9 772527 583007

JISKa

Jurnal Informatika Sunan Kalijaga

Vol. 11 No. 1, January 2026

ISSN : 2527-5836
e-ISSN : 2528-0074

Vol. 11 No. 1, January 2026

JISKa

Jurnal Informatika Sunan Kalijaga

Jurusan Teknik Informatika
Fakultas Sains dan Teknologi
UIN Sunan Kalijaga Yogyakarta

Vol. 11 No. 1 (2026): January 2026

JISKa (Jurnal Informatika Sunan Kalijaga) Vol. 11, No. 1 (2026): January 2026 Edition

PUBLISHED: 2026-01-25

ARTICLES

MAKE A SUBMISSION

AUTHOR INFORMATION

Author Guidelines

Focus and Scope

Publication Ethics

Artificial Intelligence (AI) Ethics

Privacy Statements

Article Processing Charge (APC)

Peer Review Process and Plagiarism Policy

Open Access Policy dan Copyright Notice

Submission Templates

Contacts

INDEXED BY

DOI: https://doi.org/10.14421/jiska.5075 Views: 0 Downloads: 0

Sistem Deteksi Gerakan Kecurangan UTBK Real-Time dengan YOLOv8 dan Optical Flow

Muhammad Naufal Ardiansyah, Farrel Zikri Suryahadi, Hendrico Edhent Surya Pratama, Anggraini Puspita Sari

PDF

DOI: https://doi.org/10.14421/jiska.5365 Views: 0 Downloads: 0

Comparative Analysis of Camellia and AES Across File Sizes and Types

Teja Endra Eng Tju, Fauzi Alfadhillah

PDF

DOI: https://doi.org/10.14421/jiska.5418 Views: 0 Downloads: 0

Model Prediksi Risiko Kanker Serviks dengan Pendekatan Support Vector Machine

Juwita Stefany Hutapea, Nisa Hanum Harani, Cahyo Prianto

PDF

DOI: https://doi.org/10.14421/jiska.5445 Views: 0 Downloads: 0

Comparative Analysis of Hybrid CNN-ViT and CNN for Brain Tumor Classification

Ahmad Fauzi, Achmad Lutfi Fuadi, Agus Heri Yunial

PDF

DOI: https://doi.org/10.14421/jiska.5860 Views: 0 Downloads: 0

JISKA (Jurnal Informatika Sunan Kalijaga) is published by the Dept. of Informatics Engineering, Faculty of Science and Technology, UIN Sunan Kalijaga Yogyakarta, Indonesia.

ISSN: 2527-5836 (Print), 2528-0074 (Online)

Reviewers

Agung Dewandaru, Institut Teknologi Bandung, Indonesia (Scopus h-index: 3)

Agus Mulyanto, UIN Sunan Kalijaga Yogyakarta, Indonesia (Scopus h-index: 2)

Ahmad Fathan Hidayatullah, Universitas Islam Indonesia Yogyakarta, Indonesia (Scopus h-index: 10)

Alam Rahmatulloh, Universitas Siliwangi Tasikmalaya, Indonesia (Scopus h-index: 9)

Anggi Rizky Windra Putri, Universitas Siber Muhammadiyah Yogyakarta, Indonesia (Scopus h-index: 1)

Ardiansyah Musa Efendi, Singapore Chipset Algorithm Design Lab, Huawei, Singapore (Scopus h-index: 8, Linkedin)

Bambang Sugiantoro, UIN Sunan Kalijaga Yogyakarta, Indonesia (Scopus h-index: 2)

Enny Itje Sela, Universitas Teknologi Yogyakarta, Indonesia (Scopus h-index: 5)

Ganjar Alfian, Universitas Gadjah Mada, Indonesia (Scopus:Scopus h-index: 20)

Mandahadi Kusuma, UIN Sunan Kalijaga Yogyakarta, Indonesia (Scopus h-index: 2)

Maria Ulfah Siregar, UIN Sunan Kalijaga Yogyakarta, Indonesia (Scopus h-index: 2)

Millati Pratiwi, Pusan National University, South Korea (Scopus h-index:1)

Muhammad Dzulfikar Fauzi, Telkom University Surabaya, Indonesia (Scopus h-index: 2)

Muhammad Habibi, Universitas Jenderal Achmad Yani Yogyakarta, Indonesia (Scopus h-index: 3)

Muhammad Rifqi Maarif, Universitas Jenderal Achmad Yani Yogyakarta, Indonesia (Scopus h-index: 9)

Mohd. Fikri Azli bin Abdullah, Multimedia University, Malaysia (Scopus h-index: 7)

M. Alex Syaekhoni, Who's Good, South Korea (Scopus h-index: 6, Linkedin)

Niki Min Hidayati Robbi, Universitas Gadjah Mada, Indonesia (Scopus h-index: 2)

Norma Latif Fitriyani, Sejong University Seoul, South Korea (Scopus h-index: 19)

Okfalisa, UIN Sultan Syarif Kasim Riau, Indonesia (Scopus h-index: 12)

Oman Somantri, Politeknik Negeri Cilacap, Indonesia (Scopus h-index: 4)

Puguh Jayadi, Universitas PGRI Madiun, Indonesia (Scopus h-index: 1)

Puji Winar Cahyo, Universitas Jenderal Achmad Yani Yogyakarta, Indonesia (Scopus h-index: 1)

Qorry Aina Fitroh, UIN KH. Abdurrahman Wahid Pekalongan, Indonesia (Scopus h-index: 0)

Ridho Surya Kusuma, Universitas Siber Muhammadiyah, Yogyakarta, Indonesia (Scopus h-index: 2)

Rischan Mafrur, Macquarie University, Sydney, Australia (Scopus h-index: 4)

Shofwatul Uyun, UIN Sunan Kalijaga Yogyakarta, Indonesia (Scopus h-index: 5)

Sumarsono, UIN Sunan Kalijaga, Indonesia (Scopus h-index: 3)

Sunu Wibirama, Universitas Gadjah Mada, Indonesia (Scopus h-index: 18)

Tundo, Sekolah Tinggi Ilmu Komputer Cipta Karya Informatika (STIKOM CKI), Indonesia (Scopus h-index: 1)

Windra Swastika, Universitas Ma Chung, Indonesia (Scopus h-index: 5)

Yudistira Dwi Wardhana Asnar, Institut Teknologi Bandung, Indonesia (Scopus h-index: 11)

MAKE A SUBMISSION

AUTHOR INFORMATION

Author Guidelines

Focus and Scope

Publication Ethics

Artificial Intelligence (AI) Ethics

Privacy Statements

Article Processing Charge (APC)

Peer Review Process and Plagiarism Policy

Open Access Policy dan Copyright Notice

Submission Templates

Contacts

INDEXED BY

STATISTIC

00265599 View My Stats

Visitors
124,494 311
4,675 200
3,165 188
705 178
403 147
FLAG counter

JISKA (Jurnal Informatika Sunan Kalijaga)
Vol. 11, No. 1, JANUARY, 2026, Pp. 98 – 113
ISSN: 2527 – 5836 (print) | 2528 – 0074 (online)

Comparative Analysis of Camellia and AES Across File Sizes and Types

Teja Endra Eng Tju [(1)*], Fauzi Alfadhillah [(2)]
Department of Information Systems, Budi Luhur University, Jakarta, Indonesia
e-mail : teja.endraengtju@budiluhur.ac.id, 2311501767@student.budiluhur.ac.id.
* Corresponding author.
This article was submitted on 30 July 2025, revised on 15 December 2025, accepted on 16 December 2025, and published on 25 January 2026.

Abstract

Data security is a critical aspect of modern information systems that requires processing cryptographic efficiency and resilience. This study compares two widely used symmetric encryption algorithms, named Camellia and AES, based on their performance and resistance to standard attack methods. An experimental approach was applied using 72 files across eight commonly used formats (.mp3, *.jpg, *.png, *.pdf, *.docx, *.xls, *.pptx, and .txt) in three predefined sizes: 100 KB, 1 MB, and 10 MB. Each file underwent encryption and decryption in a controlled environment, with metrics such as processing time, CPU usage, and RAM consumption recorded. Simulated Dictionary, Birthday, and Brute-Force attacks were conducted to assess algorithm robustness. Results show that AES performs faster, especially on large files, but with higher memory usage. Camellia demonstrated more consistent RAM usage and stronger resistance, successfully withstanding all attacks except one brute-force case on a small plaintext file. AES suffered multiple breaches on structured files of smaller sizes. The findings suggest that algorithm selection should consider workload characteristics and system constraints. The main contribution of this research lies in its comprehensive dataset and empirical comparison, providing practical insights to support encryption algorithm choices in real-world applications.*

Keywords: Cryptographic Efficiency, Data Encryption, Cryptanalysis, Algorithm Security, Encryption Performance

Abstrak

Keamanan data merupakan aspek krusial dalam sistem informasi modern yang menuntut efisiensi pemrosesan sekaligus ketahanan kriptografis. Penelitian ini membandingkan dua algoritma enkripsi simetris yang banyak digunakan, yaitu Camellia dan AES, berdasarkan kinerja dan ketahanannya terhadap metode serangan umum. Pendekatan eksperimental diterapkan pada 72 file dalam delapan format umum (*.mp3, *.jpg, *.png, *.pdf, *.docx, *.xls, *.pptx, dan .txt) dalam tiga ukuran: 100 KB, 1 MB, dan 10 MB. Setiap file dienkripsi dan didekripsi dalam lingkungan terkontrol, dengan pencatatan waktu proses, penggunaan CPU, dan konsumsi RAM. Serangan Dictionary, Birthday, dan Brute-Force disimulasikan untuk menilai ketahanan masing-masing algoritma. Hasil menunjukkan bahwa AES lebih cepat, terutama pada file berukuran besar, namun memerlukan memori lebih tinggi. Camellia menunjukkan penggunaan RAM yang lebih stabil dan ketahanan lebih kuat, berhasil menahan semua serangan kecuali satu kasus brute-force pada file teks kecil. AES mengalami beberapa kebocoran pada file terstruktur dengan ukuran kecil. Temuan ini menunjukkan bahwa pemilihan algoritma harus mempertimbangkan karakteristik beban kerja dan batasan sistem. Kontribusi utama dari penelitian ini adalah penyediaan dataset komprehensif dan perbandingan empiris yang memberikan wawasan praktis untuk mendukung pemilihan algoritma enkripsi dalam berbagai skenario aplikasi.

Kata Kunci: Efisiensi Kriptografi, Enkripsi Data, Kriptanalisis, Keamanan Algoritma, Kinerja Enkripsi

1. INTRODUCTION

Data security has become a primary concern as information technology continues to evolve and connects various global systems. The use of cryptographic algorithms to protect sensitive data is the main way to maintain the confidentiality and integrity of information (Manullang, 2023).

Cryptographic algorithms such as Camellia and AES (Advanced Encryption Standard) are widely used to protect sensitive data by ensuring confidentiality and integrity (Sulaiman & Hammood, 2025). Camellia, developed by NTT and Mitsubishi Electric, provides a flexible and efficient structure applicable across diverse platforms (Matsui et al., 2015), while AES, standardized by NIST, is known for its simplicity, strong security, and widespread adoption (Azhari et al., 2022). This research specifically focuses on comparing the performance of Camellia and AES in securing small-to-medium-sized files on specific devices. The study evaluates encryption and decryption efficiency in terms of run time, resource consumption, and robustness against common cyberattacks, such as Dictionary Attack (Adams, 2021; Alkhwaja et al., 2023; Hranický et al., 2025), Birthday Attack (Chavan et al., 2024; Fahdi & Ahmed, 2020), and Brute-force Attack (Hamza & Al-Janabi, 2024; Sarmila & Manisekaran, 2022; Verma et al., 2022). The choice of these two algorithms is justified by their prevalence in modern cryptographic applications and contrasting design philosophies, offering insight into optimal algorithm selection for practical data protection. In practical deployments, encryption is rarely evaluated only by theoretical strength; it must also meet operational constraints such as throughput, memory budget, and predictable behavior across heterogeneous data formats. A system that encrypts files quickly but exhibits higher memory demand or inconsistent robustness under common password-guessing and collision-based attack scenarios may create hidden security and performance risks in real applications. Therefore, evaluating efficiency and robustness together across diverse file structures is essential for making defensible design choices rather than relying on one-size-fits-all assumptions.

Previous research on Camellia and AES cryptographic algorithms has extensively covered their security and resilience against cryptographic attacks. AES has become the international standard due to its strong encryption capabilities and broad acceptance in various security applications (Bibiola et al., 2023). Camellia offers comparable security levels with a more flexible and modular structure, facilitating implementation across different platforms and scenarios (Li et al., 2024). Applications of Camellia encryption have been explored in areas such as authentication for wireless robotics and mobile platforms (Nithishma et al., 2022; Rasheed et al., 2023), quantum implementations with S-box optimizations (ZhenQiang et al., 2023), and security models for e-banking and Internet of Things (IoT) devices (Mohammed et al., 2025; Rashidi, 2021). Meanwhile, AES has been applied in securing digital images (Haris et al., 2023), medical records (Tanjung, 2022), financial transactions (Andriyanto & Sukmasetya, 2022), cloud computing (Dwina et al., 2023), and IoT healthcare networks (Rachmayanti & Wirawan, 2022). Comparative studies between Camellia and AES include quantum circuit designs (Wei et al., 2023), digital image encryption evaluations (Hidayati et al., 2021), and security implementations in 5G networks and IoT frameworks (Arabul et al., 2023; Muthavhine & Sumbwanyambe, 2023; Ning et al., 2020; Panahi & Bayılmış, 2023; Rasheed et al., 2024). Although prior studies have compared Camellia and AES in specific contexts (e.g., images, IoT frameworks, or specialized implementations), many evaluations are scoped to a narrow data type, focus on a single dimension (performance *or* security), or do not jointly examine how file structure and size interact with both resource usage and attack outcomes. This study addresses that gap by conducting a controlled and reproducible experiment across multiple real-world file formats and standardized size categories, reporting both system-level efficiency metrics and empirical robustness under common attack simulations. The resulting workflow is described in the Methods section. This study investigates efficiency differences between Camellia and AES (runtime, CPU usage, and RAM consumption) across file types and sizes, examines how file structure and size relate to empirical outcomes in dictionary, birthday, and brute-force attack simulations, and derives actionable guidelines for algorithm selection under different operational constraints.

The contributions of this paper are threefold: it provides a controlled empirical dataset and test protocol covering 72 files across eight standard formats and three standardized size categories to enable consistent comparison; it conducts a joint evaluation of efficiency (run time, CPU usage, and RAM consumption) and empirical robustness under dictionary, birthday, and brute-force attack simulations within a single experimental pipeline; and it derives evidence-based insights into how file type or structure and size shape performance–security trade-offs, translating these

findings into practical recommendations for encryption algorithm selection in real-world systems implementation.

2. METHODS

This study employs an experimental approach to evaluate the performance of the Camellia and AES algorithms based on the stages illustrated in Figure 1. The process involves collecting data or files comprising various file types, followed by the encryption and decryption of these files using Camellia and AES. Next, Attack Testing on Encrypted Files is performed to assess resistance against different cryptographic attacks. Then, the Performance Analysis of both algorithms is conducted to compare their efficiency and robustness. The study concludes with an Evaluation and Conclusion stage that provides insights into the strengths and limitations of each algorithm in the context of data protection.

The stage begins with the Data or Files Collection, which involves 72 files across eight widely used formats: *.mp3, *.jpg, *.png, *.pdf, *.docx, *.xls, *.pptx, and *.txt. This diverse selection aims to assess how the Camellia and AES algorithms handle data with varying structures in sizes and types. Files were randomly collected from various sources. For each file type, three predefined sizes were prepared: 100 KB, 1 MB, and 10 MB, including audio (.mp3), image (.jpg, .png), document (.pdf, .docx, .xls, .pptx), and plaintext (.txt).

Figure 1 Research Method Stages

Table 1 presents the distribution of file types and predefined sizes (100 KB, 1 MB, and 10 MB), which will be utilized in the encryption and decryption experiments to evaluate the efficiency and robustness of Camellia and AES across various data types and file structures. This structured dataset ensures a balanced and systematic representation of file categories and size variations, providing a consistent basis for fair comparison and reliable performance measurement across all test cases.

After data collection, the study proceeds to the File Encryption and Decryption stage using Camellia and AES. All tests were conducted on a laptop with an AMD Ryzen 7 5800H processor, 16GB RAM, and an NVIDIA GeForce RTX 3050 Ti GPU, running Windows 11. The files used in the experiment vary in format and size, each encrypted and decrypted using a 256-bit key, selected to comply with modern cryptographic standards against various cryptanalysis attacks (Housley & Schaad, 2020; Kato & Moriai, 2013). A strong password was used to derive each

encryption key (Patra & Patra, 2021). The encryption and decryption processes are carried out using a simple web-based application developed in-house with the PyCryptodome library (A et al., 2022; G et al., 2022), which records run time, CPU Usage, and RAM Usage. These procedures allow observation of how each algorithm handles different file types and workloads, highlighting differences in processing efficiency and consistency. Figure 2 presents the encryption flow for both algorithms.

Table 1 Distribution Files

Types	Count by Predefined Sizes			Total
	100 KB	1 MB	10 MB	
*.mp3	3	3	3	9
*.jpg	3	3	3	9
*.png	3	3	3	9
*.pdf	3	3	3	9
*.docx	3	3	3	9
*.xls	3	3	3	9
*.pptx	3	3	3	9
*.txt	3	3	3	9
Total Data				**72**

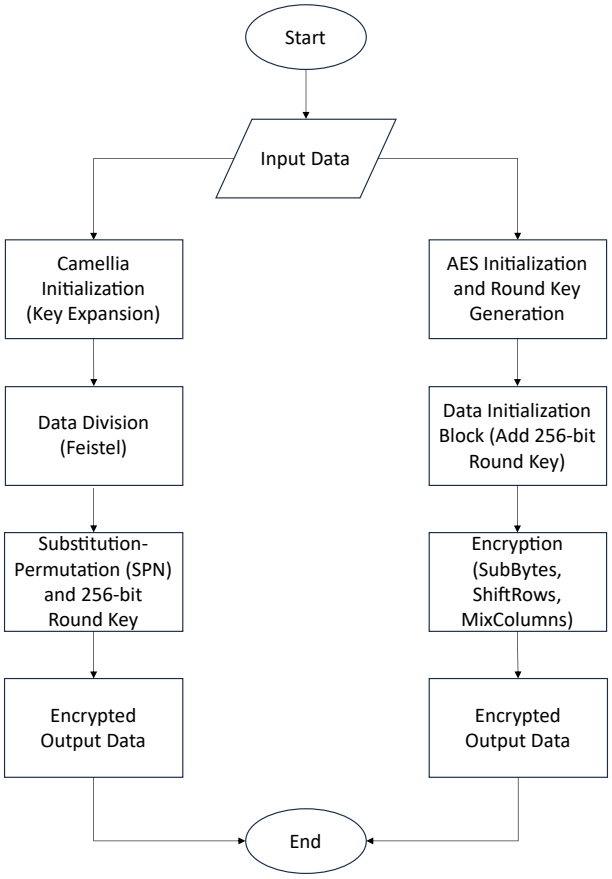

Figure 2 Encryption Process Flow of Camellia and AES Algorithms

The next stage is Attack Testing on Encrypted Files to assess each algorithm's robustness against common cryptographic threats. This study simulated Dictionary, Birthday, and Brute-Force attacks and recorded the attack time and outcome for each test case. These simulations identify potential weaknesses and determine how each algorithm can withstand unauthorized decryption

attempts. Specifically, the study examines whether specific patterns in the encryption process can be exploited to accelerate decryption without the proper key.

The fourth stage is Performance Analysis of Camellia and AES from both efficiency and robustness perspectives. Efficiency was measured using run time, CPU usage, and RAM consumption during encryption and decryption. Robustness was evaluated based on whether encrypted outputs could be successfully compromised under the three simulated attack settings. The results provide a comprehensive view of each algorithm's capabilities in handling diverse data efficiently and securely under varying conditions.

The final stage is Evaluation and Conclusion based on all obtained results. In this stage, the study will summarize the strengths and weaknesses of each algorithm in terms of efficiency and security. The evaluation results will identify the most suitable algorithm for data protection purposes. These conclusions are expected to provide a solid foundation for selecting a cryptographic algorithm that aligns with specific security requirements.

3. RESULTS AND DISCUSSION

The data collection results comprise 72 files across eight file types, with three predefined size categories. Table 2 presents three sample files for each predefined size category, labeled File 1, File 2, and File 3, with actual sizes varying within a ±5% tolerance range. The use of three files per size category serves multiple purposes. It ensures consistent and reliable results within each size group, captures minor variations in file structure and content that may affect processing, and helps mitigate potential bias from relying on a single file.

Table 2 Files Distribution and Variation

Types	Predefined Sizes	Actual Sizes		
		File 1	File 2	File 3
*.mp3	100 KB	102 KB	103 KB	102 KB
	1 MB	1.00 MB	1.04 MB	1.00 MB
	10 MB	10.03 MB	10.03 MB	10.04 MB
*.jpg	100 KB	105 KB	100 KB	100 KB
	1 MB	1.05 MB	1.05 MB	1.00 MB
	10 MB	10.03 MB	10.01 MB	10.02 MB
*.png	100 KB	101 KB	101 KB	101 KB
	1 MB	1.01 MB	1.00 MB	1.01 MB
	10 MB	10.02 MB	10.02 MB	10.04 MB
*.pdf	100 KB	100 KB	100 KB	101 KB
	1 MB	1.04 MB	1.05 MB	1.00 MB
	10 MB	10.00 MB	10.01 MB	10.01 MB
*.docx	100 KB	100 KB	104 KB	105 KB
	1 MB	1.02 MB	1.01 MB	1.01 MB
	10 MB	10.01 MB	10.05 MB	10.04 MB
*.xls	100 KB	100 KB	104 KB	100 KB
	1 MB	1.00 MB	1.01 MB	1.01 MB
	10 MB	10.04 MB	10.02 MB	10.01 MB
*.pptx	100 KB	104 KB	105 KB	100 KB
	1 MB	1.00 MB	1.05 MB	1.05 MB
	10 MB	10.04 MB	10.04 MB	10.04 MB
*.txt	100 KB	101 KB	103 KB	100 KB
	1 MB	1.01 MB	1.00 MB	1.01 MB
	10 MB	10.05 MB	10.05 MB	10.02 MB

The encryption and decryption efficiency of Camellia and AES was evaluated using Run Time, CPU usage, and RAM consumption. Run Time refers to the time taken to complete the encryption or decryption process, CPU Usage indicates the percentage of processor resources used during

execution, and RAM Consumption reflects the amount of memory utilized throughout the operation. To ensure a representative comparison, the values for each metric were averaged per file type using data from three files in every predefined size category, since the evaluation results from File 1, File 2, and File 3 showed consistent values that indicate stable performance. Measurements across formats and sizes were also relatively uniform within each group, with no significant deviations, which supports the validity of the averaged values.

The encryption results are presented in Table 3, which shows the average Run Time, CPU Usage, and RAM Consumption for each file type and predefined size. To maintain unit consistency and simplify numerical comparison, file sizes were defined as 100 KB, 1024 KB, and 10240 KB instead of mixed units with 1MB and 10MB. These metrics illustrate how each algorithm's performance responds to increasing file sizes, offering a clearer view of their efficiency and resource usage during encryption.

Table 3 Summary of Encryption Process (Averaged)

Types	Predefined Sizes	Run Time (s)		CPU Usage (%)		RAM Usage (MB)	
		Camellia	AES	Camellia	AES	Camellia	AES
*.mp3	100 KB	260.61	75.83	22.37	17.00	24.72	13.49
	1024 KB	458.64	140.65	28.49	21.22	31.76	24.33
	10240 KB	646.16	217.75	32.11	27.59	34.81	26.66
*.jpg	100 KB	137.59	77.30	18.34	21.12	21.54	16.46
	1024 KB	349.24	147.29	20.64	24.28	24.30	22.83
	10240 KB	524.70	197.32	23.62	27.75	25.57	26.77
*.png	100 KB	104.53	93.44	17.93	20.80	22.87	17.96
	1024 KB	305.27	136.36	22.13	24.59	25.43	24.85
	10240 KB	479.15	203.79	24.32	29.04	27.87	27.64
*.pdf	100 KB	152.16	192.21	23.17	25.39	21.39	25.61
	1024 KB	200.60	256.51	25.11	31.47	26.35	32.83
	10240 KB	392.14	340.52	29.29	36.25	27.99	35.03
*.docx	100 KB	91.98	94.97	22.13	20.88	20.25	23.51
	1024 KB	156.01	149.13	24.37	26.25	23.41	26.67
	10240 KB	409.58	206.79	28.81	30.49	24.68	28.47
*.xls	100 KB	96.38	107.52	22.99	22.37	21.78	25.47
	1024 KB	234.36	259.62	24.08	27.21	23.46	29.20
	10240 KB	384.74	338.61	25.61	33.40	25.59	30.26
*.pptx	100 KB	111.67	95.38	25.04	21.73	25.88	23.38
	1024 KB	185.69	156.13	28.36	24.59	28.55	27.39
	10240 KB	278.82	209.68	30.46	28.76	31.51	30.61
*.txt	100 KB	94.74	88.50	22.57	21.35	25.46	23.23
	1024 KB	199.99	157.52	26.55	24.50	28.58	27.54
	10240 KB	325.68	234.29	31.80	30.16	30.72	31.59

Based on Table 3, Figure 3 presents the graphical visualization of encryption. Each graph uses a logarithmic scale on the x-axis to represent predefined file sizes of 100 KB, 1024 KB, and 10240 KB. This allows more precise observation of efficiency trends as file size increases. The graphs illustrate the efficiency differences between the two encryption algorithms across file types and sizes. In general, larger file sizes lead to increased run time, CPU usage, and RAM usage, with algorithm efficiency varying depending on the file types. Across most formats and sizes, AES shows lower run time (higher throughput), while Camellia tends to maintain more stable memory behavior. Importantly, the magnitude of the gap is not uniform: differences become more pronounced at larger sizes and vary by file structure, indicating that efficiency conclusions depend on workload composition rather than file size alone. Similarly, the decryption metrics for Run Time, CPU Usage, and RAM Consumption were recorded and averaged across the predefined size, as shown in Table 4. Refer to Table 4, Figure 4 illustrates the decryption efficiency of Camellia and AES. Quite similar to the encryption process, larger file sizes generally result in increased run

time, CPU usage, and RAM usage, with efficiency varying depending on both the encryption algorithm and the file types.

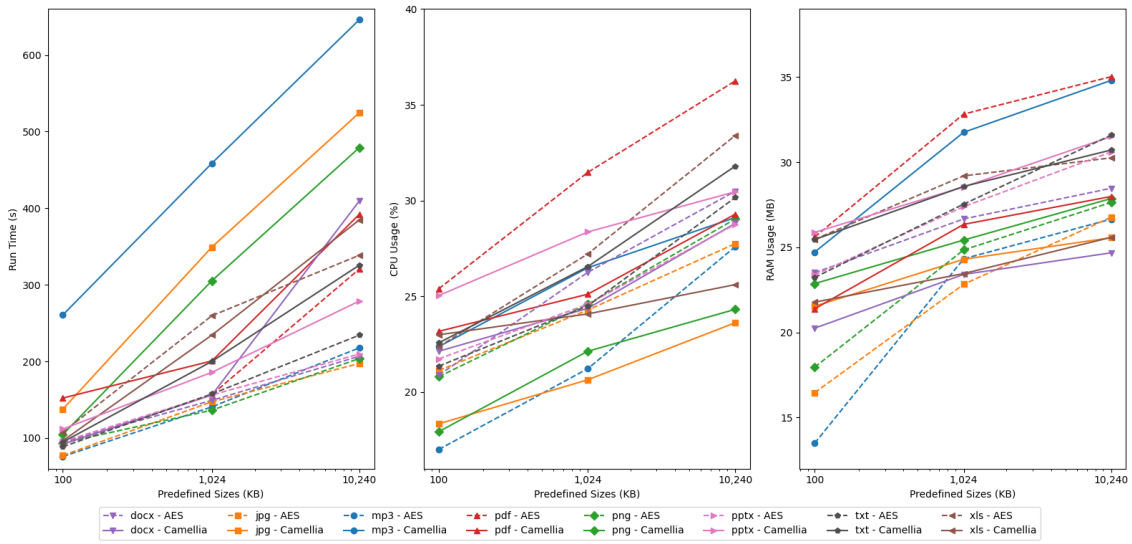

Figure 3 Visualization of Encryption Metrics

Table 4 Summary of Decryption Process (Averaged)

Types	Predefined Sizes	File 1		File 2		File 3	
		Camellia	AES	Camellia	AES	Camellia	AES
*.mp3	100 KB	282.39	78.89	24.82	23.50	29.72	21.49
	1024 KB	441.79	138.52	25.53	25.53	32.07	23.59
	10240 KB	630.44	268.60	27.50	27.51	33.57	24.48
*.jpg	100 KB	181.40	96.32	18.64	21.24	21.16	19.50
	1024 KB	240.79	154.13	21.17	23.44	24.41	23.44
	10240 KB	320.44	275.45	23.58	24.60	25.57	26.66
*.png	100 KB	121.40	73.74	28.31	21.22	23.16	23.58
	1024 KB	300.79	144.99	32.17	24.57	26.07	25.64
	10240 KB	530.44	208.12	34.59	26.29	26.90	26.64
*.pdf	100 KB	181.40	107.08	33.98	25.71	32.49	29.79
	1024 KB	340.79	325.69	35.17	29.50	35.74	32.22
	10240 KB	510.44	504.91	39.59	34.29	37.90	33.23
*.docx	100 KB	121.40	121.40	22.64	21.95	21.16	22.78
	1024 KB	260.79	219.12	24.17	25.55	23.74	24.73
	10240 KB	390.44	442.44	29.59	28.48	24.57	25.56
*.xls	100 KB	121.40	131.40	28.31	24.31	23.49	23.15
	1024 KB	230.79	277.45	30.50	28.84	24.74	24.07
	10240 KB	410.44	440.11	34.59	31.25	26.57	25.57
*.pptx	100 KB	131.40	131.40	24.98	24.31	26.16	23.49
	1024 KB	200.79	277.45	28.50	25.50	28.74	25.74
	10240 KB	330.44	440.11	30.59	29.59	31.56	27.90
*.txt	100 KB	121.40	131.40	24.64	28.64	23.16	28.49
	1024 KB	220.79	210.78	27.50	30.50	27.74	30.56
	10240 KB	410.44	330.44	31.92	32.59	30.57	32.74

Evaluating both encryption and decryption processes highlights consistent performance differences between Camellia and AES across various file types and predefined sizes (100 KB, 1024 KB, and 10240 KB). In the encryption phase, AES consistently achieved faster run times. For example, encrypting a 10 MB *.mp3 file took 217.75 seconds with AES, compared to 646.16

seconds with Camellia—nearly three times longer. Similar trends appeared in *.pdf files, where AES processed the largest size in 340.52 seconds, while Camellia required 392.14 seconds. However, this higher speed was often accompanied by increased memory usage. Encrypting a 10 MB *.pdf file consumed 35.03 MB of RAM with AES, while Camellia used only 27.99 MB. Similar patterns were found in other formats, such as *.xls and *.jpg, where AES consistently required more memory during encryption. Camellia generally showed higher processor load for CPU utilization during encryption, especially with larger files. For instance, processing a 10 MB *.png file required 24.32% CPU with Camellia and 29.04% with AES. This pattern, however, varied depending on the file format. With *.docx files, AES demonstrated slightly better CPU efficiency at 30.49%, compared to Camellia's 28.81%. In the decryption phase, AES continued to show faster processing times. Decrypting a 10 MB *.png file took 208.12 seconds with AES, while Camellia required 530.44 seconds. In terms of RAM usage, Camellia was generally more efficient. For example, decrypting a 10 MB *.xls file used 25.57 MB of memory with AES and 26.57 MB with Camellia—a relatively small difference. On the other hand, AES consumed less memory than Camellia when decrypting *.pdf files of the same size, recording 33.23 MB versus 37.90 MB. These differences suggest that file size and type influence memory usage and how each algorithm handles data internally.

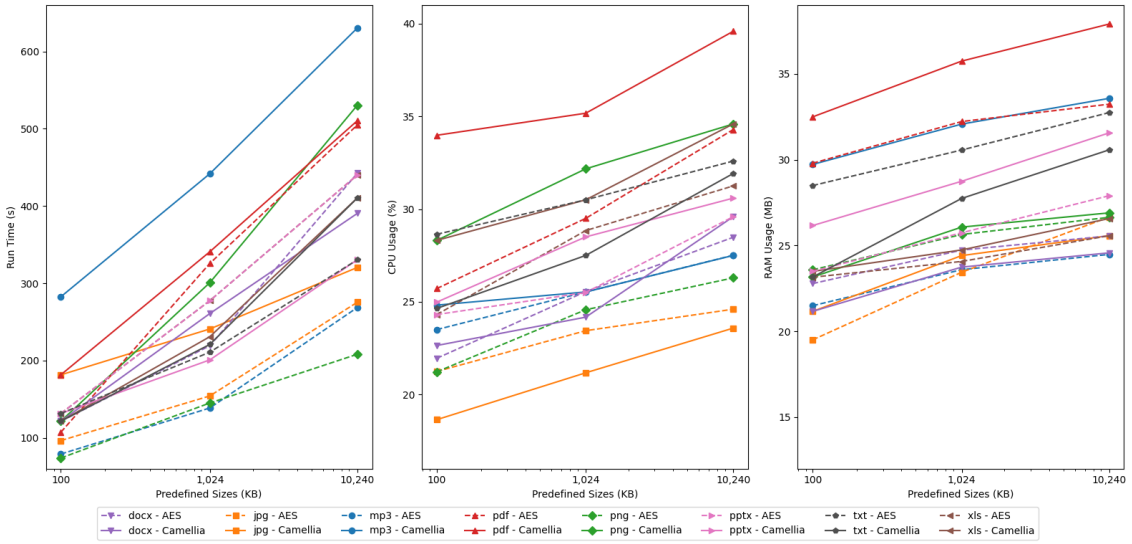

Figure 4 Visualization of Decryption Metrics

During the attack simulation phase, the outcomes varied depending on the file type, encryption algorithm, and specific method. Table 5 presents the results of the Dictionary Attack Test for both Camellia and AES. Each entry includes the Attack Time (AT) in seconds and the Attack Outcome (AO), where 'R' indicates the attack was resisted and 'B' signifies a successful breach. Shows that Camellia consistently resisted all dictionary attack attempts across various file types and sizes, with no violations observed in any scenario. In contrast, AES demonstrated vulnerabilities in specific formats, particularly in *.pdf and *.docx files at 100 KB and 1024 KB. Attack times (AT) in this table varied significantly depending on file size and type; for example, smaller files like *.pptx at 100 KB showed the shortest durations, with Camellia completing attacks in 306–326 seconds and AES in 781–801 seconds. Conversely, larger files, such as *.pdf and *.png at 10240 KB, required over 30,000 seconds for AES, whereas Camellia consistently remained under that threshold. These variations suggest that both file size and internal structure—such as compressibility or embedded metadata—play a key role in determining the efficiency of dictionary-based attacks. Figure 5 visualizes the average attack times from Table 5, offering a more precise comparison between the two algorithms. Averaging was considered valid and representative because the three attack time values for each test case were closely aligned. Moreover, the outcomes across those files were consistent, with each case resulting in complete resistance (3R) or complete breach (3B), indicating uniform behavior within each test condition. The 3B pattern,

as indicated by the '0R 3B' (zero file resisted and three files breached) outcomes, appeared exclusively in AES for *.pdf and *.docx files at 100 KB and 1024 KB, highlighting an apparent vulnerability when handling structured document formats. In contrast, Camellia maintained a consistent 3R outcome across all file types and sizes, reinforcing its resistance to dictionary attacks and efficiency across varying file characteristics.

Table 5 Dictionary Attack Results on Camellia and AES

Types	Predefined Sizes	File 1				File 2				File 3			
		Camellia		AES		Camellia		AES		Camellia		AES	
		AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO
*.mp3	100 KB	2657	R	1587	R	2677	R	1617	R	2637	R	1577	R
	1024 KB	32751	R	10002	R	2765	R	10042	R	2735	R	9972	R
	10240 KB	14808	R	32801	R	14828	R	33051	R	14798	R	33051	R
*.jpg	100 KB	726	R	426	R	736	R	436	R	726	R	416	R
	1024 KB	1739	R	9957	R	1759	R	9917	R	1719	R	10117	R
	10240 KB	21019	R	11624	R	21039	R	11724	R	20999	R	11574	R
*.png	100 KB	3648	R	15301	R	3668	R	15341	R	3628	R	15281	R
	1024 KB	23552	R	17269	R	23582	R	17319	R	23532	R	17219	R
	10240 KB	29844	R	26569	R	29794	R	26656	R	29914	R	26556	R
*.pdf	100 KB	3160	R	8934	B	3180	R	8954	B	3140	R	8914	B
	1024 KB	3616	R	27722	B	3636	R	27692	B	3596	R	27772	B
	10240 KB	25445	R	30757	R	25475	R	30787	R	25425	R	30767	R
*.docx	100 KB	639	R	7200	B	659	R	7230	B	619	R	7180	B
	1024 KB	13063	R	26408	B	13083	R	26438	B	13043	R	26378	B
	10240 KB	27309	R	52224	R	27329	R	52194	R	27289	R	52294	R
*.xls	100 KB	4900	R	8757	R	4920	R	8787	R	4880	R	8727	R
	1024 KB	22510	R	10099	R	22540	R	10069	R	22470	R	10139	R
	10240 KB	24259	R	11160	R	24229	R	11190	R	24309	R	11130	R
*.pptx	100 KB	306	R	791	R	326	R	801	R	306	R	781	R
	1024 KB	9553	R	11150	R	9573	R	11180	R	9533	R	11130	R
	10240 KB	27111	R	16929	R	27131	R	16959	R	27091	R	16899	R
*.txt	100 KB	6858	R	10058	R	6878	R	10088	R	6838	R	10028	R
	1024 KB	8185	R	10482	R	8195	R	10512	R	8165	R	10482	R
	10240 KB	10426	R	16492	R	10446	R	16522	R	10406	R	16472	R

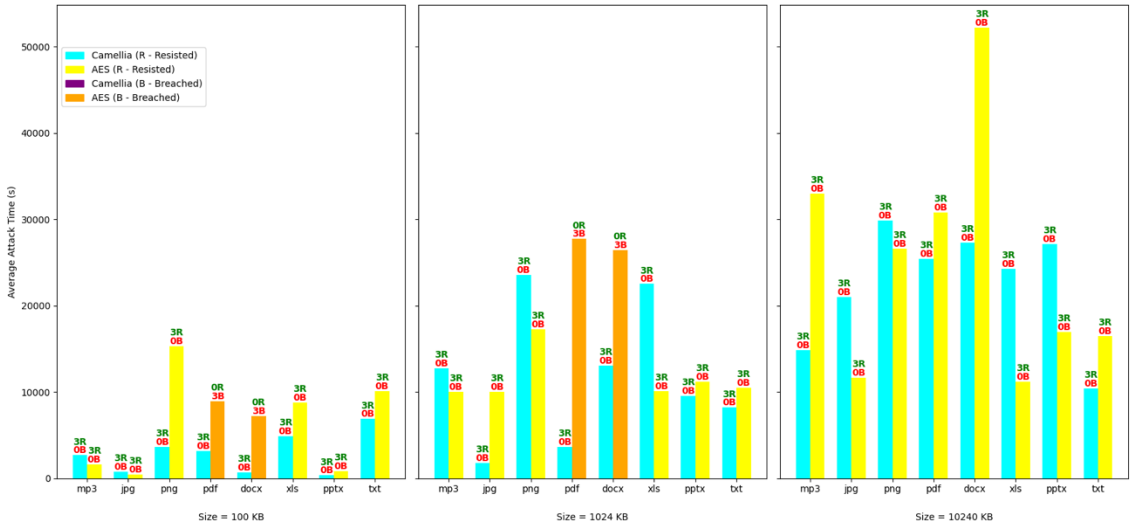

Figure 5 Visualization of Dictionary Attack

Table 6 indicates that Camellia consistently resisted all Birthday Attack attempts across every file type and size, with no breaches observed. In contrast, AES was breached in several cases, particularly in *.mp3, *.jpg, *.png, *.pdf, and *.docx at 100 KB and *.docx at 1024 KB. Attack times varied depending on file type and size—smaller files like *.jpg and *.mp3 at 100 KB had shorter durations (around 6100–11600 seconds in AES), while larger files such as *.docx and *.txt at

10240 KB required over 25,000 seconds in AES and over 24,000 seconds in Camellia. These findings indicate that while Camellia maintained complete resistance, file characteristics and algorithm behavior under collision-based attacks still influenced the computational effort involved. Figure 6 further illustrates this comparison by visualizing the average attack times between AES and Camellia, calculated from three predefined sizes of consistent test repetitions. The data confirms that AES was breached (3B) on several smaller files—specifically *.mp3, *.pdf, *.docx, and *.txt at 100 KB, as well as *.docx at 1024 KB—while Camellia maintained complete resistance (3R) across all tested files. Interestingly, even when both algorithms resisted attacks, AES consistently required longer durations, with the gap becoming most prominent at 10240 KB, suggesting that Camellia offers consistent resistance and performs more efficiently across increasing file sizes.

Table 6 Result of the Birthday Attack

Types	Predefined Sizes	File 1				File 2				File 3			
		Camellia		AES		Camellia		AES		Camellia		AES	
		AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO
*.mp3	100 KB	3616	R	11624	B	3711	R	11634	B	3602	R	11604	B
	1024 KB	6858	R	15301	R	6768	R	15331	R	6804	R	15281	R
	10240 KB	10426	R	17269	R	10403	R	17229	R	10503	R	17299	R
*.jpg	100 KB	4400	R	6132	B	4344	R	6102	B	4435	R	6162	B
	1024 KB	8757	R	18370	R	8816	R	18340	R	8803	R	18400	R
	10240 KB	21019	R	21307	R	20969	R	21337	R	20965	R	21277	R
*.png	100 KB	4900	R	8934	B	4890	R	8964	B	4993	R	8904	B
	1024 KB	11150	R	13500	R	11060	R	13530	R	11075	R	13490	R
	10240 KB	21247	R	13699	R	21206	R	13729	R	21208	R	13669	R
*.pdf	100 KB	5733	R	11150	B	5772	R	11180	B	5721	R	11120	B
	1024 KB	6858	R	12342	R	6792	R	12372	R	6926	R	12312	R
	10240 KB	8934	R	17272	R	9018	R	17302	R	8927	R	17242	R
*.docx	100 KB	13063	R	10058	B	13044	R	10088	B	13069	R	10028	B
	1024 KB	16492	R	21247	B	16449	R	21277	B	16475	R	21217	B
	10240 KB	17269	R	25608	R	17309	R	25638	R	17280	R	25578	R
*.xls	100 KB	10002	R	4486	R	10076	R	4516	R	9930	R	4456	R
	1024 KB	10099	R	21019	R	10172	R	21049	R	10027	R	20989	R
	10240 KB	15301	R	23552	R	15347	R	23582	R	15209	R	23522	R
*.pptx	100 KB	8185	R	3616	R	8283	R	3586	R	8096	R	3646	R
	1024 KB	10077	R	4400	R	10098	R	4430	R	10055	R	4370	R
	10240 KB	11624	R	21622	R	11655	R	21652	R	11697	R	21592	R
*.txt	100 KB	4486	R	4900	R	4414	R	4930	R	4386	R	4870	R
	1024 KB	9553	R	13063	R	9457	R	13093	R	9590	R	13033	R
	10240 KB	24259	R	22510	R	24292	R	22540	R	24208	R	22480	R

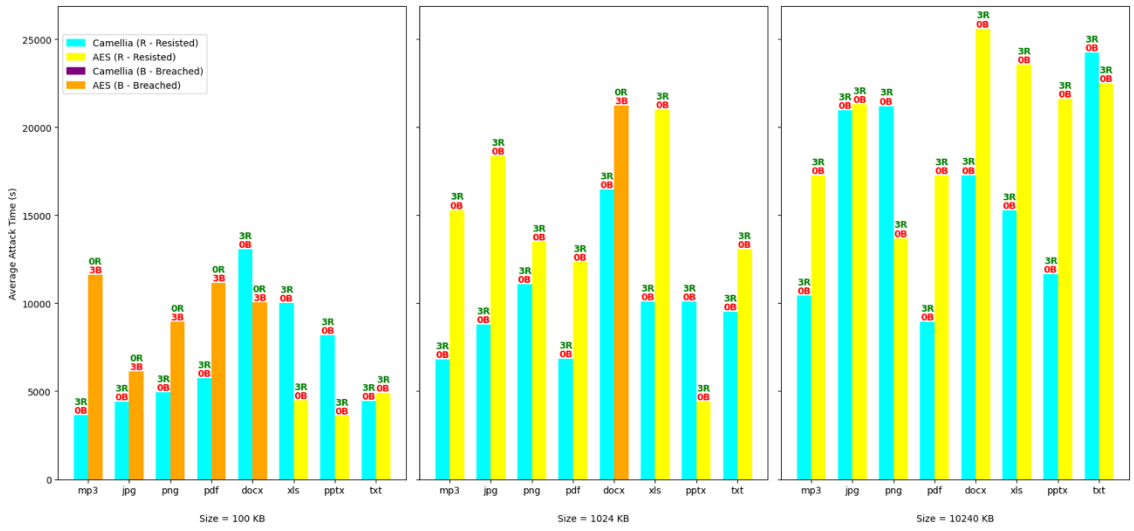

Figure 6 Visualization of Birthday Attack

Table 7 reveals a clear performance contrast between the two algorithms, particularly at the 100 KB level. AES consistently required longer attack durations than Camellia, especially for files like *.docx, *.jpg, and *.xls, which surpassed 18,000 seconds. Camellia maintained lower attack times and more consistent results across all file types and sizes. Both algorithms exhibited longer durations as file sizes increased to 10240 KB, though AES remained slower overall. These patterns suggest that Camellia handled brute-force attempts more efficiently, while certain 100 KB files appeared more susceptible, especially under AES.

Table 7 Brute-Force Attack Test Results

Types	Predefined Sizes	File 1				File 2				File 3			
		Camellia		AES		Camellia		AES		Camellia		AES	
		AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO	AT (s)	AO
*.mp3	100 KB	3943	R	8340	B	3900	R	8344	B	3843	R	8331	B
	1024 KB	11580	R	15720	R	11573	R	15604	R	11653	R	15663	R
	10240 KB	12960	R	16380	R	12983	R	16272	R	12892	R	16364	R
*.jpg	100 KB	6960	R	10260	B	6910	R	10373	B	6947	R	10111	B
	1024 KB	18360	R	14160	B	18302	R	14038	B	18361	R	14307	B
	10240 KB	19980	R	14760	R	19922	R	14746	R	20045	R	14858	R
*.png	100 KB	4860	R	5520	B	4886	R	5416	B	4805	R	5518	B
	1024 KB	9720	R	7440	R	9698	R	7538	R	9743	R	7350	R
	10240 KB	12060	R	17580	R	12057	R	17508	R	12099	R	17596	R
*.pdf	100 KB	7260	R	8220	B	7231	R	8188	B	7288	R	8303	B
	1024 KB	13380	R	11520	R	13356	R	11468	R	13346	R	11554	R
	10240 KB	13620	R	19680	R	13611	R	19670	R	13631	R	19812	R
*.docx	100 KB	5340	R	18240	B	5305	R	18210	B	5300	R	18296	B
	1024 KB	5460	R	20880	R	5471	R	20902	R	5479	R	20902	R
	10240 KB	10680	R	21022	R	10739	R	21600	R	10713	R	21517	R
*.xls	100 KB	4260	R	8760	B	4300	R	8727	B	4233	R	8718	B
	1024 KB	10320	R	21000	R	10226	R	21010	R	10299	R	21012	R
	10240 KB	15000	R	21420	R	15005	R	21336	R	15039	R	21437	R
*.pptx	100 KB	6120	R	4740	B	6031	R	4590	B	6101	R	4844	B
	1024 KB	7080	R	6000	R	7045	R	6105	R	7063	R	6024	R
	10240 KB	20520	R	18780	R	20462	R	18788	R	20478	R	18697	R
*.txt	100 KB	3660	B	4080	B	3661	B	4193	B	3595	B	4170	B
	1024 KB	8460	R	8040	R	8458	R	7911	R	8507	R	7945	R
	10240 KB	17760	R	14640	R	17782	R	14564	R	17712	R	14782	R

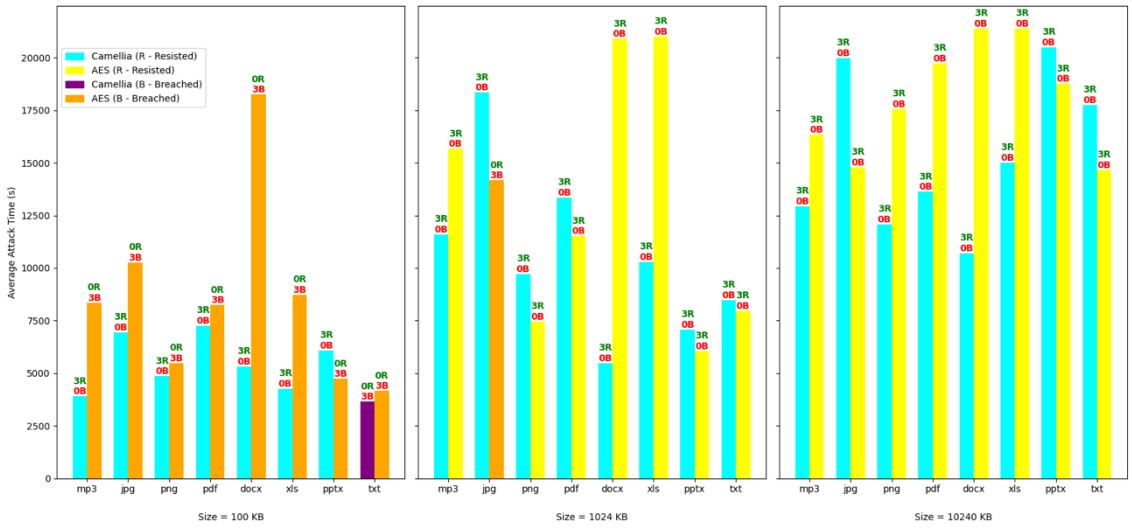

Figure 7 Visualization of Brute-Force Attack

Figure 7 presents the average brute-force attack times for AES and Camellia across various file types and sizes, based on consistent results from three predefined files per type and size in Table 7. At 100 KB, AES was breached in all file types (0R 3B), including .jpg, while Camellia showed

stronger resistance with 3R 0B—except for the .txt file, which recorded 0R 3B. Both algorithms fully resisted all attacks (3R 0B across the board) for larger sizes, with attack times increasing along with file size. These results suggest higher resilience at larger sizes and highlight a specific vulnerability in Camellia for small .txt files.

Importantly, none of these outcomes resulted from timeouts or execution limits. The breaches occurred due to actual key discovery during simulation, not from premature halting or insufficient time allocation, which confirms that algorithmic resilience—not processing duration—was the determining factor. Table 8 presents the total number of Resisted (R) and Breached (B) cases for each algorithm across all file types and attack types, summarizing the outcomes from all attack simulations. The attack simulation phase highlights distinct contrasts in the security performance of Camellia and AES across three common cryptographic threats. In the Dictionary Attack, Camellia consistently resisted all attempts across all file types and sizes. In contrast, AES was breached in specific cases—particularly for *.pdf and *.docx files at 100 KB and 1024 KB—suggesting that the key derivation or encryption patterns in AES may have been more susceptible to wordlist-based guessing in smaller or structured documents. In the Birthday Attack, which targets hash collisions and probabilistic vulnerabilities, AES again exhibited more weaknesses than Camellia. Successful breaches were recorded in AES for file types like *.mp3, *.jpg, *.png, *.pdf, and *.docx, mainly at smaller sizes. Meanwhile, Camellia resisted every test, indicating stronger protection against entropy-based collision exploitation. The Brute-Force Attack produced different observations. AES failed in all 100 KB file types tested, showing total vulnerability at smaller sizes. While more resistant overall, Camellia experienced a breach only in the 100 KB *.txt file. This suggests that straightforward file structures may reduce the number of key permutations needed to match the plaintext, making brute-force attempts more effective. These outcomes should be interpreted as empirical behavior under the defined attack configuration. The observed breaches highlight that robustness can differ across file structures at smaller sizes, reinforcing that algorithm choice should be aligned with the expected data profile and the system's threat assumptions, not solely with raw encryption speed.

Table 8 Summary of Attack Outcomes

Types	Predefined Sizes	Dictionary Attack				Birthday Attack				Brute-Force Attack				Total	
		Camellia		AES		Camellia		AES		Camellia		AES			
		B	R	B	R	B	R	B	R	B	R	B	R	B	R
*.mp3	100 KB	0	3	0	3	0	3	3	0	0	3	3	0	6	12
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.jpg	100 KB	0	3	0	3	0	3	3	0	0	3	3	0	6	12
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.png	100 KB	0	3	0	3	0	3	3	0	0	3	3	0	6	12
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.pdf	100 KB	0	3	3	0	0	3	3	0	0	3	3	0	9	9
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.docx	100 KB	0	3	3	0	0	3	3	0	0	3	3	0	9	9
	1024 KB	0	3	3	0	0	3	3	0	0	3	0	3	6	12
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.xls	100 KB	0	3	0	3	0	3	0	3	0	3	3	0	3	15
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.pptx	100 KB	0	3	0	3	0	3	0	3	0	3	3	0	3	15
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
*.txt	100 KB	0	3	0	3	0	3	0	3	3	0	3	0	6	12
	1024 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
	10240 KB	0	3	0	3	0	3	0	3	0	3	0	3	0	18
Total		**0**	**72**	**9**	**63**	**0**	**72**	**18**	**54**	**3**	**69**	**24**	**48**	**54**	**378**

The performance comparison between Camellia and AES reveals distinct characteristics in terms of both computational efficiency and cryptographic robustness. From the encryption and

decryption phases, AES consistently demonstrated superior speed, completing tasks in significantly shorter run times than Camellia across all file types and sizes. This advantage is especially apparent in larger files, where AES achieved nearly threefold faster processing, making it more suitable for time-sensitive applications. However, this speed benefit comes at the cost of higher memory usage. AES generally consumes more RAM during encryption, particularly for complex file types such as *.pdf and *.xls. In contrast, Camellia maintained more stable and lower RAM consumption, suggesting its advantage in memory-constrained environments. CPU utilization, on the other hand, varied depending on file type. However, Camellia often exhibited slightly higher processor loads during encryption, while AES showed more efficient CPU use in specific formats like *.docx. Regarding robustness, the attack simulation phase highlighted AES's relative weaknesses compared to Camellia. Camellia resisted all attacks in the Dictionary and Birthday categories and was only breached in one case during Brute-Force testing (the 100 KB *.txt file). AES, meanwhile, recorded multiple breaches, particularly in smaller file sizes. Notably, AES failed to withstand Dictionary and Birthday Attacks in formats like *.pdf and *.docx at 100 KB and 1024 KB, and was breached across all file types in the Brute-Force Attack at the smallest size. Despite its speed, these findings suggest that AES may exhibit reduced resistance against specific cryptanalytic techniques in small or structured file types. Conversely, Camellia provides more consistent security, particularly at smaller sizes, although it is marginally slower.

The variations observed across run time, CPU usage, and memory consumption confirm that the performance characteristics of both algorithms are highly dependent on file type and workload scale. As file size increases, the differences between AES and Camellia become more pronounced, offering valuable insight into their suitability for various system requirements and processing environments. The attack simulations highlight significant disparities in how each algorithm responds to cryptographic threats. Resistance levels varied across Dictionary, Birthday, and Brute-Force attack scenarios based on file structure and size. While AES exhibited multiple breaches—especially on smaller and structured files—Camellia consistently demonstrated stronger resilience in most cases, indicating more robust default protection in diverse conditions. Camellia and AES are evaluated based on the combined performance efficiency and cryptographic robustness assessment. Rather than favoring one algorithm absolutely, the findings underscore the importance of aligning algorithm selection with system priorities. One algorithm may offer operational advantages for applications requiring rapid data handling, while for scenarios prioritizing security under sustained attack, another may provide more consistent resilience. The results highlight that performance and security trade-offs are not uniform across file types or sizes, emphasizing the need for contextual decision-making in cryptographic implementations. Additionally, isolated vulnerabilities in specific structured files suggest that further refinement or hybrid encryption strategies may be required in practice.

Practical implications and system design recommendations. The findings suggest several actionable guidelines for system architects: for throughput-prioritized workflows dominated by large files and time-sensitive processing, AES is a practical choice due to consistently lower run time; for memory-constrained environments where RAM budgets are tight or multi-process encryption is expected, Camellia may be preferable because it shows more stable memory behavior across workloads; for mixed-format repositories handling heterogeneous document formats (e.g., office and PDF files), selection should consider not only speed but also empirical robustness under the assumed attack model, particularly for smaller structured files; for security-first deployments where the threat model includes sustained password-guessing attempts, system designers should prioritize secure configurations and operational controls (strong credential policies, rate limiting, and key management hardening) and consider Camellia when results indicate more consistent resistance; and rather than enforcing a single algorithm globally, a policy-based selection approach can be implemented to choose encryption based on file size/type and system constraints (time budget versus memory budget versus threat level).

4. CONCLUSIONS

This study presented a comparative performance analysis of Camellia and AES across various file types and sizes. AES consistently achieved faster encryption and decryption times, especially

on large files, making it suitable for time-sensitive applications. However, it consumed more memory and was more vulnerable to cryptanalytic attacks. Camellia exhibited more stable resource usage and greater resistance to attacks, experiencing only one breach during brute-force testing on a small plaintext file. At the same time, AES recorded multiple breaches in both dictionary and birthday attacks. Therefore, algorithm selection should be based on system constraints and required security levels. AES is recommended for high-speed environments, while Camellia is more appropriate for systems prioritizing security and resource efficiency. Overall, this study emphasizes that performance and robustness trade-offs are workload-dependent and can vary by file structure and size under practical attack conditions. By providing a controlled multi-format dataset and a unified evaluation of efficiency and empirical robustness, this paper supports more defensible, context-aware encryption decisions in real-world system design. Future research may include testing additional algorithms, expanding file variations, and evaluating performance under real-time operational conditions.

REFERENCES

A, H. M., S, F. M., & G, E. J. (2022). Implementation of Password Hashing on Embedded Systems with Cryptographic Acceleration Unit. *International Journal of Advanced Computer Science and Applications*, *13*(2), 171–175. https://doi.org/10.14569/IJACSA.2022.0130221

Adams, C. (2021). Dictionary Attack. In *Encyclopedia of Cryptography, Security and Privacy* (pp. 1–2). Springer Berlin Heidelberg. https://doi.org/10.1007/978-3-642-27739-9_74-2

Alkhwaja, I., Albugami, M., Alkhwaja, A., Alghamdi, M., Abahussain, H., Alfawaz, F., Almurayh, A., & Min-Allah, N. (2023). Password Cracking with Brute Force Algorithm and Dictionary Attack Using Parallel Programming. *Applied Sciences*, *13*(10), 5979. https://doi.org/10.3390/app13105979

Andriyanto, M. R., & Sukmasetya, P. (2022). Penerapan Algoritma Advanced Encryption Standard (AES) untuk Keamanan Data Transaksi pada Sistem E-Marketplace. *Journal of Computer System and Informatics (JoSYC)*, *4*(1), 179–187. https://doi.org/10.47065/josyc.v4i1.2451

Arabul, E., Oliveira, R. D., Emami, A., Typos, S., Vrontos, C., Wang, R., Nejabati, R., & Simeonidou, D. (2023). 100 Gbps Quantum-Secured And O-RAN-Enabled Programmable Optical Transport Network For 5G Fronthaul. *Journal of Optical Communications and Networking*, *15*(8), C223–C231. https://doi.org/10.1364/JOCN.483644

Azhari, M., Mulyana, D. I., Perwitosari, F. J., & Ali, F. (2022). Implementasi Pengamanan Data pada Dokumen Menggunakan Algoritma Kriptografi Advanced Encryption Standard (AES). *Jurnal Pendidikan Sains dan Komputer*, *2*(01), 163–171. https://doi.org/10.47709/jpsk.v2i01.1390

Bibiola, F., Kalsum, T. U., & Alamsyah, H. (2023). Penerapan Algoritma Advance Encryption Standard (AES) untuk Pengamanan File pada Aplikasi Berbasis WEB. *Jurnal Surya Energy*, *8*(1), 35. https://doi.org/10.32502/jse.v8i1.6461

Chavan, R., Gulge, A., & Bhandare, S. (2024). Moving Object Detection and Classification Using Deep Learning Techniques. *Tuijin Jishu/Journal of Propulsion Technology*, *45*(1), 1001–4055. https://www.propulsiontechjournal.com/index.php/journal/article/view/5000

Dwina, N., Khairani, N., Nasir, M., & Indrawati, I. (2023). Penerapan Metode Advanced Encryption Standard pada Sistem Penyimpanan Data Menggunakan Cloud Computing Sebagai Software-as-a-Service. *Journal of Artificial Intelligence and Software Engineering (J-AISE)*, *3*(1), 25–29. https://doi.org/10.30811/jaise.v3i1.4183

Fahdi, H. Al, & Ahmed, M. (2020). Cryptographic Attacks, Impacts and Countermeasures. In *Security Analytics for the Internet of Everything* (pp. 215–230). CRC Press. https://doi.org/10.1201/9781003010463-13

G, E. J., A, H. M., & S, F. H. M. (2022). Enhanced Security: Implementation of Hybrid Image Steganography Technique Using Low-Contrast LSB and AES-CBC Cryptography. *International Journal of Advanced Computer Science and Applications*, *13*(8), 899–905. https://doi.org/10.14569/IJACSA.2022.01308104

Hamza, A. A., & Al-Janabi, R. J. surayh. (2024). Detecting Brute Force Attacks Using Machine Learning. *BIO Web of Conferences*, *97*, Article ID: 00045. https://doi.org/10.1051/bioconf/20249700045

Haris, M., Lydia, M. S., & Sutarman, S. (2023). Pengamanan pada Citra Digital dengan Menggunakan Modifikasi Blok Data Algoritma AES - Rijndael. *Jurnal Media Informatika Budidarma*, *7*(1), 444–453. https://doi.org/10.30865/mib.v7i1.5458

Hidayati, L. N., Fitriana, G. F., & Adam, I. F. (2021). Perbandingan Keacakan Citra Enkripsi Algoritma AES dan Camelia Uji NPCR dan UACI. *JURIKOM (Jurnal Riset Komputer)*, *8*(6), 274–283. https://doi.org/10.30865/jurikom.v8i6.3624

Housley, R., & Schaad, J. (2020). *RFC 3394 - Advanced Encryption Standard (AES) Key Wrap Algorithm*. IETF Datatracker. https://datatracker.ietf.org/doc/rfc3394/

Hranický, R., Šírová, L., & Rucký, V. (2025). Beyond The Dictionary Attack: Enhancing Password Cracking Efficiency Through Machine Learning-Induced Mangling Rules. *Forensic Science International: Digital Investigation*, *52*, Article ID: 301865. https://doi.org/10.1016/j.fsidi.2025.301865

Kato, A., & Moriai, S. (2013). *RFC 3657 - Use of the Camellia Encryption Algorithm in Cryptographic Message Syntax (CMS)*. IETF Datatracker. https://datatracker.ietf.org/doc/rfc3657/

Li, Y., Wang, Q., Huang, D., Liu, J., & Xie, H. (2024). Quantum Chosen-Cipher Attack on Camellia. *Cryptology EPrint Archive, Paper 2024/1804*. https://eprint.iacr.org/2024/1804

Manullang, S. (2023). Implementasi Kriptografi Pengamanan Data File Dokumen Menggunakan Algoritma Advanced Encryption Standard Mode Chiper Block Chaining. *Jurnal Nasional Teknologi Komputer*, *3*(2), 87–95. https://doi.org/10.61306/jnastek.v3i2.69

Matsui, M., Moriai, S., & Nakajima, J. (2015). *RFC 3713 - A Description of the Camellia Encryption Algorithm*. IETF Datatracker. https://datatracker.ietf.org/doc/rfc3713/

Mohammed, A. A., Rahma, A. M. S., & Abdul Wahab, H. B. (2025). Security Encryption Model Of E-Bank Based Camellia Algorithm Using Cubic Curve. *AIP Conference Proceedings*, *3264*(1), Article ID: 030030. https://doi.org/10.1063/5.0263610

Muthavhine, K. D., & Sumbwanyambe, M. (2023). Blocking Linear Cryptanalysis Attacks Found on Cryptographic Algorithms Used on Internet of Thing Based on the Novel Approaches of Using Galois Field (GF (232)) and High Irreducible Polynomials. *Applied Sciences*, *13*(23), Article ID: 12834. https://doi.org/10.3390/app132312834

Ning, L., Ali, Y., Ke, H., Nazir, S., & Huanli, Z. (2020). A Hybrid MCDM Approach of Selecting Lightweight Cryptographic Cipher Based on ISO and NIST Lightweight Cryptography Security Requirements for Internet of Health Things. *IEEE Access*, *8*, 220165–220187. https://doi.org/10.1109/ACCESS.2020.3041327

Nithishma, A., Vijayan, A., Nanda, D., Kumar, Ch. A., Thaseen, S., & Ahmad, A. (2022). Remote User Authentication Using Camellia Encryption for Network-Based Applications. In *Security and Privacy-Preserving Techniques in Wireless Robotics* (pp. 255–279). CRC Press. https://doi.org/10.1201/9781003156406-19

Panahi, U., & Bayılmış, C. (2023). Enabling Secure Data Transmission For Wireless Sensor Networks Based Iot Applications. *Ain Shams Engineering Journal*, *14*(2), Article ID: 101866. https://doi.org/10.1016/j.asej.2022.101866

Patra, R., & Patra, S. (2021). Cryptography: A Quantitative Analysis of the Effectiveness of Various Password Storage Techniques. *Journal of Student Research*, *10*(3). https://doi.org/10.47611/jsrhs.v10i3.1764

Rachmayanti, A., & Wirawan, W. (2022). Implementasi Algoritma Advanced Encryption Standard (AES) pada Jaringan Internet of Things (IoT) untuk Mendukung Smart Healthcare. *Jurnal Teknik ITS*, *11*(3). https://doi.org/10.12962/j23373539.v11i3.97042

Rasheed, A., Baza, M., Badr, Mahmoud. M., Alshahrani, H., & Choo, K.-K. R. (2024). Efficient Crypto Engine for Authenticated Encryption, Data Traceability, and Replay Attack Detection Over CAN Bus Network. *IEEE Transactions on Network Science and Engineering*, *11*(1), 1008–1025. https://doi.org/10.1109/TNSE.2023.3312545

Rasheed, A., Baza, M., Khan, M., Karpoor, N., Varol, C., & Srivastava, G. (2023). Using Authenticated Encryption for Securing Controller Area Networks in Autonomous Mobile Platforms. *2023 26th International Symposium on Wireless Personal Multimedia Communications (WPMC)*, 76–82. https://doi.org/10.1109/WPMC59531.2023.10338834

Rashidi, B. (2021). Flexible And High-Throughput Structures Of Camellia Block Cipher For Security of the Internet of Things. *IET Computers & Digital Techniques*, *15*(3), 171–184. https://doi.org/10.1049/cdt2.12025

Sarmila, K. B., & Manisekaran, S. V. (2022). Honey Encryption and AES based Data Protection Against Brute Force Attack. *2022 Sixth International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud) (I-SMAC)*, 187–190. https://doi.org/10.1109/I-SMAC55078.2022.9987304

Sulaiman, A. S., & Hammood, M. M. (2025). Enhancing Data Security by Using Hybrid Encryption Technique Based on AES and Camellia. In *Learning and Analytics in Intelligent Systems* (Vol. 45, pp. 173–182). https://doi.org/10.1007/978-3-031-82706-8_18

Tanjung, P. Y. (2022). Penerapan Algoritma AES 625 dalam Pengamanan Data Rekam Medis. *Journal Global Technology Computer*, *1*(3), 77–83. https://doi.org/10.47065/jogtc.v1i3.2054

Verma, R., Dhanda, N., & Nagar, V. (2022). Enhancing Security with In-Depth Analysis of Brute-Force Attack on Secure Hashing Algorithms. In *Lecture Notes in Networks and Systems* (Vol. 376, pp. 513–522). https://doi.org/10.1007/978-981-16-8826-3_44

Wei, Z., Sun, S., Hu, L., Wei, M., & Peralta, R. (2023). Searching the Space of Tower Field Implementations of the $\mathbb{F}28$ Inverter - with Applications to AES, Camellia and SM4. *International Journal of Information and Computer Security*, *20*(1–2), 1–26. https://doi.org/10.1504/IJICS.2023.127999

ZhenQiang, L., Fei, G., SuJuan, Q., & QiaoYan, W. (2023). Quantum Circuit For Implementing Camellia S-Box With Low Costs. *SCIENTIA SINICA Physica, Mechanica & Astronomica*, *53*(4), Article ID: 240313. https://doi.org/10.1360/SSPMA-2022-0485