INDIAN JOURNAL OF SCIENCE AND TECHNOLOGY

ORIGINAL ARTICLE

OPEN ACCESS

Received: 23/06/2025

Accepted: 06/09/2025

Published: 29/09/2025

Citation: Gharat N, Jolly DL (2025) A Lightweight, Energy-Efficient Modified AES Algorithm for Real-Time IoT Applications. Indian Journal of Science and Technology 18(35): 2897-2907. https://doi.org/10.17485/IJST/v18i35.1156

* **Corresponding author**.

neha.gharat@vcet.edu.in

Funding: None

Competing Interests: None

Published By Indian Society for Education and Environment (iSee)

ISSN
Print: 0974-6846
Electronic: 0974-5645

A Lightweight, Energy-Efficient Modified AES Algorithm for Real-Time IoT Applications

Neha Gharat[1,2]*, Dr. Lochan Jolly[3]

1 Research Scholar, Department Electronica and Telecommunication, Thakur College of Engineering and Technology, Kandivali (E)
2 Assistant Professor at Department Electronica and Telecommunication, Vidyavardhini's College of Engineering and Technology, Vasai(W)
3 Professor, Department Electronica and Telecommunication, Thakur College of Engineering and Technology, Kandivali (E)

Abstract

Background/Objective: The modified AES algorithm introduced in this study is proposed mainly for use with resource-constrained devices and lightweight cryptography. This research examines how performance metrics, including encryption and decryption time, energy consumption, and memory usage, impact the efficiency of energy-constrained devices. **Methods:** The key modification in this algorithm is the introduction of a novel smart XOR technique in place of the Mixcolumn of standard AES. This approach allows a round-based, non-uniform XOR that reduces the processing load and improves data diffusion. **Findings:** The proposed M-AES algorithm markedly surpasses current lightweight encryption techniques in key performance criteria. M-AES attains a 33.4% superior throughput compared to AES-128 (71.11 Mbps versus 53.33 Mbps) and a 25% decrease in encryption and decryption duration. There is a 178% increase in throughput (71.11 Mbps vs. 25.6 Mbps) and a 28% decrease in processing time compared to SPECK. M-AES has a throughput that is 289% higher than SIMON (71.11 Mbps vs. 18.29 Mbps) and cuts the time it takes to encrypt and decode by 48.6%. M-AES meets high security standards and is good for data that is very sensitive. It is both fast and strong in terms of encryption. Furthermore, formal cryptanalysis indicates that M-AES exhibits robust resistance to both differential and linear attacks, providing security levels akin to those of standard AES. These improvements demonstrate that M-AES is a strong candidate for use in resource-constrained environments where lightweight, secure, and efficient cryptographic systems are required. **Novelty:** This work introduces novel M-AES, a lightweight encryption algorithm that optimizes the conventional AES-128 framework for enhanced efficiency in resource-limited IoT settings. The Smart XOR approach offers a new perspective on lightweight encryption design for real-time low-power devices. M-AES enhances key scheduling and optimizes the processing pipeline, merging the strong security of AES with markedly less computing overhead.

Keywords: Lightweight Cryptography; Modified AES; Energy-Efficient Security; Real-Time Encryption; Performance Metrics; cryptanalysis

1 Introduction

The Internet of Things (IoT) devices and embedded platforms' explosive evolution demand encryption algorithms that can function proficiently under rigorous resource constraints. The Advanced Encryption Standard (AES) is widely used for robust security, though it needs high computational power. But it becomes impractical for devices with limited resources. Recently, researchers have been working on designing a lighter version of AES so it can work better on resource-constrained devices. Using real-world test beds, current lightweight AES methods haven't shown that they work well enough on limited devices. A lot of studies either don't fully test performance or use a lot of extra energy since they are more complicated. AES needs a change that is efficient, doesn't use a lot of resources, and keeps security while speeding up encryption and using less memory and energy. This research mitigates these shortcomings by presenting and evaluating a Modified AES (M-AES) algorithm that substitutes the conventional MixColumns step with a Smart XOR-based transformation.

For an energy-efficient, secure Internet of Things, Sarker[(1)] (2025) examines lightweight cryptography and IDS algorithms, emphasizing solutions designed for low power and real-time use in limited devices. Lightweight encryption techniques designed for Internet of Things-based healthcare are reviewed by Sabri et al.[(2)] (2025), who assess their advantages and disadvantages in terms of security, energy consumption, and real-time functionality. Chakrabarty et al.[(3)] (2024) proposed a multi-tiered encryption system that integrates layered security methodologies and lightweight cryptography to enhance data protection in resource-constrained environments. Their approach demonstrated greater resilience and efficiency, rendering it suitable for secure communication in smart systems and the Internet of Things. Soto-Cruz et al.[(4)] (2024) evaluated multiple lightweight algorithms concerning power consumption, throughput, and memory economy in microcontroller-based devices. Nonetheless, their benchmarking occurred in controlled settings and may not accurately represent real-world IoT scenarios characterized by delay, environmental noise, or multi-device interactions. Al-Heayli and Aldabbagh[(5)] (2024) augmented AES with a chaos-optimized S-box with a modified Jellyfish Search method to boost nonlinearity. Zala and Khan[(6)] (2024) proposed a dual-encryption technique using both AES and Fernet combined with zlib compression to secure multimedia files. This enhances privacy and speed of processing, but it might also introduce delays and complicate key management in distributed systems. Astillero[(7)] (2024) came up with a version of AES that uses dynamic key expansion dependent on how sensitive the real-time data is. This makes it harder for brute-force and differential attacks to work. But this method makes encryption take longer and makes it harder to put into action. A different study by Pinuyut et al.[(8)] (2024) reduced the number of rounds and replaced the MixColumns step with a bit permutation technique to speed things up, but this made the encryption less strong. Zhong and Gu[(9)] (2024) give a full study of lightweight block ciphers (LWBCs) that are made for situations with limited resources, such smartcards and the Internet of Things. They put designs into six structural classes (SPN, Feistel, ARX, etc.) and use hardware and software measures like gate area, power use, delays, and security trade-offs to rate them. Li et al.[(10)] (2024) look at how vulnerable QARMA is to statistical fault attacks in IoE settings. They show that caused faults can leak key-dependent biases, which makes it possible to recover keys in lightweight circumstances. Ahirrao et al.[(11)] (2023) suggested changes to AES to secure and improve the performance characteristics for SMS encryption, smart locks, and IoT healthcare. They discussed methods to reduce computational overhead, including rapid key scheduling, dynamic S-boxes, and the removal of MixColumns. However, these modifications were effective in increasing speed. But these changes result in a more complex system, and security strength is also reduced. The construction of lightweight block ciphers for resource-constrained contexts, such as the Internet of Things, was reviewed by Zakaria et al.[(12)] in 2023. The research delineates critical topics, encompassing implementation costs, security, and energy efficiency. Selvapriya and Suganthi[(13)] (2023) employed a dynamic pipelined asynchronous methodology to develop a low-power AES cryptocore, hence enhancing efficiency. Their utilization significantly reduces power consumption, rendering it ideal for the Internet of Things and embedded applications.

Ascon v1.2 is an efficient suite for authenticated encryption and hashing, specifically developed for resource-constrained devices, as indicated by Dobraunig et al.[(14)] (2021). It utilizes a concise 320-bit permutation to attain an equilibrium between security and efficiencyAbdul Raheem et al.[(15)] (2021) propose an enhanced lightweight SPECK encryption system for cloud-based smart healthcare, improving privacy, confidentiality, and security of data for IoT sensor data. Utilizing ECDH-based certificate authority management, Ahmed[(16)] (2021) introduces ESSC, a lightweight symmetric cryptographic method that ensures secure and efficient digital certificate exchange and data encryption in resource-constrained industrial IoT applications. Rashidi[(17)] (2021) presents two hardware architectures for the SPONGENT lightweight hash function: one serialized and one utilizing full data-path width, optimizing resource utilization in resource-constrained systems. Panahi et al.[(18)] (2021) evaluate ten lightweight block ciphers, including AES, PRESENT, SIMON, and RECTANGLE, utilizing the Raspberry Pi 3 and Arduino Mega 2560. They assess metrics such as energy consumption, memory footprint, execution time, and throughput for both encryption and decryption. Thakor et al.[(19)] (2021) conduct a comprehensive assessment of over 50 lightweight cryptographic algorithms for resource-constrained IoT devices, evaluating them based on variables such as implementation cost, hardware/software performance, and resilience to attacks. For the Piccolo lightweight block cipher, Ramu et al.[(20)] (2020)

suggest hardware-optimized architectures using loop-unrolled, parallel, and pipelined designs. These are implemented on an FPGA and provide high throughputs with low space utilization, making them perfect for limited IoT systems. By modifying the TWINE cipher with a straightforward tweak schedule from SKINNY, Sakamoto et al.[21] (2020) provide T-TWINE, the first lightweight, tweakable block cipher based on a Generalized Feistel Structure. With its support for 64-bit blocks, 80- or 128-bit keys, and a 64-bit tweak, it provides effective tweakable encryption with little overhead compared to TWINE. LAES, a lightweight version of AES, is proposed by Acla & Gerardo[22] (2019) to significantly reduce hardware complexity by substituting a 128-bit permutation for the conventional MixColumn step. By using deep learning to improve assaults on round-reduced Speck32/64, Gohr[23] (2019) achieved key recovery that was noticeably better than that of conventional techniques. Key recovery took about 15 minutes thanks to the neural technique, which decreased the effective security of 11-round Speck to about 38 bits. Dhanda et.al, in[24] (2020), analysed lightweight encryption techniques and categorized them based on their computational efficacy and compatibility with different IoT devices. However, as the study did not evaluate the effectiveness of real-world attacks, their practical security has not been thoroughly verified. Table 1 shows the limitations of the existing cryptographic approach.

Table 1. Comparative Analysis of existing Algorithm

Study	AES Modification	Main Feature	Limitation
[1]	Examining IDS methods and lightweight cryptography for the Internet of Things	Focuses on low-power, energy-efficient, and real-time IDS and cryptography solutions for limited IoT devices.	Lacks a thorough analysis of algorithm-specific difficulties and practical performance
[2]	Review of low-power encryption methods for Internet of Things-based medical systems weighs the Advantages and disadvantages of energy usage, security, and real-time capacity	Recommends further study on hybrid and privacy-preserving techniques.	Experimental validation and integration techniques are not done for healthcare hybrid encryption.
[5]	S-box optimized for chaos using Jellyfish Search	Improved nonlinearity and attack resistance	Scalability has not been evaluated, and there is no realistic assessment of the attack.
[6]	Zlib compression combined with dual encryption (AES + Fernet)	Improved processing efficiency and confidentiality	Complex key handling and increased latency
[7]	Adaptive key expansion according to the sensitivity of the data	Enhanced defense against differential and brute-force attacks.	Increased implementation complexity and encryption latency
[8]	Bit permutation replaces MixColumns, reducing the number of rounds.	Higher throughput	Lower cryptographic strength
[11]	MixColumns were eliminated, and quick key scheduling and dynamic S-box were utilized.	Enhanced SMS, smart lock, and IoT healthcare speed	Increased complexity and reduced overall security
[13]	MixColumns are replaced with bit permutation, resulting in fewer rounds.	More throughput	less robust cryptography
[22]	MixColumns are replaced by a 128-bit permutation (LAES).	Reduced hardware cost	Cryptographic balance is impacted by structural changes.
This study	MixColumns are replaced by a smart X-OR operation	Low energy consumption, enhanced, improved throughput, reduced complexity, real time cryptography solutions for resource-constrained devices, improved attack resistance	-

1.1 Limitations of the Existing Approaches

There have been many suggestions for different AES variations and encryption methods. Many studies either reveal too much energy overhead or don't fully test performance because the systems are getting more complicated. The main problems with current methods are that they don't test them enough against linear and differential attacks, they don't prove that they work on real-world IoT hardware, they have higher latency because of hybrid or dual-layer encryption frameworks, they use more

energy even though they simplify structures, they add more algorithmic complexity because of dynamic S-boxes or compression layers, and they don't handle memory well, which makes throughput unpredictable when workloads change. To keep security, speed up encryption, and use less memory and energy, we need an AES modification that works well and doesn't add too much overhead.

The primary challenge is to develop a lightweight encryption technique that ensures robust security (assessed through criteria such as the avalanche effect), reduces energy and memory consumption, and provides real-time performance for limited IoT platforms. This is defined as a problem statement, and the proposed system has provided the solution by designing the M-AES method.

1.2 Motivation

This study is motivated by the gap between cryptographic strength and operational efficiency in limited contexts. A lot of lightweight encryption algorithms try to make things less complicated, but they usually give up some performance or security. Most importantly, they don't check how strong they are against attacks using quantitative standards like the avalanche effect. The goal here is to create a secure algorithm that keeps the robustness of AES while using less energy and processing power by simplifying the architecture.

1.3 Proposed Solution

To address these research challenges, we introduced a Modified AES (M-AES) that integrates a Smart XOR mechanism. This proposed method increases the nonlinearity of the S-box and improves data diffusion through enhanced XOR logic, without adding substantial computational load. Our method focuses on calculating the influence of this change on key performance metrics such as execution speed, memory footprint, and energy consumption. The ultimate aim is to evaluate whether this improved M-AES is practical for deployment in environments with limited resources. Novel Smart XOR logic replaces MixColumns, resulting in a 35-40 % reduction in encryption time and over 40-45% drop in energy consumption compared to standard AES. We also use avalanche effect analysis to check how resistant an attack is, which makes sure that security is not compromised.

2 Methodology

The AES encryption process involves crucial steps like AddRoundKey, MixColumns, and ShiftRows. MixColumns enhances the security by spreading the impact of a single byte alteration throughout the entire output. This step is made faster and more effective by using the Smart XOR approach. This approach includes randomness by utilizing constants for every round, ensuring security while reducing the complex calculations of the original MixColumns. The developed AES method is tested on a Raspberry Pi 3 Model B, which features 1 GB of RAM and a 1.2 GHz quad-core processor. It runs on a Micro USB power supply that gives up to 2.5A at 5V, making it a good fit for low-energy environments.

The Smart XOR mechanism, a streamlined yet effective approach, replaced the conventional MixColumns phase in the Modified AES algorithm to improve standard encryption. Initially in the encryption loop, the standard AddRoundKey is succeeded by the SubBytes transformation, executed over GF (2^8) utilizing an S-box, followed by the ShiftRows operation. The primary alteration transpires in Step 4, wherein each 4-bit column of the state matrix is amalgamated into a singular byte, referred to as X, by a column-wise XOR operation as delineated by **Equation (3)**. Subsequently, a round-specific constant Ri is employed to modify this value. To improve diffusion, the output from this phase is spread out across the column positions again, with some changes made for each iteration. The same round constant is used to reverse the procedure during decryption, which lets the original state be recovered exactly, as shown in **Equation (8)**. The Smart XOR approach significantly lowers computational overhead compared to the traditional MixColumns function, while maintaining a favorable security-performance balance. This makes it well-suited for IoT and other resource-constrained environments. Figure 1 illustrates the whole encryption and decryption process using this modified approach.

1. **Implementation Steps of Modified AES Algorithm:**

The encryption process in Modified AES is explained as follows.

$\oplus$: Bitwise XOR

$\|$: Concatenation

$E_{K(\bullet)}$: AES encryption under key K

$GF\left(2^8\right)$: Galois Field used in AES

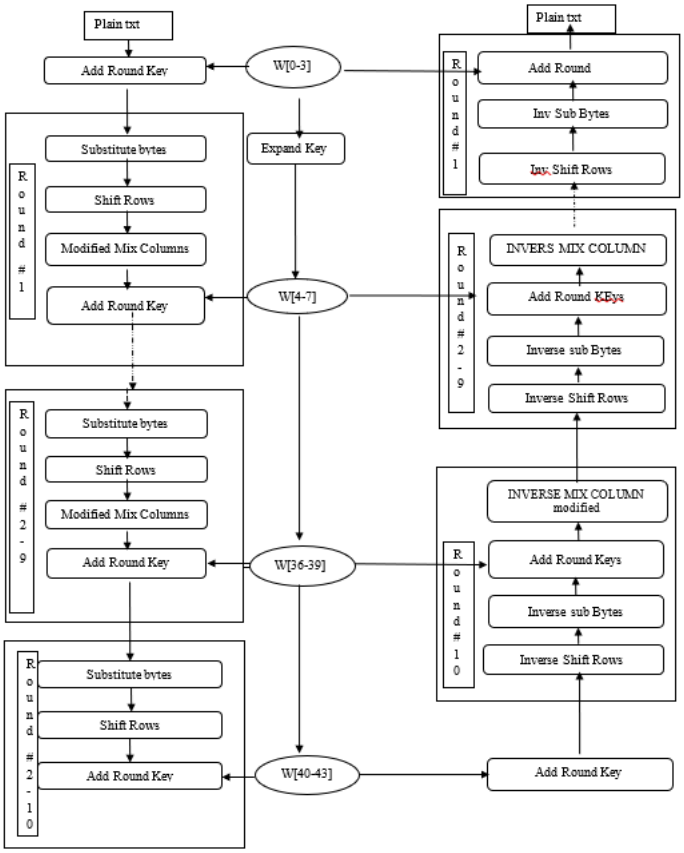

Fig 1. Modified AES encryption and decryption process

$S(\bullet)$: AES S-box substitution

Step 1: AES state and Round Key

Let $S^{(r)}$ be the AES state at round r, and K^r be the round key:

$$S^{(r)} = \{\ s_{i,j}\ \},\ \ K^r = \{\ k_{i.j}\ ^r\}\ \ for\ i,j \in \{0,1,2,3\} \tag{1}$$

These are 4×4 byte matrices.

Step 2: SubBytes (confusion)

Each byte is replaced using the AES S-box:

$$S'\ (i,\ j)\ = S_{box} S(i,j) \tag{2}$$

Step 3: ShiftRows (Diffusion)

Each row i of the state is cyclically shifted by i positions:

$$s''_{i,j} = s'_{i,(j+1)}\ mod 4 \tag{3}$$

where shift(i) is {0,1,2,3} for respective rows.

Step 4: XOR-Triplet Diffusion

Instead of MixColumns, a lighter XOR-Triplet is used:

$$M_{\oplus} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \\ 1 & 0 & 1 & 1 \end{bmatrix}$$

$$y_o = x_0 \oplus x_1 \oplus x_3 \tag{4}$$

$$y_1 = x_0 \oplus x_1 \oplus x_2 \tag{5}$$

$$y_2 = x_1 \oplus x_2 \oplus x_3 \tag{6}$$

$$y_3 = x_0 \oplus x_2 \oplus x_3 \tag{7}$$

This uses an involutory matrix

$$M_{\oplus}^{-1} = M_{\oplus} \tag{8}$$

Step 5: AddRoundKey + Round Constant

Each state byte is XORed with the round key and augmented with a round constant Ri.

Mix in the round-dependent constant to increase non-linearity and round variance:

$$s'''_{\{i,j\}} = s''_{\{i,j\}} \oplus k^{\{(r)\}}_{\{i,j\}} \oplus R_i \tag{9}$$

Step 6: For byte/column variance, the constant can be expanded as:

$$C' = \left[s''_{\{i,j\}} \oplus k^{\{(r)\}}_{\{i,j\}} \oplus R_{\{i,0\}}, ; s''_{\{i,j\}} \oplus k^{\{(r)\}}_{\{i,j\}} \oplus R_{\{i,1\}}, ; s''_{\{i,j\}} \oplus k^{\{(r)\}}_{\{i,j\}} \oplus R_{\{i,2\}}, ; s''_{\{i,j\}} \oplus k^{\{(r)\}}_{\{i,j\}} \oplus R_{\{i,3\}} \right] \tag{10}$$

The decryption process in Modified AES is explained as follows.

a) Step 1: AES State and Round Key

$$S(r) = [si,j], K(r) = [ki,j(r)], \qquad i,j \in \{0,1,2,3\} \tag{11}$$

b) Step 2: Inverse AddRoundKey + Round Constant

$$S'_{\{i,j\}} = S_{\{i,\ (j+i) mod\ 4\}} \tag{12}$$

For expanded per-column variance:

$$C = \left[si,j''' \oplus ki,j(r) \oplus Ri,0,\ si,j''' \oplus ki,j(r) \oplus Ri,1,\ si,j''' \oplus ki,j(r) \oplus Ri,2,\ si,j''' \oplus ki,j(r) \oplus Ri,3 \right] \tag{13}$$

c) Step 3: Inverse XOR-Triplet Diffusion

Since the XOR-Triplet matrix is involutory $(M \oplus -1 = M \oplus M)$

$$x0 = y0 \oplus y1 \oplus y3 \tag{14}$$

$$x1 = y0 \oplus y1 \oplus y2 \tag{15}$$

$$x2 = y1 \oplus y2 \oplus y3 \tag{16}$$

$$x3 = y0 \oplus y2 \oplus y3 \tag{17}$$

d) Step 4: Inverse ShiftRows

Each row i is cyclically shifted right by i positions:

$$si,j' = si,(j-i) mod 4 \tag{18}$$

e) Step 5: Inverse SubBytes (Confusion Removal)

Apply inverse AES S-box to each byte:

$$si,j = S-1\,(si,j') \tag{19}$$

3 Results and Discussion

The proposed algorithm, M-AES, and standard AES are implemented on Raspberry Pi 3 B+, Quad Core 1.2GHz Broadcom BCM2837, and 1 GB RAM, 32-bit Raspbian OS of 5V and current of 2.5A. The Modified AES is implemented on the Kaggle IoT healthcare dataset. To analyse the overall effectiveness and practicality of the proposed cryptographic framework, we define the key performance metrics such as encryption and decryption time, energy consumption, memory usage, and avalanche effect. The hardware implementation results of the Modified AES (M-AES) and the standard AES algorithm are shown in Figure 2a through 2d. It demonstrates a comparison of performance metrics for different data sizes, such as execution time, energy consumption, throughput, and memory use. The results validate that M-AES consistently drops energy usage across all input sizes, making it suitable for resource-constrained devices such as the Internet of Things. This is due to the reduction in execution time. The overall efficiency varies according to the size of the input. The standard AES and M-AES algorithms are tested with input files that range in size from 1 KB to 10 KB. With varying workloads and input sizes, these tests demonstrate how the new AES version functions in comparison to the standard one.

1. **Encryption Time**

   Encryption time is the duration needed to convert plaintext into ciphertext, influenced by data size, system performance, and algorithm efficiency. M-AES consistently demonstrates a reduction in encryption time of approximately 35-41% in comparison to AES across all file sizes as shown in Figure 2a. This indicates enhanced computational efficiency of M-AES compared to regular AES. This efficiency highlights M-AES's suitability for latency-sensitive tasks, especially in embedded systems and real-time communications.

2. **Decryption Time**

   Decryption time refers to the time taken to revert cipher ciphertext into its original plaintext form. Similar to encryption, decryption time varies with ciphertext size, system specifications, and algorithmic complexity. M-AES demonstrates decryption times that are 35-43% faster than AES for all file sizes as shown in Figure 2b. This consistency indicates a distinct computational superiority in both encryption and decryption efficacy.

3. **Energy Consumption**

   Energy consumption reflects the total energy expended during encryption or decryption processes. It is influenced by CPU utilization, algorithm complexity, and execution duration. The energy EE, measured in joules (J), is calculated using **Equation (10)**:

   $$E = Vcc \times I \times t \tag{10}$$

   Where:

   Vcc: Supply voltage (volts)

   I: Current (amperes)

   t: Time taken for encryption or decryption (seconds)

   The energy required for both encryption and decryption is significantly lower in M-AES, as shown in Figure 2c. M-AES uses about 35-44% less energy than AES for both encryption and decryption. This consistent level of efficiency makes M-AES a suitable choice for systems that require energy savings. As illustrated in Figure 2c, M-AES requires significantly less energy to encrypt and decrypt data.

4. **Throughput**

   Throughput is an efficiency metric that indicates how much data can be encrypted or decrypted per second. It is calculated using **equations (11, 12)**:

   $$Encryption\ Throughput = L/Encryption\ Time \tag{11}$$

   $$Decryption\ Throughput = L/Decryption\ Time \tag{12}$$

   Where L is the size of the input data in bytes. High throughput is particularly vital in scenarios demanding real-time data processing or rapid transmission.

The encryption throughput is stable throughout file sizes, ranging from 60.24 to 62.69 KB/s, whereas the decryption throughput exhibits minor variation between 57.69 and 63.21 KB/s. Both processes demonstrate efficient and consistent performance, making them suited for real-time IoT applications. It shows that both encryption and decryption exhibit high throughput across all evaluated file sizes, signifying that the system scales effectively and operates dependably.

5. **Memory consumption**

   It measures the RAM usage of a cryptographic algorithm during execution. Efficient memory usage is essential for lightweight implementations, especially in embedded or resource-limited systems. M-AES substantially minimizes memory consumption relative to AES, achieving reductions of from 9% to 43%, contingent upon the size of the file and operation. Significant reductions are observed in bigger and mid-sized files, demonstrating M-AES's efficacy in memory-limited contexts such as IoT devices.

6. **Cryptanalysis Evaluation**

   The summary comparison Table 2 presents a comparative analysis of Standard AES and the proposed Modified AES through various cryptographic and statistical assessments. The Avalanche Effect indicates that both algorithms attain near-optimal diffusion (≈0.5), with Modified AES exhibiting marginally greater bit sensitivity. Both algorithms show strong resistance to differential cryptanalysis; however, the Modified AES achieves resistance levels comparable to those of Standard AES. The Linear Cryptanalysis bias in Modified AES is lower (0.005 compared to 0.010), suggesting enhanced resistance. Shannon Entropy values approach the theoretical maximum of 8, indicating ciphertext randomness in both cases, with Modified AES exhibiting values between 7.8 and 7.9. The results of the Chi-Square Test indicate a high degree of statistical uniformity, with the Modified AES consistently ranging from 240 to 260. The Monobit Test demonstrates a nearly perfect balance of binary digits (0.500 compared to 0.499), while Byte Correlation is significantly diminished in Modified AES (0.00 versus 0.01), indicating enhanced independence among ciphertext bytes. The results indicate that the Modified AES demonstrates comparable or slightly superior performance to Standard AES regarding diffusion, randomness, and resistance to cryptanalysis.

Table 2. Comparative Analysis of Cryptanalysis Evaluation of the proposed M-AES

Test	Standard AES	M-AES
Avalanche Effect	≈ 0.49	≈ 0.50
Differential Cryptanalysis (mean)	≈ 0.498	≈ 0.501
Linear Cryptanalysis (bias)	≈ 0.010	≈ 0.005
Shannon Entropy	≈ 7.9	≈ 7.9
Chi-Square Test	≈ 250	≈ 240–260
Monobit Test	≈ 0.499	≈ 0.5
Byte Correlation	≈ 0.01	≈ 0.00

3.1 Validation of Results through Comparison with Existing Studies

A comparative analysis is conducted between the results obtained from the proposed technique and those reported in existing literature. This section examines and compares various lightweight encryption techniques to evaluate how effectively they meet the specific needs of healthcare systems based on the Internet of Things. We have compared Block size, key size, throughput, encryption, decryption time, and security are among the important characteristics. Table 3 shows a comparative analysis of lightweight encryption schemes for resource-constrained devices.

The comparative table illustrates the wide range of efficiency and security features of lightweight symmetric encryption techniques. Selecting the optimal ciphers for embedded systems and the Internet of Things requires knowledge of this information. The Proposed M-AES (s/w) performs better than the conventional AES-128 (2.4 μs) with the lowest encryption time of 1.8 μs while maintaining a high degree of security and sensitivity. This makes it perfect for real-time, resource-constrained environments. Similarly, CryptoCore and QARMA offer a strong balance between low latency (2-2.2 μs) and high throughput (150–300 Mbps and 180-250 Mbps, respectively), making them great choices for applications that need speed and security. On the other hand, algorithms such as TEA, XTEA, and KATAN have high encryption delays (5.8-6.1 μs) and lower sensitivity handling, suggesting that they are not appropriate for high-security or latency-sensitive activities.

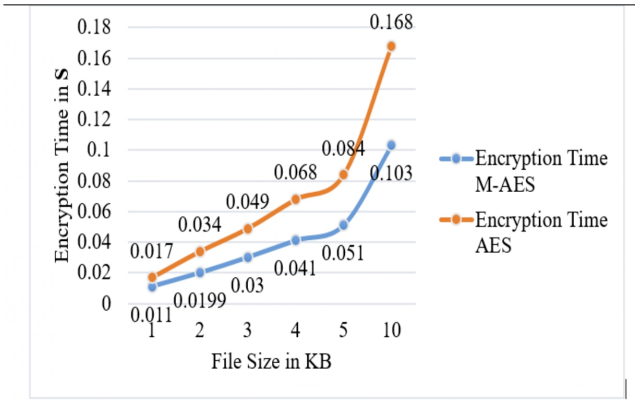

Fig. 2 (a) File Size Vs Encryption Time in Sec

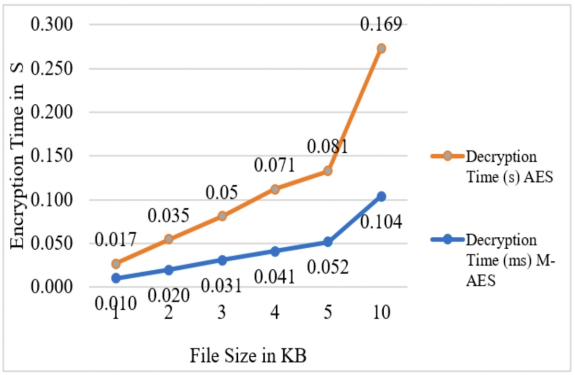

Fig. 2 (b) File Size Vs Decryption Time in Sec

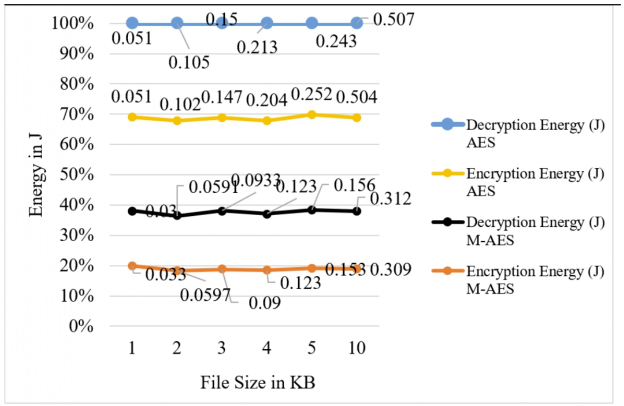

Fig. 2c. File Size Vs Energy Consumption

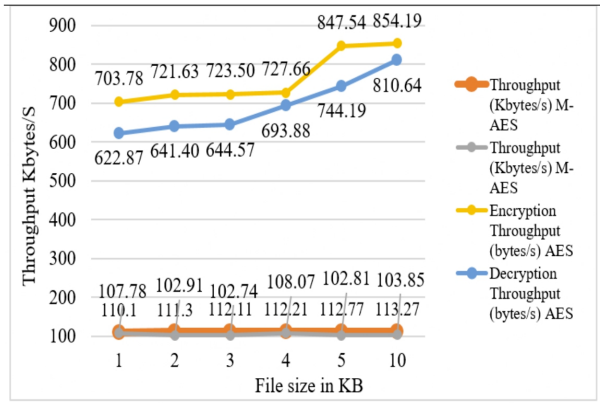

Fig. 2d. File Size Vs Throughput

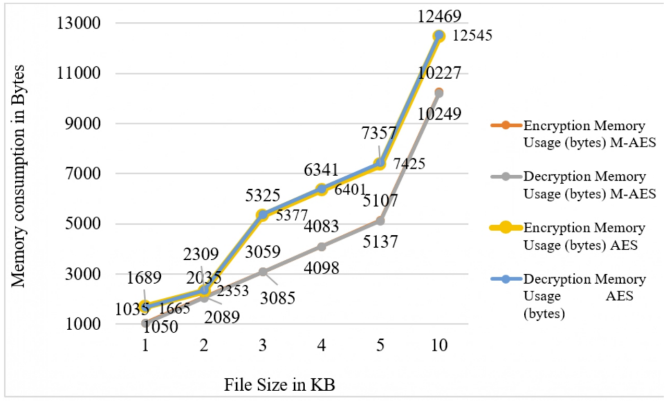

Fig. 2e. File size Vs Memory Consumption

Fig 2. Implementation Results of M-AES

Table 3. Comparative analysis of lightweight encryption schemes for resource-constrained devices

Method	Type	Block Size	Key Size	Throughput (Mbps)	Encryption Time (μs)	Decryption Time (μs)	Security Level	Sensitivity of Data
TEA	Symmetric	64 bits	128 bits	10.49	6.1	6.1	Medium	Low [1]
XTEA	Symmetric	64 bits	128 bits	11.03	5.8	5.8	Medium	Low [1]
CLEFIA	Symmetric	128 bits	128/192/256 bits	51.2	2.5	2.5	High	High [9]
CryptoCore	Symmetric	128 bits	256 bits	64	2	2	High	High [13]
PRESENT	Symmetric	64 bits	80 bits	12.8	5	5	Medium	Medium [17]
HIGHT	Symmetric	64 bits	128 bits	12.55	5.1	5.1	Medium	Medium [17]
KATAN	Symmetric	32/48/64 bits	80 bits	10.67	6	6	Medium	Low [17]
SIMON	Symmetric	32-128 bits	64-256 bits	18.29	3.5	3.5	High	High [18]
SPECK	Symmetric	32-128 bits	64-256 bits	25.6	2.5	2.5	High	High [18]
PICCOLO	Symmetric	64 bits	80/128 bits	14.22	4.5	4.5	Medium	Medium [20]
AES-128	Symmetric	128 bits	128 bits	53.33	2.4	2.4	High	High [22]
Proposed M-AES	**Symmetric**	**128 bits**	**128 bits**	**71.11**	**1.8**	**1.8**	**High**	**High**

4 Conclusion

This study presents Modified AES (M-AES), a revolutionary cryptographic technique designed especially for Internet of Things devices with limited resources. The key innovation is the use of a more effective Smart XOR approach in place of the traditional MixColumns transformation, which significantly lowers computational cost while preserving robust security features. The test results show that M-AES performs much better than AES on important performance metrics such as memory usage, execution time, throughput, and energy consumption. M-AES is more energy-efficient since it takes 35% to 43% less energy to encrypt and decrypt data. For example, when the file size is 1 KB, the energy used for encryption goes down from 0.051 J (AES) to 0.033 J (M-AES). The same reductions can be seen at greater sizes. Execution time is also much shorter, with speeds that are 24% to 41% faster, depending on the size of the file.

Also, M-AES speeds up throughput by as much as 70%, especially for smaller files. For instance, the encryption throughput goes from 60.24 KB/s (AES) to 93.09 KB/s (M-AES) at 1 KB, and the decryption throughput goes up to 75.87% at 2 KB. M-AES still uses less memory than AES, even though the savings are smaller. This makes it easier to utilize in contexts with restricted memory. There is a 178% increase in throughput (71.11 Mbps vs. 25.6 Mbps) and a 28% decrease in processing time compared to SPECK. M-AES has a throughput that is 289% higher than SIMON (71.11 Mbps vs. 18.29 Mbps) and cuts the time it takes to encrypt and decode by 48.6%.

M-AES is an excellent option for next-generation lightweight encryption as it effectively balances performance and security in highly constrained embedded devices.

Future studies may investigate the functionality of M-AES in conjunction with hardware accelerators and adaptive IoT applications, as well as conduct formal cryptanalysis of Smart XOR to assess its security. Evaluating real-time systems with diverse datasets helps enhance their scalability and resilience.

References

1) Sarker KU. A systematic review on lightweight security algorithms for a sustainable IoT infrastructure. *Discover Internet of Things*. 2025;5:47. https://doi.org/10.1007/s43926-025-00150-4.
2) Sabri O, Al-Shargabi B, Abuarqoub A, Hakami TA. A lightweight encryption method for IoT-based healthcare applications: A review and prospects. *IoT*. 2025;6(2):23. https://doi.org/10.3390/iot6020023.
3) Chakrabarty P. Enhanced data security framework using lightweight cryptography and multi-level encryption. *Proceedings of the 2024 International Conference on Communication, Computer Sciences and Engineering (IC3SE)*. 2024;p. 720–725. https://doi.org/10.1109/IC3SE62002.2024.10593191.
4) Soto-Cruz JA. A survey of efficient lightweight cryptography for power-constrained microcontrollers. *Technologies (Basel)*. 2024;13(1):3. https://doi.org/10.3390/technologies13010003.
5) Al-Heayli HAA, Aldabbagh SSM. Efficient substitution box design using modified intelligent Jellyfish Search algorithm. *J Next-Gen Fuzzy Inf Technol*. 2024;1:1–8. https://doi.org/10.69513/jnfit.v1.i0.a5.
6) Zala DK, Khan MA. Novel Dual-Encryption and Compression Strategy: AES and Fernet for Secure Multimedia Transmission. *IEEE International Conference on Intelligent Computing and Networks and Internet of Services (ICICNIS)*. 2024. https://doi.org/10.1109/icicnis64247.2024.10823305.
7) Astillero R. Optimizing Advanced Encryption Standard (AES) for Enhanced File Security in University Networks Using Dynamic Key Expansion. *Technologies: A Global Journal on Technological Developments and Scientific Innovations*. 2024;3(1). https://doi.org/10.62718/vmca.tech-gjtdsi.3.1.sc-1124-004.
8) Pinuyut J, Buol JF, Suwilo S, Zarlis M. Performance Analysis of Modified AES for Efficient Encryption and Decryption. *Jurnal Mantik*. 2024;8(1):233–240. https://doi.org/10.48176/mantik.v8i1.2462.
9) Zhong Y, Gu J. Lightweight block ciphers for resource-constrained environments: A comprehensive survey. *Future Generation Computer Systems*. 2024;157:288–302. https://doi.org/10.1016/j.future.2024.03.054.
10) Li J, Li W, Gao J, Qin M, Sun W. Statistical fault analysis of lightweight tweakable block cipher QARMA in the Internet of Everything. *Journal of Donghua University*. 2024;41(2):172–180. https://doi.org/10.1007/s42235-024-0172-0.
11) Ahirrao M, Pathak B, Dixit M. AES and its multiple modifications for various applications. IEEE. 2023. https://doi.org/10.1109/ICAST59062.2023.10454951.
12) Zakaria AA, Azni AH, Ridzuan F, Zakaria NH, Daud M. Systematic literature review: Trend analysis on the design of lightweight block cipher. *Journal of King Saud University – Computer and Information Sciences*. 2023;35(5):101550. https://doi.org/10.1016/j.jksuci.2023.04.003.
13) Selvapriya ES, Suganthi L. Design and implementation of low-power Advanced Encryption Standard cryptocore utilizing a dynamic pipelined asynchronous model. *Integration*. 2023;93:102057. https://doi.org/10.1016/j.vlsi.2023.102057.
14) Dobraunig C, Eichlseder M, Mendel F, Schläffer M. Ascon v1.2: Lightweight authenticated encryption and hashing. *Journal of Cryptology*. 2021;34(3):33. https://doi.org/10.1007/s00145-021-09398-9.
15) Raheem MA, Balogun AO. An enhanced lightweight Speck system for cloud-based smart healthcare;vol. 1455. Springer. 2021. https://doi.org/10.1007/978-3-030-89654-6_26.
16) Ahmed AA. Lightweight digital certificate management and efficacious symmetric cryptographic mechanism over industrial Internet of Things. *Sensors*. 2021;21(8):2810. https://doi.org/10.3390/s21082810.
17) Rashidi B. Efficient full data-path width and serialized hardware structures of SPONGENT lightweight hash function. *Microelectronics Journal*. 2021;115:105167. https://doi.org/10.1016/j.mejo.2021.105167.
18) Panahi P, Bayılmış Ç, Çavuşoğlu U, Kaçar S. Performance evaluation of lightweight encryption algorithms for IoT-based applications. *Arabian Journal for Science and Engineering*. 2021;46(4):4015–4037. https://doi.org/10.1007/s13369-021-05358-4.
19) Thakor VA, Razzaque MA, Khandaker MRA. Lightweight cryptography algorithms for resource-constrained IoT devices: A review, comparison, and research opportunities. *IEEE Access*. 2021;9:37311–37351. https://doi.org/10.1109/ACCESS.2021.3062534.
20) Ramu G, Mishra Z, Singh P, Acharya B. Performance optimised architectures of Piccolo block cipher for low resource IoT applications. *International Journal of High Performance Systems Architecture*. 2020;9(1):49–57. https://doi.org/10.1504/IJHPSA.2020.107175.
21) Sakamoto K. Tweakable TWINE: Building a tweakable block cipher on generalized Feistel structure. *IEICE Transactions on Fundamentals of Electronics, Communications and Computer Sciences*. 2020;E103.A(12):1629–1639. https://doi.org/10.1587/transfun.2019EAP1141.
22) Acla HB, Gerardo BD. Performance evaluation of lightweight advanced encryption standard hardware implementation. *International Journal of Recent Technology and Engineering*. 2019;8(2):1810–1815. https://doi.org/10.35940/ijrte.B1025.0782195.
23) Gohr A. Improving attacks on round-reduced Speck32/64 using deep learning. Springer. 2019. https://doi.org/10.1007/978-3-030-26954-1_7.
24) Dhanda SS, Singh B, Jindal P. Lightweight cryptography: A solution to secure IoT. *Wireless Pers Commun*. 2020;114:2531–2554. https://doi.org/10.1007/s11277-020-07448-4.