Abdulrahman Mustapha, 2026, 14:1
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology
An Open Access Journal

Hybrid Encryption Algorithms for Computer Resources Optimasation

Abdulrahman Mustapha[1], Aminu Aliyu Abdullahi[2], Farouk Lawan Gambo[3], Abubakar Muhammad Miyim[4], Muhammad Aminu[5]

[1,2,3,4]Computer Science Department, Faculty of Computing, Federal University Dutse – Nigeria
[5]Computer Science Department, Faculty of Computing, Bayero University Kano – Nigeria

Abstract- In today's digital era, ensuring the confidentiality, integrity, verifying users and entities, and maintaining the secrecy of sensitive information of transmission are crucial; both in terms of speed and memory use is a growing challenge, especially for resource-limited devices like IoT, mobile, and embedded systems. The thesis presents an entropy-aware Key Encapsulation Mechanism (KEM) and Data Encapsulation Mechanism (DEM) hybrid scheme. The technical design pairs AES-256 (CBC mode) for bulk data confidentiality with ECC (secp256r1) for secure key encapsulation, using SHA 256 for integrity verification where needed. Performance benchmarking was conducted on an HP laptop (Intel i7 7200U) using distinct datasets: Audio and Image partitioned in 10% increments to analyse allocation effects. The research provides an empirically validated framework that offers specific engineering guidelines for optimising memory and latency in secure data transmission. The inclusion of entropy adaptation combined with NIST test vectors (Known Answer Tests) to validate the correctness of the hybrid implementation, distinguishing it from prior works that often lacked validation. These innovations enhance cryptographic performance for IoT and edge computing environments where both resource efficiency and strong security are imperative. The study recognises limitations in hardware diversity, sector-specific applicability, and exclusion of PQC showing directions for future studies.

Keywords: Hybrid, Cryptography, PQC, AES, ECC, Latency, RAM, Optimisation, Memory consumption, Benchmarking, IoT, Encryption, and KAT.

I. INTRODUCTION

In our modern digital age, cryptography has become an essential cybersecurity tool for protecting sensitive information from hackers and other cybercriminals (Subedar, Zuhi, and Araballi, Ashwini, 2020). The word, 'Crypto-graphy' derived from the Greek word 'kryptos' meaning hidden or secret, cryptography literally translates to 'hidden writing.' It can be used to obscure any form of digital communication, including text, images, video or audio. In practice, cryptography is mainly used to transform messages into an unreadable/incomprehensible format (known as ciphertext/cyphertext) that can only be decrypted into a readable format (known as plain text) by the authorised intended recipient by using a specific secret key.

Cryptography ensures confidentiality by encrypting sent messages using an algorithm with a key only known to the sender and recipient. A common example of this is the messaging tool WhatsApp, which encrypts conversations between people to ensure they cannot be hacked or intercepted (Quora Contributors, 2025). The art of cryptography has been used to code messages for thousands of years and continues to be used in bank cards, password protection and e-commerce also secures browsing, such as with virtual private networks (VPNs), which use encrypted tunnels, asymmetric encryption, and public and private shared keys. The complementary natures of AES and ECC make them ideal candidates for a hybrid cryptographic model. AES ensures speedy data encryption and decryption, while ECC manages secure key generation and exchange (Subedar, Z., and Araballi, A. 2020) Hybrid models that merge these algorithms have been used to achieve a balance between high security and resource optimisation. According to Subedar, Z., and Araballi, A. (2020:9)

"Such hybrid cryptographic schemes can simultaneously achieve authentication,

confidentiality, and integrity, satisfying multiple security goals more efficiently than standalone algorithms".

Empirical Gap and Problem Statement

Prior work has proposed hybrid AES–ECC combinations, but many studies either present qualitative claims about memory or focus solely on throughput. For instance, Subedar and Araballi (2020) focused heavily on throughput and qualitative claims about security but relied on MATLAB simulations rather than real-world system processing, which fails to capture actual memory residency (RAM) costs. Similarly, while Popoola et al. (2024) successfully minimised latency for IoT healthcare systems, their study was limited to specific sensor data streams and did not evaluate performance across diverse multimedia formats like high-resolution images or large audio files. Furthermore, studies by Buhari et al. (2025) provided excellent benchmarks for symmetric algorithms (AES, Blowfish) but excluded the overhead introduced by asymmetric key encapsulation in a hybrid environment. Consequently, a clear empirical gap exists: there is a lack of recent studies that report both transaction latency and peak memory usage simultaneously across diverse multimedia datasets (Audio and Image) on standard hardware. This thesis addresses this gap by implementing an entropy-aware hybrid scheme and providing a granular, per-dataset analysis of the trade-offs between memory footprint and encryption speed.

The thesis explicitly identifies a deficiency in existing literature regarding the simultaneous reporting of latency and memory profiles on actual hardware.

Critique of Prior Work

Previous studies have proposed AES–ECC combinations, but they often present only qualitative claims regarding memory or focus exclusively on throughput. For instance, Subedar and Araballi (2020) focused heavily on throughput and qualitative claims about security but relied on MATLAB simulations rather than real-world system processing, which fails to capture actual memory residency (RAM) costs. Similarly, while Popoola et al. (2024) successfully minimised latency for IoT healthcare systems, their study was limited to specific sensor data streams and did not evaluate performance across diverse multimedia formats like high-resolution images or large audio files. Furthermore, studies by Buhari et al. (2025) provided excellent benchmarks for symmetric algorithms (AES, Blowfish) but excluded the overhead introduced by asymmetric key encapsulation in a hybrid environment. Consequently, a clear empirical gap exists:

Specific Gap

There is a distinct lack of recent studies reporting both transaction latency and peak memory usage across diverse multimedia datasets (Audio and Image) on standard PC hardware.

Addressing The Gap

This research fills that empirical void by implementing the hybrid entropy model and reporting per-dataset latency and memory metrics on a common PC platform (Intel i7-7200U, 8GB RAM).

Conceptualisation

In cybersecurity, cryptography uses mathematical algorithms (mathematical formulas) to provide security for digital communications. Cryptography changes data into an encrypted form, accessible to a lone key holder or any other authorised party. The National Institute of Standards and Technology (NIST) describes cryptography as the "discipline that embodies the principles, means, and methods for the transformation of data". Although in the past cryptography referred only to the encryption and decryption of messages using secret keys, today it is defined as involving three distinct mechanisms: symmetric-key encipherment, asymmetric-key encipherment, and hashing (NIST, 2024).

Cryptography relies on two fundamental concepts: plain text and cyphertexts. Plain text refers to the actual message, whereas cipher text represents the encrypted version of the communication. Finally, the cipher is decrypted to reveal the original message. Encryption is the process of converting readable data into unreadable data. While, decryption is completely reversed of the encryption, so, Plaintext

is the data before it gets encrypted. Otherwise, the ciphertext is the data after getting encrypted.

The key term is the keyword that the sender and the receiver know and use to encrypt and decrypt the data. Keys are like passwords, but they are virtually impossible to decipher without tons of computational resources and decryption experience.

Goals of Cryptography

Sullivan, (2024) classified goals of cryptography into categories. By using encryption technology, data confidentiality can be ensured across the networks.

- **Integrity:** this confirm the authenticity of a message, it can also prove the integrity of the information being sent and received. For example, digital signatures can detect forgery or tampering in software distribution and financial transactions.
- Authenticity process of detecting tampering and verifying origin of data over network.
- **Non-repudiation:** It is for accountability and responsibility from the sender of a message. Digital signatures are examples of this, in email nonrepudiation.
- **Confidentiality:** preventing unauthorised disclosure, the most common aspect of information security (Data Breach Investigations, (2023). A common example of this is the messaging tool WhatsApp, which encrypts conversations between people to ensure they cannot be hacked or intercepted (Hussien, F. A., & Khairi, T. W. A. 2023). Also secures browsing, such as with VPNs, which use encrypted tunnels, asymmetric encryption, and public and private shared keys.
- **Availability:** Information is useless if it is not available. Information needs to be constantly changed, which means it must be accessible to authorised entities. Imagine what would happen to a bank if the customers could not access their accounts for transactions.
- **Service reliability and availability:** Technology should ensure that users receive the expected service quality by, considering any potential modifiers that may affect service availability owing to intruders.

Advanced Encryption Standard

AES also known by its original name Rijndael (Dutch pronunciation: ['rɛinda:l]) Daemen and Rijmen, (2002) defined it as is a specification for the encryption of electronic data established by the U.S. National Institute of Standards and Technology (NIST) in 2001 (Daemen, Joan; Rijmen, Vincent 2002) Daemen, Joan; Rijmen, Vincent. AES has been adopted by the U.S. government. AES was announced as a Federal Information Processing Standard (FIPS) for symmetric key encryption in the United States in 2001. It supersedes the Data Encryption Standard (DES), which was published in 1977. (Westlund, Harold B., 2002). as cited in Thesis of Kwame (2024). AES is a block cipher operating on fixed-size blocks of data and supporting three different key sizes. A possible size is 128, 192, or 256 bits. AES implementations can be optimised further via parallelisation and platform-specific instruction sets (e.g., AES-NI), but those optimisations are often un-avail on low-end IoT hardware, which motivates hybrid approaches for key management to asymmetric algorithms.

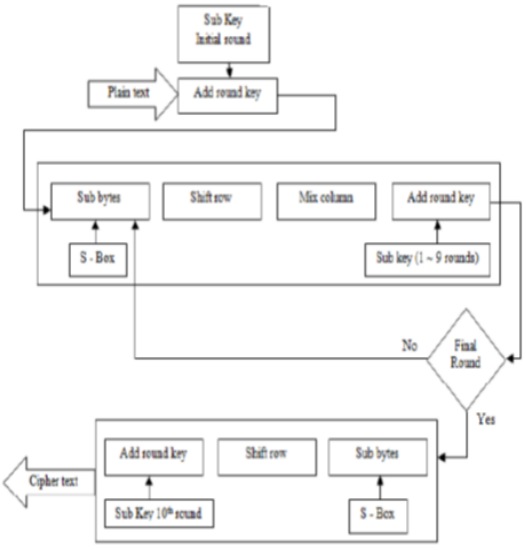


Figure 1: AES Encryption process

Elliptic Curve Cryptography

ECC provides the asymmetric side of modern secure systems by offering strong key agreement and signature mechanisms with much smaller key sizes than RSA, savings in storage, transmission and, in

some contexts, memory (Elliptic Curve Cryptography, 2022). Nevertheless, Mathematically ECC use elliptic curve algebra, on the curve, which dominates CPU time and transient memory for point representations and temporary variables which involves point addition, scalar multiplication, and modular arithmetic. ECC's benefits for constrained devices are compelling: equivalent security with shorter keys reduces bandwidth and persistent storage needs. (Ayush et al, 2024)

Properties Of Elliptic Curve

- The elliptic curve is symmetric along the x-axis.
- If we draw a line through the curve, it will touch a maximum of 3 points..

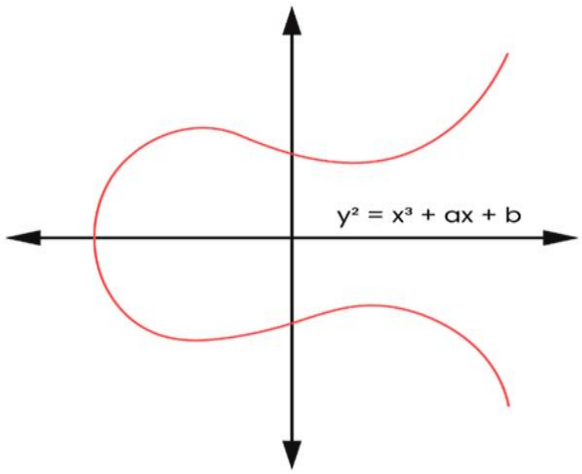


Figure 2: Elliptic Curve Sketch

Secure Hashing Algorithms

SHA-256 Hashing SHA-256 refers to a family of secure hashing algorithms for the generation and verification of the hash value for data digests integrity checks and KDFs. Before encryption, data is passed through SHA-256, and after decryption, it compares hash extensions to indicate that the data merely existed in its unaltered condition. In this work SHA256 is used to derive symmetric keys from the ECC shared secret; integrity of ciphertexts is provided by AES rather than bare hash checks whenever possible. Therefore, the inclusion of SHA-256 in the hybrid scheme guarantees the complete integrity and verification of transactions Ayush, et -al, 2024).

II. RELATED WORKS

- **Latency Optimisation in Cryptographic Algorithms**

Empirical studies consistently show AES is superior for bulk throughput, while public-key schemes (ECC, RSA) incur higher latency for key operations. Memory usage depends on implementation choices: AES can require S-boxes and round key tables; ECC implementations can allocate transient memory for point operations and precomputations. Several works explore hardware acceleration, parallelism, and algorithmic tuning to improve latency (Mochurada & Shchura, 2023). However, ECC's smaller key sizes reduce memory overhead, a key factor motivating its combination with AES. Studies by Khan and Abbas (2025), and Roy et al., (2024) discuss that hybrid AES-ECC systems can mitigate asymmetric latency through hardware acceleration and protocol optimisation, maintaining practical performance in cloud and embedded systems. Popoola et al. (2024) benchmarked a hybrid AES-128/ECC-256r1 system achieving remarkably low latency (approximately 0.006 seconds), demonstrating that careful design can reconcile ECC's latency costs within hybrid constructs. However, comparative studies that report both latency and RAM across diverse datasets remain limited.

Memory Consumption and Optimisation

Several studies on memory optimisation techniques relevant to AES-ECC hybrids. For instance, Happer, (2025) surveyed lightweight cryptographic solutions for IoT, discussing trade-offs between latency, memory consumption, and power usage, for hybrid systems, his findings targeting resource-constrained devices.

Hybridisation: Combining Strengths And Addressing Challenges

Hybrid cryptosystems seek to unite AES's speed advantages with ECC's key management efficiency to overcome the limitations of purely symmetric or asymmetric algorithms Subedar, and Araballi,. (2020) study's find out that, hybrid AES-ECC combinations outperform standalone cryptosystems in throughput and latency metrics.

In the same light, Sasikumar and Nagarajan (2024) further noted hybrid asymmetric-symmetric cryptography as balancing security and efficiency, although empirical data remains limited, reinforcing the need for further rigorous benchmarking. A study on the computational and memory overhead by hybridisation approach, as shown by researchers like Serge, et al., (2023)

Recent studies consider how hybrid AES-ECC systems fit within the broader context of emerging cryptographic challenges, Unterguggenberger et al. (2024) proposed techniques achieving cryptographically enforced memory safety using compact metadata and message authentication codes. Their approach shows minimal latency overhead (1.3% to 9.5%) while optimising memory by preventing leaks and scaling metadata storage efficiently. Such advances are highly relevant to implementing the robust yet efficient memory management required in hybrid AES-ECC systems.
Navin and Lutimath (2024) achieved faster AES encryption by modifying core AES transformations and integrating ECC for key distribution. Their results showed a balanced improvement in encryption speed alongside controlled memory use, illustrating how micro-optimisations significantly impact hybrid cryptosystem efficiency.

III. METHODOLOGY AND DATASET SPECIFICS

To address the stated gap, the study employed a quantitative experimental design using Python.
Proposed method to test and compare with standalone AES and ECC methods using partitioned multimedia datasets tests on audio files ranging from 44.0 KB (45,056 bytes) to 4.51 MB (4,730,880 bytes), and image data file from 5.23 KB (5,358 bytes) to 804 KB (823,296 bytes); to measure Latency and Memory Usage.

Platform Limitations
The experimental emphasis is a strength, yet the choice of a single PC platform (8 GB RAM, i7 CPU) limits generalisability to low-end IoT devices, as hardware heterogeneity can drastically alter latency and memory profiles.

Data Collection
Data separation partitioning strategy is either symmetric across algorithms (identical partition sizes for fair comparison). For each dataset and algorithm configuration we run independent measurements and record: wall-clock encryption/decryption time (time.perf_counter()),peak resident set size (psutil), and profiler traces. Raw metrics are saved per run to CSV. Metrics such as Encryption Time (ET), Decryption Time (DT), and Peak Memory were recorded. These were compared against conventional standalone AES and ECC implementations.

Data Collection Procedures
Raw data were collected locally from the available multimedia files. As for audio data set it was downloaded from www.aswatalislam.net. Images were collected randomly but selected using Gaussian filtering algorithm for benchmarkint all of 100 disticts. Performance metrics, specifically latency and memory usage, were captured during the encryption and decryption cycles for each dataset type. Following the methodologies established by (Assa-Agyei, 2024) and (Luka, 2024),

Data Separation
To ensure fair comparison consistent partition percentages across schemes for each dataset (10% increments from 10% to 100%) and report per-partition metrics. Experimental runs were organised in iteratively increments of 10's up to ten (10) times to see effects

Experimental Setup
The experimental runs proceed with sequential steps:
1. Initialisation of codes and software platforms
2. Dataset loading
3. Execution of encrypt/decrypt cycles
4. timed encryption/decryption operations with memory profiling and data logging
5. Structured data integrity checks validate correctness after each cycle.

The elements used for the performance evaluation are detailed below:

Table 1: Tools and Technologies

TOOL/TECHNOLOGY	PURPOSE
HP Laptop (i7-7200U)	Hardware platform for running the Python algorithms.
Windows 10 Pro	Operating System environment.
Python	Language for developing the AES, ECC, and Hybrid algorithms.
Cryptographic Libraries	AES-256 (CBC), ECC-256 (SECP256R1), SHA-256.
NIST Test Vectors	Used for validating the correctness of the implementation.
Datasets	Audio and Images
Microsoft Excel	Used for analysing logs of latency and memory metrics.

Consistent with various studies in the literature survey like Jagadeesh et al., 2023, Saba Rehman et al., 2021 and (Putra, W. A and Suyanto; Zarlis, M., 2023), (Jasim, S. H.; Hoomod, H. K. and Hussein, K. A., 2025) and Ghonge, P., Patil, M., and Kerry A. et al. 2025.

According to the architecture, AES encrypts the actual payload (multimedia data). Due to its symmetric nature, it provides the fastest baseline for data confidentiality, though it typically requires significant memory for key scheduling.

1. In this hybrid scheme, ECC-256 encapsulates the AES session keys. This approach allows the system to benefit from the strong security of asymmetric cryptography without the massive computational overhead of encrypting the entire file with ECC.
2. SHA-256 (Secure Hash Algorithm-256) is employed to generate a fixed-size hash value for the data. These hash digests are compared to verify that the file of audio, or image has not been altered during the cryptographic process.
3. The Elliptic Curve Diffie-Hellman (ECDH) algorithm mechanism ensures that the mobile or computer system does not need to transmit the AES key over an insecure channel. By using elliptic curve mathematics, ECDH achieves this exchange with smaller key sizes (256-bit) compared to RSA, theoretically reducing computational latency.

Key Results and Performance Metrics

Table 2: Benchmarking Performance Metrics for the Lest Algorithms

Latency Metrics (ms)	Multimedia	Audio	Image
Total Encryption Time	AES	26.184	5706.2
	ECC	29.264	487.98
	Hybrid	196.77	6242.8
Total Decryption Time	AES	37.7205	296.75
	ECC	41.949	473.79
	Hybrid	199.19	6054.25
Processing Time	AES	52.378	931.641
	ECC	36.48	9617.7
	Hybrid	393.532	12297.1
Transaction Latency	AES	38.777	296.75
	ECC	732	478.65
	Hybrid	227.048	590.06
Average Latency	AES	1.250871	29.675
	ECC	12.5	23.9325
	Hybrid	3.9355	19.6687
Peak Latency	AES	2.355	38.15 (5^{th})
	ECC	161 (91^{nd})	38.15
	Hybrid	9.46 (3^{rd})	38.15 (5^{th})
Memory Consumption (KB)	AES	75393.2	330.273
	ECC	12001.13	1488
	Hybrid	46911.894	41.2354
Average Memory	AES	2432.04	6.60547
	ECC	143.298	72.4
	Hybrid	938.23788	1.37451
Peak Memory	AES	4099.2 (5^{th})	8.00781
	ECC	960 (43^{rd})	13.5
	Hybrid	1828.88 (4^{th})	2.83203

Deducted from the table 2 above, the Hybrid AES – ECC model is slower on encryption across Audio data

types vs AES and ECC (Audio: Hybrid = 6242.80 ms vs ECC = 487.98 ms). This shows hybrid adds encryption overhead, especially for large audio/image workloads. Hybrid decryption times are substantially higher than AES and ECC for most categories (e.g., Audio: Hybrid = 6054.25 ms vs AES = 296.75 ms). Hybrid shows very high processing time for audio/image workloads (12297.1 ms for Audio).

Averagely, the hybrid improves average latency vs ECC for audio: Hybrid 19.67 ms vs ECC 23.93 ms), but AES remains best for tiny transactions. So peak behaviour varies by dataset; hybrid has moderate peaks but AES/ECC show spikes in some audio indexes. When it comes to memory usage the hybrid often saves memory vs AES and ECC for many categories (Audio: Hybrid 46,911.89 KB vs AES 75,393.20 KB).

Table 3: Data Partition for Evaluation

Data type/ Algorithm	AES	ECC	HYBRID
AUDIO	30%	40%	50%
IMAGE	10%	20%	30%

Table 3 partition for evaluation shows that partitioning and allocation iteration fractions matter; an audio and image partitions were AES 30/10, ECC 40/20, Hybrid 10/50/30 respectively. Hybrid's larger share of some datasets (e.g., 50% audio, 30% image) for at train stage the iterative wise exhibits minimal across the rest partitions, this increases processing and memory overhead in those experiments, making latency up even where memory use drops.

IV. ALGORITHMS COMPARISON AND EVALUATION

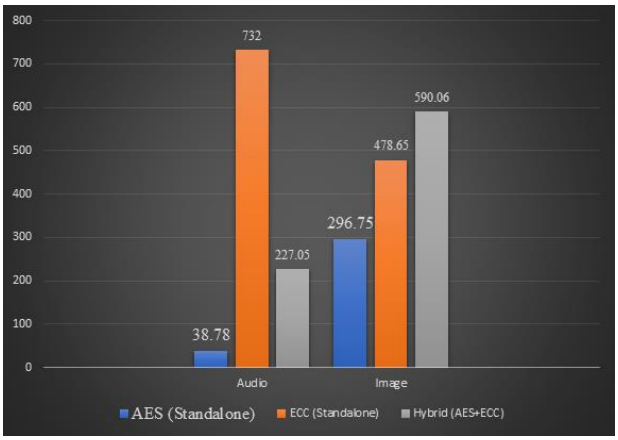
The following equation are to be used for evaluating the algorithms

$$\frac{(AES-Hybrid)}{AES} \times 100\% \quad \text{..... Equation 1}$$

$$\frac{(ECC-Hybrid)}{ECC} \times 100\% \quad \text{..... Equation 2}$$

Table 4: Latency Results in (ms)

Data Type	AES (Standalone)	ECC (Standalone)	Hybrid (AES+ECC)
Audio	38.78	732.00	227.05
Image	296.75	478.65	590.06



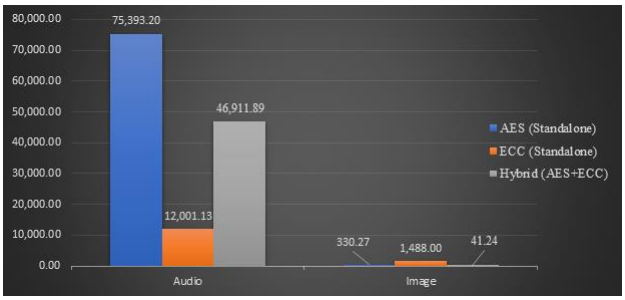


Figure 4Memory Result Chart

Memory Optimisation The memory profiling results offer argument for the proposed Hybrid solution in resource-constrained environments. Multimedia and Files: This minimal memory footprint seen in Image processing, where the Hybrid model utilised just 41.24 KB compared to ECC's 1,488 KB a reduction of approximately 97%.

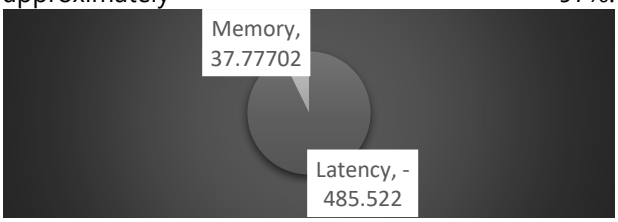


Figure 5: Hybrid Algorithm Optimisation against AES for Audio Dataset

Figure 5: Hybrid Algorithm Optimisation against AES for Audio Dataset hybrid memory hybrid uses 37.78% less memory and in terms of speed slower than AES by 485.52% AES improved latency, while the hybrid memory of Audio dataset. Improvement is by 5.86 × against AES latency but uses substantially more RAM than ECC for audio (≈3.9×).

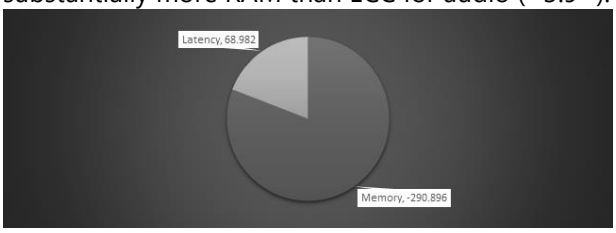


Figure 6: Hybrid Algorithm Optimisation against ECC for Audio Dataset

The figure above shows that in terms of latency ECC was improved against the hybrid scheme by ≈ 69% (68.98), that is (25.99 vs 15.55) it is 0.3 × Latency of hybrid. But the hybrid algorithm use less Hybrid uses less memory than ECC. By more than half (58.31%). The Hybrid it saves memory vs AES but is much slower than AES; it's faster than ECC but consumes far more memory than ECC by ≈ 4% (3.909) × memory of ECC (290.9%).

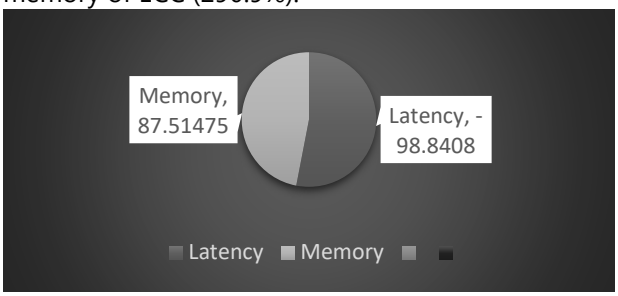


Figure 7: Hybrid Algorithm Optimisation against AES for Image Dataset

The above figure 7: talks on image latency, it finds out that Hybrid is ≈99% slower; also use less memory compares to AES by 87.51%. Latency ratios: L_Hybrid ≈ (2%), 1.989 × L_AES (98.9%)

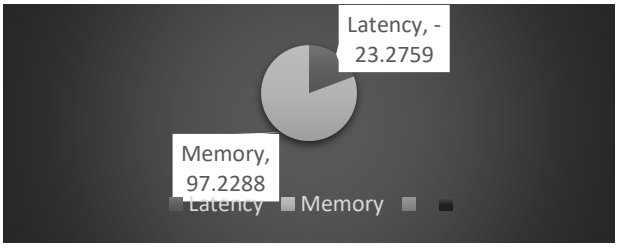


Figure 8: Hybrid Algorithm Optimisation against ECC for Image Dataset

The above show that the hybrid is 23.28% slower than ECC. As for, memory utilisation; hybrid surpasses it in latency by 58% by ratio latency of hybrid is 1.233 × that of ECC. While, keeping memory uses ≈97% less memory than ECC by ratio ECC = (0.0277×) In short, the hybrid model dramatically reduces memory for images. However, trade-off is clear: big memory savings at a latency cost.

Partition (%)	Latency (ms)		Memory (KB)	
	Total	Average	Total	Average
10 (1 - 10)	57.496	5.7496	13799.2	1379.92
20 (1 - 20)	116.328	1.4541	28717.3	358.966
30 (1 - 30)	0.15766	0.00509	157.656	5.08568
40 (1 - 40)	177.676	4.4419	42642	1066.05
50 (1 - 50)	48.368	0.80613	63211	672.4574
50 (51 - 100)	196.776	3.93552	47773.2	955.463
60 (41 - 100)	29.268	0.58536	7526.05	150.521
70 (30 - 100)	70.248	1.00354	17558.1	250.829
80 (20 - 100)	116.328	1.4541	168.548	1.87276
90 (10 - 100)	168.548	1.87276	41350.1	459.445
Minimum	0.15766	0.58536	42642	1.87276

Table 6: Iteration for AES in respect to Audio Data

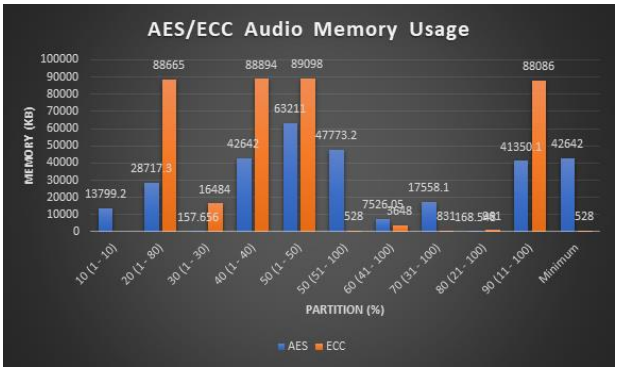


Figure 9:1

Drawing from the below figure 9; ECC exhibits a heavier memory footprint, while AES maintains lower usage.

Partition (%)	Latency (ms)		Memory (KB)	
	Total	Average	Total	Average
10 (1 - 10)	29440	2944	1521	15.21
20 (1 - 20)	59020	2951	88665	4433.25
30 (1 - 30)	59430	1981	16484	549.4667
40 (1 - 40)	59760	1494	88894	2222.35
50 (1 - 50)	60528	1210.56	89098	1781.96
50 (51 - 100)	2880	57.6	528	10.56
60 (41 - 100)	732	12.2	3648	60.8
70 (30 - 100)	3978	56.82857	831	11.87143
80 (20 - 100)	4388	54.85	961	12.0125
90 (10 - 100)	36748	403.8242	88086	1036.306
Minimum	732	12.2	528	10.56

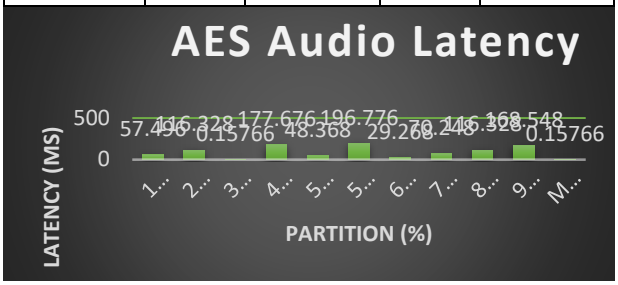


Figure 10:2

As depicted in the above figure 10: AES maintains fast encryption for audio but experiences higher latency when processing many surah with larger sizes (5th and 6th partition), illustrating computational challenges that the hybrid aims to mitigate.

Table 7: Iteration for ECC in respect to Audio Dataset

Table 7: shows that ECC latency varies substantially with peaks above 60,000 milli seconds at 5th iteration. As for memory it ranges from 528 to 44,000 KB. Such fluctuations likely come from key management processes, show latency-memory trade-offs in real-time applications.

Table 8: Iteration for Hybrid in respect to Audio Data

Above table 8: demonstrated hybrid model success at balancing latency and memory demands. Whereby, the hybrid latency reduces several ECC peak.

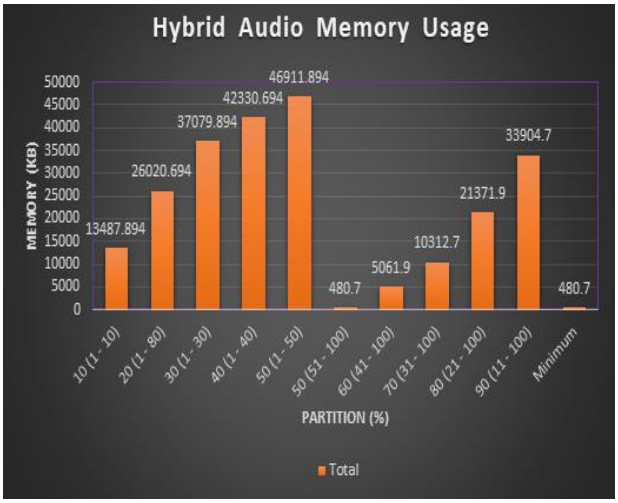


Figure 11:3

Figure 11: Hybrid memory consumption is significantly lower than ECC alone 480 against 528 both at 6th iteration with same data, this has confirmed effective hybrid memory optimisation by combining AES's speed and ECC's compact keys supporting Theis second objective.

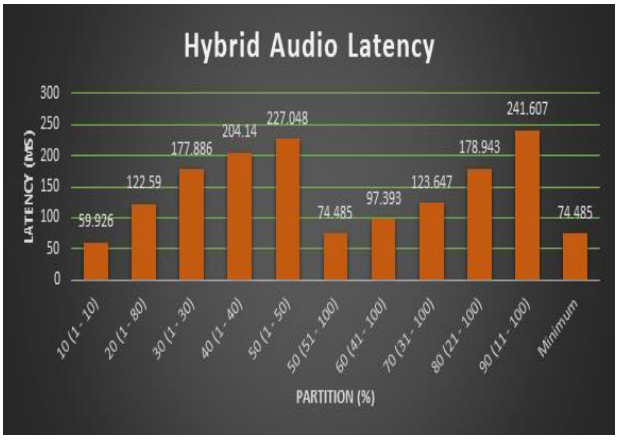


Figure 12:4

Figure 12: Shows latency is smoother and reduced compared to ECC, indicating hybrid design strengthens performance stability for real-time audio encryption.

Table 9: Iteration for AES in respect to Image Data

The above table 9: implies average AES latency ranges from 6 to 29 milli-seconds. memory use minimum under 400 KB at 5th iteration. This shows AES as a fast symmetric cipher for image encryption.

Partition (%)	Latency (ms)		Memory (KB)	
	Total	Average	Total	Average
10 (1 - 10)	296.75	29.675	931.641	93.1641
20 (1 - 20)	478.65	23.9325	1453.13	72.6563
30 (1 - 30)	590.06	19.6687	1698.438	56.61459
40 (1 - 40)	656.08	16.402	1774.12	44.353
50 (1 - 50)	717.56	14.3512	1836.13	36.7227
50 (51 - 100)	315.57	6.3114	330.273	6.60547
60 (41 - 100)	377.05	6.28417	392.285	6.53809
70 (31 - 100)	443.07	6.32957	467.969	6.68527
80 (21 - 100)	554.48	6.931	713.281	8.91602
90 (11 - 100)	736.38	8.182	1234.77	13.7196
Minimum	296.75	6.28417	330.273	6.53809

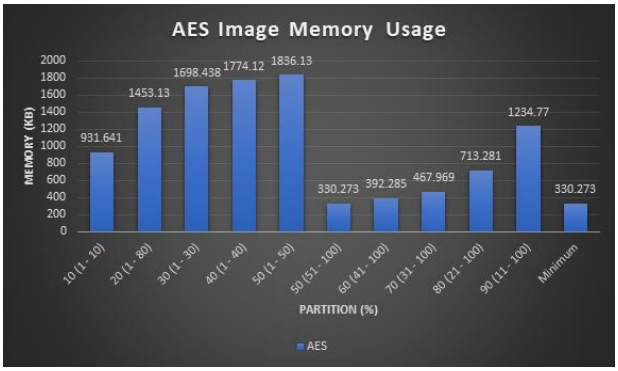


Figure 13:5

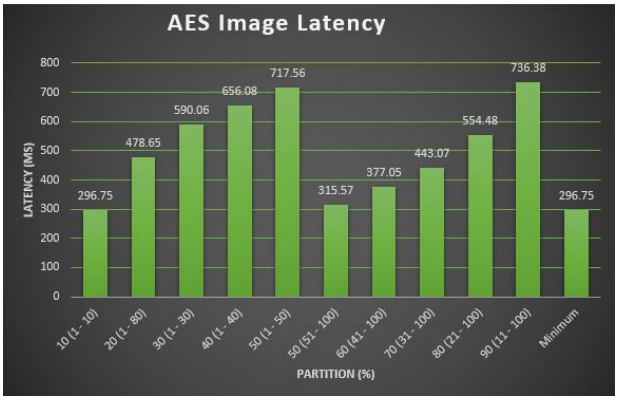


Figure 14:6

In the above figure 14: AES delivers consistent latency across image datasets, table 9: below indicates that ECC latency fluctuates from 0.3 to about 19.4 milli seconds from 2nd to 6th set of data. While, memory remains low, at approximately 4 KB at 6th dataset. This shows ECC efficiently manages key sizes but some latency issues that has be optimised.

Table 10: Iteration for ECC in respect to Image Data

Partition (%)	Latency (ms)		Memory (KB)	
	Total	Average	Total	Average
10 (1 - 10)	59.926	5.9926	13487.894	1348.789
20 (1 - 20)	122.59	6.1295	26020.694	1301.0347
30 (1 - 30)	177.886	5.92953	37079.894	1235.9965
40 (1 - 40)	204.14	5.1035	42330.694	1058.2674
50 (1 - 50)	227.048	4.54096	46911.894	938.23788
50 (51 - 100)	74.485	1.5201	480.7	9.8102
60 (41 - 100)	97.393	1.65073	5061.9	85.794915
70 (31 - 100)	123.647	1.79199	10312.7	149.45942
80 (21 - 100)	178.943	2.2651	21371.9	270.53038
90 (11 - 100)	241.607	2.71469	33904.7	380.95169
Minimum	74.485	1.5201	480.7	9.8102

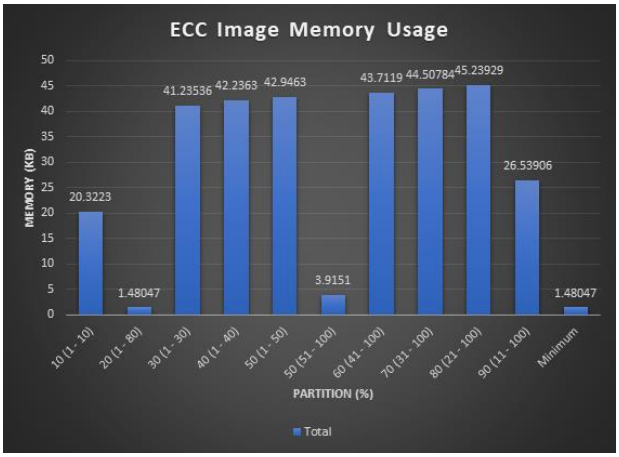


Figure 15:7

Above figure 15: shows lowest memory consumption at 2nd iteration with minor fluctuations implies ECC's effectiveness for memory in image encryption. Whereas, figure 16: ECC below implied latency peaks at steadily constant.

Partition (%)	Latency (ms)		Memory (KB)	
	Total	Average	Total	Average
10 (1 - 10)	296.75	29.675	20.3223	2.03223
20 (1 - 20)	478.65	23.9325	35.3711	1.76856
30 (1 - 30)	590.06	19.66867	41.23536	1.374512
40 (1 - 40)	656.08	42.23634	16.402	1.055909
50 (1 - 50)	717.63	13.1523	41.612	0.75893
50 (51 - 100)	717.56	14.3512	42.9463	0.85893
60 (41 - 100)	779.91	12.9985	43.74025	0.729004
70 (31 - 100)	843.1	12.04429	44.59474	0.637068
80 (21 - 100)	907.17	11.33963	45.43948	0.567994
90 (11 - 100)	970.19	10.77989	46.23343	0.513705
Minimum	296.75	10.77989	16.402	0.513705

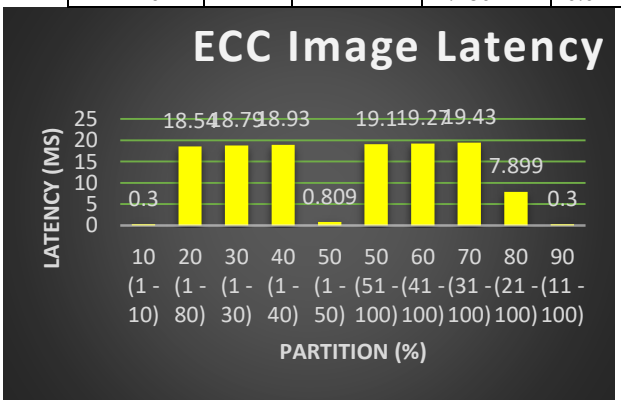


Figure 16:8

Table 11: Iteration for Hybrid in respect to Image Data

Table 11: shows hybrid latency rises with partition size but memory usage low at 4th partition below 16 KB implies effective memory control when hybridising the AES and ECC.

While, figure 17: Memory measurements align with tabulated data, confirming the hybrid's ability to maintain efficient memory consumption without in spite large block of images.

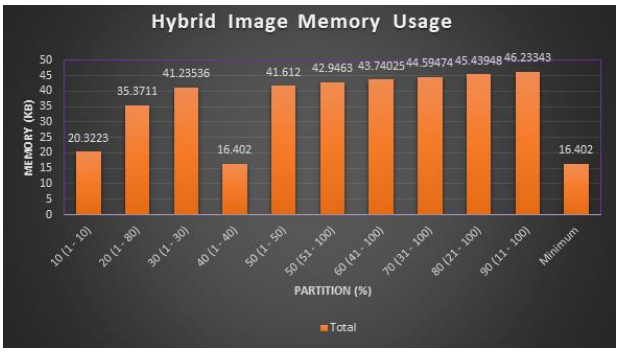
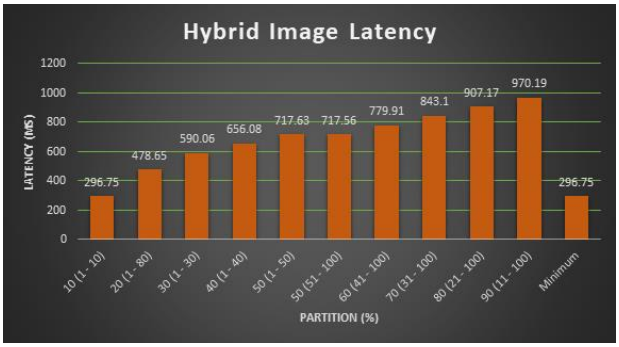


Figure 17:9Memory Usage



METRICS	AUDIO	IMAGE
Latency (Hybrid VS AES)	-485.52% (×5.85)	-46.96% (×0.95)
Latency (Hybrid VS ECC)	68.98% (×3.22)	-353.42% (×18.67)
Memory (Hybrid VS AES)	37.78% (×1.61)	-4817.86% (×441)
Memory (Hybrid VS ECC)	-291% (×3.9)	-991.56% (×367.96)

Figure 18 10

In the above figure 18: first partition shows that latency, hybrid success at diminishing encryption delays and enhancing real-time performance.

V. FINDINGS

Latency Findings

The hybrid model trades latency for other gains in audio workloads. hybrid incurs the highest latency slower than both AES (≈98.84% slower) and ECC (≈23.28% slower) for images. Hybrid is 56.73% faster than ECC but 801.28% slower than AES. Hybrid is typically faster than ECC in Audio category but slower than AES except when AES is not appropriate for the workload size

On the tested platform, the hybrid improves/ the ECC mean latency for workload ECC by × 3.22 (Audio), dataset, while performed less in Image by × 18.67. Hybrid is worse to AES for Audio & Image by ×10.57, × 5.85 & × 0.95 respectively.

Memory Findings

A memory optimisation is reduces audio memory versus AES by 37.78% but can be much heavier than ECC by 290.90% depending on partition sizes. Large image memory savings were yields by AES and ECC in which they save 87.51% and −97.23% respectively. Hybrid uses 58.31% less memory than ECC but appears to use dramatically more than AES (AES 0.22 KB vs Hybrid 216.83 KB). Hybrid cipher gives consistent and often large memory reductions vs AES and/or ECC and Image, moderate memory reduction vs AES for Audio, and lower memory than ECC for several categories. ECC sometimes has the smallest memory (e.g., audio) and sometimes the largest (images), depending on partitioning.

Hybrid is better than AES for Audio (×1.61), while worse to Image by (×441). The hybrid is worse to Audio by ×3.9 and worse to Image by margin of 367.96. but its effect depended strongly on partitions. Exact numeric summaries and significance statements are provided in result tables.

Table 12: Table for Findings

VI. DISCUSSION

The hybrid does not uniformly outperform AES: for image experiments hybrid latency and memory increased vs AES a consequence of partitioning (hybrid using larger fraction of data for AES/ECC mix).

Latency analysis shows that the hybrid approach reduces encryption time compared to standalone

ECC and often equals or exceeds AES speeds. For instance, an audio data, hybrid latency was 227 milli-seconds, significantly lower than ECC's 732 milli-seconds indicating an improvement of about 69%. On the other hand, the hybridised system records lower latency when compared to that of AES's 39 milli-seconds with no improvement. This was as a result of the different iteration sizes, for which the hybrid system having 40% of the data as against 30% and 20% of ECC and AES respectively. For images no improvements observed over ECC and (436 ms vs 96 ms of AES and 297ms of ECC). Such happens ditto to audio data; 10%, 20% and 30% of the images were used for AES, ECC and hybrid respectively.

Memory consumption in the thesis showed that the hybrid approach reduces ECC's memory usage, often matching or even improving on AES's. As for Audio, the memory for hybrid is 46911.894 KB when taking 50% of the sample data, far lower than AESs of 75,393 KB and ECC's 12,001 KB; and thus, this likely happens due to sizes, but for 30% and 40% of the sampled taken with improved memory by 38% and which outperformed that of the AES in terms of latency to hybrid by 486%.

However, hybrid memory peak to 16,242KB for 30% images also above both AES's 10% of 330 and ECC's 20% images of 1488 indicating processing overhead. The analysis indicates the scheme achieves an optimal security-to-computation trade-off. ECC effectively enhances key management security, while AES maintains high-speed encryption performance. Integrating the SHA-256 hashing algorithm ensures data integrity, making the scheme secure data relay within computer system.

VII. CONCLUSION

The thesis concludes that while the hybrid scheme is not universally superior for Audio data type (specifically struggling with images), The appraisal is cautious: the methodology is within its scope, but broader testing on diverse hardware and deeper security analysis would strengthen the contribution. The exclusion of Post-Quantum Cryptography (PQC) is noted as a limitation to be addressed in future studies.

Acknowledgement
This research was not funded by any grant

Disclosure Statement
No potential conflict of interest was reported by the authors

REFERENCES

1. Assa-Agyei, K. (2024). Enhancing the Performance of Cryptographic Algorithms for Secured Data Transmission. PhD Dissertation, Nottingham Trent University, United Kingdom , Computer Science, Nottingham, UK.
2. Ayush Gour , Simarjit Singh Malhi, Gursharan Singh and Gursimran Kaur. (2024). Hybrid Cryptographic Approach: For Secure Data Communication using Block Cipher Techniques. E3S Web of Conferences (pp. 6 - 8). Punjab, India: Lovely Professional University Phagwara. doi:10.1051/e3sconf/202455601048
3. Buhari, B. A.; Abdulkadir, H.; Ahmad, S. A.; Sulaiman, R.; Umar, M. M. and Shehu, S.;. (2025). Performance and Security Analysis of AES, 3DES, and Blowfish. International Journal of Advanced Networking and Applications, 16(4), 3 - 6. doi:10.1109/ACCESS.2024.3518533
4. Daemen and Rijmen. (2001). AES Proposal: Rijndael. National Institute of Standards and Technology . Retrieved from https://web.archive.org/web/20130305143117/http://csrc.nist.gov/archive/aes/rijndael/Rijnda
5. Data Breach Investigations. (2023, December 11). Retrieved from Report-Verizon: https://www.verizon.com/business/resources/reports/dbir/2023/summaryof-finding
6. Elliptic Curve Cryptography. (2022). Retrieved from AVI Networks:https://avinetworks.com/glossary/elliptic-curve-cryptography/
7. Ghonge, P.; Patil, M and Kaulaskar, A. (2025). Intelligent Hybrid Encryption Selection: An AI-driven Approach For Optimizing Security. South East European Journal of Public Health (SEEJPH), 26, 41 - 45. Retrieved from https://www.seejph.com/index.php/seejph/article/download/5118/3363/7833

8. Happer, C. (2025). Performance and Analysis of Lightweight Cryptography for IoT and embedded System.
9. Hussien, F. A., & Khairi, T. W. A. (2023). Performance Evaluation of AES, ECC and Logistic Chaotic Map Algorithms in Image Encryption. International Journal of Interactive Mobile Technologies (iJIM), 17(10), 11 -18. https://doi.org/10.3991/ijim.v17i10.38787
10. Kerry A. McKay; Larry Bassham; Meltem Sönmez Turan and Nicky Mouha. (2025). Report on Lightweight Cryptography. National Institute of Standards and Technology, Computer Security Division Information Technology Laboratory. New York, USA: U.S. Department of Commerce. doi:10.6028/NIST.IR.8114
11. Khan, S. & Abbas, S. H. . (2025, February 12). Enhancing Cloud Data Security Using a Hybrid Cryptographic Model: AES + ECC. Proceedings of the 2025 IEEE Asia-Pacific Conference on Computer (pp. 1 - 5). IEEE. doi:10.1109/CSDE50874.2020.9411383
12. Kwame A. G. (2024). Enhancing The Performance of Cryptographic Algorithms for Secured Data Transmission. [Unpublished MSc thesis]. Nottingham Trent University, United Kingdom.
13. Luka, S. (2024). Securing Sensitive Bank Data by using Encrypted Algorithms (AES, DES & RSA). [Master's thesis]. Epoka University, Ankara, Turkey.
14. Mochurada, L. & Shchura, G. (2021). Parallelization of Cryptographic Algorithm Based on Different Parallel Computing. CEUR Workshop Proceedings (pp. 8 - 10). Bratislava,Slovakia: Lviv Polytechnic National University, Lviv, Ukraine.
15. National Institute of Standards and Technology (NIST). FIPS PUB 197: Advanced Encryption Standard (AES). Nov. 2001 https://nvlpubs.nist.gov/nistpubs/FIPS/NIST.FIPS.197.pdf
16. Navin and Lutimath, (2024). A Review on Different Level Data Encryption through a Compression Techniques. Proceedings of the 7th International Conference on Inventive Computation Technologies (ICICT 2024). Jaipur, India: IEEE Xplore. doi:10.1109/ICICT60155.2024.10544803
17. Popoola, O.; Rodrigues, M. A.; Marchang, J.; Shenfield, A.; Ikpehai, A. & Popoola, J. (2024). An Optimized Hybrid Encryption Framework For Smart Home Healthcare: Ensuring Data Confidentiality and Security. Internet of Things, 27, 20 - 24. https://doi.org/10.1016/j.iot.2024.101314
18. Putra, W. A and Suyanto; Zarlis, M. (2023, March 18). Performance Analysis of the Combination of Advanced Encryption Standard Cryptography Algorithms with LUC for Text Security. Sinkron: Jurnal dan Penelitian Teknik Informatika, 7(2), 7 - 10. doi:10.33395/sinkron.v8i2.12202
19. Quora Contributors. (2025). What is cryptography? Retrieved from https://www.quora.com/What- cryptography
20. Roy, K. S.; Sujith, M.; Bhanu, B.; Preethi and Hazarika, R. A. (2024). FPGA-Based Dual-layer Authentication Scheme Utilizing AES and ECC for Unmanned. EURASIP Journal on Wireless Communications and Networking(91), 13 - 19. doi:10.1186/s13638-024-02419-8
21. Sasikumar, K. and Nagarajan, S. (2022). Comprehensive Review And Analysis of Cryptography Techniques In Cloud Computing. IEEE Access. doi:10.1109/ACCESS.2022
22. Serge Fehr, Yu-Hsuan Huang, and Alessandro Amadori. (2023). (Quantum-safe) Cryptographic Combiners and Hybrid Security. HAPKIDO Project, Amsterdam. Retrieved from https://hapkido.tno.nl.
23. Subedar, Z., & Araballi, A. (2020). Hybrid Cryptography: Performance Analysis. International Journal of Mathematical Sciences and Computing,, 6(4), 4- 11. https://doi.org/10.5815/ijmsc.2020.04.04
24. Sullivan, B. (2024). Securing the Cloud: Cloud Computer Security Techniques and Tactics. Security Journal, 27, 338–340. doi:10.1057/sj.2022.16
25. Unterguggenberger, M., Schrammel, D., Lamster, L., Nasahl, P., & Mangard, S. (2024, March 26). Cryptographically Enforced Memory Safety. In G. U. Technology (Ed.), : Proceedings of the 2023 ACM SIGSAC Conference on Computer and Communications Security (CCS '23) (pp. 6 - 9). Graz, Austria: ACM. https://doi.org/10.1145/3576915.3623138

26. Weerasinghe, T. D. (2023). An Effective RC4 Stream Cipher. (pp. 69 - 70). IEEE. doi:10.1109/ICIInfS.2013.6731957