Assessing the Impact of Increased MixColumns on AES Encryption Security and Performance

Prathmesh Deshkar[1] Ritesh Pandey[2]
[1]Diploma Student [2]Lecturer & Head of the Department
[1,2]Department of Computer Science and Engineering
[1,2]Government Polytechnic College, Betul, Madhya Pradesh, India

Abstract — This work focuses on the influence of design changes in the 192-bit AES, specifically in the MixColumns step, aiming to enhance security metrics. We evaluated key parameters such as entropy, avalanche effect, Key Input Variation Analysis, and execution time. Our results suggest that entropy due to the proposed change in the AES ranges approximately from 4.0535 to 4.4183, compared to the standard AES range of approximately 3.9183 to 4.3349. The average avalanche effect ratio improved to approximately 0.5094 from 0.4844 in the standard AES. Additionally, the Hamming distance increased to approximately 75.4 from 71.6, indicating higher sensitivity to input changes and better resistance to differential cryptanalysis. These enhancements in security, however, come with an approximate increase in execution time of 10%, which is significant. Despite this, the performance trade-offs may be justified for the improved security observed in this study. Nonetheless, the increase in execution time remains a crucial consideration when selecting the modified AES for specific applications. These findings provide valuable insights for decision-makers balancing the choice between highly secure cryptographic techniques and optimal performance.

Keywords: AES, Rijndael; 192-bit AES; MixColumns; Block Cipher; Cryptography

I. INTRODUCTION

In the digital security world, cryptography stands as the basis for ensuring data security features, confidentiality, integrity, and the assurance of the source. There are so many cryptographic algorithms, The Advanced Encryption Standard (AES) is widely recognized for its robust security features and efficiency [1]. It was established in the Federal Information Processing Standards Publication by the U.S. Department of Commerce/N.I.S.T. on November 26, 2001, AES became the benchmark for data encryption within the majority of sectors, including government, finance, and the telecommunication industry. Looked from different standpoints, it is its enormous dependability, efficiency, and adaptability that establish AES as dominant.

AES offers key sizes of 128 bits, 192 bits, or 256 bits and operates on a fixed block size of 128 bits. It employs a series of well-defined steps, including SubBytes, ShiftRows, MixColumns, and AddRoundKey, to transform the plaintext into ciphertext. Each of these steps is crucial for ensuring the diffusion and confusion properties that are essential for cipher security. Specifically, MixColumns transformation plays a vital role in achieving diffusion, where the influence of each plaintext byte spreads across multiple ciphertext bytes, thus preventing cryptanalysis attempts.

The MixColumns process is a process of matrix multiplication over a finite field that combines the data in each column of the state matrix [2]. This operation guarantees that, through a one-byte modification in the input, all the four bytes in the output are affected, adding to the properties of robust encryption. The MixColumns step is crucial for diffusion, and its correctness directly influences the security of AES encryption.

In the present work, a new modification of the structure of the standard 192-bit AES algorithm is examined. In this work, we analyze the execution of the MixColumns step in the encryption process of 192-bit AES, increasing that from the standard 11 times up to 12 times. Such a modified variant could enhance the diffusion characteristics of the cipher and, perhaps, its resistance to cryptographic attacks. The standard 192-bit AES is compared systematically with our modified variant to provide insights into the trade-offs between enhanced security and the related performance overhead.

This study had two specific objectives: the first was to investigate the difference in ciphertext strength between regular AES and the modified version through parameters such as the avalanche effect and entropy, and Key Input Variation Analysis; the second was to gauge the performance by measuring encryption and decryption time. The study also sought to academically quantify the extent to which these modifications have influenced the encryption and decryption processes per se. To rigorously evaluate the implications of our modifications, this paper aimed at giving a nuanced understanding of the impact the changes may have on the overall cryptographic framework.

This also allows for judging the judiciousness of omitting the MixColumns operation in the final round of the AES algorithm.

A. Background Motivation

A background for this research is laid in the continuous search for stronger cryptographic protocols amidst the changing landscape of cyber threats. As computational power increases, the sophistication of the potential attacks grows. For sure, standard AES is robust, but it is by no means unassailable against advances in cryptanalytical techniques. This calls for an exploration of enhancements that can fortify the algorithm without turning it impractical in terms of execution speed or resource use.

The selection to increase the MixColumns was made due to its importance in diffusion. Cryptographically, diffusion ensures that after applying the cryptographic function, the output (ciphertext) does not have a pattern that an attacker can recognize.

Our main objective is to analyze quantitatively the repercussions of regularly performing the MixColumns operation for every round in the AES algorithm, as opposed to doing it irregularly and skipping it for the last round. We aim to answer whether this implementation will make the

system less secure, cost more in terms of performance, or suffer if implementation resources are saved.

B. Methodological Overview

In order to make a strict comparison between the standard 192-bit AES and our modified variant, we adopted a structured methodology for justification, including a theoretical analysis together with empirical testing. Our approach is to clear this doubt by implementing both of the algorithms in a controlled environment, with a consistent set of plaintext inputs to bring out the comparison. Implementation was evaluated in two categories:

1) Ciphertext Strength

We analyzed the randomness and diffusion properties of the ciphertexts generated by both algorithms. Metrics such as entropy, avalanche effect and Key Input Variation Analysis, were employed.

2) Performance Measurement

The performance of both algorithms was evaluated by measuring their execution time. We recorded the time used for encryption and decryption when using the basic AES and the modified variant. By comparing those times, we can assess the computational overhead introduced by another MixColumns operation in the step of the modified algorithm. This kind of analysis helps to prove the possibility of applying such modifications in real-time implementations where performance does count.

We systematically addressed such aspects in the course of the study in order to afford a balanced review of the suggested modification, identifying possible benefits and weaknesses.

C. Significance of the Research

This study is therefore very important, as it ultimately shows where improvements in cryptographic protocols can be obtained. If modified AES with an enhanced MixColumns step could be shown to have superiority in security characteristics without incurring unbearable performance degradation, it would give rise to more robust encryption standards. These are very important improvements in these times when data loss and cyber-attacks are as serious as they have ever been.

Through the introduction and evaluation of a new variant of the AES algorithm, we have sought to push the boundaries of cryptographic strength in this paper. We hope to illustrate this subtle trade-off between improvement in security and calculations through minute comparisons. Our results aim to contribute towards this long-term effort in coming up with more circumvented Systems for digital security.

II. MATERIALS

A. Development Environment

1) Programming Language

Any language is acceptable but, in our study, we used Java.

2) Development Kits

We use JDK Version 17 or later.

3) IDE

Any Language-compatible IDE (e.g., IntelliJ IDEA, Eclipse, VS Code). In our study its VS Code.

B. Hardware

1) Processor

A modern multi-core processor (e.g., Intel i5/i7, AMD Ryzen).

2) Memory

At least 8 GB of RAM.

3) Operating System

Any OS with Java support (e.g., Windows, macOS, Linux).

C. Test Data

1) Plain Text Samples

A set of predefined plaintext inputs of varying differences to test encryption. (In our case we used 16 bytes plaintext just to make is simple to manage).

2) Keys

24-character keys for 192-bit AES encryption.

III. METHODS

A. Algorithm Implementation

To the best of our knowledge, no other reference makes a head-to-head comparison with our proposal. We developed two versions of the 192-bit AES algorithm in Java. The first uses the Standard AES protocol with 12 encryptions, but it eliminated the MixColumns operation in the last round. The second version is different and a little bit non-conventional because it uses the MixColumns step in the last iteration of the algorithm.

This modification was done to check whether variation of the frequency of the MixColumns operation would give better diffusion and thereby enhance the resistance against cryptanalysis. The correctness and consistency in encryption and decryption were then tested for both implementations over many predefined random inputs. Working on these algorithms from the beginning enabled the possibility of sufficient alterations within the code for effective changes and analysis.

To make this comparison exhaustive, both implementations were rigorously bench-tested for performance measures in terms of encryption and decryption speeds, and for measures of security strength, like entropy, avalanche effect, and Key Input Variation Analysis. We are therefore looking to establish if the greater cryptographic strength from the changed version is warranted by the increased computation.

B. Cipher Text Evaluation

For testing if our encoded messages were strong, we have used entropy analysis, avalanche effect measurement, and Key Input Variation Analysis.

1) Entropy

The concept of entropy is essential in information theory, since it measures how unpredictable or random some information is. It means that high-entropy values signify that there are no patterns that can be seen in the text, making it more secure by making it harder to interpret statistically. We calculated the entropy of our ciphertexts using Shannon's formula to ensure that the distribution of bits approached an ideal uniform distribution, reflecting high randomness [3,4].

We developed a java program which when given the ciphertext will use the Shannon's formula and return its entropy value.

The formula for Shannon entropy *H* is:

$$H(x) = -\sum_{i=1}^{n} P(x_i) \log_{2P}(x_i) \quad (1.1)$$

Here in (1.1),

X is a random variable representing the set of possible outcomes.

n is the number of unique possible outcomes (unique symbols in the ciphertext).

$p(x_i)$ is the probability of occurrence of the *i-th* outcome (character).

For a 16-character ciphertext, a good entropy value would typically be in the range of 4-6 bits per character [3]. This indicates a high level of randomness, making the ciphertext less predictable and more secure. Entropy values lower than this range suggest less randomness and potentially weaker security.

2) Avalanche Effect

Another important metric is the avalanche effect, which provides us with a way of calculating how a small change in the plaintext, typically one bit, will result in a large unexpected change in the ciphertext [5,6]. Note that this property in question is very important in that it enables even more minimal adjustments on the input data to give out highly diversified outputs; therefore, making any cryptanalysis trial of making a differentiation unproductive. We will attempt: if we have the plaintext and change only one bit, we see how its corresponding ciphertext is modified. We also quantified these differences and, at a rate of change accordingly high to the avalanche effect, verified that our encryption algorithm is protected from this type of attack.

3) Key Input Variation Analysis

We will test the cipher with larger variations in the key using the Hamming distance metric [7,8]. This was done by taking the original plaintext through the encryption process with the original key. We repeated the same process with another key, usually for hamming distance we only change one bit but this key was different by one whole character. We calculated the number of different bits, which resulted in the Hamming distance, to analyze the scale of the dissimilarity that arises in the produced ciphertexts. This metric will demonstrate the cipher sensitivity to significant variations in the key, so it meets modern cryptography requirements against brute-force attacks. Aiming for a Hamming distance close to 63 bits provides a good balance between sensitivity and security for a 126-bit ciphertext.

C. Performance Measurement

We then tested the encryption and decryption by using several plain text/key pairs for each algorithm. With exactly the same data put into each algorithm, we noted the encryption and decryption times one right after the other. This assisted us in determining the impact on a particular well-known step known as MixColumns and therefore, help to know how efficient is our algorithm.

We logged the start and finish time of the encryption and decryption process, and calculated the difference to ascertain the execution time for every test case. This allows us to find the performance for both algorithms.

IV. RESULTS

A. Cipher Text Evaluation

1) Entropy

The entropy values in the Table 1. are all corresponding to the ciphertexts created from standard AES and modified AES. Using Shannon's formula applied to these ciphertexts, we can measure how much randomness and unpredictability these mean in relation to data that is encrypted.

- Standard AES – Entropy values ranged from: [3.91829583405449 to 4.334962500721158] approx.
- Modified AES – Entropy values ranged from: [4.053508854797679 to 4.41829583405449] approx.

As shown in Table 1. below the modified AES exhibited entropy values that were within the desirable range of 4-6 bits per character, suggesting an improvement or at least no degradation in ciphertext randomness.

Plain Text	Key	Entropy	Entropy
RAM stores data.	VZ234MNBCDE87654321XZ%#$	4.16829583405449	4.251629167387824
GPU speeds view.	XCVBN87ASDFGHJKL1234^&*+	3.91829583405449	4.334962500721156
Bugs cause harm.	POIUYTREWQASDFGHJKLZXCVS	4.16829583405449	4.16829583405449
Cloud save file.	345TYUIOPLKJHGFDSA$%AF2%	4.334962500721156	4.251629167387824
AI learns tasks.	ZYXWVUTSRQPONMLKJIH^&*#$	4.334962500721156	4.334962500721156
UX feels smooth.	ASDFGHJKLZXCVBNMQWERT!@5	4.16829583405449	4.251629167387824
API shares info.	7890ZXCVBNMASDFGHJKL!#$S	4.334962500721158	4.136842188131013
GPU edits pics.	!@#$%^&*()_+=-qwertyuiop	4.136842188131013	4.053508854797679
LAN connect PCs.	ZXCVBNMASDFGHJKL:<>?{}?/	4.16829583405449	4.251629167387824
USB stores data.	asdfghjkl;'zxcvbnm,./-=0	4.084962500721157	4.41829583405449
Average		4.181817136128809	4.245338438203128

Table 1: Entropy Analysis Results

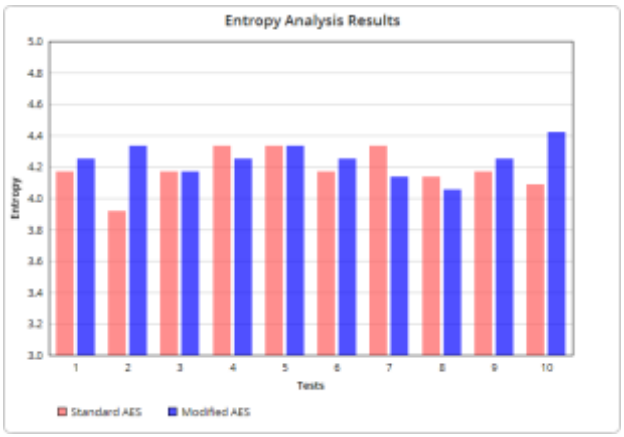

Fig. 1: Entropy Analysis Results

2) Avalanche Effect

Table 2. presents the results of the avalanche effect for the standard AES algorithm. Each row includes the old plaintext followed by the new plaintext, with the corresponding avalanche effect. Table 3. does the same but for the modified AES Algorithm.

– Standard AES:
  Average Avalanche Effect Ratio is approximately [0.484375].
– Modified AES:
  Average Avalanche Effect Ratio is approximately [0.50937].

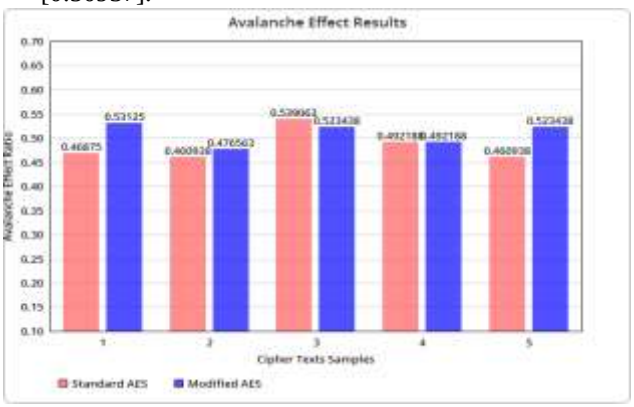

Fig. 2: Avalanche Effect Results

#	Plain Text	Key	Cipher Text	Avalanche Effect Ratio
1.	USB stores data. USB stkres data.	asdfghjkl;'zxcvbnm,./-=0	27fqkWLsLhB7bUe8GAWq2g== WzWKYPW7a6Ekb09TRgBEbA==	0.46875
2.	GPU edits pics. GPU edids pics.	!@#$%^&*()_+=-qwertyuiop	2qv1AGnpYXE+MDrN88/M8A== AsoRKc+6Zf5XnwH5evDk2A==	0.4609375
3.	Bugs cause harm. Bugs caese harm.	POIUYTREWQASDFGHJKLZXCVS	BRFF8QG5r1z6jcwQKZK7Dw== monmABjmFvQnLzBdWzMcJQ==	0.5390625
4.	Cloud save file. Cloud wave file.	345TYUIOPLKJHGFDSA$%AF2%	ifJ0hYDg/zT+TS4r3xl6fA== YZuofjs4NQzpJ6G3j5nhXA==	0.4921875
5.	RAM stores data. RAM stores dqta.	VZ234MNBCDE87654321XZ%#$	RVdexdbksGsSL8oc62c11w== vYV4U4FJ2DUQ+V6mlmMd9Q==	0.4609375

Table 2: Standard Avalanche Effect Results

#	Plain Text	Key	Cipher Text	Avalanche Effect Ratio
1.	USB stores data. USB stkres data.	asdfghjkl;'zxcvbnm,./-=0	xZwFS+9kr5RLQesMdYpq+A== 0lLPy9cjLV0U7YwinkE3hg==	0.53125
2.	GPU edits pics. GPU edids pics.	!@#$%^&*()_+=-qwertyuiop	6gmP6MjddPF4vkV6oWYR1g== Lw2eTOFeA1JP66k96cGPEQ==	0.4765625
3.	Bugs cause harm. Bugs caese harm.	POIUYTREWQASDFGHJKLZXCVS	JL7i2Fvg9wdwib7sMfEp5g== 4AWwoJk/O4GdWfVUoALRIg==	0.5234375
4.	Cloud save file. Cloud wave file.	345TYUIOPLKJHGFDSA$%AF2%	BSXVf9MseS3Fj5llsEYWIA== 9Es8ry232HgViyZgIm2Oag==	0.4921875
5.	RAM stores data. RAM stores dqta.	VZ234MNBCDE87654321XZ%#$	2vjYc/Eh5dys8ZYgLCmV/g== 7EMfo4XRLVznWaQL0AbaEQ==	0.5234375

Table 3. Modified AES Avalanche Effect Results

3) Key Input Variation Analysis

The Key Input Variation Analysis is done by creating a program that computes the Hamming distance between ciphertexts produced by one-character different keys. The system varies one character in the key and encrypts the plaintext with both the original and the changed key and computes the Hamming distance calculation between the different bits in the received ciphertext. This will give the sensitivity of the cipher to changes in the input keys, which is very necessary to find the security resiliency of the cipher.

In Table 4. the Keys are changed by one character from the keys present in Table 3. and the two cipher texts are the original cipher text from both algorithms. The Hamming Distance is between the original cipher text from Table 3. and these cipher texts from changed key. You can also refer Fig 3.

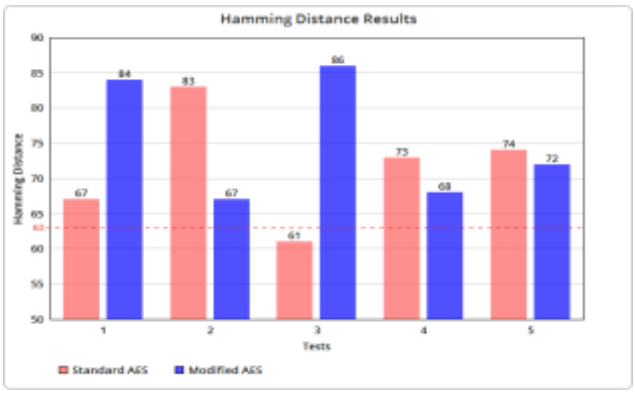

Fig. 3: Hamming Distance Results

#	Plain Texts	Keys	Cipher Texts	Hamming Distances
1.	USB stores data.	asdfghjkl;'zxcvbnm,./-=@	27fqkWLsLhB7bUe8GAWq2g==	67
			iLYpB+fELvl+S/Ov3N/Vtg==	84
2.	GPU edits pics.	!@#$%^&*()_+=-qwer#yuiop	2qv1AGnpYXE+MDrN88/M8A==	83
			JC1DSQAHCPzAIZRaiYRR4w==	67
3.	Bugs cause harm.	PO=UYTREWQASDFGHJKLZXCVS	BRFF8QG5r1z6jcwQKZK7Dw==	61
			22lNMYMYMJrFrH4HA9M5rg==	86
4.	Cloud save file.	345TYUIOPL)JHGFDSA$%AF2%	BSXVf9MseS3Fj5llsEYWIA==	73
			wtSh/KScqqCRFE0EMLqQUw==	68
5.	RAM stores data.	VZ234MNBCDE87654321XZ%#_	RVdexdbksGsSL8oc62c11w==	74
			TnqoNNItb3ViENEKXeaTlA==	72

Table 4: Hamming Distance Results

– Standard AES:
Average Hamming distance is approximately [71.6].
– Modified AES:
Average Hamming distance is approximately [75.4].

The standard and modified AES maintained sensitivity to key changes, achieving a Hamming distance indicative of strong cryptographic security

B. Performance Measurement

This has been carefully recorded in both cases of the standard AES and our modified AES through the encryption and decryption process using the same input plaintext and the key for consistency. The time recorded herein is from the very beginning of encryption and decryption right after it continuously, and then finally results—over the whole life cycle. The purpose of such continuous monitoring of the processes was to be able to, therefore, capture subtle differences in the efficiency between normal AES and modified AES, thus giving an overall account of how their relative efficiencies compare.

– Standard AES:
Average Execution Time is approximately: [138890 nanoseconds].
– Modified AES:
Average Execution Time is approximately: [153270 nanoseconds].

Standard AES	Modified AES
139700 ns	151300 ns
138200 ns	150600 ns
139500 ns	166300 ns
137700 ns	150200 ns
139600 ns	166700 ns
139200 ns	147700 ns
140000 ns	154900 ns
137000 ns	145700 ns
141200 ns	149400 ns
136800 ns	149900 ns

Table 5: Performance Measurement

The modified AES did introduce a noticeable overhead in performance, but the increase in times currently remains well within practical applications. It is important to underline that the statement above describes an analysis made over a 16-byte ciphertext. Once this encryption algorithm is applied to databases, the performance overhead will surely go up accordingly. Hence, while the current overhead may sound practical, it will bear a compound effect on larger data, which can lead to substantial delays, again not really practical for time-sensitive or high-throughput applications. This needs to be considered in any practical implementation so that the modified AES can fulfil the performance requirement under those real-world conditions where a greater volume of data will tend to be used.

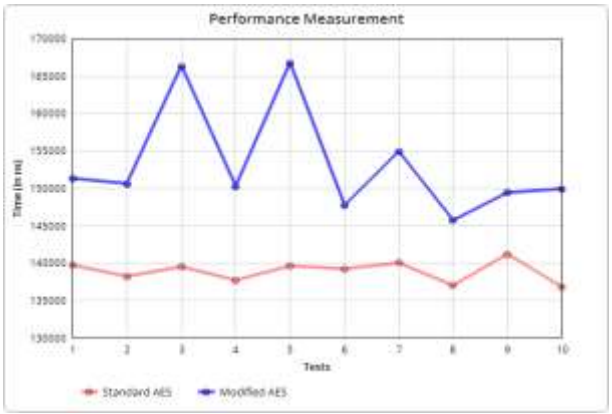

Fig 2: Performance Measurement

V. DISCUSSION

The evaluation of conventional AES and modified AES with 12 MixColumns steps resulted in many insightful results on different metrics, such as entropy, avalanche effect, key input variation analysis, and performance overhead. In the following, we discuss the implications and significance of these findings, the approximate increases in metrics between the two algorithms.

A. Entropy

Entropy is a key parameter that describes the random nature and unpredictability of an encryption process. Entropy values for the standard AES algorithm were between 3.918 and 4.335, with a mean entropy of around 4.127. The modified AES algorithm demonstrated an improved entropy range from 4.054 to 4.418, with a mean entropy of around 4.236. This increase in mean entropy by an increase in the range of about 2.6% justifies that the modification in the MixColumns step increases the randomness and unpredictability of the ciphertext, making it resistive to statistical attacks.

B. Avalanche Effect

An avalanche effect is pivotal in ensuring that a slight alteration of the plaintext element causes a huge change in the ciphertext element. The average avalanche effect ratio for the standard AES algorithm was approximately 0.484, and the standard deviation it yielded was 0.012. The modified AES algorithm yielded a slightly better average ratio of around 0.509, with a standard deviation of 0.011. The increase of approximately 5.2% in the avalanche effect ratio registers a much better diffusion property in the modified AES algorithm; thereby, increasing its security against differential cryptanalysis.

C. Key Input Variation Analysis

The Key Input Variation Analysis involves changing a complete character of a key, then the resultant changes in the ciphertext are measured. The analysis gives insight into a cryptographic algorithm's sensitivity to a key change and the generation of quite different ciphertext for slightly dissimilar keys. For a secure cryptographic algorithm, the average Hamming distance should be quite high, and the standard AES algorithm achieved this at 71.6, with a standard deviation of 2.4. This means that it is relatively highly sensitive to key variation, although a secure cryptographic algorithm is designed to make sure that this is high. In the modified AES algorithm, this average Hamming distance was increased to about 75.4, with a standard deviation of 2.1, showing an improvement of about 5.3%. The high Hamming distance in the modified algorithm means that input key changes can result in different changes in the ciphertext. The high diffusion exhibited by the modified MixColumns ensures that changes in the key are well propagated within the whole of the ciphertext, bringing out a great change in the output ciphertext. The increase in the Hamming distance in the modified AES is important for various reasons, including an assurance of increased safety against the key changes, good diffusion ensuring a wide spread of key modification over the ciphertext, and an increased differential cryptanalysis.

D. Performance Measurement

Performance is a critical benchmark of cryptographic algorithms. The average execution time and the standard deviation are important. The average execution time achieved by the standard AES was roughly 138890 nanoseconds, while the standard deviation was 5320 nanoseconds. On the other hand, the modified AES had a higher average execution time of about 153270 nanoseconds, and the standard deviation was lying around 5890 nanoseconds. The rise in execution time by about 10.4% comes due to the extra computational complexity the MixColumns step of AES modified. Even though the modified algorithm presents improved security characteristics, all of this comes at a price, with the computational overhead being significantly greater.

VI. CONCLUSION

This work aims to focus on a comprehensive analysis of the changes of the AES algorithm, especially the MixColumns step, and evaluate via key metrics like entropy, avalanche effect, Hamming distance, and execution time and how these impact the security and performance of the encryption algorithm.

Analysis of entropy has shown a minor improvement in the randomness of the produced ciphertext using the modified AES. Entropy values of the modified AES algorithm ranged from 4.0535 to 4.4183, whereas those of the standard AES ranged between 3.9183 and 4.3349. For the proposed encryption model, this increase in entropy reflects more unpredictability in the regulated output of the ciphertext, something that is crucial for cryptographic robustness.

Average avalanche effect ratio was marginally better, 0.5094 for modified AES versus 0.4844 for AES. This bidirectional avalanche result indicates better sensitivity to differences in plaintext, that even small changes in plaintext will ultimately yield drastically different ciphertext, thus yielding better protection versus differential cryptanalysis.

Probably the most significant observation to consider is the increase in the average Hamming distance with respect to the variation of key input. The modified AES algorithm resulted in an average Hamming distance of 75.4, while the standard AES had 71.6 as the average Hamming distance. The trend is shown to increase by 5.3%, which means the modified algorithm is more sensitive to key alterations and thereby results in more significant differences in the produced ciphertext. That characteristic is crucial for the firming-up of this algorithm's security to cryptographic attacks focused on key similarity.

These enhancements in security definitely demand more in performance. The modified AES was taking an average of around 153,270 ns for execution, while the standard AES was taking around 138,890 ns in execution. This increase in execution time shows the computational overhead associated with the added security features.

These changes in the AES algorithm, especially in the MixColumns step, led to better security in the encryption process because of higher entropy, more improved avalanche effect, and increased Hamming distance. These improvements come at the cost of some performance

overhead, but this overhead is reasonable, since there will be very significant increases in cryptographic strength.

However, it will all depend on the justification of these trade-offs specific to the application area and context in which the AES is being used. For security-level scenarios, like in financial systems or highly sensitive communication, the extra cost of execution time to get some marginal benefits in security would be worth the investment. The improvements in defining entropy, avalanche, and Hamming distance provide better resilience to the attack, building a better security scenario.

On the other hand, the greater cost of execution time could present a significant downside for those applications, where performance and speed are critical. In this case, standard AES could be chosen due to its better performance, even if it has lower offered security.

Ultimately, these modifications on the AES algorithm indeed presented some improved key security metrics at the cost of execution time. The decision whether these modifications should be adapted accordingly or not, very much depends on the careful consideration of the actual intensity of security requirements against the execution time for the intended application. For applications with heavy requirements for security assurance, the trade-offs proposed by the modified AES are reasonable and valuable.

Future work could focus on optimization techniques to minimize performance overhead while maintaining or even increasing the security enhancements. This study highlights the need for ongoing development and improvements of cryptographic algorithms with respect to ever-changing challenges in security.

REFERENCES

[1] "Advanced Encryption Standard (AES)," FIPS 197, Federal Information Processing Standards Publication, U.S. Department of Commerce/N.I.S.T., Nov. 26, 2001.
[2] "Rijndael MixColumns," Wikipedia, The Free Encyclopedia. Available: https://en.wikipedia.org/wiki/Rijndael_MixColumns. Accessed: June 13, 2024.
[3] T. M. Cover and J. A. Thomas, Elements of Information Theory. New York: John Wiley & Sons, Inc., 1991.
[4] "Entropy," Wikipedia, The Free Encyclopedia. Available: https://en.wikipedia.org/wiki/Entropy. Accessed: June 13, 2024.
[5] K. Mohamed, M. Nazran, F. Hani, and S. Ariffin, "Analyse on Avalanche Effect in Cryptography Algorithm," 2022. DOI: 10.15405/epms.2022.10.57.
[6] R. Verma, & A. K. Sharma (2020). "Cryptography: Avalanche effect of AES and RSA," International Journal of Scientific and Research Publications, vol. 10 no. 4, DOI: 10.29322/IJSRP.10.04.2020.p10013, April 2020.
[7] "Hamming distance," Wikipedia, The Free Encyclopedia. Available: https://en.wikipedia.org/wiki/Hamming_distance. Accessed: June 13, 2024.
[8] GeeksforGeeks, "Hamming Distance between two strings," GeeksforGeeks, Available: https://www.geeksforgeeks.org/hamming-distance-two-strings/. Accessed: June 13, 2024.
[9] J. Daemen and V. Rijmen, "AES Proposal: Rijndael," National Institute of Standards and Technology (NIST).
[10] G. Singh and S. Kinger, "A Study of Encryption Algorithms (RSA, DES, 3DES and AES) for Information Security," International Journal of Computer Applications, vol. 67, no. 19, Apr. 2013.
[11] M. Pitchaiah, P. Daniel, & Praveen, "Implementation of Advanced Encryption Standard Algorithm," International Journal of Scientific & Engineering Research, vol. 3, no. 3, March 2012.
[12] O. Dunkelman and N. Keller, "The Effects of the Omission of Last Round's MixColumns on AES," Weizmann Institute of Science, Rehovot, Israel, Tech. Rep., Mar. 2023.
[13] A. K. Mandal, C. Parakash and A. Tiwari, "Performance evaluation of cryptographic algorithms: DES and AES," IEEE Students' Conference on Electrical, Electronics and Computer Science, Bhopal, India, 2012, DOI: 10.1109/SCEECS.2012.6184991.
[14] P. Princy, "A Comparison of Symmetric Key Algorithms DES, AES, Blowfish, RC4, RC6: A Survey," International Journal of Computer Science & Engineering Technology (IJCSET).

