Chaos-Based Medical Image Encryption with Key Establishment Technique Using ChaCha20 and Diffie-Hellman Protocols

Sylia Amarouche*, Said Talbi and Mehammed Daoui
Laboratory for Computer Science Research (LARI), Department of Computer Science, Mouloud Mammeri University of Tizi-Ouzou, Tizi-Ouzou 15000, Algeria
E-mail: sylia.amarouche@ummto.dz, said.talbi@ummto.dz, mehammed.daoui@ummto.dz
*Corresponding author

Keywords: Secure key establishment technique, chacha20, diffie-hellman, chaos systems, chaotic maps

Received: March 29, 2025

In healthcare, medical images are often transmitted over unsecured networks for consultation and diagnosis, exposing them to significant security risks. Therefore, it is crucial to implement effective encryption and key establishment schemes to preserve the integrity of this sensitive data. A critical issue is the repeated use or exposure of encryption keys, which increases the risk of compromise or interception. This paper proposes a novel key establishment technique for chaos-based encryption systems. This technique integrates the Diffie-Hellman protocol for asymmetric key agreement and the ChaCha20 stream cipher for secure stream encryption. These components are combined within a cryptosystem based on 2D Zaslavsky and Henon chaotic maps. The proposed technique preserves the integrity of medical image transmission without explicit key exchange and eliminates synchronization-induced errors introduced by conventional chaos-based systems. Experiments are conducted on 100 *grayscale and RGB medical images of* 256×256 *and* 610×610 *resolutions. The evaluation is carried out using two* $64-bit$ *quad-core ARM Cortex-A53* $1.4\,GHz$ *Raspberry Pi* 3 *Model B+ devices to simulate a low-power IoT environment. The cryptosystem achieved a Peak Signal-to-Noise Ratio (PSNR)* $= 100\,dB$*, Structural SIMilarity (SSIM)* $= 1$*, and Mean Squared Error (MSE)* $= 0$*, confirming perfect reconstruction quality. Additionally, the system demonstrated strong security metrics, with entropy* > 7.99*, Number of Pixels Change Rate (NPCR)* $\geq 99.62\%$*, Unified Average Changing Intensity (UACI)* $\geq 33.50\%$*, and a key space* $= 2^{628}$*. These results confirm the proposed technique's robustness, efficiency, and suitability for secure image transmission in resource-constrained IoT applications.*

Povzetek: Predlagana je nova tehnika vzpostavljanja ključev za kaotično šifriranje medicinskih slik, ki združuje Diffie-Hellman, ChaCha20 ter Zaslavsky in Henon zemljevide s ciljem izboljšane varnosti in učinkovitosti.

1 Introduction

Digital images play a vital role in medicine, serving various purposes such as diagnosis, treatment planning, and patient health monitoring [21, 36]. Transmitting these images over networks for remote consultation and analysis is common practice. However, sending such sensitive information over an unsecured channel poses a significant security threat [23]. Researchers have investigated various chaos-based cryptosystems for image encryption to address this issue, aiming to provide robust security measures [3, 16, 18, 27, 31].

Chaos systems are widely studied due to their good properties of speed, high security, and low computational overhead [26]. These systems are a sub-category of nonlinear dynamical systems. They exhibit several key properties, including sensitive dependence on initial conditions, pseudo-random properties, and non-periodicity [38]. Despite their deterministic simplicity, these systems can exhibit unpredictable and widely divergent behavior over time [38]. The fundamental elements of these cryptosystems are the chaotic maps used to implement the principles of confusion and diffusion between image pixels [38]. Diffusion is typically accomplished through a substitution process, where substitution chaotic maps are employed to modify the pixel values. Conversely, confusion is achieved by permuting pixels using permutation chaotic maps, without altering pixel values [12, 37].

Many chaotic maps used in chaos-based cryptosystems for image encryption rely on synchronization techniques to reconstruct the same encryption and decryption key [3, 16, 18, 31]. These chaotic maps are inspired by the pioneering work of Pecora and Carroll [30]. Their study introduced a technique to coordinate two identical chaotic systems with different initial conditions. The synchronization concept involves using the output of a master system to control a slave system. As a result, the output of the slave system asymptotically follows that of the master system [29]. Most of these cryptosystems have been simulated

using MATLAB or Python.

In healthcare, each detail of patient information is significant and should be kept complete and confidential. Indeed, when physicians treat the received images, they may fail to detect potential abnormalities due to data loss. The synchronization technique within chaotic systems is known to be susceptible to data loss. This vulnerability raises major concerns when establishing a common encryption and decryption key process for two communicating entities. This is especially crucial when communication occurs between two separate devices. It highlights the importance of maintaining the integrity of the transmitted data. Additional factors, such as communication channel noise and transmission delays, further complicate the process and may compromise the decrypted image quality.

Notably, existing research in this area has not sufficiently addressed the issue of synchronization-induced losses in image quality. These studies have overlooked that the synchronization process contributes to these losses, primarily due to the sensitivity of chaotic maps to initial conditions. Consequently, during the key synchronization process, the encryption and decryption systems may operate with different initial conditions. This inconsistency can lead to the generation of mismatched encryption and decryption keys. As a result, the decrypted image may differ from the original, which is unacceptable in the medical field. Furthermore, generating a unique key for each pixel can enhance the cryptosystem robustness, as larger images lead to a larger key space. However, transmitting a key that matches the image size is inefficient in terms of time and bandwidth. Therefore, the challenge lies in generating identical keys in the encryption and decryption processes without explicit transmission. Moreover, failing to periodically update the key introduces significant vulnerabilities to the system's security, particularly the risk of long-term key exposure. If a key remains static for an extended period, it increases the likelihood of being compromised by unauthorized parties.

To address these limitations, this paper introduces a novel key establishment technique for encryption and decryption processes. This technique eliminates synchronization-induced data losses while preserving image quality. Moreover, it ensures the secure generation of identical keys for both processes by transmitting essential information in a highly secure manner. The encryption key is dynamically updated with each transmission, significantly reducing the risk of compromise by unauthorized entities. The technique leverages a hybrid approach. It combines the Diffie-Hellman protocol for a robust and secure key agreement with the ChaCha20 stream cipher, an advanced encryption algorithm. This combination enables the secure transmission of the secret information required for key generation. The Diffie-Hellman key and a nonce are used as inputs to the ChaCha20 cipher, which encrypts the necessary information. The results are then employed by the receiver to reconstruct the same decryption keys as those used for encryption. This ensures secure communication without direct key transmission. In addition to this combination, the proposed technique employs the Zaslavsky and Henon chaotic maps. These maps are used to generate keys that encrypt the Blake hash digest. This digest represents one of the secret components used in constructing the encryption and decryption keys of medical images.

Overall, the proposed technique provides an innovative and efficient solution to enhance the security and integrity of medical image transmission in healthcare applications.

The remainder of this paper is structured as follows. Section 2 provides a brief overview of existing key exchange works, including symmetric, asymmetric, and chaos-based techniques. Section 3 outlines the principle of Diffie-Hellman for secure key agreement and introduces the ChaCha20 stream cipher. It also describes the 2D Zaslavsky and Henon chaotic maps used in the proposed cryptosystem. Section 4 presents a detailed aspect of the production and combination of the proposed key establishment technique. Section 5 discusses the performance metrics and compares the experimental results with those of related works in the literature. Section 6 concludes this work.

2 Related work

In recent years, significant advancements have been made in the area of secure key exchange and establishment within the Internet of Things (IoT) domain. Various cryptographic approaches have been explored, including symmetric, asymmetric, and chaos-based techniques. However, these methods present shortcomings that reduce their effectiveness in practical IoT applications.

Several research have been conducted in cryptography based on chaotic systems. This interest is due to their inherent randomness, sensitivity to initial conditions, and suitability for lightweight implementations [11, 15, 18, 31, 32]. Nevertheless, in terms of key exchange efficiency, these systems face a common data loss issue during the decryption process. This loss is primarily caused by the synchronization technique, where communicating entities must align precisely on control parameters and initial conditions. Even a slight change can lead to data loss for the receiver, potentially preventing accurate image reconstruction. This issue undermines the reliability of chaotic systems in real-time IoT environments, where maintaining data integrity is essential.

In contrast, traditional cryptographic methods offer more robust solutions for key establishment. Indeed, both symmetric and asymmetric techniques help overcome synchronization-induced limitations in chaos-based systems.

Adeshina et al. [2] propose a Least Significant Bit (LSB) steganography technique to embed clinical data within medical images. The approach indicates high image quality regarding PSNR and SSIM test results with PSNR $> 30\ dB$ and SSIM ≈ 1. While the method effectively hides data with small degradation, it remains vulnerable to image processing operations such as compression or filtering, which

may distort or destroy the embedded data.

Almola et al. [4] propose a secure encryption key generation approach. This approach uses biometric fingerprint data integrated with Convolutional Neural Networks (CNNs) and Particle Swarm Optimization (PSO). The system eliminates key transmission risks by enabling local key generation, which enhances security for applications such as IoT healthcare and cloud storage. The method achieves a correlation coefficient = 0.00628 and zero repetition. However, the reported entropy value of 7.89 remains below the ideal randomness expected for encrypted images. Moreover, the effectiveness of the proposed system relies heavily on the quality and uniqueness of the fingerprints. Any low contrast issues with fingerprint quality during scanning will affect the accuracy of key generation.

Hsieh et al. [20] propose a secure and efficient chaotic-based user-authentication and key-agreement scheme for healthcare centers. The scheme ensures secure communication channels with mutual authentication and session-key negotiation. These mechanisms comply with the privacy and security regulations of the Health Insurance Portability and Accountability Act (HIPAA). The proposed scheme uses the chaotic map and Diffie-Hellman key exchange technique to negotiate the session keys between parties. It ensures the protection of transmitted Patient Health Information (PHI). This approach proves that combining the Diffie-Hellman protocol with a chaotic map for key exchange enhances the efficiency and security of user authentication and key agreement schemes. However, despite offering enhanced security features, the proposed scheme ensures an average execution time of 0.4839 seconds. Such latency is unsuitable for real-time applications, particularly in the medical domain.

Mir et al. [25] propose a new real-time image encryption scheme that involves a modified Vigenère cipher and chaotic maps, This scheme ensures confusion and diffusion phases. It uses the Diffie-Hellman protocol to encrypt the keys that can be openly shared with the receiver. Although Diffie-Hellman provides a secure way to exchange keys, it lacks confidentiality and integrity protection. As a result, the transmitted data remains vulnerable to eavesdropping and tampering. Another notable limitation of this approach is the inherent inefficiency of the Vigenère cipher. It can introduce computational delays, especially when handling large image datasets in real-time applications. Furthermore, the encryption time remains inefficient for real-time applications, especially in time-sensitive medical scenarios where rapid image processing is crucial. In addition, the approach shows a low SSIM value (0.0018) reflecting poor structural preservation of the encrypted images. Consequently, the visual quality is significantly degraded. Such degradation compromises the usability of the images in medical applications.

Msolli et al. [27] propose a novel key management scheme using a pool-hash mechanism for Wireless Sensor Networks (WSN) and the IoT. It aims to ensure energy-efficient network connectivity, resilience against node capture, message protection, and node authentication. The approach employs probabilistic key pre-distribution and Shift for secure key distribution and session key generation. However, the method relies heavily on the robustness of the hashing function, making it vulnerable to weaknesses in the hash algorithm. Additionally, its reliance on the pool-hash function introduces computational overhead for resource-constrained devices in WSN and IoT applications.

Seyhan et al. [33] introduce a novel Bilateral Generalization of the Inhomogeneous Short Integer Solution Key Exchange (Bi-GISIS KE) protocol. This quantum-resistant protocol is designed for resource-constrained IoT devices. The solution enables the usage of the same key in various sessions, enhancing efficiency for Device-to-Device (D2D) communication. It also reduces time consumption and improves key management. The authors emphasize the importance of this development in light of the vulnerabilities that quantum computing poses to traditional key exchange methods. However, although using reusable keys improves efficiency, it introduces significant security risks if not properly managed or protected. If an attacker compromises these keys, they could potentially gain access to multiple communication sessions and expose session key security.

Shahid et al. [34] propose a Blockchain-Driven Chaotic Tent Map Encryption Scheme (BCTMES). The scheme employs sine map permutation, tent map-based diffusion, and prime circulant matrix substitution. The key establishment relies on three secret keys and $SHA-256$ hashing, with verification handled with the blockchain. The scheme offers strong security (key space = 10^{72}, entropy = 7.98, NPCR = 99.62%, UACI = 33.46%) and prevents tampering through integrity checks. However, it depends on classical chaotic maps, namely Tent and Sine maps. These maps exhibit limited dynamical complexity, low Lyapunov exponents, and limited chaotic range, which affect long-term robustness. Furthermore, the reported encryption time for a 256×256 grayscale image was $43ms$ on a high-performance desktop (Intel $i7$). However, this evaluation may not accurately reflect the efficiency of the approach when deployed on resource-constrained, low-power embedded systems. In addition, the integration of blockchain technology may introduce latency issues, making it less suitable for real-time applications.

Sureshkumar et al. [35] introduce a robust mutual authentication protocol based on Elliptic Curve Cryptography (ECC) for cloud-enabled smart Vehicle-to-Grid (V2G) networks. This protocol includes an enhanced key establishment. The authors combine symmetric and asymmetric cryptographic techniques to facilitate secure key exchange. This combination ensures the confidentiality, integrity, and authenticity of communications while minimizing the risk of key compromise. The evaluations indicate that the protocol provides improved computational efficiency and reduced energy consumption. However, it introduces latency and data privacy concerns, especially when handling sensitive information in real-time environments.

Table 1 provides a concise summary of the aforemen-

Table 1: Comparative analysis of cryptosystems proposing key exchange methods

Work	Approach	Objective	Strength	Limitation
[2]	LSB-based steganography for medical image security	Secure clinical data embedding with high visual quality	High imperceptibility (PSNR > 30 *dB*, SSIM ≈ 1)	Sensitive to image processing attacks (e.g. compression, filtering), low robustness
[4]	Biometric key generation using fingerprints based on CNNs, and PSO	Propose secure local key generation, avoid transmission vulnerabilities	Eliminates key transmission, low correlation (0.00628), zero repetition	Entropy below ideal (7.89), key generation depends on fingerprint quality
[20]	Chaotic-map-based user-authentication and key-agreement scheme for healthcare centers	Ensure secure communication with mutual authentication and session-key negotiation compliant with HIPAA	Combines chaotic-map-based Diffie–Hellman key exchange for secure session key negotiation	Diffie-Hellman alone does not provide complete confidentiality and integrity, substantial key agreement delay
[25]	Real-time image encryption using modified Vigenère cipher and chaotic maps	Ensure end-to-end encryption of image data during transmission	Diffie-Hellman algorithm provides secure key exchange, chaotic maps enhance encryption robustness	Diffie-Hellman does not provide confidentiality or integrity, Vigenère cipher inefficiency can cause delays, especially with large image datasets, substantial encryption delay, very low SSIM (0.0018)
[27]	Key management scheme using pool-hash for WSN and IoT	Energy-efficient network connectivity, node authentication, and message protection	Resilient against node capture, uses probabilistic key pre-distribution techniques	Vulnerabilities in the hashing function could compromise security, and computational overhead for resource-constrained devices due to the pool-hash function
[33]	Bi-GISIS KE quantum-resistant key exchange protocol using reusable keys	Enhance efficiency during key generation processes in resource-constrained IoT devices	Efficient for D2D communication reduces time consumption, improves key management, ensures quantum resistance with lattice-based cryptography	Increases computational complexity due to matrix multiplications, risks posed by reusable keys if not properly managed, compromised keys expose multiple sessions
[34]	Blockchain-based encryption using tent and sine chaotic maps, and circulant matrix substitution	Secure medical image encryption and integrity verification in cloud environments	Strong security (key space = 10^{72}, entropy = 7.98, NPCR = 99.62%, UACI = 33.46%), blockchain for tamper-proof integrity	Limited chaotic dynamics (low Lyapunov exponent), high implementation complexity due to blockchain, potential latency, evaluated only on high-end PC
[35]	Authentication protocol using ECC for cloud-enabled smart V2G networks	Improve computational efficiency and reduce energy consumption in V2G networks	Enhanced computational efficiency, reduced energy consumption	Introduces latency and data privacy concerns when handling sensitive information in real-time

tioned works. It highlights the key exchange and establishment approaches based on symmetric and asymmetric techniques, their objectives, strengths, and limitations.

The limitations of chaotic systems, as well as symmetric and asymmetric techniques, underscore the need for a hybrid approach. Such an approach should combine the strengths of each method while addressing their respective weaknesses. Therefore, this paper proposes a novel key establishment technique. It combines the robustness of the asymmetric Diffie-Hellman method, the efficiency of the symmetric ChaCha20 algorithm, and the unpredictability of the 2D Zaslavsky and Henon chaotic maps. This combination enhances the process in terms of speed, robustness, and security. The proposed technique periodically updates the keys without transmitting them, thereby minimizing the risk of key exposure and associated vulnerabilities. Moreover, it encrypts the Blake hash digest used to generate the encryption and decryption keys, further reinforcing the cryptosystem security. By addressing the shortcomings of existing techniques, this technique ensures rapid and secure key establishment while preserving data integrity. Notably, no existing solution in the literature combines Diffie-Hellman and ChaCha20 with a chaos-based system for secure key establishment, marking a significant contribution to the field.

3 Preliminaries

This section introduces the Diffie-Hellman protocol, a fundamental technique for secure key establishment, along with the ChaCha20 stream cipher, a modern and efficient encryption algorithm. Their integration facilitates the secure transmission of control parameters and initial conditions for the 2D-Zaslavsky map and the control parameters of the

2D-Henon chaotic maps. This allows the receiver to independently regenerate the decryption keys without the need for direct key exchange. Additionally, this section details the mathematical formulations of the 2D-Zaslavsky and 2D-Henon chaotic maps. The combined outputs of these maps are employed to encrypt the Blake hash digest. This digest serves as a crucial element in generating the encryption and decryption keys required for secure medical image transmission.

3.1 Diffie-Hellman protocol

The Diffie-Hellman key exchange protocol [10] enables two parties to generate and exchange public keys and compute a shared secret. This process ensures secure key generation without directly transmitting it. Based on the discrete logarithm problem in a finite cyclic group, it provides a robust method for secure key establishment over insecure communication channels [7, 10]. This protocol has become a cornerstone of modern cryptography, widely used in secure communication standards. The process involves two parties exchanging public keys and computing a shared secret, ensuring secure key generation without exposing the key itself. The Diffie-Hellman protocol process can be summarized as follows:

1. Setup phase, both communicating entities agree on a prime number p, and a primitive root g, where $g \in p$. These parameters are publicly known.

2. Key generation phase, each entity privately selects a random secret number a and b for the sender and receiver, respectively, known as a private key. After that, each entity computes a public key A and B, respectively for the sender and the receiver, as follows:

$$A = g^a \; mod \; p$$

$$B = g^b \; mod \; p$$

   Once computed, the public keys are exchanged openly.

3. Shared secret key computation phase, after receiving the public key from both entities, they can independently compute the shared secret key. The sender takes the public key B received from the receiver and computes $(B^a) \; mod \; p$. After that, the receiver performs the same operation, taking the public key A received from the sender and computes $(A^b) \; mod \; p$. Due to the properties of modular exponentiation, both entities result in the same shared secret key.

3.2 ChaCha20 algorithm

ChaCha20 is a symmetric stream cipher widely used for data encryption. Designed as an improved variant of Salsa20 [8], its efficiency and security make it a popular choice for modern encryption tasks. ChaCha20 operates through 20 rounds of rotation, modular addition, and XOR operations to generate an encryption sequence [1, 9, 24]. The process involves three main steps:

1. Input parameters, ChaCha20 requires a $256-bit$ secret key, a $96-bit$ nonce, a $32-bit$ counter, and a $128-bit$ constant ("expand $32-byte$ k") as inputs.

2. Encryption sequence generation, these parameters are organized into a 4×4 matrix, as seen in Table 2. The matrix undergoes 20 transformation rounds, each round applies on the block 4 quarter rounds [9] (alternating between column and row operations). The final matrix is XORed with the initial matrix to produce the encryption key.

Table 2: Initial parameters of the Chacha20 stream cipher

Constant 1	Constant 2	Constant 3	Constant 4
Key 1	Key 2	Key 3	Key 4
Key 5	Key 6	Key 7	Key 8
Counter	Nonce 1	Nonce 2	Nonce 3

3. Data encryption, plaintext data is divided into blocks, and the ChaCha20 function encrypts each block using the generated key block regarding the current counter. The encrypted output is obtained by concatenating all encrypted blocks.

3.3 Zaslavsky chaotic map

The 2D-Zaslavsky chaotic map [39] generates two chaotic orbits, used in different applications including image encryption due to its parameter sensitivity, providing pseudo-randomness [18]. The initial seed point (X_i, Y_i) produces a random sequence (X_{i+1}, Y_{i+1}), which is iterated n times with $i = 0 \; .. \; n-1$ to obtain the sequence $((X_1, Y_1), (X_2, Y_2), .., (X_n, Y_n))$ as seen in Equation 1.

$$\begin{cases} X_{i+1} = (X_i + \nu \times (1 + \mu \times Y_i) + \varepsilon \times \nu \times \mu \times \\ \qquad \cos(2 \times \pi \times X_i)) \; mod \; 1 \\ \\ Y_{i+1} = e^{-\tau} \times (Y_i + \varepsilon \times \cos(2 \times \pi \times X_i)) \end{cases} \tag{1}$$

where

- μ, ε, ν and τ are control parameters of 2D-Zaslavsky map.
- e is the exponential function.
- $mod \; 1$ is used to wrap values into the $[0, 1]$ interval, ensuring periodicity and keeping the system bounded.
- $\mu = \frac{1-e^{-\tau}}{\tau}$
- X_0 and Y_0 are initial conditions of 2D-Zaslavsky map

This map is chaotic for values: $\varepsilon = 0.3$, $\nu = \frac{400}{3}$, $\tau = 3.0$, $X_0 = 0.3$ and $Y_0 = 0.3$

3.4 Henon chaotic map

The 2D-Henon chaotic map [19] is characterized by control parameters $a = 1.4$ and $b = 0.3$ to ensure diffusion (see Equation 2). These classical values induce chaos on the map. However, altering a and b may result in chaotic, intermittent, or periodic behaviors.

$$\begin{cases} X'_{i+1} = Y'_i + 1 - a \times (X'_i)^2 \\ Y'_{i+1} = b \times X'_i \end{cases} \tag{2}$$

where

- X'_0 and Y'_0 are initial conditions of the 2D-Henon map.
- (X'_i, Y'_i) is the initial seed point of the 2D-Henon map.
- X'_{i+1} and Y'_{i+1} are the resulted random sequences with $i = 0 \,..\, n-1$.

This map is chaotic for values: $a = 1.4$, $b = 0.3$, $X'_0 = 0.1$ and $Y'_0 = 0.1$

Note

The two chaotic maps were selected due to their strong dynamic properties. The 2D-Zaslavsky map exhibits robust two-dimensional chaos, characterized by high sensitivity to initial conditions and a large Lyapunov exponent [28]. Likewise, the 2D-Henon map demonstrates excellent chaotic behavior, including a high chaos index and strong ergodicity [19].

4 Contributions

As mentioned in the introduction section, the synchronization technique within chaotic systems, although helpful, presents a significant challenge in establishing a reliable encryption and decryption key process between two communicating entities. This often leads to data loss, especially when the systems operate on different devices. This arises from additional disorders and errors in the synchronization technique. These problems highlight the critical need to preserve the quality and integrity of the exchanged data to ensure both accuracy and security.

To address the aforementioned limitations of key exchange mechanisms based on chaotic synchronization techniques, this work proposes a novel approach. The objective is to ensure a robust and secure encryption process for medical images. To do this, the proposed system aims to achieve the following research objectives:

1. **Prevent the key reuse** by dynamically generating and periodically updating encryption keys without explicitly transmitting the full key, reducing the risk of key exposure.
2. **Enhance encryption robustness and security** by introducing a hybrid key establishment technique that combines:
   (a) The Diffie-Hellman protocol for secure asymmetric key agreement within ChaCha20 algorithm.
   (b) The ChaCha20 stream cipher algorithm for fast and efficient symmetric encryption of secret parameters.
   (c) The 2D Zaslavsky and Henon chaotic maps for key generation for medical image and Blake digest encryption, increasing randomness and strengthening the cryptographic process.
3. **Preserve image quality during encryption and decryption** by eliminating synchronization-induced errors and maintaining the whole integrity between original and decrypted images.
4. **Robust against common attacks** (e.g., brute-force, statistical, differential attacks) by validating the system through:
   (a) High PSNR, SSIM, and MSE values for decrypted images (PSNR = $100dB$, SSIM = 1, MSE = 0).
   (b) Large key space (key space = 1.16×10^{189})
   (c) Strong key sensitivity ensuring high sensitivity to input changes and effective confusion and diffusion (NPCR > 99%, UACI > 33%).
   (d) High entropy indicating strong randomness and resistance to statistical analysis (entropy > 7.99 for encrypted images).
   (e) Efficient execution time demonstrates that the key establishment technique is suitable for resource-constrained environments.

The proposed key establishment technique is illustrated in Figure 2. The proposed technique involves using the 256-bit key generated from the Diffie-Hellman protocol as input for the ChaCha20 stream cipher algorithm. It also takes a $96 - bit$ nonce and a $32 - bit$ counter as inputs. The resulting $256 - bit$ encryption key is then used to encrypt the control parameters and initial conditions required by the chaotic maps. In parallel, the Blake2b [6] hash digest of the plain image is computed and encrypted using a key derived from the Zaslavsky and Henon maps. This encrypted digest and the secret information are transmitted to ensure that the receiver can generate the same key used in the encryption process to successfully decrypt the image.

The following subsections describe the proposed cryptosystem, including the encryption key generation for medical images, the key establishment technique, and the encryption and decryption processes (see Figure 1).

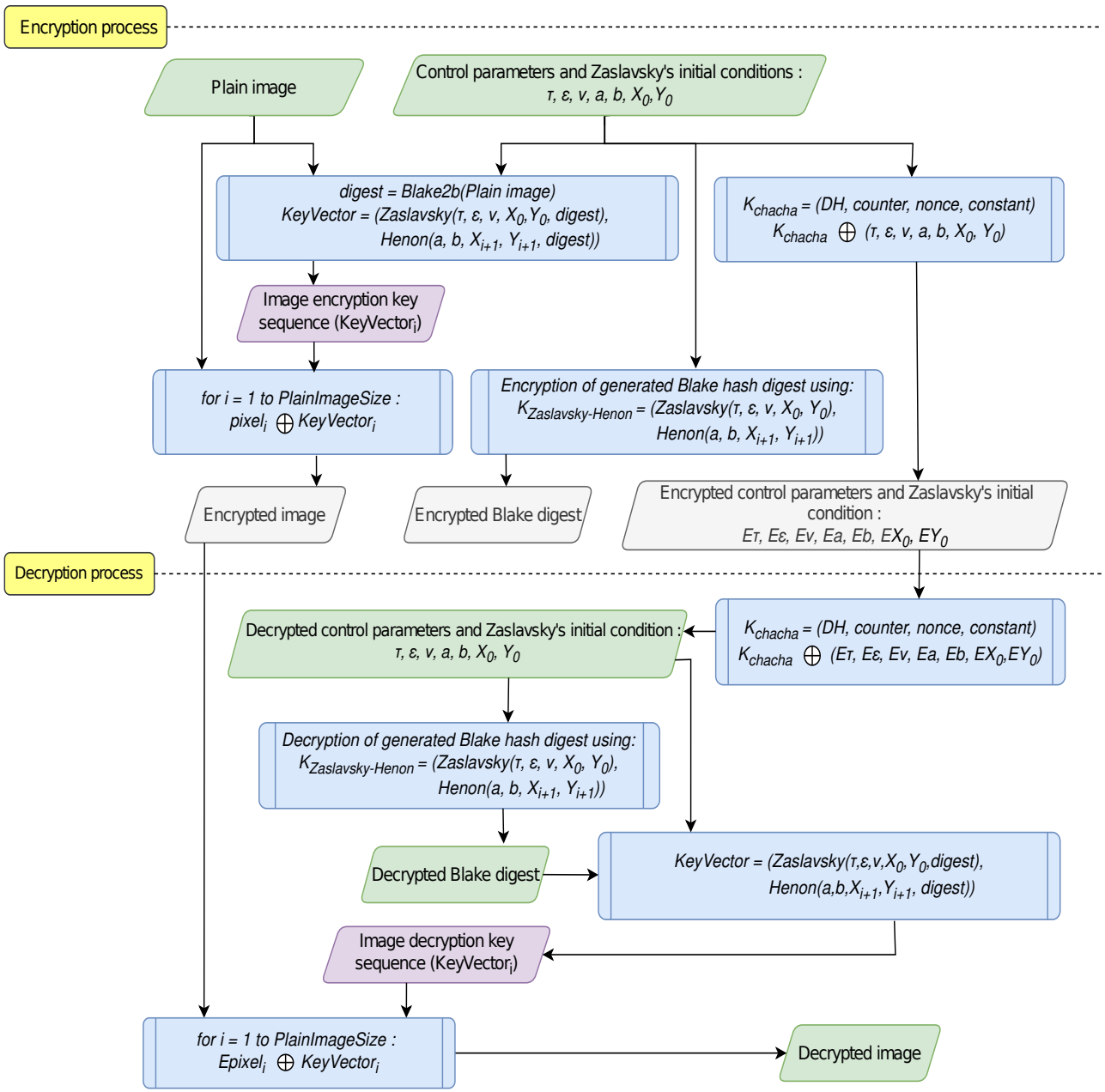

Figure 1: Global architecture of the proposed cryptosystem

4.1 Image encryption key generation

The encryption key generation process for medical images employs a Pseudo Random Key Sequence Generator (PRKSG). First, the plain image undergoes the Blake2b hash function resulting in a $512 - bit$ hash digest divided into 64 blocks of $8 - bit$. Then, each block is used as a new input parameter to the 2D-Zaslavsky chaotic map (see Equation 3), multiplying it by each occurrence of X_i resulting in X_{i+1} and Y_{i+1} random sequences.

$$\begin{cases} X_{i+1} = (X_i \times digest_k + \nu \times (1 + \mu \times Y_i) + \varepsilon \times \nu \\ \quad \times \mu \times \cos(2 \times \pi \times X_i \times digest_k)) \bmod 1 \\ Y_{i+1} = e^{-\tau} \times (Y_i + \varepsilon \times \cos(2 \times \pi \times X_i \times digest_k)) \end{cases} \tag{3}$$

where

– $digest_k$ is a hash digest block with $k = 0..63$.

After that, each block of the same digest and the Zaslavsky outputs are used as new input parameters in the

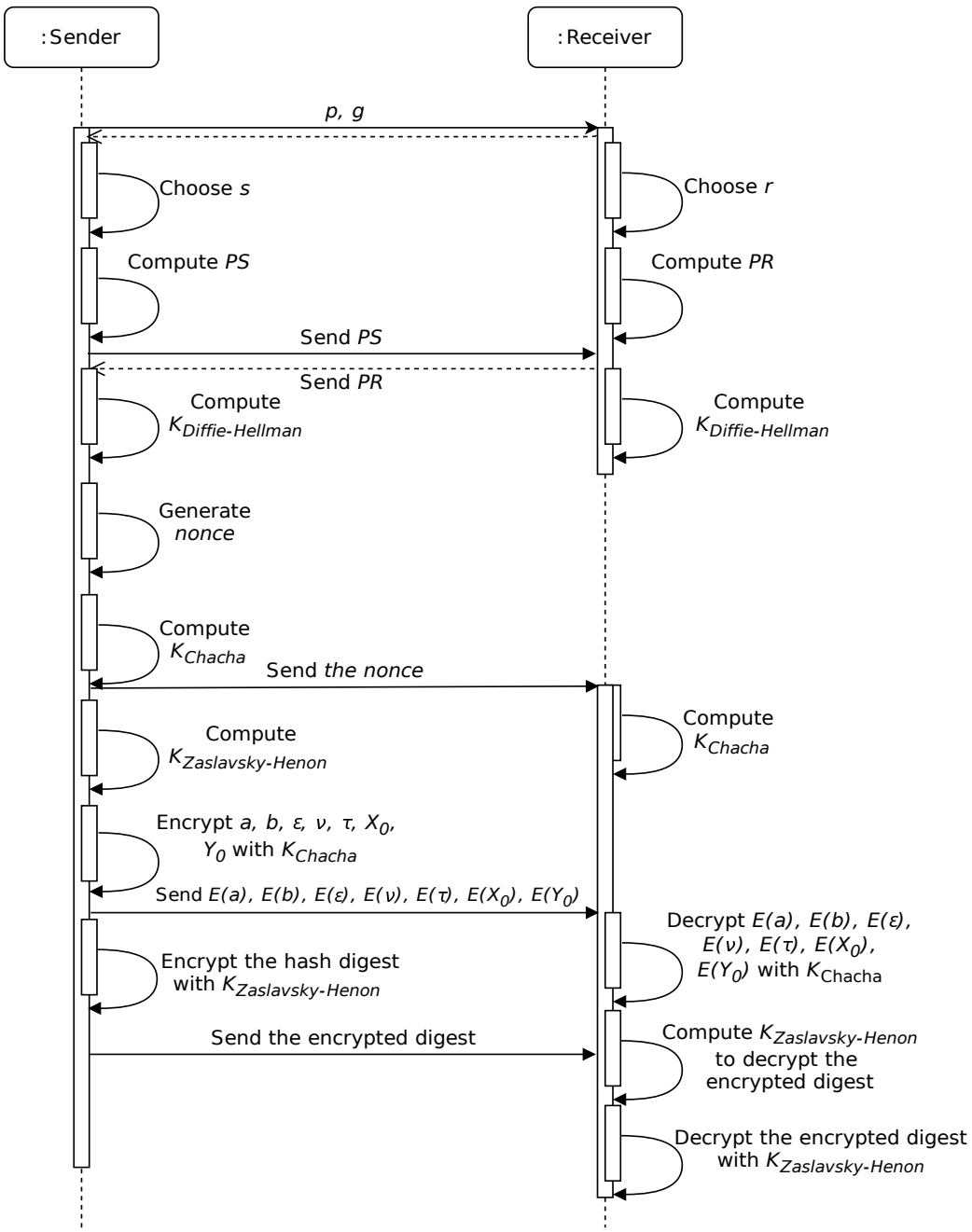

Figure 2: The process of the proposed key establishment technique

Henon chaotic map (see equation 4). The Zaslavsky outputs are used as random sequences of the Henon map, multiplying each digest block by each occurrence of X_{i+1}. The X'_{i+1} random sequences resulting from this map are used to generate a vector of a pseudo-random sequence of image encryption keys (KeyVector). This sequence is generated by choosing the 8 least significant bits of the X'_{i+1} sequences. It results in keys between 0 and 255 used to encrypt medical images in the encryption process.

$$\begin{cases} X'_{i+1} = Y_{i+1} + 1 - a \times (X_{i+1} \times digest_k)^2 \\ Y'_{i+1} = b \times X_{i+1} \times digest_k \end{cases} \quad (4)$$

The final encryption key sequence vector is obtained at the 8^{th} round after $NbRnd = 8$ rounds over the image. This results in $8 \times image\_size$ iterations, where each X'_{i+1} evolves

iteratively and each generated key depends on the previous one. This iterative process ensures that the final key vector, whose length matches the image size, is the result of a cumulative and highly sensitive chaotic evolution across the 8 rounds. The PRKSG is detailed in the Algorithm 1. It takes as input the plain image (PlainImage), Zaslavsky chaotic map initial conditions (X_0, Y_0) and control parameters (τ, ν, ε), and Henon chaotic map control parameters (a, b). Resulting in a pseudo-random encryption key sequence vector (KeyVector), used in the encryption process.

Algorithm 1 Pseudo Random Key Sequence Generator

Require: PlainImage, τ, ν, ε, a, b, $(X_0,\ Y_0)$

Ensure: KeyVector

```
Hash_Digest = Blake2b(PlainImage, digest_size = 64)
DigestVector = 64_blocks(Hash_Digest)
[Row, Column] = size(PlainImage)
image_size = Row× Column
NbRnd = 8
k = 0
for l in (NbRnd) do
    for i in (image_size) do
        (X_{i+1}, Y_{i+1}) = ZaslavskyMap((X_i, Y_i), τ, ν, ε,
                             DigestVector(k))
        (X'_{i+1}, Y'_{i+1}) = HenonMap((X_{i+1}, Y_{i+1}), a, b,
                             DigestVector(k))
        k = k + 1
        if k == 64
            k = 0
        KeyVector(i) = X'_{i+1}(8_LSB)
    end for
end for
```

4.2 Image encryption process

For the image encryption process, an XOR operation is performed between each pixel from the plain image and its corresponding key from the random key sequence vector. This ensures that each pixel of the plain image has a unique encryption key, thereby enhancing security, improving robustness, and ensuring the confidentiality of the image. The diffusion process is thus achieved, resulting in an encrypted image (EncImage) (see Algorithm 2).

4.3 Proposed key establishment technique

Rather than transmitting encryption keys directly, the proposed technique requires only securely sending the encrypted digest, initial conditions, and control parameters. This enables both parties to generate the image encryption keys independently, thereby ensuring data confidentiality and integrity. The proposed key establishment technique, which integrates the Diffie-Hellman protocol, the ChaCha20 stream cipher, and the 2D Zaslavsky and Henon chaotic maps is detailed in the following five phases (see Figure 2).

Algorithm 2 Image Encryption

Require: PlainImage, KeyVector

Ensure: EncImage

```
[Row, Column] = size(PlainImage)
k = 0
for i in (Row) do
    for j in (Column) do
        EncImage(i, j) = KeyVector(k) ⊕ PlainImage(i, j)
        k = k + 1
    end for
end for
```

4.3.1 Encryption key generation phase

When the receiver is ready to receive the encrypted image, the cryptosystem initiates the key generation process. During this process, the Diffie-Hellman protocol is used, where the sender and the receiver independently select random secret numbers (private keys) s and r, respectively.

Then, they compute their public keys, PS and PR, respectively, using the agreed prime number p and its primitive root g. The two entities exchange their public keys openly and independently calculate the $256-bit$ secret key $K_{Diffie-Hellman}$ as follows:

- The sender:

$$K_{Diffie-Hellman} = (PR)^s \ mod\ p$$

- The receiver:

$$K_{Diffie-Hellman} = (PS)^r \ mod\ p$$

Based on the properties of modular exponentiation, the sender and the receiver use the generated $K_{Diffie-Hellman}$ key to derive an encryption key from the ChaCha20 algorithm. To do this, the $K_{Diffie-Hellman}$ key is hashed using the SHA−256 algorithm resulting in a $256-bit$ digest used as secret key suitable for ChaCha20, which strengthens its entropy. This permits both communicating entities to compute the same K_{Chacha} encryption key independently rather than sharing it.

The encryption K_{Chacha} key is derived using several components. A $256-bit$ secret key obtained from the $K_{Diffie-Hellman}$ key, the $96-bit$ nonce generated using a cryptographically secure pseudo-random number generator[1], the $32-bit$ counter and the $128-bit$ constant. These elements are represented in a 4×4 initial block where each cell is a $32-bit$, as previously seen in Table 2. This block undergoes 20 rounds, with each round consisting of 4 quarter-round operations. For odd rounds, the quarter-rounds operate on the columns of the block, while during even rounds,

[1] https://www.geeksforgeeks.org/python-os-urandom-method/

they operate on the rows of the block. After 20 rounds, the final resulting block is XORed with the initial block to generate the K_{Chacha} encryption key.

To drive the same K_{Chacha} encryption key on the receiver's side, the sender transmits the nonce only. Since the sender generates a new nonce for each image, a unique K_{Chacha} encryption key is produced for each image. As a result, the risk of nonce reuse is eliminated, ensuring the security and integrity of the encryption process.

4.3.2 Secret information encryption phase

In this phase, the ChaCha20 algorithm encrypts the control parameters and initial conditions of the 2D-Zaslavsky map $(\tau, \nu, \varepsilon, X_0, Y_0)$ and the control parameters of the 2D-Henon map (a, b). These parameters are used for generating the image encryption keys. Hence, an XOR operation is performed between the K_{Chacha} encryption key and each secret information. The process should be as follows:

$$\begin{aligned} E(\tau) &= \tau \oplus K_{Chacha} \\ E(\nu) &= \nu \oplus K_{Chacha} \\ E(\varepsilon) &= \varepsilon \oplus K_{Chacha} \\ E(a) &= a \oplus K_{Chacha} \\ E(b) &= b \oplus K_{Chacha} \\ E(X_0) &= X_0 \oplus K_{Chacha} \\ E(Y_0) &= Y_0 \oplus K_{Chacha} \end{aligned}$$

Finally, the encrypted secret information $E(\tau)$, $E(\nu)$, $E(\varepsilon)$, $E(a)$, $E(b)$, $(E(X_0), E(Y_0))$ and the nonce are sent to the receiver.

4.3.3 Secret information decryption phase

When receiving the encrypted secret information $E(\tau)$, $E(\nu), E(\varepsilon), E(a), E(b), (E(X_0), E(Y_0))$ and the nonce, the receiver decrypts them with the same K_{Chacha} encryption key used in the encryption process. This key is computed using the $256 - bit$ secret key derived from $K_{Diffie-Hellman}$ key. The generation also involves the received $96 - bit$ nonce, a $32 - bit$ counter, and a constant. The process reproduces the same keys used during the Encryption key generation phase. Then, the ChaCha20 decryption algorithm is used to decrypt the secret information. In this algorithm, an XOR operation is performed between the decryption key K_{Chacha} and each encrypted information as follows:

$$\begin{aligned} \tau &= E(\tau) \oplus K_{Chacha} \\ \nu &= E(\nu) \oplus K_{Chacha} \\ \varepsilon &= E(\varepsilon) \oplus K_{Chacha} \\ a &= E(a) \oplus K_{Chacha} \\ b &= E(b) \oplus K_{Chacha} \\ X_0 &= E(X_0) \oplus K_{Chacha} \\ Y_0 &= E(Y_0) \oplus K_{Chacha} \end{aligned}$$

4.3.4 Blake hash digest encryption phase

The PRKSG employs the 2D-Zaslavsky (see equation 1) and 2D-Henon (see equation 2) chaotic maps for Blake hash digest encryption key generation as seen in Algorithm 1 without merging the 64 blocks Blake digest.

Hence, based on the secret information, the 2D Zaslavsky-Henon encryption algorithm generates a vector of 64 encryption keys, denoted as $(K_{Zaslavsky-Henon})_k$. The key generation process begins with the initial sequences (X_k, Y_k) used by the Zaslavsky map to produce (X_{k+1}, Y_{k+1}) random sequences after 8 iteration. These resulting sequences then serve as inputs to the Henon map, which undergoes another 8 iteration to generate the (X'_{k+1}, Y'_{k+1}) random sequences. Finally, the $(K_{Zaslavsky-Henon})_k$ key vector corresponds to the (X'_{k+1}) sequence, extracted from the Henon map output (see equation 5).

$$\begin{cases} (X_{k+1},\, Y_{k+1})_l = ZaslavskyMap((X_k,\, Y_k)_l) \\ (X'_{k+1},\, Y'_{k+1})_l = HenonMap((X_{k+1},\, Y_{k+1})_l) \\ (K_{Zaslavsky-Henon})_k = (X'_{k+1})_l \end{cases} \tag{5}$$

– where $k = 0..63$ and $l = 0..7$

Then, an XOR operation is achieved between the first $8 - bit$ of each $(K_{Zaslavsky-Henon})_{k=0..63}$ key and the $8 - bit$ of each block digest, resulting in 64 encrypted blocks digest.

4.3.5 Blake hash digest decryption phase

The receiver decrypts the encrypted digest using the $(K_{Zaslavsky-Henon})_{k=0..63}$ decryption keys. These keys are generated in the same way as in the encryption process using the decrypted secret information resulting from the Secret information decryption phase. After that, an XOR operation is achieved between the first $8 - bit$ of each $(K_{Zaslavsky-Henon})_{k=0..63}$ decryption key and each $8 - bit$ encrypted block digest. The results are a sequence of $8 - bit$ decrypted blocks, concatenated to form the $512 - bit$ Blake hash digest.

4.4 Image decryption process

After decrypting the Blake hash digest and the secret information, the receiver uses them in the PRKSG to generate the same pseudo-random key sequence vector as the sender (see Algorithm 1). This vector is then used in the decryption phase, where each pixel of the encrypted image is XORed with the corresponding key value. This process results in the decrypted image (DecImage) (see Algorithm 3).

Note

The proposed key establishment technique can be applied to all existing cryptosystems based on chaotic maps such

Algorithm 3 Image Decryption

Require: EncImage, KeyVector
Ensure: DecImage
[Row, Column] = size(EncImage)
$k = 0$
for i in (Row) **do**
  for j in (Column) **do**
    DecImage(i, j) = KeyVector(k) ⊕ EncImage(i, j)
    $k = k + 1$
  end for
end for

as Arnold Cat [5], Baker [14], Logistic [22, 40], and others. This technique enables to secure the exchanged control parameters and initial conditions between communicating entities. In cases where a hash function is applied, such as Blake or another, the chaotic map used can be employed to encrypt the generated digest. This approach ensures the security and integrity of the exchanged data. It leverages the properties of symmetric and asymmetric algorithms in chaotic systems for cryptographic purposes.

5 Security analysis and evaluation

This section presents a detailed overview of the simulation environment and the metrics used to evaluate the proposed key establishment technique and the overall cryptosystem. It also includes an in-depth analysis, discussion, and comparison of the results to demonstrate the cryptosystem's effectiveness.

5.1 Simulation environment

Empirical experiments were conducted to evaluate the proposed key establishment technique on a dataset. This dataset includes 100 various grayscale and RGB medical endoscopic images. These images have different resolutions, specifically 256×256 and 610×610 pixels (see Figure 3). A public hospital provided them, and we then added them to the Kaggle open-access dataset (https://www.kaggle.com). The simulation was conducted on two Raspberry Pi 3 Model B+ boards [13, 17]. Each board is equipped with a Broadcom BCM2837B0, $64-bit$ quad-core ARM cortex-A53 $1.4\,GHz$ processor, a memory of $1\,GB$ LPDDR2 SDRAM, and a Raspberry Pi operating system ($32-bit$, based on Debian 11 Bullseye). This economical, palm-sized computer board is helpful in terms of minimizing the required system hardware.

The entire chaos-based system, including the novel key establishment technique, was implemented in Python 3.11. Several libraries were used, such as NumPy for matrix operations, OpenCV for image handling, and hashlib for Blake2 hashing. The X'_{i+1} Henon chaotic outputs were quantized by extracting the 8 least significant bits from their $32-bit$ floating-point binary representation. This discretization process reduces numerical instability and ensures the generation of a uniformly distributed $8-bit$ key vector suitable for XOR encryption and decryption.

Figure 4 illustrates the overall encryption and decryption cryptosystem, implemented using two Raspberry Pi devices. Each Raspberry Pi establishes a Virtual Network Computing (VNC) session for seamless remote access. One assumes the role of the transmitter, capturing and encrypting the images, while the other acts as the receiver, decrypting and processing the received images. To assess the computational efficiency of the encryption and decryption processes, execution time and corresponding throughput were evaluated on the proposed dataset. For the 256×256 grayscale images, the average encryption/decryption time was 0.1 seconds, while for the 256×256 RGB images, it was 0.2 seconds. For images of 610×610 resolutions, the encryption/decryption time was 0.6 seconds (grayscale) and 1.5 seconds (RGB). These results correspond to throughput of approximately 10 images/s (256×256 grayscale), 5 images/s (256×256 RGB), 1.7 images/s (610×610 grayscale), and 0.7 images/s (610×610 RGB).

Specific values of the control parameters (τ, ν, ε, a, b), initial conditions (X_0, Y_0), and the number of iterations have been carefully selected to ensure chaotic behavior the cryptosystem (see Table 3). These values are not derived from the classical chaotic conditions of the individual maps. Instead, they are tailored to produce a robust and unpredictable pseudo-random behavior when combined within the key generation mechanism. This strengthens the performance of the cryptosystem by enhancing entropy and resistance to attacks while preserving the underlying chaotic dynamics.

Table 3: Parameter values of the proposed PRKSG and the key establishment technique

Parameters	Values
a	1.4
b	0.3
ε	9.1
τ	3.0
ν	12.6695
X_0	0.1
Y_0	0.1
Number of iterations	$8 \times image\_size$

5.2 Metrics

To allow fair comparisons between the evaluated solutions, various metrics were considered. These metrics assess the effectiveness of both the cryptosystem and the novel key establishment technique:

– The histogram analysis test was conducted to compare the pixel distributions of the original, encrypted, and decrypted images to measure their uniformity.

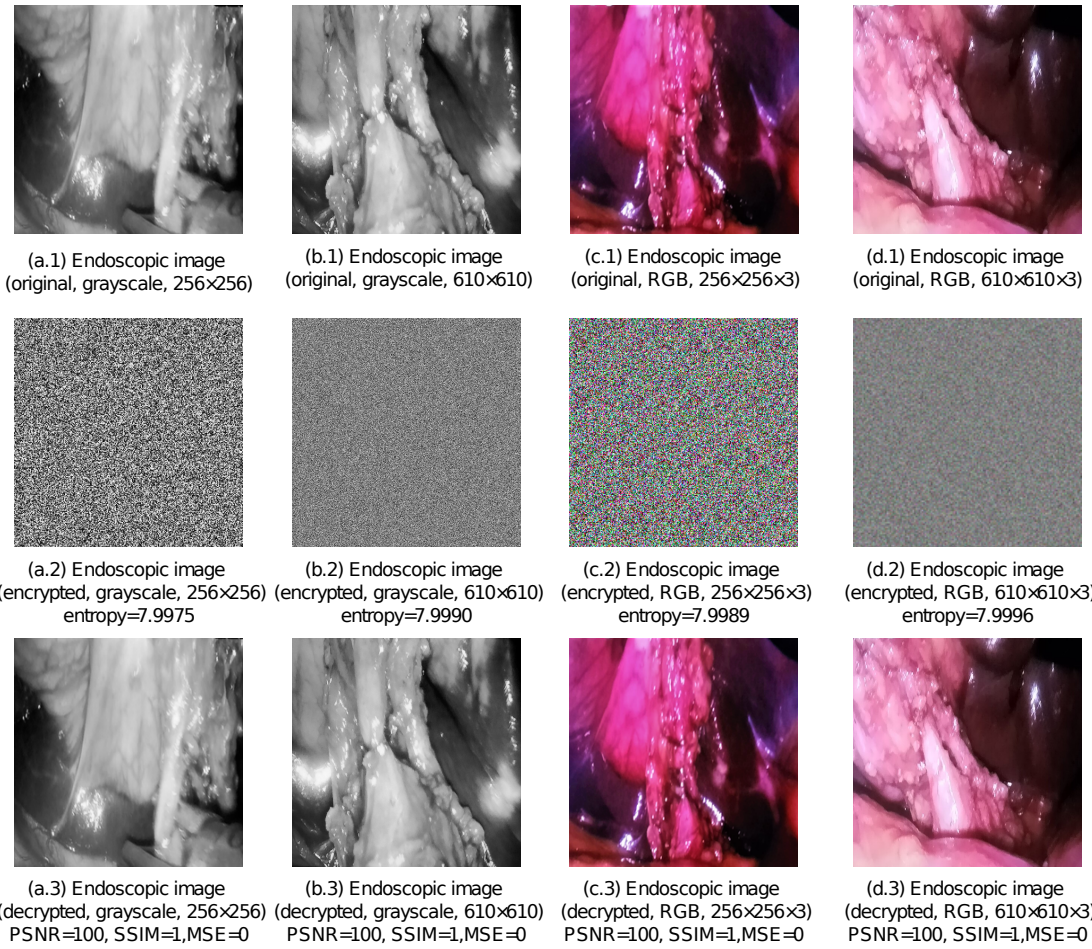

Figure 3: Original, encrypted, and decrypted endoscopic image illustrations in grayscale and RGB formats at two resolutions (256×256 and 610×610). The encrypted images include entropy values to assess information randomness and uniformity. The decrypted images include PSNR, SSIM, and MSE scores to evaluate the integrity of image recovery

- Time consumed by the novel key establishment technique to encrypt the secret information and the Blake hash digest.
- PSNR, SSIM, and MSE measurements were used to evaluate the quality of the decrypted images.
- Entropy was conducted to assess the uniformity of the encrypted images and, consequently, to evaluate the cryptosystem's resistance to entropy attacks.
- NPCR and UACI metrics were used to assess the effectiveness of the proposed cryptosystem against differential and known plain-text attacks.
- Time and space complexity were conducted to theoretically analyze the scalability and performance of the encryption and decryption processes.
- Key space was analyzed to assess the robustness of the proposed key establishment technique against brute-force attacks.

5.3 Results analysis

The results obtained from the experimentation are analyzed based on the metrics listed above. This analysis evaluates the performance of the proposed key establishment technique within the chaos-based cryptosystem. The results are then compared with those of other works presented in [2], [4], [18], [20], [25], [34] and [35]. These studies were selected as they propose well-established key techniques and evaluate their performances using metrics comparable to those employed in this work.

5.3.1 Histogram analysis

The histogram analysis assesses the pixel value distribution within plain, encrypted, and decrypted images. When comparing the histograms of the RGB and grayscale decrypted images (256×256 and 610×610 pixels) with their original images (see Figure 5), the results show that both share identical patterns and intensity levels. This similarity in his-

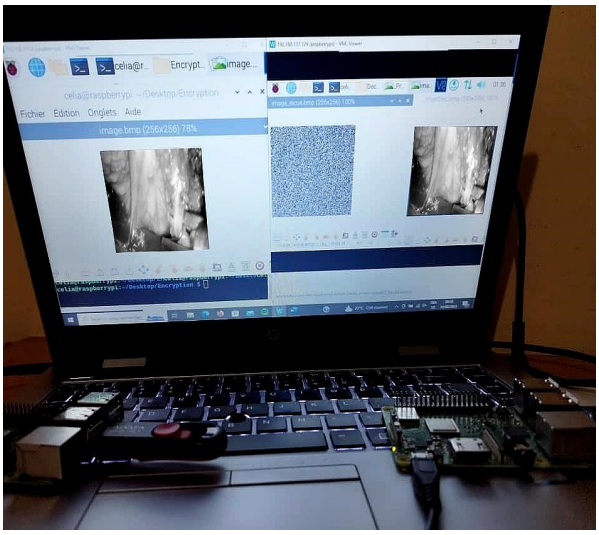

Figure 4: Global encryption and decryption system in the Raspberry Pi's

Table 4: Time Comparison

Key establishment/exchange	Time (s)
Diffie-Hellman-Chacha	0.00017
2D Zaslavsky-Henon	0.000116
Hsieh et al. [20]	0.4
Sureshkumar et al. [35]	0.00794674

tograms evidences that the decryption process was successful. All the information from the original images was accurately retrieved without any loss, maintaining the integrity of the medical images. The encrypted images exhibit a uniform histogram distribution, reflecting effective encryption and strong randomness.

5.3.2 Time consumption analysis

The time required for the proposed key establishment process was evaluated. It includes the execution time of the Diffie–Hellman–ChaCha20 scheme used to encrypt the secret information. It also includes the Zaslavsky–Henon system used to encrypt the Blake hash digest. The key establishment time remains constant and is independent of the image size.

Table 4 presents a comparative analysis of the key establishment process speed between the proposed technique and the techniques presented in [20] and [35], measured in seconds. The results demonstrate that the proposed technique process operates in real time illustrating the minimal latency experienced during the process. Moreover, it outperforms the techniques in [20] and [35] in terms of speed, mainly due to the latency issues encountered by the existing method. The real-time nature of this technique enhances the security of the secret information transmission and ensures a smooth and uninterrupted data transfer experience.

Table 5: Image quality comparison

Works	PSNR	MSE	SSIM
Proposed cryptosystem	100	0	1
Adeshina et al. [2]	99.9	–	0.9969
Hamza et al. [18]	73.5965	0.0028	0.9998
Mir et al. [25]	–	–	0.0018

Table 6: Entropy test

Images	Entropy
$[256 \times 256]$	7.9975
$[256 \times 256 \times 3]$	7.9989
$[610 \times 610]$	7.9990
$[610 \times 610 \times 3]$	7.9996

5.3.3 Image quality analysis

The performance of the proposed technique within the cryptosystem in preserving the quality and integrity of the medical image was assessed. To do this, the generated grayscale and RGB decrypted images were analyzed using PSNR, MSE, and SSIM tests. The PSNR should be at least 30 *dB* to consider that the decrypted image has good quality [18]. The SSIM should be as close as possible to 1, and the MSE should be close to 0. Based on the results presented in Table 5, the proposed technique demonstrates outstanding quality and highly desirable features for decrypted images. It also exhibits no data loss, compared to the existing works [2], [18], and [25]. Indeed, the results show a PSNR value of 100 *dB* indicating an exceptionally high level of fidelity. The MSE value of 0 indicates a perfect match between the original and decrypted images, and the SSIM value of 1 indicates a total structural similarity. These results highlight the effectiveness of the proposed cryptosystem based on the key establishment technique in preserving the integrity of transmitted medical images.

5.3.4 Information entropy analysis

The average entropy values of the encrypted images produced by the proposed cryptosystem on the given dataset are analyzed and evaluated. As shown in Table 6, the entropy values are consistently close to 8, indicating a high level of randomness. This confirms the effectiveness of the proposed cryptosystem in converting key-frames into uniformly random images, demonstrating its strong resistance to entropy attacks.

In comparison to [4], [18], and [34] well-established cryptosystems, Table 7 shows that the proposed cryptosystem achieves on average, better entropy results, validating its effectiveness.

5.3.5 Differential and known plain-text attacks resistance analysis

To assess the robustness of the proposed cryptosystem against differential attack (DA) and known-plaintext attack

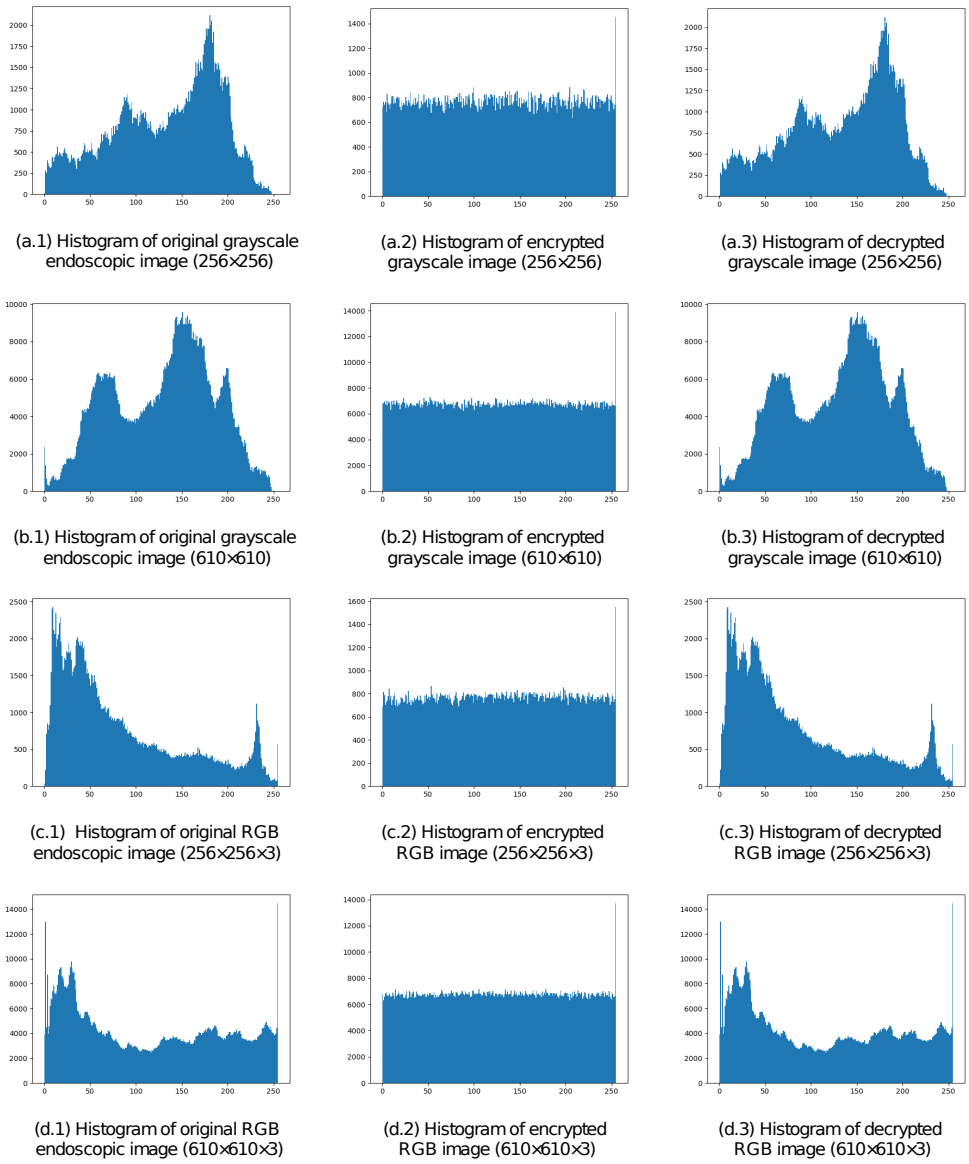

Figure 5: Histograms comparison for original, encrypted, and decrypted endoscopic images in grayscale and RGB formats at two resolutions (256×256 and 610×610). The encrypted images exhibit a uniform histogram distribution, reflecting effective encryption and strong randomness. Decrypted histograms demonstrate a similarity to the original ones, validating the accuracy of image recovery.

Table 7: Entropy comparison

Works	$[256 \times 256]$	$[610 \times 610 \times 3]$
Proposed cryptosystem	7.9975	7.9996
Almola et al. [4]	7.89	–
Hamza et al. [18]	–	7.9994
Shahid et al. [34]	7.98	–

(KPA), the NPCR and then UACI metrics were employed [18]. NPCR evaluates the percentage of pixel changes in the cipher image resulting from a one-pixel modification in the plain image. This reflects the system's sensitivity to slight input changes. On the other hand, UACI quantifies the average intensity difference between both the plain and encrypted images. It serves as an indicator of how effectively the encryption algorithm diffuses pixel values.

1. Differential attack resistance analysis:

   A comparison between the original images of the proposed dataset and their encrypted images was conducted using the NPCR and UACI tests. As reported in Table 8, the obtained values of $NPCR \geq 99.62\%$ and $UACI \geq 33.50\%$ indicate a high degree of sensitivity

Table 8: NPCR and UACI for DA resistance

Images	NPCR	UACI
$[256 \times 256]$	99.62	33.50
$[256 \times 256 \times 3]$	99.64	35.07
$[610 \times 610]$	99.62	33.61
$[610 \times 610 \times 3]$	99.62	34.40

to small changes in the inputs. These results confirm the cryptosystem's strong diffusion property.

Well-established cryptosystems [18] and [34] show results of NPCR = 99.60% and UACI = 33.46% for 610×610 RGB images and NPCR = 99.62% and UACI = 33.46% for 256×256 grayscale images respectively. In comparison, the proposed method exhibits comparable or higher resistance to differential attacks.

2. Known plaintext attack resistance analysis:

   A slight modification was introduced to the original images dataset to assess the robustness of the proposed cryptosystem against KPA. The resulting encrypted images for the original and modified images were then compared. The high NPCR and UACI values reported in Table 9 indicate that the encrypted outputs exhibit significant differences despite slight changes. These results demonstrate the cryptosystem's high sensitivity to plain image variations. This confirms its robustness against KPA and its effectiveness in achieving a strong confusion property.

Table 9: NPCR and UACI for KPA resistance

Images	NPCR	UACI
$[256 \times 256]$	99.62	33.46
$[256 \times 256 \times 3]$	99.60	33.46
$[610 \times 610]$	99.61	33.45
$[610 \times 610 \times 3]$	99.60	33.45

5.3.6 Time and space complexity analysis

The proposed encryption and decryption processes consist of three main steps: pseudo-random key sequence generator, key establishment, and diffusion. The PRKSG used the hash of the plain image combined with two 2D chaotic maps (Zaslavsky and Henon) to generate a pseudo-random key sequence of size equal to the image ($Rows \times Columns$). It processes each pixel, resulting in a time and space complexity of $O(Rows \times Columns)$. The key establishment technique integrates the Diffie–Hellman protocol with the ChaCha20 algorithm, along with the Zaslavsky and Henon chaotic maps. This combination securely generates identical keys between the communicating entities without transmitting them. As this step is independent of the image size, its complexity is considered constant ($O(1)$) from the perspective of encryption and decryption processes. The diffusion step applies an XOR operation between the image and the generated pseudo-random key sequence. Since each pixel is processed once, this step also has a time and space complexity of $O(Rows \times Columns)$. Consequently, the overall time and space complexity of the encryption and decryption processes is $O(Rows \times Columns)$.

The proposed cryptosystem exhibits linear time and space complexity concerning the image size. This confirms its scalability and suitability for real-time and resource-constrained environments. Moreover, the constant time of the key establishment technique contributes to the overall efficiency and reinforces the system's ability. It ensures fast and secure image transmission without compromising system performance or robustness.

5.3.7 Key space analysis

The objective of the key space analysis is to evaluate the robustness and security of the proposed technique against brute-force attacks. A $256 - bit$ key is employed for secret information encryption, resulting in a key space of 2^{256}. As for the encryption of the Blake hash digest, the key space is expanded by incorporating initial values and control parameters as secret keys (τ, ν, ε, μ, a, b, X_0, Y_0). This enhances the key space of the cryptosystem by a power of 8. Leveraging a $64 - bit$ double precision, the precision numbers can reach approximately 10^{15} [26]. In our case, the experiments were conducted with a precision of 10^{14}. Consequently, the key space is estimated to be around ($1.16 \times 10^{77} \times 10^{(14\times8)} \approx 2^{628}$), providing a robust resistance against brute-force attacks.

Table 10: Key space comparison

Works	Key space
Proposed cryptosystem	1.16×10^{189}
Hamza et al. [18]	13.74×10^{112}
Mir et al. [25]	1.16×10^{140}
Shahid et al. [34]	10^{72}

A key space comparison between the proposed cryptosystem and those presented by [18], [25], and [34] is provided in Table 10. The results indicate that the proposed cryptosystem possesses the largest key space and hence is highly resistant to brute-force attacks.

5.4 Results discussion

This subsection presents a comparative analysis of the proposed key establishment technique within the cryptosystem against existing state-of-the-art cryptosystems. This comparison particularly emphasizes the technique's suitability for resource-constrained medical applications. While several comparison tests have already been presented and analyzed in the previous sub-sections, this discussion aims to synthesize the key findings. It also highlights the broader significance of the results. As summarized in Table 11, the proposed cryptosystem consistently outperforms a range of existing cryptosystems reported in the literature ([2], [4],

Table 11: Results comparison

Works	Images	Key space	Entropy	NPCR	UACI
Proposed	$[256 \times 256]$	1.16×10^{189}	7.9975	99.62	33.50
cryptosystem	$[610 \times 610 \times 3]$	1.16×10^{189}	7.9996	99.62	34.40
Almola et al. [4]	$[96 \times 96]$	–	7.89	–	–
Hamza et al. [18]	$[640 \times 480 \times 3]$	13.74×10^{112}	7.9994	99.60	33.46
Mir et al. [25]	$[256 \times 256]$	1.16×10^{140}	–	–	–
Shahid et al. [34]	$[256 \times 256]$	10^{72}	7.98	99.62	33.46

[18], [20], [25], [34], and [35]). Specifically, the proposed technique achieves a faster key establishment time compared to the methods in [20] and [35]. It also ensures total image recovery after decryption compared to well-established works such as [2], [18], and [34]. The cryptosystem demonstrates high entropy, as confirmed by elevated entropy values. It shows strong resistance to differential and known-plaintext attacks, confirmed by high NPCR and UACI values. It also offers robust protection against brute-force attacks thanks to its extensive key space.

These improvements result from the association of two key elements in the proposed cryptosystem. First, the integration of the Zaslavsky and Henon chaotic maps along with the Blake hash function introduces high degrees of confusion and diffusion. This results in high statistical unpredictability and uniformity in the encrypted medical images. Second, the hybrid key establishment technique leverages the Diffie-Hellman protocol with ChaCha20. This combination ensures secure information transmission. In parallel, the Zaslavsky and Henon maps are employed to securely transmit the Blake hash digest. This significantly mitigates risks of key interception or reuse, ensuring dynamic session-based key establishment. Moreover, it overcomes the limitations of the purely chaos-derived key scheduling observed in many previous works.

Unlike traditional chaos-based encryption approaches, the proposed cryptosystem integrates the strengths of symmetric, asymmetric, and chaotic cryptographic techniques. This combination ensures both speed and resistance to common cryptanalytic attacks. This novel integration, combined with the lightweight nature of the employed components, enhances the efficiency of the proposed key establishment technique. As a result, the chaos-based cryptosystem becomes particularly suitable for constrained healthcare environments, such as IoT or telemedicine systems. Within such critical application domains, ensuring confidentiality, performance, and data integrity is essential.

6 Conclusion

Conventional methods for transmitting medical image data through unsecured channels pose significant security risks. In chaos-based cryptosystems, the communicating entities are synchronized using the synchronization technique. However, this technique has proven to be inadequate, often resulting in substantial information loss during image decryption, which can compromise the data integrity. Another significant concern is the prolonged use of the same encryption key. This can increase the risk of interception or compromise by unauthorized entities. Additionally, large encryption keys, equal to the image size, lead to inefficiencies. Indeed, transmitting such a large key significantly increases time and bandwidth overhead, making secure image transmission more complex.

In response to these challenges, we have proposed a novel key establishment technique that integrates Diffie–Hellman, ChaCha20, Zaslavsky, and Henon mechanisms. This technique combines symmetric, asymmetric, and chaos encryption methods within the chaos-based cryptosystem. Diffie-Hellman and ChaCha20 are combined to securely transmit the secret information required for generating the encryption and decryption keys. Simultaneously, the 2D Zaslavsky and Henon chaotic maps ensure secure transmission of the Blake hash digest. This method enables both communicating entities to independently generate identical encryption and decryption keys for images without directly transmitting them. Extensive analysis results demonstrate that the proposed technique is secure, robust, and fast. Moreover, it shows the effectiveness of this technique in preserving medical image integrity without any loss, as indicated by the PSNR, MSE, and SSIM tests. The proposed technique provides a promising solution for the secure, reliable, and fast transmission of medical images, ensuring the integrity and confidentiality of sensitive data.

As a direction for future work, we are currently developing a hardware implementation of the proposed cryptosystem on an FPGA platform. The main objective is to achieve higher execution speed. This is essential for real-time processing in resource-constrained IoT-based medical applications. The hardware design is expected to reduce latency and improve overall system performance.

Declarations

Availability of data

The datasets used in this study are publicly available at the Kaggle open access dataset, accessible via grayscale endoscopy images and RGB endoscopy images.

Competing interests

The authors declare that they have no conflict of interest.

Funding

This research received no specific grant from any funding agency in the public, commercial, or not-for-profit sectors.

Author contributions

All authors contributed to the study conception and design. Data collection and analysis were performed by Sylia Amarouche. The first draft of the manuscript was written by Sylia Amarouche and Said Talbi and all authors commented on previous versions of the manuscript. All authors read and approved the final manuscript.

Acknowledgement

This work was carried out in the framework of the computer research lab LARI of Mouloud Mammeri University of Tizi Ouzou, through the university training research project PRFU (Code:C00L07UN1501 20220001).

References

[1] Aamir M., Sharma S., and Grover A., ChaCha20-in-Memory for Side-Channel Resistance in IoT Edge-Node Devices. IEEE Open Journal of Circuits and Systems, 2: 833-842, 2021. https://doi.org/10.1109/OJCAS.2021.3127273

[2] Adeshina A. M., Razak S. F. A., Yogarayan S., and Sayeed S., Measuring Fidelity of Steganography Approach in Securing Clinical Data Sharing Platform using Peak Signal to Noise Ratio (PSNR) and Structural Similarity Index Measure (SSIM). Informatica, 49(11), 2025. https://doi.org/10.31449/inf.v49i11.5661

[3] Alexan W., Chen Y. L., Por L. Y., and Gabr M., Hyperchaotic maps and the single neuron model: A novel framework for chaos-based image encryption. Symmetry, 15(5): 1081, 2023. https://doi.org/10.3390/sym15051081

[4] Almola S. A., Khudeyer R. S., and Younis H. A., Biometric-Based Secure Encryption Key Generation Using Convolutional Neural Networks and Particle Swarm Optimization. Informatica, 49(16), 2025. https://doi.org/10.31449/inf.v49i16.7779

[5] Arnol'd V. I. and André A., Problèmes ergodiques de la mécanique classique. Gauthier-Villars, 9, 1967.

[6] Aumasson J. P., Neves S., Wilcox-O'Hearn Z., and Winnerlein C., "BLAKE2: simpler, smaller, fast as MD5", Applied Cryptography and Network Security: 11th International Conference, 11: 119-135, 2013. https://doi.org/10.1007/978-3-642-38980-1_8

[7] Bashir Z., Malik M. A., Hussain M., and Iqbal N., Multiple RGB images encryption algorithm based on elliptic curve, improved Diffie Hellman protocol. Multimedia Tools and Applications, 81(3): 3867-3897, 2022. https://doi.org/10.1007/s11042-021-11687-1

[8] Bernstein D. J., Salsa20 specification. eSTREAM Project algorithm description, 2005.

[9] Bernstein D. J., ChaCha, a variant of Salsa20. In Workshop record of SASC, 8(1): 3-5, 2008.

[10] Diffie W. and Hellman M. E., New directions in cryptography. In Democratizing Cryptography: The Work of Whitfield Diffie and Martin Hellman 365-390, 2022. https://doi.org/10.1145/3549993.3550007

[11] ElKamchouchi H., Salma W. M., and Abouelseoud Y., New video encryption schemes based on chaotic maps. IET Image Processing, 14(2): 397-406, 2020. https://doi.org/10.1049/iet-ipr.2018.5250

[12] Feigenbaum M. J., Quantitative universality for a class of nonlinear transformations. Journal of Statistical Physics, 19(1): 25-52, 1978. https://doi.org/10.1007/BF01020332

[13] Flurry G., Raspberry Pi 3 Model B+ Setup. Java on the Raspberry Pi: Develop Java Programs to Control Devices for Robotics, IoT, and Beyond, 21-48, 2021. https://doi.org/10.1007/978-1-4842-7264-0_2

[14] Fox R. F., Construction of the Jordan basis for the baker map. Chaos: An Interdisciplinary Journal of Nonlinear Science, 7(2): 254-269, 1997. https://doi.org/10.1063/1.166226

[15] Gao H. and Wang X., Chaotic image encryption algorithm based on zigzag transform with bidirectional crossover from random position. IEEE Access, 9: 105627–105640, 2021. https://doi.org/10.1109/ACCESS.2021.3099214

[16] Ghazvini M., Mirzadi M., and Parvar N., A modified method for image encryption based on chaotic map and genetic algorithm. Multimedia Tools and Applications, 79(37): 26927-26950, 2020. https://doi.org/10.1007/s11042-020-09058-3

[17] Halfacree G. and Upton E., Raspberry Pi user guide. John Wiley & Sons, 2016.

[18] Hamza R., Yan Z., Muhammad K., Bellavista P., and Titouna F., A privacy-preserving cryptosystem for IoT E-healthcare. Information Sciences, 527: 493-510, 2020. https://doi.org/10.1016/j.ins.2019.01.070

[19] Henon M., A two-dimensonal mapping with a strange attractor. Communications in Mathematical Physics, 50: 376-392, 2004. https://doi.org/10.1007/978-0-387-21830-4_8

[20] Hsieh Y. P., Lee K. C., Lee T. F., and Su G. J., Extended chaotic-map-based user authentication and key agreement for HIPAA privacy/security regulations. Applied Sciences, 12(11): 5701, 2022. https://doi.org/10.3390/app12115701

[21] Hussain S., Mubeen I., Ullah N., Shah S. S. U. D., Khan B. A., Zahoor M., and Sultan M. A., Modern diagnostic imaging technique applications and risk factors in the medical field: a review. BioMed research international, 2022(1): 5164970, 2022. https://doi.org/10.1155/2022/5164970

[22] Lopez-Ruiz R. and Perez-Garcia C., Dynamics of two logistic maps with a multiplicative coupling. International Journal of Bifurcation and Chaos, 2(2): 421-425, 1992. https://doi.org/10.1142/S0218127492000410

[23] Magdy M., Hosny K. M., Ghali N. I., and Ghoniemy S., Security of medical images for telemedicine: a systematic review. Multimedia Tools and Applications, 81(18): 25101-25145, 2022. https://doi.org/10.1007/s11042-022-11956-7

[24] McLaren P., Buchanan W. J., Russell G., and Tan Z., Deriving ChaCha20 key streams from targeted memory analysis. Journal of Information Security and Applications, 48: 102372, 2019. https://doi.org/10.1016/j.jisa.2019.102372

[25] Mir U. H., Lone P. N., Singh D., and Mishra D. C., A public and private key image encryption by modified approach of Vigener cipher and the chaotic maps. The Imaging Science Journal, 71(1): 82-96, 2023. https://doi.org/10.1080/13682199.2023.2175436

[26] Mousavi M. and Sadeghiyan B., A new image encryption scheme with Feistel like structure using chaotic S-box and Rubik cube based P-box. Multimedia Tools and Applications, 80(9): 13157-13177, 2021. https://doi.org/10.1007/s11042-020-10440-4

[27] Msolli A., Ajmi N., Helali A., Gassoumi A., Maaref H., and Mghaieth R., New key management scheme based on pool-hash for WSN and IoT. Journal of Information Security and Applications, 73: 103415, 2023. https://doi.org/10.1016/j.jisa.2022.103415

[28] Naik R.B. and Singh K., A Review on Applications of Chaotic Maps in Pseudo-Random Number Generators and Encryption. Annals of Data Science, 1-26, 2022. https://doi.org/10.1007/s40745-021-00364-7

[29] Park J. H., Chaos synchronization between two different chaotic dynamical systems. Chaos, Solitons & Fractals, 27(2): 549-554, 2006. https://doi.org/10.1016/j.chaos.2005.03.049

[30] Pecora L. M. and Carroll T. L., Synchronization in chaotic systems. Physical review letters, 64(8): 821, 1990. https://doi.org/10.1103/PhysRevLett.64.821

[31] Rehman M. U., Shafique A., Khalid S., and Hussain I., Dynamic substitution and confusion-diffusion-based noise-resistive image encryption using multiple chaotic maps. IEEE Access, 9: 52277-52291, 2021. https://doi.org/10.1109/ACCESS.2021.3069591

[32] Roy A., Mahanta D. R., and Mahanta L. B., A Semi-Synchronous Federated Learning Framework with Chaos-Based Encryption for Enhanced Security in Medical Image Sharing. Results in Engineering, 103886, 2025. https://doi.org/10.1016/j.rineng.2024.103886

[33] Seyhan K., Nguyen T. N., Akleylek S., Cengiz K., and Islam S. H., Bi-GISIS KE: Modified key exchange protocol with reusable keys for IoT security. Journal of Information Security and Applications, 58: 102788, 2021. https://doi.org/10.1016/j.jisa.2021.102788

[34] Shahid U., Kanwal S., Bano M., Inam S., Abdalla M. E. M., and Shaikh Z. A., Blockchain driven medical image encryption employing chaotic tent map in cloud computing. Scientific Reports, 15(1): 6236, 2025. https://doi.org/10.1038/s41598-025-90502-5

[35] Sureshkumar V., Mugunthan S., and Amin R., An enhanced mutually authenticated security protocol with key establishment for cloud enabled smart vehicle to grid network. Peer-to-Peer Networking and Applications, 15(5): 2347-2363, 2022. https://doi.org/10.1007/s12083-022-01350-3

[36] Tunali I., Gillies R. J., and Schabath M. B., Application of radiomics and artificial intelligence for lung cancer precision medicine. Cold Spring Harbor perspectives in medicine, 11(8): a039537, 2021. https://doi.org/10.1101/cshperspect.a039537

[37] Vivek J. and Gadgay B., Video steganography using chaos encryption algorithm with high efficiency video coding for data hiding. Interna-

tional Journal of Intelligent Engineering and Systems, 14(5): 15-24, 2021. http://dx.doi.org/10.22266/ijies2021.1031.02

[38] Wang M., Fu X., Yan X., and Teng L., A new chaos-based image encryption algorithm based on discrete fourier transform and improved joseph traversal. Mathematics, 12(5): 638, 2024. https://doi.org/10.3390/jimaging4010017

[39] Zaslavsky G. M., The simplest case of a strange attractor. Physics Letters A, 69(3): 145-147, 1978. https://doi.org/10.1016/0375-9601(78)90195-0

[40] Zhang, Y., Wang, J., and Chen, X., Research on Detection and Positioning Technology of UHV GIS Based on Multi-Sensor Fusion and Chaotic Cuckoo Algorithm. Informatica, 49(8), 2025. https://doi.org/10.31449/inf.v49i8.7044