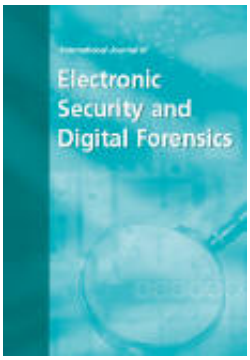



**International Journal of Electronic Security and Digital Forensics**

ISSN online: 1751-9128 - ISSN print: 1751-911X
https://www.inderscience.com/ijesdf

# A new encryption system for IoT devices using embedded key cryptosystem

Shadi R. Masadeh

**DOI:** 10.1504/IJESDF.2023.10046009

**Article History:**
Received: 13 January 2022
Accepted: 18 February 2022
Published online: 15 December 2022

# A new encryption system for IoT devices using embedded key cryptosystem

Shadi R. Masadeh

Faculty of Information Technology,
Department of Cyber Security,
Isra University,
Amman, Jordan
Email: shadi.almasadeh@iu.edu.jo

**Abstract:** IoT constrained devices have special phenomena of constrained resources such as power source, memory and processing power. Besides, it is vital to achieve an acceptable level of security and privacy while preserving the IoT device resources. In this paper, a new cryptographic algorithm is developed that would be suitable for securing IoT devices using embedded key cryptosystem. The encryption/decryption processes are achieved by segmenting the message into blocks of certain length. Each block is encrypted/decrypted using a key that is generated according to the segment itself, i.e., the keys are embedded into the blocks. The algorithm adopts two tables: one for the intended character set and the other for the key elements generation. The secrecy of these tables is responsible for securing the key strength. Experimental implementation proves the algorithm feasibility and strength against hackers and intruders.

**Keywords:** symmetric cryptosystems; key embedding; IoT security; network security.

**Reference** to this paper should be made as follows: Masadeh, S.R. (2023) 'A new encryption system for IoT devices using embedded key cryptosystem', *Int. J. Electronic Security and Digital Forensics*, Vol. 15, No. 1, pp.56–65.

**Biographical notes:** Shadi R. Masadeh is currently an Associate Professor in the Cyber Security Department at Isra University, Amman, Jordan. He obtained his MSc and PhD in Computer Information Systems/Information Security from the Arab Academy of Banking and Financial Sciences, in 2003 and 2009, respectively. His research interests include e-learning management, encryption and decryption systems networking, wireless security and robotics.

## 1 Introduction

The security of data in storage or in transit is becoming a real hazard for IoT applications. The customer's worries of database systems utilising the cloud for their bulk data storage are increasing due to the ever increasing processing speed and development of advanced software capabilities. This increased computer efficiency empowers hackers and intruders to breach security measures, hence, new and more powerful encryption techniques have to be regularly developed. For example, powerful cryptosystems, such as the data encryption standard (DES) (Singh et al., 2017) which was secure enough since its

approval by the National Institute of Standard and Technology (NIST) in the 1976 (Stallings, 2009). It encrypts the data as blocks of 64 bits' length using a key of 56 bits' length. DES was secure enough till around the end of the twentieth century as it was becoming vulnerable for attacks due to the advancement in the growing computer speed, then it was replaced by 3DES (Karn, n.d.), which can be operated with two or three keys. Although, 3DES is highly secure till now, unfortunately it requires long encryption and decryption times which might not be acceptable for most applications. Hence, the advanced encryption standard (AES), also known by its original name Rijndael cryptosystem, (after their developers' names; Vincent Rijmen and Joan Daemen) (*Federal Information Processing Standards Publication 197*, 2001). It is developed and adopted as a replacement for DES. It has three difference versions with key lengths of 128, 192 and 256 bits. AES cryptosystem proved that it is secure enough and became widely and effectively used at the moment for encryption/decryption of information all over the world. Both 3DES and AES cryptosystems have proved their strength and have been in use for some time for highly confidential and sensitive applications. However, for some less sophisticated applications, that needs secure but heavy data transit or bulky storage and retrieval of data such as databases utilising the storage capacity of computer cloud still require some classical encryption techniques that are fast enough with acceptable security level. This paper presents an encryption/decryption algorithm which could serve this purpose.

After the brief introduction in section 1, section 2 lists some important related work. Section 3 outlines the proposed cryptography system. Implementations, results and discussion will be given in Section 4, and finally Section 5 concludes the paper.

## 2 Related work

The internet of things (IoT) is a group of devices connected one to another over the Internet network to perform traditional operation in an automated way without human interaction, Solangi et al. (2018) defined the IoT as "wide range smart devices network that provides smart solutions using internet for automated communication among each other without requiring any human input".

The scope of IoT is vast, it presents a huge opportunity for many players in various businesses and industry domains. Many organisations are not necessarily aware of being using many IoT devices and many other organisations from different sectors have their own IoT to perform specific operations or tasks. As a result, it is important that organisations understand their use of IoT because many IoT devices affect cybersecurity and privacy risks differently than conventional information technology (IT) devices do (Boeckl et al., 2018).

Security and privacy concerns arise especially with new business models requirements which push the market to produce cheap plug and play devices (McDermott et al., 2018). This is expected to lead to rapid creation of insecure connected devices over computer networks, a thing that will certainly arise the threats of another paradigm shift to an internet of insecure things (Habibi et al., 2017). Integrating multiple functions within different IoT platforms will have its impact on the operation of critical appliances and systems in all fields, such as health, industry, military, defense, agriculture and many others. Therefore, this impact could exceed to cause harms and damages or many effects

on physical structure, cyberspace (stealing information, compromising systems, crashing web services, etc.), and many other impacts on the IoT platforms.

Social engineering threat is considered as one of cyber-attacks that incurs IoT platforms, as it could deceive the users to enable the attacker into performing their own intentions that will violate IoT System. Gan and Heartfield (2016) stated many social engineering crimes which started by human to human then expand their threat and damage to the whole system, for example, they quoted 80,000 people were impacted in one of the cases in Ukraine. An example of such damage occurred in December 2014, social engineering crime damaged a German steel mill furnace, as intruders gained accessed and acquired users' credentials by targeted phishing emails, hence compromised information affecting the complicated the mill production line, resulting into serious consequences. Lots of other examples can be spoken of here, such as the unforgettable blackout of 23rd of December 2015, where hackers used phishing emails causing a wide blackout in Ukraine. The users at that company were tricked by some phishing emails as issued by the prime minister. This attack is believed to be the first cyberattack which stopped the functions of the complete power grid in the country, resulting into no electricity in thousands of homes. Also, on 28th September 2019 the diversity in the things and devices connected over the IoT make securing IoT network more difficult.

Zhang et al. (2014) stated that this heterogeneity and complexity in the computer networks and internet services, make IoT security extremely complicated problem as compared to other security problems facing different computer network users in all walks of life.

The constrained resources of the IoT devices requires lightweight encryption algorithms to preserve the device's resources. Many researches proposed lightweight cryptosystem algorithms, for example Rajesh et al. (2019) proposed new (NTSA) algorithm that provides more dynamic confusion to the keys for each round to assure that a forgery cannot recognise the relationship between previously used keys. The proposed algorithm has 32 cycles and each cycle is composed of two rounds resulting in 64 rounds. Usman et al. (2017) proposed cryptographic algorithm named as (SIT). The implementation of SIT gave promising results suggesting that such algorithm can be a suitable candidate to be implemented for IoT applications. SIT is a hybrid approach based on Feistel concept and substitution-permutation networks, it is symmetric key block cipher that constitutes of 64-bit key and plain-text.

Fan et al. (2013) proposed a lightweight cipher WG-8 as a cryptographic algorithm, that is structured from the Welch-Gong cipher family. Their algorithm consists of 20 stages, each stage has 80-bit secret key together with 80-bit initial vector (IV), which can present a nonlinear filter generator over finite field. Bogdanov et al. (2007) proposed a lightweight block cipher called PRSENT. Their goal was to design an ultra-lightweight cipher dealing with 64-bit block size and an 80-bit key. Beaulieu et al. (2015) proposed SIMON and SPECK as two families of block cipher with 80-bit, 96-bit, and 128-bit key size. Nakajima and Moriai (2000) described Camellia block cipher that has key sizes of 128-bit, 192-bit, and 256-bits.

In the proposed algorithm, encryption/decryption processes rely on segmenting the input message into blocks of certain length. Each block is encrypted/decrypted using a key that is generated according to the segment itself. Hence, the keys are embedded into the blocks, so the suggested algorithm employs two tables, the first one lists and allocates numerals for the used character set, while the second determine the functions used to generate the encryption/decryption key.

## 3 The embedded cryptography algorithm

A new cryptosystem scheme is suggested in this paper. It is a block cipher symmetric system with a data block size of 32 bits. It starts with a preparatory phase were two tables are created. The character set of the language under consideration is used to build the first tables for character coding of the text message, while the second table provides the key generation strategy. Then, the encryption (or the decryption) algorithm is processed, which consists of the sequence of steps that fulfil the Shannon concept in order to achieve the required security. They include message segmenting into blocks, substitutions, secret key generation, data manoeuvring, logical operations, modular addition, reordering, etc. These processes will be shown in details below.

### *3.1 Text set and key preparation*

The chosen character set in this work deals with 55 characters (of which 26 alphabets, 10 numerals, and some 19 chosen special characters). These chosen character set elements are arranged into a text table (T1), and numbered from 0 to 54, as listed in Table 1. It is used for writing the plain messages. As the data block is of 32 bits length, the encryption key will be also of 32 bits length, and is generated from the elements of the data blocks by using the functions listed in the key generation table (T2). Table 1 (T1) characters are arranged alphabetically, and numbered in an ascending order, however, practically the choice for this sequence arrangement can be left to the communicating parties. Hence, each communication users can arrange their own character's sequence, which means a considerable increase in the security strength. Likewise, the key generation functions listed in T2 can also be left as communication parties' choice, adding more difficulties for intruders and hackers.

**Table 1** The chosen character set for the proposed scheme

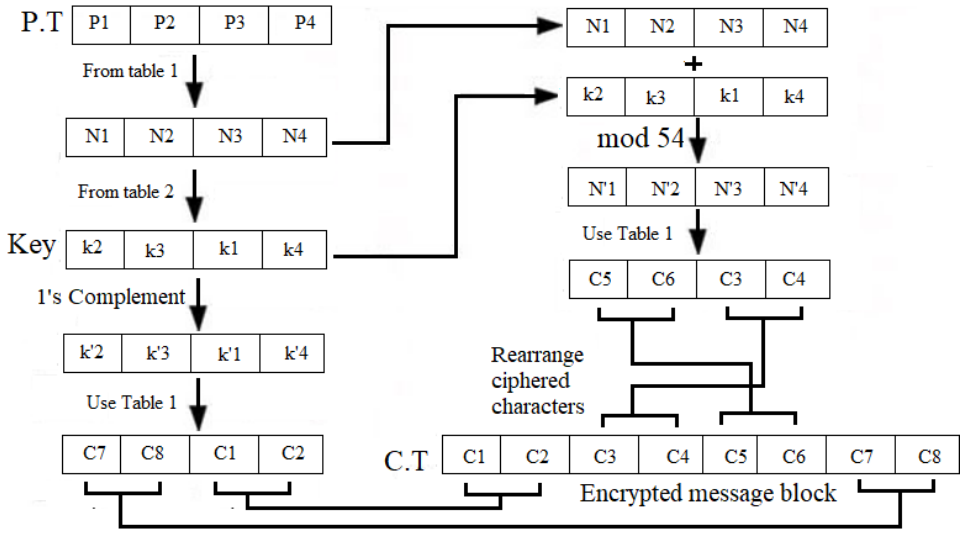
| A | B | C | D | E | F | G | H | I | J | K |
|---|---|---|---|---|---|---|---|---|---|---|
| 00 | 01 | 02 | 03 | 04 | 05 | 06 | 07 | 08 | 09 | 10 |
| L | M | N | O | P | Q | R | S | T | U | V |
| 11 | 12 | 13 | 14 | 15 | 16 | 17 | 18 | 19 | 20 | 21 |
| W | X | Y | Z | 0 | 1 | 2 | 3 | 4 | 5 | 6 |
| 22 | 23 | 24 | 25 | 26 | 27 | 28 | 29 | 30 | 31 | 32 |
| 7 | 8 | 9 | ? | ! | : | , | . | ( | ) | @ |
| 33 | 34 | 35 | 36 | 37 | 38 | 39 | 40 | 41 | 42 | 43 |
| # | % | / | * | + | - | = | Δ | & | _ | $ |
| 44 | 45 | 46 | 47 | 48 | 49 | 50 | 51 | 52 | 53 | 54 |

**Table 2** Functions used for generating the encryption key elements

| |
|---|
| $k_1 = (N1 + N2) \bmod 54$ |
| $k_2 = (N3 + N4) \bmod 54$ |
| $k_3 = (N1 + N4) \bmod 54$ |
| $k_4 = (N2 + N3) \bmod 54$ |

Notes: N1, N2, N3, and N4 are the corresponding numeral or the coded contents using Table 1.

**Figure 1** Encryption algorithm block diagram




## *3.2 Encryption*

To encrypt an input text message, it is first sliced in blocks of 32 bits length (i.e., each block consists of four characters). Then the blocks are treated in order by a sequence of processing operations using the encryption key, whose elements are generated from the block contents, hence, each block will be encrypted using its own generated key. Figure 1, give the block diagram of the message encryption process. Then the encryption process steps are described in the steps below:

Step 1 The text message block characters $P_1$, $P_2$, $P_3$, and $P_4$ are substituted by the by corresponding numeral listed in Table 1 (T1), resulting into the coded contents N: $N_1$, $N_2$, $N_3$, and $N_4$.

Step 2 Generate the encryption key components using Table 1 (T2) functions, resulting into the key for this block, i.e., K: $k_1$, $k_2$, $k_3$ and $k_4$, then convert them to binary representation.

Step 3 Get the 1's complement of the results of step 2. Then use only the six least significant bits, convert them back to decimal numbers and determine (mod 54) of the result.

Step 4 Get the characters corresponding to the obtained decimal numbers in step 3, i.e., $C_7$, $C_8$, $C_1$, and $C_2$, using Table 1 (T1).

Step 5 Add the results of steps 2 and 3 and apply mod 54, i.e., (N+K) mod 54, producing N'.

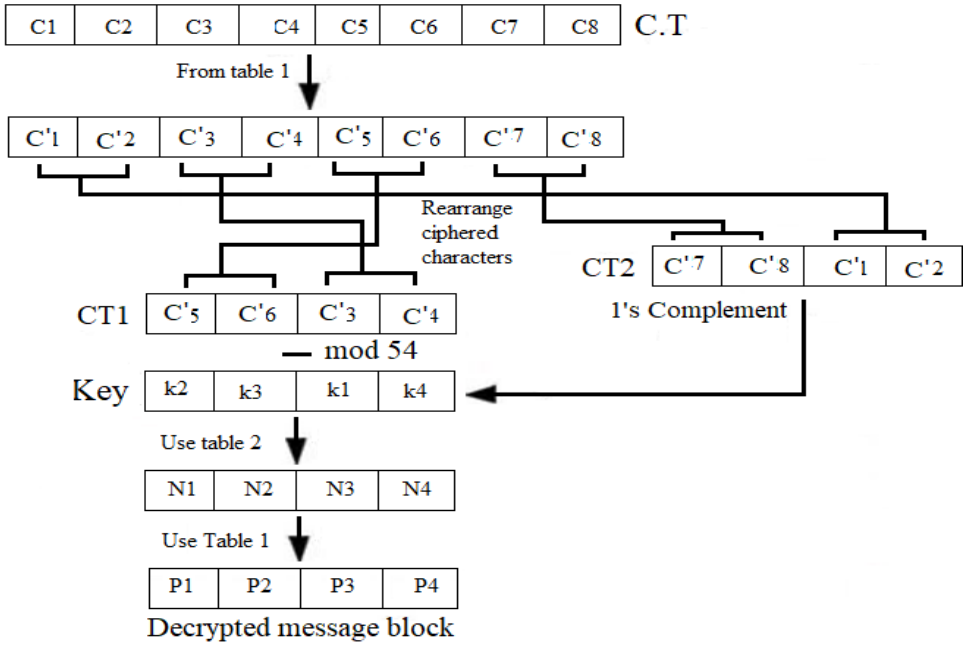
Step 6 Get the characters corresponding to the obtained decimal numbers in step 5, i.e., $C_5$, $C_6$, $C_3$, and $C_4$, using Table 1 (T1).

Step 7 Execute a transposition to the resulting characters of steps 4 and 6, as shown in the block diagram of Figure 1. The obtained character string is the ciphertext of the input plaintext.

The above steps from 1 to 7 are repeated for each four characters block of the input message, resulting into the cipher text of all the input message.

Due to the embedding of the encryption key into the encrypted message and because each block is encrypted with different key, it is obvious that the resulting encrypted message will have double the size of the original message. Although, this feature looks as drawback, it can be considered as a cause for strengthening the encryption security and safe data transfer and storage.

**Figure 2** Decryption algorithm block diagram




## *3.3 Decryption*

For the decryption of the ciphered message, the cipher text is first sliced into blocks of 8 characters length. Then the components of each block are replaced by their corresponding decimal value from Table 1 (T1) and converted to binary representations; C'1, …, C'8. The sequence of processes shown in the block diagram of Figure 2 are performed on the contents of each block. These processes can be summarised as listed in the following steps:

Step 1 Replace the character contents of each segment of the 8 characters by their equivalent numerical values using Table 1 (T); (C'1, …, C'8).

Step 2 Shuffle the numerical values string and split them in two equal parts, one contains the cipher text part (CT1; C'5, C'6, C'3, C'4) and the other contains the ciphered key part (CT2; C'7, C'8, C'1, C'2), see Figure 2.

Step 3 Perform 1's complement to CT2, then from Table 1 (T2) to get the block decryption key (K; $k_1$, …, $k_4$).

Step 4 Subtract the results of steps 3 from CT1 and apply mod 54, i.e., (CT1-K) mod 54, producing (N1, N2, N3, N4)

Step 5 Convert the outcome of step 4 (i.e., N1, N2, N3, N4 to their corresponding numerals using Table 1 (T1).

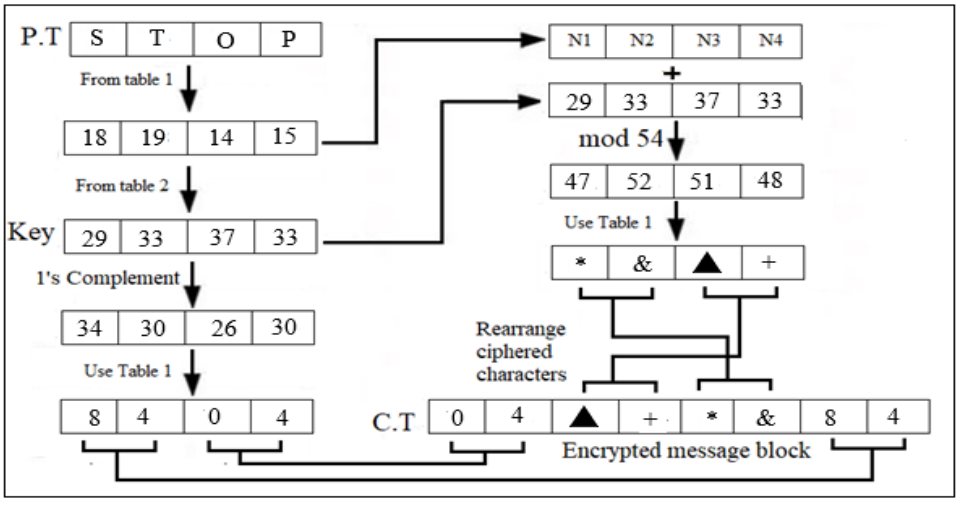
To recover the whole original message, steps 1 to 5 are applied sequentially on all the cipher text message blocks.

## 4 Implementation and discussion

### *4.1 Implementation example*

To clarify the validity of the proposed algorithm, a computer program is written to print out the output of each processing step for both encryption and decryption of a message block, as shown below.

**Figure 3** Encryption steps results for the message block 'STOP'



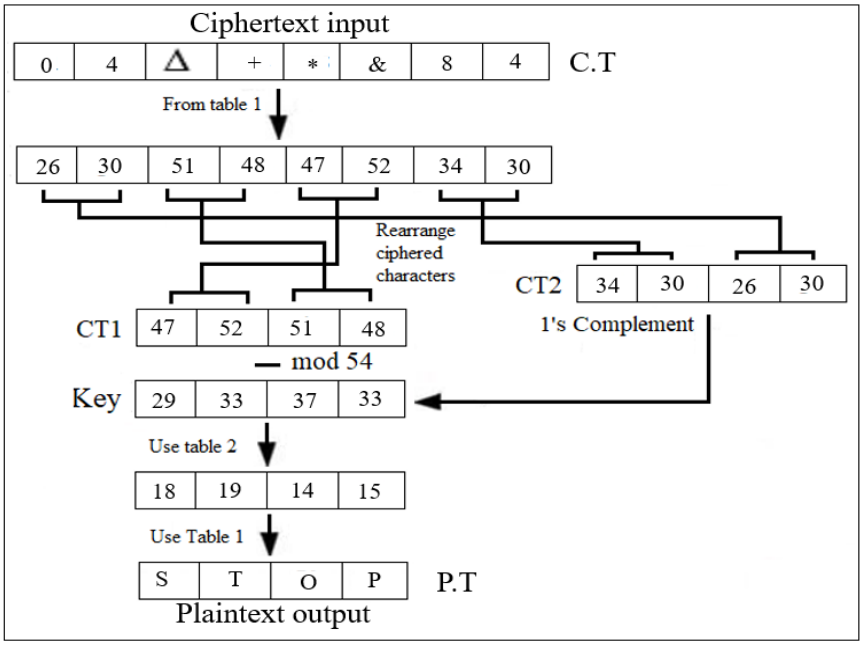

### *4.1.1 Encryption*

Consider the example of encrypting of the plaintext message block, M = 'STOP'. Process this message block following the algorithm steps given in section 3.2, the detailed calculation results are illustrated in the block diagram shown in Figure 3. The output ciphered block is C = '04Δ+*&84'.

### *4.1.2 Decryption steps*

The received ciphertext block C = '04Δ+*&84' is taken as input to the decryption algorithm. It is processed following the decryption steps given in Section 3.3, then the detailed calculation results illustrated in the block diagram illustrated in Figure 4. The final output will produce the original plaintext message block, i.e., M = 'STOP'.

**Figure 4** The decryption steps result for the ciphered block '04Δ+*&84'




### *4.2 The algorithm security*

The security strength of the proposed algorithm comes from two inherent design concepts, which are in short as follows.

- the input messages are segmented into blocks using 55 characters set that is arranged into a table which may differ from one user group to another, giving a key space of 55! (55 factorial)
- the encryption keys are derived from the message blocks, and as the message tables are not the same for different groups of users, so there will also be 55! different keys for different users.

Therefore, the key space is estimated to be $(55!)^2$ due to the fact that both Table 1 arrangement and the key K combined to determine the key space.

### 4.3 Features comparison

Message block length, number of iterations, encryption key length and space for any cryptosystem are of import features to look for at the decision time to choose a security algorithm for any application. Table 3 lists these features for the proposed algorithm as compared with some of the most public ally used schemes, including DES, AES, Blowfish, as well as many published lightweight cryptosystems, such as PRESENT, SIT, HIGHT, etc.

**Table 3** Algorithms features comparison

| *Algorithm* | *Block size (bit)* | *No. of rounds* | *Key size (bit)* | *Key space (key)* |
|---|---|---|---|---|
| DES | 64 | 16 | 56 | $2^{56}$ |
| AES-128 | 128 | 10 | 128 | $2^{128}$ |
| Blowfish | 64 | 16 | 448 | $2^{448}$ |
| PRESENT | 64 | 32 | 80 | $2^{80}$ |
| SIT | 64 | 5 | 64 | $2^{64}$ |
| HIGHT | 64 | 32 | 128 | $2^{128}$ |
| LED-64 | 64 | 32 | 64 | $2^{64}$ |
| TWINE-80 | 64 | 16 | 80 | $2^{80}$ |
| KLEIN-64 | 64 | 12 | 64 | $2^{64}$ |
| LBLOCK | 64 | 25 | 80 | $2^{80}$ |
| EKCA | 64 | 1 | 64 | $2 \times 2^{64}$ |
| Proposed | 32 | 1 | 32 | $(55!)^2$ |

Although, the data segments and encryption key size for the suggested algorithm here are shorter than other algorithm in the list, the good advantage is that a new key is created for each message segment, hence it produces instant key randomness, thus making key guessing extremely difficult for intruders or hackers.

It is worth stating that due to the embedding of the key for each message segment, the produced message ciphertext is twice the length of the original text message. It is obvious that the gained increase in the security strength is at the cost of increased ciphertext length.

## 5 Conclusions

The encryption/decryption approach presented in the algorithm her is a symmetric cryptosystem which relies on a continuously variable and embedded key depending on the message contents and based on secret encryption tables. It can be concluded that security strength obtained in the proposed algorithms is moderate when compared with the traditional DES and AES cryptographic algorithms. Such advantage would be greatly welcomed by many daily applications that involves huge amount of data in IoT applications utilising the available and cheap storage on the computer cloud. Therefore, it is anticipated by the authors that such system would be widely welcomed by such applications.

Messaging and data storage among IoT devices are 70% weak to cyber-attacks, as cited by Kumar et al. (2016). Therefore, secure lightweight encryption algorithm, such as the proposed Algorithm can be implemented and compared with other algorithm in order to achieve acceptable security.

Furthermore, implementing the proposed algorithm principle in applications involving the encryption of other multimedia such as audio, photos and movies would be a good area of research in the future.

## References

Beaulieu, R., Treatman-Clark, S., Shors, D., Weeks, B., Smith, J. and Wingers, L. (2015) 'The SIMON and SPECK lightweight block ciphers', *Design Automation Conference (DAC), 2015 52nd ACM/EDAC/IEEE*, IEEE.

Boeckl, K. et al. (2018) *Considerations for Managing Internet of Things (IoT) Cybersecurity and Privacy Risks*, National Institute of Standards and Technology, NISTIR 8228 (Draft).

Bogdanov, A., et al. (2007) 'PRESENT: an ultra-lightweight block cipher', *International Workshop on Cryptographic Hardware and Embedded Systems*, Springer, pp.450–466.

Fan, X., Mandal, K. and Gong, G. (2013) 'Wg-8: a lightweight stream cipher for resource-constrained smart devices', *International Conference on Heterogeneous Networking for Quality, Reliability, Security and Robustness*, Springer, pp.617–632.

*Federal Information Processing Standards Publication 197*, Announcing the Advanced Encryption Standard (AES) (2001).

Gan, D. and Heartfield, R. (2016)'Social engineering in the internet of everything', *Cutter IT Journal* (Cutter Information Corp), Vol. 29, No. 7, pp.20–29.

Habibi, J., Midi, D., Mudgerikar, A. and Bertino, E. (2017) 'Heimdall: mitigating the internet of insecure things', *IEEE Internet of Things Journal*, IEEE, Vo. 4, No. 4, pp/968–978.

Karn, P., Metzger, P. and Simpson, W. (1995) *The ESP triple DES Transform*, RFC1851.

McDermott, C.D., Petrovski, A.V. and Majdani, F. (2018) 'Towards situational awareness of botnet activity in the internet of things', *Institute of Electrical and Electronics Engineers*.

Nakajima, J. and Moriai, S. (2000) *A Description of the Camellia Encryption Algorithm*, August.

Rajesh, S., Paul, V., Menon, V.G. and Khosravi, M.R. (2019) 'A secure and efficient lightweight symmetric encryption scheme for transfer of text files between embedded IoT devices', *Symmetry*, MDPI AG, Vol. 11, No. 2, p.2.

Singh, S., Sharma, P.K., Moon, S.Y. and Park, J.H. (2017) 'Advanced lightweight encryption algorithms for IoT devices: survey, challenges and solutions', *Journal of Ambient Intelligence and Humanized Computing*, Vol. 2, pp.1–18, Springer Verlag.

Solangi, Z.A., Solangi, Y.A., Chandio, S., Syarqawy bin Hamzah, M. and Shah, A. (2018) 'The future of data privacy and security concerns in Internet of Things', *2018 IEEE International Conference on Innovative Research and Development (ICIRD)*, IEEE, pp.1–4.

Stallings, W. (2009) *Wireless Communications & Networks*. Pearson Education, India.

Usman, M., Ahmed, I., Aslam, M.I., Khan, S. and Shah, U.A. (2017) 'SIT: a lightweight encryption algorithm for secure internet of things', *arXiv preprint arXiv:1704.08688*.

Zhang, Z-K., Cho, M.C.Y., Wang, C-W., Hsu, C-W., Chen, C-K. and Shieh, S. (2014) 'IoT security: ongoing challenges and research opportunities', *2014 IEEE 7th International Conference on Service-Oriented Computing and Applications*, IEEE, pp.230–234.