–446.

14. Y. Chen, J. Yang, W. Trappe, and R. P. Martin, “Detecting and localizing identity-based attacks in wireless and sensor networks,” IEEE Transactions on Vehicular Technology, vol. 59, no. 5, pp. 2418–2434,2010.

15. S. Chen, G. Yang, and S. Chen, “A security routing mechanism against sybil attack for wireless sensor networks,” in Communications and Mobile Computing (CMC), 2010 International Conference on, vol. 1. IEEE, 2010, pp. 142–146.

16. L. Eschenauer and V. D. Gligor, “A key-management scheme for distributed sensor networks,” in Proceedings of the 9th ACM conference on Computer and communications security. ACM, 2002, pp. 41–47.

17. M. Katagi and S. Moriai, “Lightweight cryptography for the internet of things,” Sony Corporation, pp. 7–10, 2008. S. A. Kumar, T. Vealey, and H. Srivastava, “Security in internet of things: Challenges, solutions and future directions,” in 2016 49th Hawaii International Conference on System Sciences (HICSS). IEEE,2016, pp. 5772–5781.

18. M. Ebrahim, S. Khan, and S. S. U. H. Mohani, “Peer-to-peer network simulators: an analytical review,” arXiv preprint arXiv:1405.0400, 2014.

19. M. Ebrahim, S. Khan, and U. B. Khalid, “Symmetric algorithm survey: A comparative analysis,” International Journal of Computer Applications (0975 – 8887), vol. 61, no. 20, 2014.

20. M. Usman, I. Ahmed, M. I. Aslam, S. K. and U. A. Shah, “SIT: A Lightweight Encryption Algorithm for Secure Internet of Things”, Iqra University, Defence View and Department of Electronic Engineering, International Journal of Advanced Computer Science and Applications, Vol. 8, No. 1, 2017.

21. Vikash Kumar Jha,” Cryptanalysis of Lightweight Block Ciphers” Aalto University School of Science Degree Programme of Computer Science and Engineering, Master's Thesis, November 18, 2011.

22. B. Ray, S. Douglas, S. Jason, T. Stefan, W. Bryan, and W. Louis, “The simon and speck families of lightweight block ciphers,” Cryptology ePrint Archive, Report./404, Tech. Rep., 2013.

23. T. Mourouzis, G. Song, N. Courtois, and M. Christofii, “Advanced differential cryptanalysis of reduced-round simon64/128 using largeround statistical distinguishers,” 2015.

24. S. Khan, M. S. Ibrahim, K. A. Khan, and M. Ebrahim, “Security analysis of secure force algorithm for wireless sensor networks,” arXiv preprint arXiv:1509.00981, 2015.

25. D. J. Wheeler and R. M. Needham, “Tea, a tiny encryption algorithm,” in Fast Software Encryption. Springer, 1994, pp. 363–366.

26. E. Brickell, D. Denning, S. Kent, D. Maher, and W. Tuchman, “The skipjack algorithm,” Jul, vol. 28, pp. 1–7, 1993.

27. E. Souto, D. Sadok, J. Kelner et al., “Evaluation of security mechanisms in wireless sensor networks,” in null. IEEE, 2005, pp. 428–433.

28. A. E. Standard, “Federal information processing standards publication 197,” FIPS PUB, pp. 46–3, 2001.

29. D. J. Wheeler and R. M. Needham, “Correction to xtea,” Unpublished manuscript, Computer Laboratory, Cambridge University, England, 1998.

30. J. Lee, K. Kapitanova, and S. H. Son, “The price of security in wireless sensor networks,” Computer Networks, vol. 54, no. 17, pp. 2967–2978, 2010.

31. Bogdanov, A., Knudsen, L.R., Leander, G., Paar, C., Poschmann, A., Robshaw, M., Seurin, Y., Vikkelsoe, C., “PRESENT: an ultralightweight block cipher”. In: Paillier, P., Verbauwhede, I. (eds.) CHES 2007. LNCS, Springer, vol. 4727, pp450–466, 2007.

32. Wu, Wenling, and Lei Zhang. (2011, January). LBlock: a lightweight block cipher. In Applied Cryptography and Network Security (pp. 327-344). Springer Berlin Heidelberg.

33. Shibutani, K., Isobe, T., Hiwatari, H., Mitsuda, A., Akishita,T., Shirai, T.: Piccolo: an ultra-lightweight blockcipher. In:Cryptographic Hardware and Embedded Systems (CHES 2011),Springer, LNCS, 6917, pp. 342–357 (2011).

34. Suzaki, T., Minematsu, K., Morioka, S., & Kobayashi, E. (2011, November). Twine: A lightweight, versatile block cipher. In ECRYPT Workshop on Lightweight Cryptography (Vol. 2011).

35. Wei, Y., Xu, P., & Rong, Y. (2019). Related-key impossible differential cryptanalysis on lightweight cipher TWINE. Journal of Ambient Intelligence and Humanized Computing, 10(2), 509-517.

36. AL-Rummana, G. A., Shinde, G. N., & Al-Ahdal, A. H. MapReduced Based: A New Stream Cipher Technique for Data Encryption.

37. A. Webster and S. E. Tavares, “On the design of s-boxes,” in Conference on the Theory and Application of Cryptographic Techniques. Springer,1985, pp. 523–534.

38. Somaraj, S., & Hussain, M. A. (2015). Performance and security analysis for image encryption using key image. Indian Journal of Science and Technology, 8(35), 1.

39. Kanso, A., &Ghebleh, M. (2018). An efficient lossless secret sharing scheme for medical images. Journal of Visual Communication and Image Representation, 56, 245-255.

40. Kandar, S., Chaudhuri, D., Bhattacharjee, A., &Dhara, B. C. (2019). Image encryption using sequence generated by cyclic group. Journal of information security and applications, 44, 117-129.

41. Hodeish, M. E., Bukauskas, L., &Humbe, V. T. (2019). A new efficient TKHC-based image sharing scheme over unsecured channel. Journal of King Saud University-Computer and Information Sciences.

42. Ahmad, M., Doja, M. N., & Beg, M. S. (2018). Security analysis and enhancements of an image cryptosystem based on hyperchaotic system. Journal of King Saud University-Computer and Information Sciences.

43. D. Dinu, A. Biryukov, J. Großschädl, D. Khovratovich, Y. L. Corre, L. Perrin, “FELICS – Fair Evaluation of Lightweight Cryptographic Systems”, University of Luxembourg, July 2015.