ISSN: 2617-5517 (Print)
Al-Farabi Journal of Engineering Sciences
https://www.iasj.net/iasj/journal/392/issues
Published by Al-Farabi University College

Enhanced Security of Advanced Encryption Standard (AES) Algorithm

Louay Flaieh Hasan

Al-Farabi University College

Corresponding Author: louay.hasan@alfarabiuc.edu.iq

ABSTRACT

The S-Box, also known as the substitution box, plays a crucial role in the advanced encryption standard AES, introduced in 2001 by Rijndael, affiliated with the US National Institute of Standards and Technology (NIST). The S-Box functions as a non-linear byte substitution, independently operating on each byte of a 4x4 matrix. Its purpose is to prevent attacks that exploit simple algebraic properties used in AES and enhance nonlinearity. To ensure the main principles established by Shannon for designing block cipher algorithms (confusion and diffusion), AES effectively incorporates a 16x16 byte two-dimensional S-Box. Substitution is achieved through the utilization of SubByte transform, which is used in keyschedule and in the encryption/decryption process at each round. SubByte serves as a fundamental component of the round function. The linear attack is widely recognized as one of the most prominent attack methods targeting the AES encryption algorithm. It exploits the S-box by analyzing the input-output pairs at specific positions. This paper provides an in-depth exploration of this attack and its mechanisms. In addition to employ eight S-Boxes constructed according to eight different irreducible polynomials over GF(2^8). Based on the round key, two of these S-Boxes are selected for each round, with one allocated for key schedule and another for encryption/decryption. Among the proposed S-Boxes, four are utilized in the key schedule operation, while the remaining four are used for the encryption/decryption operation. The discussion covers the effect of increasing algorithm complexity on the attacker and algorithm security. The randomness properties of the generated sequences were compared between the proposed and original algorithms using NIST tests to assess their level of randomness. Additionally, the enhanced algorithm's compliance with cipher design criteria is assessed through Avalanche factor and Strict Avalanche Criterion (SAC) testing. Finally, this study concludes that all these measures are successful, indicating that the proposed algorithm is suitable for cryptographic applications.

Keywords: S-Box, Block, AES, Encryption, Decryption, Algorithm, SubByte, Substitution.

1. Introduction

The United States government selected the Advanced Encryption Standard (AES), a symmetric block encryption, to protect sensitive data. it is widely used in both hardware and software globally to encrypt important information, playing a crucial role in ensuring the security of government computers, cybersecurity, and the protection of electronic data (Siddiqi et al., 2021).

In 1997, the US national institute of standards and technology (NIST) initiated the development of AES as an alternative to the vulnerable data encryption standard (DES), which was increasingly susceptible to brute-force attacks. NIST emphasized that the new encryption algorithm should be unclassified and capable of effectively protecting sensitive government information well into the 21st century (Banoth & Regar, 2023). It was designed to be easily implementable in both hardware and software, while also providing strong defenses against various attack techniques, even in restricted environments like smart cards (Aissaoui et al., 2023).

AES was primarily created for the united states government but is also available for voluntary, free use in public or private, commercial or noncommercial programs that provide encryption services. It consists of three versions, each utilizing a different cipher that operates on 128-bit blocks and employs cryptographic keys of 128, 192, and 256 bits, respectively. These versions are known as AES-128, AES-192, and AES-256.

It is a secret key encryption algorithm, where the same key is used for both encryption and decryption. Therefore, the sender and receiver must possess and utilize the same secret key. The united states government classifies information into three levels: confidential, secret, and top secret (Kshirsagar et al., n.d.). For the protection of confidential and secret information, all key lengths can be used. However, top secret information requires either AES-192 or AES-256. AES-128 consists of 10 rounds, AES-192 has 12 rounds, and AES-256 has 14 rounds. Each round involves various processing steps, such as substitution, transposition, and mixing of the input plaintext to produce the ciphertext output.

Substitution is achieved through the utilization of SubByte, which is used in generating round keys for subkey updates and in the encryption/decryption process. SubByte acts as a non-linear byte substitution, operating independently on each byte of a 4x4 byte 2D matrix. It is implemented as a 16x16 byte matrix. For practical purposes, it has been determined that 128-bit keys offer sufficient security (de Ree et al., 2022).

So, the objective of this study is to enhance the complexity and security of the AES-128 version. The proposal involves the inclusion of eight S-Boxes, each consisting of a 16x16 byte 2D structure. A comparison was conducted to evaluate the complexity, performance, and statistical randomness of both the standard AES-128 and the proposed enhancement.

The study organization will cover the concepts of confusion and diffusion in section 2. In section 3, the AES linear attack is explained. The basic structure of AES is covered in section 4, and the decryption process is discussed in section 5. The proposed algorithm is then explored in section 6. Next, the experimental results are presented in section 7. Finally, the conclusions are explained in section 8.

2. Confusion and Diffusion

When designing a block cipher algorithm, it is important to consider two fundamental principles established by Shannon: confusion and diffusion

S-Box, these relationships would hold true (and false) at the same time with a probability of 0.5 for all bit positions at the input and output (Yang et al., 2023).

A linear attack can exploit this fact to make accurate estimations for the key bits, particularly in the final round of a block cipher. It is crucial to note that this relationship must hold true for all possible combinations of 0's and 1's at the relevant bit positions. For instance, if the list of output bit positions is empty, there will be 128 out of 256 possible bit patterns at the input that satisfy the linear relationship described in Formula (2).

$$x_{i1} \otimes x_{i2} \ldots \otimes x_{im} \otimes y_{i1} \otimes y_{i2} \ldots \otimes y_{in} = 0 \quad (2)$$

In this case, when an input bit pattern contains an even number of 1's, the Exclusive-OR sum will be zero. (This occurs in 128 out of 256 cases.) The same applies when considering an empty set of input bits and all possible variations of output bits. It is important to note that a linear attack exploits not only the bit positions where the linear relationship is often true, but also the positions where such relationships are often false. The inequality case of these linear relationships is more accurately referred to as affine relationships. As mentioned earlier, for an ideal S-Box, all linear (and affine) relationships of this kind hold with a probability of 0.5 (Liu et al., 2021). In other words, when you input the 256 possible bit patterns into an S-Box, these relationships should hold 128 times for all possible combinations of input and output bit positions. Any deviation from this average is known as linear approximation bias. It is this bias that a linear attack on a block cipher exploits.

4. AES Basic Structure

(Shah et al., 2023). AES effectively incorporates these properties through the SubByte component, which applies substitution for confusion and permutation for diffusion. In the SubByte transformation, each byte of the state matrix undergoes a substitution using a 16x16 substitution matrix, as well as a two-step data permutation procedure (Mamvong, 2023):

- Inversion: Each byte is multiplicatively inverted.
- Affine Transformation is performed over GF(2) mod irreducible polynomial $x^8 + x^4 + x^3 + x + 1$ and is defined as follows:

$$b = X(S_r, C)^{-1} + y \quad (1)$$

Where S_r is state matrix row, C is state matrix column, and y is constant.

Furthermore, the permutation operation is accomplished by utilizing the linear components ShiftRows and MixColumns.

3. AES Linear Attack

A linear attack on a block cipher is a known plaintext attack. In this type of attack, the attacker has access to a subset of plaintexts and their corresponding ciphertexts (Alawida et al., 2022). Unlike a differential attack, the attacker does not specifically select a subset of these plaintexts. Instead, a linear attack exploits linear relationships between the input bits to the S-Box and the output bits. Let the pattern $(x_0, \ldots, x_7)$ represent the input bits to the S-Box, and the pattern $(y_0, \ldots, y_7)$ represent the output bits. If certain bit positions $i_1, \ldots, i_m$ at the input and bit positions $j_1, \ldots, j_n$ at the output exist, such that the following relationship is frequently true or false: “often true” indicates 'significantly above average', and “often false” indicates 'significantly below average. For an ideal

4.3 Mixed Columns (Mixcol)

This process is also a linear diffusion transformation. It involves multiplying a column vector over GF(2^8) modulo the irreducible polynomial $x^8 + x^4 + x^3 + x + 1$ with a fixed matrix. In this process, bytes are treated as polynomials of degree less than 4.

4.4 Add Round Key (AddRoundKey)

In each round of the AES process, every byte of the array is added to a corresponding byte from the round subkeys over GF(2). The round keys are generated through a procedure known as key schedule or round key expansion.

These sub-keys are derived from the primary keys (PK) by XORing two previous columns. For columns that are multiples of four, the process includes addition of round constants, byte substitution, and shift operations.

5. Decryption Process

The decryption process follows the same structure as encryption, but with the inverses of SubByte, ShiftRow, and MixColumn operations used instead. These inverses are SubBytes inverse, ShiftRows inverse, MixColumns inverse, and AddRoundKey. Therefore, the same hardware can be used for both processes. However, the inverse MixColumn operation requires matrix elements that are more complex compared to {01}, {02}, or {03} used in the forward operation. This leads to slightly more complex deciphering hardware. The operations discussed above are illustrated in Figure 1.

A comprehensive explanation of AES can be found in FIPS 197 (Mouha & Dworkin, 2021). However, for the purpose of comprehension, we will outline the structure of AES in this section. AES is a block cipher developed to address the limited key size of the Data Encryption Standard (DES). To provide a mathematical description of AES, it is defined over GF (2^8). The operation is divided into rounds, with each round consisting of four components that process data at either the byte or bit level (Patel & Desai, n.d.). The byte structure is suitable for low-profile microprocessors, such as 8-bit CPUs and microcontrollers. The array of bytes, organized as a 4x4 matrix, is referred to as the "state". The four steps that define one round of AES, BytesSub, ShiftRow, MixColumn, and AddRoundKey, are also known as layers. Except for the first and last round, AES with a 128-bit round key proceeds for nine iterations. In the first round of encryption, an XOR operation is performed with the original key, and in the last round, the MixColumn transformation is skipped.

4.1 Byte Substitution (SubByte)

This operation involves a non-linear byte substitution. It consists of two sub-transformations: multiplicative inverse and affine transformation. In typical implementations, these two sub-steps are combined into a single table lookup known as a substitution box or S-box.

4.2 Shift Rows (ShiftRow)

This process is a linear diffusion transformation that operates on each individual row of the array. It circular shift each row by a specific number of byte positions.

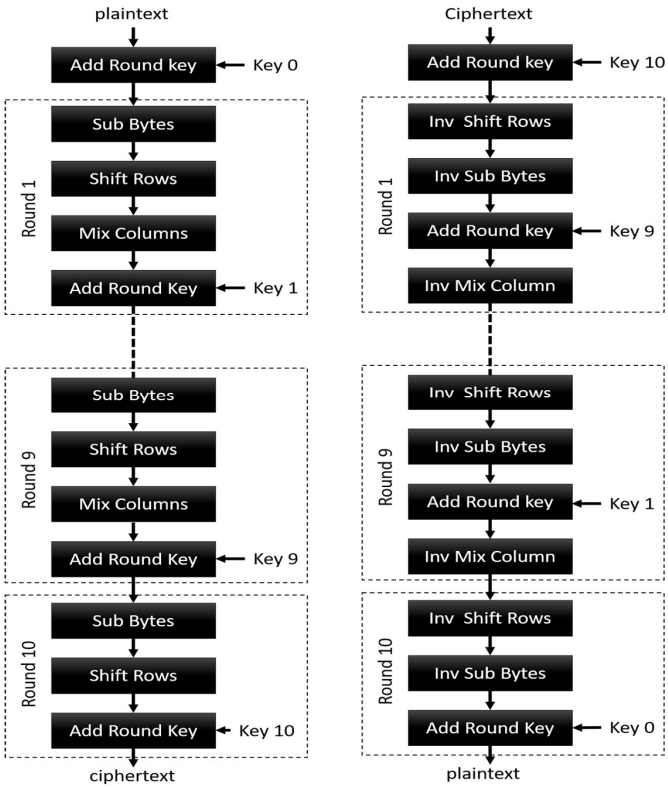

Figure 1. AES Structure Block Diagram

The input key undergoes expansion, resulting in an array of key schedule words, where each word is composed of four bytes. In the case of an AES-128 key, the complete key schedule comprises 44 words. Figure 2 illustrates the block diagram of the key schedule stage in the AES algorithm.

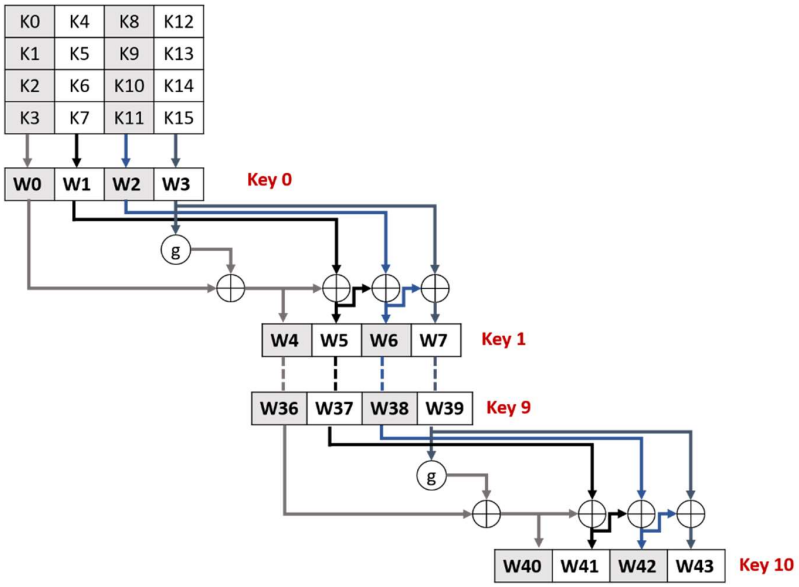

Figure 2. AES Keyschedule

(Ghosh, 2023). The extended Euclidean algorithm was used to calculate the MI. These S-Boxes are round key-dependent, which means that the S-box selection in each round is based on the key round and number of rounds. These proposed round key-based S-boxes make it more challenging for attackers to perform offline analysis of them.

Table 1. Irreducible polynomials (Ghosh, 2023)

index	Polynomial
1	$x^8 + x^4 + x^3 + x + 1$
2	$x^8 + x^5 + x^3 + x + 1$
3	$x^8 + x^6 + x^4 + x^3 + x^2 + x + 1$
4	$x^8 + x^6 + x^5 + x + 1$
5	$x^8 + x^6 + x^5 + x^2 + 1$
6	$x^8 + x^6 + x^5 + x^3 + 1$
7	$x^8 + x^7 + x^6 + x + 1$
8	$x^8 + x^4 + x^3 + x^2 + 1$

Figure 3 shows the block diagram for a proposed AES algorithm.

6. Enhancement Proposal Algorithm

In recent years there were many cryptanalysis attempts on block ciphers. Most of the attacks prove its effectiveness either on the simplified version or the full version of AES. For this reason, we proposed this cipher to help satisfy the current and foreseeable future requirements.

6.1 Key-Dependent S-Boxes Construction

The use of a single S-box means that the same S-box is used in every round. This single S-box can be analyzed by attackers to identify vulnerabilities. In this paper, it was proposed to construct 8 S-boxes independently over GF(2^8) using the multiplicative inverse (MI) module the irreducible polynomial 1 to 8, as shown in Table 1

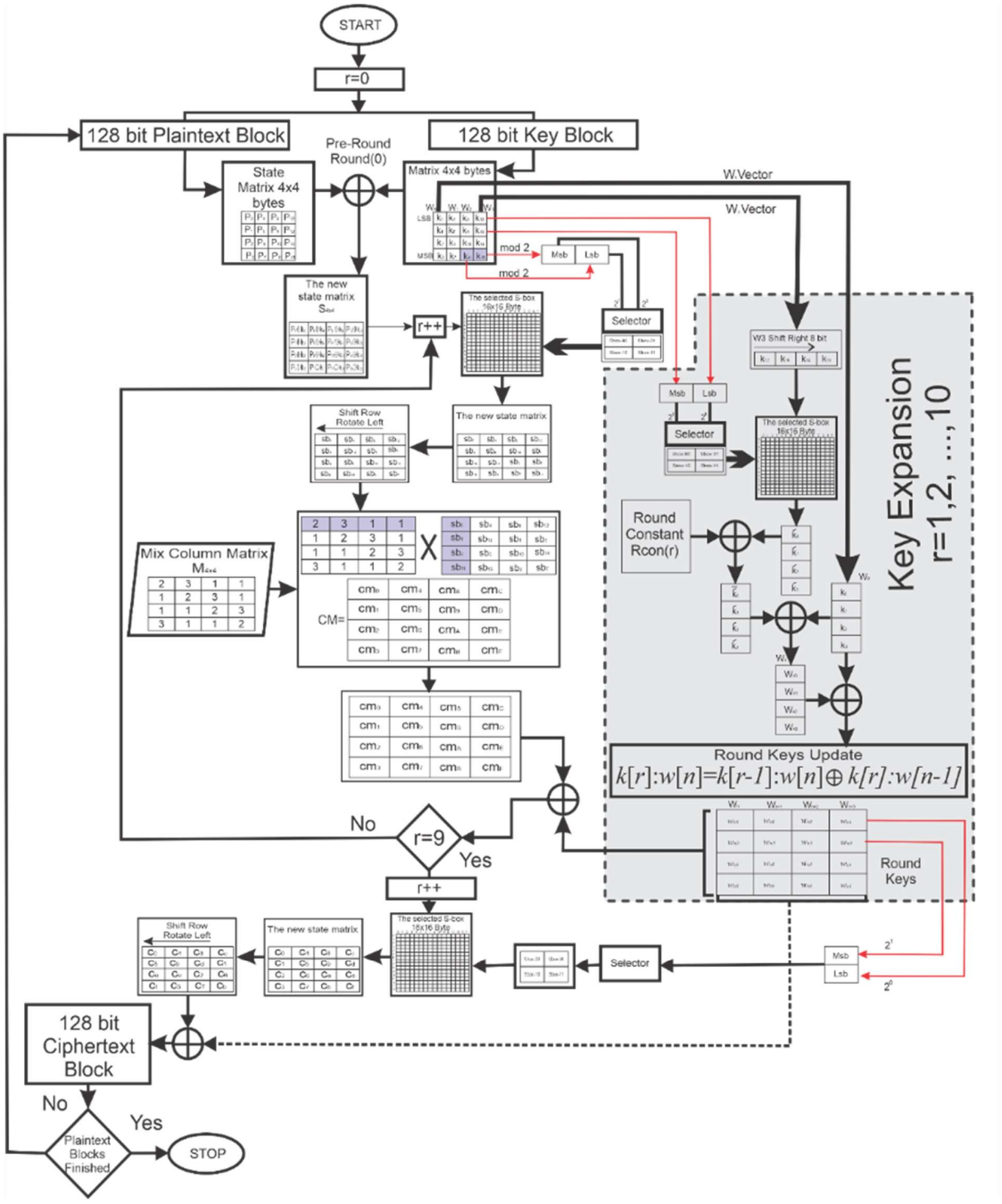

Figure 3. The proposed AES

allocated for the encryption function. Similarly, the round key word w3 determines a value between (0..3) by considering the least significant bits of the first and second bytes. This value is used to identify one of the four proposed S-boxes allocated for the key schedule. The selection of these S-boxes is illustrated in Figure 4.

6.2 Round S-Boxes Selecting

At each round, a key word w2, w3 determines a value between (0..3) by considering the least significant bits of the last byte of each. This value is used to identify one of the four proposed S-boxes

been numerous attempts to cryptanalyze the cipher. To ensure that our proposed design enhances the security of AES, we will conduct two primary evaluation tests: the NIST Statistical Test (comprising 14 tests) and an Avalanche.

7.1 NIST Randomness Tests Suite

The NIST Statistical Test Suite is a widely used tool for analyzing the randomness of sequences. The success of the test is determined by comparing the calculated P-value with the pass mark, $\alpha = 0.01$. If the calculated P-value is greater than the α value, the test is considered successful; otherwise, it is deemed a failure.

A total of 4 files of different types and sizes were prepared to be encrypted using both the original AES algorithm and the proposed algorithm. The NIST randomness tests suit experiments were conducted using identical keys for both algorithms encrypted these files. The results were obtained and are as presented in Table 2.

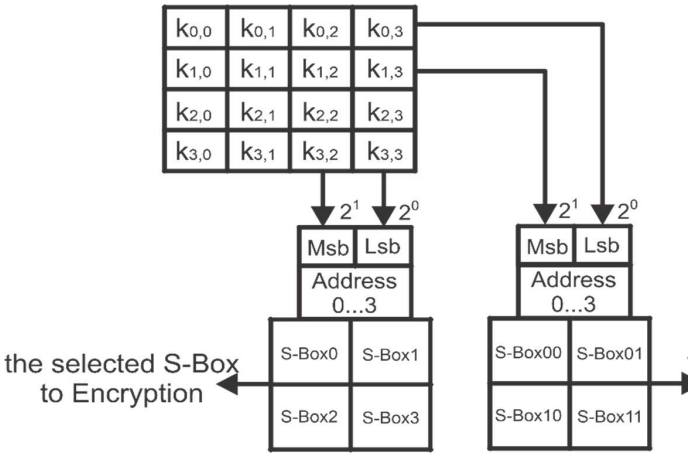

Figure 4. S-Boxes Selecting

During decryption, the Inverse SubBytes operation utilizes the Inverse S-box to perform the inverse operation and return to the original value.

7. Experimental results

Having a strong block cipher does not guarantee better security automatically. It is crucial for a block cipher to withstand cryptanalysis, which involves identifying weaknesses and compromising cryptosystems through attacks. Since Rijndael was adopted as the AES standard in 2001, there have

Table 2. NIST Randomness tests results

N	Test	Para.	Par. Val	Text			Image			Audio			Video		
				P-Value		Result	P-Value		Result			Result	P-Value		Result
				Std	Mod		Std	Mod		Std	Mod		Std	Mod	
1	Frequency	-	-	0.302	0.290	Pass	0.680	0.898	Pass	0.410	0.731	Pass	0.680	0.1067	Pass
2	Frequency within a Block	Block Length	300000	0.929	0.376	Pass	0.579	0.051	Pass	0.751	0.263	Pass	0.579	0.162	Pass
3	Runs	-	-	0.542	0.628	Pass	0.254	0.442	Pass	0.109	0.776	Pass	0.254	0.757	Pass
4	Longest Run	-	-	0.532	0.744	Pass	0.171	0.966	Pass	0.9645	0.201	Pass	0.171	0.797	Pass
5	Binary Matrix Rank	Rows	400	0.746	0.164	Pass	0.691	0.122	Pass	0.018	0.456	Pass	0.691	0.020	Pass
		Columns	400												
6	discrete Fourier transform	-		0.131	0.135	Pass	0.211	0.321	Pass	0.015	0.018	Pass	0.083	0.311	Pass
7	Non-OverLapping Template Matching	Block Length	3156	0.761	0.396	Pass	0.198	0.357	Pass	0.530	0.353	Pass	0.198	0.819	Pass
		Template Length	9												
		Number of Patterns	49												
8	OverLapping Template Matching	Block Length	300	0.062	0.047	Pass	0.384	0.339	Pass	0.999	0.997	Pass	0.384	1.000	Pass
		Template Block Length	8												
		Chi^2	10.497											0.998	
9	Maurer Universal Statistical	Block Length	4	0.995	0.995	Pass	0.999	0.999	Pass	0.996	0.995	Pass	0.999	0.917	Pass
		Number of Blocks of the Test Segment	6296												
		Sequence Length int bits	64												

N	Test	Para.	Par. Val	Text			Image			Audio			Video		
				P-Value		Result	P-Value		Result			Result	P-Value		Result
				Std	Mod		Std	Mod		Std	Mod		Std	Mod	
		for the Initial Segment													
10	Linear Complexity	Block Length	15	0.992	0.847	Pass	0.965	0.203	Pass	0.488	0.270	Pass	0.484	0.471	Pass
		Chi^2	7.961												
11	Serial	Extended Length	3	0.744	0.469	Pass	0.781	0.696	Pass	0.077	0.409	Pass	0.781	0.471	Pass
		Chi^2(1)	1.952												
		Chi^2(2)	0.481	0.786	0.328		0.866	0.446		0.075	0.151		0.866	0.654	
12	Approximate Entropy	m Patterns	3	0.819	0.569	Pass	0.794	0.962	Pass	0.011	0.5484	Pass	0.794	0.834	Pass
13	Cumulative Sums	-	-	0.336	0.402	Pass	0.543	0.294	Pass	0.700	0.234	Pass	0.543	0.166	Pass
				0.372	0.578		0.812	0.375		0.517	0.443		0.812	0.201	
14	Random Excursions					Pass			Pass			Pass			Pass
	X=-4	-	-	0.355	0.043		0.591	0.837		0.313	0.746		0.591	0.695	
	X=-3	-	-	0.163	0.057	Pass	0.273	0.133	Pass	0.246	0.505	Pass	0.273	0.2627	Pass
	X=-2	-	-	0.012	0.011	Pass	0.212	0.524	Pass	0.090	0.463	Pass	0.212	0.861	Pass
	X=-1	-	-	0.135	0.702	Pass	0.162	0.069	Pass	0.398	0.127	Pass	0.162	0.644	Pass
	X=1	-	-	0.712	0.964	Pass	0.042	0.735	Pass	0.937	0.212	Pass	0.042	0.702	Pass
	X=2	-	-	0.436	0.998	Pass	0.105	0.750	Pass	0.616	0.162	Pass	0.105	0.265	Pass
	X=3	-	-	0.633	0.999	Pass	0.025	0.570	Pass	0.687	0.730	Pass	0.025	0.592	Pass
	X=4	-	-	0.775	0.999	Pass	0.062	0.319	Pass	0.950	0.049	Pass	0.062	0.232	Pass
15	Random Excursions Variant					Pass			Pass			Pass			Pass
	X=-9	-	-	0.431	0.170		0.686	0.973		0.716	0.750		0.686	0.011	
	X=-8	-	-	0.403	0.144	Pass	0.561	0.970	Pass	0.762	0.924	Pass	0.561	0.022	Pass
	X=-7	-	-	0.359	0.556	Pass	0.672	0.937	Pass	0.600	0.866	Pass	0.672	0.042	Pass
	X=-6	-	-	0.226	1.000	Pass	0.810	0.765	Pass	0.694	0.633	Pass	0.8100	0.034	Pass
	X=-5	-	-	0.125	1.000	Pass	0.907	0.706	Pass	0.704	0.653	Pass	0.907	0.054	Pass
	X=-4	-	-	0.082	0.789	Pass	0.943	0.831	Pass	0.522	0.712	Pass	0.943	0.289	Pass
	X=-3	-	-	0.072	0.752	Pass	0.945	0.849	Pass	0.466	0.278	Pass	0.950	0.396	Pass
	X=-2	-	-	0.043	1.000	Pass	0.828	0.935	Pass	0.357	0.613	Pass	0.828	0.2445	Pass
	X=-1	-	-	0.052	1.000	Pass	0.281	0.479	Pass	0.181	0.7111	Pass	0.281	0.206	Pass
	X=1	-	-	0.270	0.479	Pass	0.013	0.396	Pass	0.1812	0.092	Pass	0.013	0.693	Pass
	X=2	-	-	0.765	0.683	Pass	0.104	0.514	Pass	0.194	0.276	Pass	0.104	0.732	Pass
	X=3	-	-	0.862	0.752	Pass	0.345	0.487	Pass	0.281	0.246	Pass	0.345	1.000	Pass
	X=4	-	-	0.883	0.789	Pass	0.489	0.521	Pass	0.622	0.247	Pass	0.489	0.632	Pass
	X=5	-	-	0.965	0.814	Pass	0.839	0.278	Pass	0.896	0.213	Pass	0.839	0.429	Pass
	X=6	-	-	0.740	0.831	Pass	0.671	0.067	Pass	0.637	0.059	Pass	0.671	0.26	Pass
	X=7	-	-	0.706	0.844	Pass	0.567	0.050	Pass	0.343	0.057	Pass	0.567	0.292	Pass
	X=8	-	-	0.867	0.855	Pass	0.757	0.290	Pass	0.454	0.077	Pass	0.757	0.374	Pass
	X=9	-	-	0.950	0.864	Pass	0.750	0.631	Pass	0.559	0.057	Pass	0.750	0.514	Pass

7.2 Avalanche Factor Measurements

According to the Strict Avalanche Criterion (SAC), flipping any input bit should result in exactly half of the output bits changing. whereas The Avalanche criterion, also known as the Avalanche factor, describes the phenomenon where a small alteration in the plaintext or key leads to significant modifications in the ciphertext (August & Smith, 2023). This behavior is observed in block ciphers with an n-bit input and m-bit output. If a block cipher fails to exhibit the avalanche effect to a significant extent, it is considered to have poor randomization. In a high-quality block cipher, even a minor modification in the key or plaintext should cause a drastic change in the ciphertext. In this study, we evaluated 4 different data types and sizes using both the original and proposed algorithms. The SAC test was conducted to assess the confusion and diffusion properties, and the results are presented in Table 3. These results demonstrate that the enhanced version of AES improves the security of the encryption by introducing both the confusion and diffusion properties.

Table 3. The SAC and Avalanche Factor Tests Results

N.	File Type	File Length in Bits	Avalanche Factor for Plaintext change	Avalanche Factor for Key change	SAC For Plaintext change	SAC For Key change
1	Text	13696	0.5033	0.5077	0.4979	0.5077
2	Image	706304	0.5008	0.5004	0.5077	0.5004
3	Audio	2358400	0.4998	0.5001	0.5002	0.5001
4	Video	23108864	0.5001	0.5000	0.5002	0.5000

The results from Tables 2 and 3 indicate that the sequences generated by both the standard and proposed algorithms have very similar statistical properties. This means that the proposed algorithm effectively increased

the complexity and security level by enhancing resistance to linear attacks, while maintaining the statistical properties and randomness of the generated sequences.

8. Conclusions

AES is designed to provide strong protection against classic attacks like linear cryptanalysis. However, the algebraic nature of Rijndael, which relies on simple algebraic properties, makes it vulnerable to a linear attack that exploits the presence of a single compensation table repeated in each cycle during encryption and key scheduling. This paper presents a new design that aims to enhance the security of the AES algorithm against this potential vulnerability. The proposed approach ensures that the original AES algorithm remains mathematically unchanged while improving its security. Additionally, the results obtained from implementing NIST statistical tests and Avalanche tests show that the properties of the sequences generated by the proposed algorithm preserve the good random statistical properties of the sequences generated by the standard AES. This paper proposes the construction of 8 different multiple S-boxes, where two of these S-boxes are selected at each round based on the round key. These S-boxes are built using 8 different polynomials on the GF(2^8) field.

References

Aissaoui, R., Deneuville, J.-C., Guerber, C., & Pirovano, A. (2023). A survey on cryptographic methods to secure communications for UAV traffic management. *Vehicular Communications*, 100661.

Alawida, M., Teh, J. Sen, Mehmood, A., & Shoufan, A. (2022). A chaos-based block cipher based on an enhanced logistic map and simultaneous confusion-diffusion operations. *Journal of King Saud University-Computer and Information Sciences*, *34*(10), 8136–8151.

August, D. A., & Smith, A. C. (2023). PudgyTurtle Mode Resists Bit-Flipping Attacks. *Cryptography*, *7*(2), 25.

Banoth, R., & Regar, R. (2023). *Classical and Modern Cryptography for Beginners*. Springer Nature.

de Ree, M., Mantas, G., & Rodriguez, J. (2022). A cryptographic perspective to achieve practical physical layer security. *GLOBECOM 2022-2022 IEEE Global Communications Conference*, 4038–4043.

Ghosh, D. (2023). *The Improvement of Period of Pseudo Random Number Sequence: an Algebraic Approach*.

Kshirsagar, K., Patekar, P., Kolhe, S., Nimbalkar, N., & Pathak, P. N. (n.d.). *SECURE CLOUD STORAGE WITH DEDUPLICATION TECHNIQUE*.

Liu, F., Isobe, T., & Meier, W. (2021). Cryptanalysis of full LowMC and LowMC-M with algebraic techniques. *Advances in Cryptology–CRYPTO 2021: 41st Annual International Cryptology Conference, CRYPTO 2021, Virtual Event, August 16–20, 2021, Proceedings, Part III 41*, 368–401.

Mamvong, J. (2023). *Efficient Security Algorithm for Provisioning Constrained Internet of Things (IoT) Devices*.

Mouha, N., & Dworkin, M. (2021). Review of the advanced encryption standard. *Technical Report, National Institute of Standards and Technology*.

Patel, D. D., & Desai, S. (n.d.). *Securing textual information with an image in the image using a visual cryptography AES algorithm.*

Shah, T., Khan, D. A., & Ali, A. (2023). Design of nonlinear component of block cipher using quaternion integers. *Multimedia Tools and Applications*, 1–18.

Siddiqi, M. A., Tsintzira, A.-A., Digkas, G., Siavvas, M. G., & Strydis, C. (2021). Adding security to implantable medical devices: Can we afford it? *EWSN*, 67–78.

Yang, S., Tong, X., Wang, Z., & Zhang, M. (2023). S-box generation algorithm based on hyperchaotic system and its application in image encryption. *Multimedia Tools and Applications*, 1–25.