

Dynamic adaptive session-based advanced encryption standard

Jerico S. Baladhay, Marlon V. Gamido

Department of Information Technology, College of Computer Studies, Tarlac State University, Tarlac, Philippines

Article Info

Article history:

Received Jan 22, 2026

Revised Aug 24, 2026

Accepted Sep 4, 2026

Keywords:

Advanced encryption standard

Avalanche effect

Bit permutation

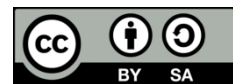
Dynamic round configuration

Symmetric key cryptography

ABSTRACT

Dynamic adaptive session-based advanced encryption standard (DASE) is an adaptive modification of the advanced encryption standard (AES) that allows the number of rounds to be adjusted to the security needs and timing constraints of each session. Conventional AES deployments run a fixed number of rounds regardless of data sensitivity, imposing needless processing on low-risk information and reducing flexibility in how security is applied. DASE defines three configurations: DASE-6, DASE-8, and DASE-10, and replaces the computationally heavy MixColumns transformation with a lightweight bit-permutation layer that preserves strong diffusion. Cryptographic robustness was evaluated using the avalanche effect and the strict avalanche criterion (SAC) across diverse plaintext patterns, while performance was measured on heterogeneous file types and sizes. DASE reached an average SAC of up to 53%, matching or exceeding standard AES-10, while DASE-6 delivered average encryption and decryption speed-ups of 50.55% and 38.94% over AES-10. DASE-8 and DASE-10 remained 40.46% and 32.54% faster in encryption, respectively, with DASE-10 retaining a clear advantage despite using the same round count as AES. The results demonstrate a tunable spectrum of security-performance trade-offs suitable for resource-constrained, high-throughput, and real-time data-protection scenarios.

This is an open access article under the [CC BY-SA](#) license.



Corresponding Author:

Jerico S. Baladhay

Department of Information Technology, College of Computer Studies, Tarlac State University

Tarlac, Philippines

Email: jsbaladhay@tsu.edu.ph

1. INTRODUCTION

Advanced encryption standard (AES) maintains its proficiency as the predominant algorithm in symmetric key cryptography since its standardization by the National Institute of Standards and Technology (NIST) in 2001 [1]. The AES algorithm has 3 variants consisting of 128 bits, 192 bits, and 256 bits which continues to validate its theoretical security and proven resistance to standard linear analysis attacks and differential attacks [2], [3]. AES remains cryptographically secure after more than 20 years and became an institution for its excellence where impossible differential attacks require 2^{66} chosen plaintext and integral attacks demand 2^{88} computational steps [4], [5] concluding that no practical attacks have broken full AES [6].

Recent progress in lightweight cryptography has led academics to explore alterations to AES that achieve a balance between security and computational efficiency, especially in resource-limited internet of things (IoT) contexts [7], [8]. Numerous studies have investigated reduced-round implementations and alternate transformation architectures to mitigate the performance constraints associated with conventional AES [9], [10]. Lightweight cryptographic algorithms like SIMON, SPECK, and ASCON have shown promise for embedded systems, but AES is still the gold standard for security testing since it has a long history of cryptanalysis [11], [12].

AES remains the predominant choice for data encryption, though it presents certain computational limitations. The MixColumns transformation, for instance, is based on a structure that uses a lot of math and makes the computer work harder while it is running because it is the subsequent complex operation in AES, following the S-Box. This hardware operation consumes substantial resources and induces prolonged critical path delays, a concern frequently raised by very large scale integration (VLSI) designers. MixColumns works by multiplying numbers in the Galois field (GF)(2^8), which is based on polynomial arithmetic and requires special hardware to work well [13]. Among all AES transformations, only SubBytes rivals its resource demands. In practice, MixColumns draws heavily on available hardware and extends critical path delays, making it a bottleneck in VLSI-based implementations [14].

The typical AES has a rigid round structure that limits the number of rounds (10, 12, or 14) to what the key length is and can't be changed while the program is running [15]. Computing power in modern IoT systems can be very different, from powerful gateway devices to nodes with very minimal resources. This creates a problem: the set number of rounds is too many for strong devices but not enough for weak ones. The main schedule approach was revised in a scientific study, and the number of rounds was lowered from 10 to 6. This means that the new AES is still safe and can quickly encrypt and decrypt files [16].

This paper presents the dynamic adaptive session-based advanced encryption standard (DASE), a model that makes the traditional AES algorithm more flexible by adding 6, 8, and 10 rounds. This makes it much faster and safer. The latest version of AES makes a substantial change to the MixColumns transformation by adding a new bit permutation layer that uses XOR-based byte interdependencies and lookup table transformations [17], [18]. It adds session-based parameter adaptation, which lets it change the number of encryption rounds while the program is running without having to recompile the algorithm [19], [20].

This work makes three main contributions. First, the design and implementation of a bit permutation layer that takes the place of MixColumns while keeping the diffusion qualities by cascading exclusive OR (XOR) dependencies over state bytes. Second, a session-based options interface that lets users change the number of rounds, enabling security-performance trade-offs suited to a given deployment. Third, a full experimental comparison of DASE with regular AES on several measures, such as encryption throughput, memory use, and cryptographic strength indicators [21]. The suggested technique is still compatible with the usual S-Box, ShiftRows transformation, and key expansion mechanism used in AES.

Unlike prior reduced-round AES schemes that simply lower the round count and risk weaker diffusion [22], or hardware-oriented MixColumns optimizations that still rely on GF arithmetic [23], DASE introduces three jointly novel elements. First, a bit-permutation diffusion layer that replaces MixColumns entirely while preserving avalanche behavior through cascading XOR byte-interdependencies. Second, runtime session-based round selection (6, 8, or 10 rounds) that requires no recompilation of the algorithm. Third, an empirically validated security-performance trade-off measured across heterogeneous file workloads. Whereas earlier permutation-based reduced-round work established a single fixed configuration [18], DASE generalizes this into an adaptive, configurable framework and benchmarks it against standard AES on both diffusion (avalanche and SAC) and throughput.

2. METHOD

This section discusses the construction and experimental testing of the DASE. The techniques section contains comprehensive details on replicating the study, enabling other researchers to reproduce it. The proposed technique implements three significant modifications to the standard AES; i) it dynamically alters the round configuration, ii) it adjusts session-dependent parameters, and iii) it introduces a novel bit permutation layer to supplant the MixColumns transformation.

2.1. Algorithm design overview

The dynamic adaptive session-based AES technique uses a 128-bit key length and works with 128-bit data blocks, just like standard AES-128. The modified method has a session-based dynamic round selection which has three options: DASE-6, DASE-8, and DASE-10. The options correspond to 6, 8, or 10 encryption rounds, respectively. MixColumns operation has been replaced with a simple permutation method that keeps the diffusion features while lowering the cost of computing. Figure 1 shows DASE. DASE is based on standard AES-128 but has three differences; i) dynamic round configuration, ii) session-based parameters, and iii) bit permutation layer.

2.2. Dynamic round configuration

The dynamic round configuration mechanism enables adaptive selection of encryption rounds based on session requirements. Table 1 presents the three DASE variants compared against standard AES-128.

Although DASE-10 has the same number of rounds as standard AES-128, it is more efficient and secure because it replaces the resource-intensive MixColumns transformation with the optimized bit permutation layer. This reduces GF arithmetic overhead while keeping the diffusion properties.

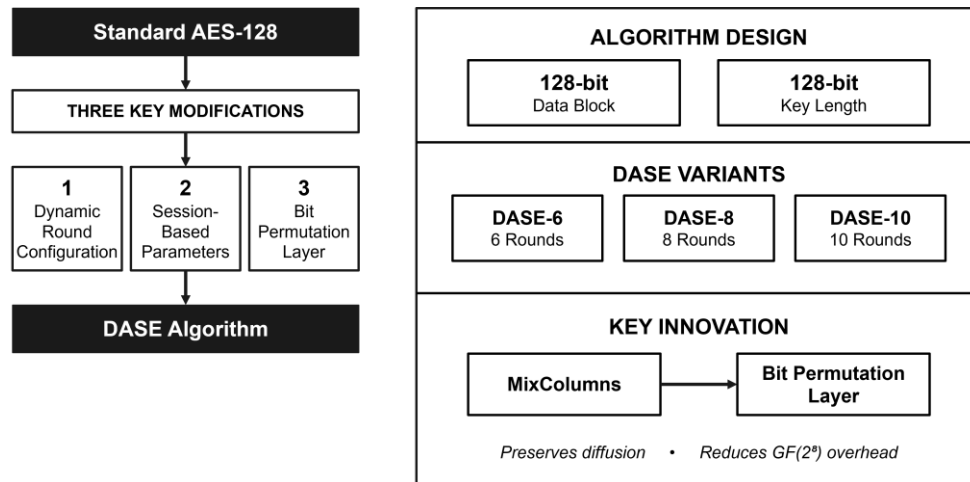


Figure 1. DASE framework

Table 1. Default and configurable round parameters

Variant	Rounds	Standard AES-128
DASE-6	6	10 rounds
DASE-8	8	10 rounds
DASE-10	10	10 rounds

The dynamic round configuration approach of the DASE algorithm is shown in Figure 2. The first step in the process is to figure out what level of security is needed for the session. This starts the round selection process, where the system chooses one of three possibilities that can be changed: DASE-6 ($r=6$), DASE-8 ($r=8$), or DASE-10 ($r=10$). The key scheduler uses the formula $(r+1) \times 4$ to figure out how many round keys to make. It can get 28, 36, or 44 round keys this way. It can modify the number of encryption rounds while the application is running with this adaptive method. It doesn't need to recompile the algorithm. This allows the algorithm to find the best balance between security and computing speed for each deployment. The key scheduler makes a certain number of expanded keys based on the round count. For a round count of r , the key scheduler makes $(r+1) \times 4$ round keys, each of which is made up of 32-bit words.

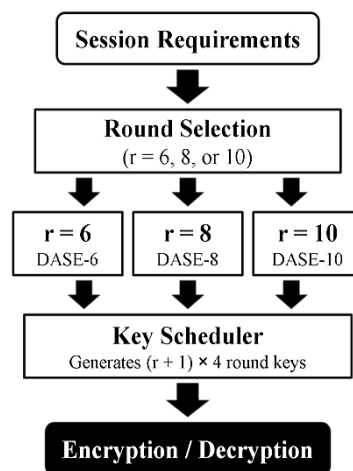


Figure 2. Dynamic round configuration mechanism

2.3. Session-based parameter adaptation

The session-based adaptation approach lets the cryptographic settings change at runtime without having to recompile the algorithm. The session options include the following parameters: operation mode, custom round count, and initialization vector specification.

Algorithm 1. DASE session initialization

```

Input: Key K[0..n-1], IV[0..15], Options O
Output: Initialized DASE cipher instance
1: keyLength <- length(K) x 8
2: if O.rounds != null then
3:   rounds <- O.rounds
4: else
5:   rounds <- DefaultRounds[keyLength]
6: end if
7: roundKeys <- KeyScheduler.expand(K, rounds)
8: mode <- O.mode OR CBC
9: initVector <- BlockTo32Ints(IV)
10: return CipherInstance(roundKeys, mode, initVector)

```

The initialization procedure first determines the key length in bits by multiplying the key array length by 8. If a custom round count is specified in the session options, it overrides the default value. The key scheduler then expands the master key into the required number of round keys.

2.4. Bit permutation layer design

The most significant modification in DASE is the replacement of the MixColumns transformation with a novel bit permutation layer. This layer achieves diffusion through XOR-based byte interdependencies and lookup table transformations. In (1a)–(1d) formalize the bit permutation process for a single state word.

The bit permutation layer works on four 32-bit state words at the same time. The transition happens like this for each word: First, individual bytes are taken and sent through the S-Box substitution. Second, the outputs of the S-Box from nearby bytes are XORed together to make dependencies between bytes. Third, the results of the XOR are utilized to look up values in four different permutation tables: BP1, BP2, BP3, and BP4. Finally, the outputs of the permutations are put together to make the new state word. In (1) makes this change official:

$$b0' = BP1[S[s0,0] \text{ XOR } S[s1,1]] \quad (1a)$$

$$b1' = BP2[S[s1,1] \text{ XOR } S[s2,2]] \quad (1b)$$

$$b2' = BP3[S[s2,2] \text{ XOR } S[s3,3]] \quad (1c)$$

$$b3' = BP4[S[s3,3] \text{ XOR } b0'] \quad (1d)$$

where $S[x]$ denotes the S-Box lookup, s_i, j represents the byte at row i and column j of the state matrix, and $BPn[x]$ denotes the n -th permutation table lookup. In (1d) produces a feedback dependency, which means that the fourth byte depends on the first byte that has already been calculated. This creates a cascading relationship inside each column.

2.5. Encryption process

Algorithm 2 shows the whole process of block encryption. The encryption process comprises three steps: an initial AddRoundKey operation, several primary rounds with bit permutation, and a final round without bit permutation.

Algorithm 2. DASE block encryption

```

Input: State S[0..3], RoundKeys RK[0..4r+3]
Output: Encrypted state S'[0..3]
1: rounds <- (length(RK) / 4) - 1
2: keyIndex <- 0
3: for i <- 0 to rounds do
4:   if i != 0 AND i != rounds then
5:     S <- SubBytes(S)
6:     S <- ShiftRows(S)
7:     S <- BitPermutation(S)
8:   else if i = rounds then
9:     S <- SubBytes(S)
10:    S <- ShiftRows(S)

```

```

11:     end if
12:     S <- AddRoundKey(S, RK[keyIndex..keyIndex+3])
13:     keyIndex <- keyIndex + 4
14: end for
15: return S

```

2.6. Decryption process

The decryption procedure uses the opposite processes in the opposite order. The inverse bit permutation operation is placed after the AddRoundKey operation, which is a key difference from normal AES. Algorithm 3 explains how to decrypt.

Algorithm 3. DASE block decryption

```

Input: Ciphertext C[0..3], RoundKeys RK[0..4r+3]
Output: Plaintext P[0..3]
1: rounds <- (length(RK) / 4) - 1
2: keyIndex <- length(RK) - 1
3: S <- C
4: for i <- rounds downto 0 do
5:   if i != rounds then
6:     S <- InvShiftRows(S)
7:     S <- InvSubBytes(S)
8:   end if
9:   S <- AddRoundKey(S, RK[keyIndex-3..keyIndex])
10:  keyIndex <- keyIndex - 4
11:  if i != 0 AND i != rounds then
12:    S <- InvBitPermutation(S)
13:  end if
14: end for
15: return S

```

2.7. Inverse bit permutation

The inverse bit permutation procedure uses four matching inverse tables (I_BP1, I_BP2, I_BP3, I_BP4). The inverse operation processes bytes in reverse order, starting with the least significant byte and rebuilding the original values through cascading XOR operations. In (2a)-(2d) formalize this process:

$$r3 = I_BP4[s3] \text{ XOR } (s0 \gg 24) \quad (2a)$$

$$r2 = I_BP3[s2] \text{ XOR } r3 \quad (2b)$$

$$r1 = I_BP2[s1] \text{ XOR } r2 \quad (2c)$$

$$r0 = I_BP1[s0] \text{ XOR } r1 \quad (2d)$$

2.8. Data representation and processing

For better performance, the system processes data in 32-bit words. Using big-endian byte ordering, byte arrays that are input are turned into 32-bit integers. Algorithm 4 shows how the cipher changes bytes into words.

Algorithm 4. Byte array to 32-bit word conversion

```

Input: Byte array B[0..15]
Output: Word array W[0..3]
1: for j <- 0 to 3 do
2:   i <- j x 4
3:   W[j] <- (B[i] << 24) | (B[i+1] << 16) |
             (B[i+2] << 8) | B[i+3]
4: end for
5: return W

```

2.9. Comparison with standard advanced encryption standard

Table 2 shows how DASE and the traditional AES algorithm are different in terms of structure, including the most important changes made in this work. Figure 3 indicates that the suggested DASE algorithm is built differently than standard AES-128. Both systems start with 128-bit plaintext blocks and execute an AddRoundKey operation. The main variation is in the basic round structure: standard AES-128 uses the MixColumns transformation with GF multiplication over GF(2⁸). DASE, on the other hand, uses a lightweight XOR-based Bit Permutation layer. DASE specifies how many rounds are needed (6, 8, or 10), whereas AES-128 always has 10 rounds. This allows a balance between security and performance while preserving the crucial parts of diffusion that make cryptography robust.

Table 2. DASE vs standard AES comparison

Feature	Standard AES	DASE
Round count	Fixed (10/12/14)	Configurable
Diffusion layer	MixColumns (GF multiply)	Bit Permutation (XOR+LUT)
SubBytes	Standard S-Box	Standard S-Box
ShiftRows	Standard rotation	Standard rotation
Parameter adaptation	None	Session-based options
Lookup tables	8 (S-Box, Inv S-Box, and 6 GF-mult)	10 (S-Boxes+8 BP)

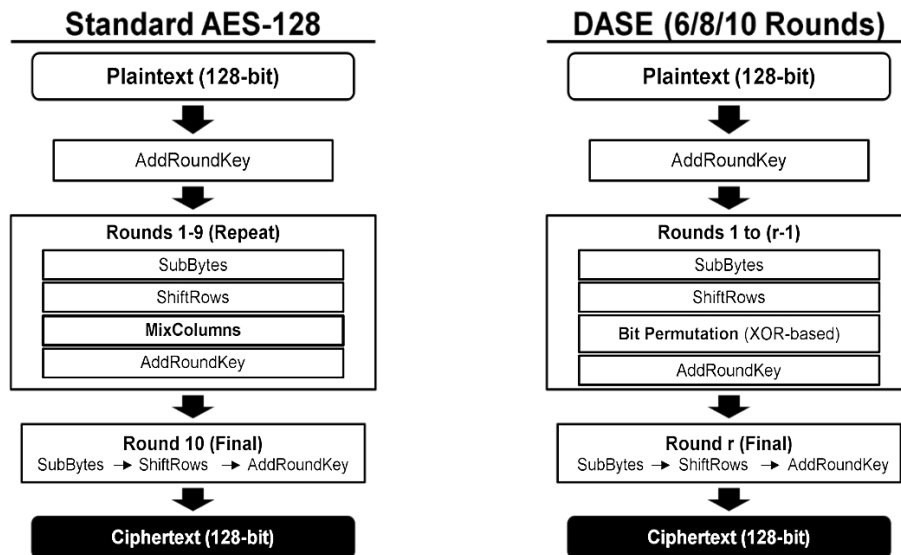


Figure 3. DASE vs standard AES: round structure comparison

3. RESULTS AND DISCUSSION

This part shows the results of the tests of the DASE algorithm. The analysis looks at three main things: measuring the avalanche effect to see how strong the encryption is, performance benchmarks for different file kinds and sizes, and a comparison of throughput to normal AES-10. Each subsection has the test findings and what they mean for putting them into practice.

3.1. Avalanche effect analysis and cryptographic properties

The empirical evaluation substantiates the resilience of the suggested DASE algorithm versions by extensive avalanche effect testing [24]. Table 3 (see Appendix) shows the avalanche effect percentages, which show that all of the algorithms have strong encryption. It also shows how sensitive encryption algorithms are to small changes in the input. An ideal cipher should produce about a 50%-bit difference in the output when a single bit is changed in the plaintext. The results below show that all versions of DASE have significant avalanche properties that are similar to AES. The change percentages range from 46.09% to 59.38% throughout the 10 test scenarios.

In summary of Table 3, DASE-6 and DASE-8 achieved higher diffusion rates than the conventional AES, with DASE-8 remarkably recording the highest score of 59.38% in Test 8. While in Table 4, it summarizes the avalanche effect performance and strict avalanche criterion (SAC) values for AES and DASE variants across ten test cases. The results reveal that the variants of DASE achieved higher SAC values than conventional AES, with DASE-8 obtaining the highest score of 53.436%, followed by DASE-6 at 53.124%, and DASE-10 at 51.797%, while the standard AES recorded 46.875%. Hence, these experiments show that DASE demonstrates dominant bit diffusion properties, suggesting enhanced resistance to differential cryptanalysis compared to standard AES implementation.

The SAC values reveal a consistent ordering in which all DASE variants meet or exceed the ~50% ideal, whereas standard AES averages 46.9%. DASE-8 and DASE-6 record the highest SAC (53.4% and 53.1%), indicating that the bit-permutation layer diffuses single-bit input changes at least as effectively as MixColumns. This directly supports the study's central claim: removing MixColumns does not weaken diffusion, so the observed speed gains do not come at a security cost.

Figure 4 demonstrates the extent to which AES and DASE variations work in ten different test cases with the avalanche effect. The graph clearly reveals that traditional AES is below the 50% optimum level in

most of the 10 situations, with big drops at Tests 1, 6, and 8. DASE versions 6, 8, and 10, on the other hand, show more consistent performance around or over the 50% mark. DASE-8 and DASE-6, on the other hand, show the largest peaks, which reach about 58-59%. The picture shows that the modified AES variants spread bits better and more consistently than regular AES. This shows that the algorithm works well, which is important for cryptographic strength.

Table 4. Avalanche effect and SAC performance comparison

Test	AES	DASE 6	DASE 8	DASE 10
1	43.75	53.91	52.34	46.09
2	49.22	50.00	57.81	49.22
3	47.66	52.34	49.22	50.78
4	50.00	52.34	57.03	54.69
5	47.66	58.59	52.34	56.25
6	42.19	47.66	53.12	48.44
7	46.09	53.12	48.44	50.00
8	44.53	53.91	59.38	57.03
9	52.34	53.12	53.12	58.59
10	45.31	56.25	51.56	46.88
SAC	46.875	53.124	53.436	51.797

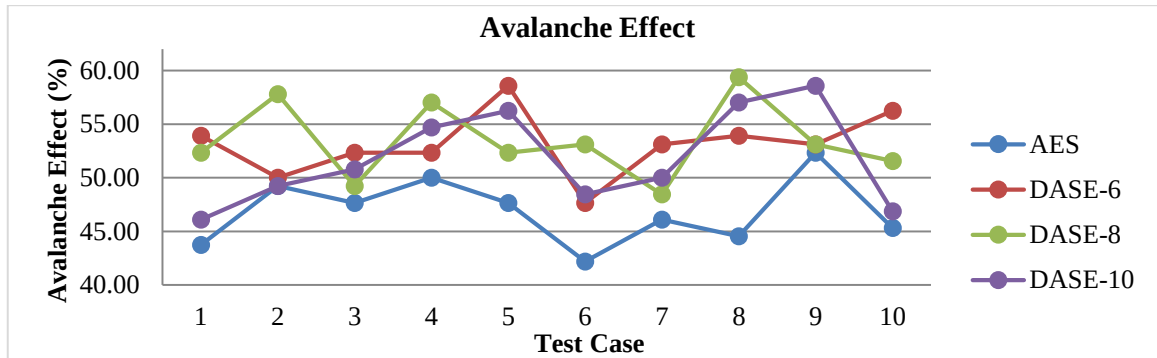


Figure 4. Avalanche effect

3.2. Performance evaluation across different file types

Table 5 presents the performance of AES-10, DASE-6, DASE-8, and DASE-10 across different file types and sizes ranging from 1 MB to 100 MB. All DASE variants achieved faster execution times than conventional AES-10 across every file type and size. For example, DASE-6 encrypted a 10 MB video in 460.24 ms compared with 743.67 ms for AES-10, and this advantage widened at larger sizes: a 100 MB video encrypted in 3008.56 ms under DASE-6 versus 7067.57 ms under AES-10. These results confirm that DASE provides a consistently more computationally efficient alternative to conventional AES.

Table 5. Different file types performance evaluation

File type	Size (MB)	Operation	DASE-6	DASE-8	DASE-10	AES-10
Text (.txt)	1	Encryption	50.4222	57.4772	63.9838	78.2337
		Decryption	53.4305	54.7693	69.0801	99.4686
Image (.jpg)	2	Encryption	101.4814	115.8498	126.7881	159.7921
		Decryption	93.2487	121.2332	136.4272	188.2861
Audio (.mp3)	3	Encryption	157.5899	172.3695	191.0092	248.1053
		Decryption	160.9684	186.0531	205.8545	272.319
Video (.mp4)	5	Encryption	275.9116	306.7992	341.3981	424.8326
		Decryption	286.8655	326.6545	357.5792	506.7722
Video (.mp4)	10	Encryption	460.2429	514.9953	580.2566	743.6709
		Decryption	486.539	559.3267	633.1643	887.7925
Audio (.wav)	28	Encryption	949.5834	1126.4770	1141.7602	1893.7544
		Decryption	1702.4514	2177.8239	2127.4475	2952.3102
Audio (.wav)	38	Encryption	1301.9033	1481.6308	1739.3387	2440.4515
		Decryption	2329.9366	2732.2257	2657.4729	3822.7428
Video (.mp4)	50	Encryption	2123.7787	2504.4017	2920.2516	3990.2164
		Decryption	3603.8397	4563.3233	4691.5986	6342.0189
Video (.mp4)	100	Encryption	3008.5560	3869.6902	4394.9853	7067.5665
		Decryption	6417.2161	6926.1958	7609.7335	9715.4257

Across all file types, execution time scales with file size while the DASE variants maintain a stable and, at larger sizes, widening proportional advantage over AES-10. The consistency of this gap across text, image, audio, and video files confirms that the improvement stems from the algorithm's reduced per-block cost rather than any file-type-specific effect.

3.2.1. Comparative performance analysis

Table 6 summarizes the comparative performance of the DASE variants (6, 8, and 10 rounds) and AES-10 in encrypting and decrypting files of varying sizes. Across all tests, DASE-6 consistently demonstrates the fastest execution, followed by DASE-8 and DASE-10, while AES-10 remains the slowest. Aggregated across all file sizes, DASE-6 outperforms AES-10 by 50.55% in encryption and 38.94% in decryption; DASE-8 achieves gains of 40.46% and 28.80%; and DASE-10 records improvements of 32.54% and 25.41%, respectively. The encryption advantage becomes more pronounced as file size increases, indicating that DASE's reduced per-block cost and lightweight diffusion layer scale favorably with larger workloads.

Table 6. Comparative performance analysis of DASE variants and AES-10 (execution time in milliseconds)

Test	Encryption				Decryption			
	DASE-6	DASE-8	DASE-10	AES-10	DASE-6	DASE-8	DASE-10	AES-10
1	50.4222	57.4772	63.9838	78.2337	53.4305	54.7693	69.0801	99.4686
2	101.4814	115.8498	126.7881	159.7921	93.2487	121.2332	136.4272	188.2861
3	157.5899	172.3695	191.0092	248.1053	160.9684	186.0531	205.8545	272.319
4	275.9116	306.7992	341.3981	424.8326	286.8655	326.6545	357.5792	506.7722
5	460.2429	514.9953	580.2566	743.6709	486.539	559.3267	633.1643	887.7925
6	949.5834	1126.4770	1141.7602	1893.7544	1702.4514	2177.8239	2127.4475	2952.3102
7	1301.9033	1481.6308	1739.3387	2440.4515	2329.9366	2732.2257	2657.4729	3822.7428
8	2123.7787	2504.4017	2920.2516	3990.2164	3603.8397	4563.3233	4691.5986	6342.0189
9	3008.5560	3869.6902	4394.9853	7067.5665	6417.2161	6926.1958	7609.7335	9715.4257
%	50.5505	40.4592	32.5393		38.9421	28.8034	25.4115	

3.2.2. Hardware implementation considerations

The throughput analysis in Figure 5 shows that all DASE variants significantly outperform the standard AES-10 implementation. DASE-6 achieves the highest throughput, reaching 23.76 MB/s for encryption and 17.67 MB/s for decryption, corresponding to improvements of 78.77% and 75.84% over AES-10, respectively. These gains are attributable to the reduced number of rounds and the use of the efficient bit-permutation layer in place of the computationally intensive MixColumns transformation. Notably, even with the same number of rounds, DASE-10 surpasses AES-10 by 38.11% in encryption and 37.96% in decryption, demonstrating the effectiveness of the proposed diffusion mechanism. Overall, the DASE variants provide consistently higher encryption and decryption throughput than conventional AES.

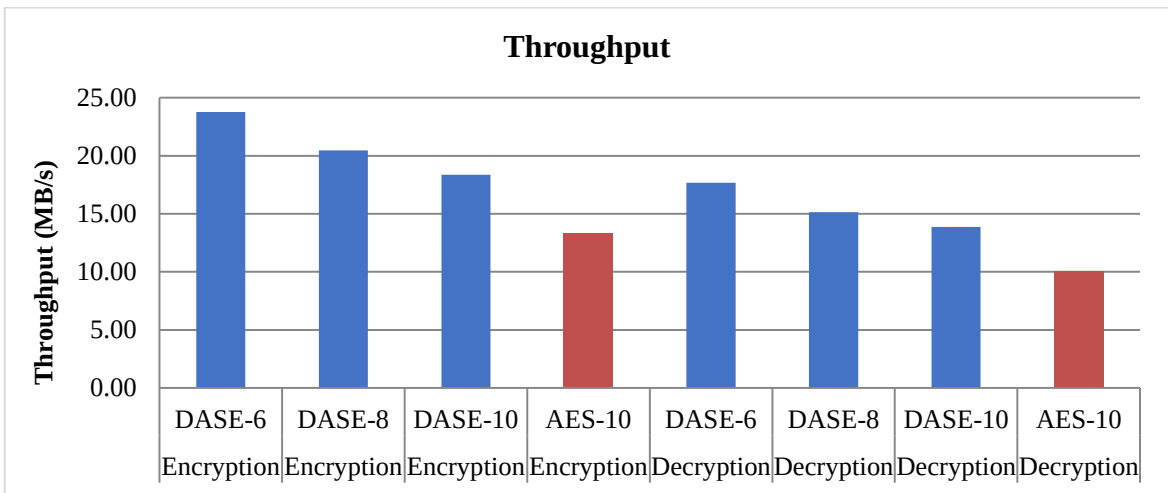


Figure 5. Throughput performance

3.3. Comparison with other lightweight ciphers

To position DASE within the broader lightweight-cryptography landscape, its design is contrasted with representative lightweight ciphers evaluated in recent literature. Comprehensive assessments of substitution-permutation designs such as PRESENT and ARX-based families such as SIMON and SPECK report small hardware and software footprints for constrained devices, but these ciphers adopt reduced block sizes and are not AES-compatible [25]. More recent clean-slate proposals, such as the GFRX generalized-Feistel cipher, continue this direction toward resource-constrained IoT nodes [26], while ASCON, the NIST lightweight-cryptography standard, targets authenticated encryption on constrained devices [27]. Unlike these clean-slate designs, DASE preserves full AES-128 compatibility, retaining the standard S-Box, ShiftRows, and key schedule, and improves efficiency by replacing MixColumns with a bit-permutation layer and allowing configurable rounds. DASE is therefore best understood as a drop-in-oriented enhancement of an established standard rather than a new primitive requiring separate security validation.

A direct empirical benchmark of DASE against these ciphers under an identical implementation and hardware harness is beyond the software scope of this study and is identified as future work. The controlled comparison against standard AES-128 reported here provides a like-for-like baseline that isolates the effect of the proposed modifications.

3.4. Computational complexity analysis

For a message of n 128-bit blocks encrypted with r rounds, both AES and DASE exhibit a time complexity of $O(n \cdot r)$, since each round applies a constant number of operations to the fixed 128-bit state. The practical difference lies in two constants. First, the configurable round count allows r to be reduced to 6 or 8, linearly lowering the total work relative to the fixed $r=10$ of standard AES. Second, DASE replaces the MixColumns transformation, which performs $GF(2^8)$ multiplications, with a bit-permutation layer composed of S-Box substitutions, XOR operations, and table lookups. Substituting multiplication-heavy field arithmetic with lookup-and-XOR operations reduces the per-round constant, which is consistent with the measured throughput gains reported in Table 6 and Figure 5.

In terms of processing overhead, the dominant per-round cost of standard AES is shared between the SubBytes and MixColumns transformations. By eliminating MixColumns while retaining its diffusion effect, verified empirically by the avalanche and SAC results in Tables 3 and 4, DASE removes one of the two principal per-round bottlenecks. The remaining operations, namely SubBytes, ShiftRows, and AddRoundKey, are identical to those of standard AES, so no additional per-round overhead is introduced beyond the lightweight permutation layer.

Regarding memory consumption, DASE trades a small increase in static storage for its computational savings. The reference implementation uses ten byte-wise lookup tables, namely the S-Box, the inverse S-Box, and eight bit-permutation tables (four forward and four inverse), compared with eight tables for the corresponding AES implementation, namely the S-Box, the inverse S-Box, and six Galois-Field multiplication tables. Assuming 256-byte tables, this corresponds to approximately 2.5 KB for DASE against 2.0 KB for AES, an increase of only about 0.5 KB, while the working memory of the cipher state remains unchanged at 128 bits. This modest storage overhead is acceptable for typical IoT-class devices and constitutes the principal cost of the efficiency gains.

3.5. Security analysis

DASE preserves the 128-bit key length and the AES key-expansion schedule; only the round count and the diffusion layer are modified. The effective key space therefore remains 2^{128} , for which exhaustive key search is computationally infeasible with current and foreseeable technology. All three variants, DASE-6, DASE-8, and DASE-10, share this key space and thus the same resistance to brute-force attacks.

Resistance to differential and linear cryptanalysis requires strong diffusion, which is a necessary condition for mitigating these attacks. The avalanche results (approximately 46% to 59%) and the SAC values (up to 53.4%) in Tables 3 and 4 show that a single-bit change in the input propagates across the output at least as thoroughly as in standard AES, providing empirical evidence of strong diffusion. It should be noted that MixColumns is a maximum-distance-separable transformation with an optimal branch number of five, which gives standard AES a provable lower bound on the number of active S-Boxes. The bit-permutation layer achieves diffusion through cascading XOR dependencies rather than an MDS matrix; therefore, a formal branch-number and differential/linear bound for the proposed layer is identified as an important direction for future work. The present results establish practical diffusion strength but are not a substitute for provable bounds.

Because DASE reuses the AES key schedule, it inherits the established resistance of AES to key-schedule and related-key weaknesses at full rounds. The configurable-round design also makes the security margin explicit: the reduced-round variants, particularly DASE-6, trade a portion of this margin for speed and are therefore intended for lower-sensitivity sessions, whereas DASE-10 retains the full ten-round

structure for high-sensitivity data. A formal cryptanalytic evaluation of the reduced-round variants against state-of-the-art reduced-round AES attack techniques is recommended prior to deployment in high-assurance settings.

3.6. Practical deployment considerations

The evaluation in this study is software-based, so the reported execution times and throughput characterize the algorithmic efficiency of DASE relative to standard AES on a general-purpose platform rather than absolute performance on a specific embedded device. Because DASE retains the standard AES-128 structure and modifies only the round count and the diffusion layer, it can be deployed as a drop-in software replacement within existing AES-based pipelines without changes to surrounding protocols. In practical terms, the session-based round selection allows a deployment to match the security level to both the sensitivity of the data and the capability of the device: resource-constrained or high-throughput endpoints may adopt DASE-6, whereas high-assurance channels can retain DASE-10. The consistent, size-independent performance advantage observed across text, image, audio, and video files suggests that these benefits will carry over to representative real-world workloads such as IoT gateway traffic and bulk file protection. Absolute performance, energy consumption, and memory behavior on constrained hardware, such as microcontrollers or FPGA targets, depend on platform-specific factors and are identified as future work.

4. CONCLUSION

This study set out to reduce the computational cost of AES on resource-constrained devices without weakening its cryptographic strength. To that end, DASE introduced three modifications to AES-128: a dynamic round configuration (6, 8, or 10 rounds), session-based parameter adaptation that adjusts security at runtime without recompilation, and a bit-permutation layer replacing the resource-heavy MixColumns operation. Experimentally, all DASE variants matched or exceeded standard AES in diffusion, with DASE-8 and DASE-6 reaching average SAC values of 53.4% and 53.1% against 46.9% for AES-10; in performance, DASE-6 encrypted and decrypted up to 50.55% and 38.94% faster than AES-10, and DASE-10 retained a 38.11% throughput advantage despite an identical round count. Several limitations remain: the evaluation is software-based and single-key-length (128-bit), avalanche and SAC establish diffusion but not formal resistance to differential, linear, or side-channel attacks, and testing used files up to 100 MB. Practically, the tunable security-performance spectrum makes DASE well suited to IoT and real-time applications where fixed-round AES is either wasteful or too heavy. Future work will pursue hardware field-programmable gate array (FPGA) implementation, formal cryptanalytic and side-channel evaluation, and benchmarking against other lightweight ciphers.

FUNDING INFORMATION

Publication funding by Tarlac State University.

AUTHOR CONTRIBUTIONS STATEMENT

This journal uses the Contributor Roles Taxonomy (CRediT) to recognize individual author contributions, reduce authorship disputes, and facilitate collaboration

Name of Author	C	M	So	Va	Fo	I	R	D	O	E	Vi	Su	P	Fu
Jerico S. Baladhay	✓	✓	✓	✓	✓	✓		✓	✓	✓	✓	✓		✓
Marlon V. Gamido	✓	✓		✓	✓	✓	✓		✓	✓	✓	✓	✓	✓

C : **C**onceptualization

M : **M**ethodology

So : **S**oftware

Va : **V**alidation

Fo : **F**ormal analysis

I : **I**nterpretation

R : **R**esources

D : **D**ata Curation

O : **O**rganizing - **O**riginal Draft

E : **E**ditorial - **R**eview & **E**dit

Vi : **V**isualization

Su : **S**upervision

P : **P**roject administration

Fu : **F**unding acquisition

CONFLICT OF INTEREST STATEMENT

Authors state no conflict of interest.

DATA AVAILABILITY

The data that support the findings of this study are available from the corresponding author, [JSB], upon reasonable request.

REFERENCES

- [1] T. Kovats, N. Rameshan, K. G. Karunaratne, I. Giannopoulos, and A. Sebastian, "In-memory encryption using the advanced encryption standard," *Philos. Trans. R. Soc. A Math. Phys. Eng. Sci.*, vol. 383, no. 2288, Jan. 2025, doi: 10.1098/rsta.2023.0396.
- [2] M. Liu, C. Hou, X. Lu, S. Wang, and D. Lin, "Differential-Linear Cryptanalysis from an Algebraic Perspective," *J. Cryptol.*, vol. 39, no. 1, p. 7, Nov. 2025, doi: 10.1007/s00145-025-09564-3.
- [3] D. Pal, M. R. Ali, A. Das, and D. R. Chowdhury, "A cluster-based practical key recovery attack on reduced-round AES using impossible-differential cryptanalysis," *J. Supercomput.*, vol. 79, no. 6, pp. 6252–6289, Apr. 2023, doi: 10.1007/s11227-022-04872-y.
- [4] G. Leurent and C. Pernot, "New Representations of the AES Key Schedule," *J. Cryptol.*, vol. 38, no. 1, p. 2, Jan. 2025, doi: 10.1007/s00145-024-09522-5.
- [5] O. Dunkelmann, N. Keller, E. Ronen, and A. Shamir, "The Retracing Boomerang Attack, with Application to Reduced-Round AES," *J. Cryptol.*, vol. 37, no. 3, p. 32, Jul. 2024, doi: 10.1007/s00145-024-09512-7.
- [6] Amrita, C. P. Ekwueme, I. H. Adam, and A. Dwivedi, "Lightweight Cryptography for Internet of Things: A Review," *EAI Endorsed Trans. Internet Things*, vol. 10, Mar. 2024, doi: 10.4108/eetiot.5565.
- [7] F. Thabit, O. Can, A. O. Aljahdali, G. H. Al-Gaphari, and H. A. Alkhzaimi, "Cryptography Algorithms for Enhancing IoT Security," *Internet of Things (Netherlands)*, vol. 22, p. 100759, Jul. 2023, doi: 10.1016/j.iot.2023.100759.
- [8] M. El-hajj, H. Mousawi, and A. Fadlallah, "Analysis of Lightweight Cryptographic Algorithms on IoT Hardware Platform," *Futur. Internet*, vol. 15, no. 2, p. 54, Jan. 2023, doi: 10.3390/fi15020054.
- [9] I. Radhakrishnan, S. Jadon, and P. B. Honnavalli, "Efficiency and Security Evaluation of Lightweight Cryptographic Algorithms for Resource-Constrained IoT Devices," *Sensors*, vol. 24, no. 12, p. 4008, Jun. 2024, doi: 10.3390/s24124008.
- [10] M. Rana, Q. Mamun, and R. Islam, "Enhancing IoT Security: An Innovative Key Management System for Lightweight Block Ciphers," *Sensors*, vol. 23, no. 18, p. 7678, Sep. 2023, doi: 10.3390/s23187678.
- [11] V. A. Thakor, M. A. Razzaque, and M. R. A. Khandaker, "Lightweight Cryptography Algorithms for Resource-Constrained IoT Devices: A Review, Comparison and Research Opportunities," *IEEE Access*, vol. 9, pp. 28177–28193, 2021, doi: 10.1109/ACCESS.2021.3052867.
- [12] R. Parthasarathy, P. Saravanan, S. R. Srivatsav, and C. M. Manisha, "Lightweight implementation of AES for resource constrained environment," *Analog Integr. Circuits Signal Process.*, vol. 123, no. 1, p. 10, Apr. 2025, doi: 10.1007/s10470-025-02343-x.
- [13] K. Kumar, K. R. Ramkumar, and A. Kaur, "A lightweight AES algorithm implementation for encrypting voice messages using field programmable gate arrays," *J. King Saud Univ. - Comput. Inf. Sci.*, vol. 34, no. 6, pp. 3878–3885, Jun. 2022, doi: 10.1016/j.jksuci.2020.08.005.
- [14] A. Malal and C. Tezcan, "First Fully Pipelined High Throughput FPGA Implementation and GPU Optimization of Wider Variant of AES," *J. Cryptogr. Eng.*, vol. 16, no. 1, p. 1, Jan. 2026, doi: 10.1007/s13389-025-00388-2.
- [15] C. Tezcan, "Optimization of Advanced Encryption Standard on Graphics Processing Units," *IEEE Access*, vol. 9, pp. 67315–67326, 2021, doi: 10.1109/ACCESS.2021.3077551.
- [16] C. Mu, "Application of optimizing advanced encryption standard encryption algorithm in secure communication of vehicle controller area network bus," *Front. Mech. Eng.*, vol. 10, Jul. 2024, doi: 10.3389/fmech.2024.1407665.
- [17] N. Ibrahim and J. Agbinya, "Design of a Lightweight Cryptographic Scheme for Resource-Constrained Internet of Things Devices," *Appl. Sci.*, vol. 13, no. 7, p. 4398, Mar. 2023, doi: 10.3390/app13074398.
- [18] J. S. Baladhay and E. M. De L. Reyes, "AES-128 reduced-round permutation by replacing the MixColumns function," *Indones. J. Electr. Eng. Comput. Sci.*, vol. 33, no. 3, pp. 1641–1652, 2024, doi: 10.11591/ijeecs.v33.i3.pp1641-1652.
- [19] A. Sevin and A. A. O. Mohammed, "A survey on software implementation of lightweight block ciphers for IoT devices," *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 3, pp. 1801–1815, Mar. 2023, doi: 10.1007/s12652-021-03395-3.
- [20] P. R. Kumar and S. Goel, "A secure and efficient encryption system based on adaptive and machine learning for securing data in fog computing," *Sci. Rep.*, vol. 15, no. 1, p. 11654, Apr. 2025, doi: 10.1038/s41598-025-92245-9.
- [21] B. Sarkar, A. Saha, D. Dutta, G. De Sarkar, and K. Karmakar, "A Survey on the Advanced Encryption Standard (AES): A Pillar of Modern Cryptography," *Int. J. Comput. Sci. Mob. Comput.*, vol. 13, no. 4, pp. 68–87, Apr. 2024, doi: 10.47760/ijcsmc.2024.v13i04.008.
- [22] M. Qasimeh, R. S. Al-Qassas, and M. Ababneh, "Software design and experimental evaluation of a reduced aes for iot applications," *Futur. Internet*, vol. 13, no. 11, p. 273, Oct. 2021, doi: 10.3390/fi13110273.
- [23] O. Azzouzi, M. Anane, M. C. Ghanem, Y. Himeur, and H. Kheddar, "Efficient Fine-Grained LuT-Based Optimization of AES MixColumns and InvMixColumns for FPGA Implementation," *Electron.*, vol. 14, no. 24, p. 4912, Dec. 2025, doi: 10.3390/electronics14244912.
- [24] D. Upadhyay, N. Gaikwad, M. Zaman, and S. Sampalli, "Investigating the Avalanche Effect of Various Cryptographically Secure Hash Functions and Hash-Based Applications," *IEEE Access*, vol. 10, pp. 112472–112486, 2022, doi: 10.1109/ACCESS.2022.3215778.
- [25] N. G. Zinabu, Y. W. Marye, K. K. Tune, and S. A. Demilew, "Comprehensive Analysis of Lightweight Cryptographic Algorithms for Battery-Limited Internet of Things Devices," *Int. J. Distrib. Sens. Networks*, vol. 2025, no. 1, Jan. 2025, doi: 10.1155/dsn/9639728.
- [26] X. Zhang, S. Tang, T. Li, X. Li, and C. Wang, "GFRX: A New Lightweight Block Cipher for Resource-Constrained IoT Nodes," *Electron.*, vol. 12, no. 2, p. 405, Jan. 2023, doi: 10.3390/electronics12020405.
- [27] J. Kaur, A. C. Canto, M. M. Kermani, and R. Azarderakhsh, "A Survey on the Implementations, Attacks, and Countermeasures of the NIST Lightweight Cryptography Standard: ASCON," *ACM Comput. Surv.*, vol. 58, no. 1, pp. 1–16, Jan. 2025, doi: 10.1145/3744640.

APPENDIX

Table 3. Detailed avalanche effect test results with cipher outputs

Test	Plain text (original vs. flipped bit)	Algorithm	Cipher output 1	Cipher output 2	%
1	2222222222222222 2222222222222222	AES	40fa8572eeb2e44cd2cb9806ad95a8f7	f4f90ded7a22be1962a0f5b797bbe8a6	43.75
		DASE 6	5cfc850bcc3ed7f0b3b93b12f2e05ae2	4e63643c6986b693a9a7c58b4d78b324	53.91
		DASE 8	7a02fc0e72fe8d5f99c16a95f754b357	e2d2826934c8a8dd380a36479ea97d38	52.34
		DASE 10	1fed45bc3360b99bf3be4cc3d597f765	1611b1ff98e018fb87a9e8ffeedb61b6	46.09
2	0000111100001111 0000111100001111	AES	255e8251dd6d047d1b40d217f8700e10	8f451583c878f85994b310fd9ed43655	49.22
		DASE 6	7a74c692aebee6e0829fee7471da9829	acce0b75dc298beede034ea738c992b5	50.00
		DASE 8	e71ef9548d1681383426104d95c9be61	38442d00417952748d686df39317e926	57.81
		DASE 10	d2357662bb477c0de7685d99ddecc12e	dca6bccfca051f5a533b36bf9e54b9df	49.22
3	A5A5A5A5A5A5A5A5 A5A5A5A5A5A5A5A5	AES	808204e8966ac4d23cf6ea425ed05bae	ad84394a18e2646c78233dba0d564c83	47.66
		DASE 6	e239e9b6ffc6255ecfb0fbaec751ef4d	c222da8ad7fab21315cf4dd9d8e5b735	52.34
		DASE 8	de5464ce45dd1b942d644d4ac664c845	f3b0bea34dc6f499b9e5d8882fb5b8ae	49.22
		DASE 10	b8c66ad0617230d7b99910f280c8b2fc	522107511103628a8f3abff52a8a9142	50.78
4	0000000ffffffffff 0003000ffffffffff	AES	de7dd0b5043d0b77fd176c893e97fddb	2869f0e2139cce7c40be5f423048505f	50.00
		DASE 6	a03a494cf1f1e63c2d438b7ee02011aa	9ebdb2bc03513d1cb2d1a5bc91dc d78c	52.34
		DASE 8	5bc086c5eeae869a3969916a20ab1541	d53989153e3d61d6bd840b95563ef2fb	57.03
		DASE 10	9c4b678b7f372cfbe3ae6486cca12e68	65823af68e90da564bbbfdead01ba f5c	54.69
5	1a1b1c1d1e1f1g1h 1a1b1c1d1e1f1g1h	AES	bdd9f0f5759c9205208da7b3e3e9f35c	67ca0f0dc9bf72e5358e56e1f2d671c4	47.66
		DASE 6	be46a26cdea1eeba6c96dff4fea5a650	a86dc574b44f1a999945e32e51d261be	58.59
		DASE 8	a35c88fffdde0fde22461ce0a921ef19	84fa7bdb4beac1f36e5fd80b59e71764	52.34
		DASE 10	f772598fc99865ad9e605647da586552	e98c2d7ff16ba8c6130d8ff09d73666b	56.25
6	0123456789ABCDEF A123456789ABCDEF	AES	83e3a17a29c9fb7ed25b5971a2c60455	61ab2c1906cdc22563efdf9638b0418	42.19
		DASE 6	7556c73179d73b75b67745cd1cd058f3	79ecd14c2b052426cc1485dd0a573d05	47.66
		DASE 8	0eba85b70ffa71354b08d97a0a6c2996	0c0c2af9ccd79def12cbeb19dc5b7f64	53.12
		DASE 10	0e843c5f185b3409a73e697b140c7297	fa7f2ba92a7c6805d5bc2ee8f4edf ace	48.44
7	F0E1D2C3B4A59687 F0E1D2C3B4A59587	AES	a9f3e13b280dcb14d49d48b3dda007cd	58b14141a24f4f4aa1471aaaaffeff54e	46.09
		DASE 6	fd9c77de032122b38ae680b9b9d972bd	3bb254af5be50f5cc90f4b0213d10d64	53.12
		DASE 8	1194e3c5954f53bbe82670bcf6c026f3	45a4450d5965bc4b66ac00843862985e	48.44
		DASE 10	6573c825d8678b972b38abd6f07915ee	2bfe241e22f09d3e6a713bf703ca9094	50.00
8	abcd123456789012 abcd123456789012	AES	5802cf3411c75b8e73c8b0e4186c89fe	9dcb96a87babd70583efac979a54d876	44.53
		DASE 6	33bed6831c7c56866d4e52b9f4688579	08ce329f607b631eb391e928bb5a0e12	53.91
		DASE 8	188dfa899be1c5c594244e84566129e8	f8dbafcf6f51cfe657e65fc7eaa8e5f03	59.38
		DASE 10	59bb7271ed5b4fcde67371502b422f55	169d8fdf520d08fc84b96e2f6774c273	57.03

Table 3. Detailed avalanche effect test results with cipher outputs (*continued*)

Test	Plain text (original vs. flipped bit)	Algorithm	Cipher output 1	Cipher output 2	%
9	h 1g2f3e4d5c6b7a8 g 1g2f3e4d5c6b7a8	AES	171dfd82de79b04dbec5bf5c32f6b88f	b561e6eb0f17a6ab875c099863cd6bd1	52.34
		DASE 6	7179b4603b1448372a6150a9c73c6733	a4e51815952655922e16a76eaf7b15f9	53.12
		DASE 8	c6e41bff004d0d4ee35d97e6a590a0a2	f831ee7769433dba089e69dabb89038c	53.12
		DASE 10	17b24834b08664350d7733054cbb9686	e0edbd4d0c2773be63857f9b319c95ed	58.59
10	rstuvwxy z abcde g rstuvwxy z abcde g	AES	7a7abf446f4ff242b2cb80e914a4f623	6a1c43aefdd56012d78811ad818283da	45.31
		DASE 6	ebfa3e620a79b0a60c4ed1f76ac59e5b	64116bde69848a2095f73485e0933728	56.25
		DASE 8	f6e148fa4874b38bad3779928c776a08	031ec9a84e82eb5ccac31bae8038ab12	51.56
		DASE 10	609e15262a61a9ec6057f74b71539c74	a0b106e867d666d9c59cb6882b434c72	46.88

BIOGRAPHIES OF AUTHORS



Jerico S. Baladhay is an Instructor I at the College of Computer Studies, Tarlac State University (TSU), Tarlac City, Philippines. He brings expertise in information technology, software development, and cybersecurity research. His academic career reflects a strong commitment to both instruction and practical application of technology solutions in the fields of algorithm development, data security, and system performance optimization. His research interests lie at the crucial intersection of algorithms, cybersecurity, and digital system performance optimization. His work focuses on cryptographic algorithm modification, specifically exploring innovative approaches to optimize symmetric key encryption without compromising security integrity. He can be contacted at email: jsbaladhay@tsu.edu.ph.



Dr. Marlon V. Gamido has more than 30 years of academic experience and is an associate professor at Tarlac State University, Philippines. He is currently serving as Vice President for Administration at Tarlac State University, and previously served as Vice President for Administration and Finance (2020–2022), Dean of the College of Computer Studies (2014–2020), and Director of Management Information Systems (2005–2014; 2016–2019; 2023–present). He represented the Philippines in the United Nations Asian and Pacific Training Center for ICT for Development (UN APCICT/ESCAP). He is also a licensed electrical engineer and an active member of the Tarlac ICT Council. His research expertise is in the fields of ICT4D, enterprise architecture, IT project management, and e-governance. He is currently taking his DIT major in E-Governance. He can be contacted at email: mvgamido@tsu.edu.ph.