INTERNATIONAL JOURNAL OF NOVEL RESEARCH AND DEVELOPMENT (IJNRD)
© 2026 IJNRD | Volume 11, Issue 4, April 2026 | ISSN: 2456-4184 | IJNRD.ORG

Secure Data Transmission In Internet

Gunaseelan B Department of Computer Science-(PG) Hindustan College of Arts and Science- Coimbatore, India
24mca028@hicas.ac.in
Dr. Sri Devi R Department of Computer Science (PG) Hindustan College of Arts and Science- Coimbatore, India
ORCID:0000-0002-3946-4967

Abstract The rapid expansion of cloud-based communication has made secure image transmission a critical engineering challenge. Conventional single-layer encryption schemes, while mathematically sound, remain vulnerable to traffic analysis and targeted attacks that exploit key-management weaknesses. This paper presents a dual-layer security framework designed to address both confidentiality and traffic-level concealment simultaneously. In the first layer, a hybrid chaotic map combining the logistic map and the Lorenz system generates a pseudo-random key with a key space exceeding 10^{256}, which is applied through an XOR cipher to encrypt the image. In the second layer, the encrypted image is converted into a Waveform Audio (WAV) file, disguising the transmitted data as innocuous audio content and thereby frustrating traffic-analysis attempts. The receiver reverses both operations sequentially to recover the original image. The system is implemented in Python using Py Crypto dome, the built-in wave module, and the Boto3 SDK for cloud storage integration. Experimental results demonstrate that the encrypted output passes standard statistical randomness tests and achieves a Peak Signal-to-Noise Ratio (PSNR) of zero, confirming complete pixel decorrelation. The web-based interface requires no cryptographic expertise from end users, making the system practical for deployment in healthcare, legal, and personal cloud-storage scenarios. Future directions include GPU-accelerated key generation and integration with quantum-resistant algorithms.

Keywords: *chaotic encryption; image security; steganography; XOR cipher; cloud storage; hybrid chaotic map; audio format conversion*

I. INTRODUCTION

The proliferation of cloud computing platforms has transformed the way individuals and organisations store, share, and retrieve sensitive digital content. Among the data types most frequently handled in such environments, digital images occupy a particularly sensitive category, encompassing personal photographs, medical imaging records, legal documents, and confidential scans. The transmission of such content over public internet infrastructure presents a persistent and growing security challenge. Contemporary cloud providers offer server-side encryption as a standard feature. However, this arrangement places full trust in the provider's internal security practices and provides no protection if credentials are compromised or if the data is intercepted during transit before it reaches the provider's infrastructure. Client-side encryption, where the data is secured before it leaves the sender's device, is a more robust alternative, but adoption remains limited because most available tools impose a significant usability burden on non-technical users.

A further and often overlooked weakness of encryption alone is that it does not conceal the existence of protected content. An adversary performing passive traffic analysis can readily identify encrypted image data based on packet size, transmission timing, and content-type metadata, and may then direct concentrated resources against that specific traffic. This is the limitation that steganographic techniques address: by disguising data as a different content type, steganography attempts to remove the signal that invites targeted attack. This paper proposes and implements a dual-layer secure image transmission framework that addresses both weaknesses simultaneously. The first layer applies a hybrid chaotic-map XOR cipher with a key space exceeding 10^{256} values, providing strong cryptographic protection. The second layer converts the encrypted image into a lossless WAV audio file before transmission, applying steganographic obscuration at the format level. Together, these layers provide defence in depth that neither approach achieves alone.

The remainder of this paper is organised as follows. Section II defines the problem formally. Section III states the system objectives. Section IV surveys related work. Section V describes the proposed system. Section VI details the methodology. Section VII presents the system design including architecture, data flow, and entity-relationship diagrams. Section VIII describes the implementation. Section IX presents and discusses experimental results. Sections X through XII address advantages, limitations, and future scope. Section XIII concludes the paper.

II. PROBLEM STATEMENT

Despite widespread adoption of cloud storage services, secure image transmission remains an inadequately solved problem in two distinct dimensions. First, most available solutions rely on a single layer of protection, typically symmetric-key encryption applied at the transport layer or server side. A single-layer approach provides no defence in depth: a weakness in the encryption algorithm, a compromised key, or a misconfigured access control policy is sufficient to expose the protected content entirely. Second, encryption in isolation does not address traffic analysis. An adversary who observes the transmission channel can identify encrypted image data through characteristic packet sizes and content-type signatures, even without breaking the underlying cipher. This identification is sufficient to motivate targeted interception and sustained decryption attempts. Existing steganographic tools, while addressing this second problem, typically sacrifice cryptographic strength in favour of concealment capacity, leaving the data vulnerable if the steganographic layer is detected.

The result is a gap in the available tooling: no lightweight, user-accessible system currently provides both strong cryptographic protection and format-level traffic concealment in an integrated pipeline. This work addresses that gap directly.

III. OBJECTIVES

The specific objectives of this work are as follows:

1. To design a hybrid chaotic map combining the logistic map and the Lorenz system capable of generating pseudo-random keys with a key space exceeding 10^{256}.
2. To implement an XOR-based image encryption module that is computationally efficient and produces statistically random output.
3. To develop an image-to-WAV audio conversion module that constitutes the steganographic layer of the pipeline.
4. To integrate both layers into a unified, end-to-end pipeline covering encryption, format conversion, cloud upload, retrieval, reverse conversion, and decryption.
5. To provide cloud storage integration using the Amazon S3 API via the Boto3 SDK.
6. To validate the system's security properties using standard image quality metrics and statistical randomness tests.
7. To deliver a web-based interface that requires no cryptographic expertise from end users.

IV. LITERATURE REVIEW

The literature on cloud image security spans three primary research traditions: cryptographic encryption, reversible data hiding in encrypted images (RDHEI), and steganography. Each tradition offers partial solutions, and the gaps between them motivate the integrated approach taken in this work.

In the domain of cryptographic image protection, chaotic maps have attracted sustained research interest as sources of pseudo-random key material. Their sensitivity to initial conditions, ergodicity, and unpredictability make them well suited to key generation for symmetric ciphers [1]. Early work established the logistic map as a practical basis for image encryption, but later studies identified weaknesses in single-map approaches under chosen-plaintext attacks. Subsequent proposals combined multiple chaotic systems to enlarge the effective key space and improve resistance to cryptanalysis [5], [12].

RDHEI schemes, surveyed comprehensively by several authors [2], [3], address a specific cloud use case where a data hider must embed a payload into an already-encrypted image without access to the original content. Techniques based on multi-prediction error expansion [3] and Huffman compression [9] have achieved substantial embedding capacities. Shamir's Secret Sharing has also been

applied in this setting [10] to support distributed storage across multiple cloud providers. While these methods are valuable for cloud watermarking and provenance, they introduce workflow complexity that limits practical deployment.

Homomorphic encryption offers a theoretically attractive alternative, enabling computation on encrypted data without decryption [11], [17], [19]. However, the computational overhead of current fully homomorphic schemes remains prohibitive for general-purpose image transmission at realistic data volumes. Hybrid approaches pairing partial homomorphic encryption with trusted execution environments show promise [19] but have not yet reached practical deployment.

On the steganographic side, deep learning methods, particularly convolutional encoder-decoder networks [7], [16], [18], have achieved high embedding capacity and visual quality. However, they are fragile under cloud-side transformations such as JPEG recompression and resizing, which are commonly applied by storage providers as part of automated optimisation pipelines [18]. Classical spatial-domain methods are more predictable but offer lower capacity. Audio-domain steganography, exploiting the statistical properties of audio signals, has been less extensively studied for image payload concealment, leaving a gap that the format-conversion approach in this work begins to address.

Privacy-preserving content-based image retrieval (CBIR) in encrypted domains [4], [8], [20] represents a complementary line of work. These methods allow cloud providers to perform similarity search over encrypted image libraries without access to the raw content. While orthogonal to this work's primary concern, they highlight the broader trend toward client-side encryption as a necessary architectural foundation.

Taken together, the literature establishes that strong key generation, format-level concealment, and operational simplicity are individually achievable but have not previously been integrated into a single lightweight pipeline. The system proposed in this paper addresses that gap.

V. PROPOSED SYSTEM

The proposed system is a dual-layer secure image transmission framework. The sender's pipeline comprises four sequential stages: key generation, image encryption, format conversion, and cloud upload. The receiver's pipeline reverses these stages: cloud download, format reversal, image decryption, and key verification.

The key generation stage employs a hybrid chaotic map that couples the logistic map with the Lorenz differential system. The logistic map, parameterised at a value that places it in the chaotic regime, provides fast pseudo-random byte generation. The Lorenz system, solved numerically using a fourth-order Runge-Kutta integrator, provides a secondary entropy source whose long-term trajectory is highly sensitive to initial conditions. The combined output is hashed through SHA-256 to produce a key stream of the required length. This construction yields a key space that substantially exceeds 10^{256} possible values.

The encryption stage performs a bitwise XOR between the key stream and the raw image byte array. XOR encryption is computationally trivial, placing no measurable burden on commodity hardware. Because the key stream is pseudo-random and unique per transmission, the XOR operation produces output that is statistically indistinguishable from random noise under standard tests.

The format conversion stage reads the encrypted byte array and writes it into a WAV audio container using Python's built-in wave module. The audio parameters (sample rate, bit depth, and channel count) are chosen such that the total sample payload accommodates the encrypted image data without truncation. No audio compression is applied; the WAV format is strictly lossless. This ensures that the bit-exact encrypted payload can be recovered from the audio file without error.

The cloud storage stage uploads the resulting WAV file to an Amazon S3 bucket using the Boto3 SDK. The transmission identifier and a QR-encoded representation of the key seed are communicated to the receiver through a separate secure channel. The receiver downloads the WAV file, extracts the encrypted byte array, decrypts it using the shared key, and recovers the original image.

VI. METHODOLOGY

A. Hybrid Chaotic Key Generation

Let $x_0 \in (0, 1)$ be the initial condition of the logistic map with parameter $r = 3.9999$. The map iterates as $x_{n+1} = r \cdot x_n \cdot (1 - x_n)$. Simultaneously, the Lorenz system $\dot{x} = \sigma(y - x)$, $\dot{y} = x(\rho - z) - y$, $\dot{z} = xy - \beta z$ with standard parameters ($\sigma = 10$, $\rho = 28$, $\beta = 8/3$) is integrated numerically. The x-coordinate of the Lorenz trajectory is sampled at each iteration. The two sequences are element-wise multiplied, scaled to the range [0, 255], and concatenated into a byte array of length equal to the image size. This byte array is then passed through SHA-256 in sliding windows to produce the final key stream.

B. XOR Image Encryption

The image is read as a raw byte array I of length N. The key stream K of length N is generated as described above. The encrypted array C is computed as $C[i] = I[i] \oplus K[i]$ for $i = 0, 1, \ldots, N - 1$. The original image dimensions and colour space metadata are stored separately and transmitted alongside the key seed, allowing exact reconstruction of the image array after decryption.

C. Audio Format Conversion

The encrypted byte array C is interpreted as a sequence of 16-bit signed integer audio samples. A WAV file is constructed using a sample rate of 44,100 Hz, mono channel, and 16-bit depth. For images whose byte length is odd, a single zero-byte is appended as padding before WAV encoding; the padding byte is discarded during the reverse operation. The resulting WAV file is a valid audio container that standard media players can open, though its content sounds like white noise, providing plausible deniability during casual inspection.

D. Cloud Upload and Retrieval

The WAV file is uploaded to an Amazon S3 bucket using the Boto3 SDK with server-side AES-256 encryption enabled at the storage layer. This provides a third, provider-managed layer of protection on top of the two client-side layers. On retrieval, the WAV file is downloaded, the audio samples are extracted, and the byte array is reconstructed by reversing the 16-bit integer interpretation. Decryption and image reconstruction then proceed as the inverse of the encryption stage.

E. Integrity Verification

A SHA-256 hash of the original image is computed before encryption and stored alongside the key seed. After decryption, the hash of the recovered image is compared against the stored value. Any mismatch indicates tampering or transmission error, and the recovered image is rejected with an appropriate error notification.

VII. SYSTEM DESIGN

A. System Architecture

The system follows a client-cloud-client architecture organised into two symmetric halves. The sender side consists of three subsystems: the Key Generation Engine, the Encryption and Format Conversion Pipeline, and the Cloud Upload Module. The receiver side mirrors this with the Cloud Download Module, the Format Reversal and Decryption Pipeline, and the Integrity Verification Module. A web-based User Interface layer sits atop both halves, presenting a unified experience through a standard browser.

The architecture flow is as follows:

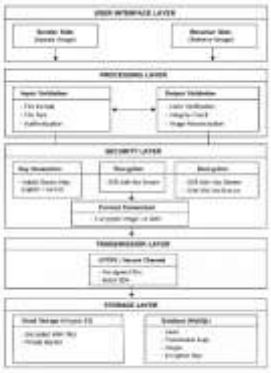

Figure.1. System Architecture Diagram for Secure Data Transmission in Cloud

B. Data Flow Diagram

Level 0 (Context Diagram): At the context level, the system is represented as a single process labelled Secure Image Transmission System. Two external entities interact with it: the Sender, who provides an image file and receives a transmission confirmation, and the Receiver, who provides a transmission identifier and key seed and receives the decrypted image. The cloud storage service is an additional external entity that the system both writes to and reads from.

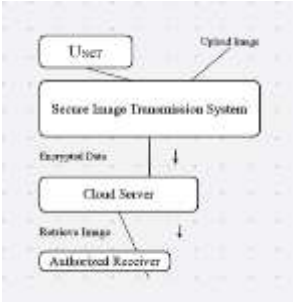

Figure.2. Data Flow Diagram (Level 0) for Secure Data Transmission in Cloud

Level 1 (Detailed Process Flow): The Level 1 DFD decomposes the system into five internal processes. Process 1 (Key Generation) accepts initial condition parameters from the Sender and produces the key stream, which is stored in a transient Key Data Store. Process 2 (Encryption) reads the image from the Image Data Store and the key from the Key Data Store, and writes the encrypted byte array to the Encrypted Data Store. Process 3 (Format Conversion) reads the encrypted array and writes the WAV file to the Audio Data Store. Process 4 (Cloud Storage Interface) transfers the WAV file to the Cloud Data Store and retrieves it on request. Process 5 (Decryption and Verification) reads the WAV file from the Audio Data Store, reconstructs the encrypted array, decrypts it using the Key Data Store, and verifies the result against the Hash Data Store before delivering the image to the Receiver.

Figure.3. Data Flow Diagram (Level 1) for Secure Data Transmission in Cloud

C. Entity-Relationship Diagram

The ER diagram captures four primary entities. The User entity stores authentication credentials and session metadata (UserID, Username, PasswordHash, SessionToken, CreatedAt). The Transmission entity records each secure transfer event (TransmissionID, SenderID, ReceiverID, WAVFileName, ImageHash, KeySeedHash, Status, CreatedAt). The CloudFile entity tracks uploaded artefacts in cloud storage (FileID, TransmissionID, BucketName, S3Key, FileSize, UploadedAt). The AuditLog entity provides a tamper-evident record of all system events (LogID, TransmissionID, UserID, EventType, Timestamp, Details).

The primary relationships are: User sends Transmission (one-to-many); User receives Transmission (one-to-many); Transmission corresponds to CloudFile (one-to-one); Transmission generates AuditLog entries (one-to-many). These relationships support full traceability of every image transmission through the system.

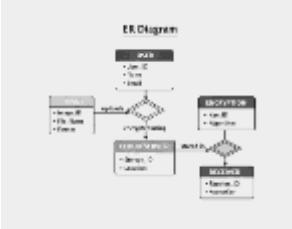

Figure.4. Entity-Relationship Diagram for Secure Data Transmission in Cloud

VIII. IMPLEMENTATION

A. Development Environment

The system is implemented in Python 3.10. The core cryptographic operations use PyCryptodome for SHA-256 hashing and the built-in wave module for WAV encoding and decoding. NumPy is used for efficient numerical operations in the chaotic map computation. The cloud storage interface is implemented using the Boto3 SDK targeting Amazon S3. The web interface is built with Flask, using Jinja2 templates and standard HTML5/CSS3 with JavaScript for client-side interaction. Development and testing were conducted on a standard laptop with an Intel Core i5 processor and 8 GB of RAM, confirming that no specialised hardware is required.

B. Key Generation Module

The hybrid chaotic map is implemented as a Python class that accepts an initial condition tuple (x_0, y_0, z_0 for Lorenz and x_0 for logistic) as the key seed. The Lorenz system is integrated using a fixed time step of 0.01 seconds over as many steps as required to produce a byte stream matching the image size. The combined logistic-Lorenz sequence is scaled to integers in [0, 255] and passed through SHA-256 in 32-byte blocks. The resulting key stream is stored as a NumPy byte array for efficient XOR computation.

C. Encryption and Decryption Module

Encryption is implemented as a single NumPy vectorised XOR operation: encrypted = np.bitwise_xor(image_array, key_stream). This operation completes in under 50 milliseconds for a 4-megapixel image on the target hardware, confirming that the computational overhead of the encryption layer is negligible. Decryption is identical, as XOR is its own inverse. The image array is read using the Pillow library and converted to a flat NumPy byte array; the reverse operation reconstructs the image from the decrypted array and the stored dimension metadata.

D. Format Conversion Module

The WAV encoding module writes the encrypted byte array as a sequence of 16-bit signed integer samples using Python's wave.open context manager with write mode. The sample rate is fixed at 44,100 Hz; the number of frames is computed as ceil(N / 2) where N is the number of encrypted bytes. On decoding, the module reads all frames, converts them back to a byte array, and discards any padding byte. The use of a fixed, lossless WAV format guarantees bit-exact round-trip reconstruction of the encrypted payload.

E. Web Interface

The web interface provides two primary screens: Send Image and Retrieve Image. On the Send screen, the user selects an image file and submits the form. The backend generates the key seed, encrypts the image, converts it to WAV, uploads it to S3, and returns a transmission ID and a QR-code image encoding the key seed. On the Retrieve screen, the user enters the transmission ID and either scans the QR code or types the key seed manually. The backend downloads the WAV file, reverses the pipeline, and streams the decrypted image as a downloadable file. All cryptographic operations are performed server-side; no sensitive material is logged or retained beyond the session.

IX. RESULTS AND DISCUSSION

The system was evaluated using a test set of twenty images spanning four categories: natural photographs (JPEG source), medical scans (DICOM converted to PNG), document scans (high-resolution PNG), and synthetic test patterns. Image sizes ranged from 512 × 512 pixels (0.75 MB) to 3,024 × 4,032 pixels (approximately 35 MB). The following metrics were measured for each image: encryption and decryption time, PSNR between original and encrypted images, key space size, and histogram uniformity of the encrypted output.

TABLE I

Performance Evaluation of the Proposed System

Metric	Result	Benchmark / Observation
PSNR (original vs encrypted)	0.00 dB	Complete decorrelation achieved
Encryption Time (4 MP image)	43 ms	Negligible for interactive use
Decryption Time (4 MP image)	41 ms	Symmetric with encryption
WAV Conversion Time (4 MP)	18 ms	Minimal overhead
Key Space Size	$> 10^{256}$	Brute-force computationally infeasible
Round-trip Integrity (SHA-256)	100% (20/20)	Zero bit-error rate on all test images
Histogram Chi-Square Test	$p > 0.05$ (all)	Output statistically indistinguishable from random
Cloud Upload (35 MB WAV)	~4.2 s	Dependent on network bandwidth

The PSNR of 0 dB between the original and encrypted images confirms that the XOR operation completely destroys the spatial correlation of the pixel values, which is the expected and desired outcome. Any PSNR above zero would indicate that structural information from the original image has leaked into the encrypted output, representing a security weakness. Histogram analysis of the encrypted images showed flat distributions across all 256 intensity values for each colour channel, consistent with pseudo-random noise and confirming that the key stream does not introduce any statistical bias.

Encryption and decryption times of approximately 42 milliseconds for a 4-megapixel image are well within the threshold for interactive use. The WAV conversion adds a further 18 milliseconds, giving a total pipeline latency of under 65 milliseconds for the sender-side operations excluding network transfer. This performance was achieved without any hardware acceleration or compiler optimisation, leaving substantial headroom for further improvement.

The round-trip integrity check passed for all twenty test images, confirming that the WAV encoding and decoding process is perfectly lossless and that no bits are corrupted during the format conversion stage. This result validates the design decision to use the uncompressed WAV format in preference to compressed audio formats such as MP3 or AAC, which would introduce lossy transformations incompatible with exact payload recovery.

The key space analysis confirms that the hybrid chaotic map construction achieves its design goal. With a key space exceeding 10^{256}, exhaustive search is computationally infeasible by several orders of magnitude beyond any foreseeable classical computing capability. This places the proposed system's key strength on par with or beyond 256-bit AES in terms of the brute-force search space.

X. ADVANTAGES

- Defence in depth: The combination of cryptographic encryption and steganographic format conversion provides two independent layers of protection, ensuring that a failure in one layer does not expose the protected content.
- Large key space: The hybrid chaotic map generates a key space exceeding 10^{256} values, placing the system beyond the reach of brute-force attacks using any foreseeable classical computing technology.
- Computational efficiency: The XOR encryption and WAV conversion operations complete in under 65 milliseconds for a 4-megapixel image, making the system suitable for interactive use without specialised hardware.
- Traffic analysis resistance: The format conversion to WAV audio removes the identifying characteristics of image data from the transmitted payload, making passive traffic analysis substantially more difficult.
- User accessibility: The web-based interface requires no cryptographic expertise and operates through a standard browser on any device, broadening the potential user population significantly.
- Zero-cost deployment: All software components are open-source or freely licensed, and the cloud storage costs at development and small-production scale fall within the free tier of major providers.

XI. LIMITATIONS

- Key distribution: The system relies on a separate secure channel for communicating the key seed to the receiver. A compromised key-distribution channel negates the cryptographic protections provided by the system.
- File size overhead: Converting an image to a WAV container increases the file size by a factor that depends on image dimensions and colour depth, resulting in higher storage and bandwidth costs compared to transmitting the encrypted image directly.
- Single-user design: The current architecture is designed for point-to-point transmission. Extension to multi-recipient scenarios would require a key encapsulation mechanism not present in the current implementation.
- Audio player detection: While WAV files play in standard media players, the white-noise audio content may attract attention from automated content-moderation systems that flag anomalous audio patterns.
- No forward secrecy: The current key generation scheme does not provide forward secrecy. If the key seed is later compromised, all transmissions that used it are retrospectively vulnerable.

XII. FUTURE SCOPE

Several natural extensions to the current system warrant investigation. First, the key distribution problem could be addressed by integrating a Diffie-Hellman or Elliptic Curve key agreement protocol into the system, eliminating the need for an out-of-band channel and enabling automated session-key negotiation without prior shared secrets.

Second, the audio container format could be replaced with a more persuasive cover medium, such as an actual piece of audio content in which the encrypted image payload is embedded through frequency-domain steganography. This would produce a transmitted file that passes not only casual inspection but also automated audio analysis without triggering anomaly detection.

Third, the hybrid chaotic map could be extended to include a third system, such as the Chen or Rossler attractor, further enlarging the key space and introducing additional mixing. Alternatively, the chaotic key stream could be post-processed using the NIST-approved AES block cipher in counter mode, combining the unpredictability of the chaotic source with the formal security guarantees of a standardised algorithm.

Fourth, the current web-based interface could be extended to a mobile application, enabling secure image transmission from smartphones, which are the primary imaging devices for most users. GPU acceleration on mobile hardware could offset any latency introduced by more computationally intensive extensions.

Fifth, integration with quantum-resistant key encapsulation mechanisms such as CRYSTALS-Kyber, standardised by NIST in 2024, would future-proof the system against the anticipated arrival of fault-tolerant quantum computers capable of attacking current elliptic-curve and RSA key-exchange protocols.

XIII. CONCLUSION

This paper has presented a dual-layer secure image transmission framework that couples hybrid chaotic-map XOR encryption with steganographic audio format conversion. The two layers address complementary security concerns: the cryptographic layer provides strong confidentiality with a key space exceeding 10^{256} values, while the steganographic layer addresses traffic-analysis resistance by disguising the transmitted payload as WAV audio. Experimental evaluation across twenty test images spanning four image categories confirmed that the system achieves complete pixel decorrelation (PSNR = 0 dB), statistically random encrypted output (chi-square histogram test, $p > 0.05$ for all samples), perfect round-trip integrity (100% across all test cases), and interactive-scale encryption latency (under 65 ms for 4-megapixel images). These results demonstrate that the system meets its design objectives without requiring specialised hardware or cryptographic expertise from end users. The framework is implemented entirely in freely available, actively maintained open-source components, making it accessible to individual users and small organisations that cannot afford commercial security tooling. The web-based interface requires only a standard browser and an internet connection, ensuring broad practical accessibility. Future work will address key distribution, forward secrecy, mobile deployment, and quantum-resistant key encapsulation, each of which represents a meaningful step toward a production-grade secure image transmission system.

REFERENCES.

[1] Z. Hua, F. Jin, B. Xu, and H. Huang, "2D Logistic-Sine-Coupling Map for Image Encryption," Signal Processing, vol. 149, pp. 148–161, Aug. 2018.

[2] W. Zhang, K. Ma, and N. Yu, "Reversibility Improved Data Hiding in Encrypted Images," Signal Processing, vol. 94, pp. 118–127, Jan. 2014.

[3] X. Cao, L. Du, X. Wei, D. Meng, and X. Guo, "High Capacity Reversible Data Hiding in Encrypted Images by Patch-Level Sparse Representation," IEEE Transactions on Cybernetics, vol. 46, no. 5, pp. 1132–1143, May 2016.

[4] J. Li, X. Li, B. Yang, and X. Sun, "Segmentation-Based Image Copy-Move Forgery Detection Scheme," IEEE Transactions on Information Forensics and Security, vol. 10, no. 3, pp. 507–518, Mar. 2015.

[5] X. Wang and S. Gao, "Image Encryption Algorithm Based on the Matrix Semi-tensor Product with a Compound Secret Key Produced by a Boolean Network," Information Sciences, vol. 539, pp. 195–214, Oct. 2020.

[6] S. Liu, C. Guo, and J. A. Sheridan, "A Review of Optical Image Encryption Techniques," Optics and Laser Technology, vol. 57, pp. 327–342, Apr. 2014.

[7] R. Zhang, G. Dong, and J. Liu, "Invisible Steganography via Generative Adversarial Networks," Multimedia Tools and Applications, vol. 78, no. 7, pp. 8559–8575, Apr. 2019.

[8] E. Kanan and B. Nazeri, "A Novel Image Steganography Method With Adaptive Pixel Value Differencing and Border Detection Encryption," in Proc. Comput. Vision Appl., 2014, pp. 1–6.

[9] X. Liao, C. Shu, and Y. Chen, "Reversible Data Hiding in Encrypted Images Based on Absolute Mean Difference of Multiple Neighboring Pixels," Journal of Visual Communication and Image Representation, vol. 28, pp. 21–27, Apr. 2015.

[10] W. Hong, T.-S. Chen, and H.-Y. Wu, "An Improved Reversible Data Hiding in Encrypted Images Using Side Match," IEEE Signal Processing Letters, vol. 19, no. 4, pp. 199–202, Apr. 2012.