International Journal of
INTELLIGENT SYSTEMS AND APPLICATIONS IN ENGINEERING

ISSN:2147-6799 www.ijisae.org Original Research Paper

CryptShareX: Secure File Sharing Platform with AES and Steganography

Vaishali Savale[1], Balamurali Nambiar, Vedashri Chaudhari, Manthan Vasant, Sanskruti Umak, Ajay Uikey

Submitted: 27/01/2024 **Revised**: 05/03/2024 **Accepted**: 13/03/2024

Abstract: In the evolving landscape of contemporary data exchange, the critical need to simultaneously ensure efficiency and security in cloud-based file transfers takes precedence. This research delves into the nuanced intricacies of file transfer mechanisms, proposing a holistic approach characterized by dual encryption through advanced algorithms like AES and RSA. Through the integration of cryptographic techniques, coupled with the infusion of steganography, this study strives to establish a resilient framework. Its aim is not only to bolster the security of cloud-based file transfers but also to facilitate an effective and seamless data exchange process within the dynamic cloud environment. The amalgamation of these encryption methodologies, strategically complemented by the incorporation of steganographic principles, represents an innovative stride towards achieving an optimal equilibrium between data protection and the streamlined operations of file transfers in the cloud.

Keywords: *Cryptography, Advanced Encryption Standard Algorithm (AES), Rivest Shamir Adleman Algorithm (RSA), Steganography*

1. Introduction

Ensuring the security of data transfer is imperative in the contemporary landscape to protect sensitive information from unauthorized access, uphold privacy, mitigate cyber threats, and preserve the integrity of digital communications. This is indispensable for safeguarding individuals, businesses, and organizations against potential financial or reputational harm arising from data breaches. The growing importance of data as a crucial asset for organizations results from advancements in technology, including sensor systems, IoT, cloud computing, and data analytics. Despite the extensive research conducted on data security and privacy in the past thirty years, new challenges have emerged. These challenges are driven by increased concerns about privacy, particularly in areas like homeland protection, counterterrorism, and health security, where finding a balance between privacy and data utilization is essential. Moreover, the widespread integration of IoT devices has expanded the scope of potential vulnerabilities and risks in the realm of data security [1].

In the present context, the inherent significance of data as an essential asset is beyond dispute. Acknowledging its pivotal role, we have judiciously integrated advanced methodologies, specifically cryptography and steganography. This comprehensive approach is designed to effectively support the fundamental goal of ensuring the secure transmission of data, thereby mitigating the inherent risks associated with potential breaches or unauthorized access. Through the implementation of this amalgamated framework, our aim is to bolster both the confidentiality and integrity of data, establishing a resilient and seamless conduit for information transfer within the intricate and ever-evolving landscape of the contemporary world.

In the dynamic landscape of our modern world, safeguarding data integrity holds paramount importance. The growing adoption of cloud storage by individual users is evident, enticed by the accessibility of free services and economical alternatives such as platforms like sync.com, OneDrive etc. However, this convenience introduces a caveat—a potential compromise in service reliability. The cost-effectiveness of these cloud services raises pertinent questions regarding the trustworthiness of providers, intensifying the call for enhanced data protection. The cloud environment, susceptible to various recognized attacks, exposes vulnerabilities leading to privacy infringements and security breaches [6]. In year 2022, 45% of enterprises faced a cloud-based data breach or audit failure in 12 months, indicating a 5% rise from the preceding year [7]. This upsurge emphasizes growing concerns related to the protection of confidential data from cyber threats. Instances like in the year 2021, LinkedIn faced a data scraping incident affecting 700 million profiles. Although the compromised data was predominantly public, the situation took a disconcerting turn when the information from the breach emerged on a dark web forum in June of the same year. [8]. Consequently, users are driven to secure their data either by retaining it on personal computers or embracing encryption algorithms as viable and robust solutions.

Cryptography is a fundamental pillar within the domain of

[1,2,3,4,5,6]*Vishwakarma Institute of Technology, Pune*
vaishali.savale@vit.edu, balamurali.nambiar21@vit.edu, shekhar.vedashri21@vit.edu, manthan.vasant21@vit.edu, sanskruti.umak21@vit.edu, ajay.uikey21@vit.edu

secure data communication, offering a sophisticated and mathematical framework for encoding information, thereby making it indecipherable to unauthorized entities. This intricate field of secure communication incorporates diverse algorithms and protocols, each meticulously crafted to guarantee the confidentiality, integrity, and authenticity of transmitted data. Cryptography utilizes complex mathematical algorithms to convert data into a cipher, rendering it incomprehensible to unauthorized entities. This involves employing cryptographic keys, such as symmetric keys for shared secrets and asymmetric keys for public-private pairs. During the encryption phase, data is transformed into ciphertext, securely transmitted, and then decrypted by the recipient using the corresponding key. This comprehensive process guarantees confidentiality and safeguards data integrity through the incorporation of hash functions or digital signatures [2].

Steganography is a technology that hides sensitive data from detection by embedding it within seemingly innocent carriers, such text, audio files, or photographs. To guarantee that the concealed data is safe and unreadable in the absence of the decryption key, steganography techniques sometimes include encryption. Furthermore, methods like error correction and data compression can be applied to raise the effectiveness and dependability of steganographic systems. Steganography techniques are still being developed for use in clandestine communication and digital security as both offensive and defensive tools. Steganography, an intricate technique in information security, conceals confidential data within inconspicuous carrier mediums to avoid detection. Unlike cryptography, which encrypts data, the primary goal of steganography is to obscure the presence of hidden information. This involves discreetly embedding sensitive data within less perceptible components of a medium, such as manipulating pixels in images or adjusting text spacing. The recipient, informed of the method, can retrieve the information using a decoding algorithm. Steganography is employed in secure communication, digital watermarking, and covert information exchange, leveraging digital file features as covert channels. Its ongoing development tackles detection challenges, striking a careful balance between effective information hiding and evading detection by adversaries.[3]

This scholarly investigation explores the intricate domain of file transfer mechanisms, presenting an all-encompassing strategy characterized by dual encryption employing cutting-edge algorithms, including AES and RSA. The research endeavours to forge a resilient framework by integrating cryptographic techniques and enhancing them with steganography. Its primary objective is to fortify the security of file transfers within the cloud, concurrently ensuring an efficient and smooth data exchange milieu. The amalgamation of these encryption methodologies, coupled with the strategic integration of steganographic principles, represents an innovative initiative aimed at attaining an ideal equilibrium between safeguarding data and optimizing cloud-based file transfer operations.

2. Related Work

The following research papers were studied for the state-of-the-art comparison between various techniques and methods. First five papers have implemented data encryption using AES algorithm and remaining four papers have implemented steganography methods.

K. Jaspin [4] offers a unique method of file protection for cloud-stored files by using the AES and RSA algorithms for double encryption. With a key length of 128 bits, the author employs AES128 to fend off a quantum assault with 64 bits of strength. The Dropbox cloud is where the encrypted files are kept safe. It takes two seconds to encrypt data using AES and RSA, and one second to decode it using DES. Several factors, including encryption speed, security, data secrecy, data correctness, and cipher text, are used to evaluate the model. As a future research project, the authors use deep learning and machine learning to explain cryptography. It focuses primarily on the AES and RSA algorithms for double encryption, ignoring other encryption techniques that might increase security. Furthermore, the study does not fully examine the potential performance impact—in terms of additional processing cost or delay—that double encryption deployment in a cloud context may have. Furthermore, it was not possible for the research to fully analyze all possible gaps or attacks that may exploit weaknesses in the recommended double encryption strategy.

A file security system that uses the AES algorithm for encryption and LPC coefficients as the key for file decryption is proposed by Lian Li [5] and JunHu Kuai. The AES algorithm receives the derived linear predictive coding coefficients from the audio file. Since voice transmissions are linear in form, encryption uses both the verbal signal pattern and the LPC technique. A 16-dimensional hybrid LPC is produced by adding 4 more dimensions from the spoken signal pattern to the 12 dimensions of LPC coefficients. The extraction of LPC coefficients requires a specific file format (.txt) and audio file, which limits the extent of generalizability.

Two phases are used by the authors Flevina [6] and her co-author to present an AES security improvement technique. The first step is to generate a Dynamic Key, which jumbles the data and adds complexity. The second phase is to put Dynamic S-Box Generation into practice, which makes it harder for attackers to breach a predetermined set of static S-boxes. The encryption key is produced dynamically, considering the sender's login time. The outcome of performing XOR operations on every byte in the round key yields the shift value that is utilized for the circular shift on the S-box. Predictability in the key can be introduced by time-dependent key creation. The study paper's main flaw is that it doesn't use the recommended mitigating methods.

Two designs for the AES-128 algorithm with 10 rounds and a key expansion module producing 10 keys are presented by Shady Mohamed Soliman [7]. The primary goal is to minimize both the space and power consumption so that it may be used with IoT low power modules. The first design reduces the number of pipeline registers for low power consumption by combining pipelining rounds with iterative looping. In the second approach, the data is processed five times over the course of two rounds and two key schedules. Between these rounds and key schedules, pipeline registers are used to store temporary data as it passes through several pipeline stages. The drawback is that the designs are implemented for the AES-128 algorithm and synthesized on specific FPGA (Field Programmable Gate Array) devices reducing its general applicability and they have not thought-out countermeasures or fully investigated the effects of side-channel assaults.

G. Peng [8] uses FPGA to improve the conventional AES-128-bit algorithm structure, paying particular attention to the encryption/decryption and key expansion parts. In order to increase overall efficiency, the proposed approach divides the encryption process into phases and uses pipeline technology for parallel processing. The system was tested in the real world by building a PCB circuit, and it was compiled using the Quartus tool. Because FPGA devices are costly and limited, the suggested method is not as useful.

The difficulties with digital data security are highlighted by M. Kumar [9] and co-authors, who also emphasize how crucial it is to protect the validity of this data. For the encryption of test data and digital images, respectively, LSB steganography and AES cryptography are used. The hybrid model underwent evaluation based on certain metrics, including Mean Square Error and Peak Signal-to-Noise Ratio. The results indicated that the secret data could be successfully hidden inside the image with minimum distortion and limited visibility, owing to low MSE and high PSNR. However, the amount of secret data that can be hidden in a picture is limited and depends on the LSBs that are accessible.

The authors [10] provide a safe picture data transmission approach that combines the differential expansion method with two well-known encryption algorithms (RSA and AES). The enciphered secret key is embedded into a cipher picture using the differential expansion NNPL (Non-Parametric-Patch-based Local) method, which involves adjusting the image's pixel values while minimizing the noticeable changes to the image. Because of this, it is more difficult for attackers to view the plaintext picture without the enciphered key and cipher text. The suggested model only supports one type of picture format, although the key file sizes might range from 25 to 97.

Odd/even pixel allocation is used in the authors' technique [11] to improve the image's security and imperceptibility. To do this, there are four main phases. Step 1 is picture Preparation, which is dividing the cover picture into blocks and choosing pixels for embedding from inside these blocks. Huffman coding is used in step two to reduce the text's size before embedding it to expand the data capacity. Step 3 is hiding the encrypted message behind the cover image's LSB layer. In the final phase, Step 4, steganography keys, Hénon map function, and Huffman coding are used to extract information from the image. The system has excellent PSNR values, indicating strong performance.

The authors [12] provide three techniques for video steganography, including fuzzy logic and the secure base LSB approach, enabling secure communication. Data is hidden within the video data's least significant bits (LSB) using the LSB technique. Input and output variables, as well as functions and factors for fuzzification and defuzzification, are all part of fuzzy logic. The authors propose that this strategy enhances the resilience of steganography methods. The PSNR and MSE measures are used to assess the provided techniques. One potential area of future research for this study is the integration of new algorithms to improve security.

3. Methodology

The platform CryptShareX provides a comprehensive answer to the intricate security problems associated with file sharing in cloud settings. Through the integration of advanced cryptographic methods like AES algorithms with steganography, the platform guarantees strong security for confidential information while it is being sent. AES encryption improves data security and integrity by generating a 128-bit secret key and using numerous encryption cycles. Furthermore, the platform's cutting-edge steganographic techniques, such as encrypting bespoke pictures and QR codes with secret keys, increase concealment and make it more difficult for unwanted parties to access or discover encrypted data. This all-inclusive strategy improves data security while enabling smooth and effective file sharing in dynamic cloud environments. Additionally, GoFile Server integration makes it easier to store and retrieve encrypted data, guaranteeing user convenience without sacrificing security. CryptShareX is a potential answer to the pressing demand for safe data sharing in contemporary cloud environments. It excels in its skillful synthesis of encryption and concealment approaches.

The developed system aims to combine the strengths of the following to develop a comprehensive solution for secure file encryption and concealment:

A.AES Algorithm

B. Steganography

C. GoFile Server

3.1. AES Algorithm

Sensitive data and information are encrypted and decrypted using the symmetric encryption method known as the Advanced Encryption Standard (AES). To generate a secret key that is required for both data encryption and decryption, utilize the 'Fernet' library. Symmetric encryption is utilized, which means that the encryption and decryption processes share the same key. The data from the input file is transformed into cipher text, which is a random bit sequence that cannot be deciphered without a secret key. The input file's data and information are encrypted using a secret key with a bit size of 128. To further secure the data, the plain text is subjected to ten rounds of XOR operations using the secret key. Four main operations make up each round: SubBytes, Rows Shift

3.1.1. SubBytes

A predetermined substitution table known as the S-Box is used to match each input byte with a matching byte. This increases the data's nonlinearity, which makes it more challenging to evaluate.

3.1.2. ShiftRows

Every row is cyclically moved to the left, except for the first row. This technique makes the data more erratic while it is being encrypted.

3.1.3. ShiftColumns

To add diffusion and guarantee that every byte of the output relies on every byte of the input, each column is multiplied by a fixed polynomial, modulo another fixed polynomial.

3.1.4. RoundKey

The secret key for the current round is XORed with the input bytes. This makes sure that every cycle utilizes a distinct set of key bits and adds key material to the encryption process.

The same procedures must be followed in reverse order, beginning with the last round, and working backward through the rounds to decrypt the encrypted data. Round keys are employed in the decryption rounds and are obtained from the initial encryption key. The stages in the encryption process are reversed for each decryption cycle. If the right decryption key is employed, the decrypted plaintext that is acquired at the conclusion of the procedure ought to match the original plaintext that was before encryption. The AES algorithm's system diagram is shown in Figure 1.

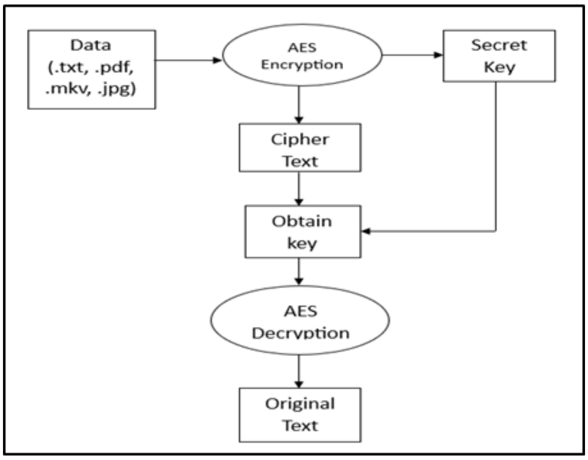


Fig 1. System Diagram of AES Data Encryption

3.2. Image based Steganography

Steganography is the process of secretly hiding a file, video, picture, or message inside another file, image, or video such that spectators cannot see it. One popular type of steganography is text hiding, in which text is hidden inside an image file. Two techniques for steganography-based data encryption are employed in this system:

I. Encryption using QR code

II. Encryption of data into selected images.

3.2.1. Encryption using QR

In this encryption method, the input data is encrypted using AES algorithm and the secret key generated is concealed in the QR code image generated. This method is used to enhance security of the secret key as for every input file to be encrypted, the QR code generated is unique.

The file at first is encrypted using the AES algorithm to generate the secret key. The secret key generated by the AES algorithm is hidden within the QR code using the ‘pillow’ library. The python library ‘qrcode’ is utilized in order to generate the QR code images for each input file. It is then converted into binary format such that each character in the text is represented as an 8-bit (1 byte) ASCII code. To encode the key onto the QR code the Least Significant Bit (LSB) algorithm is applied. In this method, the least significant bit of each colour channel in the QR image is replaced with a bit from the key. Since the least significant bit contributes the least to the overall colour intensity, modifying it has minimal impact on the visual appearance of the image, hence making it imperceptible to the human eye. This image is to be saved and is passed through the decryption algorithm. The extracted LSBs are concatenated to reconstruct the binary representation of the original key. This binary sequence corresponds to the ASCII values of the characters of the key that were encoded into the image. Figure 2 represents the system diagram for the steganography algorithm.

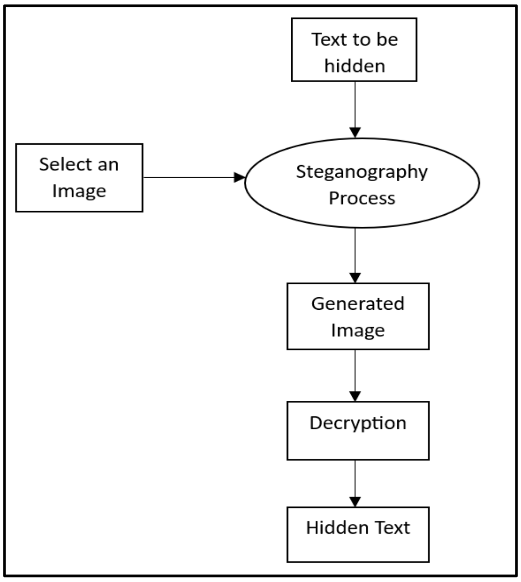


Fig 2. System Diagram of Steganography

3.2.2. Encryption of data into selected images

Carrier files or cover mediums, are in formats like JPEG, PNG, or BMP, and serve as the camouflage for the hidden message. Before embedding the message, it is encoded in a plaintext or binary data format. The embedding process involves subtly altering the pixel values of the cover medium, by modifying the least significant bit (LSB) of each pixel. This alteration must be imperceptible to the human eye to maintain the visual integrity of the cover medium. During decryption, the custom image is subjected to a reverse process. LSB extraction is performed, gathering the bits necessary to reconstruct the original secret key. This reconstructed key is then employed to decrypt the desired data encrypted with AES. This method bolsters security by utilizing the unique characteristics of the custom image to safeguard the secret key, thereby strengthening the encryption process against potential threats. Main functions used in this process are mentioned below.

‘convert_to_binary’(message): This function converts the message into its binary representation.

‘next_pixel’(cover_image): This function retrieves the next pixel from the cover image.

‘modify_LSB’(pixel, bit): This function modifies the least significant bit (LSB) of the pixel to the specified bit value.

‘extract_LSB’(pixel): This function extracts the LSB of the pixel.

‘convert_to_original_format’(binary_message): This function converts the binary message back to its original format

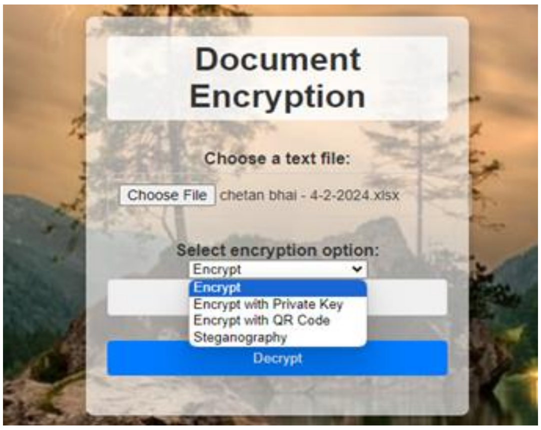
3.3. GoFile Server

[13] GoFile Server is a predominantly used platform for file sharing. It allows the user to store confidential data for long term purposes. This platform is used by the system to store and save every encrypted and decrypted file uploaded and requested respectively. After requesting to encrypt a file, a download link of the encrypted file is generated for the user. The same is done for accessing the decrypted file.

4. Results

The web application was integrated with HTML / CSS to have a better user interface. At first, the user is given an option to upload any file to encrypt or decrypt. For encryption, the user is given four choices:

- Encrypt
- Encrypt with private key
- Encrypt with QR code
- Steganography


Document Encryption

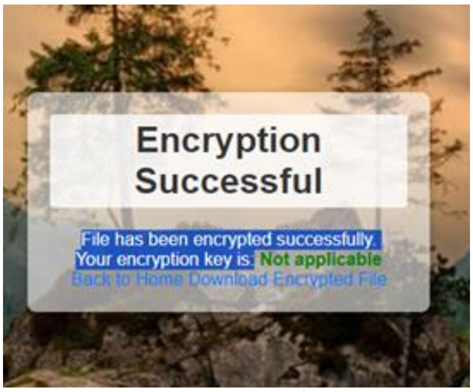
Choose a text file:

chetan bhai - 4-2-2024.xlsx

Select encryption option:

Fig 3. Different methods of Encryption used in the system

In Fig 3. ‘Encrypt’ option can be used by the user to only provide users with a straightforward method to encrypt their files using the AES encryption algorithm. According to Fig 4. the system does not provide the user with a decryption key. This means that once the file is encrypted, it cannot be decrypted using the tool alone. The encrypted file remains secure and inaccessible without the appropriate decryption key.

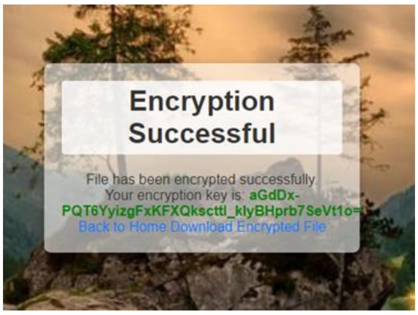

Encryption Successful

File has been encrypted successfully.

Your encryption key is: Not applicable

Fig 4. Encrypt: Only Encrypts file cannot be decrypted

The ‘Encrypt with Private Key’ option in the system provides users with a means to encrypt their files using the AES encryption algorithm while also allowing for decryption using a secret key provided by the user. In Fig 5. Unlike the latter option where no decryption key is provided, the 'Encrypt with Private Key' option allows the user to specify a decryption key. This means that the user can decrypt the encrypted file using the same secret key that was used for encryption.

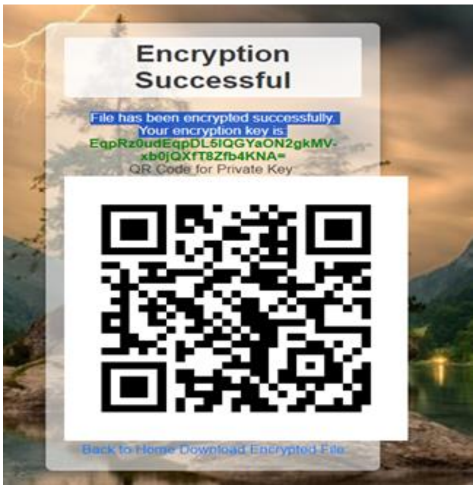


Encryption Successful

File has been encrypted successfully.
Your encryption key is: aGdDx-PQT6YyizgFxKFXQkscttl_klyBHprb7SeVt1o=

Fig 5. Encrypt with Private Key

The 'Encrypt with QR Code' option offers users a unique and innovative method to encrypt their files using AES encryption while embedding the secret key within a QR code image. This approach combines the security of AES encryption with the concealment capabilities of steganography. As seen in Fig 6. once the file is encrypted and the secret key is generated, the secret key is embedded within a QR code image.



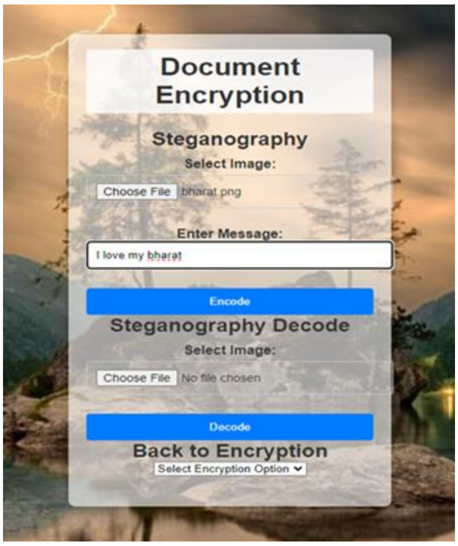
Encryption Successful

File has been encrypted successfully.
Your encryption key is: EqpRz0udEqpDL5lQGYaON2gkMV-xb0jQXfT8Zfb4KNA=

QR Code for Private Key

Fig 6. Encryption of file using QR code

To decrypt the encrypted file, users can scan the QR code image using a compatible QR code scanner. The scanned QR code reveals the embedded secret key, which can then be used to decrypt the encrypted file and retrieve the original contents.



Document Encryption

Steganography

Select Image: Choose File bharat.png

Enter Message: I love my bharat

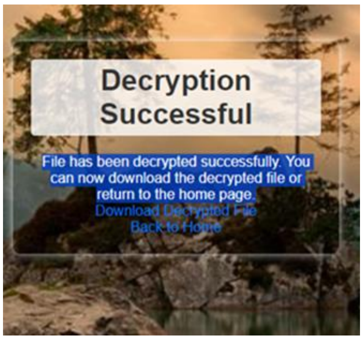
Steganography Decode

Select Image: Choose File No file chosen

Fig 7. Uploading File for Steganography Encryption

In Fig 7. 'Steganography' option provides users with a method to hide the data from their input file within a custom image. Users begin by selecting both the file they wish to encrypt and conceal, along with the message they want to hide. Through advanced steganographic techniques, the plaintext data is embedded to the custom image making imperceptible alterations to its visual elements.

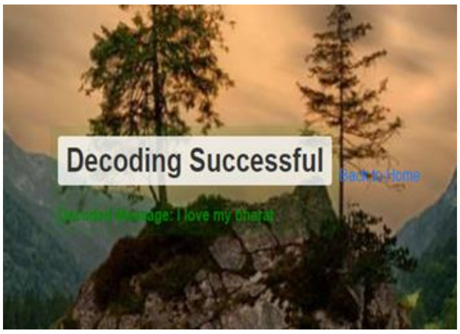
At the same-time, for decrypting and decoding the encrypted files, three corresponding modes are provided. In Fig 8, and Fig 9 users can download decrypted files through the GoFile Server link generated at the end of the process and extract hidden messages from images respectively.



Decryption Successful

File has been decrypted successfully. You can now download the decrypted file or return to the home page.

Fig 8. Decryption for 'Encryption with Private key','QR code'



Decoding Successful

Decoded Message: I love my bharat

Fig 9. Decoding message using Steganography

Also, the computation complexity of the encryption algorithms implemented in the application is examined. The primary encryption scheme utilized is the Fernet symmetric key encryption from the cryptography library.

Experiments are conducted to measure the time required for encryption under various conditions. File sizes are systematically varied to get the results. Specifically, the 'cryptography.hazmat.primitives.ciphers' module was employed to execute AES encryption. This process involved multiple stages, starting with the generation of a random key and Initialization Vector (IV). Following this, the data was encrypted using AES in CBC (Cipher Block Chaining) mode. The results of the encryption time analysis are presented in Table 1.

Table 1. Example of full-page table

Sno	*File Size (MB)*	*Time taken for AES encryption*
1	1	0.002039
2	20	0.045012
3	100	0.213728
4	1000	10.413033

5. References and Footnotes

5.1. References

References need not be cited in text. When they are, they appear on the line, in square brackets, inside the punctuation. Multiple references are each numbered with separate brackets. When citing a section in a book, please give the relevant page numbers. In text, refer simply to the reference number. Do not use "Ref." or "reference" except at the beginning of a sentence: "Reference [3]

5. Conclusion

The solutions presented in this paper have proven an important step towards the changing issues with protecting sensitive data. The model encrypts different types of data formats such as txt file, pdf, audio file, video file, etc. The model is deployed into a website for easy and handy access for users to encrypt and decrypt the sensitive or important data. Users can upload a file of maximum 256MB for encryption. The key length can be increased to 192 and 256 bits to significantly increase the security level and therefore reducing the chances of brute force attacks near to zero. Using more algorithms to increase the robustness at each level of encryption can be the possible future scope of this study.

Acknowledgements

Our sincere thanks go to our advisor Prof. Vaishali Savale for their invaluable guidance and support. We are also thankful to our peers for their insightful discussions and constructive feedback.

Author contributions

Vaishali Savale: Writing - Original Draft, Review & Editing **Balamurali Nambiar:** Software Development: AES Algorithm Methodology. Writing- Literature Survey

Vedashri Chaudhari: Software Development**:** QR Code Methodology Writing - Literature Survey and Abstract.

Manthan Vasant: Software Development: AES Algorithm, Steganography, QR Code Methodology.

Sanskruti Umak: Writing- Introduction and Abstract, Software Development: QR Code Methodology.

Ajay Uikey: Writing - Conclusion, Review & Editing.

Conflicts of interest

The authors declare no conflicts of interest.

References

[1] E. Bertino, "Data Security and Privacy: Concepts, Approaches, and Research Directions," 2016 IEEE 40th Annual Computer Software and Applications Conference (COMPSAC), Atlanta, GA, USA, 2016, pp. 400-407, doi: 10.1109/COMPSAC.2016.89.

[2] Pradeep, Vijayan & Vaishali, & Shetty, Vandan & M, Varsha. (2022). A Review Paper on Network Security and Cryptography. International Journal of Advanced Research in Science, Communication and Technology. 133-138. 10.48175/IJARSCT-7081. .

[3] N. Subramanian, O. Elharrouss, S. Al-Maadeed and A. Bouridane, "Image Steganography: A Review of the Recent Advances," in IEEE Access, vol. 9, pp. 23409-23423, 2021, doi: 10.1109/ACCESS.2021.3053998..

[4] K. Jaspin, S. Selvan, S. Sahana and G. Thanmai, "Efficient and Secure File Transfer in Cloud Through Double Encryption Using AES and RSA Algorithm," 2021 International Conference on Emerging Smart Computing and Informatics (ESCI), Pune, India, 2021, pp. 791-796, doi: 10.1109/ESCI50559.2021.9397005.

[5] L. Li and J. Kuai, "File Encryption System Based on the Hybrid LPC Coefficient and AES Algorithm," 2020 International Conference on Wireless Communications and Smart Grid (ICWCSG), Qingdao, China, 2020, pp. 1-4, doi: 10.1109/ICWCSG50807.2020.00008. keywords: {Encryption;Erbium;Conferences;Wireless communication;Smart grids;Hybrid power systems;Data processing;hybrid LPC coefficient;AES algorithm;file encryption system},

[6] D'souza, Flevina & Panchal, Dakshata. (2017). Advanced encryption standard (AES) security enhancement using hybrid approach. 647-652. 10.1109/CCAA.2017.8229881.

[7] Mohamed, Shady & Magdy, Baher & Abd El Ghany, Mohamed. (2016). Efficient implementation of the AES algorithm for security applications. 206-210. 10.1109/SOCC.2016.7905466.

[8] G. Peng and S. Zhu, "FPGA Implementation of AES Encryption Optimization Algorithm," 2021 International Conference on Intelligent Transportation, Big Data & Smart City (ICITBS), Xi'an, China, 2021, pp. 650-653, doi: 10.1109/ICITBS53129.2021.00165.

[9] M. Kumar, A. Soni, A. R. S. Shekhawat and A. Rawat, "Enhanced Digital Image and Text Data Security Using Hybrid Model of LSB Steganography and AES Cryptography Technique," 2022 Second International Conference on Artificial Intelligence and Smart Energy (ICAIS), Coimbatore, India, 2022, pp. 1453-1457, doi: 10.1109/ICAIS53314.2022.9742942.

[10] B. Mahalakshmi, G. Deshmukh and V. N. L. N. Murthy, "Image Encryption Method Using Differential Expansion Technique, AES and RSA Algorithm," 2019 Fifth International Conference on Image Information Processing (ICIIP), Shimla, India, 2019, pp. 363-366, doi: 10.1109/ICIIP47207.2019.8985665.

[11] N. Manohar and P. V. Kumar, "Data Encryption & Decryption Using Steganography," 2020 4th International Conference on Intelligent Computing and Control Systems (ICICCS), Madurai, India, 2020, pp. 697-702, doi: 10.1109/ICICCS48265.2020.9120935.

[12] Abbas, N.A.F., Abdulredha, N., Ibrahim, R.K., and Ali, A.H. (2022). "Security and imperceptibility improving of image steganography using pixel allocation and random function techniques." International Journal of Electrical and Computer Engineering (IJECE), vol. 12, no. 1, pp. 694-705.

[13] https://gofile.io/, GoFile Server