

Large file encryption in a Reduced-Round Permutation-Based AES file management system

Jerico S. Baladhay, Heidilyn V. Gamido, Edjie M. De Los Reyes

College of Computer Studies, Tarlac State University, Tarlac, Philippines

Article Info

Article history:

Received Feb 13, 2024

Revised Feb 24, 2024

Accepted Mar 10, 2024

Keywords:

Algorithm

Encryption

File management system

Permutation

Security

ABSTRACT

In the rapid evolving digital landscape, the imperative to ensure data security has never been more crucial. This paper addresses the pressing challenges in data security by introducing a file encryption management system, leveraging a modified advanced encryption standard (AES) algorithm with reduced round iterations and bit permutation. This system aims to comprehensively secure various file types, providing a dependable solution for file exchange. Our findings reveal substantial improvements in both encryption and decryption processes using the reduced-round permutation-based AES (RRPBA). The adapted algorithm demonstrates a significant 38.8% acceleration in encryption time and a remarkable 44.86% improvement in decryption time, positioning it as a pivotal component for efficient file operations within the management system. Moreover, the throughput assessments showcase a remarkable 33.73% improvement in encryption and 23.72% in decryption, outperforming the original AES, emphasizing the algorithm's superior computational effectiveness, signaling positive implications for future high-performance applications. In conclusion, the study not only addresses critical security challenges but also presents a viable solution with tangible speed advantages for file encryption and decryption processes within digital file management systems.

This is an open access article under the [CC BY-SA](https://creativecommons.org/licenses/by-sa/4.0/) license.



Corresponding Author:

Jerico S. Baladhay

College of Computer Studies, Tarlac State University

Tarlac, Philippines

Email: j.baladhay0785@student.tsu.edu.ph

1. INTRODUCTION

The need for security is a crucial imperative in today's digital era [1], applicable to both organization and individual users. The ubiquity of file sharing, editing, and collaborative work necessitates robust security measures to safeguard sensitive information [2]. It addresses a significant problem in the realm of computer security, specifically focusing on the secure management of large files [3]. Furthermore, in the computer domain, the global challenge revolves around the secure handling of digital data, with a particular emphasis on large file sizes [4]. As organizations increasingly rely on digital platforms for their operations, the vulnerability of data to unauthorized access and breaches poses a universal concern [5], [6]. The need to address this challenge is not confined to specific regions or industries but is applicable across diverse sectors.

One notable approach within cryptographic transformations is symmetric encryption, where a single secret key, or cipher key, is used for both encryption and decryption processes [7]. This reversible transformation ensures that only authorized parties with the correct key can access the original information, thereby safeguarding data privacy. Employing widely adopted cryptographic standards, such as the advanced encryption standard (AES), further fortifies the security file encryption management systems [8]. Introduced by the National Institute of Standards and Technology (NIST) in 1997 as the successor to the data encryption

standard (DES) [9], AES offers various key sizes, including 128 bits, 192 bits, and 256 bits, ensuring a robust and adaptable security framework for encrypting sensitive data [10]. In essence, file management systems not only provide a shield against unauthorized access [11] but also adhere to established cryptographic practices, contributing to the overall resilience of digital security in today's dynamic landscape [12].

The absence of a file management system within an organization face risks like unauthorized access, data breaches, and compromised confidentiality [13]. Encryption is crucial for protecting sensitive information and ensuring regulatory compliance. In remote work scenarios, the absence of encryption poses a threat to secure data sharing, making interception during transit a concern [14]. The identified limitation in current file encryption management systems pertains to their constrained compatibility with specific file types and limited support for small file sizes. Addressing and rectifying this limitation would constitute a significant enhancement for organizations aspiring to adopt a comprehensive and flexible approach to file encryption [15], [16].

Additionally, some file encryption management systems are limited to specific file type, thus, recognizing the need for improvement [3], adopting various file types will promote seamless collaboration within the organization. This emphasizes the importance of a flexible approach to file encryption, addressing diverse data formats encountered in modern workflows. Several scholarly investigations have been published, each reaching the consensus that the encryption of files [17], involving the application of a customized encryption algorithm to ensure the security of files [18], or the incorporation of an electronic document management system, designed to facilitate the efficient administration of documents within an organizational framework, both contribute to guaranteeing the preservation of data integrity and confidentiality.

The study proposes a flexible approach to file encryption, unlike conventional systems, our developed file management system is designed to accommodate various file types, promoting seamless collaboration within organizations. By addressing the diverse data formats encountered in modern workflows, our approach ensures that the file encryption system is not limited to specific file types. The developed system involves the management of users, key storage, and file upload facilitated by a modified AES algorithm that is applied for larger file sizes for accelerated performance. In addressing the secure management of large files in computer security, this study contributes significantly to the field by introducing a flexible file encryption approach. Overcoming the limitations of current systems restricted to specific file types and small sizes, our research pioneers the development of a comprehensive file management system, fostering seamless collaboration within organizations by accommodation diverse data formats. Furthermore, it proposes a modified algorithm tailored larger file sizes, ensuring both heightened security and accelerated performance in file encryption management systems.

2. METHOD

This section discusses the integration of a file encryption algorithm, reduced-round permutation-based AES (RRPBA), into the development of the file management system. Thus far, it presents the design, development, and methods used in the developed system. Additionally, it highlighting the strategic incorporation of RRPBA to ensure secure and efficient data handling throughout the file management process.

2.1. System methodology

The rapid application development (RAD), as depicted in Figure 1, methodology was employed in the development and implementation of the file management system. In the RAD model, the initial phase involves assembling development teams and creating a loose roadmap based on client-defined requirements. This collaborative process, often conducted through workshops or meetings, outlines the necessary functionalities for the project [19]. The subsequent stage focuses on prototyping, where both back-end and front-end coding are utilized to create tangible modules [20], [21]. This phase is iterative, allowing for multiple revisions based on client feedback until mutual satisfaction is achieved. Continuous check-ins and beta-tests ensure alignment with client expectations, emphasizing the RAD model's adaptability. Finally, the model enables the rapid integration of components, leading to the optimization and finalization of the application.

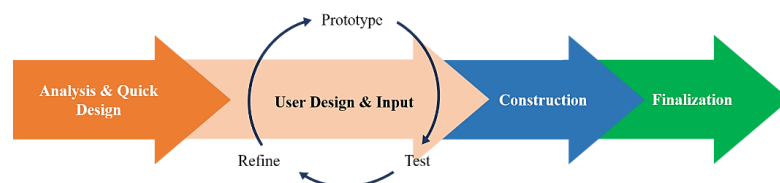


Figure 1. Rapid application development model

2.1.1. Analysis and quick design

The procedural workflow of the file management system is delineated in Figure 2. Commencing with user authentication, a unique key is generated to facilitate secure access. Subsequently, the user initiates the file upload process. Concurrently, the uploaded file undergoes encryption, and its validity is verified. In the event of an invalid file, the user is prompted to re-upload a file that meets the stipulated criteria. Upon successful encryption, the generated key is stored in the database, affording the administrator the capability to manage the repository of stored keys. Subsequent to key storage, the system proceeds to decrypt the file, culminating in its display for the user. This sequential process ensures the secure and efficient management of files within the system, encompassing user authentication, encryption, key storage, and file decryption. Upon successful encryption of the file in accordance with its designated key, access to its contents becomes contingent upon decryption. Subsequently, an administrative protocol is activated in the event of file release. If affirmative, the authorized user possessing access privileges will proceed to view the file; conversely, if the release is denied, the procedural sequence terminates.

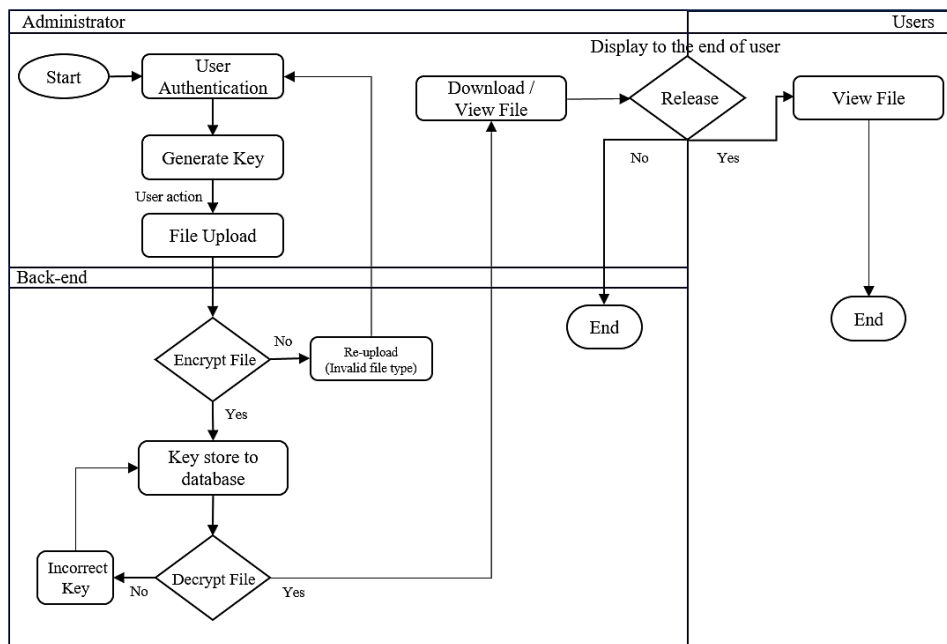


Figure 2. File management system flow structure

2.1.2. Construction and finalization

In the last phase of the model, the project progresses from prototyping to seamlessly integrate components for the file management system. This stage emphasizes swift assembly, with development teams collaborating closely on both back-end and front-end coding. Regular check-ins and beta-tests serve as critical checkpoints, enabling adjustments and optimization to align with client expectations. The RAD model's adaptability becomes evident as components are rapidly integrated, resulting in the efficient construction and finalization of the application.

2.2. System’s operational design–LAN-based configuration

Figure 3 illustrates the operational system within a LAN-based configuration. The user initiates the process by logging in and selecting a file, which can be of various formats such as text, audio, video, or image. Subsequently, the user chooses recipients with whom to share the file and proceeds to initiate the sharing process. The system then encrypts the selected file, using the modified algorithm, before transmitting it to the server. The server retains the necessary decryption keys, ensuring that the receiver is unable to access the file without proper authorization. Upon completion of the sharing process, the intended receiver logs in to the system, retrieves the shared file, and gains the ability to either view or download the file securely. This multi-step procedure enhances the security and controlled access to shared files within the LAN-based system.

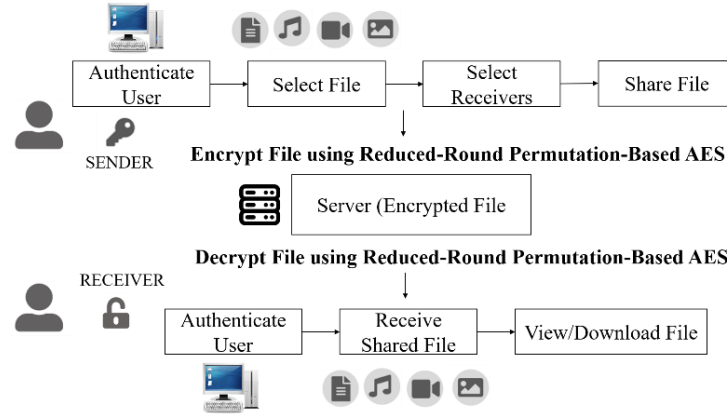


Figure 3. Operational design LAN-based

2.3. Reduced-round permutation-based algorithm

The process of file encryption is facilitated through the implementation of a customized algorithm known as the reduced-round permutation-based AES. This algorithm, as shown in Figure 4, has been designed to ensure the security of files on a large scale, accommodating substantial file sizes. The modification introduced to enhance its effectiveness involves a reduction in the number of rounds of iterations from the standard 10 to 6, coupled with the substitution of the MixColumns function with a bit permutation technique, drawing inspiration from a relevant study [22]. This tailored encryption approach aims to fortify the protection of digital assets, particularly in scenarios involving extensive file dimensions.

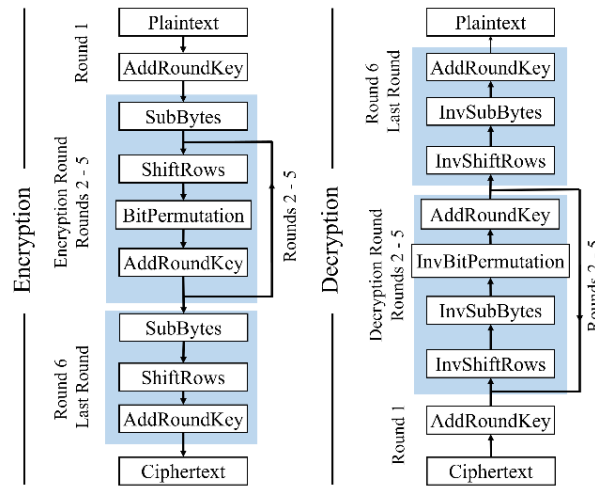


Figure 4. Reduced-round permutation-based AES (RRPBA) diagram

The process of Bit Permutation involves rearranging the values of a byte according to a predefined bit permutation table. To comprehend this procedure, it is essential to be familiar with the following terms: i) arrays-collections of data. ii) Index-the position or sequence of data within an array, denoted by a numerical value. iii) Elements-the individual data units within an array.

A byte, in this context, functions as an array of binary data where elements can only take on values of 1 or 0. To carry out the permutation of the byte, the bit permutation table comes into play. This comprises a sequence of indices that dictate the shuffling order of the byte data. Consider a byte with the binary data: 10011001. The default index sequence is consistently 01234567, where the value of the element at index 0 is 1, at index 1 is 0, at index 2 is 0, and at index 3 is 1, and so forth. Byte permutation is achieved through the utilization of bit permutation table. Each entry in the table signifies the new position of an element in the

sequence, dictating a comprehensive rearrangement. This iterative process continues until all elements have been appropriately repositioned

The procedure involves performing an XOR operation on the succeeding byte, followed by binary conversion and subsequent permutation of the binary representation using a fixed permutation table. Moreover, with the utilization of a precomputed result, the range of potential outcomes is restricted to 255. This process, while seemingly straightforward, exhibits robustness such that a singular alteration of a bit yields a completely distinct result.

2.4. Cipher block chaining

The conventional AES typically operates on 128-bit blocks, employing cipher block chaining (CBC) mode for distinct encryption of each block. This mode, applicable across various AES block sizes, involves XORing the initial block with a 128-bit initialization vector (IV) before encryption, and this process repeats until the final block [23]. It's important to note that the flexibility of CBC mode extends beyond AES 128, allowing for varied encryption configurations [24]. However, the key management aspect of this mode may introduce additional execution time considerations [25].

3. RESULTS AND DISCUSSION

In this section, encompasses pivotal aspects, covering system development information, system prototype design. The section also includes a thorough evaluation of the speed and throughput for both the AES and modified AES algorithms. This approach not only delves into the intricacies of system development and prototype design but also highlights the positive strides made in algorithmic speed and throughput.

3.1. System development

The development of the file management system involved the integration of NodeJS, NestJS, and ReactJS, strategically selected for their prowess in backend and front-end development. Concurrently, the architecture of the system's database was designed, employing the widely-adopted MySQL. The selection of NodeJS [26], NestJS, and ReactJS [27] underscores a commitment to utilizing contemporary and efficient frameworks, thereby optimizing the overall user experience. Additionally, the prudent choice of MySQL as the database management system further fortifies the system's robustness and reliability, establishing it as an innovative solution for efficient and streamlined file management. The system is deployed in a localhost; hence it runs in a local area network (LAN)-based system.

3.1.1. User authentication

As depicted in Figure 5, it visually presents these components, necessitates the inclusion of username and password fields, a show password option, and a login button for secure user verification. Thus indicates a robust user authentication process; users input their credentials, upon successful verification, gain system access. This design prioritizes security without compromising user convenience, aligning with contemporary usability expectations. In essence, it emphasizes on a user-friendly approach enhances the overall system experience, making it a noteworthy contribution to the field.

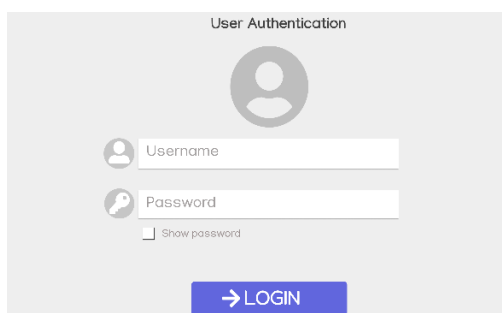


Figure 5. User authentication module

3.1.2. Generate key

Figure 6 demonstrates the system features a "Generate Key" module, as depicted, accessible from the side navbar, choose "Keys", where users can select file types. The module includes a list of existing keys

and an option to add new keys. Clicking “Add Key” triggers a modal for key generation, allowing users to seamlessly generate keys for selected file types within the intuitive interface.

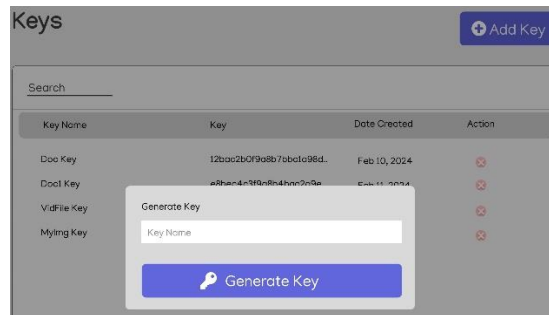


Figure 6. Generate Key module

3.1.3. File upload

In Figure 7, as depicted in the file upload module, users initiate the file upload process by clicking the “Upload File” button, triggering a modal. Within the modal, users can select the desired file and choose a corresponding key before proceeding to upload the file. This user-friendly approach streamlines the file upload experience within the system. Following the file upload and key selection process, the system proceeds to encrypt the uploaded file. Once the file upload and key selection process is completed, the system automatically initiates the encryption of the uploaded file. This ensures a secure data transfer, reinforcing the confidentiality and integrity of the user’s data within the system.

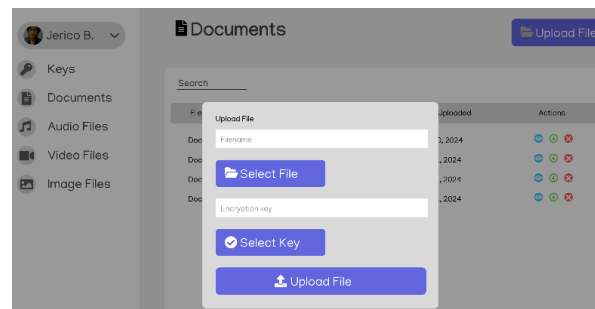


Figure 7. File upload module

3.1.4. Encrypted/decrypted files

Figure 8 demonstrates the successful execution of the file upload procedure; it is imperative to underscore that any endeavors to access the encrypted file subsequent to the upload operation are to be avoided. This precautionary directive is rooted in the encryption protocol employed. The visual representation contained therein elucidates the composition of the encrypted file and its corresponding decrypted files. Figure 8(a) displays the text file in its encrypted state; Figure 8(b) showcases the decrypted text file. In Figure 8(c), you can observe the encrypted audio file, while Figure 8(d) exhibits the decrypted and playable audio file. The encrypted image is depicted in Figure 8(e), with Figure 8(f) illustrating the decrypted image. As for the encrypted video file, Figure 8(g) portrays it, and Figure 8(h) reveals the decrypted video file.

3.1.5. Key storage database

In the Key Storage Database module illustrated in Figure 9, the system systematically stores and manages keys, establishing a centralized repository for efficient key handling and ensuring systematic organization within the system. It is essential to emphasize that the current system does not support the editing of keys. This intentional restriction is in place to maintain the security and integrity of the system, as any modification to keys would require decrypting and re-encrypting associated files. Unauthorized removal or editing of keys by users without the corresponding ID is restricted to enhance overall system security.

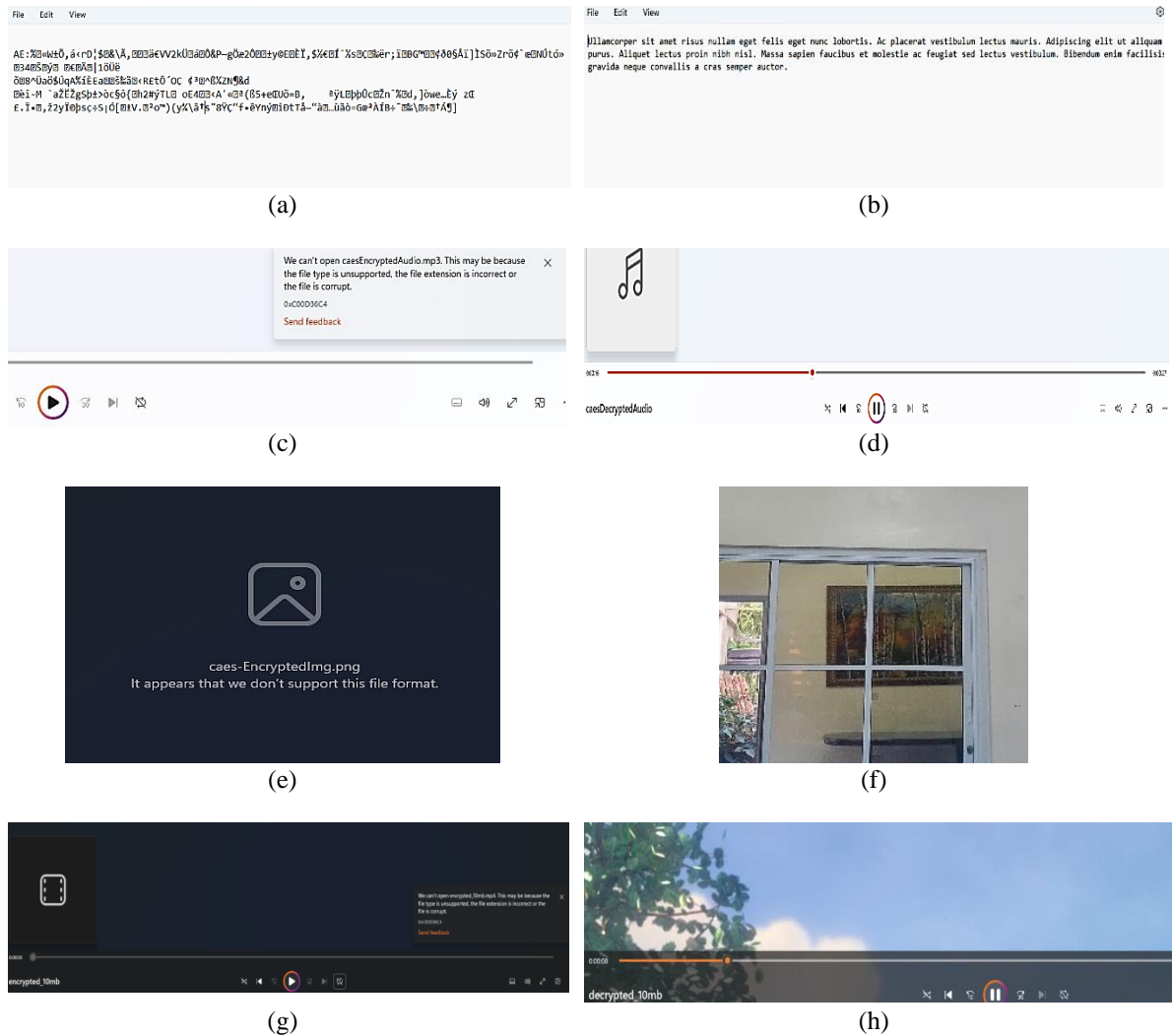


Figure 8. Encrypted and decrypted files visualization in: (a) text encrypt, (b) text decrypt, (c) audio encrypt, (d) audio decrypt, (e) image encrypt, (f) image decrypt, (g) video encrypt, and (h) video decrypt

Search

Filename	Key Used	Date Uploaded
Docs	12bac2b0f9a8b7bbca98d..	Feb 10, 2024
Doc Key	e8bec4c3f9a8b4bqc2a9e..	Feb 11, 2024
Doc Key	47ace679c2qff22mtbaz84..	Feb 11, 2024
Doc Key	117cc4q2d9eeb743c1295q..	Feb 11, 2024

Figure 9. Key storage database module

3.1.6. Share file module

In Figure 10, the depicted file management system exhibits a robust file sharing feature, enhancing collaborative efforts by allowing users to access shared files. This functionality not only promotes collaborative viewing but also provides a user-initiated sharing process, wherein opting for the “Share a File” option triggers the dialog box associated with this feature. While selecting “Cancel” closes the dialog box, ensuring user control over the sharing process.

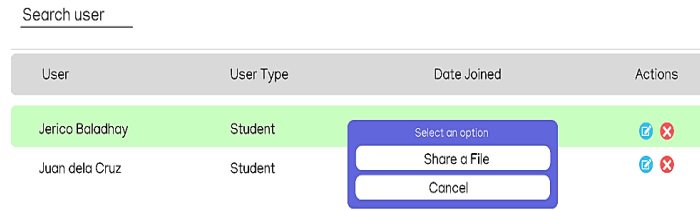


Figure 10. File sharing module

3.2. Time consumption

This study investigated the effects of the RRPBA on encryption and decryption processes. Table 1 showcases significant improvements in both encryption and decryption processes with the RRPBA. Notably, the adapted algorithm exhibits a 38.8% acceleration in encryption time and a remarkable 44.86% improvement in decryption time, emphasizing its pronounced speed advantage.

The study stands out by providing a comprehensive analysis of the RRPBA's impact on encryption and decryption durations. These results align with the consensus in existing literature, indicating that optimized algorithms, like the RRPBA, can substantially enhance processing speed. Despite the positive outcomes, it is essential to acknowledge the limitations, including the specific conditions and configurations under which the evaluations took place. Recognizing these constraints ensures a nuanced understanding of the algorithm's performance.

The evident improvements in encryption and decryption speed hold practical implications for file operations within the system. Additionally, these findings pave the way for further exploration into the adaptability and scalability of the modified algorithm across diverse computing environments and varying levels of computation resources. In summary, our study demonstrates the substantial acceleration achieved by the Reduced-Round Permutation-Based AES in encryption and decryption processes. The pronounced speed advantage positions the adapted algorithm as a crucial element for efficient file operations within the file management system. While contributing to the optimization discourse, it is imperative to recognize the study's limitations and set the stage for future research exploring the algorithm's adaptability and scalability in diverse computing environments.

Table 1. Time consumption AES vs. modified AES

File	File Type	Size in MB	AES		Modified AES	
			Encryption	Decryption	Encryption	Decryption
File 1	JPEG	10.2	590.0382 ms	693.6231 ms	289.7804 ms	406.6628 ms
File 2	TXT	10.5	410.9780 ms	517.4951 ms	208.5448 ms	313.4558 ms
File 3	WAV	21.2	1076.6240 ms	1506.5650 ms	628.1635 ms	917.2006 ms
File 4	PNG	32.9	1734.2740 ms	2381.2520 ms	1021.9310 ms	1367.3270 ms
File 5	MP4	48.3	2403.3270 ms	3520.5480 ms	1598.4450 ms	2059.6830 ms
File 6	FLAC	70.2	3422.3149 ms	7786.3630 ms	2533.3040 ms	3548.7760 ms
File 7	AVI	71.8	3801.0870 ms	9466.4010 ms	2688.4010 ms	4198.6990 ms

3.3. Throughput

In examining the impact of the modified algorithm on the file management system's throughput, we identify significant gaps in existing research. While encryption and decryption processes have been studied extensively, the investigation addresses the need for a more efficient algorithm to optimize file operations. Figure 11 aptly illustrates the transformative effects of our modified algorithm on the file management system's throughput. Notably, it demonstrates a remarkable 33.73% improvement for encryption and 23.72% for decryption, surpassing the original AES, which achieved 20.14% for encryption and 13.31% for decryption.

Our findings resonate with the growing consensus on the positive impact of algorithm optimization on processing speed. The comprehensive evaluation methodology, involving ten trials for both encryption and decryption, sets the study apart. It has been observed that the modified algorithm's throughput improvements align with the broader discourse on the efficacy of optimized algorithms. Acknowledging the limitations of this study is paramount. While our research offers valuable insights, it is crucial to recognize specific conditions that may influence the outcomes. Future in-depth studies are warranted to confirm the robustness of the modified algorithm, especially concerning its performance under diverse scenarios.

The positive outcomes regarding the modified algorithm's speed advantage hold promising implications for future high-performance applications. Subsequent studies may delve into exploring the algorithm's adaptability to varying conditions and the feasibility of integrating it into practical applications. As a final point, our study provides compelling evidence of the superior computational effectivity of the modified algorithm. The graphical representation facilitates a clear visual comparison, emphasizing its significant speed advantage in both encryption and decryption processes. This breakthrough underscores the algorithm's potential to optimize file operations and its positive implications for advancing high-performance applications.

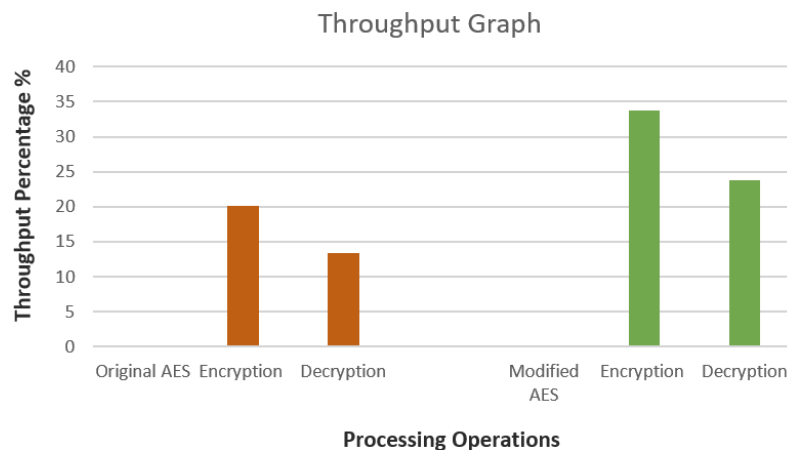


Figure 11. Throughput AES vs. modified AES

4. CONCLUSION

This research makes a substantial contribution to the domain of medium-to-large file encryption by introducing and exploring a modified algorithm based on Reduced-Round Permutation-Based AES. The study underscores the pivotal role of a file management system in ensuring the security of files for users. The outcomes reveal that the adapted algorithm significantly enhances computational effectiveness, with a notable 38.8% acceleration in encryption and a remarkable 44.86% improvement in decryption for medium to large-sized files. These findings hold significant implications for the research field and the broader community. The demonstrated performance of the algorithm in optimizing cryptographic processes, especially with substantial data volumes, positions it as a practical solution for large-scale file encryption. In future research endeavors, the focus will extend to optimizing overall system performance and integrating with emerging technologies such as quantum-resistant cryptography. Moreover, a thorough security analysis will be conducted to fortify the algorithm's resilience against evolving threats. These avenues of research collectively pave the way for further advancements in large file encryption, addressing the escalating demand for robust security solutions in our ever-evolving technological landscape. The study not only enhances our understanding of file encryption methodologies but also offers practical insights that can positively impact the broader research community and contribute to the ongoing evolution of secure data management systems.





REFERENCES

- [1] M. Caprolu, R. Di Pietro, F. Lombardi, and S. Raponi, "Edge computing perspectives: architectures, technologies, and open security issues," *Proceedings - 2019 IEEE International Conference on Edge Computing, EDGE 2019 - Part of the 2019 IEEE World Congress on Services*, pp. 116–123, 2019, doi: 10.1109/EDGE.2019.00035.
- [2] K. Ishii, M. M. Lyons, and S. A. Carr, "Revisiting media richness theory for today and future," *Human Behavior and Emerging Technologies*, vol. 1, no. 2, pp. 124–131, 2019, doi: 10.1002/hbe2.138.
- [3] B. Seth, S. Dalal, V. Jaglan, D. N. Le, S. Mohan, and G. Srivastava, "Integrating encryption techniques for secure data storage in the cloud," *Transactions on Emerging Telecommunications Technologies*, vol. 33, no. 4, 2022, doi: 10.1002/ett.4108.
- [4] M. R. D. Rahardjo and G. F. Shidik, "Design and implementation of self encryption method on file security," *Proceedings - 2017 International Seminar on Application for Technology of Information and Communication: Empowering Technology for a Better Human Life, iSemantic 2017*, vol. 2018-January, pp. 181–186, 2017, doi: 10.1109/ISEMANTIC.2017.8251866.
- [5] P. Yang, N. Xiong, and J. Ren, "Data security and privacy protection for cloud storage: a survey," *IEEE Access*, vol. 8, pp. 131723–131740, 2020, doi: 10.1109/ACCESS.2020.3009876.
- [6] B. P. Bhattarai *et al.*, "Big data analytics in smart grids: State-of-the-art, challenges, opportunities, and future directions," *IET Smart Grid*, vol. 2, no. 2, pp. 141–154, 2019, doi: 10.1049/iet-stg.2018.0261.




- [7] S. R. Maniyath and V. Thanikaiselvan, "A novel efficient multiple encryption algorithm for real time images," *International Journal of Electrical and Computer Engineering*, vol. 10, no. 2, pp. 1327–1336, 2020, doi: 10.11591/ijece.v10i2.pp1327-1336.
- [8] Y. Alemami, M. A. Mohamed, and S. Atiewi, "Advanced approach for encryption using advanced encryption standard with chaotic map," *International Journal of Electrical and Computer Engineering*, vol. 13, no. 2, pp. 1708–1723, 2023, doi: 10.11591/ijece.v13i2.pp1708-1723.
- [9] S. Devi and H. D. Kotha, "AES encryption and decryption standards," *Journal of Physics: Conference Series*, vol. 1228, no. 1, 2019, doi: 10.1088/1742-6596/1228/1/012006.
- [10] R. Lavanya and M. Karpagam, "Enhancing the security of AES through small scale confusion operations for data communication," *Microprocessors and Microsystems*, vol. 75, p. 103041, Jun. 2020, doi: 10.1016/j.micpro.2020.103041.
- [11] K. Muttaqin and J. Rahmadoni, "Analysis and design of file security system AES (advanced encryption standard) cryptography based," *Journal of Applied Engineering and Technological Science*, vol. 1, no. 2, pp. 113–123, 2020, doi: 10.37385/jaets.v1i2.78.
- [12] G. Kapil, A. Agrawal, A. Attaallah, A. Algarni, R. Kumar, and R. A. Khan, "Attribute based honey encryption algorithm for securing big data: Hadoop distributed file system perspective," *PeerJ Computer Science*, vol. 2020, no. 2, pp. 1–31, 2020, doi: 10.7717/peerj-cs.259.
- [13] S. Mukherjee, "Overview of the importance of corporate security in business," *SSRN Electronic Journal*, 2019, doi: 10.2139/ssrn.3415960.
- [14] A. M. Maigida, S. M. Abdulhamid, M. Olalere, J. K. Alhassan, H. Chiroma, and E. G. Dada, "Systematic literature review and metadata analysis of ransomware attacks and detection mechanisms," *Journal of Reliable Intelligent Environments*, vol. 5, no. 2, pp. 67–89, 2019, doi: 10.1007/s40860-019-00080-3.
- [15] J. He, S. Huang, S. Tang, and J. Huang, "JPEG image encryption with improved format compatibility and file size preservation," *IEEE Transactions on Multimedia*, vol. 20, no. 10, pp. 2645–2658, 2018, doi: 10.1109/TMM.2018.2817065.
- [16] K. Patel, "Performance analysis of AES, DES and Blowfish cryptographic algorithms on small and large data files," *International Journal of Information Technology (Singapore)*, vol. 11, no. 4, pp. 813–819, 2019, doi: 10.1007/s41870-018-0271-4.
- [17] E. M. de Los Reyes, A. M. Sison, and R. P. Medina, "File encryption based on reduced-round AES with revised round keys and key schedule," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 16, no. 2, pp. 897–905, 2019, doi: 10.11591/ijeecs.v16i2.pp897-905.
- [18] H. V. Gamido, M. V. Gamido, and A. M. Sison, "Developing a secured image file management system using modified AES," *Bulletin of Electrical Engineering and Informatics*, vol. 8, no. 4, pp. 1461–1467, 2019, doi: 10.11591/eei.v8i4.1317.
- [19] S. Al-Saqqa, S. Sawalha, and H. Abdelnabi, "Agile software development: methodologies and trends," *International Journal of Interactive Mobile Technologies*, vol. 14, no. 11, pp. 246–270, 2020, doi: 10.3991/ijim.v14i11.13269.
- [20] J. W. Tabor, "Chaos: exploring an engaging online model for rapid application during the pandemic," *Educational Technology Research and Development*, vol. 69, no. 1, pp. 97–100, 2021, doi: 10.1007/s11423-020-09878-y.
- [21] D. Palupiningtyas, A. Dewi Maria, T. Adhistyo Wijoyo, A. Prarasdya Alyka, and K. Z. Putri Brawarso, "Application of rapid application development method in designing knowledge management system to improve employee knowledge and performance at ministry of agriculture," *Journal of Information and Technology*, pp. 29–35, Jan. 2024, doi: 10.60083/jidt.v6i1.468.
- [22] J. S. Baladhay and E. M. De Los Reyes, "AES-128 reduced-round permutation by replacing the MixColumns function," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 33, no. 3, p. 1641, 2024, doi: 10.11591/ijeecs.v33i3.pp1641-1652.
- [23] H. T. Assafli and I. A. Hashim, "Security enhancement of AES-CBC and its performance evaluation using the avalanche effect," *2020 3rd International Conference on Engineering Technology and its Applications, IICETA 2020*, pp. 7–11, 2020, doi: 10.1109/IICETA50496.2020.9318803.
- [24] S. S. Abdul-Jabbar, A. E. Abed, S. G. Mohammed, and F. G. Mohammed, "Fast 128-bit multi-pass stream ciphering method," *Iraqi Journal of Science*, vol. 64, no. 5, pp. 2589–2600, 2023, doi: 10.24996/ijs.2023.64.5.40.
- [25] L. Lapworth, "Parallel encryption of input and output data for HPC applications," *International Journal of High Performance Computing Applications*, vol. 36, no. 2, pp. 231–250, 2022, doi: 10.1177/10943420211016516.
- [26] S. A. Bafna, "Review on study and usage of MERN stack for web development," *International Journal for Research in Applied Science and Engineering Technology*, vol. 10, no. 2, pp. 178–186, 2022, doi: 10.22214/ijraset.2022.40209.
- [27] S. Chen, U. R. Thaduri, and V. K. R. Ballamudi, "Front-end development in react: an overview," *Engineering International*, vol. 7, no. 2, pp. 117–126, 2019, doi: 10.18034/ei.v7i2.662.

BIOGRAPHIES OF AUTHORS






Jerico S. Baladhay     he is a dedicated researcher; he has been actively involved in the field since 2020. Previously, he served as an Information Technology College Instructor at Dominican College of Tarlac under the department of College of Computer Studies, as well as a software developer at the same institution under the Management Information Systems Office from 2020. Currently pursuing a master's degree in Information Technology at Tarlac State University, his expertise encompasses software development, with a specialization in web applications, mobile applications, and game development. He holds skills certifications from completing Cisco Certified Network Associate Version 7 (CCNAv7) courses in Networking. Moreover, he is a law student at Tarlac State University, engaging deeply in the vibrant nexus of legal studies and technology within the academic environment of this esteemed institution. Notably, he has already contributed to the field, having published a paper in the domain of data security and encryption algorithms. His research interests include data security and encryption algorithms. He can be contacted at email: j.baladhay0785@student.tsu.edu.ph.



Heidilyn V. Gamido    holds a doctorate in information technology from the Technological Institute of the Philippines, Quezon City, attained through the CHED K-12 transition program scholarship. Presently serving as an Associate Professor within the College of Computer Studies at Tarlac State University, she concurrently assumes the role of Director at the Management Information Systems Office since 2014. In addition to her academic achievements, she has garnered skills certifications, including MOS, ICDL, and Network Security Associate. Her research pursuits encompass areas such as data security, image processing, and information systems. She can be contacted at email: htvgamido@tsu.edu.ph.



Edjie M. De Los Reyes    He holds the distinguished position of Research Director and currently serves as an Associate Professor IV at Tarlac State University, accumulating two decades of dedicated service in the realm of academics. With a robust background in academia, he assumed the role of the university's Research Director, having previously held the position of Associate Dean from 2014 to 2016. His extensive skill set is underscored by various certifications, including Cisco Certified Network Associate, Cisco Certified Academic Instructor, Microsoft Office Specialist, and Electronic Data Processing Specialist-Programmer. Actively participating in academic and research communities, he is a valued member of prestigious organizations such as the Philippine Society of Information Technology Educators (PSITE), Philippine Schools Universities and Colleges Computer Education and Systems Society (PSUCCESS), and the International Association of Multidisciplinary Research. Additionally, he has contributed significantly to the body of knowledge, publishing numerous research papers in Scopus-indexed journals with a particular emphasis on data security. He can be contacted at email: emdelosreyes@tsu.edu.ph.