Accredited SINTA 2 Ranking
Decree of the Director General of Higher Education, Research, and Technology, No. 158/E/KPT/2021
Validity period from Volume 5 Number 2 of 2021 to Volume 10 Number 1 of 2026

Published online at: **http://jurnal.iaii.or.id**

JURNAL RESTI
(Rekayasa Sistem dan Teknologi Informasi)
Vol. 9 No. 2 (2025) 374 - 384 e-ISSN: 2580-0760

Securing Electronic Medical Documents Using AES and LZMA

Toto Raharjo[1]*, Yudi Prayudi[2]
[1,2]Informatics Master's Programme, Faculty of Industrial Technology, Universitas Islam Indonesia, Yogyakarta, Indonesia
[1]toto.raharjo@students.uii.ac.id, [2]prayudi@uii.ac.id

Abstract

With increasing threats in cyberspace, maintaining the integrity of electronic medical data is crucial. This study aims to develop a method that integrates encryption using Advanced Encryption Standard (AES) and compression with the Lempel-Ziv-Markov Algorithm (LZMA) to protect DICOM files containing sensitive information. This method is designed to address two main challenges: the growth of file sizes after the encryption process and the efficiency in data storage. In this study, an experimental design with random sampling was applied, testing 427 DICOM files from open libraries ranging in size from 513.06 KB to 513.39 KB to evaluate the implementation of this method in reducing file size, encryption time, and maintaining data integrity. The results show that this method is able to reduce file size by between 40-50% with an average encryption time of about 0.2-0.3 seconds per file. In addition, the data remains intact before and after the encryption process, which indicates that the integrity of the data is well maintained. Further analysis revealed that CPU usage during the encryption process reached 94.05%, while memory usage was recorded at 92.95 KB. In contrast, in the decryption process, CPU usage decreased to 78.16% with a much lower memory consumption, which was 31.07 KB. The findings have significant implications for medical information systems, allowing developers to easily implement these methods through APIs. This research is expected to be a reference for future studies that focus on data security in health information systems and provide new insights into the combination of encryption and compression in the context of medical data.

Keywords: AES Encryption; LZMA Compression; Electronic Medical Data; DICOM Files; Data Security

How to Cite: T. Raharjo and Yudi Prayudi, "Securing Electronic Medical Documents Using AES and LZMA", *J. RESTI (Rekayasa Sist. Teknol. Inf.)* , vol. 9, no. 2, pp. 374 - 384, Apr. 2025.
DOI: https://doi.org/10.29207/resti.v9i2.6260

1. Introduction

The security of medical data is becoming an increasingly pressing concern due to the rising use of electronic documents in hospitals and clinics. These electronic medical records contain sensitive information that necessitates robust data protection measures [1]. As the frequency of data breaches in the medical field continues to rise, it is essential to implement effective strategies to safeguard this information. In this context, the implementation of encryption and compression techniques emerges as a critical solution to enhance data security and optimize storage. This paper will explore the effectiveness of these techniques in safeguarding electronic medical records. Recent global incidents have highlighted the significant threats to the security of health information, underscoring the urgent need for comprehensive protective measures [2].

Therefore, previous research has focused on developing encryption systems that can address this threat. To address the challenges inherent in medical data security, numerous prior studies have contributed significantly to the advancement of encryption systems applicable to medical data. Among these is a study focused on developing a medical image encryption system employing AES encryption methods alongside Base64 encoding at the byte level. This approach has been demonstrated to protect medical image data from unauthorized access, yielding promising outcomes concerning both security and the velocity of the encryption process [3]. Another study focused on security applications for patient medical record data that are Android-based, utilizing cryptography with AES and Rivest Shamir Adleman (RSA) methods, which emphasizes the importance of protecting medical data on mobile devices and offers practical solutions to improve the security of patient information [4]. In addition, research on the application of data security to

Received: 26-12-2024 | Accepted: 12-04-2025 | Published Online: 19-04-2025

documents using the AES cryptographic algorithm shows that this algorithm can effectively protect sensitive data and provide insights into the application of cryptography in the field of education and research [5]. Furthermore, research on the application of the 512-bit Base64 algorithm as an innovation in encryption techniques to improve the security of patient data in hospitals [6]. While Base64 is typically employed for data encoding, this study investigates the possibility of utilizing these methods in data encryption contexts. This approach offers a viable alternative for developing health information systems, enhancing the safeguarding of sensitive patient data. The study endeavors to demonstrate that 512-bit Base64 can improve data protection and deliver a more adaptable and efficient solution in healthcare information [7] in the form of an Application Programming Interface (API) [8]. Lastly, research comparing the Chain Code proximity technique with the LZMA algorithm in compressing binary images offers valuable insights into data compression effectiveness, which is crucial for the management and storage of medical data, particularly for Digital Imaging and Communications in Medicine (DICOM) [9].

Numerous studies have significantly contributed by creating encryption systems for medical images using AES and Base64 encoding methods, which have effectively ensured the security of medical records. However, these methods result in substantially larger file sizes, posing a challenge for storage, especially in settings with limited capacity. Furthermore, another study comparing the Chain Code and LZMA for binary image compression prioritizes data compression efficiency but lacks in-depth exploration of security aspects. This shortcoming is particularly problematic for medical data, which necessitates strong protection of sensitive information.

To overcome these limitations, this study aims to address the challenges in medical data security by combining AES encryption and LZMA compression [10] as a novelty. Although existing encryption methods have proven to be a solution, they result in large file sizes, making it difficult to store and transmit data [11]. This study not only ensures the security of data but also achieves a reduction in file size, thereby offering a more optimal solution for the storage and management of health information, and corroborating prior research on Secure Communication for Digital Forensics [12].

Through this methodology, this research advances the creation of more practicable solutions for the storage and transmission of medical data, while also enhancing the adaptability in deploying cloud-based health information systems [13]. This is highly pertinent, given that recent data breach incidents clearly illustrate that threats to medical data security persist. For instance, in March 2023, approximately 18.5 million user records from BPJS Kesehatan were reported to have been illicitly sold on dark forums, with threats to disclose the information should a ransom not be paid [14]. The origin of this breach is attributed to hackers who allege unauthorized access to the BPJS Kesehatan database. The compromised data encompasses sensitive information, including full names, population identification numbers, addresses, dates of birth, along with health information and membership status. This incident raises significant concerns regarding the potential for identity theft and the misuse of personal data by unscrupulous entities. Furthermore, an additional breach affected nearly 4 million individuals associated with incidents at medical transcription companies. This breach reportedly took place between March 27, 2023, and May 2, 2023, impacting approximately 3,998,162 individuals. The source of this leak is identified as Perry Johnson & Associates, a provider of medical transcription services. Unauthorized entities succeeded in accessing their network and acquiring copies of certain files containing personal information of Concentra patients, including demographic and clinical data [15].

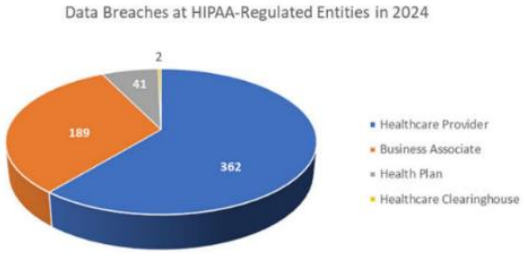

Figure 1. Electronic Document Data Leak

Recent statistics show that incidents like this are not isolated cases. Based on Figure 1 shows that information system providers are the sector that receives the highest number of data leak cases. The incident highlights the importance of protecting medical data and shows that health information leaks can have far-reaching impacts, including the risk of identity theft and loss of public trust in the health system [16]. This incident shows that many healthcare facilities do not have adequate security systems in place to protect patient data from cyber threats. These leaks not only threaten patient privacy, but also open up opportunities for information misuse for crimes such as identity theft and insurance fraud [17]. Efforts to improve the security of medical data are essential in maintaining the integrity and privacy of patient information. This case further emphasizes the need for a better security system, especially in dealing with the risk of data leakage in the digital world [18]. One of the efforts that can be made is to implement strong and implementable encryption methods to protect sensitive data [19].

2. Research Methods

This study adopts an experimental approach to evaluate the results of a combination of compression and encryption to secure and optimize the size of electronic medical data. The research process began with

encoding the data using the Base64 algorithm, which serves to convert binary data into a text format that is easier to handle and transfer. Furthermore, the encoded data is compressed using the LZMA algorithm, which is a lossless compression method [20], thereby achieving a substantial reduction in file size without any loss of information. Subsequent to the compression phase, the files generated undergo encryption utilizing the AES algorithm, which ensures an elevated level of security, thus safeguarding the data against unauthorized access. For the purpose of data retrieval, files in their encrypted state are subjected to decryption via the same AES algorithm, followed by decompression using the LZMA algorithm, to reinstate their original size and configuration in Base64 format. Ultimately, the file is reverted to its initial format, thereby ensuring that the data can be accessed with complete integrity.

This entire process is executed in a Docker environment [21] with 7th generation Intel Core hardware processors with 12 Central Processing Unit (CPU) threads and 16 GB Random Access Memory (RAM) that ensures isolation, consistency and scalability of results across multiple platforms. The developed modeling is executed using specific Docker commands. This command maps port 5000 on the host, i.e. the computer running Docker, to port 5000 inside the container. This means that when an application runs inside a container, users can access it through port 5000 on their computer. In addition, this command also maps the current working directory to the /app directory inside the container. This way, any changes made to the files in the local directory will be instantly reflected inside the container, allowing for more efficient and interactive development. The image used to run the application is totorajo/flask:3.9-auto-reload. This image is based on Flask, a popular web framework for building web applications with Python. The auto-reload feature provided by this image allows the app to automatically reload whenever there is a change to the code, so developers don't have to manually stop and restart the app every time they make a change. The study also measured four key parameters to assess the effectiveness of the combination of AES encryption methods and LZMA compression in electronic medical data processing. These parameters include file size, processing time, data integrity and security level. Using the Python programming language library[22] as the executor of the calculation.

Various Python libraries are employed to augment the functionality of this application. Flask operates as the principal web framework, offering the foundational structure for constructing web applications. The cryptography library version 41.0.3 is utilized for data encryption and decryption, a critical component in ensuring the security of information transmitted and stored within the application. The psutil library, version 5.9.5, is instrumental in monitoring and managing system processes, providing insights on resource utilization and system status. For data manipulation and analysis, the pandas library is implemented, providing significant utility in processing data in a tabular format, thus enabling complex analyses with proficiency. Furthermore, openpyxl is harnessed for handling Excel files, facilitating the reading and writing of data in prevalent spreadsheet formats. The PyYAML library, version 5.4.1, is applied for the processing of YAML files, which are frequently utilized for configuration and data storage. By using this library, applications can proficiently read and write the necessary configuration files. Lastly, b2sdk version 1.9.0 is employed to interface with the Backblaze B2 Cloud Storage service, allowing for the efficient storage and retrieval of data from cloud storage. Collectively, this configuration is meticulously designed to ensure the experimental environment's replicability. Through the utilization of Docker and select libraries, this research fortifies the validity and reliability of the results acquired, while also promoting further collaboration and advancement in the future.

The selection of AES and LZMA algorithm in this study is based on several important considerations. AES is a symmetric encryption algorithm that offers a high level of security with variable key lengths, making it more efficient than RSA [23], which is slower and less suitable for processing large amounts of data. LZMA is known for its high compression ratio and good speed [24], making it a more efficient option than gzip [25], which often results in a lower compression ratio. AES is a prominently acknowledged encryption algorithm utilized extensively in various data security scenarios [26]. Offering diverse key lengths such as 128, 192, or 256 bits, AES provides a robust level of defense, which is essential for safeguarding sensitive medical records like DICOM files with patient details. These methods are optimized for both hardware and software to ensure brisk encryption and decryption. Quick data access is crucial in medical contexts, where prompt responses are vital. Additionally, AES supports multiple operating modes, such as Galois/Counter Mode (GCM), Cipher Block Chaining (CBC), and Counter Mode (CTR), offering customization in encryption and integration with authentication, thereby boosting data protection. Meanwhile, LZMA excels in achieving high compression ratios, particularly advantageous for typically large DICOM files. By diminishing file sizes, LZMA aids in conserving storage and expediting data transfer, a key element of medical data administration. Although LZMA compression might be time-consuming, the decompression is rapid, facilitating swift data access necessary for diagnosis and treatment. Moreover, LZMA is adept at managing structured data, making it ideal for DICOM files that contain significant additional information alongside images. Both methods are suitable for medical data with complex structures and substantial sizes, where AES can encrypt in blocks and LZMA can compress efficiently. The synergy of AES and LZMA permits the concurrent attainment of

security and storage efficiency by pre-compressing data before encryption to shrink file size.

This research focuses on the processing of electronic medical data, especially medical images stored in DICOM format [27]. The data source used comes from the DICOM Library with open source for research and education. The library provides access to a variety of anonymous medical datasets [28] to maintain the confidentiality of patient information. The data taken includes various types of medical images such as Magnetic Resonance Imaging (MRI), Computed Tomography (CT) Scan, Ultrasound, and Radiography. All available datasets are stored in DICOM format.

The process of data acquisition is conducted through the downloading of a DICOM file. In this research, sampling was performed by acquiring DICOM files from the website 3dicomviewer.com, which provides 427 anonymized DICOM files for educational purposes. The particular file utilized in this analysis is a scan obtained from the Harvard University Dataverse, offering a distinctive 3D perspective of the impact of viral pneumonia on the patient's lungs. DICOM files are accessible and downloadable at no cost via the platform's provided button. The selection of this file is aimed at ensuring that the data employed in this study is relevant and representative for the analysis undertaken.

In this study, the selection of DICOM files was conducted with several criteria in mind, designed to ensure that the data used is pertinent and of high quality. The first criterion is clinical relevance, where the selected file must have a significant connection with the medical condition under investigation. In this context, a file elucidating the impact of viral pneumonia on the patient's lungs was selected, as it could facilitate a deeper comprehension of the disease and its ramifications for the patient's health.

The second criterion concerns data anonymity, wherein only anonymous files were deemed acceptable to uphold patient privacy and comply with ethical principles in this study. That is to ensure that its use is carried out in an ethical and non-misleading manner. The analysis and interpretation carried out must still respect the integrity and context of the existing data. DICOM files acquired from 3dicomviewer.com meet this criterion, as all patient identity information has been removed or anonymized. This action is essential to uphold the privacy rights of individuals and to ensure adherence to existing regulations.

Furthermore, the quality of images represents a critical factor in the selection process. DICOM files of high resolution and clarity are preferred to ensure that subsequent analyses produce precise and reliable results. High image quality is paramount in the field of medical analysis, where minute details can affect the interpretation of findings. Similarly, accessibility is crucial in the selection of files. The selected files must be readily accessible and available for download at no cost. By using freely accessible DICOM files, the study ensures the replication of results by future researchers, thereby enhancing transparency and collaboration in the research field.

Additionally, the diversity of scan types and medical conditions represented is also evaluated. Selecting files that encompass a variety of scan types and medical conditions is intended to provide a more comprehensive portrayal of the impact of pneumonia associated with viral infection. In consideration of these criteria, the DICOM files employed in this study are expected to contribute substantially to the understanding of medical data security and the development of encryption methods that efficiently conserve storage media capacity.

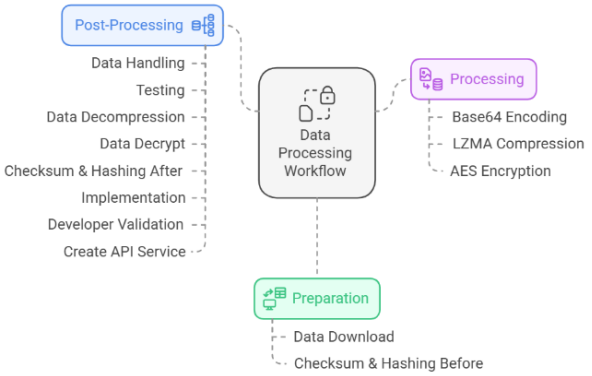

Figure 2. Research Testing Process Flow

Figure 2 explains the flow of the research testing process as follows. Once the data is obtained, the next step includes encoding the binary data using the Base64 algorithm [29] to facilitate handling and transfer. Next, the data is compressed using the LZMA algorithm to reduce the file size. Finally, the files are encrypted using the AES algorithm to protect the data from unauthorized access. Using datasets from the DICOM Library, this study aims to explore and test the results of the combination of encryption and compression in maintaining the security and efficiency of electronic medical data storage.

The integrity of the data in this study was measured by using the checksum and hash value methods to ensure that the data remained intact during processing. A checksum is a value generated from an algorithm that calculates the total bits of data. By comparing the checksum values before and after processing, it can be determined whether the data has changed or is still retained. If the checksum values are the same, this indicates that the integrity of the data is maintained [30].

The hash value generated from the MD5 algorithm is also used to measure integrity. The hash value will change if there is any modification to the data. When calculating the hash value for the original data and comparing it to the hash value of the data that has been processed, the similarity between the two confirms that the data remains intact. By applying these two methods,

the research shows rigor in maintaining data integrity, ensuring that the data used is accurate and reliable.

In this study, statistical analysis was applied to evaluate the results of combining encryption and compression methods applied to DICOM files. The analysis methods used include descriptive analysis and inferential analysis. Descriptive analysis is performed to provide an overview of the data collected, including the file size before and after the encryption and compression process. Descriptive statistics such as averages, medians, and standard deviations are calculated to better understand the distribution of file sizes. Furthermore, inferential analysis is used to test hypotheses regarding the use of applied methods. The t-test is used to compare file sizes before and after encryption and compression, with the aim of determining if there is a statistically significant difference. Additionally, regression analysis can be applied to evaluate the relationship between different variables, such as compression rate and final file size. All statistical analysis is done using Python, utilizing libraries such as NumPy and SciPy for statistical calculations, as well as Matplotlib and Seaborn for data visualization. The results of the analysis are presented in the form of tables and graphs for easy interpretation. By applying this statistical analysis method, this study aims to provide empirical evidence regarding the success of the combination of AES encryption and LZMA compression in reducing DICOM file size while maintaining data integrity.

The final results are delivered to the health information system developers for implementation and to find out if the proposed methods are acceptable and implemented in the development of health information systems, by providing API endpoints for developers to try.

3. Results and Discussions

This study adopts a systematic approach [31]. Upon conducting experiments and evaluating the outcomes of the study, it was determined that the proposed methodology for DICOM data management exhibited significant efficacy. The comprehensive process, encompassing file downloading, encoding, compression, and encryption, has been demonstrated to not only safeguard data but also reduce file size. The employed methods to ensure data integrity indicate that the data remains accurate and reliable for subsequent analysis. Nonetheless, it is observed that the implementation of the compression algorithm and the encryption algorithm may intensify the CPU load, potentially impacting system performance. For instance, the utilization of the LZMA compression algorithm necessitates extended processing durations for compression and decompression, compared to simpler compression methods, possibly leading to delays in data processing, particularly when the system is required to manage a substantial volume of DICOM files concurrently. Additionally, data encryption utilizing the AES algorithm demands considerable CPU resources, especially when encrypting or decrypting sizable files. Insufficient CPU capacity within the system can culminate in overall performance degradation, manifesting as slower response times in applications accessing such data. Therefore, it is imperative to contemplate mitigation strategies, including algorithm optimization or the employment of parallel processing, to ameliorate these impacts and sustain optimal system performance

3.1 Result

The test was carried out by preparing 427 standard files for medical document storage, namely DICOM files, which then went through a simultaneous compression and encryption process for each file.

```
Checksum MD5 Awal: 1f781c86ffabf2f9f30669a15457e96b
Checksum MD5 Enkripsi: 35b1340cc5bd2b84cfdad34a70629116
Ukuran File Asli: 513.06 KB
Ukuran File Setelah Kompresi: 4.96 KB
Penghematan Ukuran: 99.03%
Ukuran File Terenkripsi: 5.00 KB
Durasi Enkripsi: 0.22 detik
Penggunaan CPU: 69.03%
Penggunaan Memori: 5696.00 KB
Total files processed for encryption: 1
root@272a8a921aed:/app#
```

Figure 3. Encryption Process

Figure 3 demonstrates the efficacy and performance of the encryption technique, encompassing reductions in file size and resource utilization. This approach suggests that encryption can be executed swiftly and effectively, with significant reductions in size. For instance, the initial file size of 513.06 KB was efficiently decreased to 4.96 KB following encryption, resulting in a 99.03% size reduction. This illustrates that encryption is not only secure but also improves storage efficiency.

```
Checksum MD5 Awal: 35b1340cc5bd2b84cfdad34a70629116
Checksum MD5 Dekripsi: 1f781c86ffabf2f9f30669a15457e96b
Ukuran File '/app/trial-1/file-decrypt/56364397.dcm': 513.06 KB
Durasi Dekripsi: 0.11 detik
Penggunaan CPU: 75.59%
Penggunaan Memori: 6408.00 KB
Total files processed for decryption: 1
root@272a8a921aed:/app#
```

Figure 4. Decryption Process

Figure 4: Explanation of the efficiency and performance of the decryption process. This process shows that the decryption can be done quickly, taking only 0.11 seconds, and using reasonable CPU and memory resources. The MD5 checksum value ensures that the decrypted data has returned to its original form with integrity preserved.

Figure 5 illustrates the correlation between file size, measured in kilobytes, and the duration required for the encryption process, measured in seconds. The analysis of the graph reveals a tendency for encryption time to increase as file size increases. Overall, the trend indicates that larger files require longer encryption periods.

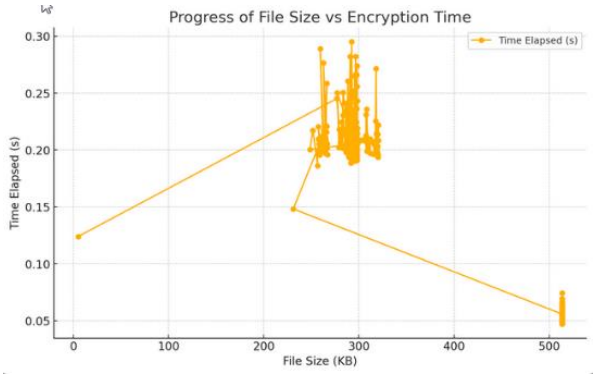

Figure 5. File Size vs Encryption Time

This research on compression efficiency involves comparing file sizes pre- and post-compression, as well as assessing the time taken for compressing and decompressing. Consequently, it enables the assessment of the advantage the LZMA algorithm provides for managing DICOM data and its effect on the overall workflow. The compression level is computed using the formula presented below, known as Equation 1.

$$E = 100\, x\, \frac{(O-C)}{O} \quad (1)$$

Compression efficiency (E) is quantified in terms of percentages and is determined utilizing the original file size (O) in kilobytes (KB), along with the compressed file size (C) in kilobytes (KB).

Table 1. Analysis of File Size Reduction Through Compression

File Name	Original Size	Compressed Size	Compression Ratio (%)
56364397.dcm	513,06	4,96	99,03
56364409.dcm	513,47	287,24	44,06
56364411.dcm	513,47	288,70	43,78
56364509.dcm	513,46	297,36	42,09
56364401 dcm	513,46	277,15	46,02

Table 1 shows the compression results of various files with identical formats, detailing the original file size, compressed file size, and percentage savings achieved. The compression efficiency is significantly affected by the content of the file characterized by simple data patterns, such as 56364397.dcm, showing savings of up to 99%. In contrast, files containing complex data achieve size reductions ranging from 42% to 44%. Monochromatic-dominated images generally experience a greater degree of compression compared to images with color, which is caused by reduced information content and simpler patterns. Such insights are essential for assessing the effectiveness of the compression algorithm and understanding how data patterns affect its performance.

Figure 6 demonstrates that as the original file size increases, the percentage of compression savings also rises, indicating that larger files generally offer greater savings when compressed. This relationship underscores the effectiveness of compression algorithms in managing larger datasets. However, while compression can enhance storage and transmission, it is vital to ensure that the process does not jeopardise the quality or integrity of the data being compressed.

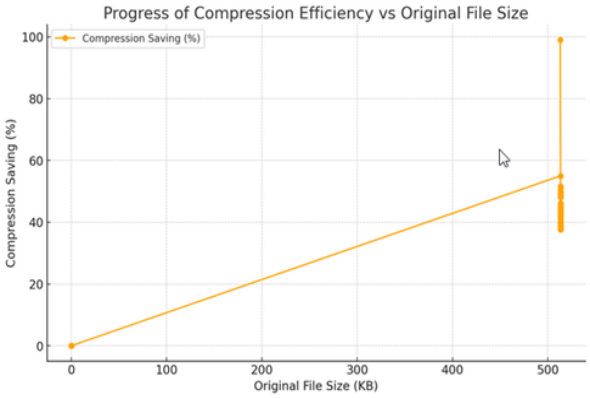

Figure 6. Compression Efficiency vs Original File Size

Data integrity is a principle that ensures that data remains accurate, consistent, and reliable [32]. Maintaining this integrity is particularly important in fields where precision is paramount, such as healthcare. For data quality, it is essential that the information utilized for decision-making is precise. Altering data in a way that removes critical information could critically impact health-related decisions, potentially leading to inaccurate diagnoses or ineffective treatments.

Figure 7. DICOM Files before Encryption and Compression

Figure 7 shows the DICOM file before the encryption and compression process. It is a visual representation of medical data which in this case is the result of a CT scan that shows the internal structure of the body.

Figure 8 shows DICOM files after going through the encryption and compression process. Here, data that was previously viewable in the form of images has been converted into a text format that is unreadable by humans. This indicates that the data has been encrypted to maintain security and compressed to reduce the file size.

Figure 8. DICOM Files after Encryption and Compression

Table 2. Results of Checksum MD5

File Name	Before and After
56364397.dcm	1f781c86ffabf2f9f30669a15457e96b
56364398.dcm	3a5a27238660dac23519229dc8e5181d
56364399.dcm	265305a774b44681ba7ff0c255e695e8
56364400.dcm	265305a774b44681ba7ff0c255e695e8
56364401 dcm	b9bac0795a210d55715ab88c2decdc62

Table 2 shows that the encryption and decryption process successfully preserved data integrity for all DICOM files. MD5 checksum results confirm that the file values remained identical throughout the process, indicating no data loss or alterations occurred. This demonstrates the methodology's ability to maintain data integrity effectively. The process ensures files remain secure, retrievable, and identical to their original state, which is critical in electronic medical data processing. It protects data during storage and transmission while enabling recovery without damage or loss. Combining compression and encryption enhances security and optimizes storage. Compression reduces file size for efficiency and adds a layer of complexity against unauthorized access. Overall, this approach achieves data integrity while addressing storage optimization needs.

To further validate the effectiveness of the proposed methodology, a statistical analysis using the T-test was conducted. This test aimed to determine whether there were any significant differences between the original and processed files in terms of key metrics such as file size and processing time.

Figure 9 compares the original and compressed file sizes. The t-statistic of 243.33 and p-value of 0.0 confirm a statistically significant difference, rejecting the null hypothesis. These results show that the compression process effectively reduces file size.

```
   Original File Size (KB)  Compressed File Size (KB)  Compression Saving (%)
0               513.064453                   4.960938               99.033077
1               513.466797                 276.882812               46.075810
2               513.466797                 279.164062               45.631526
3               513.466797                 280.570312               45.357652
4               513.466797                 281.621094               45.153008

T-statistic: 243.33048412008912
P-value: 0.0

Reject the null hypothesis: There is a significant difference between the two groups.
```

Figure 9. Results of the T-Test

To further analyze the relationship between original and compressed sizes, regression analysis was performed. This assessed the strength of the association and how well compression scales with larger files.

Figure 10 shows a strong correlation between compression rate and compressed file size. Using the Ordinary Least Squares (OLS) model with an R-squared value of 1.000, the model explains 100% of the variance in compressed file sizes. A coefficient of -513.4775 indicates that each unit increase in compression rate reduces file size by approximately 513.48 KB. The p-values for both coefficients are highly significant (0.000), confirming the strength of the relationship. However, potential issues such as atypical residual distribution and overfitting were noted. To further validate the robustness of these findings, additional regression diagnostics were conducted. This analysis aimed to address any anomalies in the residuals and ensure the model's reliability across different datasets.

OLS Regression Results

Dep. Variable:	Compressed Size	R-squared:	1.000
Model:	OLS	Adj. R-squared:	1.000
Method:	Least Squares	F-statistic:	2.940e+10
Date:	Fri, 17 Jan 2025	Prob (F-statistic):	0.00
Time:	09:31:54	Log-Likelihood:	1994.2
No. Observations:	427	AIC:	-3984.
Df Residuals:	425	BIC:	-3976.
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
const	513.4689	0.001	3.95e+05	0.000	513.466	513.471
Compression Rate	-513.4775	0.003	-1.71e+05	0.000	-513.483	-513.472

Omnibus:	474.372	Durbin-Watson:	0.443
Prob(Omnibus):	0.000	Jarque-Bera (JB):	62077.097
Skew:	-4.644	Prob(JB):	0.00
Kurtosis:	61.334	Cond. No.	32.3

Notes:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.

Figure 10. Regression Test Results

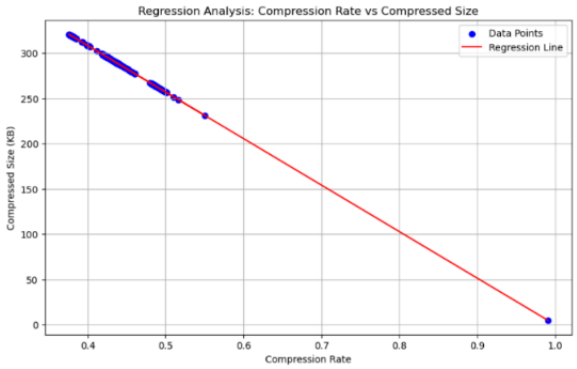

Figure 11. Regression Analysis

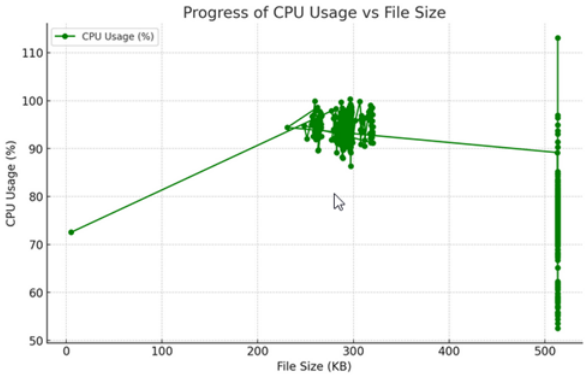

Figure 12. CPU Usage vs File Size

Figure 11 shows a regression analysis between the compression rate and the compressed file size. The blue dots represent the actual data, while the red lines indicate the resulting regression model. From the graph, it can be seen that there is a strong negative relationship: the higher the compression level, the smaller the compressed file size. This shows that increasing the compression level significantly reduces the file size.

3.2 Discussions

In this analysis, the relationship between the file size and the time it takes to encrypt the file is compared. Compression efficiency is measured by looking at the percentage of original file size savings after compression. Evaluations are also carried out on CPU usage which is an important part of performing processing, as well as memory usage and how both relate to the size of the file being processed. Here are the results of the testing process.

Figure 12 shows the relationship between the size of a file in kilobytes and the percentage of CPU usage during encryption and compression. From this graph, it can be seen that CPU usage remains stable, ranging between 80% and 100%, even as the file size varies from 0 to 500 KB. This indicates that the method is efficient in utilizing CPU resources, ensuring the system does not become overly burdened when processing files of different sizes.

To further analyze the efficiency of the method, a comparison of resource usage against file size was conducted. This analysis provides deeper insights into how various system resources, including memory and processing power, scale with increasing file sizes, offering a comprehensive understanding of the method's overall performance.

Figure 13 compares CPU and memory usage based on file size (in kilobytes). This graph shows that CPU usage remains low and stable, while memory usage increases significantly as file sizes grow, particularly above 300 KB. Memory usage peaks at around 500 KB, indicating that while CPUs are not overloaded, memory usage can be a critical concern when managing large datasets. This highlights the importance of optimizing memory efficiency in the proposed method. In addition to resource usage, it is also essential to evaluate how efficiently the method performs in terms of encryption time. A comparison of efficiency against encryption time provides insights into the trade-offs between processing speed and resource consumption, ensuring the method remains practical for real-world applications.

Figure 14 shows the relationship between the time spent on the encryption process in seconds and the compression efficiency in percentages. This graph shows that compression efficiency improves with

encryption time, and some data points show compression savings of nearly 100%. Although there was variation in compression efficiency, most data points were distributed between 20% and 60% savings, suggesting that the proposed method remains efficient in terms of compression despite varying encryption times. This confirms that this approach is reliable in practice, with a balance between encryption speed and storage efficiency.

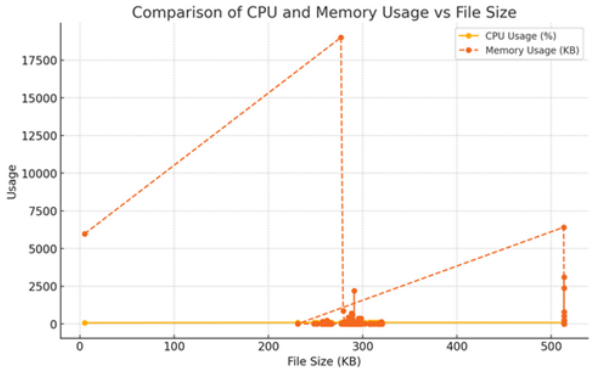

Figure 13. Comparison of Resource Usage vs File Size

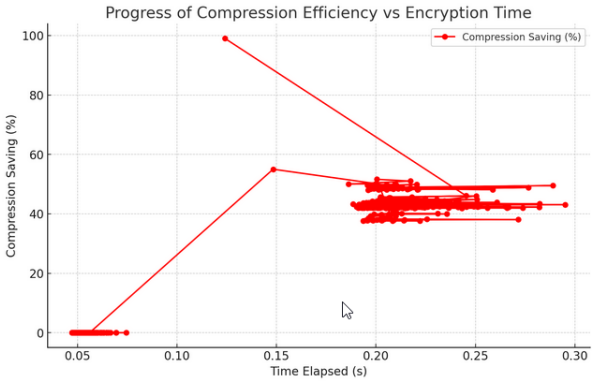

Figure 14. Comparison of Efficiency vs Encryption Time

Based on previous testing, there is a possibility that resources may be depleted, causing the process to halt. This condition can disrupt operations, especially in systems that require high performance, such as in medical data processing. Therefore, mitigation is necessary to anticipate these risks and ensure that the system continues to function optimally, even under high workload conditions. Mitigation constitutes an action or series of actions aimed at reducing or controlling the negative impact of a risk, issue, or threat. When multiple users concurrently access a system, an increase in CPU load may occur, potentially leading to diminished response times and a less satisfactory user experience. This aspect is particularly critical within a medical context, where rapid and precise access to patient data is imperative for knowledgeable decision-making. Elevated CPU usage can result in heightened operational costs, notably within cloud-based models where costs are frequently determined by resource utilization. Organizations handling medical data must account for these expenses within their financial planning, as excessive CPU usage presents a challenge to effective cost management. Moreover, scalability constraints may emerge if the system proves incapable of accommodating high workloads, thus hindering the system's capacity to expand in line with the growth of data and user volume.

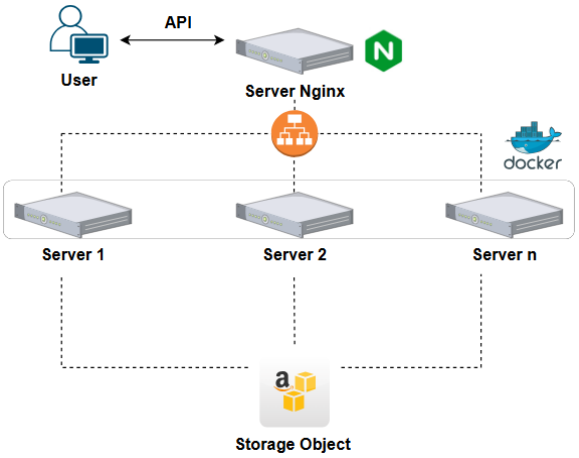

Figure 15. Parallel Processing Scheme

Figure 15 shows the parallel processing scheme providing an explanation to overcome the challenges there are several mitigation measures that can be proposed. One of them is the use of parallel processing. By implementing parallel processing, workloads can be distributed among multiple CPU cores, allowing encryption and compression processes to occur simultaneously. This not only speeds up the overall processing time, but also reduces the strain on a single CPU core, thereby improving overall system performance. The image above illustrates a system architecture designed to support scalability and efficiency in big data management. In this architecture, users interact with the system through APIs, which serve as an entry point for all requests. Nginx acts as a web server that manages and distributes user requests to multiple servers, such as Server 1, Server 2, and Server n. Nginx acts as a load balancer, allowing for an even distribution of requests across multiple servers, thus preventing overloading on a single server and improving system availability and responsiveness. The use of Docker allows applications to run in isolated containers, making it easy to quickly manage and deploy applications. Containers can be created, stopped, or moved between servers, providing flexibility in resource management. The object storage elements in this architecture allow data to be stored separately from the application server, which is designed to handle large amounts of data and provide fast and efficient access. This architecture is particularly relevant for a variety of real-world applications, such as content management systems and e-commerce platforms, due to its ability to scale and adapt to evolving business needs. As such, this architecture not only supports scalability but also provides flexibility and adaptability to changing needs in a dynamic business environment.

In addition, algorithm optimization can also be an effective mitigation measure. Using more efficient encryption algorithm and compression algorithm in terms of CPU usage can help reduce the load on system resources. Further research can be conducted to explore an algorithm that is lighter but still secure, so that it can reduce CPU usage without sacrificing data security.

The scalability of resources is also important to consider. Adopting a cloud architecture that allows for dynamic adjustment of resources as needed can help manage CPU usage. For example, increasing CPU capacity as the workload increases and decreasing it when not needed can help maintain optimal system performance. Then, performance monitoring and analysis can be implemented to analyze CPU usage and system performance in real time. With a good monitoring system, bottlenecks can be quickly identified and resolved, and encryption and compression processes can be optimized to improve efficiency. With these mitigation measures, cloud-based medical record systems can better manage CPU resource usage, improve efficiency, and ensure optimal performance in medical data management.

This research has several limitations that must be considered. First, the sample size used is relatively small, which can limit the validity of the findings and the generalization of the results. In addition, there is a possibility of bias in the data collection process, which can affect the accuracy of the research results. The results obtained may not be generalizable to the larger population, suggesting the need for further research with larger sample sizes. Lastly, technical factors, including the specifications of the hardware used in this study, have a significant impact on the results obtained. First, the amount of CPU and RAM available during testing can result in variations in results, as limited resources can cause the application to not function optimally. For example, if there is insufficient RAM, the application may experience delays in processing data or even experience failures in performing some functions. Additionally, when an application is implemented on an existing server, there is the potential to disrupt the performance of other applications running on the same server. If the server doesn't have enough resources to handle the additional load of a new application, this can lead to performance degradation, such as increased latency or downtime, which can impact the overall user experience. Therefore, it is important to perform testing on CPUs with AMD architecture, as differences in CPU architecture can affect how applications operate. Testing on different architectures will provide additional insight into the performance of the application and help identify potential issues that may arise, so that the results of the study can be more comprehensive and reliable. By taking all these factors into account, research can be carried out better and the results obtained can be more valid and relevant.

In the context of learning ahead, the use of more efficient encryption algorithm such as ChaCha20 and precise block size settings with GCM mode for AES can provide significant advantages in terms of performance and resource efficiency. While AES remains an excellent choice for maintaining data integrity, ChaCha20 offers a faster and more efficient alternative, especially on devices with limited resources. By considering these two algorithms, researchers and practitioners can choose the encryption method that best suits the specific needs of their application, both in terms of security and performance. This will help in the development of more efficient and secure systems for the security of electronic medical documents and other applications.

4. Conclusions

In conclusion, this research successfully developed an innovative method that integrates AES Algorithm and LZMA Algorithm to enhance the security and efficiency of electronic medical data storage. The method allows for simultaneous encryption and compression, significantly improving the protection of sensitive medical data while optimizing storage space. The results demonstrate that this approach can reduce DICOM file sizes by 40-50% and achieve an average encryption time of approximately 0.2-0.3 seconds per file. This method is particularly relevant for cloud-based medical record systems, where data protection and storage efficiency are critical priorities. The findings suggest that the integration of encryption and compression can address the challenges of managing large medical files while maintaining data integrity and accessibility. For future research, it is recommended to test this method on a variety of other medical data formats and explore the incorporation of advanced processing techniques based on artificial intelligence. Additionally, analyzing the potential for cyberattacks on systems utilizing this method could provide valuable insights into enhancing security measures.

References

[1] A. Sarce Joel *et al.*, "Information Technology-Based Medical Record Management in Handling Confidentiality and Security of Patient Data Using Cryptographic Methods (In Indonesian Language)," *Jurnal Indonesia : Manajemen Informatika dan Komunikasi*, vol. 4, no. 3, pp. 837–848, Sep. 2023, doi: 10.35870/JIMIK.V4I3.287.

[2] H. S. Lallie *et al.*, "Cyber security in the age of COVID-19: A timeline and analysis of cyber-crime and cyber-attacks during the pandemic," *Comput Secur*, vol. 105, Jun. 2021, doi: 10.1016/j.cose.2021.102248.

[3] M. W. Malik, D. Husna, I. K. E. Purnama, I. Nurtanio, A. N. Hidayati, and A. A. P. Ratna, "Development of Medical Image Encryption System Using Byte-Level Base-64 Encoding and AES Encryption Method," in *ACM International Conference Proceeding Series*, Association for Computing Machinery, Nov. 2020, pp. 153–158. doi: 10.1145/3442555.3442580.

[4] R. Wahyudi and Moh. A. Romli, "Android-based Patient Medical Record Data Security Application using AES and RSA Method Cryptography," *Int J Comput Appl*, vol. 185, no. 40, pp. 34–39, Nov. 2023, doi: 10.5120/ijca2023923205.

[5] M. Azhari, J. Perwitosari, and F. Ali, "Implementation of Data Security on Documents Using Advanced Encryption Standard (AES) Cryptographic Algorithm (In Indonesian Language)," *Jurnal Pendidikan Sains dan Komputer*, vol. 2, no. 1, pp. 2809–476, 2022, doi: 10.47709/jpsk.v2i1.1390.

[6] A. K. Muthaura and J. Kandiri, "Data protection in Healthcare Information Systems Using Cryptographic Algorithm with Base64 512 bits," *Journal of the Kenya National Commission for UNESCO*, vol. 4, no. 2, Jul. 2024, doi: 10.62049/jkncu.v4i2.105.

[7] A. R *et al.*, "Securing e-Health application of cloud computing using hyperchaotic image encryption framework," *Computers and Electrical Engineering*, vol. 100, p. 107860, May 2022, doi: 10.1016/J.COMPELECENG.2022.107860.

[8] K. A. Seputra, A. A. G. Y. Paramartha, G. A. Pradnyana, and K. Y. E. Aryanto, "A Middleware Applications Design for Health Information Sharing," *Jurnal RESTI (Rekayasa Sistem dan Teknologi Informasi)*, vol. 8, no. 3, pp. 321–332, Jun. 2024, doi: 10.29207/resti.v8i3.5707.

[9] Resdiansyah, J. Darmawan, A. H. Wijaya, L. Hakim, and H. Tannady, "Comparing Freeman Chain Code 4 Adjacency Algorithm and LZMA Algorithm in Binary Image Compression," in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Feb. 2021. doi: 10.1088/1742-6596/1783/1/012045.

[10] R. Sowmyalakshmi *et al.*, "An Optimal Lempel Ziv Markov Based Microarray Image Compression Algorithm," *Computers, Materials and Continua*, vol. 69, no. 2, pp. 2245–2260, 2021, doi: 10.32604/CMC.2021.018636.

[11] Z. Mishra and B. Acharya, "High throughput and low area architectures of secure IoT algorithm for medical image encryption," *Journal of Information Security and Applications*, vol. 53, Aug. 2020, doi: 10.1016/J.JISA.2020.102533.

[12] Y. Prayudi and A. Ashari, "A Study on Secure Communication for Digital Forensics Environment," *Int J Sci Eng Res*, vol. 6, no. 1, pp. 1036–1043, Jan. 2015, doi: 10.14299/IJSER.2015.01.010.

[13] M. Hasyim Ratsanjani, I. Fitria Risnandari, T. Widya Sulaiman, and V. Meida Hersianty, "Literature Review: The Role of SAAS Applications in E-Commerce Business Activities (In Indonesian Language)," *SINOMIKA Journal: Publikasi Ilmiah Bidang Ekonomi dan Akuntansi*, vol. 1, no. 4, pp. 1009–1020, Nov. 2022, doi: 10.54443/sinomika.v1i4.491.

[14] B. Hartono, "Ransomware: Understanding the Digital Security Threat (In Indonesian Language)," *Bincang Sains dan Teknologi*, vol. 2, no. 02, pp. 55–62, May 2023, doi: 10.56741/bst.v2i02.353.

[15] "10 largest healthcare data breaches of 2024 | TechTarget." Accessed: Jan. 19, 2025. [Online]. Available: https://www.techtarget.com/healthtechsecurity/feature/Largest-healthcare-data-breaches

[16] H. Wijayanto, D. Daryono, and S. Nasiroh, "Forensic Analysis On Peduli Lindungi Application Against Personal Data Leaks (In Indonesian Language)," *Jurnal Teknologi Informasi dan Komunikasi (TIKomSiN)*, vol. 9, no. 2, p. 11, Nov. 2021, doi: 10.30646/tikomsin.v9i2.572.

[17] H. Rizki Kurnia, A. Zahrah, E. Ichsazene, and N. Aini Rakhmawati, "Bibliometric Analysis of Data Breach Issues Publications Using VOSviewer (In Indonesian Language)," *Jurnal Informatika Sunan Kalijaga)*, vol. 8, no. 3, pp. 231–242, 2023, doi: https://doi.org/10.14421/jiska.2023.8.3.231-242.

[18] I. Keshta and A. Odeh, "Security and privacy of electronic health records: Concerns and challenges," *Egyptian Informatics Journal*, vol. 22, no. 2, pp. 177–183, Jul. 2021, doi: 10.1016/J.EIJ.2020.07.003.

[19] O. G. Khoirunnisa and D. Djuniadi, "Implementation of AES Algorithm for Medical Record Data Security (In Indonesian Language)," *PETIR*, vol. 15, no. 1, pp. 21–27, Dec. 2021, doi: 10.33322/petir.v15i1.1333.

[20] R. Rojas-Hernández, J. L. Díaz-De-león-santiago, G. Barceló-Alonso, J. Bautista-López, V. Trujillo-Mora, and J. C. Salgado-Ramírez, "Lossless Medical Image Compression by Using Difference Transform," *Entropy*, vol. 24, no. 7, Jul. 2022, doi: 10.3390/E24070951.

[21] A. R. Ekaputra and A. S. Affandi, "Utilizing cloud computing services and docker containers to improve web application performance (In Indonesian Language)," *Journal of Information System and Application Development*, vol. 1, no. 2, pp. 138–147, Sep. 2023, doi: 10.26905/jisad.v1i2.11084.

[22] J. T. Informasi, A. Ridho, and C. R. Niani, "Encryption Implementation With Vigenere Cipher And Reverse Cipher Using Python Programming Language (In Indonesian Language)," *Jurnal Teknologi Informasi*, vol. 1, no. 1, pp. 9–15, May 2022, doi: https://doi.org/10.35308/.v1i1.5486.

[23] Z. Chen and G. Ye, "An asymmetric image encryption scheme based on hash SHA-3, RSA and compressive sensing," *Optik (Stuttg)*, vol. 267, p. 169676, Oct. 2022, doi: 10.1016/J.IJLEO.2022.169676.

[24] M. Hlayel, H. Mahdin, M. Hayajneh, S. H. AlDaajeh, S. S. Yaacob, and M. M. Rejab, "Enhancing unity-based AR with optimal lossless compression for digital twin assets," *PLoS One*, vol. 19, no. 12, p. e0314691, Dec. 2024, doi: 10.1371/JOURNAL.PONE.0314691.

[25] E. Öztürk and A. Mesut, "Learning-based short text compression using BERT models," *PeerJ Comput Sci*, vol. 10, 2024, doi: 10.7717/peerj-cs.2423.

[26] A. Hafsa, A. Sghaier, J. Malek, and M. Machhout, "Image encryption method based on improved ECC and modified AES algorithm," *Multimed Tools Appl*, vol. 80, no. 13, pp. 19769–19801, May 2021, doi: 10.1007/S11042-021-10700-X.

[27] D. A. Clunie, "DICOM Format and Protocol Standardization—A Core Requirement for Digital Pathology Success," *Toxicol Pathol*, vol. 49, no. 4, pp. 738–749, Jun. 2021, doi: 10.1177/0192623320965893.

[28] K. Masters, "Ethical use of Artificial Intelligence in Health Professions Education: AMEE Guide No. 158," *Med Teach*, vol. 45, no. 6, pp. 574–584, 2023, doi: 10.1080/0142159X.2023.2186203.

[29] R. O. Bura and H. S. Nida, "Image Transmission Using Base64 Encoding and Advanced Encryption Standard Algorithm Based on Socket Programming," *Proceedings - IWBIS 2021: 6th International Workshop on Big Data and Information Security*, pp. 115–120, 2021, doi: 10.1109/IWBIS53353.2021.9631846.

[30] A. Meylan, M. Cherubini, B. Chapuis, M. Humbert, I. Bilogrevic, and K. Huguenin, "A Study on the Use of Checksums for Integrity Verification of Web Downloads," *ACM Transactions on Privacy and Security*, vol. 24, no. 1, Nov. 2020, doi: 10.1145/3410154.

[31] D. Prasetyo, A. Utami, and T. G. Laksana, "Website Based Academic Information System Design Using Extreme Programming Method," *Journal of Informatics Information System Software Engineering and Applications (INISTA)*, vol. 6, no. 2, pp. 134–143, Jul. 2024, doi: 10.20895/inista.v6i2.1214.

[32] R. Salwa, M. Irwan, P. Nasution, F. Ekonomi, and D. Bisnis, "Data Privacy Preservation Techniques In Cloud Database (In Indonesian Language)," *Kohesi: Jurnal Sains Dan Teknologi*, vol. 3(7), pp. 11–20, 2024, doi: https://doi.org/10.3785/kohesi.v3i7.3735.

