Journal of Modern Technology and Engineering
Vol.7, No.3, 2022, pp.253-266

DESIGN AND IMPLEMENTATION OF TINY ADVANCED ENCRYPTION STANDARD FOR IoT APPLICATION

Alaa S. Abdalrazzaq, Salah Abdulghani Alabady*

College of Engineering, Computer Engineering Department, University of Mosul, Iraq

Abstract. One of the most important communication technologies of our day is the Internet of Things (IoT). The IoT system faced issues with device power consumption, battery life, memory space, and cost of performance. Cyberattacks and security dangers are on the rise. Cybersecurity aims to lower the threats it poses to computer systems by safeguarding sensitive data and ensuring privacy during transmission. Encryption algorithms are one of the most crucial ways to guarantee the confidentiality of data while it is being transmitted. One of the most used encryption algorithms is called Advanced Encryption Standard (AES). In this paper, the Tiny Advanced Encryption Standard (TAES) algorithm has been proposed with two scenarios to improve the AES algorithm's encryption speed and flexibility when used for small data. It is also proposed to achieve a high level of encryption and reduce the complexity of the algorithm while maintaining a level of security that is higher than or equal to the level of security found in the original AES algorithm. Using the MATLAB program, the performance of the suggested algorithm was compared to that of the original AES algorithm on texts, and images. From the results, it can be concluded that when the first scenario of the proposed algorithm was implemented on data with a size of 2048 bytes, the speed of encryption implementation increased by 82.69 %. And in the second scenario, it went up by 57.54%. Additionally, throughput rates grew at a rate similar. For images, the encryption speed ratio for the four scenarios was 75% and 61.41% when the scenarios of the proposed algorithm were applied to an image of size ($280\Gamma - 280$ pixels).

Keywords: IoT, Cybersecurity, Encryption Algorithms, AES Algorithm, Signal to Noise Ratio(SNR), Speedup, Throughput.
AMS Subject Classification: 68M15.
***Corresponding author:** Salah A., Alabady, College of Engineering, Computer Engineering Department, University of Mosul, Iraq, e-mail: *eng.salah@uomosul.edu.iq*
Received: 2 October 2022; Revised: 28 November 2022; Accepted: 3 December 2022; Published: 29 December 2022.

1 Introduction

The Internet of Things (IoT) is gaining ground in a variety of industries and organizations, with predictions that there will be 75 billion devices will be connected to the Internet by 2025 (Syed et al., 2021). According to another research, more than 125 billion will be linked devices by 2030 (Espinoza et al., 2020). Data is generated, delivered, monitored, and processed by IoT network devices, and it is sent and stored on private or public clouds (Syed et al., 2021). Sports, education, commerce, infrastructure, transportation, and health are some of the domains where IoT is employed. Other applications include factories, buildings, cities, electrical networks, homes, and energy (Espinoza et al., 2020; Andrade et al., 2020). Cybersecurity in the IoT can promote global accessibility, integrity, availability, scalability, confidentiality, and interoperability while also protecting participants' privacy (Espinoza et al., 2020; Lu and Da Xu, 2018). The processing unit, energy, and storage limitations of heterogeneous devices will challenge the imple-

mentation of traditional security solutions on constrained IoT devices (Abdullah et al., 2019b). The purpose of IoT cybersecurity is to secure IoT assets and privacy while reducing cybersecurity risk for organizations and users (Lee, 2020). The encryption algorithm is a process of converting unencrypted plain text information and data into ciphertext data. The reason for this is to hide data and information to prevent unauthorized users from accessing this information. The main objective of the encryption process is the confidentiality of the transmitted information. The encryption process is carried out on the sender's side. And the decryption process is on the recipient's side. The encryption process needs two conditions to complete its occurrence, which is the encryption key and the encryption algorithm (Shinde et al., 2014; Mallouli et al., 2019). AES is a block cipher that employs symmetric key cryptography. It is a quicker algorithm than others and almost resistant to current practical attacks. The number of rounds to be done depends on the key length; for 128-bit keys, this is 10 rounds, for 192-bit keys, 12 rounds, and for 256-bit keys, 14 rounds depending (Arman et al., 2020; Gamido et al., 2018). To increase the efficiency of encryption for small data, a Tiny Advanced Encryption Standard (TAES) is proposed in this paper. The encryption process starts with a data size of 4 bytes, and the algorithm only runs through six rounds to ensure that it can be implemented faster than the original AES algorithm while maintaining an A safety rating that is higher than or equal to that of the original algorithm. The content of this paper is as follows. Section 2 reviews recent research on IoT, cybersecurity, AES, and encryption algorithms. Section 3 gives suggestions and methodologies. The outcomes of the suggested algorithm are presented in Section 3 to wrap up this paper.

2 Related Works

IoT cybersecurity methods and cyber risks are reviewed in the article Lee (2020) the four layers that make up the IoT cyber risk management framework are the IoT ecosystem layer, IoT electronic infrastructure layer, IoT cyber risk assessment layer, IoT cyber performance layer, the IoT cyber risk assessment, assessment, and prioritization layer particularly identifies, assesses, and prioritizes IoT cyber hazards. Article Abdullah et al. (2019a) discussed one instance of cybersecurity in the IoT space, focusing on the security requirements and challenges that each IoT layer encounters. In addition to describing certain security precautions to avoid these challenges and risks. Gathers and studies practical cryptographic algorithms analyze them and determine the best performing algorithms in real-time to assure the security of information transported over the network (Shinde et al., 2014). The effectiveness of several symmetric and asymmetric algorithms was evaluated in Panda (2016), according to the simulation results, AES performs better than other algorithms in terms of throughput and decryption speed.

The research Kubadia et al. (2019) provided a comparison of symmetric encryption algorithms based on throughput, encryption time, and decryption time. The AES algorithm is the most secure because it performed the best in terms of throughput and larger block sizes when compared to the other algorithms. The AES method was used as an encryption and decryption algorithm to offer a secure data transmission channel. To encrypt both texts and images, a key length of 128 bits is employed. The results that were displayed in a manner that prevented the unauthorized user from spotting them were found using MATLAB (Aparna et al., 2019). The researcher in Lu and Tseng (2002) proposes a new method with a straightforward core structure that combines AES and decrypted algorithms. The overall hardware complexity is reduced to 68% when compared to alternative solutions that carry out encryption and decryption using multiple modules and ROMs. 400-bit key and 200-bit text are used in the development of the AES-algorithm. The technique generates unique keys for each round using a 400-bit key divided into 200-200 pieces, increasing security. The researchers in Gupta et al. (2017) improved security and execution speed over the original algorithm by reducing the number of rounds to 5 and omitting the Mix Column step twice. In Dang and Vo (2019) suggested using dynamic key

generation to put the enhanced AES algorithm into practice. 16 bytes of data will be layered onto the frame and transmitted one by one. This order will have a significant impact. With this significant modification, the security of data transfer in IoT systems will be enhanced.

Boomerang attacks on rounds 5 and 6 of AES have been shown in Biryukov (2004) to be much faster than global search. It also shown that AES has a wide enough safety margin against Boomerang attacks. In the paper Biryukov (2006) a simple idea of extracting leakage from block cipher is proposed. The idea is to extract parts of the internal state in certain rounds and give it an output key stream. This mainly depends on the strength of the encoder's circular function and on the strength of the encoder's main schedule. Setting LEX (Leak Extraction) and resynchronizing which is just setting one AES key and one AES encryption is much faster than most other stream ciphers. The design uses AES in a natural way, in each AES round we output a certain four bytes from the intermediate state. AES can be used with the three different key lengths (128, 192, 256). The difference with AES is that the attacker never sees the full 128-bit ciphertext but only parts of the intermediate state.

3 System Model and the Proposed Method

This paper proposes a new encryption algorithm called Tiny AES (TAES) that builds on AES. The fundamental idea behind TAES is to make the proposed algorithm flexible and efficient so that it can be used to implement encryption for data ranging in size from 4 bytes to an unspecified number of bytes while maintaining a level of security greater than or equal to the level found in the original algorithm. The master key input is 16-byte (key[0],key[1]... key[15]), with the first 4-byte (key[0-3]) as the initial key input while the other bytes are used in the g function as in Figure 1. Additionally, reducing the number of rounds to six rounds will speed up the encrypting and decrypting processes, improve throughput, and speed up the process. Two scenarios are put out in this paper, and the following sections will go into further detail on each.

3.1 First Scenario

The Mix Column step from the stander AES is eliminated in the first scenario for all rounds. Six rounds make up the cipher process, and each round comprises one of three different transform functions:

- **Sub Byte**: It is a permutation stage in which the entries of a predetermined $16\Gamma - 16$ matrix called substitution box (S-box) are substituted for the contents of each cell of the state array. For decryption, a permutation of 256 values is stored in one table (Inverse S box) of $16\Gamma - 16$ bytes.
- **Shift Row**: State's first row remains unchanged. A 1-byte circular left shift is performed for the second row.
- **Add Round Key**: the State's 32 bits are bitwise XORed with the round key's 32 bits.

The following steps make up the reverse procedure:

- **Inverse Shift Row**: the inverse shift row transformation, or InvShiftRows, executes the circular shifts in the reverse direction with a 1-byte circular right shift for the second row.
- **Inverse Substitute Bytes**: The method, also known as InvSubByte, is the same for the forward sub byte except that a new byte is allocated to each byte of the state matrix from an inverse s-box.
- **Add Round Key**: The inverse Add Round Key transformation is equivalent to the forward Add Round Key transformation because the XOR operation is its inverse. The schematic diagram of the first scenario is shown in Figure 2.

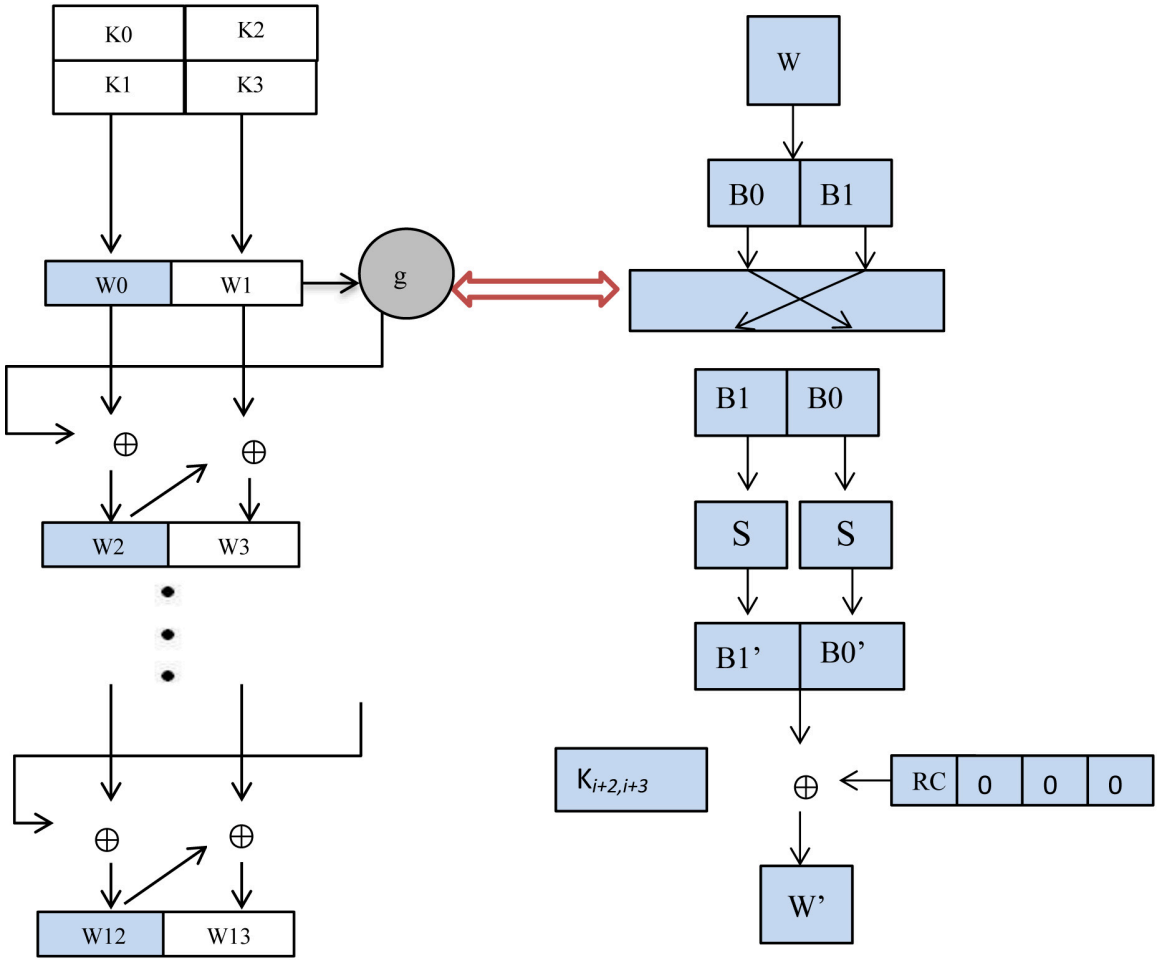


Figure 1: TAES key expansion

3.2 Second Scenario

The difference from the previous scenario is that the encryption and decryption processes have been made more complex to ensure higher encryption and to make it more difficult for an unauthorized person to steal information. In terms of the steps involved in each round, this scenario is very similar to the AES algorithm. The original mix column step is not removed from the rounds in the encryption and decryption process. This scenario's first five rounds are completely made up of:

- **Substitute Bytes**
- **Shift Row**
- **Mix Column:**Also referred to as mix col, this process takes place between the matrix resulting from the shift row and the size of this matrix $(2\Gamma - 2)$ and between the mix col matrix consisting of $(4\Gamma - 4)$. The mix col operation is performed by transforming the resulting shift row into a $1\Gamma - 4$ matrix and multiplying (row $\Gamma-$ column). The result is a matrix $(4\Gamma - 1)$ that is reformatted into a matrix $(2\Gamma - 2)$ as usual as shown in Figure 3.

- **Add Round Key** The decryption process of the first five rounds consists of:

  Inverse Shift Row
  Inverse Substitute Bytes
  Add Round Key

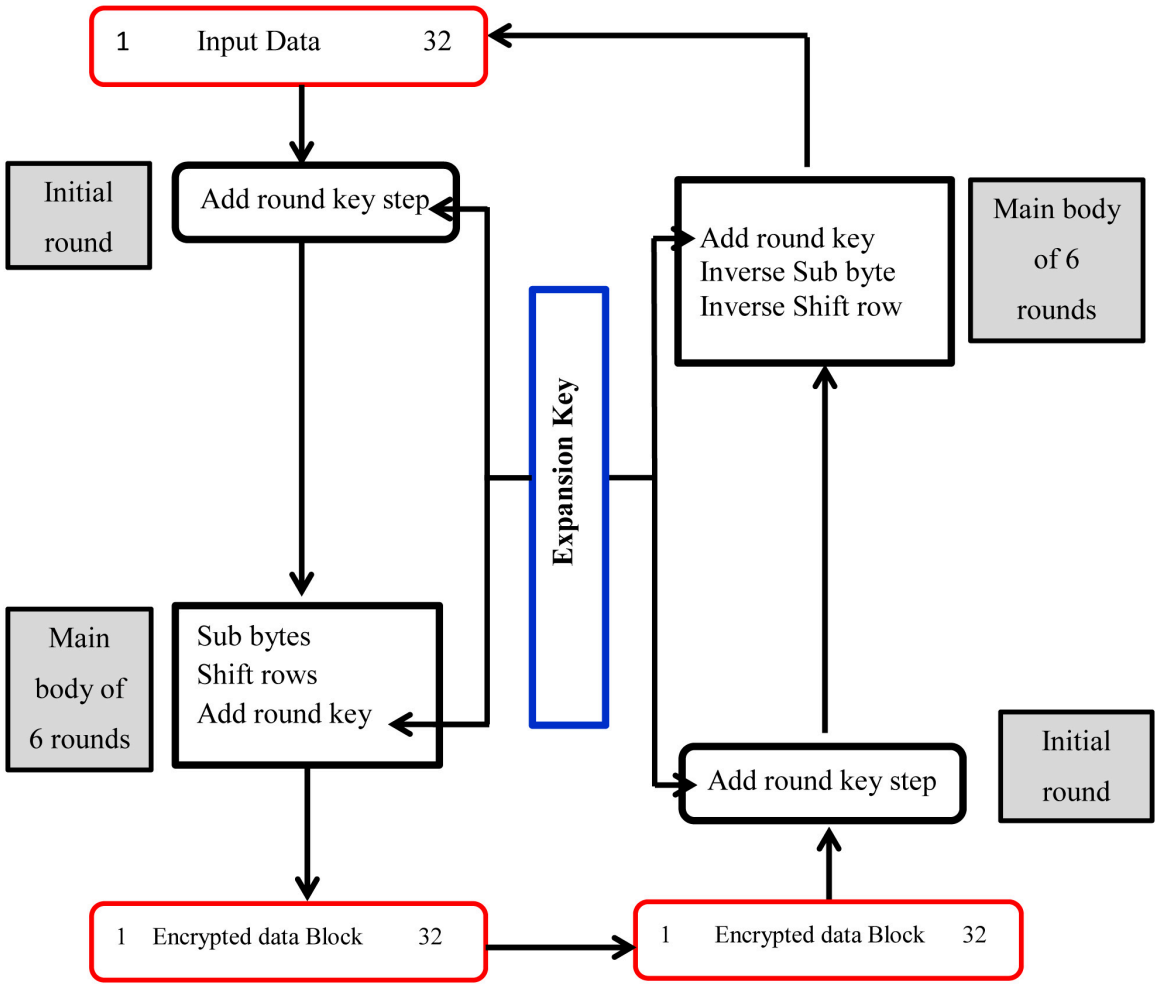


Figure 2: The First Scenario of TAES

- **Inverse Mix Column**: The inverse mix column step is executed on the proposed algorithm using the same equations and principles as the original algorithm, with the exception that the state matrix is $2\Gamma - 2$ instead of $4\Gamma - 4$. Figure 4 explains this step.

The sixth and final round consists of

- Inverse Shift Row
- Inverse Substitute Bytes
- Add Round Key

Figure 5 shows the encryption and decryption procedure for the second scenario of the proposed algorithm.

4 Simulation Results Analysis

The performance of the proposed algorithm was evaluated by two types of inputs texts and images. The MATLAB simulation program is used for the implementation and evaluation of the TAES. The performance of the TAES is evaluated according to the following criteria:

- **Encryption Time:** The length of time needed to transform plaintext into ciphertext. The encryption time is measured in seconds and depends on the size of the message

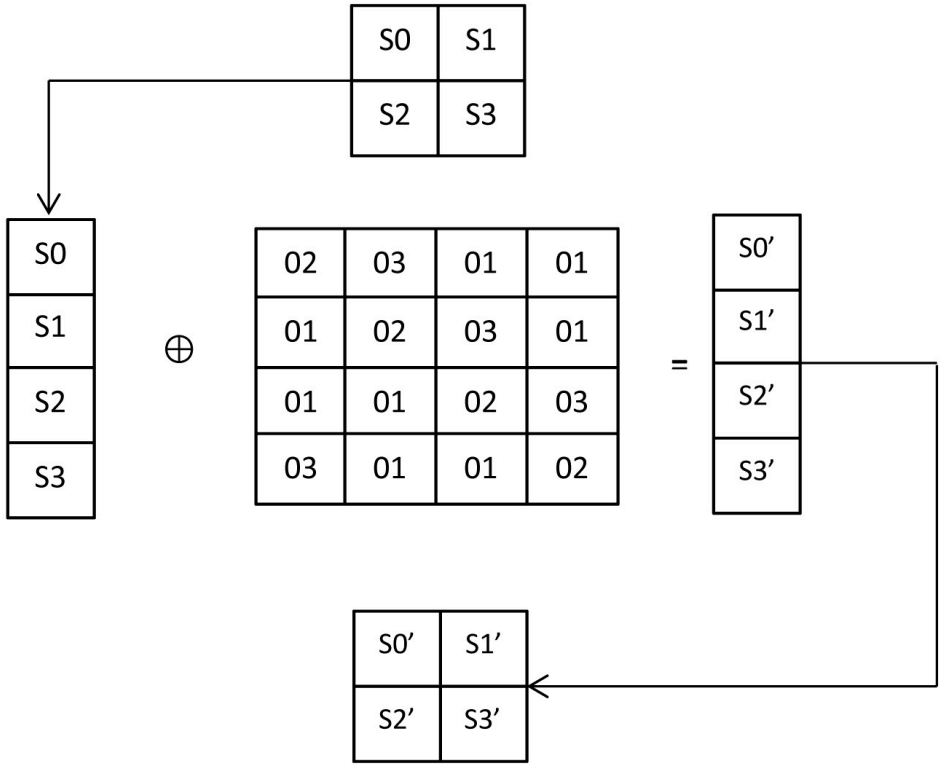


Figure 3: TAES Mix Column Step

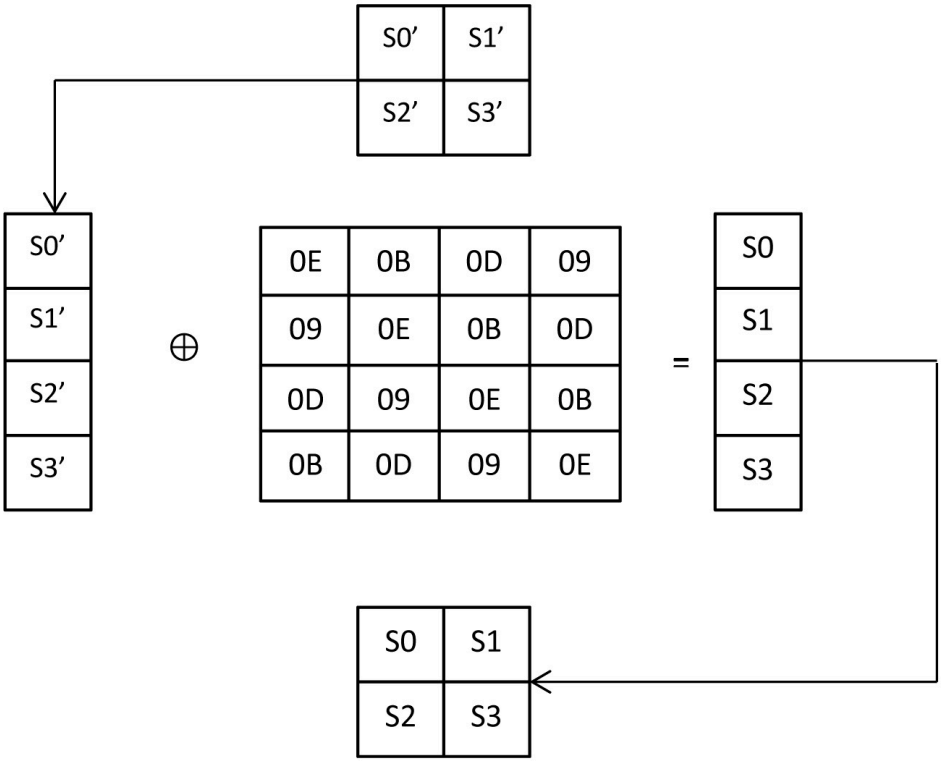


Figure 4: TAES TAES Inverse Mix Column Step

block and the key. It directly affects how well the encryption algorithm performs. Every cryptographic algorithm had a minimum encryption time required to create a quick and responsive encryption system.

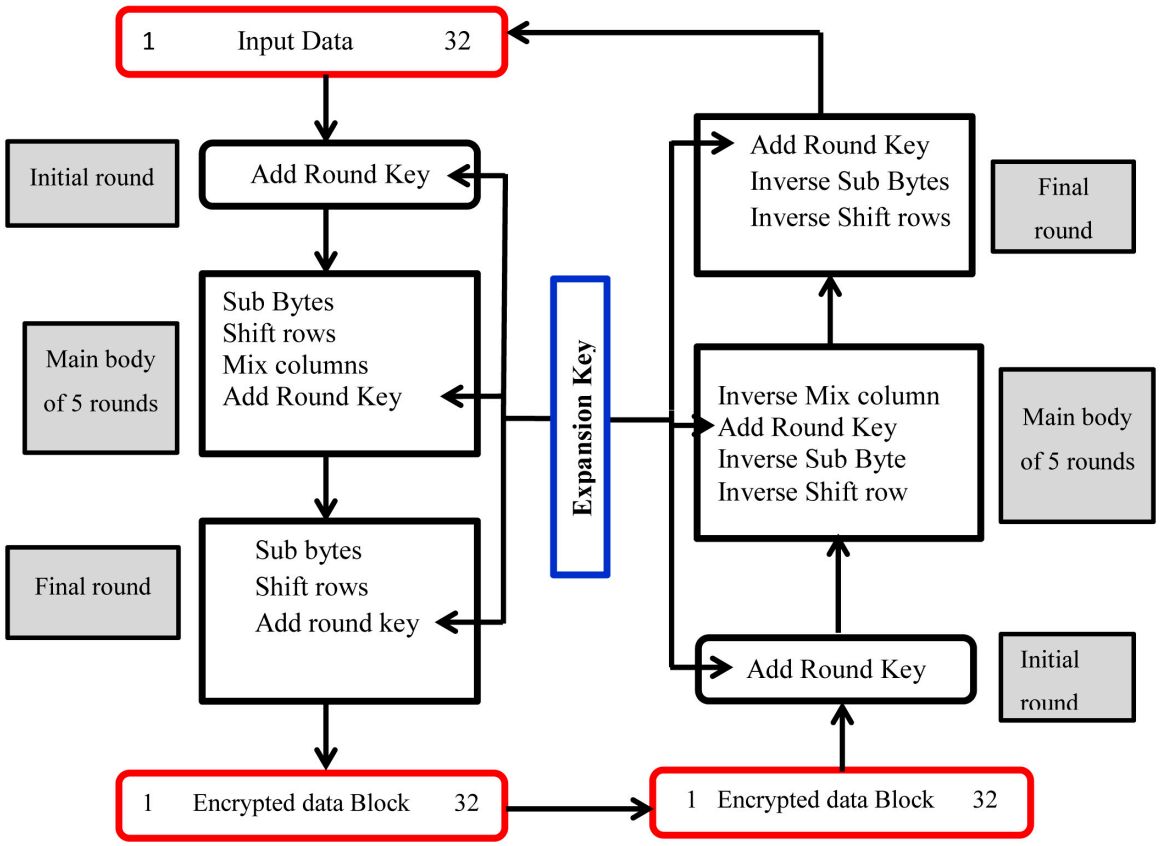


Figure 5: The Second Scenario of TAES

- **Decryption Time:** The amount of time needed to decipher ciphertext and find the plaintext. The decryption time should be less similar to the encryption time and measured in seconds to make the cryptographic algorithm quick and responsive.

- **Throughput:** It is the proportion of messages that are successfully delivered across a communication channel in telecommunications networks. The throughput rate, which is the speed of the communication channel, is typically expressed in kilobytes (KB) per second. By dividing the total size of the encrypted block (Tp) by the total encrypting time (Et), one can determine the transmission rate of an encryption technique. The power consumption of the algorithm will decrease as throughput grows Equation (1) represents the throughput.

$$Throughput = \frac{Tp}{Et} \tag{1}$$

- **The Signal-to-Noise Ratio (SNR)**: is a measurement that contrasts the strength of the desired signal with the strength of noise in science and engineering. SNR is referred to as the signal-to-noise ratio and is frequently stated in decibels (dB) as a metric for describing an image's quality. Equation (2) represents the SNR.

$$SNR = 10log\frac{P_{signal}}{P_{noise}} \tag{2}$$

- **Speedup**: Can be defined as the ratio between the execution time of the encryption and decryption process without the use of optimization, and the full implementation of the encryption and decryption using optimization. Equation (3) represents the speedup.

Table 1: Comparison between the TAES scenarios and AES algorithm in terms of encryption

Data Size (Byte)	TAES 1st scenario encryption time (sec)	TAES 2nd scenario encryption time (sec)	AES Encryption Time (sec)
4	0.000469	0.002273	0
8	0.002703	0.003340	0
16	0.001952	0.003215	0.019834
64	0.002549	0.003662	0.011651
128	0.001705	0.003588	0.011868
256	0.001675	0.003378	0.012943
1024	0.003487	0.003715	0.013145
2048	0.001921	0.004713	0.011101

$$Speedup = \frac{E_w}{E_e} \tag{3}$$

Where, Ew = Execution time without optimized algorithm. Ee = Execution time using the optimized algorithm

The text test involves entering texts with the following byte sizes: 4, 8, 16, 64, 128, 256, 1024, and 2048. There are letters, numbers, symbols, and spaces in these texts. The encryption and decryption times, throughput, and speedup are then calculated for each scenario and compared to the original AES algorithm.

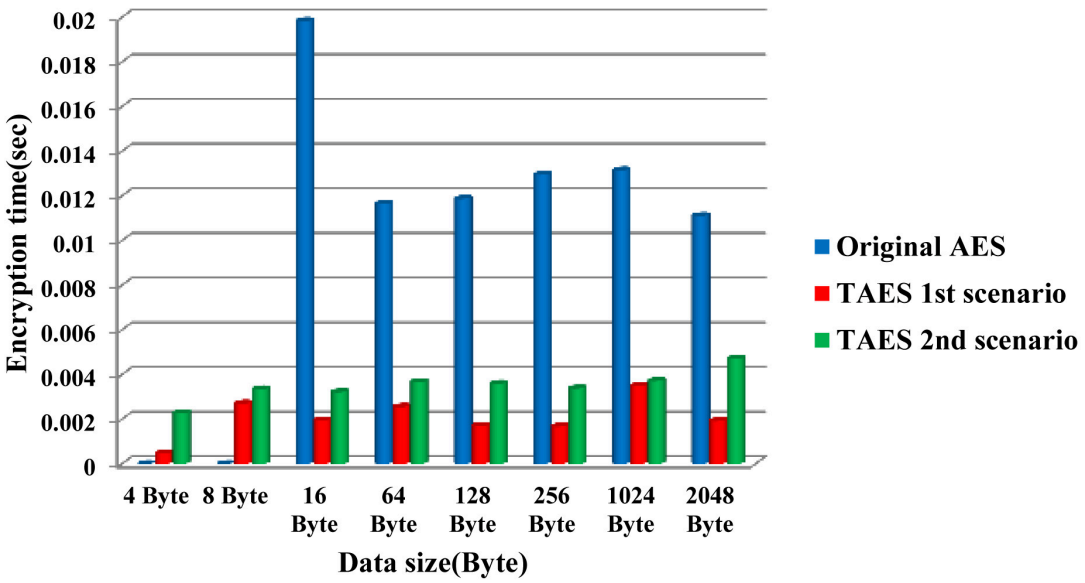


Figure 6: Comparison of the TAES scenarios with the original AES's encryption times

Because the complexity of the proposed algorithm is reduced during implementation, Table 1 and Figure 6 show that the proposed algorithm encrypts data faster than the original AES algorithm performed for the same volume of data. The proposed algorithm is also more flexible than the original AES algorithm because it begins encrypting data at a size of 4 bytes.

Because the complexity of the proposed algorithm is reduced during implementation, Table 2 and Figure 7 show that the proposed algorithm decrypted data faster than the original AES algorithm performed for the same volume of data.

Table 2: Comparison between the TAES scenarios and AES algorithm in terms of decryption time

Data Size (Byte)	TAES 1st scenario decryption time (sec)	TAES 2nd scenario decryption time (sec)	AES Decryption Time (sec)
4	0.000431	0.003392	0
8	0.000800	0.002711	0
16	0.000786	0.002479	0.021726
64	0.001113	0.002626	0.014639
128	0.000687	0.003000	0.013324
256	0.000685	0.002309	0.014431
1024	0.000671	0.002662	0.013305
2048	0.000758	0.002737	0.014318

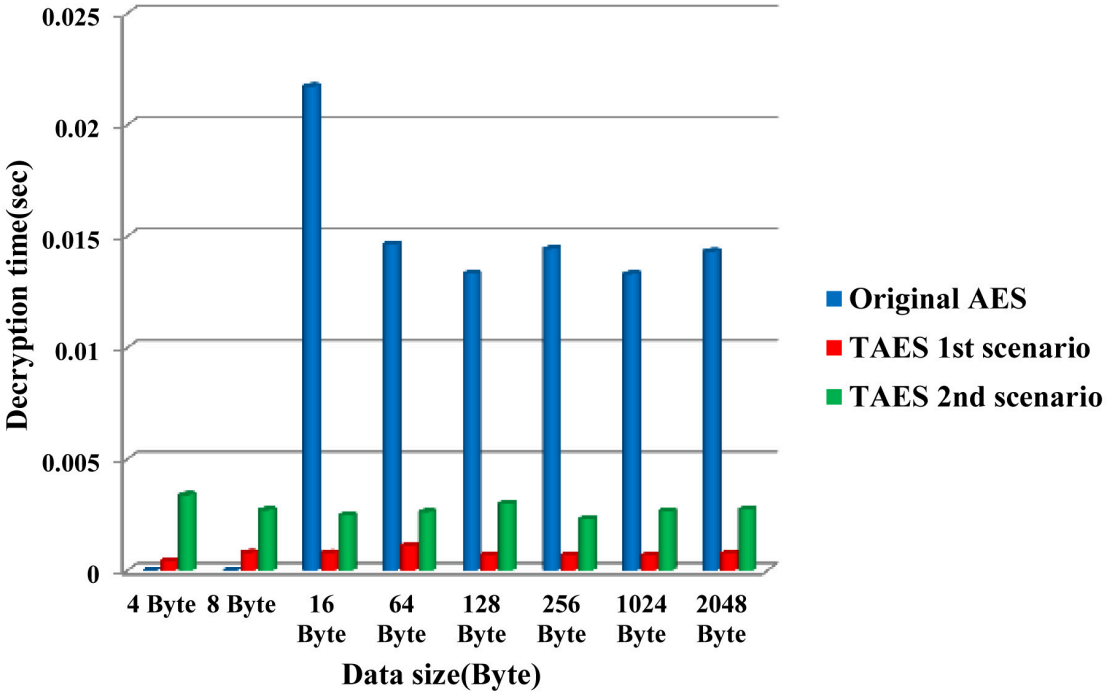


Figure 7: Comparison of the TAES scenarios with the original AES's decryption times

Table 3: Comparison between the TAES scenarios and AES algorithm in terms of throughput

Data Size (Byte)	Throughput of TAES 1st scenario (KB/sec)	Throughput of TAES 2nd scenario (byte/sec)	Throughput of AES(KB/sec)
4	8.329	1.72	0
8	2.89	2.34	0
16	8.004	4.86	0.788
64	24.52	17.07	5.36
128	73.31	34.84	10.532
256	149.25	74.008	19.315
1024	286.78	269.18	76.07
2048	1037.06	618.01	179.46

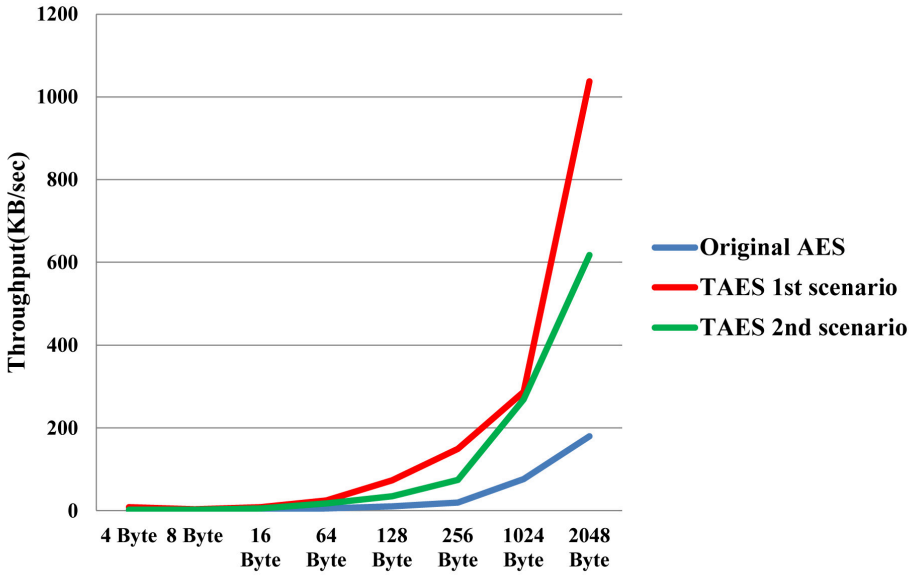


Figure 8: Comparison of the TAES scenarios with the original AES's throughput

Table 4: Comparison between the TAES scenarios and AES algorithm in terms of speedup of encryption and decryption process

Data Size (Byte)	Speedup for the encryption process using 1st scenario of TAES	Speedup for the encryption process using 2nd scenario of TAES	Speedup for the decryption process using 1st scenario of TAES	Speedup for the decryption process using 2nd scenario of TAES
16	10.161	6.17	27.64	8.76
64	4.57	3.18	13.15	5.58
128	6.96	3.31	19.39	4.44
256	7.73	3.83	21.06	6.25
1024	3.77	3.54	19.83	4.99
2048	5.78	2.36	18.89	5.23

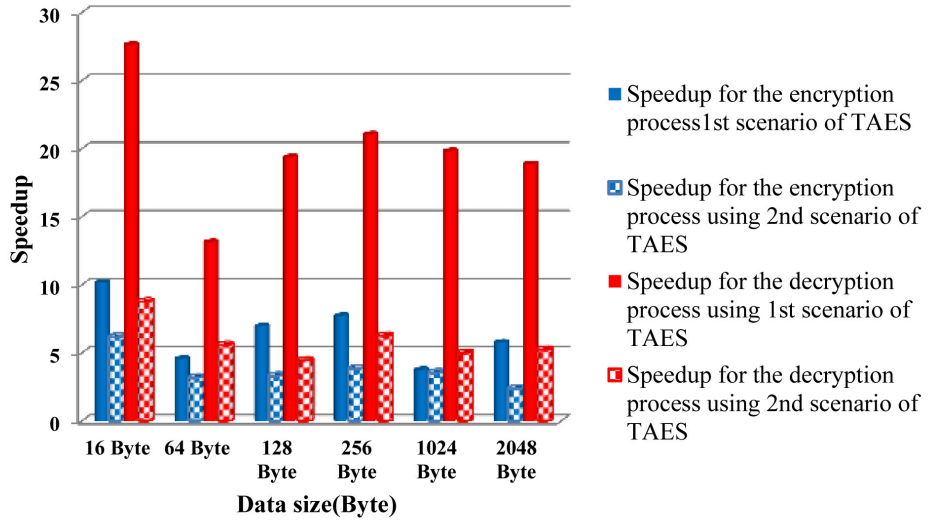


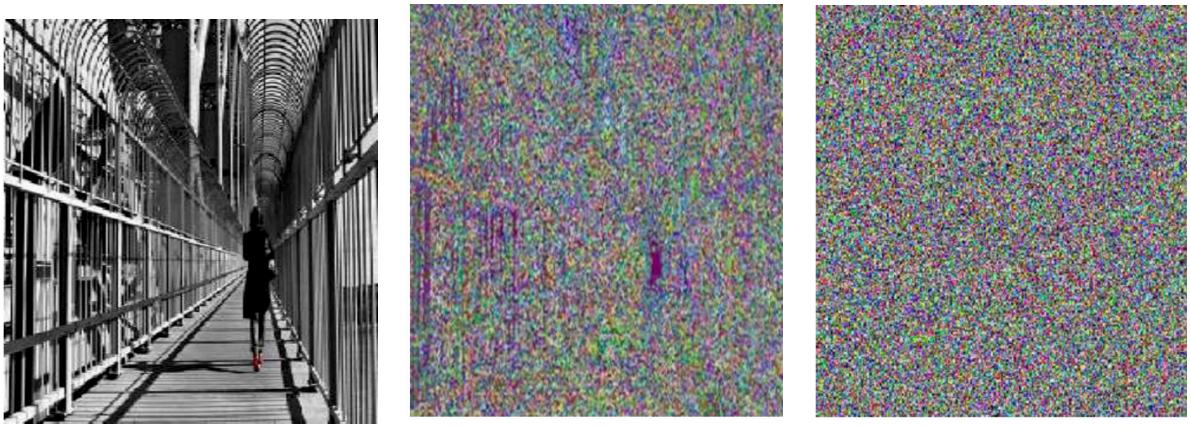
Figure 9: Comparison of the TAES scenarios with the original AES's speedup

Table 3 and Figure 8 show that, when compared to the original AES algorithm, the first and second scenarios of the proposed algorithm are more throughputs since the encryption using the

proposed algorithm is carried out more quickly. The high speedup values for the encrypting and decryption processes are shown in Table 4 and Figure 9 respectively.

Because the encryption and decryption processes of the proposed algorithm are faster than those of the original AES algorithm, the execution of the proposed method's encryption and decryption scenarios is carried out more speedup.

Images are used as a second test to determine how accurate the proposed algorithm is. SNR, throughput, encryption, and decryption times are the measures used. The image has a size of $280\Gamma - 280$ (54,479 bytes) pixels and is of a (bridge in black and white). Figure 10 shows the result of using the proposed algorithmic scenarios to encrypting image.



(a) (b) (c)

Figure 10: (a, b, c) Image encryption using TAES scenarios. (a) Original Image. (b) By using first scenario of TASE. (c) By using second scenario of TASE.

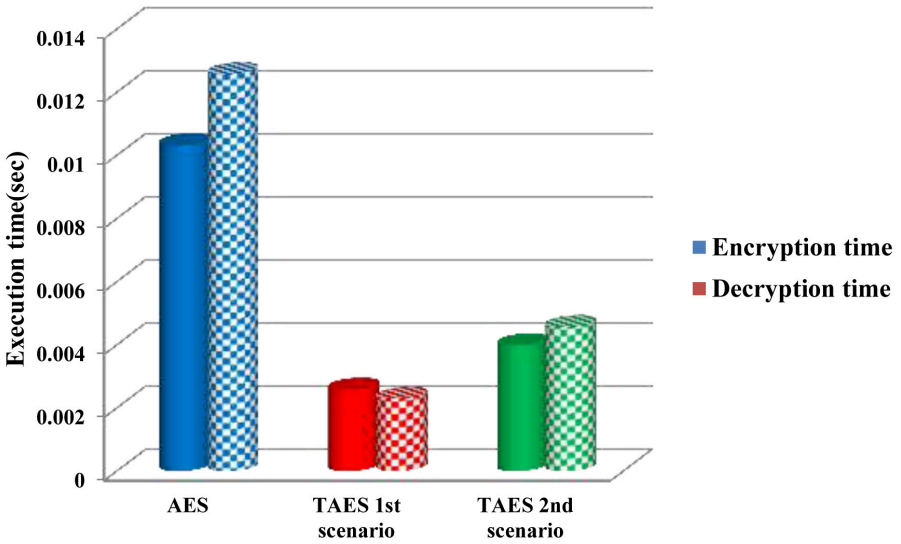


Figure 11: The efficiency of the TAES algorithm scenarios in comparison to the efficiency of the original AES algorithm in terms of encryption and decryption time for the bridge in black and white images

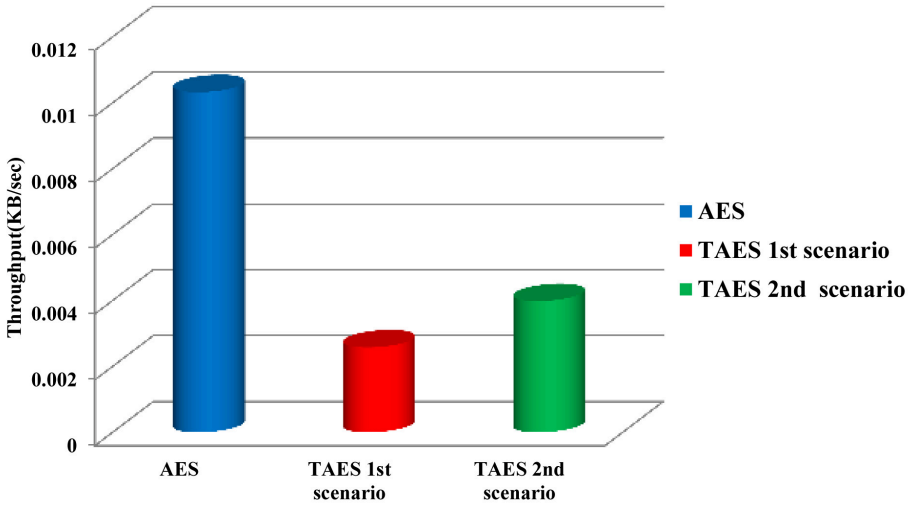


Figure 12: TThe efficiency of the TAES algorithm scenarios in comparison to the efficiency of the original AES algorithm in terms of throughput for the bridge in black and white images

Table 5: SNR, encryption time, decryption time, and throughput for the bridge in black and white image encrypted using the TAES scenarios and original AES algorithm

Items	SNR(dB)	Encryption time(sec)	Decryption time(sec)	Throughput(KB/sec)
1st scenario	1.63	0.00258	0.002295	23181.76
2nd scenario	1.4334	0.003984	0.004564	13353.95
AES	1.4423	0.010324	0.012571	4301.93

The proposed algorithm outperforms the previous algorithm in terms of encryption and decryption times as well as throughput, as shown in Table 5 and Figures 11 and 12. The number of rounds, which was reduced to only 6 rounds instead of 10, and the number of stages in each cycle, makes the proposed scenarios less complex than the original AES algorithm.

5 Conclusion

In this paper, the improvements to the AES encryption algorithm have been proposed. These improvements would make the AES algorithm more flexible and quick to implement, with a security level that is higher than or equivalent to the AES algorithm's security. It was suggested that the data size be decreased to start at 4 bytes rather than 16 bytes. This gave the proposed algorithm flexibility in that it could be used for little amounts of data and could also be used for large amounts of data by separately encrypting each block of four bytes. Another suggestion was to employ six rounds as opposed to the AES 128 algorithm's 10 rounds. MATLAB software was used to examine the results shown. The results also showed that when the first scenario of the proposed algorithm was executed on 2048-byte data, the encryption execution speed increased by 82.69%. In the second scenario, it increased by 57.54%. In addition, throughput rates grew at a similar rate. For images, the encryption execution speed ratio for the two scenarios was 75%, and 61.41% when applying the scenarios of the proposed algorithm to an image of size (280 x 280 pixels).

References

Abdullah, A., Hamad, R., Abdulrahman, M., Moala, H., and Elkhediri, S. (2019a). Cybersecurity: a review of internet of things (iot) security issues, challenges and techniques. pages

1–6.

Abdullah, T. A., Ali, W., Malebary, S., and Ahmed, A. A. (2019b). A review of cyber security challenges attacks and solutions for internet of things based smart home. *Int. J. Comput. Sci. Netw. Secur*, 19(9):139.

Andrade, R. O., Yoo, S. G., Tello-Oquendo, L., and Ortiz-Garcés, I. (2020). A comprehensive study of the iot cybersecurity in smart cities. *IEEE Access*, 8:228922–228941.

Aparna, V., Rajan, A., Jairaj, I., Nandita, B., Madhusoodanan, P., and Remya, A. A. (2019). Implementation of aes algorithm on text and image using matlab. pages 1279–1283.

Arman, S., Rehnuma, T., and Rahman, M. (2020). Design and implementation of a modified aes cryptography with fast key generation technique. In *2020 IEEE International Women in Engineering (WIE) Conference on Electrical and Computer Engineering (WIECON-ECE)*, pages 191–195. IEEE.

Biryukov, A. (2004). The boomerang attack on 5 and 6-round reduced aes. In *International Conference on Advanced Encryption Standard*, pages 11–15. Springer.

Biryukov, A. (2006). The design of a stream cipher lex. pages 67–75.

Dang, T. N. and Vo, H. M. (2019). *Advanced AES algorithm using dynamic key in the internet of things system*.

Espinoza, M. G. T., Melendrez, J. R. N., and Clemente, L. A. N. (2020). A survey and an iot cybersecurity recommendation for public and private hospitals in ecuador. *Advances in Science, Technology and Engineering Systems*, 5(3):518–528.

Gamido, H. V., Sison, A. M., and Medina, R. P. (2018). Modified aes for text and image encryption. volume 11, pages 942–948. Institute of Advanced Engineering and Science.

Gupta, A., Jaiswal, M., et al. (2017). An enhanced aes algorithm using cascading method on 400 bits key size used in enhancing the safety of next generation internet of things (iot). pages 422–427.

Kubadia, A., Idnani, D., and Jain, Y. (2019). Performance evaluation of aes, arc2, blowfish, cast and des3 for standalone systems: Symmetric keying algorithms. pages 118–123.

Lee, I. (2020). Internet of things (iot) cybersecurity: Literature review and iot cyber risk management. *Future Internet*, 12(9):157.

Lu, C.-C. and Tseng, S.-Y. (2002). Integrated design of aes (advanced encryption standard) encrypter and decrypter. In *Proceedings IEEE International Conference on Application-Specific Systems, Architectures, and Processors*, pages 277–285. IEEE.

Lu, Y. and Da Xu, L. (2018). Internet of things (iot) cybersecurity research: A review of current research topics. *IEEE Internet of Things Journal*, 6(2):2103–2115.

Mallouli, F., Hellal, A., Saeed, N. S., and Alzahrani, F. A. (2019). A survey on cryptography: comparative study between rsa vs ecc algorithms, and rsa vs el-gamal algorithms. pages 173–176.

Panda, M. (2016). Performance analysis of encryption algorithms for security. pages 278–284.

Shinde, H. N., Raut, A. S., Vidhale, S. R., Sawant, R. V., and Kotkar, V. A. (2014). A review of various encryption techniques. *International Journal of Engineering And Computer Science*, 3(9):8092–8096.

Syed, A. S., Sierra-Sosa, D., Kumar, A., and Elmaghraby, A. (2021). Iot in smart cities: a survey of technologies, practices and challenges. *Smart Cities*, 4(2):429–475.