I.J. Information Technology and Computer Science, **2026, 1, 14-41**
Published Online on February 8, 2026 by MECS Press (http://www.mecs-press.org/)
DOI: 10.5815/ijitcs.2026.01.02

EdSri: An Enhanced Approach Towards Optimizing Cloud Communication by Hybridizing Cryptography and Steganography with Huffman Compression Method

Edwin Xorsenyo Amenu*
Faculty of Computer Applications, Marwadi University, Morbi Road, 360 003, Rajkot, India
E-mail: eddynap@gmail.com
ORCID iD: https://orcid.org/0009-0006-9177-7553
*Corresponding author

Sridaran Rajagopal.
Faculty of Computer Applications, Marwadi University, Morbi Road, 360 003, Rajkot India
E-mail: sridaran.rajagopal@marwadieducation.edu.in
ORCID iD: https://orcid.org/0000-0001-7397-7611

Received: 13 August 2025; Revised: 16 October 2025; Accepted: 16 November 2025; Published: 08 February 2026

Abstract: In this rapidly progressing technological age, the likelihood of transmitting inadequately secured data in the cloud is high. Over the past several decades, experts in data and information security have tested and experimented with numerous hashing combinations. However, these have proven to be insufficient in preventing the interception and decoding of confidential text during transmission. Therefore, methodologically, in this current research, cryptography and steganography is diversified into using three different encryption algorithms of RSA, AES and LSB with further compressing of the to-be-communicated data, to enable the use of limited space in the cloud, as well as permit fast and quick embedded message transmission. The proposed architecture, EdSri has been implemented and tested against few existing models and found to show improved performance in terms of measured security metrics such as password strength, time between login attempts, login attempt rate, failed login attempt rate, device and browser fingerprinting, and also, measuring compression parameters like structural similar index, compression time, compression ratio, compression speed, bit per pixel, saving percentage, mean squared error, and peak to noise signal ratio, EdSri would hopefully become viable platform for exchanging secured information among the cloud communicators when hosted.

Index Terms: Huffman Coding, Compression, Cloud, Cryptography, Stego, Image.

1. Introduction

The search for a much more enhanced trustworthy secured means of communicating information from a particular source to another destination is an increasing concern for all users of sensitive data including organizations. Amidst plethora of such already existing means of exchanging information across networks such as emails, instant messaging apps (WhatsApp), file sharing services (Dropbox) is the pre-dominant challenge of the emergence of intercepting and deciphering technologies, which make it mostly easy because of non-implementation of sophisticated and complex techniques to sway the intruder. It is in regard of this perspective that the motivation for the conduct of this study is conceptualize, employing the use of RSA, AES, LSB steganography and Huffman compression encoding in a novel and optimized fashion to fuel a maximum, subtle, secured, private transfer of information between users, but at the same time being able to cut down on the quantity of data in storage. Therefore, the primary objective of this research is to optimize the security of data, improve its privacy, and to maximize the cloud storage space of data on servers.

Cloud computing has over a decade since its outset provided useful services such as platform as a service, software as a service and infrastructure as a service among others to the users of its systems. Contingent on pay per used grounds, the users are capable to access the data instantly on-demand services ubiquitously over internet connectivity irrespective of the user location. Cloud computing, associated with numerous advantages, is very significant in scalability of resources, flexibility of its services, resource pooling, and etcetera [1].

With the advent and continuous development of cloud structures, both processed and unprocessed data is consistently being transferred from one node to another. The communication channels as well as the kind of data under transfer may change: videos, texts, images, etc., and could be conveyed among different cloud networks The necessity to send information on the internet calls for a healthy well-protected scheme and secure data protection. Otherwise, data that is not well-secured can be seriously conceded when being sent [2]. The practice of transmitting information on the internet is increasingly becoming unsafe. This is largely due to the growing dependence on the internet by firms and people to transfer data or information from one place to other places [3]. While data stored on a computer's desktop may be considered relatively protected, the file or data is still susceptible to viruses and malware, which can result in data loss. Therefore, the importance of security for both the data and systems cannot be overstated in order to guard against hacking attempts [4]

1.1. Overview of Cloud Computing Issues

The prevalent cloud computing issues, as far as the multi-cloud system is concerned, include data security, data privacy, data storage, the efficiency of services performance, cloud and its data reliability, real-time data tracking, allocation of resources, data storage location, protocol supports and among others. These issues unfortunately remain unsolved by both public and private stakeholder sectors. However, this current research will focus on finding an improved solution to three (3) crucial cloud computing issues, namely, cloud data security, managing allocated storage space, and the privacy of communicated data among the many cloud computing concerns [5].

A. Cloud Data Security – AES and RSA Encryption

Data Security is the protection of digital data through its cycle of life to safeguard it from getting corrupt, damaged or into illicit access. Data security pivots around the integrity, confidentiality and availability of data. The effectiveness of data security requires that whether data is in storage or transit, it should be insulated enough from being accessed and compromised by an un-supposed user, hence, made available only to the legitimate user [6]. As widely as there may be several means of ensuring the security of data in the cloud, it starts first of all with data cryptography [7].

Cryptography involves the study and application of methods to protect private messages, information, or data from unauthorized third-party access by using secure communication techniques. This process encompasses aspects of information security disciplines such as integrity, authentication, availability (CIA), and non-repudiation, which are fundamental to modern cryptography [8]. The encryption of data in a cloud environment is a primary concern for researchers. This is due to the fact that data or information in the space of the cloud can be accessed by users who have authorized permission from any point that internet is available. Cryptography aids to secure data which is being transmitted. There are a number of existing algorithms that have been engaged to encrypt text ever since the concept to encrypt and decrypt files begun. Some of these are DES - Data Encryption Standard, AES – Advanced Encryption Standard, IBE – Identity Based Encryption, and RSA – Rivest Sharmir Adleman [9] The figure 1 below further buttresses the argument of encryption and decryption of a sender's message whose plaintext is encrypted before transmitted through a channel to the receiver. The cipher message upon reaching the recipient is decrypted into a plaintext again for the ingesting of the receiver.

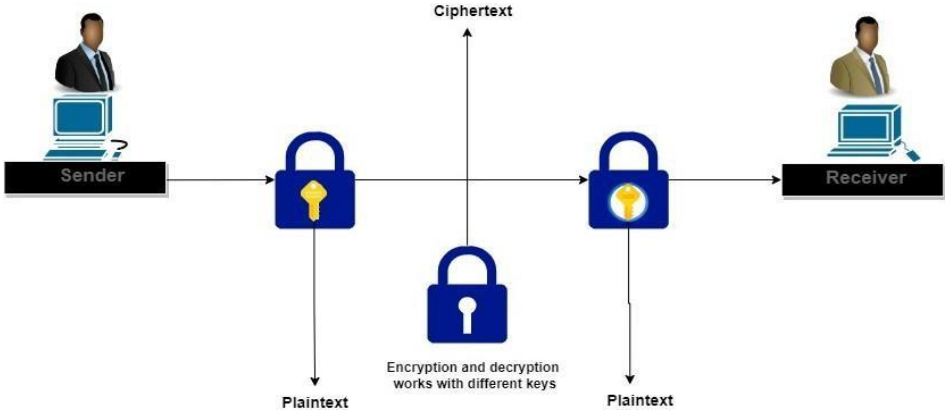


Fig.1. Data encryption and decryption

Data encryption primarily exists in two forms: symmetric and asymmetric. Symmetric encryption performs it's both encryption and decryption of files by employing the usage of one particular key. This method, which has a long history of use, is often chosen when large amount of data needs to be transferred. In symmetric encryption, both the one who sends and the one who receives share that one key to encrypt and to decrypt a file. However, the use of the same key for both processes is a drawback of symmetric encryption compared to asymmetric encryption. Examples of symmetric encryption include DES, Blowfish, MD5, and AES [10].

Asymmetric encryption, on the other hand, is generally regarded as a more secure form of encryption than symmetric encryption [11]. This type of encryption uses two different keys: one for encryption on the sender's side and another for

decryption on the recipient's side. As a result, asymmetric encryption can keep the intended message as the recipient's secret, until it reaches its end point for decryption. Unlike symmetric encryption, which uses only a private key and can lead to the message being tampered with if revealed, asymmetric encryption is more secure because its makes use of two different keys: public key (for encryption) and a private key (for decryption). Commonly used types of asymmetric encryption include RSA, Diffie-Hellman, Elliptic curve techniques, and PKCS algorithm [12].

The Advanced Encryption Standard (AES) operates as a symmetric block cipher capable of encryption. AES uses plaintext in blocks of 128 bits, with keys of 128, 192, and 256, converting it into cipher text. AES is able to insist on the aggregation of plaintexts that are converted into cipher text, as indicated by the key size involved in the AES cipher process. The totality of rounds involved in changing AES blocks are multiplicative processes of stages, dependent on the specific key used during encryption. After a number of reverse rounds, the cipher text is then converted back into plaintext. A drawback, by means of the AES encoding is that its numerical assembly is fundamentally basic, making it challenging in implementation [13].

The Rivest-Shamir-Adleman (RSA) is a widely used file encoding formula which is very often utilized in combination with one other encryption type, which frequently works alongside digitally added signatures to verify the legitimacy and validity of a message. RSA is not entirely reliable on its own and requires fewer resources to operationalize when juxtaposed with symmetric encoding methods like AES. It is infrequent to have RSA applied all alone for data encryption. In the case that security of application will be optimized, RSA is most of the time fused with some other encoding method(s). To decode RSA encoded message, there is always the requirement of private key, while public key is applied to encode the envisioned text [14].

B. Cloud Data Storage - Lossless Text Compression Using Huffman Algorithm

Data Storage is the state of saving both processed and unprocessed data in a digital environment on a server that is out of site location. The offsite servers are usually hosted and manned by providers who are known to be third parties. It is therefore, a mandatory requisite to access, retrieve and use the same stored data only by the medium of internet connectivity. This aspect of cloud technology allows companies to purchase and own their storage space, and maximize it to their advantage [15] . Saving data storage space in the cloud, is often mainly attained either through the medium of lossless or lossy compression or both.

Data compression, is as well referred to as data compaction, a technique that is applied for the purposes of cutting down to the minimum amount of size as possible the totality of the quantity of the data that will be stored and transferred from a source to a destination [16]. In general, the technique in compressing data can be realized through either the means of lossless or lossy compression. Lossless method of compression aims at retaining every necessary integrity features of the data. This way, when the same data is decompressed, it is able to return the compressed data to its initial state of originality. The technique involves in applying lossless data compression is such that the redundancy of text in data to be communicated is cut-off, but added back when undertaking decompressing process. This technique is useful in compression when the ultimate intent is to as much as possible avoid data compromising and loss. However, in the case of lossy compression, the technique scans through the entire text and finds pieces of data to remove completely. Although, this algorithm offers the benefit of providing a smarter and faster compression ratio in comparison to lossless compression model, it definitely as always result in some degree of loss of data from the original text [17]

The fundamental recommended framework coding procedure depicts how data/text can be processed from source (sender) to its destination (recipient). It defines by diagram the stages of compressing a file. Firstly, an RSA private key is generated and shared with the recipient via email. The plaintext to be communicated with its intended recipient is encrypted, by the use of AES algorithm. Then, by applying Huffman lossless data algorithm, the data is compressed. LSB is finally employed to embed the encrypted compressed data from cover image, which is now stego image, and stored on the cloud application platform. The stored message with the stego image is now accessible to the receiver on the cloud platform.

Huffman coding also referred to as Huffman encoding is explained to be a technique that is applied to compress data without losing the value of the message [18] It produces the length of characters as illustrated in table 1 below. It functions such that it is able to generate the lowermost likely significance of typical dimension of code of a specific text. For example, if three (3) different text (txt1, txt2 & txt3) are sent as being the most regular occurred texts, and have the possibility frequency order of 0.3, 0.25 and 0.15 in character measurement coding, text is coded using the number of characters of bit contingent on the extent of occurrence of the text in a specific data. The data is coded by the use of bit of k; where the k is in length, and number [l, n]. It is argued by researchers that using the Huffman coding technique produces a lot of upsides as against the downsides: It produces the highest accuracy and optimized code performance. Nonetheless, the point of decoding in tantrum with the original text is acute [19].

Numerous benefits are associated with the use of the Huffman algorithm when it is likened with other data mechanisms for compressing. Every bit of data element is included in the data compression implementation process from the order of the least to the highest. The Huffman algorithm model is efficient in restoring the compressed data back to its original state while producing an implementation tree. The Huffman algorithm cuts down data duplication to a very large extent even though file quality is retained [20]. For instance, the mechanism involved in producing the Huffman codes is illustrated in figure 2. The probable numbers are written in the order of the highest to the lowest. Then a pair of probabilities are added and written over the next probability that corresponds in value with the addition of the last two numbers. The rest of the probable numbers are then written below, as the cycle continues in that order. The underdrawn

structure is representative of the length of the Huffman code word.

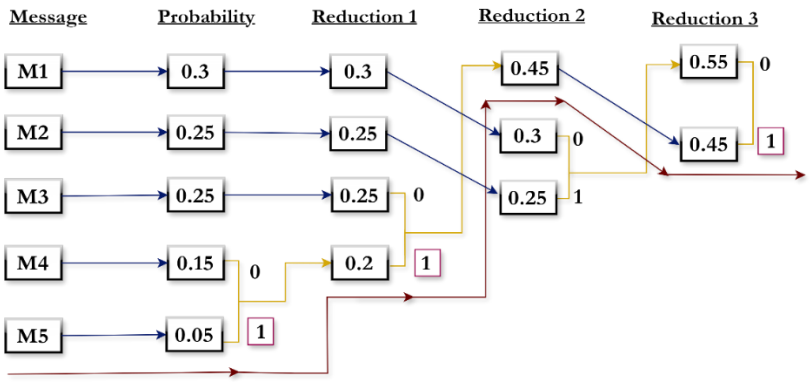


Fig.2. Huffman value reduction process

Table 1. Huffman variable length coding

Message	Probability	Code	Length
M1	0.3	00	2
M2	0.25	01	2
M3	0.25	10	2
M4	0.15	110	3
M5	0.05	111	3

And, following is the length of the password by averages:

$$Lavg=\sum(i=1)\text{▒}[[Pi\times li]] \quad (1)$$

$$=(n\times p)+(n\times p)+(n\times p)+(n\times p)+(n\times p) \quad (2)$$

$$=(2\times 0.3)+(2\times 0.25)+(2\times 0.25)+(3\times 0.15)+(3\times 0.05) \quad (3)$$

$$Lavg=0.6+0.5+0.5+0.45+0.5 \quad (4)$$

$$Lavg=2.2 messages \quad (5)$$

Where, Lavg is the length of the average code.

Definition 1: N denotes distinct symbol number in the source alphabets.
Definition 2: Pi is the probability of frequency of the i-th symbol.
Definition 3: li signifies length of the code-word for the i-th symbol.
Definition 4: ∑ is the sum of all contribution symbols in the formula.

Entropy measures the quantity of loss or gained data in the compression process. In the given example, it is evident that the mean figure of bits reduces to 2.2 bits when coding the variable by their length. The set of codes then achieve its peak performance if the probability distribution of the source is identified beforehand, and individual bits from the foundation figure bits is encrypted as an integer.

Hence, the entropy;

$$H(x)=-\sum(i=1)n\text{▒}[[p(xi)log2\text{▒}[[p(xi)]]]] \quad (6)$$

$$=-(3/5log2\text{▒}[[3/5+2/5log2\text{▒}[[2/5]]]]) \quad (7)$$

$$=-(3/5(-0.737)+2/5(-1.322)) \quad (8)$$

$$=-((-0.442)+(-0.529)) \quad (9)$$

$$=-(-(0.971))H(x) \quad (10)$$

$$=0.971 \quad (11)$$

Where, $H(x)$is the entropy of the random variable X.

Definition 1: $\sum$ is the summation of the overall probable outcomes of x.
Definition 2: $p(x_i)$is the probability of frequency value of x_i
Definition 3: log_2 denotes the logarithm to base 2

C. Cloud Data Privacy - Image Embedding Using Least Significant Bit

Data privacy involves safeguarding the identities of individuals whose data is being handled across various levels. It encompasses the collection, storage, utilization, disclosure, and disposal of data, ensuring it remains relevant and secure [21]. In cloud computing, indemnities and liabilities often revolve around breaches of privacy and security, leading to data loss, misuse, regulatory penalties, as well as service interruptions or failures to meet agreed service levels. However, the privacy of data in the cloud can be achieved through the adoption of several mechanism such as using robust authentication, implementing encryption, enforcing access control, steganography, and among others [22].

Steganography is an ancient discipline and an art form that involves concealing a secret message or data within another message. This technique is particularly useful for embedding information or data within images, videos, and so on. The act of hiding data within files is a crucial aspect of steganography, seeking to protect data from unauthorized users [23]. In contrast, with cryptography, a scrambled text sent to an intermediary can be easily decoded and potentially destroyed. Nevertheless, steganography hides undisclosed note inside façade files, which can largely prevent unwanted access during communication. This practice of hiding files inside an image is known as image steganography. In this method, a text or data is concealed within an image, which is then sent from the sender to the recipient through a communication channel. Consequently, even when the steganography image is intercepted, one is often unaware of the hidden data that is inserted. Upon receipt of the stego message, stego tools are used to decode the image for access to the hidden data or information. Such tools are available in various formats, including text, videos, audios, images, and others [24]

The underneath figure 3 demonstrates the concept of the least significant bit (LSB). The LSB file concealing method, involves eight bits, and the least significant number eight (8) is transformed into a data bit of eight as well. This refers to the rightmost bit in a binary number. The value of the LSB is 2^0, which is 1. Thus, LSB=BinaryNumbermod 2. For example, the result of this formula will be 0 or 1 representing the value of the least significant bit. Hence, the LSB value of the binary number 1101 is depicted as LSB =1101mod 2=1. So, the least significant bit is 1 [25]. The choice of Least Significant Bit (LSB) over other data embedding techniques includes its ability to incorporate huge quantity of data within an image, it is applicable to varied data formats, and it easily retains the quality of the original image [26,27].

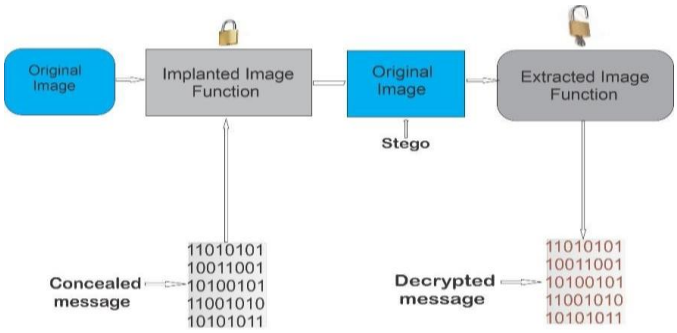


Fig.3. Steganography image and functional process

The approach of the LSB steganography algorithm is deemed to be quicker, computationally less complex, and easier to implement, with superior reliability value likened to certain other steganography algorithm methods. Given the advantages of LSB over other steganography techniques, this current research work opts to utilize its algorithmic methods to conceal text for secure and safe messaging to the intended receiver [28].

1.2. Organization of the Paper

Section 1 provides background information that puts the whole study into its context and relevance. It also helps to from broad perspective zero down the study to specific objective scope. The following sections of the EdSri Model research are organized in the order of section two, thus, the organization of related work which reviews the research work of other authors that have significant bearing on the EdSri architectural study. Section 2 also further provides the analysis of few related works taken from the literature. The proposed EdSri architecture and its constituents are thoroughly explain under section 3, and the analysis of the results which describes, interrogates, and adds meaning to the relevance of the findings in this research work has been covered under section 4. Lastly, the conclusion section 5, sums up the totality of the model; the limitations, the strength, and the future work prospect of the undertaken work.

2. Related Works

The volume and frequency of data transmission over the cloud by users have more than doubled in recent times. Given this increase, issues related to user authentication and the integrity of data that is under transport in the cloud space have garnered significant attention from security experts in research.

The authors, Abdelwahab et al., [29] introduce a model that combines compression and security, demonstrating its effectiveness in specific domains. Notably, the incorporation of Huffman coding is highlighted as influential, particularly in sensitive areas such as genome and DNA sequences. The proposed enhancement involves addressing the overhead associated with sharing keys and eliminating the tree. This could be achieved through the adoption of adaptive and dynamic coding techniques. However, their research work acknowledges its limitations, indicating room for improvement in the algorithm: The paper does not mention the measuring parameters of PSNR, and MSE as the current paper does. It also does not seek to improve on a particular algorithm like this current paper seeks to do to Huffman, even as it bolsters security.

The paper of Wahab et al,. [30] present an integrated approach combining RSA, Huffman Coding, and Least Significant Bit in one example, and RSA, Huffman Coding, DWT, and Least Significant Bit in another example to achieve secure image compression and steganography. It explores in separate mode both lossy and lossless compression methods and evaluates them using metrics such as PSNR, SSIM, MSE, and compression ratio etcetera. The use of RSA enhances security, while Huffman compresses text, and DWT contributes to reducing image size without compromising quality. LSB is then used to implant both examples offered in one work. The method reportedly reduces message bits by up to 25%, with strong stego-image quality, averaging about 38dB PSNR. The approach aims to balance data security, compression efficiency, and image quality. Overall, it proposes a compact and secure solution for image data transfer. However, this current EdSri proposed work focusing on their example of using RSA for security, Huffman coding for text compression and LSB for privacy and as well as security to some extent, improved on their overall result while also further bolstering security by combining RSA and AES. In addition even though the research of Wahab et al,. [30] does not mention explicitly any compared reference work, our work compares our results to theirs, and also implemented five (5) other additional security metrics which are absent in the reference research.

"A Hybrid Data Security Technique Using Chaos Encryption, RC4 Encryption, Huffman Data Compression and LSB Steganography" by Negi et al., [31] introduces a multi-layered method of securing data. With the use of chaos theory, RC4 encryption, and combining Huffman compression with LSB steganography, the authors seek to improve both confidentiality and imperceptibility of data transmission. The hybrid model is promising in terms of increasing complexity for the attackers and decreasing data size. Yet, two major weaknesses are found in their research: first, the application of RC4, a vulnerable and outdated cipher, undermines the overall cryptographic security of the system. Second, the use of Chaos Encryption, even though may be a promising algorithm, it is not yet widely tested to standardized acceptability. Also, the results of the metrics measured in their work is still lower in comparison to this current EdSri project.

The study of Lateef & Shakir, [32] presents a hybrid approach that combines RSA encryption and Huffman encoding with LSB image steganography to enhance data security and payload capacity. The secret data is first encrypted using RSA and then compressed with Huffman coding before being embedded into the least significant bits of a 24-bit RGB image. The system achieved strong performance metrics with high PSNR values indicating minimal perceptible difference between cover and stego images, and low MSE confirming reduced distortion. However, the work suffers from two main weaknesses. First, RSA encryption, while secure, it is not combined with another securing algorithm to foster higher security. In addition, the paper lacks a comprehensive steganalysis evaluation to test the robustness of the stego-images against detection techniques, since only two parameters, MSE and PSNR are the only tested metrics.

Rahman et al, [33] present a steganography technique that works on the achromatic (grayscale) part of an image using multi-level encryption and Huffman coding to ensure maximum data security and file size reduction. The compressed and encrypted data is embedded using the LSB technique without destroying the color channels to ensure imperceptibility. The proposed technique demonstrated high PSNR and SSIM values, showing that image quality and structure remains largely intact post-embedding, just as the current _EdSri work. Nevertheless, the study presents two shortcomings. Thus, by focusing solely on the achromatic channel, it limits the overall payload capacity compared to full-color embedding. Also, the authors implemented three out of the current work's eight image metrics, SSIM, MSE, and PNSR, yet two of the parameters, thus SSIM, and MSE record similar performances in comparison to EdSri, this current work,

The work proposed by Elgaier et al., [34] is concerned with hiding secret information in edge areas of an image to improve imperceptibility, employing Huffman coding for compression and a hybrid edge detection technique to identify embedding regions. The compressed secret information is embedded in the LSBs of the detected edge regions with the goal of achieving capacity and visual distortion balance. The method shows good PSNR and capacity metrics, implying a successful trade-off between data hiding and image quality. Despite its strengths, the study has two significant limitations. First, the accuracy of the hybrid edge detection directly influences the embedding quality; unreliable edge maps could increase detectability. Second, the absence of a variety of test images, limits generalization of results, an issue concerning performance consistency across different image types. The research also has limited parameter measurement of just SSIM, MSE and PNSR compared to eight in the case of EdSri.

The work of Awadh et al., [35] introduces a hybrid model of information security that integrates Discrete Wavelet Transform (DWT) for image compression, AES for encryption, and LSB for steganography. The secret image is compressed with DWT, followed by encryption with AES, and embedded in a cover image by the application of LSB approach. The proposed model has PSNR of 51.967.8 dB and SSIM of 0.93148, which provides superior visual quality and structural maintenance of the stego-image. But, several issues come to light in implementation. One, though DWT works, it does not compress text data as effectively as Huffman coding, potentially minimizing payload optimization. Two, the system measures two metrics – SSIM and PNSR only in their whole research work creating vacuum for a lot more of such metrics for consideration. EdSri, the current work obtains 0.999 approximately 1 as the highest SSIM, and 51.142 for PNSR,

The research paper authored by Agrawal, [36] presents a new Compressed Sensing Image Steganography (CSIS) scheme towards improving the performance of image steganography. In this approach, the cover image is sparsified in a block wise manner to generate linear measurements. Later, certain permissible measurements are selected to embed the secret encrypted data, in which 2 bits of the encrypted data are inserted into each permissible measurement. The reconstruction of the stego-image uses optimization techniques in the form of ADMM and LASSO. The presented experimental results demonstrate the performance of the scheme that shows an impressive embedding capacity, which is a 1.53-fold increase. Apart from that, it achieves mean PSNR of 37.92 dB and SSIM and NCC coefficients close to 1, indicating satisfactory image quality. But the use of ADMM and LASSO for reconstruction in the paper introduces computational complexity, which may pose challenges for real-time or resource-constrained applications. Also, all the parameters of EdSri, the current paper still outperform the metrics measured in the paper

In the research paper of Amenu et al., [37], the algorithms of AES, RSA and LSB were engaged to encrypt secret information , and as well as to embed the information within cover image. The authors used RSA to encrypt the user registration and authentication process, hence bolstering the security of the user at the login gateway of the system. The authors also engaged AES to encrypt the transmitted data. This highly safeguard the data from plaintext state, thereby enhancing information security, Finally, LSB is applied to hide the intended encrypted secret message within a deceptive original QR code cover image before communicating it over the cloud. Even though, this work employs enough security mechanism of AES, RSA and LSB, it does not offer any parameters/metrics to measure the performance of the data and or the image in the cloud hence, the need for the extended version which is the current paper.

The authors Raiyan, [38] propose a hybrid approach that significantly enhances image steganography by merging LSB embedding with Fernet symmetric encryption and Reed-Solomon error correction. The authors achieve a data payload of 3 bits per pixel (BPP) with a Peak Signal-to-Noise Ratio (PSNR) of up to 48 dB, implying zero distortion in images. Robustness is also confirmed by Bit Error Rate (BER) analysis, showing nearly zero data loss with simulated transmission errors. Structural similarity index (SSIM) measures also touch 0.996, confirming high visual similarity. Two glaring shortfalls are the lack of RSA or AES encryption support integration, restricting its versatility in public-key environments; constrained performance testing with real-world image manipulation quality using data processing capacity. In addition the current research, EdSri performance is comparatively better in most of the measured parameters' results.

Table 2. Tabulated related works

Author(s)	Algorithm(s)	Parameters
Abdelwahab et al., [29] (2021)	Huffman Encoding (compression focus)	Compression metrics only, no PSNR, MSE. etc.
Wahab et al,. [30] (2021)	RSA, LSB, Huffman Coding	PSNR, SSIM, MSE, CR, CS, CT, SP, BPP: all lower than EdSri, Current Work
EdSri (proposed work, 2025)	RSA, LSB, Huffman Coding, AES	Overall Improvement on [30] eight (8) image and data performance metrics , plus five (5) additional EdSri security metrics: password strength, Time between login attempts, login attempt rate, Failed login attempt rate, device and browser fingerprinting
Negi et al., [31] (2024)	RC4, Chaos Encryption, Huffman, LSB	PSNR, SSIM: lower, weaknesses: RC4: vulnerability, Chaos: unstandardized
Lateef & Shakir, [32] (2023)	RSA, Huffman, LSB	PSNR: higher, MSE: lower, limited evaluation: Only PSNR & MSE
Rahman et al, [33] (2023)	Multilevel Encryption, Huffman, LSB (grayscale)	PSNR, SSIM: higher, MSE, similar, limited payload & Metric capacity
Elgaier et al., [34] (2024)	Huffman Coding, LSB, Hybrid Edge Detection	PSNR: higher, SSIM, MS: Balanced, but limited metrics & images
Awadh et al., [35] (2022)	AES, LSB, DWT	PSNR: higer, SSIM: lower, limited to only the two metrics
Agrawal, [36] (2021)	CSIS, ADMM, LASSO	PSNR: lower, SSIM: higher, dissadvantage in higher computational complexity
Amenu et al., [37] (2024)	AES, RSA, LSB	Focus: security (user authentication, cloud transfer), no metrics
Raiyan, [38] (2025)	Fernet Encryption, LSB Reed-Solomon Error Correction	Payload 3BPP, PSNR, SSIM: lower, Limited metrics
Ghrare et al., [39] (2024)	AES, LSB	PSNR, MSE: lower; Payload: 1.5 BPP; lacks CR, CS, SP, etc.

The research conducted by Ghrare et al., [39] works on securing data in double layer mode. The text for communication in the research is encrypted with AES algorithm, and then embedded in a cover image. This hybrid model significantly increases imperceptibility and confidentiality with a Peak Signal-to-Noise Ratio (PSNR) of up to 54.3dB, a very high visual quality, and a Mean Square Error (MSE) as low as 0.13, which demonstrates minimal distortion. The system further supports high payload capacity of 1.5 bits per pixel, exhibiting effective embedding of data. Nonetheless, the project has some main limitations: Thus, the system does not have any compression (e.g., Huffman coding) incorporated, which might further increase capacity and performance. Also, since the research measured limited metrics, it cannot guarantee the performance of other parameters like Saving Percentage, SSIM among others if they were incorporated, which are all measured in EdSri.

3. Proposed EdSri Architecture

Crypto-Steganography is a method that merges the principles of cryptography and steganography to obscure and encode data within an image, making it invisible to the naked eye [40,41]. This presents a challenge to attackers, who must consider the potential existence of concealed data. In the event that the used keys are however obtained by the wrong person, a variety of encryption and decryption techniques and tools can be employed to retrieve and decipher the embedded and encrypted data [42]. The combined use of cryptography and steganography significantly enhances data security. In our research, we introduce EdSri, a system that provides strong security features, making it extremely difficult for the human eye to differentiate that of the original image from the stego. The authors utilize both AES, and RSA - public and private keys in encrypting and decrypting the data for onwards transmission. The encrypted message is however, first of all compressed using the Huffman encoding algorithm, and ultimately concealed within the image noise, as depicted in figure 4. This current EdSri proposed work security scheme is an upgraded version of the related comparative work of Wahab et al,. [30] model, which uses only RSA for text security purposes. The current EdSri work is therefore compared with the outcome of Wahab et al,. [30] model (the best performing results among the literature reviewed from 2021 to 2025) which, uses RSA, Huffman Coding and Least Significant Bit (LSB).

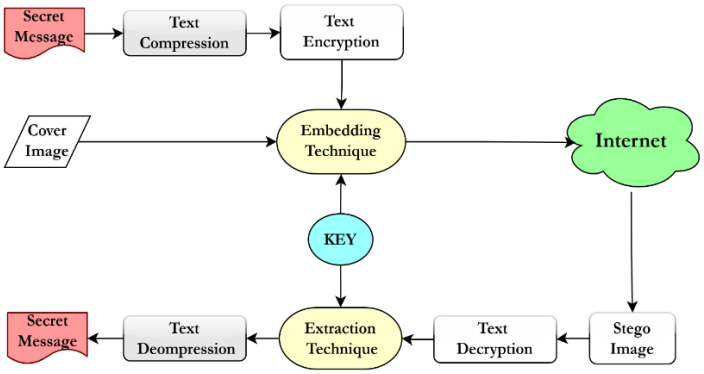


Fig.4. EdSri architecture

3.1. Algorithms Employed

The EdSri architecture employs the usage of mainly four (4) different algorithms in total. These include Rivest Sharmir Adleman (RSA), Advanced Encryption Standard (AES), Huffman encoding and Least Significant Bit (LSB). RSA is engaged in the development of the application to encrypt the user account registration and login section. This provides a resilient security entryway rendering its login portal possibly impossible for system attackers to bruteforce through it. The communicated information from sender to recipient is encoded using AES 128 key to foster a faster data transmission, and to deliver more proficient security, making it extremely difficult to crack. Huffman coding algorithm helps to process text into smaller size which affords the cloud data storage space to be managed more economically and efficiently. Finally, the EdSri architecture applies Least Significant Bit algorithm to privately and confidentially successfully implant the compressed and ciphered exchanged data into an image.

A. RSA Algorithm

RSA is used to encrypt the user's password, such that even if the user's password is detected, the account cannot be logged in, without the RSA private key. RSA is used to encrypt the shared AES key for messaging between two users. So, neither the sender nor receiver knows the AES key used for encrypting and decrypting the shared messages. Instead, the generated AES key is encrypted with the sender's and receiver's public RSA key. The message sender is responsible for providing their private key during message sharing to decrypt the encrypted AES.

B. Measured Security Metrics

In order to verify the security performance of the RSA login from the application programming interface, and also as a way of improving on the security of the base paper, Wahab et al., [30], the following five (5) monitoring security metrics are introduced into the EdSri model.

a. Password Strength

This security metric determines the intensity of resistance to a guessed and attempted cracking password by a malicious user. It is contingent on several variables including password length, complexities, exclusivity, and irregularity. The harder the password the more difficult it is to brute-force, or harder it is to use dictionary attack to crack, thereby, diminishing the tendency of unauthorized access. This measure allows for the assessment of user password strength and the enforcement of policies for constructing strong credentials [43].

b. Time Between Login Attempts

This measure counts the interval between successive login attempts by an individual or system. Exceptionally quick successful login effort may symbolize system automated bruteforce, and or stuffing of credentials, whist a longer successfully login attempt may represent a frantic human login effort. By monitoring this measure, security controls like rate limiting, account lockout, or CAPTCHA challenges can be enforced to prevent attackers [44].

c. Login Attempt Rate

The rate of failed login attempts is the rate at which a user or system makes login attempts within a particular timeframe. An abnormally high rate usually indicates malicious behavior, i.e., bots attempting multiple credentials in quick succession. If this parameter is monitored, systems easily recognize login irregularities and trigger alerts, or as well, enforce controls to safeguard against excessive attacks [45].

d. Failed Login Attempt Rate

This parameter shows the expanse of unsuccessful login attempts. The higher the failed login attempts, the higher the rate of a forgotten password possibility. This therefore, means that there is an efficient password verification or authentication requirements, and that password cracking or guessing attempts are highly unsuccessful [46]

e. Device and Browser Fingerprinting

Device and browser fingerprinting is the collection of specific attributes from a user's device, operating system, and browser to form a unique identifier. The measurement helps identify abnormal or suspicious login patterns, such as the same account accessed from other or unknown devices. It secures authentication systems by enhancing password-based security and assists in identifying account takeover attempts or fraudulent access [47]

C. Method and Modules

Python's rsa framework is utilized for:

- RSA key generation using its newkeys function.
- RSA key loading using its load_pkcs1 function.
- RSA encryption using its encrypted function.
- RSA decryption using its decrypt function.

```
import rsa, base64, os
from cryptography.exceptions import UnsupportedAlgorithm
```

Fig.5. Evidence of the RSA modules employed in the programming

3.2. AES Algorithm

AES is used to encrypt the to-be-communicated message before applying it to steganographic function.

A. Method and Modules

Python's cryptography framework is used for:

- AES key generation, using the derive method of the PBKDF2HMAC class generated from a SHA512 hashes
- AES encryption using the padder method of padding and the encryptor of the Cipher class.
- AES decryption using the decryptor of the Cipher class and unpadder of padding.

```
from cryptography.hazmat.primitives import hashes
from cryptography.hazmat.primitives.kdf.pbkdf2 import PBKDF2HMAC
from cryptography.hazmat.primitives import padding
from cryptography.hazmat.primitives.ciphers import Cipher, algorithms, modes
```

Fig.6. Evidence of the AES modules employed in the programming

3.3. Huffman Encoding Algorithm

Huffman encoding is employed to compress the text to be shared between users into bits before image steganography was performed after encryption. Huffman decoding converts the extracted compressed message from the image into an encrypted format before decrypting it with AES key.

Method and Modules

Python's heapq framework is used for:

- Python's heapq package is utilized to create a priority queue for managing the nodes of the Huffman tree.
- The pickle package is used to save the frequency dictionary of characters to a file later used for decoding

```
import heapq
from heapq import heappop, heappush
import pickle
```

Fig.7. Evidence of the Huffman modules employed in the programming

3.4. LSB Algorithm

LSB is used to hide messages in images after the message has been compressed with Huffman encoding, and encrypted with AES. LSB is applied for message retrieval during decryption before the retrieved message is decompressed and decrypted for viewing by the user.

A. Method and Module

Python's lsb framework from the stegano package is engaged for:

- Hiding messages in images using the lsb module hide method.
- Retrieving the hidden message in images using the lsb module reveals the method.

```
from stegano import lsb
```

Fig.8. Evidence of the LSB module employed in the programming

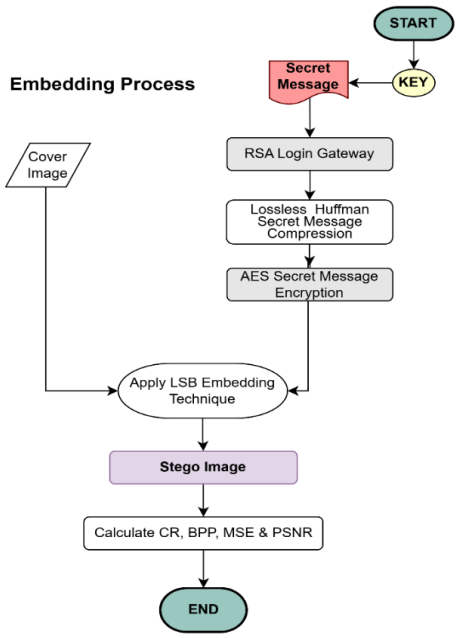


Fig.9. The process of embedding text

B. Embedding Algorithm

The embedding process starts by a user who logs into the system via RSA encoded entryway with sharing of keys. The user then compresses a secret message using Huffman encoding. The compressed message is then encrypted using AES, the encrypted secret text is then embedded using LSB function resulting into a stego image with the application of the compression metrics

Pseudocode 1: Text Embedding Process

```
compressed_data, huffman_codes = compress_text_huffman(text)
encrypted_data = encrypt_aes_128(compressed_data, aes_key)
stego_image = embed_in_image(image_path, encrypted_data)
save_metadata(huffman_codes)
return stego_image
def compress_text_huffman(text):
freq_dict = calculate_character_frequencies(text)
huffman_tree = build_huffman_tree(freq_dict)
huffman_codes = generate_codes(huffman_tree)
compressed_data = encode_text(text, huffman_codes)
return compressed_data, huffman_codes
def encrypt_aes_128(compressed_data, key):
cipher = initialize_aes_cipher(key)
padded_data = add_padding(compressed_data)
encrypted_data = cipher.encrypt(padded_data)
return encrypted_data
def embed_in_image(image_path, encrypted_data):
image = load_image(image_path)
binary_data = convert_to_binary(encrypted_data)
for y in range(image.height):
    for x in range(image.width):
        a. pixel = get_pixel(image, x, y)
        b. for color_channel in range(3):  # RGB
            i. if more_data_to_embed():
            ii. new_value = modify_lsb(pixel[color_channel],
next_bit())
:           a. set_pixel_channel(image, x, y, color_channel,
new_value)
:       a. return modified_image
def main_process(text, image_path, aes_key):
```

C. Extracting Algorithm

With the extracting process, the user logs in through the RSA encoded gateway. The text embedded image is then selected. LSB extract is then performed on the image for output of image binary to reveal the encoded message. The message is then decoded into Huffman compressed text. The compressed secret text is then finally decompressed and decrypted into plaintext

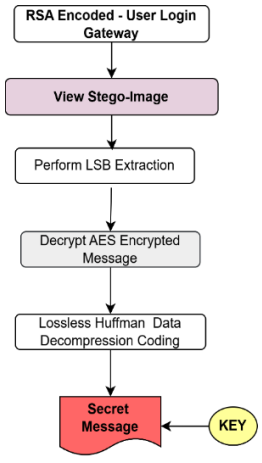


Fig.10. The process of extracting text

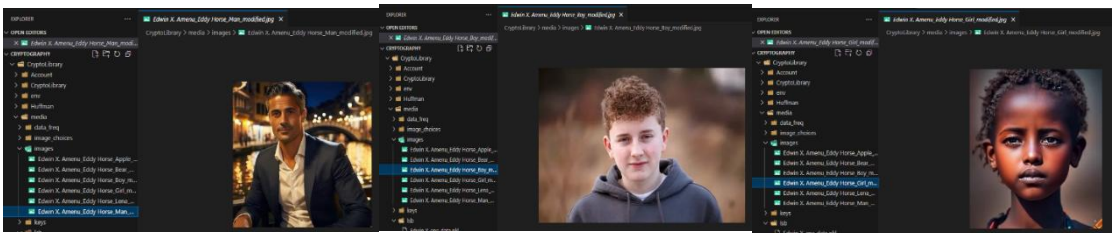
Pseudocode 2: Text Extraction Process

```
encrypted_data = extract_from_image(stego_image)
compressed_data = decrypt_aes_128(encrypted_data, aes_key)
Load saved Huffman codes
huffman_codes = load_metadata()
original_text = decompress_huffman(compressed_data, 6:huffman_codes)
return original_text
def extract_from_image(stego_image):
  binary_data = ""
for y in range(stego_image.height):
      for x in range(stego_image.width):
        a. pixel = get_pixel(stego_image, x, y)
        b. for color_channel in range(3):
          i.  if not end_of_data():
          ii.  bit = extract_lsb(pixel[color_channel])
          iii. binary_data += str(bit)
extracted_data = convert_to_bytes(binary_data)
 return extracted_data
def decrypt_aes_128(encrypted_data, key):
decipher = initialize_aes_decipher(key)
decrypted_data = decipher.decrypt(encrypted_data)
unpadded_data = remove_padding(decrypted_data)
return unpadded_data
def decompress_huffman(compressed_data, huffman_codes):
 reverse_codes = reverse_huffman_codes(huffman_codes)
decompressed_text = ""
 current_code = ""
  for bit in compressed_data:
      a. current_code += bit
      b. if current_code in reverse_codes:
        i.  character = reverse_codes[current_code]
        ii.  decompressed_text += character
        iii. current_code = ""
  return decompressed_text
def main_extraction_process(stego_image, aes_key):
```

D. Image Comparison

The pictures showing below in figure 11, are the inputted images into the application system before applying them to the embedding process.

Fig.11. Inputted images/original images



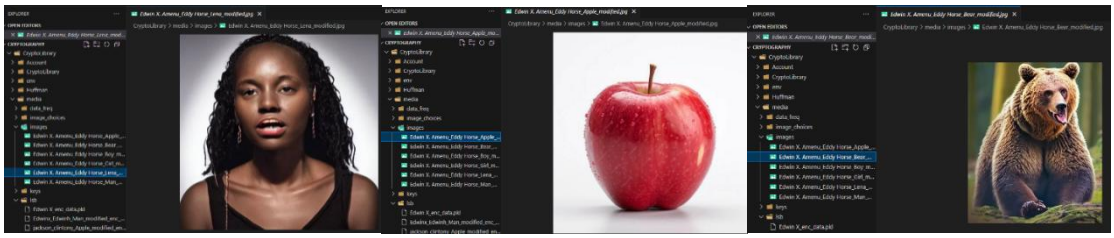


Fig.12. Outputted images /stego images

Figure 12 above demonstrates the high quality exhibition of the back-end stego images with the evidence of the app modules captured along with the outputted images. Thus, how the various images look after they are used to conceal the encrypted and compressed text. It is undoubtedly evident that if natural eye is used to view both inputted and outputted images in comparison, one can strongly affirms that there is no detection of distorted/noisy characteristics in the stego image(s) when juxtaposed vis-à-vis the original cover image.

Step 1: Users' Registration and Login Interfaces

In this research work, the authors develop a cloud-based application that allows only the duly registered valid users of the application to interact and exchange messages with a shared key among themselves. This access is granted only to those who are authorized.

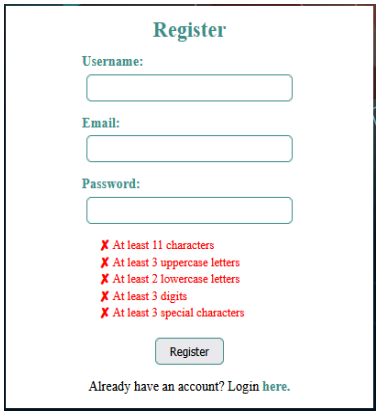


Fig.13. Registration interface

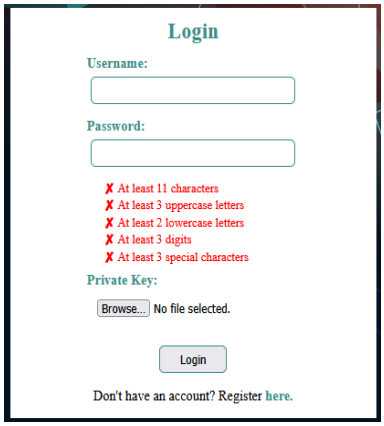


Fig.14. Login interface

Step 2: Users' Key Sharing Interfaces

At this point of the application, user 'A' shares a public key with user 'B' who are both already registered with/in system.

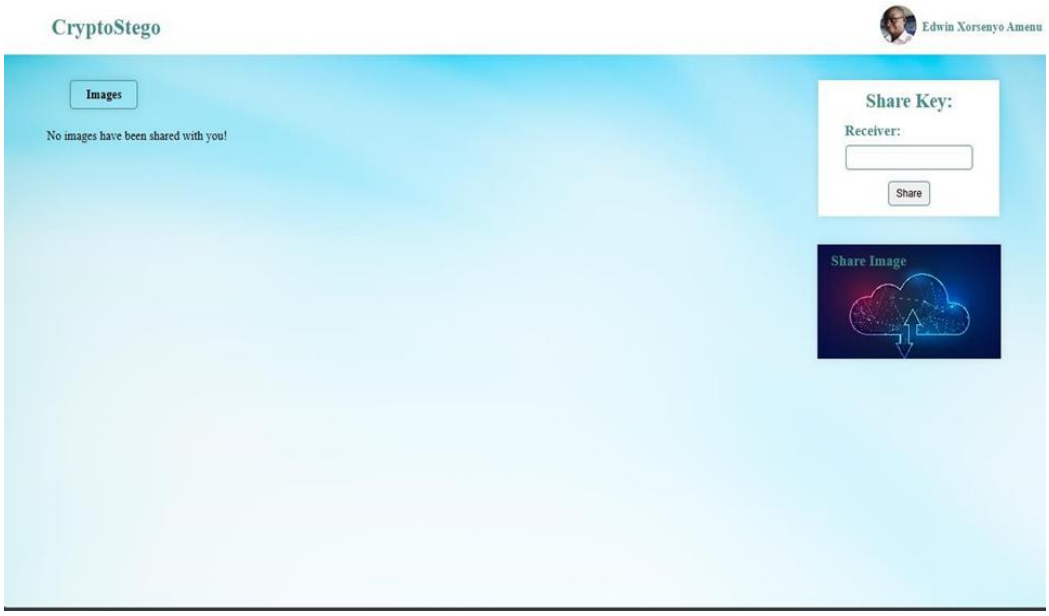


Fig.15. Key sharing interface

Step 3: Users' Messaging and Image Selection Interfaces

Registered users are able to share data/information between each other's account, by typing the to-be-shared message in the box, and select a picture of their choice and then click share

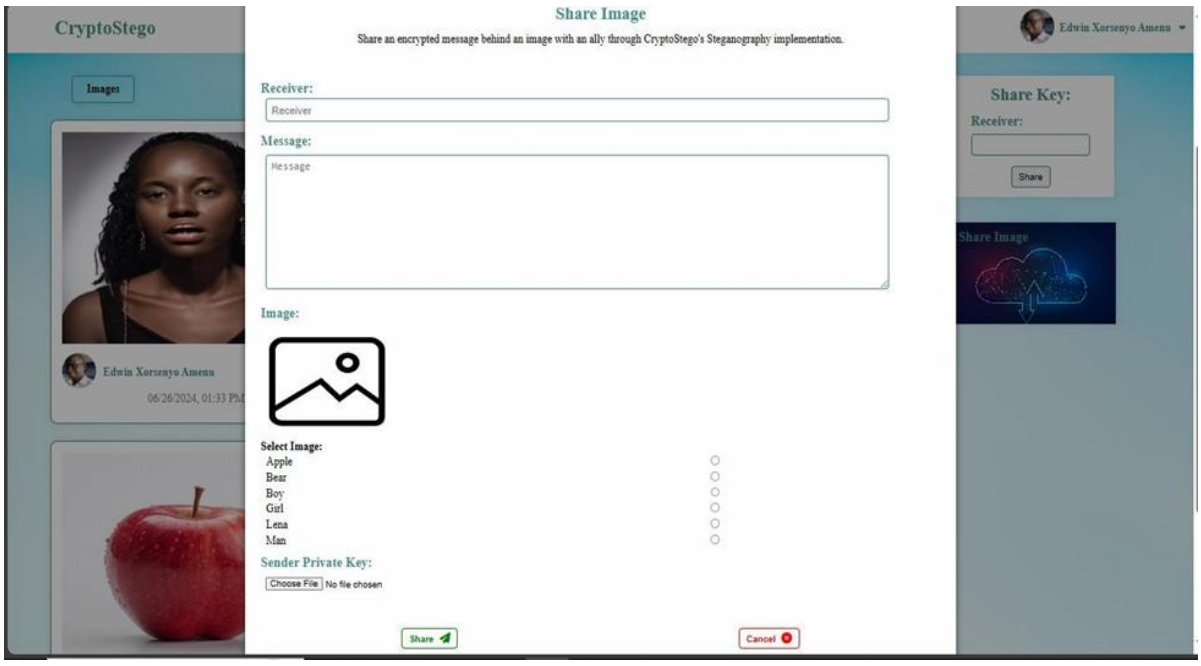


Fig.16. Messaging and image selection interface

Step 4: Users' Message Embedded Interface

The interface below, indicates that a particular user successfully embeds into a particular chosen image a compressed and encrypted text by applying an LSB technique.

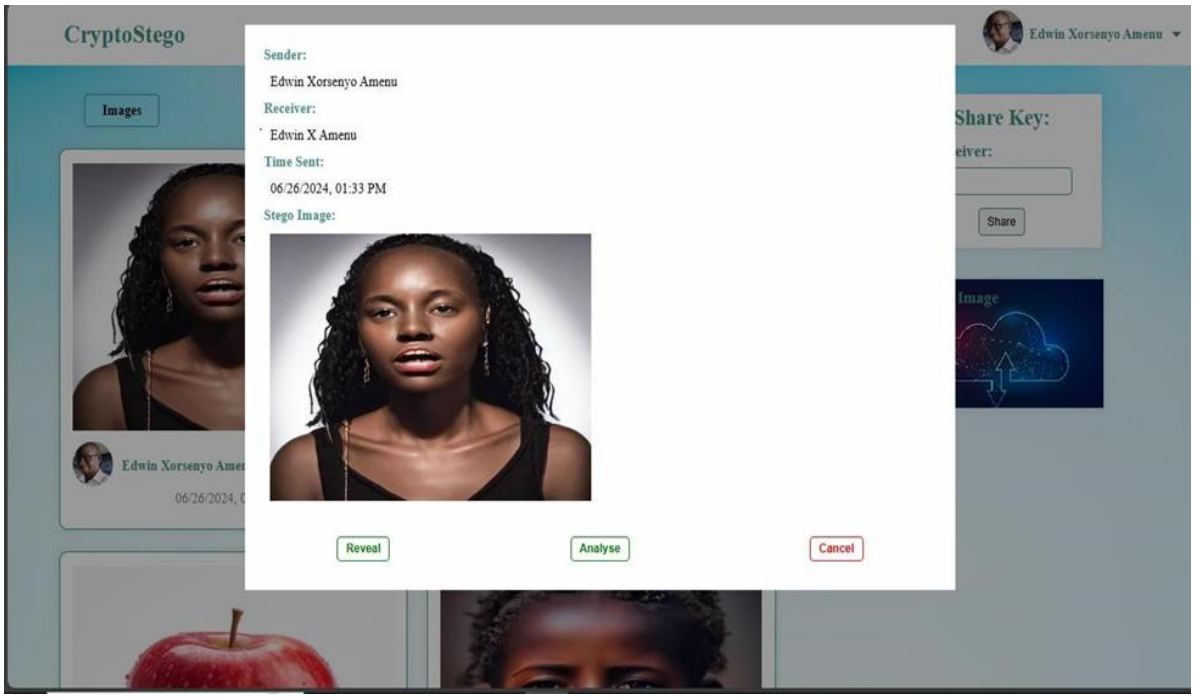


Fig.17. Compressed, encrypted text, and LSB embedded interface

Step 5: Users' Message Compressed Interface

The figure 18 below, reveals the Huffman compressed text

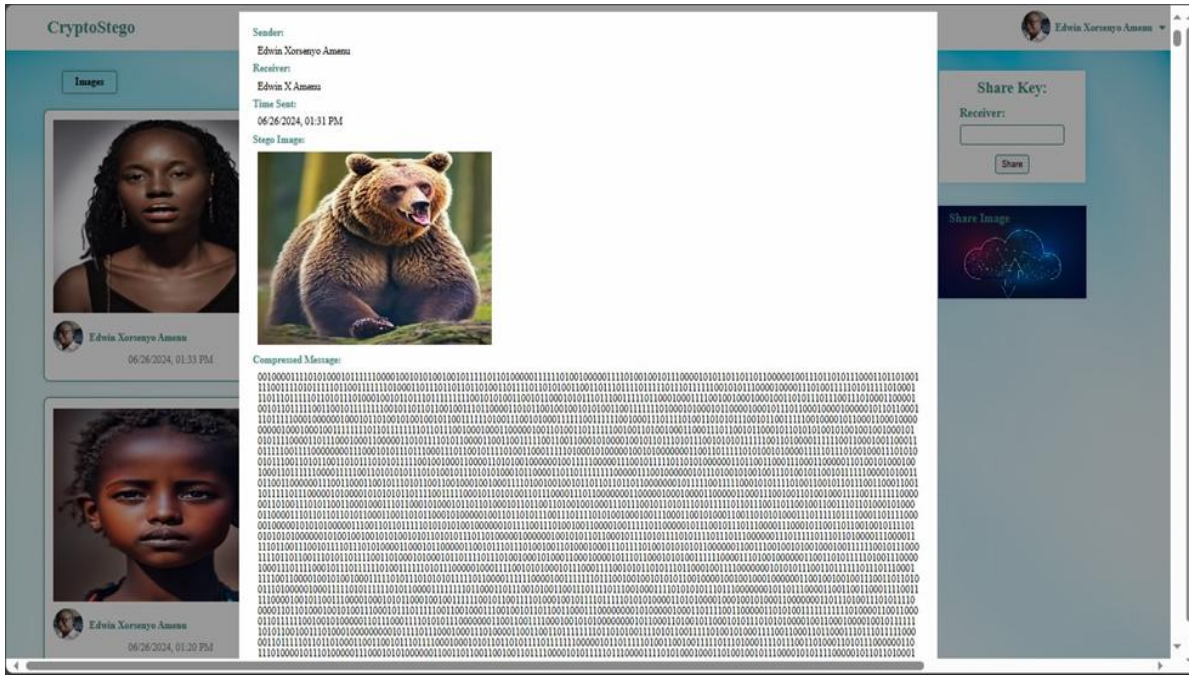


Fig.18. Huffman compressed text interface

Step 6: Users' Message Encrypted Interface

The figure 19 below, displays the AES encrypted text.

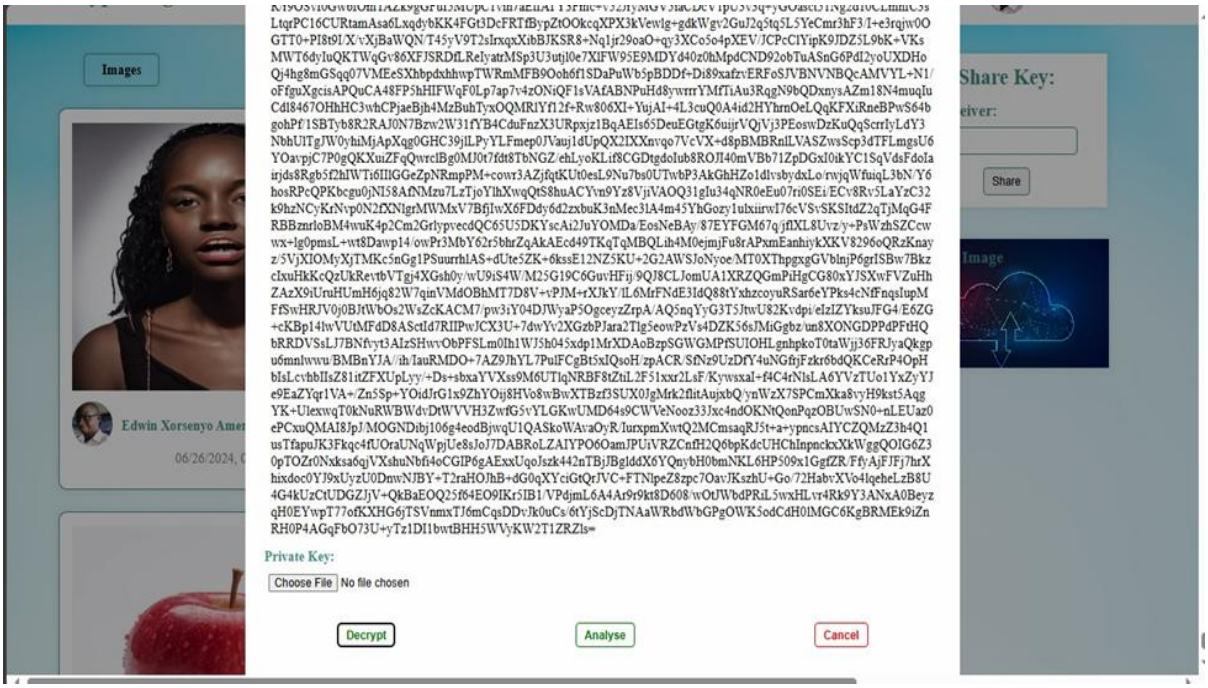


Fig.19. Encrypted text interface

Step 7: Users' Message Decrypted Interface

This interface underneath displays the decrypted and decompressed message when a user clicks on the "reveal" button.

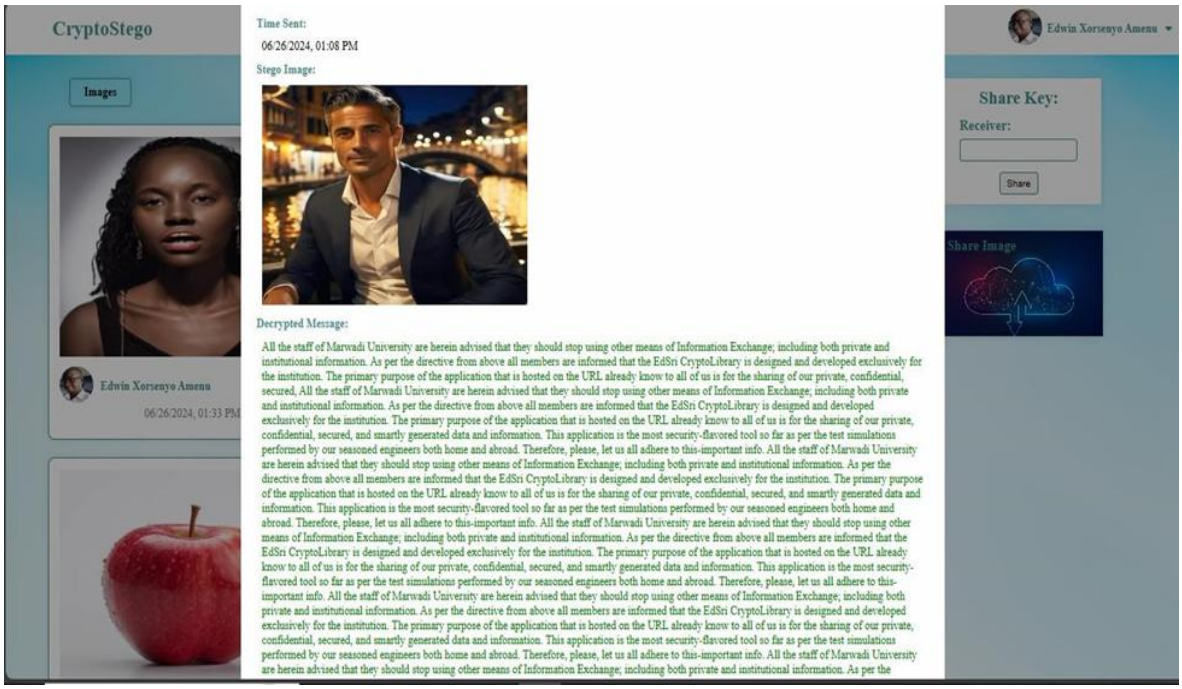


Fig.20. Decrypted/decoded text interface

Step 8. Users' Result Analysis Interfaces

This figures 21 to 26 below, show the proof outcome of the implementations, alongside the corresponding graphs.

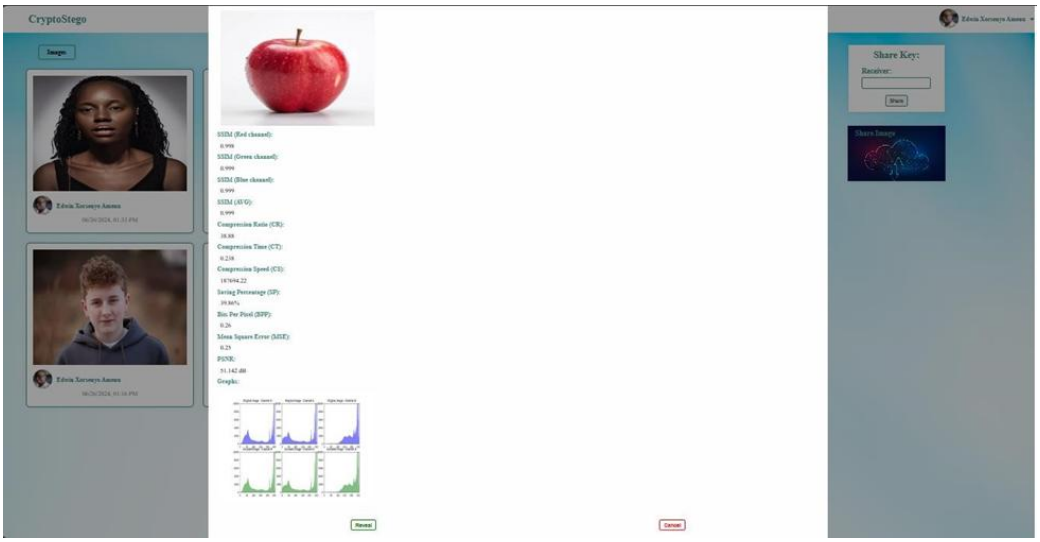


Fig.21. Proof of implementation results of apple

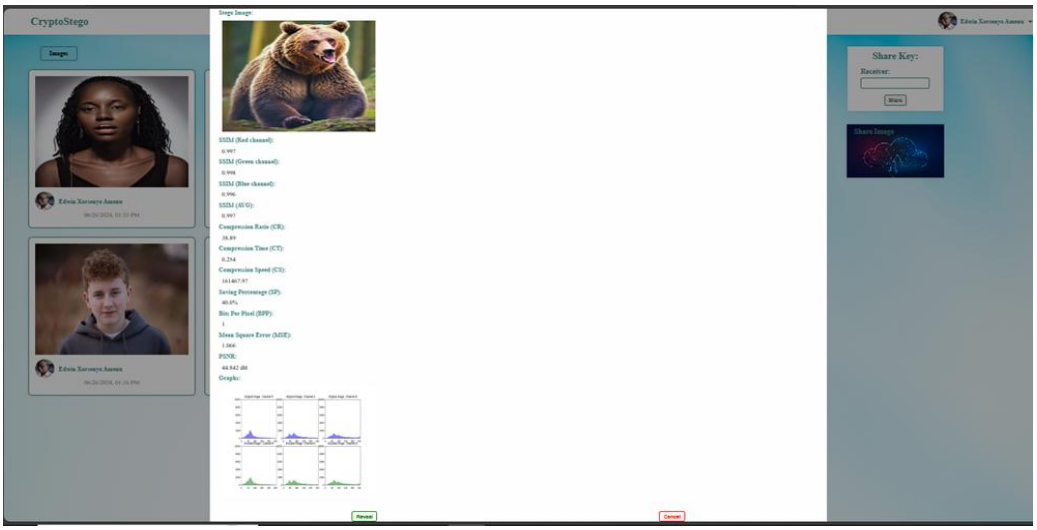


Fig.22. Proof of implementation results of bear

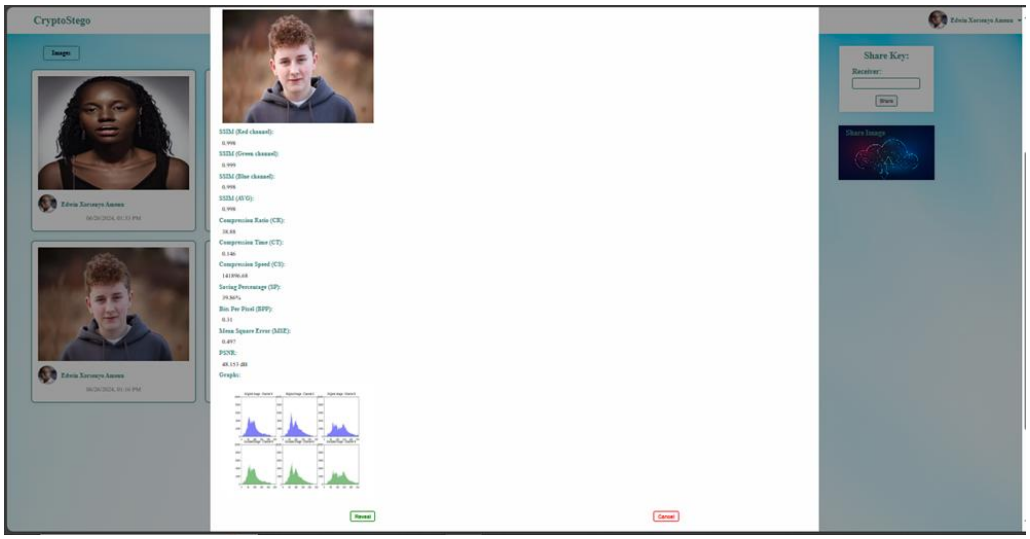


Fig.23. Proof of implementation results of boy

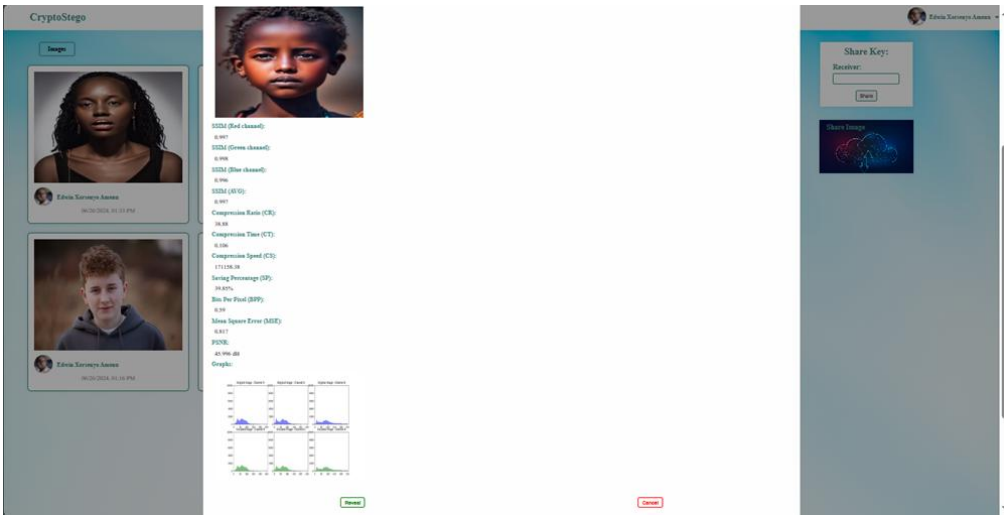


Fig.24. Proof of implementation results of girl

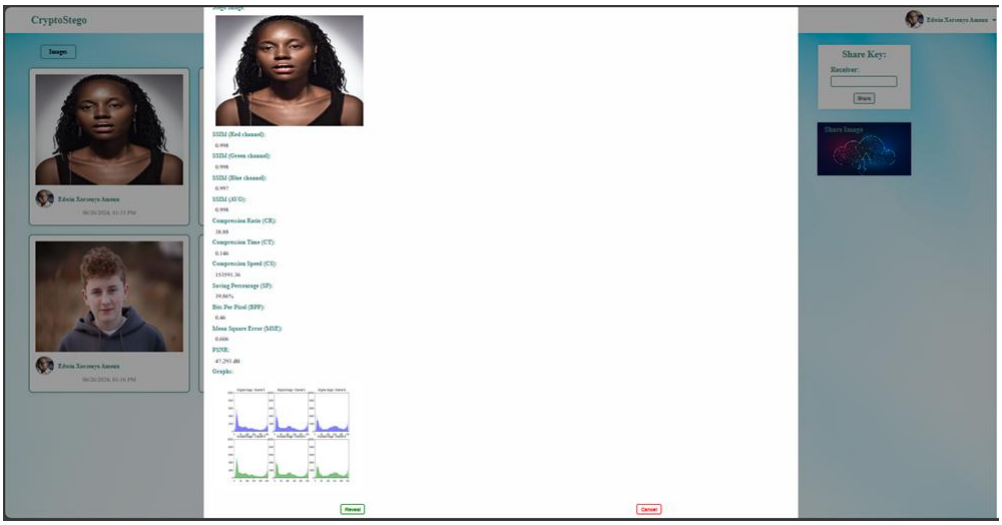


Fig.25. Proof of implementation results of Lena (woman)

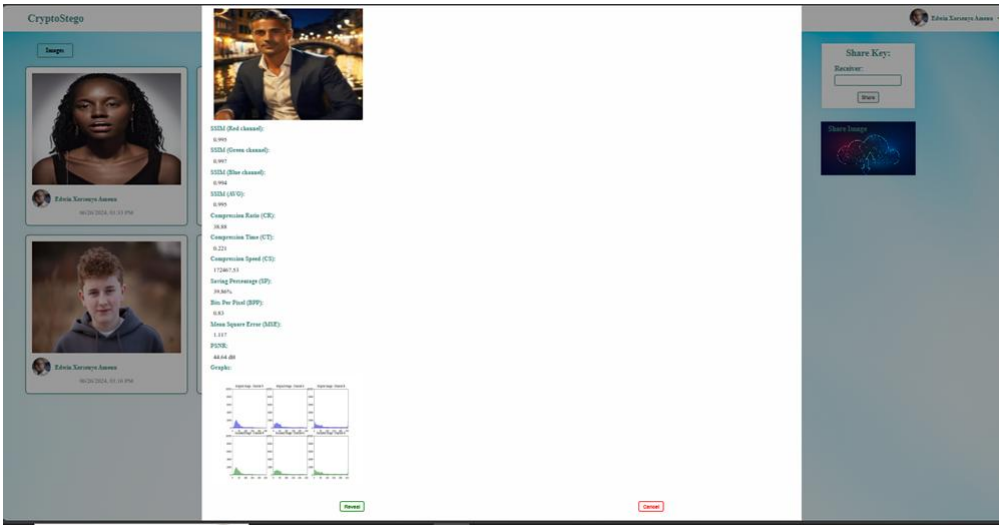


Fig.26. Proof of implementation results of man

Step 9: Users' Stego Images Interface

The interface in the figure 27, captures in display, all the six LSB steganography images employed in the implementation stages.

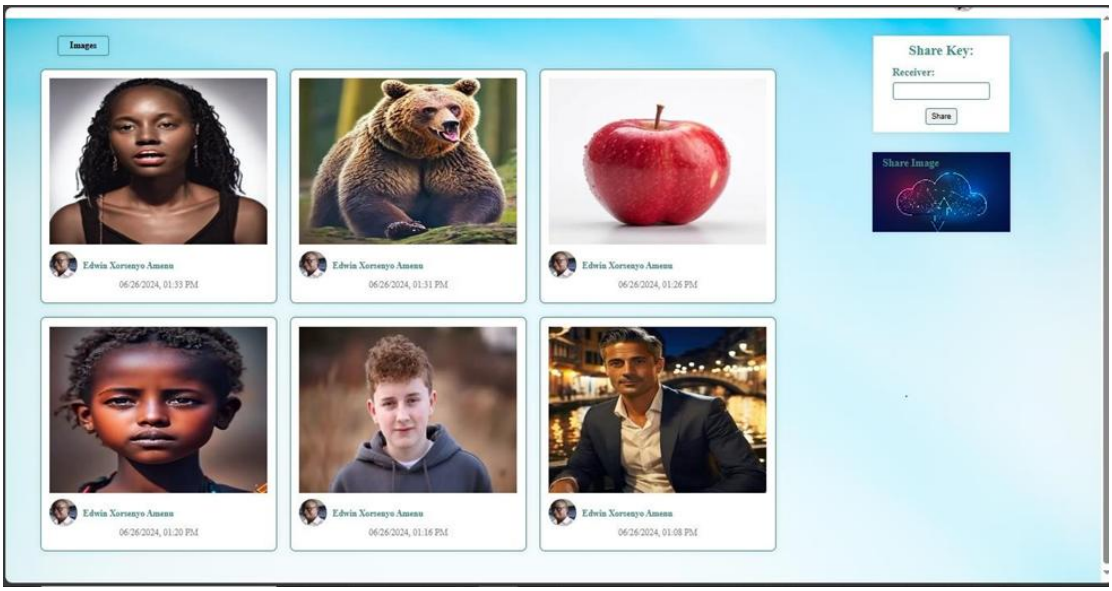


Fig.27. Evidence of all six steganography images interface

4. Result Analysis and Discussion

The first step for users to be able to use the application requires that they both register in the system. A user is then required to share a public key with the other registered user. At this stage the sender populates the information intended for communication, and selects a photo to embed the information. The sender now clicks on the "share button" This then triggers the application to compress, encrypt the file, and embed the information before finally duplicating the same information into the registered account of the second user, thus, the receiver. Once, the recipient of the information logs into the system with their accurate credentials, they are required to use their private key to access the communicated information. The receiver, at this point clicks on "decrypt button" to reveal both the encrypted and compressed data as a plaintext. And, a click on the "analyze button" displays the results that are generated alongside the information transferred.

Table 3. Value obtained: EdSri proposed-model

Number	Cover image	Message size	Huffman Coding (Lossless) Measuring Parameters							
			SSIM (unit=s}	CR (unit= %)	CT (unit=s}	CS (unit=s)	SP (unit= %)	BPP (unit=mm)	MSE (unit=#)	PSNR (unit=dB)
1	**Man**	**38278**	0.995	38.88	0.221	172467.5	39.86	0.83	1.117	44.64
2	**Boy**	**25000**	0.998	38.88	0.221	141896.68	39.86	0.31	1.497	48.153
3	**Girl**	**30178**	0.997	38.88	0.106	171158.38	39.85	0.59	0.817	45.996
4	**Apple**	**33145**	0.999	38.88	0.238	187694.22	39.86	0.26	0.25	51.142
5	**Bear**	**43123**	0.997	38.89	0.254	161467.97	40.0	1.0	1.066	44.842
6	**Lena**	**22652**	0.998	38.88	0.146	153591.36	39.86	0.46	0.606	47.293
7	**Average**	**32062.7**	0.99733	38.88	0.19767	164713	39.8817	0.575	0.89217	47.011
8	**Interpretation**		THTB	THTB	TLTB	THTB	THTB	TLTB	TLTB	THTB
Interpretation: THTB = The Higher the Better, & TLTB = The Lower the Better										

Table 3 shows the performance analysis of the proposed EdSri model over a range of cover images under Huffman coding (lossless) constraints. The model was evaluated over eight quantitative measures: Structural Similarity Index (SSIM), Compression Ratio (CR), Compression Time (CT), Compression Speed (CS), Saving Percentage (SP), Bits Per Pixel (BPP), Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR). The interpretation rules categorize metrics as either The Higher the Better (THTB) or The Lower the Better (TLTB).

The SSIM values are high and constant for all test images, ranging from 0.995 to 0.999, with an average of 0.99733, reflecting outstanding structural fidelity between the reconstructed and original images. Analogously, CR values are close to being constant at 38.88%, reflecting the model's excellent compression performance.

The average figure of the Compression Time is 0.19767 which suggest a light load of compression. The various compression times range from 0.1606 to 0.254. The corresponding Compression Speed (CS), calculated in the units of processing throughput offers an average figure of 164713, thereby confirming the efficiency of EdSri for successful hosting.

The Saving Percentage is continuously high for images, thus, 39.86 – 40.0%, averaging to 39.8817%,. This implies a good characteristics of signal compression and picture quality. The mean value of Bit Per Pixel (BPP) IS 0.575 ranging from 0.26 to 1.0. These figures represent a good balance existing between the quality of an image and the efficiency of their compression.

The MSE values are between 0.25 and 1.497 with a fairly low average of 0.89217, confirming the low reconstruction error of the model. Additionally, the PSNR values are between 44.64 and 51.142 dB with an average of 47.011 dB, which reflects high visual quality and minimal noise in the reconstructed images.

In summary the outcomes of the EdSri model attains excellent similarity index, high compression presentation, slight unnoticed distortion and effectual compression. The negligible differences realized with the various images present the robustness and generalizability of the model's strength. It is worth-noting that all the THTB parameters constantly perform very well, and the TLTB parameters like BPP, CT, MSE also maintain high levels, hence, reinforcing the effectiveness of EdSri in lossless compression capacity.

Table 4. Wahab et al,. [30] model average results vs EdSri model average results

Measurement Parameters		SSIM	CR	CT	CS	SP	BPP	MSE	PSNR
Average	Wahab et al,. [30] Model	0.8450	35.02	0.236	118428	17.2	1.58	1.56	37.18
	EdSri Model (Proposed Model)	0.99733	38.88	0.19767	164713	39.882	0.575	0.89217	47.011
([30] - EdSri) Significance		-0.15233	-3.86	0.03833	-46285	-22.682	1.005	0.66783	-9.831
Significance Acronym		NVBR	NVBR	PVBR	NVBR	NVBR	PVBR	PVBR	NVBR
Significance Interpretation: NVBR = Negative Value Better Result, & PVBR = Positive Value Better Result									

Table 4 outlines a comparative study between the suggested EdSri model and the baseline method Wahab et al,. [30], based on average performance for eight key measurement parameters. The measures are Structural Similarity Index (SSIM), Compression Ratio (CR), Computation Time (CT), Compression Speed (CS), Saving Percentage (SP), Bits Per Pixel (BPP), Mean Squared Error (MSE), and Peak Signal-to-Noise Ratio (PSNR).

SSIM of the EdSri model (0.99733) is significantly better than the baseline (0.8450), indicating improved structural content preservation in reconstructed images. CR also does from 35.02% to 38.88%, which portrays better compaction strength. But as better CR is desirable (THTB), -3.86 negative difference is a Negative Value Better Result (NVBR).

EdSri proves to be much more computationally efficient, with CT being decreased from 0.236 s to 0.19767 seconds, and CS increasing from 118,428 to 164,713. Both increases are tagged as NVBR, since smaller CT and larger CS represent improved performance. The increase in CS by more than 46,000 units indicates heavily improved throughput.

There is a tangible improvement in saving percentage (SP) results. Thus, 17.2% to 39.882%. This 22.682% optimal performance pinpoints the effectiveness of EdSri in creating sparser outcome situation that is appreciable for efficient data compaction. The reduction in BPP from 1.58 to 0.575 (PVBR) also highlight the optimized coding efficacy.

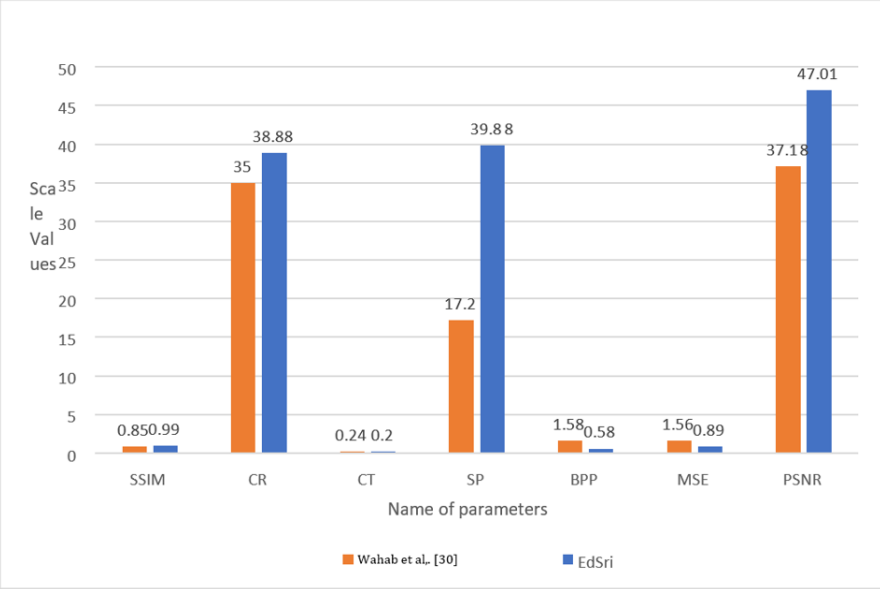


Fig.28. Bar chart comparing the performance of Wahab et al,. [30] model and EdSri model

MSE is decreased by 0.66783, from 1.56 to 0.89217, corresponding to improved reconstruction quality. This improvement is read as a Positive Value Better Result (PVBR). Concurrently, PSNR also increases drastically from 37.18 dB to 47.011dB corresponding to the improved visual quality and noise resistance. The difference is obtained as -9.831, the metric is defined as NVBR since greater PSNR is better.

In summary, the comparative analysis also attests that the envisioned EdSri model universally performs better than the baseline model in all primary performance indexes. Enhancement is particularly evident in SSIM, PSNR, SP, BPP, and CS, testifying to the superiority of EdSri in structural fidelity, coding sparsity, and computational complexity. The overall classification of the variations in NVBR and PVBR emphasizes the strength and power of these optimizations. The outputs reaffirm EdSri as a fully superior solution for lossless compression of data and reconstruction of signals, providing quantifiable advantages both in quality and efficiency over the 2021 benchmark.

Figure 28 displays comparative presentation parameters analysis existing between the recommended EdSri model and method defined by Wahab et al,. [30]. Evaluation involves seven (7) crucial parameters except Compression Speed, because of its high numbers, out of the eight (8) metrics (as mentioned in table 4).

As shown in the graph above, the EdSri model shows graphically a significant improvement on all the parameters. The SSIM value achieved by EdSri (0.99) far exceeds that of Wahab et al. (0.85), reflecting better maintenance of structural content and image fidelity. The model also gets a better Compression Ratio (38.88) compared to 35 for the baseline, which reflects its better compression efficiency. Moreover, Compression Time decreases from 0.24s to 0.20s, indicating that the EdSri algorithm achieves higher convergence speed and better computational efficiency.

A significant improvement is seen in the Saving Percentage (SP), where EdSri measures 39.88 compared to 17.2 for the reference model, as it indicates its optimal use of space and data. Likewise, both Bits Per Pixel (BPP) and Mean Squared Error (MSE) exhibit considerable decreases — 0.58 vs. 1.58 and 0.89 vs. 1.56, respectively—as these reflect improved compression at the cost of minimal image information loss. The Peak Signal-to-Noise Ratio (PSNR) obtained by EdSri (47.01 dB) also supports its supremacy, symbolizing a better reconstruction quality and reduced distortion than 37.18 dB of Wahab et al,. [30].

Overall, these results confirm that the suggested EdSri method provides a more effective and efficient image compression model. It not only minimizes redundancy and computational cost but also maximizes visual quality, performing better than the existing benchmark in all key performance metrics.

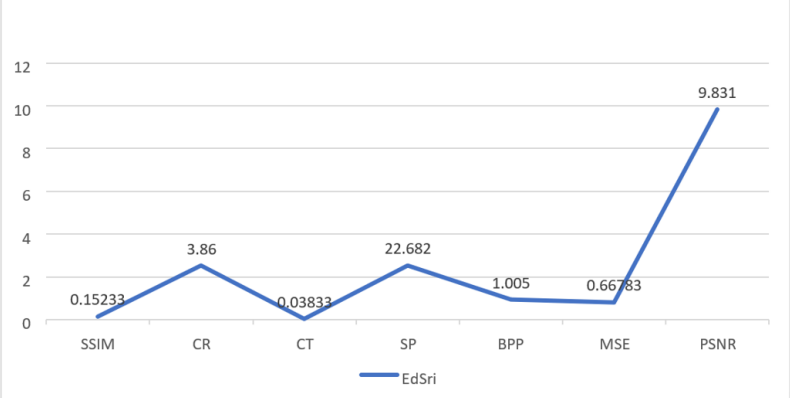


Fig.29. Line graph of improvement made (in positive values) by current proposed EdSri model

Figure 29 shows the improvement in performance obtained by the proposed EdSri method for seven most important evaluation metrics: SSIM, CR, CT, SP, BPP, MSE, and PSNR. The results show a mixed level of improvement, reflecting the strengths of the method as well as relative effectiveness over different areas of performance.

Interestingly, much larger improvements are seen in Saving Percentage (SP) and Peak Signal-to-Noise Ratio (PSNR), by 22.682 and 9.831, respectively. All the already mentioned figures show that EdSri algorithm outperforms the existing model in improving clarity of signal and the compression metrics proving higher performance of image quality and signal reconstruction application in the aspects of image genuineness and sparsity.

Regarding Compression Ratio (CR), a modest improvement of 3.86 was realized, which serves to indicate the efficacy of the approach to minimizing data size without unacceptable loss of quality. Increases in Mean Squared Error (MSE) and Bit Per Pixel (BPP), (0.66783) and (1.005) respectively are low, and less significant improvement in the efficiency of the application encoding. Hence, even though EdSri gains obviously high improvements in the majority of metrics, it only insignificantly enhances the aspects of similarity index, compression time and mean squared error.

Structural Similarity Index (SSIM) and Compression Time (CT) improvements are negligible, increasing by 0.15233 and 0.03833, respectively. This therefore, shows that even as the EdSri model made conspicuous enhancements in many aspects, the effectiveness of EdSri in the areas of similarity index and compression time is somewhat minor.

However, the totality of high improvement gains made by EdSri far exceeds the low gains only in the aspects of only similarity index, compression time and mean squared error.

4.1. Performance of the Added Security Metrics

The following parameters are the performance of the five (5) additional security metrics implemented against the RSA login system/interface to monitor and verify the system's both lawful and unlawful login attempts. The captures below are the exact snapshots from the registered user(s) email and the software interface of a user login effort(s), and browser/device details, which are missing elements in Wahab et al., [30] model paper of this research.

Password Strength : As per the system's access control/restriction displayed in figure 30, the minimum password length is eleven (11) which is made up of three (3) minimum uppercases, two (2) minimum lowercases, three (3) minimum numbers, and three (3) minimum special characters respectively. This strong password validation functionality helps to highly mitigate dictionary attacks, password cracking or hacking and bruteforce activities.

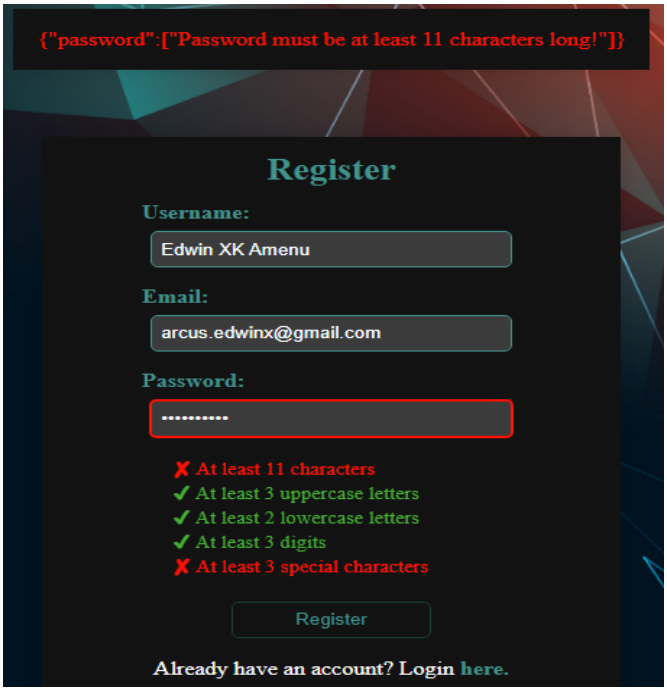


Fig.30. Evidence of user login interface validating authentication controls

Login Attempt Rate: For the purposes of sufficient and efficient regular system monitoring, the EdSri model app is configured to tract the number of logins attempts within every 30 minutes timeframe. The user is also limited to make only five (5) login attempts within the thirty (30) minutes window period. The figure 31 below portrays three (3) successful login attempts out of the five (5) times system limit.

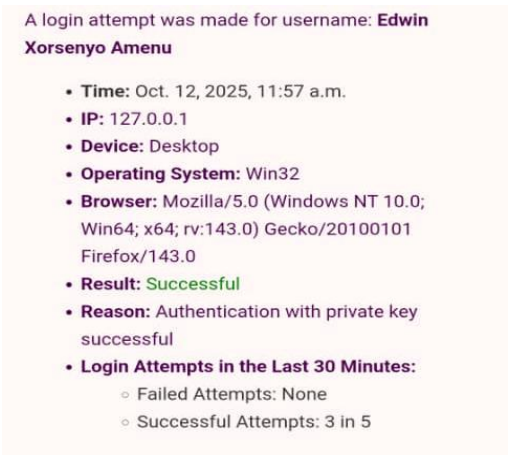
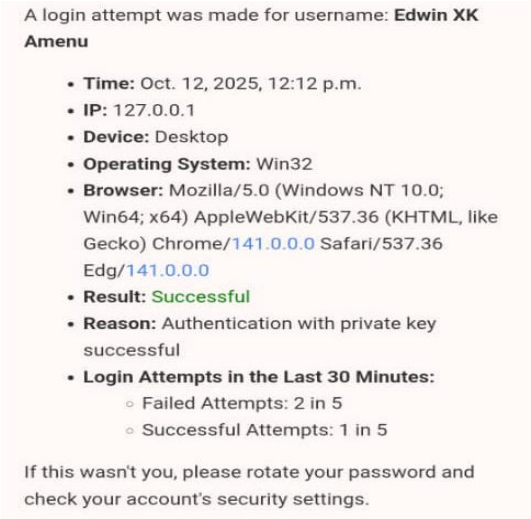


Fig.31. Evidence of login attempt rate details transported to registered user's email.

Failed Login Attempt Rate: The system has proven by evidence of figure 32 below that it tracks the number/percentage of failed login attempts, indicative of possible password guessing/credential stuffing/phishing attacks. It is also able to send emails of any attempted failed login to registered user, even as it is shown on the interface.



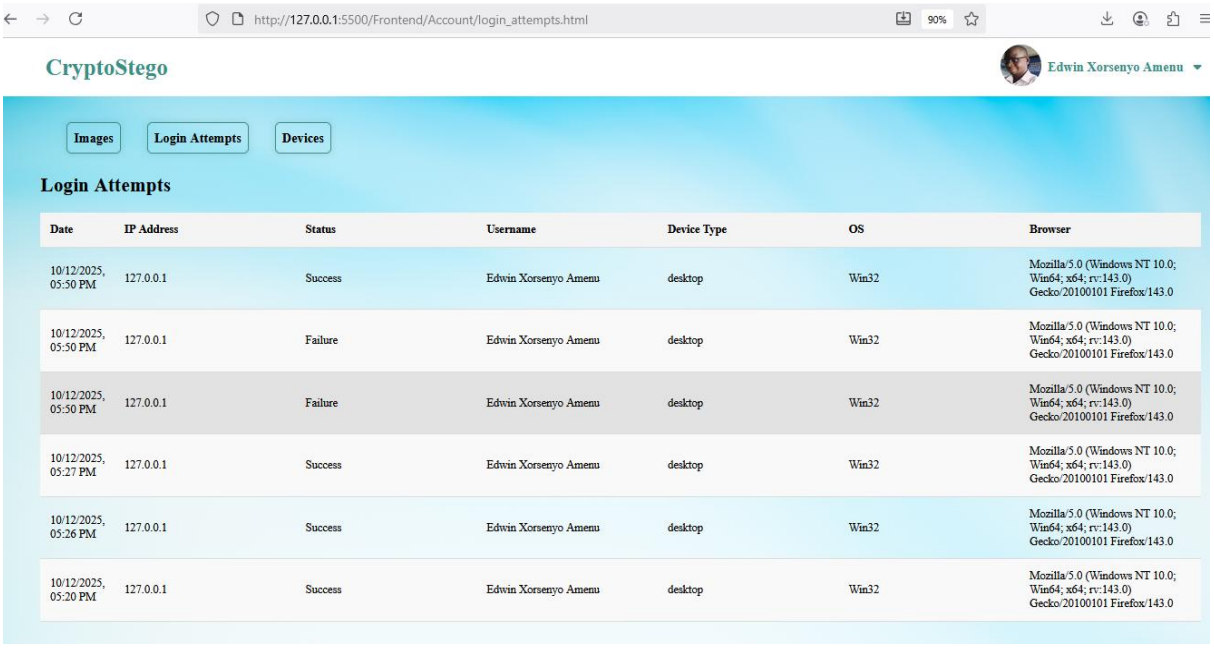
A login attempt was made for username: **Edwin XK Amenu**

- **Time:** Oct. 12, 2025, 12:12 p.m.
- **IP:** 127.0.0.1
- **Device:** Desktop
- **Operating System:** Win32
- **Browser:** Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/141.0.0.0 Safari/537.36 Edg/141.0.0.0
- **Result:** Successful
- **Reason:** Authentication with private key successful
- **Login Attempts in the Last 30 Minutes:**
  - Failed Attempts: 2 in 5
  - Successful Attempts: 1 in 5

If this wasn't you, please rotate your password and check your account's security settings.

Fig.32. Evidence of failed login attempt details sent to registered user's email

Device and Browser Fingerprinting: The EdSri system model, as per the screenshot below proves to collect device and browser details such as device types, OS, browser version so as to identify trusted device sources. It therefore helps to detect suspicious login attempts from unknown untrusted sources. Such details are displayed on the interface, and also sent to the legitimate user's email.



Date	IP Address	Status	Username	Device Type	OS	Browser
10/12/2025, 05:50 PM	127.0.0.1	Success	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0
10/12/2025, 05:50 PM	127.0.0.1	Failure	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0
10/12/2025, 05:50 PM	127.0.0.1	Failure	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0
10/12/2025, 05:27 PM	127.0.0.1	Success	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0
10/12/2025, 05:26 PM	127.0.0.1	Success	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0
10/12/2025, 05:20 PM	127.0.0.1	Success	Edwin Xorsenyo Amenu	desktop	Win32	Mozilla/5.0 (Windows NT 10.0; Win64; x64; rv:143.0) Gecko/20100101 Firefox/143.0

Fig.33. Device and browser fingerprinting, and time between login attempts interface

Time Between Login Attempts: The analyses between the login attempts show, it takes averagely two (2) minutes between two different successful login attempts, but less than a minute in between two (2) different unsuccessful login attempts as shown in the figure 33 above. This less-than-a-minute interval between the unsuccessful login attempts highly depicts the immediate next login attempt of a frustrated user who might be guessing their forgotten password, whilst the two minutes average successful interval is indicative of a normal registered user login, logout and login scenario. In addition, from the capture proof above in figure 32, every registered user login attempt is also sent into the registered user's email, even as it is displayed on the system's interface. This therefore helps the user to take action if hacking attempt is suspected on their registered account.

4.2. Measuring Compression Performances

A number of conditions can be used to measure the efficacy of the compression algorithm. The following parameters are used in this research for the evaluation of the compressed data.

A. Compression Ratio (CR)

The Compression Ratio (CR) is critical to measuring the pressure of the efficiency. It defines the ratio of the original message as against the compressed message. The data/image quality improves with increasing compression ratio. It therefore, means that the higher the CR, the better the compression. The CR is expressed mathematically as (CR = Original Image OR Data Size/Compressed Image or Data Size [48]

B. Compression Time (CT)

The Compression Time explains the amount of time taken by the technology to compress the original image or data into a more efficient non-redundant data image. CT is known as the smaller the compressed value the better efficient the data/image [49]

C. Compression Speed (CS)

This is observed by the percentage of the size of the compressed output file per unit time required to compress the file in seconds (CS = Compressed File Size/Compressed Time). The higher the value, the better the speed of compression [50]

D. Saving Percentage (SP)

SP is the reduced value size of the compressed text or image as against the uncompressed file value. The bigger the compressed percentage, the better the result is considered. SP = (Uncompressed File Size – Compressed File Size)/Original * File Size [51].

E. Bits Per Pixel (BPP)

This is the data saved or stored by the bits of each pixel in a specific file. It is the overall number of bit recorded in a given reduced file by the original file. BPP = Number of bits in a compressed file/Total Number of pixels in the File. The larger the Bits Per Pixel, the bigger the Memory Storage Space needed to save the file, and vice versa. However, the closer the BPP amount is to zero, the better efficiency [52]

F. Mean Squared Error (MSE)

MSE is calculated by determining the changes made to the compressed file as against the state of the original uncompressed file. It helps to evaluate the quality of the compressed file in comparison to the uncompressed file. The closer the MSE percentage is to zero (0), the better, the quality of the result. Therefore, if the MSE assumes zero, it means the compressed file and the uncompressed file are almost the same, and can therefore be termed as lossless data reduction [53].

$$\text{MSE} = 1/n\sum(yi+yi)2 \tag{12}$$

$$i=1 \tag{13}$$

Definition 1: n is the total number of data point
Definition 2: I is the actual value of the i-th data point,
Definition 3: yi is the estimated (predicted) value of the i-th data point,
Definition 4: ∑ sumps up all the value of the ŷ operations.
Definition 5: (yi−ŷi)2 is the squared difference between the actual and estimated values of the i-th data point.

G. Peak Signal to Noise Ratio (PSNR)

It is the differentia ratio between the strength of the signal and the noise which occurs in the signals. This formula is contingent on the file quality after compression. The better the PSNR, the better the file quality, or the lesser the MSE, the higher the PSNR = 10log10 (MAX2/MSE). Higher PSNR is indicative of better outcome. MAX2 is 2552 being the highest strength of pixels in the perfect file. The quality of the file will be checked by the use of PSNR. The higher the PSNR (more than 40Db) the better the file quality [54].

H. Structural Similarity Index (SSIM)

It is the perceived measurement difference that occurs between 2 alike files. It is challenging to figure out which of the files/objects has undergone any kind of processing or compression [55]

$$\text{SSIM}=((2\mu x\mu y+C1)(2\sigma x\sigma y+C2))/((\mu x2+\mu y2+C1)(\sigma x2+\sigma y2+C2)) \tag{14}$$

Definition 1: μ_x *and* μ_y are explained using illuminating each image/file in the x and y directions.
Definition 2: σ_x and σ_y stands for the standard deviations, as well as the estimated contrast between the signals.
Definition 3: C1 and C2 are negligible use in managing the instability for either of the files signaling its closeness to

zero. Nevertheless, the constants are avoidable in the case of global quality indexes. The closer SSIM is to figure one (1), the better its value to unifying the two files. Thus, the higher the performance of each image.

4.2. Experimental Setup

The current methodology is assisted by the use of python 3.11 flavor/version programming language. Python Language as a choice for this project is engineered by the fact that it supports efficiently and effectively data analytics, visualization, computation and simulation. It is also helpful in interacting with application development process for data encryption, compression as well as decryption and decompression. The many libraries available for this language also gives the functionality for outstanding graph integration capabilities. This program is run on windows 10 Pro, because it accommodates the installation of all python libraries for application development and usage at all levels. The operating system is installed on a hardware HP fifth generation personal computer consisting specifications of Intel(R) core i5 processor speed of 2.20GHz, RAM of 8:00GB, and 1TB of harddisk capacity. The sizes of all the six (6) varied black and white, and color pictures used as original images which are processed into stego images range from 28.4 kilobytes as the least, to 49.2 kilobytes, the biggest size. The windows 10 Pro system software, notepad is used as the measuring tool to generate all the various dataset sizes.

5. Conclusions and Future Work

The use of Huffman encoding together with RSA, AES and LSB is suggested strongly by the authors of this research. This is because it is useful in removing redundancy in the intended to-be-communicated text, along with embedding the encrypted text into an original image, which serves as a cover image with the intent to communicate light weight text within high quality stego image. The RSA technology is used in EdSri model for user registration and login to withstand any form of bruteforce attack at the gateway. AES algorithm is employed for text encryption to enhance security of the app. Again, in this EdSri architecture, the tool that can be used to compress the text in a lossless manner is discussed extensively. The compressed text with the stego image for storage and transmission are measured on the parameters of SSIM, CT, CR, BPP, SP, MSE, and PNSR, with higher percentage differences of 0.78%, 99.9%, 0.15%, 31%, 0.45%, 0.69% and 36.6% respectively in each case.

When the simulation outcome of this current work was compared to the already existing research, Wahab et al., [30] model, it pointed out that the overall average of all the measured parameters outperformed the, existing model significantly, which means the current implementation is comparatively a comprehensive overall improvement. As a future enhancement to EdSri, it may be planned to further improve on the overall performance of all the parameters involved in EdSri proposed work, especially, the compression time (CT) of apple and bear images by re-engineering the algorithm, and to also measure more security metrics such as geographical login distribution, account lockout frequency, login attempt patterns, among others. In addition, Discrete Wavelet Transform (DWT) algorithm will be reconstructed to compress the cover image to get a lighter image with a lightweight text implanted in it so as to further maximize storage space in the cloud environment.

Acknowledgement

We express our sincere gratitude to God almighty, who always grants us the ultimate strength, and also, beloved, Gloria Boafo for constantly giving the corresponding author external motivation in keeping him enrooted to put together this paper.

Funding Information

The research is solely funded by the corresponding author

Conflict of Interest

The authors have no conflicts of interest to declare

Author Contributions

This research was discovered by Edwin Xorsenyo Amenu. The writing and coding are all done by him. Dr R. Sridaran approved of it, and proofread and edited it.

Data Availability

Every data is currently on the corresponding authors' computer, and available upon request. In the near future after patency, the program will be hosted with every single data available online.

Ethical Approval

Not Applicable

Informed Consent

Not Applicable

Statement Regarding Research Involving Human Participants and/or Animals

Not Applicable

Consent to Participate

Not Applicable

Consent to Publish

Not Applicable (Open Access)

Competing Interests

Not Applicable

References

[1] C. Wu, A. N. Toosi, R. Buyya, and K. Ramamohanarao, “Hedonic Pricing of Cloud Computing Services,” IEEE Trans. Cloud Computing., vol. 9, no. 1, pp. 182–196, 2021, doi: 10.1109/TCC.2018.2858266.

[2] M. A. Islam, M. A. -A. K. Riad and T. S. Pias, "Enhancing Security of Image Steganography Using Visual Cryptography," 2021 2nd International Conference on Robotics, Electrical and Signal Processing Techniques (ICREST), DHAKA, Bangladesh, 2021, pp. 694-698, doi: 10.1109/ICREST51555.2021.9331225.

[3] K. C. Nunna and R. Marapareddy, “Secure data transfer through internet using cryptography and image steganography,” Conf. Proc. - IEEE SOUTHEASTCON, vol. 2, 2020, doi: 10.1109/SoutheastCon44009.2020.9368301.

[4] S. Selvarajan, M. Mohan, and B. R. Chandavarkar, “Techniques to Secure Address Resolution Protocol,” 2020 11th Int. Conf. Comput. Commun. Netw. Technol. ICCCNT 2020, 2020, doi: 10.1109/ICCCNT49239.2020.9225413.

[5] M. Guillermo et al., “Implementation of Automated Annotation through Mask RCNN Object Detection model in CVAT using AWS EC2 Instance,” IEEE Reg. 10 Annu. Int. Conf. Proceedings/TENCON, vol. 2020-Novem, pp. 708–713, 2020, doi: 10.1109/TENCON50793.2020.9293906.

[6] B. Bharathkumar, S. Logeswar, S. Sudharsun, and B. Karthikeyan, “An Enhanced Triple Prime Encryption Approach for Image Encryption in LSB Steganography,” 2023 1st Int. Conf. Adv. Electr. Electron. Comput. Intell. ICAEECI 2023, pp. 24–27, 2023, doi: 10.1109/ICAEECI58247.2023.10370817.

[7] F. Al-Shaarani and A. Gutub, “Securing matrix counting-based secret-sharing involving crypto steganography,” J. King Saud Univ. - Comput. Inf. Sci., vol. 34, no. 9, pp. 6909–6924, 2021, doi: 10.1016/j.jksuci.2021.09.009.

[8] L. Negi and L. Negi, “Image Steganography Using Steg with AES and LSB,” 7th Int. Conf. Comput. Eng. Des. ICCED 2021, pp. 4–7, 2021, doi: 10.1109/ICCED53389.2021.9664834.

[9] M. Tahir, M. Sardaraz, Z. Mehmood, and S. Muhammad, “CryptoGA: a cryptosystem based on genetic algorithm for cloud data security,” Cluster Comput., vol. 24, no. 2, pp. 739–752, 2021, doi: 10.1007/s10586-020-03157-4.

[10] T. R. Prasad and J. Naulegari, "Investigation of symmetric and asymmetric eye patterns for bell’s palsy diagnosis," 2023 OITS International Conference on Information Technology (OCIT), Raipur, India, 2023, pp. 655-659, doi: 10.1109/OCIT59427.2023.10431126.

[11] N. Kshetri and N. York, “algoTRIC: Symmetric and asymmetric encryption algorithms for Cryptography - A comparative analysis in AI era,” no. December, pp. 1–18, 2024. https://doi.org/10.48550/arXiv.2412.15237

[12] B. Preksha, R. Harish, B. Sreenivas, and M. Vasanthalakshmi, “Image Steganography using RSA Algorithm for Secure Communication,” 2021 IEEE Int. Conf. Mob. Networks Wirel. Commun. ICMNWC 2021, pp. 4–7, 2021, doi: 10.1109/ICMNWC52512.2021.9688352.

[13] A. Ahyuna, S. Syamsuddin, H. Hasriani, A. Ardimansyah, I. Irmawati, and S. Wahyuni, “The Application of LSB Steganography for Secure Text and Hiding Confidential Information Using AES Cryptography,” 3rd Int. Conf. Cybern. Intell. Syst. ICORIS 2021, pp. 10–13, 2021, doi: 10.1109/ICORIS52787.2021.9649497.

[14] C. L. Krishna, P. Anudeep, C. S. N. V. S. V. Lakshmi, M. V. P. C. S. Rao, S. V. Chereddy, and B. R. Krishna, “Concealment of Data using RSA Cryptography and Steganography Techniques,” Proc. 2023 2nd Int. Conf. Electron. Renew. Syst. ICEARS 2023, no. April, pp. 778–783, 2023, doi: 10.1109/ICEARS56392.2023.10085355.

[15] T. S. Deepak and V. Enireddy, “High Payload Capacity using Steganography Combined with Cryptography,” Proc. 5th Int. Conf. I-SMAC (IoT Soc. Mobile, Anal. Cloud), I-SMAC 2021, pp. 1448–1453, 2021, doi: 10.1109/I-SMAC52330.2021.9640859.

[16] A. Gopinath and M. Ravisankar, “Comparison of Lossless Data Compression Techniques,” in Proceedings of the 5th International Conference on Inventive Computation Technologies, ICICT 2020, Institute of Electrical and Electronics Engineers Inc., Feb. 2020, pp. 628–633. doi: 10.1109/ICICT48043.2020.9112516.
[17] Y. Wu, S. Xu, and C. Xi, “A Dynamic and Parallel Two-Stage Lossless Data Compression Method for Smart Grid,” IEEE Access, vol. 11, no. November, pp. 143475–143485, 2023, doi: 10.1109/ACCESS.2023.3343436.
[18] S. Abed, N. H. Al-Huwais, Y. A. Atiyah, S. Parvin, and A. Gawanmeh, “An Improved Least Significant Bit Image Steganography Method,” in 2020 4th International Conference on Multimedia Computing, Networking and Applications, MCNA 2020, Institute of Electrical and Electronics Engineers Inc., Oct. 2020, pp. 90–96. doi: 10.1109/MCNA50957.2020.9264299.
[19] A. K. Jha, A. Shankar, R. R. Borana, and C. Gururaj, “Compression in communication security using huffman encoding and cipher block chaining,” Proc. 2021 1st Int. Conf. Adv. Electr. Comput. Commun. Sustain. Technol. ICAECT 2021, 2021, doi: 10.1109/ICAECT49130.2021.9392490.
[20] P. Mishra, C. Bhaya, A. K. Pal, and A. K. Singh, “Compressed DNA Coding Using Minimum Variance Huffman Tree,” IEEE Commun. Lett., vol. 24, no. 8, pp. 1602–1606, Aug. 2020, doi: 10.1109/LCOMM.2020.2991461.
[21] B. Song, M. Deng, S. R. A. J. Pokhrel, Q. Lan, and R. Doss, “Digital privacy under attack: challenges and enablers”,08 November 2025. doi:10.1145/3759487
[22] W. Nie, X. Xiao, Z. Wu, Y. Wu, F. Shen, and X. Luo, “The research of information security for the education cloud platform based on appscan technology,” Proc. - 5th IEEE Int. Conf. Cyber Secur. Cloud Comput. 4th IEEE Int. Conf. Edge Comput. Scalable Cloud, CSCloud/EdgeCom 2018, pp. 185–189, 2018, doi: 10.1109/CSCloud/EdgeCom.2018.00040.
[23] M. Sharifzadeh, M. Aloraini, and D. Schonfeld, “Adaptive Batch Size Image Merging Steganography and Quantized Gaussian Image Steganography,” IEEE Trans. Inf. Forensics Secur., vol. 15, no. c, pp. 867–879, 2020, doi: 10.1109/TIFS.2019.2929441.
[24] N. F. Soliman, M. I. Khalil, A. D. Algarni, S. Ismail, R. Marzouk, and W. El-Shafai, Efficient HEVC steganography approach based on audio compression and encryption in QFFT domain for secure multimedia communication, vol. 80, no. 3. Multimedia Tools and Applications, 2021. doi: 10.1007/s11042-020-09881-8.
[25] T. Fupeng, “Research on Education Big Data Security Strategy under Network Environment,” 2021 IEEE 6th Int. Conf. Big Data Anal. ICBDA 2021, pp. 19–22, 2021, doi: 10.1109/ICBDA51983.2021.9403030.
[26] M. M. Mahmoud and H. T. Elshoush, “Enhancing LSB Using Binary Message Size Encoding for High Capacity, Transparent and Secure Audio Steganography-An Innovative Approach,” IEEE Access, vol. 10, pp. 29954–29971, 2022, doi: 10.1109/ACCESS.2022.3155146.
[27] B. Kumar Pandey et al., “Encryption and steganography-based text extraction in IoT using the EWCTS optimizer,” Imaging Sci. J., vol. 69, no. 1–4, pp. 38–56, 2021, doi: 10.1080/13682199.2022.2146885.
[28] R. Adee and H. Mouratidis, “A Dynamic Four-Step Data Security Model for Data in Cloud Computing Based on Cryptography and Steganography,” pp. 1–23, 2022. https://doi.org/10.3390/s22031109
[29] O. F. Abdelwahab, A. I. Hussein, H. F. A. Hamed, H. M. Kelash, and A. A. M. Khalaf, “Efficient Combination of RSA Cryptography, Lossy, and Lossless Compression Steganography Techniques to Hide Data,” Procedia Comput. Sci., vol. 182, pp. 5–12, 2021, doi: 10.1016/j.procs.2021.02.002.
[30] O. F. A. Wahab, A. A. M. Khalaf, A. I. Hussein and H. F. A. Hamed, "Hiding Data Using Efficient Combination of RSA Cryptography, and Compression Steganography Techniques," in IEEE Access, vol. 9, pp. 31805-31815, 2021, doi: 10.1109/ACCESS.2021.3060317
[31] L. Negi, S. Kumar, and M. Bharti, “A Hybrid Data Security Technique Using Chaos Encryption, RC4 Encryption, Huffman Data Compression and LSB Steganography,” Proc. - 2nd IEEE Int. Conf. Device Intell. Comput. Commun. Technol. DICCT 2024, no. May, pp. 288–293, 2024, doi: 10.1109/DICCT61038.2024.10532929.
[32] E. Lateef and S. Shakir, “A Secure and High-Capacity Image Steganography Approach Using Huffman Coding and RSA Encryption,” vol. 15, no. 2, 2023. doi: https://doi.org/10.29304/jqcm.2023.15.2.1231
[33] S. Rahman et al., “OPEN A Huffman code LSB based image steganography technique using multi - level encryption and achromatic component of an image,” Sci. Rep., pp. 1–19, 2023, doi: 10.1038/s41598-023-41303-1.
[34] A. M. Elgaier and A. M. Abushaala, "An Effect of Huffman Encoding with Hybrid Edge Detection in LSB Image Steganography," 2024 IEEE 4th International Maghreb Meeting of the Conference on Sciences and Techniques of Automatic Control and Computer Engineering (MI-STA), Tripoli, Libya, 2024, pp. 403-409, doi: 10.1109/MI-STA61267.2024.10599686.
[35] W. A. Awadh, A. S. Alasady, and A. K. Hamoud, “cryptography, and image steganography Hybrid information security system via combination of compression, cryptography, and image steganography,” no. September, pp. 6574–6584, 2022, doi: 10.11591/ijece.v12i6.pp6574-6584.
[36] A. Agrawal “CSIS_ Compressed sensing-based enhanced-embedding capacity image steganography scheme, 2021 - IET Image Processing - Wiley Online Library.pdf.” 08 March 2021 https://doi.org/10.1049/ipr2.12161
[37] E. X. Amenu, S. Rajagopal, and I. Science, “Optimizing the Security and Privacy of Cloud Data Communication; Hybridizing Cryptography and Steganography Using Triple Key of AES, RSA and LSB with Deceptive QR Code Technique: A Novel Approach,” pp. 1–7, 2024.
[38] S. R. Raiyan, “SCR EED S OLO: A Secure and Robust LSB Image Steganography Framework with Randomized Symmetric Encryption and Reed – Solomon Coding” 10 Aug 2025, https://doi.org/10.48550/arXiv.2503.12368.
[39] S. E. Ghrare, M. A. Abouras, and I. A. Akermi, “African Journal of Advanced Pure and Applied Sciences (AJAPAS) Development of Hybrid Data Security System using LSB Steganography and AES Cryptography,” vol. 3, no. 2, pp. 86–95, 2024.
[40] Jan, A., Parah, S.A., Hussan, M. et al. Double layer security using crypto-stego techniques: a comprehensive review. Health Technol. 12, 9–31 (2022). https://doi.org/10.1007/s12553-021-00602-1
[41] M. Sites et al., “A Dynamic and Efficient Crypto-Steganography System for Securing Multiple Files in Cloud,” no. February, pp. 21–24, 2023, doi: 10.1109/KST57286.2023.10086908.
[42] N. Manohar and P. V. Kumar, “Data Encryption Decryption Using Steganography,” Proc. Int. Conf. Intell. Comput. Control Syst. ICICCS 2020, no. Iciccs, pp. 697–702, 2020, doi: 10.1109/ICICCS48265.2020.9120935.
[43] S. Sarkar and M. Nandan, "Password Strength Analysis and its Classification by Applying Machine Learning Based Techniques," 2022 Second International Conference on Computer Science, Engineering and Applications (ICCSEA), Gunupur, India, 2022, pp. 1-5, doi: 10.1109/ICCSEA54677.2022.9936117

[44] L. Wang, “Design and Analysis of Network Security Access Control System,” Proc. - 2019 3rd Int. Conf. Data Sci. Bus. Anal. ICDSBA 2019, pp. 426–429, 2019, doi: 10.1109/ICDSBA48748.2019.00092.
[45] S. Wiefling, S. Thunem, and L. L. O. Iacono, “Pump Up Password Security! Evaluating and Enhancing Risk-Based Authentication on a Real-World Large-Scale,” vol. 26, no. 1, 2022, doi: 10.1145/3546069.
[46] Jan, A., Parah, S.A., Hussan, M. et al. Double layer security using crypto-stego techniques: a comprehensive review. Health Technol. 12, 9–31 (2022). https://doi.org/10.1007/s12553-021-00602-1
[47] S. S. Gujar, "Exploring Device Fingerprinting for Password-Less Authentication Systems," 2024 Global Conference on Communications and Information Technologies (GCCIT), BANGALORE, India, 2024, pp. 1-7, doi: 10.1109/GCCIT63234.2024.10862062
[48] H. Shiomi, T. Shimobaba, T. Kakue, and T. Ito, “Lossless Compression Using the Ramanujan Sums: Application to Hologram Compression,” IEEE Access, vol. 8, pp. 144453–144457, 2020, doi: 10.1109/ACCESS.2020.3014979.
[49] P. Mishra, C. Bhaya, A. K. Pal, and A. K. Singh, “Compressed DNA Coding Using Minimum Variance Huffman Tree,” IEEE Commun. Lett., vol. 24, no. 8, pp. 1602–1606, 2020, doi: 10.1109/LCOMM.2020.2991461.
[50] P. Roy, D. Roy and R. Banerjee. “A Proficient Video Compression using Block based Fragmented Cosine Transform (BFCT) in Wireless Multimedia Sensor Network (WMSN),” pp. 10–13, 2020, doi: 10.1109/NCETSTEA48365.2020.9119930.
[51] G. Shrividhiya, K. S. Srujana, S. N. Kashyap, and C. Gururaj, “Robust data compression algorithm utilizing LZW framework based on huffman technique,” 2021 Int. Conf. Emerg. Smart Comput. Informatics, ESCI 2021, pp. 234–237, 2021, doi: 10.1109/ESCI50559.2021.9396785.
[52] O. F. Abdelwahab, A. Steganography, I. Hussein, H. F. A. Techniques, and M. Hamdy, “ScienceDirect ScienceDirect ScienceDirect Efficient Combination RSA Cryptography, Lossless Learning of Compression Steganography Techniques to Hide Data Efficient Combination of RSA Cryptography, Lossless Efficient Combination of RSA Cryptography, Lossy, and Lossless Compression Hide Data Compression Steganography Techniques to Hide Data,” Procedia Comput. Sci., vol. 182, pp. 5–12, 2021, doi: 10.1016/j.procs.2021.02.002.
[53] R. Aswathy Krishna, N. Subramanian, and K. Jain, “Image Steganography Using HBV And Padded RSA,” ISDFS 2023 - 11th Int. Symp. Digit. Forensics Secur., no. May, pp. 11–12, 2023, doi: 10.1109/ISDFS58141.2023.10131891.
[54] M. Alanzy, R. Alomrani, B. Alqarni, and S. Almutairi, “applied sciences Image Steganography Using LSB and Hybrid Encryption Algorithms,” 2023, https://doi.org/10.3390/app132111771.
[55] D. Mehta and D. Bhatti, “Blind image steganography algorithm development which resistant against JPEG compression attack,” Multimed. Tools Appl., vol. 81, no. 1, pp. 459–479, 2022, doi: 10.1007/s11042-021-11351-8.

Authors’ Profiles

Edwin Xorsenyo Amenu is pursuing his PhD. in Computer Science (Cybersecurity) in Marwadi University, Rajkot -India. As part of his appointment in this university, he is responsible for teaching, research, mentoring, assisting and, he is also the cybersecurity lab-in-charge. Edwin X.K Amenu studies at Kwame Nkrumah University of Science and Technology, Kumasi - Ghana, for his Masters in Information Technology. Before enrolling on his full-time PhD. studies, he was part of the Academic staff of the University of Health and Allied Sciences, Ho - Ghana. He earns his Bachelor Degree in Information Science and English-Combined Major from the University of Ghana, Accra. He has been in the teaching field and professional industry for the past fifteen (15) years, with ambitious research interests in the areas of cybersecurity, information security and computer system security.

Professor (Dr.) Sridaran Rajagopal is the current Dean of the Faculty of Computer Applications at Marwadi University, He obtained his PhD. in Computer Science from the University of Indian Institute of Technology at Bombay. He completed his Masters and Bachelor Degree in Computer Applications and Computer Science from the University of National Institute of Technology, Trichy and University of Madras respectively. He is currently the Chairman of the International Conference on Advancements in Smart Computing and Information Security (ASCIS). With plethora of published papers to his credit, he is an astute researcher and has served as a keynote speaker on many national and international platform.

How to cite this paper: Edwin Xorsenyo Amenu, Sridaran Rajagopal., "EdSri: An Enhanced Approach Towards Optimizing Cloud Communication by Hybridizing Cryptography and Steganography with Huffman Compression Method", International Journal of Information Technology and Computer Science(IJITCS), Vol.18, No.1, pp.14-41, 2026. DOI:10.5815/ijitcs.2026.01.02