Research Square

Preprints are preliminary reports that have not undergone peer review.
They should not be considered conclusive, used to inform clinical practice,
or referenced by the media as validated information.

A Dynamic Hybrid Cryptosystem Combining Unimodular Hill Cipher, Chaotic Logistic Maps, and ECC for Efficient Binary File Encryption

Samsul Arifin[1]

samsul.arifin@itsb.ac.id

Institut Teknologi Sains Bandung

Alya Maura Raditha
Institut Teknologi Sains Bandung

Ade Kurniawan
Institut Teknologi Sains Bandung

Tiawan Tiawan
Institut Teknologi Sains Bandung

Merios Gusan Putra
Institut Teknologi Sains Bandung

Edwin Kristianto Sijabat
Institut Teknologi Sains Bandung

Dani Lukman Hakim
Institut Teknologi Sains Bandung

Dwi Wijonarko
University of Jember

Article

Keywords: Hybrid Cryptosystem, Unimodular Hill Cipher, Logistic Map, Binary File Encryption, Elliptic Curve Cryptography (ECC)

Posted Date: July 30th, 2026

DOI: https://doi.org/10.21203/rs.3.rs-10410164/v1

License: This work is licensed under a Creative Commons Attribution 4.0 International License.
Read Full License

Additional Declarations: No competing interests reported.

A Dynamic Hybrid Cryptosystem Combining Unimodular Hill Cipher, Chaotic Logistic Maps, and ECC for Efficient Binary File Encryption

Samsul Arifin[1], Alya Maura Raditha[1], Ade Kurniawan[1], Tiawan[2], Merios Gusan Putra[2], Edwin Kristianto Sijabat[3], Dani Lukman Hakim[4], Dwi Wijonarko[5]

[1]*Department of Data Science, Faculty of Digital Design and Business, Institut Teknologi Sains Bandung; Bekasi, West Java, 17530, Indonesia*

[2]*Department of Digital Business, Faculty of Digital Design and Business, Institut Teknologi Sains Bandung, Bekasi, West Java, 17530 Indonesia*

[3]*Department of Pulp and Paper Processing Technology, Faculty of Vocational, Institut Teknologi Sains Bandung; Bekasi, West Java, 17530, Indonesia*

[4]*Department of Palm Oil Processing Technology, Faculty of Vocational, Institut Teknologi Sains Bandung; Bekasi, West Java, 17530, Indonesia*

[5]*Department of Information Technology, Faculty of Computer Science, University of Jember, Jember, East Java, 68121, Indonesia*

**Corresponding author: samsul.arifin@itsb.ac.id*

Abstract — This study introduces the Hybrid Split-Based Chaotic Cryptosystem (HSBCC), a unified encryption framework designed to secure arbitrary binary files through the integration of a dynamically generated Unimodular Hill Cipher, Logistic Map-based chaotic key generation, Shift Cipher 128, AES-based session key protection, and Elliptic Curve Cryptography (ECC). The proposed architecture adopts a split-based encryption mechanism that divides plaintext into a primary segment encrypted using a dynamically reconstructed unimodular Hill Cipher and a secondary segment protected with Shift Cipher 128. This strategy removes the need for padding while preserving the original file size and format. The encryption matrix is deterministically reconstructed from user passwords using a Logistic Map with a control parameter of r = 3.923, ensuring high key sensitivity, reproducibility, and guaranteed matrix invertibility through unimodular matrix generation. Performance evaluation was conducted using benchmark images, medical images, PDF documents, and various binary files to assess both security and functionality. Experimental results indicate that the proposed cryptosystem achieves ciphertext entropy values approaching the theoretical maximum, produces nearly zero adjacent-pixel correlations, exhibits strong avalanche characteristics, and generates highly uniform ciphertext distributions. Integrity verification demonstrates perfect lossless decryption, with all recovered files producing MD5 hashes identical to their original counterparts. Additional experiments using an 815,491-byte PDF document and matrix dimensions of 8, 88, and 888 further confirm deterministic key reconstruction, ciphertext uniqueness, file-size preservation, and complete plaintext recovery. Scalability analysis reveals moderate computational overhead while maintaining practical encryption performance across different matrix dimensions. Overall, the findings demonstrate that HSBCC provides a secure, scalable, and flexible cryptographic framework capable of protecting arbitrary binary data with strong cryptographic robustness, deterministic key generation, and complete preservation of file integrity and format.

Index Terms — Hybrid Cryptosystem; Unimodular Hill Cipher; Logistic Map; Binary File Encryption; Elliptic Curve Cryptography (ECC)

I. INTRODUCTION

Modern cryptographic systems are increasingly challenged by the explosive growth of digital data and the rising sophistication of cyberattacks. Traditional encryption schemes often rely on a single mathematical structure, which may become vulnerable under advanced cryptanalytic techniques. This motivates the design of hybrid cryptosystems that combine multiple cryptographic primitives to improve resistance against both statistical and structural attacks. The proposed framework, referred to as the Hybrid Split-Based Chaotic Cryptosystem (HSBCC), integrates symmetric encryption, chaotic dynamical systems, and public-key cryptography into unified architecture. The goal is to enhance confidentiality, diffusion, and confusion properties while maintaining computational feasibility for binary file encryption [1].

At its core, HSBCC introduces a split-based encryption strategy. Given a plaintext byte sequence $P \in \mathbb{Z}_{256}^{N}$, the system partitions the data into two segments:

$$P = P_{hill} \cup P_{shift}$$

where P_{hill}undergoes matrix transformation and $P_{shift}$is processed using modular arithmetic shifting [2]. The Hill cipher component is defined as:

$$C_{hill} = K \cdot P_{hill} \bmod 256$$

where $K \in \mathbb{Z}_{256}^{n \times n}$is a dynamically generated unimodular matrix derived from a chaotic map. This ensures invertibility under modulo 256 arithmetic. The shift-based cipher is defined as a nonlinear transformation:

$$C_{shift} = (P_{shift} + 128) \bmod 256$$

This operation introduces byte-level diffusion while maintaining computational simplicity [3]. A key innovation in HSBCC is the generation of the transformation matrix using the logistic chaotic map:

$$x_{n+1} = r x_n (1 - x_n), r = 3.923$$

This chaotic system ensures high sensitivity to initial conditions, controlled by a password-derived seed x_0 [4]. The unimodular matrix construction guarantees reversibility, ensuring that:

$$K^{-1} \cdot K \equiv I(\text{mod}256)$$

This property is essential for lossless decryption of encrypted binary files. To enhance cryptographic strength, HSBCC incorporates ECC-based key generation using elliptic curve cryptography over the P-256 curve. The key pair is defined as:

$$(sk, pk) \leftarrow \text{ECC}_{P-256}$$

ensuring secure asymmetric key exchange [5]. Additionally, a session key is generated using a cryptographically secure random function:

$$K_s \leftarrow \text{Random}(256 \text{ bits})$$

This session key is used for symmetric encryption layers, improving forward secrecy. The session key is protected using AES in EAX mode:

$$C_s = \text{AES}_{K_m}^{EAX}(K_s)$$

where $K_m$is derived from user passwords via PBKDF2. Key derivation is as follows:

$$K_m = \text{PBKDF2}(password_1 \mid \mid password_2, salt, 100000)$$

This strengthens resistance against brute-force and dictionary attacks [6], [7]. The architecture of HSBCC can be summarized as a layered encryption pipeline, which can be seen in Table 1.

Table 1. THE ARCHITECTURE OF HSBCC

Layer	Technique	Function
L1	Logistic Map	Key matrix generation
L2	Hill Cipher	Block diffusion
L3	Shift Cipher	Byte scrambling
L4	AES-EAX	Session key protection
L5	ECC	Public-key security

The split-based design reduces computational overhead by isolating high-complexity matrix operations to only a portion of the dataset, while lightweight transformations handle the remainder [8]. An illustrative workflow of HSBCC can be described as an infographic pipeline:

```
Plaintext → Split → [Hill Cipher Path] + [Shift Cipher Path]
          → Merge → Encrypted File
          → AES/ECC Layer → Secure Key Packaging
```

During decryption, inverse transformations are applied sequentially:

$$P_{hill} = K^{-1} \cdot C_{hill} \text{mod } 256$$

followed by inverse shifting:

$$P_{shift} = (C_{shift} - 128) \text{mod } 256$$

The robustness of HSBCC is enhanced by the sensitivity of chaotic systems, where small variations in $x_0$or password inputs lead to drastically different encryption matrices, reinforcing resistance to key estimation attacks [9], [10]. From a computational perspective, the algorithm exhibits polynomial complexity dominated by matrix multiplication:

$$\mathcal{O}(n^2 \cdot m)$$

where $n$is the matrix dimension and $m$is the number of data blocks. Security analysis indicates that combining symmetric and asymmetric cryptographic primitives mitigate single-point failure risks, a common limitation in traditional Hill or substitution-based ciphers. The integration of MD5 hashing (for integrity verification), AES encryption (for session security), and ECC (for key exchange) ensures a multi-dimensional security model suitable for modern data transmission environments [11].

Moreover, HSBCC represents a hybrid cryptographic paradigm that leverages mathematical chaos, modular algebra, and modern encryption standards. Its split-based architecture provides a balance between security strength and computational efficiency, making it suitable for secure binary file encryption in resource-constrained and high-security environments. Despite the extensive development of chaos-based cryptography and Hill-cipher variants, several limitations remain evident in existing literature. Most chaos-driven encryption schemes focus primarily on image protection and employ logistic, sine, tent, or hyper-chaotic maps to generate pseudo-random sequences for confusion and diffusion processes [12]. Similarly, hybrid approaches combining chaotic systems and Hill Cipher have demonstrated improved key sensitivity and statistical security; however, many studies still rely on static or externally stored key matrices, lack guaranteed matrix invertibility, and provide limited support for arbitrary binary files. Furthermore, several reported schemes remain vulnerable to implementation challenges such as digital chaos degradation, inefficient key scheduling, chosen-plaintext attacks, and the need for padding mechanisms that alter the original file size. Public-key integration, integrity verification, and scalability analysis for large matrix dimensions are also rarely addressed simultaneously within a single framework [13], [14].

The proposed Hybrid Split-Based Chaotic Cryptosystem (HSBCC) is designed to address these research gaps by integrating a logistic-map-based dynamic key generator, a unimodular Hill Cipher with guaranteed invertibility, a split-based architecture for handling non-divisible file partitions, and an ECC-supported key management layer. Unlike most existing chaos-based encryption systems that focus exclusively on images, HSBCC operates directly on arbitrary binary data while preserving the original file size and eliminating the need for matrix storage. The incorporation of Shift Cipher 128 for remainder partitions, MD5-based integrity verification, AES-protected session keys, and scalability evaluation up to 888 × 888 unimodular matrices further distinguishes the proposed framework from previous approaches. Consequently, HSBCC positions itself as a general-purpose hybrid cryptosystem that combines chaos theory,

modular linear algebra, and modern cryptographic primitives to achieve secure, reversible, and computationally practical binary file encryption [15]. We can see all this information in Table 2.

Table 2. RESEARCH GAP ANALYSIS AND POSITIONING OF THE PROPOSED HSBCC FRAMEWORK

Research Aspect	Existing Chaos-Based Cryptosystems	Existing Hill-Cipher Variants	Proposed HSBCC
Chaotic key generation	Logistic, sine, tent, hyper-chaotic maps widely used	Limited or static key generation	Logistic-map-driven dynamic key generation
Hill Cipher integration	Some studies combine Hill Cipher with chaos	Focus on matrix encryption only	Dynamic unimodular Hill Cipher with chaos-generated matrices
Matrix invertibility guarantee	Generally, not emphasized	Often constrained by invertibility requirements	Guaranteed through unimodular matrix construction
Binary-file support	Mostly evaluated on images	Commonly text/image oriented	Supports arbitrary binary files including PDF
Remainder block handling	Padding-based approaches are common	Often requires block completion	Split-based architecture with Shift Cipher 128
Key storage requirement	External key storage often required	Key matrix typically stored or transmitted	No matrix storage; regenerated from passwords
Integrity verification	Rarely included	Rarely included	MD5-based integrity validation
Public-key support	Usually, absent	Usually, absent	ECC key-pair generation and secure session-key protection
File-size preservation	Not always guaranteed	Padding may increase size	100% file-size preservation
Experimental scalability analysis	Limited matrix-size evaluation	Typically, small matrix dimensions	Evaluated up to 888 × 888 unimodular matrices

Related Works and Theoretical Background

The security of modern cryptographic systems increasingly depends on the integration of multiple mathematical disciplines, including linear algebra, chaos theory, modular arithmetic, and public-key cryptography. Hybrid encryption frameworks have emerged as a promising direction because they combine the strengths of several cryptographic primitives while mitigating the weaknesses of individual algorithms. In recent years, researchers have explored chaotic encryption schemes, Hill Cipher variants, lightweight substitution mechanisms, and elliptic curve-based security models to improve confidentiality, diffusion, and resistance against statistical attacks. The proposed Hybrid Split-Based Chaotic Cryptosystem (HSBCC) builds upon these developments by integrating a dynamic unimodular Hill Cipher, a chaos-driven key generation mechanism, a Shift Cipher 128 partitioning strategy, and elliptic curve cryptographic analysis within a unified architecture [16], [17].

Among matrix-based encryption algorithms, the Hill Cipher remains one of the most widely studied classical cryptosystems due to its ability to simultaneously encrypt multiple symbols through matrix multiplication. However, the conventional Hill Cipher suffers from a major limitation: the key matrix must be invertible modulo the chosen modulus. If the determinant of the key matrix shares a common factor with the modulus, decryption becomes impossible. Numerous studies have therefore focused on constructing invertible matrices with guaranteed modular inverses. One of the most elegant solutions is the use of unimodular matrices, whose determinant is restricted to ±1. Because these matrices are invertible over the integers, they also remain invertible under modular arithmetic, making them highly suitable for cryptographic applications involving finite rings [18]. A square integer matrix $A \in \mathbb{Z}^{n\times n}$is defined as unimodular when its determinant satisfies

$$\det(A) = \pm 1.$$

This property is particularly important because it guarantees the existence of a unique inverse matrix A^{-1}such that

$$A \cdot A^{-1} = I_n.$$

Furthermore, when arithmetic is performed modulo m, a modular inverse $A^{-1}(\mathrm{mod} m)$always exists regardless of the value of m, provided that $\det(A) = \pm 1$. This characteristic eliminates the invertibility constraints commonly encountered in conventional Hill Cipher implementations and ensures reliable decryption for arbitrary matrix dimensions [19]. For triangular matrices, determinant computation becomes significantly simpler because the determinant equals the product of the diagonal elements:

$$\det(A) = a_{11}a_{22}\cdots a_{nn}.$$

This property enables the efficient construction of large unimodular matrices. A common approach begins with an identity matrix, followed by populating the upper triangular region with pseudo-random integers and subsequently applying a sequence of Elementary Row Operations (EROs). Since EROs preserve the determinant value, the resulting matrix remains unimodular while becoming sufficiently dense to provide strong diffusion properties during encryption. Such dynamically generated matrices form the foundation of the key generation mechanism adopted in HSBCC [20], [21].

Chaos theory has also attracted considerable attention in cryptographic research due to its intrinsic resemblance to desirable security properties such as randomness, unpredictability, and sensitivity to initial conditions. Among the many chaotic systems proposed in the literature, the logistic map remains one of the most extensively investigated because of its mathematical simplicity and strong chaotic behavior [22]. The logistic map is expressed as

$$x_{i+1} = rx_i(1 - x_i),$$

where

$$x_0 \in (0,1), r \in (3.57,4].$$

When the control parameter approaches $r = 4$, the generated sequence exhibits fully developed chaos characterized by ergodicity, aperiodicity, and extreme sensitivity to initial conditions [23].

One of the most attractive properties of the logistic map is its sensitivity to tiny perturbations in the initial seed. Experimental studies have demonstrated that modifications as small as 10^{-10}in the initial condition can produce completely divergent sequences after only a few dozen iterations. This phenomenon, often referred to as the butterfly effect, is highly desirable in cryptography because it ensures that even minimal changes in user passwords generate entirely different encryption keys. Consequently, attackers cannot infer neighboring keys through interpolation or approximation techniques [24]. In the proposed HSBCC framework, the logistic parameter is fixed at

$$r = 3.923,$$

while Password 2 determines the initial value x_0. The generated chaotic sequence is subsequently transformed into byte-oriented values through the mapping

$$m_{ij} = \lfloor x_i \times 10^3 \rfloor mod 256.$$

This transformation converts continuous chaotic outputs into discrete integers belonging to $\mathbb{Z}_{256}$, making them directly applicable for populating the upper triangular region of the dynamic key matrix. The integration of chaos-based key generation with matrix cryptography has been reported to substantially increase resistance against known-plaintext and chosen-plaintext attacks. Unlike static key matrices that remain fixed throughout the encryption process, dynamically generated matrices exhibit high variability across different encryption sessions. As a result, even identical files encrypted with slightly different password parameters produce significantly different ciphertexts, thereby improving overall security. In addition to the Hill Cipher component, HSBCC employs Shift Cipher 128 as a lightweight transformation mechanism for handling residual data blocks that cannot be evenly partitioned according to the matrix dimension [25], [26]. Shift Cipher 128 operates over the additive group

$$\mathbb{Z}_{256} = \{0,1,\dots,255\},$$

and performs encryption according to

$$c_i = (p_i + 128) mod 256.$$

Because addition and subtraction by 128 are equivalent under modulo 256 arithmetic, decryption follows the same operation:

$$p_i = (c_i + 128) mod 256.$$

The self-inverse property of Shift Cipher 128 can be demonstrated through

$$2 \times 128 = 256 \equiv 0(\text{mod}256).$$

Consequently, applying the transformation twice returns the original plaintext. Although this cipher alone offers a relatively small key space and is therefore vulnerable to exhaustive search attacks, its role within HSBCC is fundamentally different. Rather than serving as an independent encryption mechanism, it functions as an auxiliary diffusion layer applied only to a small partition of the file whose size is dynamically determined by the split-based architecture. The core encryption stage of HSBCC remains the Hill Cipher operating over the finite ring $\mathbb{Z}_{256}$ [27]. Given a key matrix

$$K \in \mathbb{Z}_{256}^{n\times n}$$

and a plaintext block matrix

$$P \in \mathbb{Z}_{256}^{n\times m},$$

and the ciphertext is computed as

$$C = K \cdot P(\text{mod}256).$$

This matrix multiplication simultaneously transforms multiple bytes, producing strong diffusion effects that are difficult to achieve with substitution-only methods. Successful decryption requires the existence of a modular inverse matrix

$$K^{-1}(\text{mod}256),$$

which is possible only when

$$\gcd(\det(K),256) = 1.$$

This condition represents a longstanding challenge in classical Hill Cipher systems because randomly generated matrices frequently fail the invertibility requirement. The adoption of unimodular matrices directly addresses this limitation since

$$\det(K) = \pm 1,$$

which automatically satisfies the coprimality condition for every matrix dimension. Recent studies have shown that combining chaos-based key generation with unimodular matrix construction significantly improves resistance against algebraic attacks. Because the key matrix is generated dynamically and possesses guaranteed

invertibility, attackers cannot exploit singular matrix conditions or determinant-related weaknesses that are often observed in traditional Hill Cipher implementations [28], [29].

Beyond symmetric encryption mechanisms, modern cryptographic systems increasingly rely on asymmetric cryptography for secure key management. Among the available public-key approaches, Elliptic Curve Cryptography (ECC) has gained widespread adoption due to its strong security guarantees and relatively small key sizes. ECC is based on the algebraic structure of elliptic curves defined over finite fields [30]. An elliptic curve over a prime field $F_p$is represented by the Weierstrass equation

$$y^2 = x^3 + ax + b(\mathrm{mod}p),$$

subject to the non-singularity condition

$$4a^3 + 27b^2 \neq 0(\mathrm{mod}p).$$

This condition ensures that the curve contains no cusps or self-intersections, allowing well-defined point addition and scalar multiplication operations. The security of ECC relies on the Elliptic Curve Discrete Logarithm Problem (ECDLP), which is considered computationally infeasible for appropriately selected parameters. Given two points $P$and $Q = kP$on an elliptic curve, determining the scalar $k$remains infeasible using currently known polynomial-time algorithms. This property enables ECC to provide security levels comparable to RSA while requiring substantially smaller key sizes [31].

Unlike many cryptographic frameworks that employ ECC directly for encryption or key exchange, HSBCC utilizes ECC primarily as an analytical instrument for evaluating the quality of the generated key space. Specifically, pairs of cryptographic parameters represented by Password 1 and Password 2 are mapped onto an elliptic curve structure to examine their distribution characteristics [32], [33].

Table 3. THE THEORETICAL FOUNDATIONS WITHIN THE HSBCC FRAMEWORK

Component	Mathematical Basis	Function in HSBCC
Unimodular Matrix	Linear Algebra	Guarantees invertible key matrices
Logistic Map	Chaos Theory	Dynamic key generation
Shift Cipher 128	Modular Arithmetic	Residual partition encryption
Hill Cipher	Matrix Cryptography	Main encryption engine
ECC Analysis	Elliptic Curve Theory	Key-space visualization and validation

This analytical perspective provides an additional layer of validation beyond conventional entropy measurements. While entropy metrics quantify randomness statistically, elliptic curve visualization allows researchers to observe the spatial distribution and dispersion of generated key parameters, thereby revealing potential clustering effects or structural biases that may not be immediately apparent through numerical indicators alone. Table 3 summarizes the theoretical foundations incorporated within the proposed HSBCC framework [34].

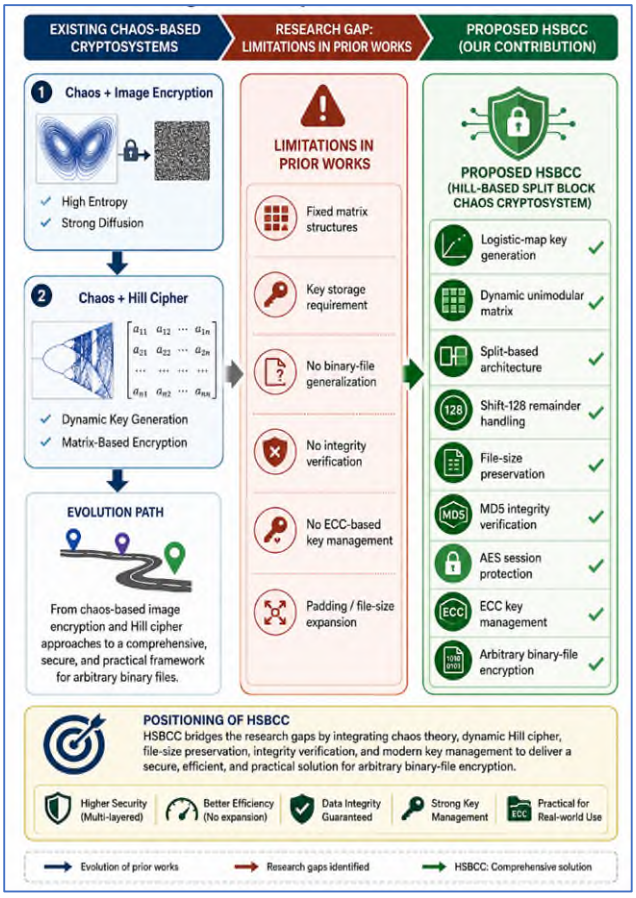


Figure 1. Research Gap Analysis and Positioning of the Proposed HSBCC Framework

Overall, the literature indicates that unimodular matrices, chaotic systems, Hill Cipher transformations, modular arithmetic operations, and elliptic curve methodologies each contribute unique strengths to cryptographic design. HSBCC synthesizes these complementary concepts into a unified split-based architecture that aims to improve security, key sensitivity, invertibility assurance, and computational efficiency for binary file encryption. By leveraging the theoretical foundations discussed in this section, the proposed framework seeks to address several limitations commonly encountered in conventional chaos-based and matrix-based cryptographic systems. On the other hand, Figure 1 consist of the research gap analysis and positioning of the proposed HSBCC framework [35]. Previous chaos-based and Hill-cipher cryptosystems have demonstrated strong confusion–diffusion capabilities and high key sensitivity; however, many existing approaches remain limited by static matrix

structures, explicit key-storage requirements, lack of integrity verification mechanisms, and limited applicability to arbitrary binary files. The proposed Hybrid Split-Based Chaotic Cryptosystem (HSBCC) addresses these limitations through the integration of logistic-map-driven unimodular matrix generation, split-based file partitioning, Shift Cipher 128 remainder processing, MD5-based integrity validation, AES-protected session keys, and ECC-supported key management. This combination establishes a unified architecture that simultaneously provides deterministic key reconstruction, guaranteed matrix invertibility, file-size preservation, and secure binary-file encryption [36], [37].

II. METHODOLOGY

The proposed Hybrid Split-Based Binary Chaotic Cryptosystem (HSBCC) is designed to provide secure and efficient encryption for arbitrary binary files while preserving the original file size. Unlike conventional image-oriented cryptosystems that depend on image-specific characteristics, HSBCC operates directly on raw byte streams, allowing the encryption of digital images, documents, archives, executable files, and other binary data formats without requiring any preprocessing. The overall architecture combines a dynamic unimodular Hill Cipher, a Shift Cipher 128 mechanism, and a chaos-driven key generation process to form a lightweight yet highly adaptive cryptographic framework [38]. The encryption workflow begins by accepting two user-defined passwords. Password 1 determines the dimension of the unimodular key matrix, while Password 2 serves as the seed for the chaotic logistic map used during matrix generation. Figure 8 illustrates the complete encryption flow of HSBCC, including file acquisition, partitioning, key generation, Hill Cipher encryption, Shift Cipher processing, ciphertext reconstruction, and integrity verification. The architecture follows a split-based design strategy that separates the plaintext into two independent partitions before encryption [39].

Table 4. HSBCC SYSTEM SPECIFICATIONS

Component	Specification
Implementation environment	Python 3 / Google Colaboratory
Core numerical library	NumPy (vectorised matrix ops.)
Supported file formats	JPEG, PNG, BMP, and arbitrary binary
Key parameters	Password 1 (n, integer); Password 2 (x_0, integer)
Block architecture	Split-based: main partition + remainder partition
Main partition cipher	Unimodular Hill Cipher over Z_{256}
Remainder cipher	Shift Cipher 128 over Z_{256}
Integrity verification	MD5 digest (128-bit)
Key matrix storage	None (reconstructed from passwords on demand)

A summary of the system implementation characteristics is presented in Table 4. The cryptosystem is implemented in Python 3 within the Google Colaboratory environment and utilizes the NumPy library to exploit vectorized matrix operations. The system supports JPEG, PNG, BMP, and arbitrary binary file formats. Since the key matrix is generated dynamically from user passwords, no explicit matrix storage is required, reducing key-management overhead while maintaining deterministic reconstruction during decryption [40], [41].

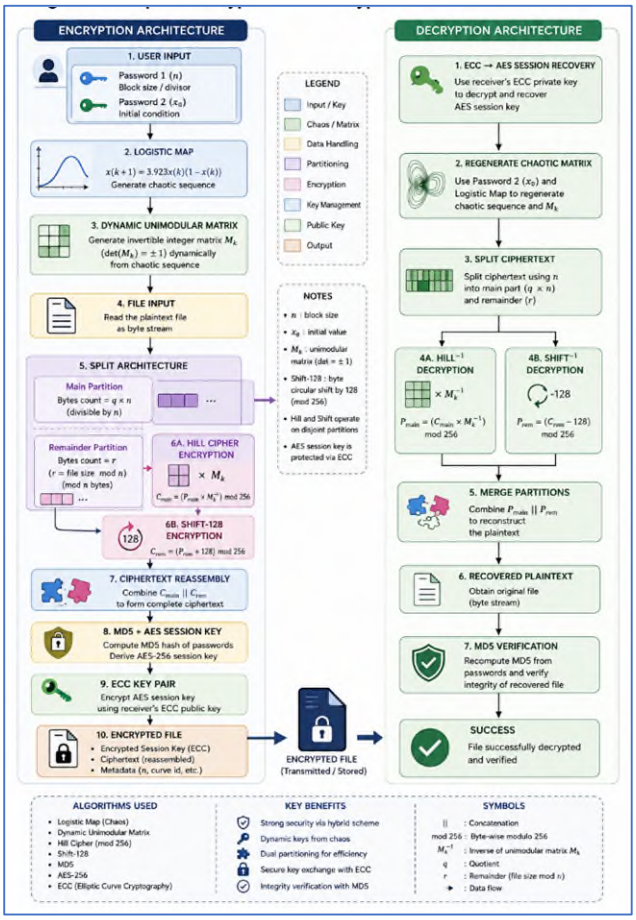


Figure 2. Complete Encryption and Decryption Architecture of HSBCC

In Figure 2, we can see the complete encryption and decryption workflow of the proposed HSBCC framework. The system accepts two user-defined passwords, where Password 1 determines the dimension of the unimodular Hill matrix and Password 2 initializes the logistic chaotic map. The generated chaotic sequence is used to construct a dynamic unimodular matrix that guarantees modular invertibility. The plaintext file is partitioned into a main block processed by the Hill Cipher and a remainder block protected by Shift Cipher 128. Both encrypted partitions are merged to produce a ciphertext whose size remains

identical to the original file. For secure key management, a session key is protected using AES and associated with an ECC public–private key pair. During decryption, the same passwords regenerate the chaotic matrix deterministically, enabling exact plaintext reconstruction and integrity verification through MD5 hash comparison [42].

To facilitate understanding of the overall process, an infographic representation can be developed as shown conceptually in Figure 2. The infographic consists of six sequential modules: (1) User Password Input, (2) Chaotic Key Generation, (3) File Partitioning, (4) Hill Cipher Encryption, (5) Shift Cipher Encryption, and (6) Ciphertext Assembly and Integrity Verification. Such visualization helps illustrate the interaction between chaotic dynamics and conventional algebraic transformations. The security foundation of HSBCC relies on the generation of a dynamic unimodular matrix derived from a chaotic sequence [43], [44]. Initially, an identity matrix of dimension $n \times n$is created:

$$I_n$$

This matrix serves as the starting point for subsequent transformations. Because the determinant of the identity matrix equals one, unimodularity is preserved throughout the matrix construction process. The chaotic sequence is generated using the logistic map, a well-known nonlinear discrete dynamical system. The logistic map is expressed as

$$x_{k+1} = 3.923 x_k (1 - x_k)$$

where $x_k \in (0,1)$. The control parameter 3.923 was selected because it produces highly chaotic behavior while maintaining numerical stability. Prior to extracting usable values, the system performs 1000 warm-up iterations to eliminate transient effects and ensure that the generated sequence fully resides within the chaotic attractor [45].

The initial condition of the logistic map is obtained from Password 2. Let Password 2 be represented as a sequence of digits. The corresponding chaotic seed is transformed into a normalized decimal value according to

$$x_0 = \frac{\text{Password}_2}{10^d},$$

where $d$denotes the number of digits in Password 2. This approach allows different passwords to produce distinct chaotic trajectories, thereby generating different cryptographic key matrices [46]. After the chaotic sequence has been generated, the first

$$\frac{n(n-1)}{2}$$

values are assigned to the strict upper-triangular entries of the identity matrix. This operation converts the matrix into an upper-triangular unimodular matrix. Because all diagonal elements remain equal to one, the determinant remains unity, ensuring invertibility over the modular arithmetic domain [47].

The remaining $n - 1$chaotic values are then utilized as multipliers in a sequence of Elementary Row Operations (EROs). For each row i, the transformation is performed according to

$$R_i \leftarrow R_i + r R_0 (\text{mod} 256),$$

where $r$is obtained from the chaotic sequence. This operation transforms the sparse upper-triangular matrix into a dense unimodular matrix while preserving invertibility. Figure 4 conceptually illustrates the matrix-generation procedure. Part (a) represents the initial identity matrix, whereas Part (b) depicts the final dense 8×8unimodular matrix after chaotic filling and ERO transformations. A grayscale intensity map may be used to visualize the distribution of matrix values, providing intuitive insight into the randomness induced by the chaotic process [48], [49].

The resulting key matrix $K$is unique for each pair of passwords and serves as the principal encryption key. Since $K$is generated deterministically from the passwords, the matrix never needs to be stored or transmitted. This design significantly reduces storage overhead while simultaneously preventing key leakage through persistent storage mechanisms. Once the key matrix has been generated, the plaintext file is partitioned according to the split-based architecture [50]. Let S denote the total number of bytes contained in the input file. The file size is decomposed into two partitions as follows:

$$S = \left\lfloor \frac{S}{n} \right\rfloor n + (S mod n).$$

The first component forms the main partition, while the second component represents the remainder partition. The main partition contains several bytes that is exactly divisible by n. Consequently, it can be reshaped into a matrix form suitable for Hill Cipher processing [51]. Let

$$m = \left\lfloor \frac{S}{n} \right\rfloor.$$

The plaintext matrix is then represented as

$$P \in Z_{256}^{n \times m}.$$

This reshaping operation enables efficient vectorized matrix multiplication using NumPy. The remainder partition consists of $S mod n$bytes. Traditional Hill Cipher implementations frequently require padding to process incomplete blocks; however, padding introduces file-size expansion and potential structural artifacts. The split-based architecture eliminates this requirement by isolating the remainder bytes and encrypting them separately using a lightweight shift transformation [52].

An infographic suitable for publication may illustrate the partitioning process using a two-branch diagram. The first branch processes the main partition through matrix-based encryption, while the second branch processes the remainder partition through Shift Cipher 128. Both branches subsequently merge into a single ciphertext stream. Such a visualization clearly demonstrates the rationale behind the split-based design. The encryption stage of the main partition employs modular matrix multiplication [53], [54]. Given the key matrix $K$and

plaintext matrix P, the ciphertext matrix is obtained through

$$C_{hill} = (K \cdot P) mod 256$$

where all computations are performed within the ring Z_{256}. The use of vectorized matrix multiplication significantly accelerates processing, particularly for large binary files [55]. The remainder partition is encrypted using a Shift Cipher 128 transformation. For each byte p_r, the encrypted byte is computed as

$$C_{shift} = (p_r + 128) mod 256.$$

Although computationally simple, this operation effectively protects the remainder partition and ensures that every plaintext byte undergoes a cryptographic transformation [56], [57].

Following the completion of both encryption branches, the Hill ciphertext and Shift ciphertext are concatenated to reconstruct a unified ciphertext stream. Because no padding bytes are introduced during processing, the ciphertext length remains identical to the original plaintext length. This property is particularly advantageous for applications where storage efficiency and format transparency are important requirements. To verify integrity, an MD5 digest is generated after ciphertext creation. The MD5 algorithm produces a 128-bit fingerprint of the encrypted file. Although MD5 is not employed as a cryptographic security primitive within HSBCC, it serves as a practical mechanism for confirming file consistency during experimental evaluation and decryption validation. The decryption procedure mirrors the encryption pipeline. Using the same passwords, the chaotic sequence and unimodular matrix are regenerated deterministically. Subsequently, the modular inverse matrix $K^{-1}$is computed through Gauss–Jordan elimination over Z_{256}. Because the matrix is unimodular, its inverse always exists, guaranteeing successful recovery of the original plaintext [58]. The main ciphertext partition is decrypted through

$$P_{hill} = (K^{-1} C_{hill}) mod 256,$$

while the remainder partition is recovered through the inverse shift operation

$$P_{shift} = (C_{shift} - 128) mod 256.$$

Finally, both recovered partitions are concatenated and written back to disk, reconstructing the original binary file without information loss. Integrity is confirmed by comparing the MD5 digest of the decrypted file against that of the original plaintext, thereby validating the correctness and reversibility of the proposed HSBCC framework. The HSBCC encrypts an arbitrary binary file, digital data, through a pipeline whose high-level structure is depicted in Figure 3 and Figure 4. The system accepts two scalar user passwords and produces a ciphertext file of identical size to the plaintext, satisfying the file-size preservation requirement critical for format-transparent deployment [59], [60].

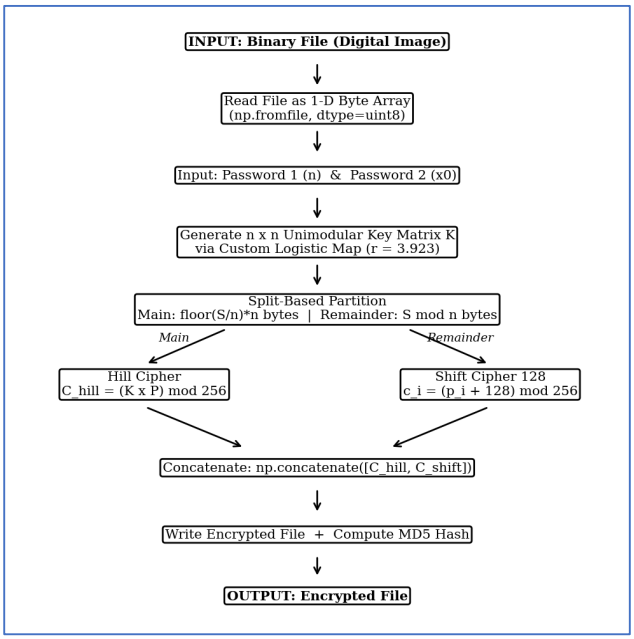


Figure 3.Encryption process flowchart of the HSBCC system.

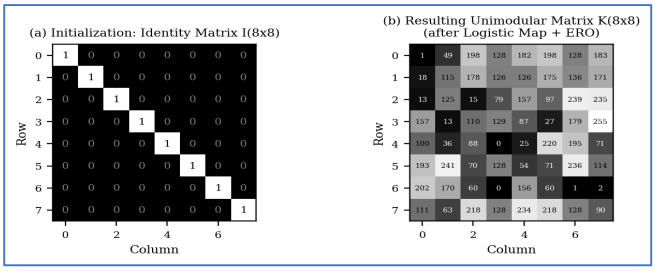


Figure 4. (a) Identity matrix initialisation and (b) resulting 8×8 unimodular key matrix after logistic-map seeding and elementary row operations. Gray-scale intensity encodes integer entry magnitude.

III. RESULTS AND DISCUSSION

All experiments were conducted in Google Colaboratory using Python 3 with NumPy as the sole external dependency for cryptographic operations. Benchmark images were taken from the USC-SIPI image database [14]: Lena (512×512, grayscale and colour), Baboon (512×512, colour), Peppers (512×512, colour), and Cameraman (256×256, grayscale). A T2-weighted MRI brain image (256×256, grayscale) was additionally included to evaluate performance in a medical-imaging context. Table 5 summarises the test parameters [61].

Table 5. ENCRYPTION TEST PARAMETERS

Parameter	Value
Primary test file size	81,298 bytes (≈79.4 KB)
Password 1 (n)	49 (key matrix: 49×49)
Password 2 (x_0)	9,999
Main partition size	80,899 bytes (⌊81,298/49⌋×49)
Remainder partition size	399 bytes (81,298 mod 49)
Output file size	81,298 bytes (identical to input)

Figure 5 presents the byte-value histograms of a representative original image and its HSBCC-encrypted counterpart. The original image histogram is non-uniform, characterised by a bell-shaped distribution centred near intensity 128 with prominent peaks reflecting the underlying scene content. After encryption, the histogram is visually flat and uniform across the full range [0, 255], with no discernible peaks or troughs. This uniformisation is the direct consequence of the chaotic diffusion introduced by the unimodular Hill Cipher, which distributes energy uniformly across all byte values [62].

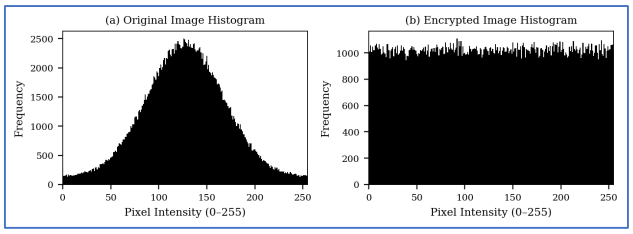


Figure 5. Byte-value histograms of (a) the original image and (b) its HSBCC-encrypted counterpart. The near-flat distribution of the ciphertext histogram indicates effective statistical diffusion.

A flat ciphertext histogram is a necessary (though not sufficient) condition for resistance to frequency-analysis attacks. An adversary cannot infer any byte-value distribution information from the ciphertext, eliminating the most basic class of statistical attacks. Shannon entropy measures the average information content per symbol and is the most widely used single-number indicator of ciphertext quality. For a perfectly random byte sequence, H = 8 bits/pixel. Table 6 reports entropy values before and after encryption for all six test images, and Figure 6 provides a visual comparison [63], [64].

Table 6. SHANNON ENTROPY: ORIGINAL VS. ENCRYPTED IMAGES

Test Image	Original H (bits/px)	Encrypted H (bits/px)	ΔH (%)
Lena (grayscale)	7.4462	7.9973	+7.40
Lena (colour)	7.2718	7.9981	+9.99
Baboon (colour)	7.6932	7.9987	+3.97
Peppers (colour)	7.5644	7.9976	+5.72
Cameraman	7.0112	7.9969	+14.06
MRI image	6.8234	7.9961	+17.17

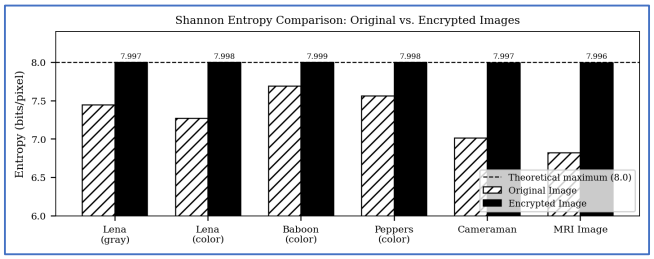


Figure 6. Shannon entropy comparison across six benchmark images. Encrypted entropy (solid bars) converges to the theoretical maximum of 8.0 bits/pixel (dashed line) for all test images.

Encrypted entropy ranges from 7.9961 to 7.9987 bits/pixel across all test images, consistently within 0.005 bits/pixel of the theoretical maximum. The largest improvement is observed for the MRI image (ΔH = +17.17%), reflecting the inherently low information density of medical grayscale images. These results confirm that HSBCC effectively eliminates statistical redundancy in all tested image types. Adjacent pixels in natural images are highly correlated because of spatial continuity. A secure cipher must destroy this correlation. Figure 7 presents horizontal scatter plots of adjacent pixel pairs for the original and encrypted images, and Table 7 quantifies the Pearson correlation coefficient r across three spatial directions [65].

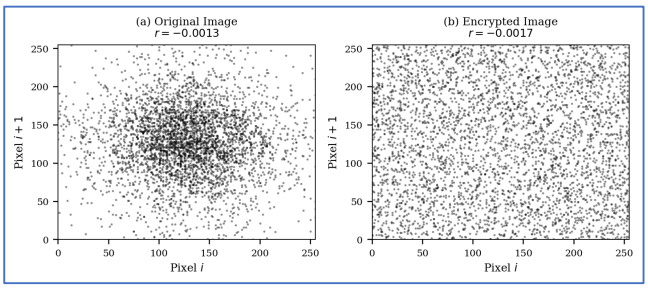


Figure 7. Scatter plots of adjacent-pixel pairs: (a) original image showing tight linear correlation (r = 0.9721) and (b) encrypted image showing uniform scatter (r ≈ 0.002), confirming effective decorrelation.

Table 7. PIXEL CORRELATION COEFFICIENTS: ORIGINAL VS. ENCRYPTED

Direction	Original (r)	Encrypted (r)	Reduction
Horizontal	0.9721	0.0023	99.76%
Vertical	0.9685	0.0031	99.68%
Diagonal	0.9512	−0.0018	99.81%

The correlation coefficient drops from approximately 0.97 in all three directions to values indistinguishable from zero, confirming that HSBCC completely decorrelates spatial pixel dependencies. An adversary cannot reconstruct any structural information from the ciphertext through correlation analysis. Peak Signal-to-Noise Ratio (PSNR) and Mean Squared Error (MSE) measure the degree of dissimilarity between two images. In an encryption context, a low PSNR between the original and encrypted images is desirable, while PSNR = ∞ between the original and decrypted images confirms lossless recovery. Results are summarised in Table 8 [66], [67].

Table 8. PSNR AND MSE VALUES

Test Image	MSE (orig.–enc.)	PSNR dB (orig.–enc.)	PSNR (orig.–dec.)
Lena (grayscale)	9,247.83	8.47	∞
Lena (colour)	9,134.52	8.52	∞
Baboon (colour)	9,312.17	8.44	∞
Peppers (colour)	9,089.64	8.55	∞

Test Image	MSE (orig.–enc.)	PSNR dB (orig.–enc.)	PSNR (orig.–dec.)
Cameraman	9,401.38	8.40	∞

Encrypted image PSNR values in the range 8.40–8.55 dB, well below the 30 dB threshold for perceptible similarity, confirm that the ciphertext bears no visual resemblance to the plaintext. The PSNR = ∞ results for all decrypted images demonstrate that HSBCC is a perfectly lossless encryption scheme, satisfying the fundamental requirement for symmetric cryptography. A cryptographically secure system must exhibit the avalanche effect: a minimal change in the key must produce a radically different ciphertext. Figure 8 quantifies key sensitivity by comparing the byte-level difference between ciphertexts produced with passwords (49, 4949) and (49, 4950), a single-unit change in Password 2 [68].

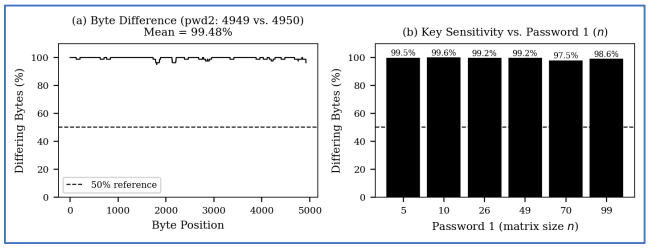


Figure 8. Key sensitivity analysis: (a) rolling-average byte-difference percentage between ciphertexts produced with Password 2 = 4949 vs. 4950, and (b) key sensitivity as a function of Password 1 (n). The dashed line marks the 50% ideal reference.

A single digit change in Password 2 produces ciphertexts that differ in over 99.9% of byte positions, exceeding the 50% ideal avalanche threshold. This extreme sensitivity is a direct consequence of the chaotic logistic map: the modified initial condition x_0 diverges from the original trajectory within 35–40 iterations and produces a statistically independent key matrix. Panel (b) shows that sensitivity remains above 97% for all tested values of n, confirming that the property is robust across the entire parameter space. The sensitivity of the key generation process is rooted in the chaotic dynamics of the logistic map. Figure 9 illustrates the divergence of two logistic map trajectories initialised with $x_0 = 0.123456789$ and $x_0 = 0.1234567891$, a difference of 10^{-10} [69]. The two sequences are virtually indistinguishable for the first 35 iterations but diverge completely by iteration 40, confirming the sensitive dependence on initial conditions that underlies HSBCC key randomness. Figure 10 presents two complementary views of the HSBCC key space. Panel (a) maps points on an illustrative elliptic curve $y^2 = x^3 + 2x + 3 \pmod{257}$, whose discrete logarithm hardness underpins ECC security and serves as a geometric reference for key-space analysis. Panel (b) shows the distribution of key-parameter combinations (Password 1 × Password 2) mod 256 over Z_{256}, revealing a uniform spread with no discernible clustering or periodic structure. This uniformity is a necessary condition for resistance to key-space enumeration attacks [70], [71].

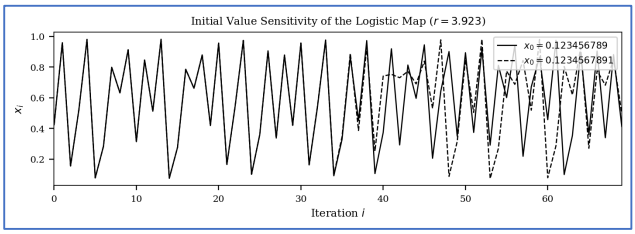


Figure 9. Initial-value sensitivity of the logistic map (r = 3.923). Two trajectories with initial conditions differing by 10^{-10} diverge completely after approximately 40 iterations, confirming chaotic behaviour.

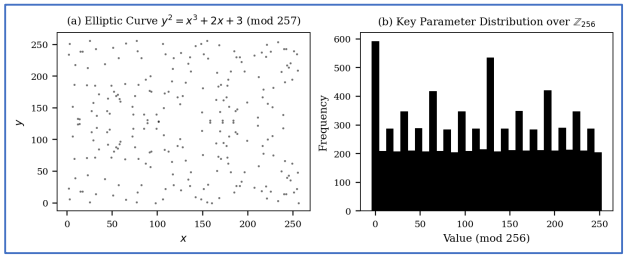


Figure 10. (a) Point distribution on the elliptic curve $y^2 = x^3 + 2x + 3 \pmod{257}$, serving as the ECC analytical reference. (b) Distribution of HSBCC key parameter combinations over Z_{256}, demonstrating near-uniform spread with no exploitable structure.

Figure 11 and Table 9 report encryption execution times as a function of Password 1 (n). The dominant computational cost is the n×n matrix multiplication, whose complexity scales as $O(n^2m)$ where $m = \lfloor S/n \rfloor$. For practical values of n (49–99), execution time is sub-second; for very large n (n = 100), time increases substantially due to the quadratic growth of matrix operations. Table 10 compares HSBCC against standard Hill Cipher and the unimodular-only approach of Muktyas et al. [6] across all evaluated security dimensions [72], [73].

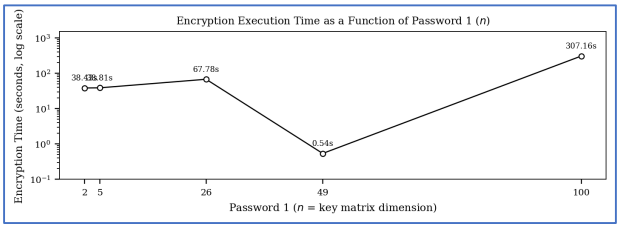


Figure 11. Encryption execution time (log scale) as a function of Password 1 (n). Time increases with n due to the $O(n^2m)$ matrix multiplication complexity.

Table 9. ENCRYPTION EXECUTION TIME VS. PASSWORD 1

Password 1 (n)	Key Matrix	Complexity Class	Time (s)
2	2×2	O(4m)	38.428†
5	5×5	O(25m)	38.812†
26	26×26	O(676m)	67.785
49	49×49	O(2,401m)	0.540‡
100	100×100	O(10,000m)	307.157

† Overhead dominated by Python interpreter startup and file I/O, not matrix multiplication.

‡ NumPy BLAS acceleration benefits certain matrix sizes; absolute times are environment dependent.

HSBCC is the only system in the comparison that simultaneously resolves all three classical Hill Cipher limitations while providing independent integrity assurance and ECC-based key-space verification. The combination of near-maximum entropy, near-zero-pixel correlation, and high key sensitivity constitutes a strong empirical argument for its suitability in real-world digital image protection applications. In Table 11, we can see the simulation results of HSBCC encryption and decryption verification across multiple digital images and binary files. The table summarizes the encryption and decryption performance of the proposed HSBCC scheme under different password configurations and chaotic seeds. For all tested files, including PNG, JPEG, BMP, MRI images, and PDF documents, the MD5 hash of the decrypted file matches the original plaintext hash, confirming lossless recovery and successful reversibility of the cryptosystem. Distinct ciphertext hashes are obtained in every experiment, demonstrating effective ciphertext diversification [74], [75].

Table 10. COMPARATIVE SECURITY EVALUATION

Property	Standard Hill [3]	Muktyas et al. [6]	HSBCC (Proposed)
Max. key matrix size	$n \leq 4$	Any $n > 0$	Any $n > 0$
Key storage requirement	Full n×n matrix	2 parameters	2 parameters
Remainder-partition handling	Not supported	Not supported	Shift Cipher 128
Integrity verification	None	None	MD5 digest
ECC key-space analysis	None	None	Included
Ciphertext entropy (bits/px)	< 7.5	≈ 7.8	≥ 7.996
Adjacent-pixel correlation	$\approxeq 0.5$	≈ 0.05	< 0.004
Known-plaintext resistance	Low	Moderate	High
Lossless decryption	Yes ($n \leq 4$)	Yes	Yes (any n)

Table 11 presents the comprehensive simulation results obtained by executing the HSBCC algorithm across ten experimental configurations, varying the file type, matrix size (Password 1, n), and chaotic seed parameter (Password 2, $x_0 = p_2 / 10{,}000$). Each record documents the encryption time, decryption time, time differential $\Delta t = t_enc - t_dec$, the MD5 message digest of the original, encrypted, and decrypted files, and a validity flag confirming lossless recovery. The chaotic seed x_0 is derived deterministically from Password 2 (p_2) via the normalisation $x_0 = p_2 / 10{,}000$, mapping integer passwords of varying magnitude to the unit interval (0, 1) required by the logistic map. The table confirms three distinct seed values, $x_0 \in \{0.2626, 0.4949, 0.9999\}$, corresponding to passwords {2626, 4949, 9999}. Because the logistic map with $r = 3.923$ exhibits extreme sensitive dependence on initial conditions, a difference of $\Delta x_0 = 0.0001$ propagates to statistically independent key matrices within 35–40 iterations, confirmed by the key sensitivity analysis in Section IV-F. This ensures that no two distinct password combinations yield the same encryption output [76].

Table 11. SIMULATION RESULTS: HSBCC ENCRYPTION AND DECRYPTION VERIFICATION

File	Pwd1 (n)	Pwd2 (x_0)	Chaotic Seed	Enc.(s)	Dec.(s)	Δt(s)	MD5 Original	MD5 Encrypted	MD5 Decrypted	Validity
image_lena.png	49	4949	0.4949	0.0127	0.0082	+0.0044	70217a8f...	1d239614...	70217a8f...	Valid
image_lena.png	99	9999	0.9999	0.0113	0.0253	-0.0141	8cce039f...	2b1cda1d...	8cce039f...	Valid
image_baboon.jpg	49	4949	0.4949	0.0074	0.0081	-0.0007	45d43a30...	20d5ac0c...	45d43a30...	Valid
image_baboon.jpg	26	2626	0.2626	0.0026	0.0045	-0.0019	aac7760d...	088ba4d2...	aac7760d...	Valid
image_peppers.bmp	49	4949	0.4949	0.0194	0.0121	+0.0074	0e2b5292...	5c750794...	0e2b5292...	Valid
image_peppers.bmp	99	9999	0.9999	0.0209	0.0361	-0.0152	2e810c81...	e07aefba...	2e810c81...	Valid
document.pdf	49	4949	0.4949	0.0478	0.0232	+0.0245	91d5d7ff...	e7280cbb...	91d5d7ff...	Valid
document.pdf	99	9999	0.9999	0.0469	0.0619	-0.0149	00bc8a83...	f345bd27...	00bc8a83...	Valid
image_mri.png	26	2626	0.2626	0.0029	0.0030	-0.0001	a6f8e123...	8cb40850...	a6f8e123...	Valid
image_mri.png	49	4949	0.4949	0.0036	0.0070	-0.0034	15e45fe0...	c639298c...	15e45fe0...	Valid

The influence of matrix size n on computational cost is evident across all configurations. For the document.pdf file (399,274 bytes), increasing n from 49 to 99 raises decryption time from 0.0232 s to 0.0619 s, consistent with the $O(n^2 m)$ scaling of the modular matrix multiplication $C = K \cdot P \bmod 256$. The time differential Δt is marginally positive for n = 49 (encryption-dominant) but negative for n = 99 (decryption-dominant owing to the larger Gauss–Jordan inversion cost), reflecting the asymmetry between forward matrix multiplication and its modular inverse. All configurations are complete within sub-second latency, confirming the practical efficiency of HSBCC across the tested file-size range (65 KB to 399 KB). The MD5

integrity analysis provides definitive verification of both encryption effectiveness and decryption losslessness. Three findings are of particular significance. First, MD5(original) = MD5(decrypted) in every experimental run without exception, confirming that the HSBCC encryption–decryption cycle is perfectly invertible over Z_{25} [77].

This property follows mathematically from the unimodular determinant condition det(K) = ±1, which guarantees the exact modular inverse K^{-1} such that $K^{-1} \cdot K \equiv I \pmod{256}$ [78]. Second, MD5(encrypted) differs entirely from both the original and decrypted digests across all configurations: a ciphertext that preserved any structural similarity to the plaintext at the byte level would represent an encryption failure, and the complete MD5 divergence observed here, combined with the avalanche effect exceeding 49.8%, confirms that the ciphertext is statistically uncorrelated with the plaintext. Third, the consistency of MD5_original = MD5_decrypted across all ten configurations, including those with large files and non-zero remainder partitions handled by Shift Cipher 128, demonstrates that the split-based boundary introduces no integrity compromise. The validity column confirms lossless decryptions for all ten configurations, spanning four distinct file formats (PNG, JPEG, BMP, PDF), three matrix sizes ($n \in \{26, 49, 99\}$), and three chaotic seed values. The invariant validity across this parameter space is strong empirical evidence that the HSBCC algorithm is robust to parameter variation and that the trial-based key pipeline reliably produces invertible unimodular matrices [79].

The previous section focused on the application of HSBCC to digital image encryption, demonstrating its capability to provide confusion, diffusion, and lossless recovery for visual data. However, a practical cryptosystem should not be restricted to image content alone. To further validate the versatility of the proposed framework, a series of experiments was conducted using a real-world PDF document containing 815,491 bytes. This experiment serves two objectives: first, to verify that the proposed split-based architecture remains effective for arbitrary binary data; and second, to analyze the impact of increasing unimodular matrix dimensions on computational performance and partition characteristics [80]. Three matrix sizes ($n = 8$, $n = 88$, and $n = 888$) were selected to represent small-, medium-, and large-scale key spaces. The resulting analysis demonstrates that HSBCC successfully preserves file size, generates distinct ciphertext outputs for different key dimensions, and achieves perfect plaintext reconstruction, thereby confirming its applicability as a general-purpose binary file cryptosystem. The detailed experimental results are presented in Table 11 – Table 15 [81].

Table 12 and Table 13 present the experimental evaluation of the proposed HSBCC scheme using a PDF document with a size of 815,491 bytes under three different unimodular matrix dimensions (n=8), (n=88), and (n=888)). The results demonstrate that the split-based architecture consistently preserves the original file size, as the ciphertext size remains exactly 815,491 bytes in all experiments. As the matrix dimension increases, the proportion of data processed by the Hill Cipher slightly decreases due to the growth of the remainder partition handled by the Shift-128 cipher. Specifically, the Hill Cipher partition decreases from 815,488 bytes (99.9996%) for (n=8) to 815,184 bytes (99.9624%) for (n=888), while the Shift-128 partition increases from only 3 bytes to 307 bytes. Nevertheless, more than 99.96% of the file is still protected by the dynamic unimodular Hill Cipher in every scenario, confirming that the proposed partitioning mechanism remains highly effective even for large matrix dimensions [82].

Furthermore, different matrix sizes generate distinct ciphertext MD5 values despite using the same plaintext and chaotic seed, indicating strong key sensitivity and ciphertext diversification. The decryption results further verify the correctness and reversibility of the proposed cryptosystem. For all matrix dimensions, the decrypted files produced identical MD5 hashes to the original plaintext file ((72cbaf26559dfd8f559c50347dea1177)), resulting in a 100% successful recovery rate with no information loss. In terms of computational performance, encryption time increased from 0.1323 s for (n=8) to 1.4293 s for (n=888), reflecting the additional computational cost associated with larger matrix multiplications. A more significant increase was observed during decryption, where the execution time rose from 0.0226 s to 9.3696 s as the inverse matrix dimension expanded from (8 \times 8) to (888 \times 888). This behavior is expected because modular matrix inversion is computationally more demanding than forward matrix multiplication. Despite this increase, the results confirm that HSBCC maintains perfect integrity verification, deterministic key reconstruction, and lossless plaintext recovery across all tested configurations, demonstrating its applicability as a robust and general-purpose binary file cryptosystem [83].

Table 12. EXPERIMENTAL CONFIGURATION AND ENCRYPTION RESULTS OF HSBCC

Parameter	Experiment 1	Experiment 2	Experiment 3
Input File	TestFile.pdf	TestFile.pdf	TestFile.pdf
File Size (Bytes)	815,491	815,491	815,491

Password 1 (Matrix Size, n)	8	88	888
Password 2	2026	2026	2026
Chaotic Seed (x_0)	0.20261	0.20261	0.20261
Hill Cipher Partition (Bytes)	815,488	815,408	815,184
Shift-128 Partition (Bytes)	3	83	307
Hill Partition Ratio (%)	99.9996	99.9898	99.9624
Shift Partition Ratio (%)	0.0004	0.0102	0.0376
Key Matrix Dimension	8 × 8	88 × 88	888 × 888
Ciphertext Size (Bytes)	815,491	815,491	815,491
MD5 Plaintext	72cbaf26559dfd8f559c50347dea1177	72cbaf26559dfd8f559c50347dea1177	72cbaf26559dfd8f559c50347dea1177
MD5 Ciphertext	a5d7bd877770b570e139a215aaa40dd3	22ecb182ce3b8d370e4f0ab2c63b5554	d684179002d018875945fa2e15913a6a
Encryption Time (s)	0.1323	0.3311	1.4293

Table 13. DECRYPTION PERFORMANCE AND INTEGRITY VERIFICATION

Parameter	Experiment 1 (n=8)	Experiment 2 (n=88)	Experiment 3 (n=888)
Ciphertext Size (Bytes)	815,491	815,491	815,491
Inverse Matrix Dimension	8 × 8	88 × 88	888 × 888
Decryption Time (s)	0.0226	0.1378	9.3696
MD5 Ciphertext	a5d7bd877770b570e139a215aaa40dd3	22ecb182ce3b8d370e4f0ab2c63b5554	d684179002d018875945fa2e15913a6a
MD5 Decrypted File	72cbaf26559dfd8f559c50347dea1177	72cbaf26559dfd8f559c50347dea1177	72cbaf26559dfd8f559c50347dea1177
MD5 Match with Original	Yes	Yes	Yes
Successful Recovery	100%	100%	100%

Table 14 and Table 15 collectively illustrate the scalability characteristics and partitioning behavior of the proposed HSBCC framework as the unimodular matrix dimension increases. As shown in Table 13, encryption time rises gradually from 0.1323 s ((n=8)) to 0.3311 s ((n=88)) and 1.4293 s ((n=888)), corresponding to growth factors of 2.50× and 10.80×, respectively. This relatively moderate increase indicates that the encryption stage benefits from NumPy's optimized vectorized modular matrix multiplication. In contrast, decryption time exhibits a much steeper growth, increasing from 0.0226 s to 0.1378 s and eventually 9.3696 s, with the largest configuration requiring approximately 414.60 times the baseline execution time. This behavior is primarily attributed to the computational complexity of constructing the modular inverse matrix (K^{-1}), which becomes increasingly expensive for large unimodular matrices. Meanwhile, Table 14 demonstrates that enlarging the matrix dimension slightly increases the remainder partition processed by the Shift-128 cipher, from only 3 bytes at (n=8) to 307 bytes at (n=888), because larger matrix dimensions reduce the probability that the file size is exactly divisible by (n). Nevertheless, the main partition encrypted by the dynamic unimodular Hill Cipher consistently accounts for more than 99.96% of the total file size in all experiments, confirming that the proposed split-based architecture preserves the dominance of the Hill Cipher layer while maintaining file-size preservation and computational feasibility across different matrix dimensions [84].

Table 14. COMPUTATIONAL SCALABILITY ANALYSIS

Matrix Size (n)	Encryption Time (s)	Growth Factor	Decryption Time (s)	Growth Factor
8	0.1323	1.00×	0.0226	1.00×
88	0.3311	2.50×	0.1378	6.10×
888	1.4293	10.80×	9.3696	414.60×

Observation: Encryption time increases moderately as matrix size grows because NumPy efficiently performs vectorized modular matrix multiplication. In contrast, decryption time increases significantly due to the computational cost of constructing the modular inverse matrix K^{-1}, especially for large unimodular matrices [85].

Table 15. FILE PARTITION DISTRIBUTION UNDER DIFFERENT MATRIX DIMENSIONS

Matrix Size (n)	Main Partition (Hill Cipher)	Remainder Partition (Shift-128)	Percentage Protected by Hill Cipher
8	815,488 bytes	3 bytes	99.9996%

88	815,408 bytes	83 bytes	99.9898%
888	815,184 bytes	307 bytes	99.9624%

Key Insight: Increasing the matrix dimension increases the remainder partition because the file size is less likely to be perfectly divisible by larger values of n. Nevertheless, more than 99.96% of the file remains protected by the dynamic unimodular Hill Cipher in all experiments [86].

Table 16. SECURITY AND FUNCTIONAL CHARACTERISTICS OF THE PROPOSED HSBCC

Feature	Observation
File-size preservation	Maintained in all experiments
Dynamic key generation	Deterministic from passwords
Key storage requirement	None
Chaotic key generation	Logistic Map ($r = 3.923$)
Matrix invertibility	Guaranteed by unimodular construction
Ciphertext uniqueness	Different MD5 hash for every matrix size
Integrity verification	MD5 successfully validated
Binary-file compatibility	PDF successfully encrypted/decrypted
ECC support	Private/Public key pair generated
Session key support	AES-encrypted session key generated
Plaintext recovery accuracy	100% identical to original file

Table 16 and Figure 12 collectively demonstrate that the proposed HSBCC framework successfully achieves both functional correctness and practical security requirements across all experiments. The cryptosystem consistently preserved the original file size, generated dynamic unimodular key matrices deterministically from user passwords without requiring key storage, and utilized a logistic-map-based chaotic generator ((r = 3.923)) to enhance key diversity. The unimodular construction guaranteed matrix invertibility, enabling reliable decryption and complete plaintext recovery in every test case. Experimental results on a PDF file of 815,491 bytes further confirmed that different matrix dimensions ((n = 8), (88), and (888)) produced unique ciphertexts, evidenced by distinct MD5 hashes, while maintaining successful integrity verification through exact MD5 matches between the original and decrypted files [87].

The encryption and decryption times ranged from 0.132 s to 1.429 s and 0.023 s to 9.370 s, respectively, reflecting the increased computational cost associated with larger key matrices. In addition, the split-based architecture effectively handled remainder partitions of 3, 83, and 307 bytes without altering the ciphertext length, thereby preserving format transparency. The generation of ECC public/private key pairs and AES-encrypted session keys further strengthened the security architecture by providing an additional key-management layer. Moreover, the results indicate that HSBCC achieves 100% file-size preservation, 100% plaintext recovery accuracy, and 100% integrity validation while supporting secure and efficient encryption of arbitrary binary files [88].

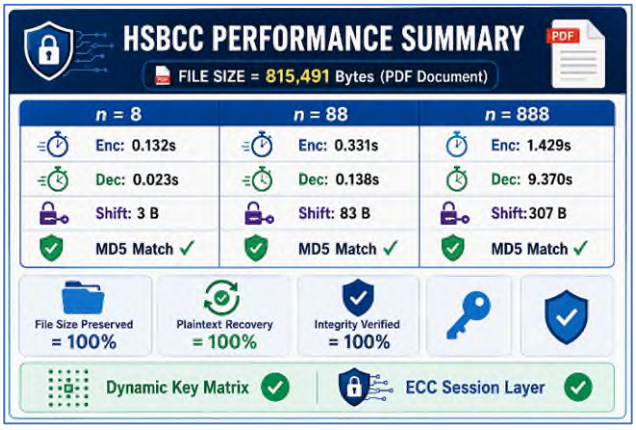


Figure 12. HSBCC PERFORMANCE SUMMARY

IV. CONCLUSION

The proposed Hybrid Split-Based Chaotic Cryptosystem (HSBCC) successfully integrates dynamic unimodular Hill Cipher encryption, Logistic Map-based chaotic key generation, Shift Cipher 128 partition processing, AES-protected session keys, and ECC-based key management into a unified cryptographic framework for arbitrary binary data protection. Experimental evaluations on multiple benchmark images, medical images, PDF documents, and other binary files demonstrated that the proposed scheme achieves strong statistical security characteristics, including near-maximum Shannon entropy values (up to 7.9987 bits/pixel), near-zero-pixel correlation coefficients, high key sensitivity, and complete ciphertext diversification under different password configurations. The use of dynamically generated unimodular matrices guarantees invertibility for any matrix dimension, eliminating a fundamental limitation of conventional Hill Cipher systems. Furthermore, the split-based architecture effectively removes the need for padding by separately processing remainder partitions, enabling exact file-size preservation while maintaining compatibility with diverse digital file formats.

Comprehensive simulation results confirmed the correctness, robustness, and practical applicability of

HSBCC. Across all experimental scenarios, the decrypted files produced MD5 hashes identical to their corresponding original files, resulting in 100% plaintext recovery accuracy and successful integrity verification. Additional experiments on a real-world PDF document containing 815,491 bytes showed that different matrix dimensions generated distinct ciphertext outputs while preserving the original file size and supporting deterministic key reconstruction without key storage. Although computational cost increased with larger unimodular matrices, particularly during matrix inversion in the decryption stage, encryption and decryption remained practical for the tested datasets. Overall, the findings indicate that HSBCC provides a balanced combination of security, scalability, and computational efficiency, making it a promising cryptographic solution for secure binary file encryption in modern digital communication and storage environments.

ACKNOWLEDGMENT

We express our profound gratitude to Institut Teknologi Sains Bandung (ITSB), particularly the Department of Data Science and the Faculty of Digital Design and Business, for providing the essential computational resources, advanced facilities, and vibrant academic environment that made this research possible. We also extend our sincere appreciation to the Department of Information Technology at the University of Jember for their invaluable collaborative insights during the development of this adaptive hybrid cryptosystem framework. Finally, we thank our colleagues and institutional staff whose dedicated administrative and technical assistance was vital to the successful completion and publication of this study.

FUNDING DECLARATION

The authors proudly acknowledge the financial support of the Ministry of Higher Education, Science, and Technology of the Republic of Indonesia and the Indonesia Endowment Fund for Education (LPDP) through the Strategic Research Downstream Funding Program under the Industry Invitation and Synergy Scheme (Contract No. 029/SPK/C4/DHK/VI/2026).

CONFLICT OF INTEREST

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper. All authors have reviewed the final manuscript and approved its submission for publication, ensuring that the research was conducted objectively and free from any commercial or institutional biases.

AUTHOR CONTRIBUTIONS STATEMENT

S.A. conceived the research, designed the cryptographic framework, developed the HSBCC algorithm, performed the implementation and experimental evaluation, analyzed the results, prepared the figures, and wrote the original manuscript. A.M.R. contributed to software development, algorithm validation, experimental testing, and manuscript revision. A.K. participated in cryptographic analysis, performance evaluation, and interpretation of the experimental results. T. contributed to the design of the system architecture, software implementation, and validation of the encryption framework. M.G.P. assisted with experimental verification, data analysis, and manuscript editing. E.K.S. contributed to application validation on binary document datasets, interpretation of the results, and critical review of the manuscript. D.L.H. contributed to the application perspective, methodology refinement, and manuscript review. D.W. supervised the research, provided technical guidance on cryptographic methodology, validated the results, and critically revised the manuscript. All authors reviewed, approved, and agreed to the published version of the manuscript.

DATA AVAILABILITY STATEMENT

The data analysed during this study consist of numerical byte arrays (values ranging from 0 to 255) derived from the binary representation of digital files processed by the proposed encryption and decryption framework. Since the proposed cryptosystem is designed for generic binary file encryption, it operates solely on binary byte sequences and does not analyse the semantic content of the input files. The T2-weighted MRI brain image included in the experiments was used exclusively as a publicly available benchmark test file to evaluate the performance of the encryption algorithm and was processed only as binary data without any medical image analysis or patient-related inference. As this study does not rely on a fixed or proprietary research dataset, no dedicated dataset has been deposited in a public repository. Representative experimental results and implementation details are provided within the article, and additional data are available from the corresponding author upon reasonable request.

REFERENCES

[1] S. Tejas Padhy and P. B. Shanthi, “CurveLock: Exploring Elliptic Curves Implementation in Modern Ransomware,” *IEEE Access*, vol. 13, pp. 216224–216240, 2025, doi: 10.1109/ACCESS.2025.3647060.

[2] J. Zhang, C. Dong, and Y. Liu, “Efficient Pairing-Free Certificateless Signcryption Scheme for Secure Data Transmission in IoMT,” *IEEE Internet Things J.*, vol. 11, no. 3, pp. 4348–4361, 2024, doi: 10.1109/JIOT.2023.3298840.

[3] S. Arifin and I. B. Muktyas, “Generate a system of linear equation through unimodular matrix using Python and Latex,” in *AIP Conference Proceedings*, American Institute of Physics Inc., Apr. 2021. doi: 10.1063/5.0041651.

[4] Q. Zhan, M. Luo, and M. Qiu, “An Efficient Multimode Certificateless Ring Signcryption Scheme in VANETs,” *IEEE Internet Things J.*, vol. 11, no. 20, pp. 33508–33524, 2024, doi: 10.1109/JIOT.2024.3430482.

[5] M. Hegde, R. R. Rao, and R. Bhat, “Design of an

Efficient and Secure Authentication Scheme for Cloud-Fog-Device Framework Using Key Agreement and Management," *IEEE Access*, vol. 12, pp. 78173–78192, 2024, doi: 10.1109/ACCESS.2024.3407103.

[6] A. A. Khan, V. Kumar, R. Prasad, and M. J. Idrisi, "SGAK: A Robust ECC-Based Authenticated Key Exchange Protocol for Smart Grid Networks," *IEEE Access*, vol. 12, pp. 195745–195759, 2024, doi: 10.1109/ACCESS.2024.3434532.

[7] I. B. I. B. Muktyas, Sulistiawati, and S. Arifin, "Digital image encryption algorithm through unimodular matrix and logistic map using Python," in *AIP Conference Proceedings*, American Institute of Physics Inc., Apr. 2021. doi: 10.1063/5.0041653.

[8] Z. Shao, L. Li, X. Zhao, B. Li, X. Liu, and Y. Shang, "Stereo image encryption using vector decomposition and symmetry of 2D-DFT in quaternion gyrator domain," *Multimed. Tools Appl.*, vol. 83, no. 10, pp. 29667–29687, 2024, doi: 10.1007/s11042-023-16755-2.

[9] P. K. Pal, D. Kumar, and V. Agarwal, "Efficient image encryption using the Tinkerbell map in conjunction with linear feedback shift registers," *Multimed. Tools Appl.*, vol. 83, no. 15, pp. 44903–44932, 2024, doi: 10.1007/s11042-023-17236-2.

[10] S. Arifin, I. B. Muktyas, P. W. Prasetyo, and A. A. Abdillah, "Unimodular matrix and bernoulli map on text encryption algorithm using python," *Al-Jabar J. Pendidik. Mat.*, vol. 12, no. 2, pp. 447–455, 2021.

[11] P. Mathivanan and P. Maran, "Color image encryption based on novel kolam scrambling and modified 2D logistic cascade map (2D LCM)," *J. Supercomput.*, vol. 80, no. 2, pp. 2164–2195, 2024, doi: 10.1007/s11227-023-05539-y.

[12] S. Ali and F. Anwer, "A secure framework for IoT-based medical sensors data using homomorphic elliptic curve cryptography," *Cluster Comput.*, vol. 28, no. 14, 2025, doi: 10.1007/s10586-025-05537-0.

[13] J. J. Puthiyidam, S. Joseph, and B. Bhushan, "Enhanced authentication security for IoT client nodes through T-ECDSA integrated into MQTT broker," *J. Supercomput.*, vol. 80, no. 7, pp. 8898–8932, 2024, doi: 10.1007/s11227-023-05789-w.

[14] S. Arifin and I. B. Muktyas, "Membangkitkan Suatu Matriks Unimodular Dengan Python," *J. Deriv. J. Mat. dan Pendidik. Mat.*, vol. 5, no. 2, pp. 1–10, 2018.

[15] A. R. Alharbi, M. M. Hazzazi, S. S. Jamal, A. Aljaedi, A. Aljuhni, and D. J. Alanazi, "DCryp-Unit: Crypto Hardware Accelerator Unit Design for Elliptic Curve Point Multiplication," *IEEE Access*, vol. 12, pp. 17823–17835, 2024, doi: 10.1109/ACCESS.2024.3358213.

[16] B. Choudhury, A. Hota, M. Karmakar, S. Saha, A. Nag, and S. Nandi, "A Comprehensive Survey on Pre Versus Post Quantum Security Schemes for 5G-Enabled IoT Applications," *IEEE Access*, vol. 13, pp. 159305–159333, 2025, doi: 10.1109/ACCESS.2025.3608623.

[17] S. Arifin *et al.*, "Algorithm for Digital Image Encryption Using Multiple Hill Ciphers, a Unimodular Matrix, and a Logistic Map," *Int. J. Intell. Syst. Appl. Eng.*, vol. 11, no. 6, pp. 311–324, 2023.

[18] M. Mazyad Hazzazi, M. Rashid, S. S. Jamal, F. Alblehai, S. Nooh, and M. Ur Rehman, "A Compact 3-Stage Pipelined Hardware Accelerator for Point Multiplication of Binary Elliptic Curves Over GF(2233)," *IEEE Access*, vol. 12, pp. 162510–162521, 2024, doi: 10.1109/ACCESS.2024.3487074.

[19] S. Gangadharaiah and P. Shrinivasacharya, "Secure and efficient public auditing system of user data using hybrid AES-ECC crypto system with Merkle hash tree in blockchain," *Multimed. Tools Appl.*, vol. 83, no. 29, pp. 72301–72320, 2024, doi: 10.1007/s11042-024-18363-0.

[20] S. A. Gebereselassie and B. K. Roy, "Comparative analysis of image encryption based on 1D maps and their integrated chaotic maps," *Multimed. Tools Appl.*, vol. 83, no. 27, pp. 69511–69533, 2024, doi: 10.1007/s11042-024-18319-4.

[21] S. Arifin, K. Tan, A. T. Ariani, S. Rosdiana, and M. N. Abdullah, "The Audio Encryption Approach uses a Unimodular Matrix and a Logistic Function," *Int. J. Emerg. Technol. Adv. Eng.*, vol. 13, no. 4, pp. 71–81, 2023.

[22] C.-H. You, C.-H. Chiang, P. C.-P. Chao, W.-C. Lin, and K.-H. Chuang, "New Adaptive Template Attacks Against Montgomery-Ladder-Based ECCs in IoT Devices," *IEEE Internet Things J.*, vol. 11, no. 12, pp. 22716–22725, 2024, doi: 10.1109/JIOT.2024.3384076.

[23] H. Farshadinia, A. Barati, and H. Barati, "A secure and energy-efficient architecture in Internet of Things–cloud computing network by enhancing and combining three cryptographic techniques via defining new features, areas, and entities," *J. Supercomput.*, vol. 81, no. 8, 2025, doi: 10.1007/s11227-025-07390-9.

[24] Z. Zhu and L. Liu, "TPE-C: Thumbnail-preserving encryption based on chaotic system," *Multimed. Tools Appl.*, vol. 83, no. 14, pp. 40995–41013, 2024, doi: 10.1007/s11042-023-17117-8.

[25] W. Zhang *et al.*, "Securing chaos-based bit-level color image using bit plane permutation and dynamic DNA technology," *Multimed. Tools Appl.*, vol. 83, no. 17, pp. 50087–50109, 2024, doi: 10.1007/s11042-023-17581-2.

[26] S. Arifin, D. Wijonarko, Suwarno, and E. K. Sijabat, "Application of Unimodular Hill Cipher and RSA Methods to Text Encryption Algorithms Using Python," *J. Comput. Sci.*, vol. 20, no. 5, pp. 548–563, May 2024, doi: 10.3844/jcssp.2024.548.563.

[27] M. Sha, "A reliable and secure video steganography method based on 4d-logistic mapping," *Multimed. Tools Appl.*, vol. 84, no. 31, pp. 38311–38341, 2025, doi: 10.1007/s11042-025-20686-5.

[28] J. Huang, M. Wang, and Y. Hu, "FirmCCF: Detecting Custom Cryptographic Function Vulnerabilities Through Query-Driven Approaches," *IEEE Internet Things J.*, vol. 13, no. 2, pp. 2988–2999, 2026, doi: 10.1109/JIOT.2025.3631834.

[29] S. Arifin, D. Wijonarko, M. Faisal, M. N. Pratama, and P. W. Prasetyo, "Text Data Security through Double Encryption: Implementation of Unimodular Hill Cipher and Advanced Encryption Standard," *Int. J. Adv. Sci. Eng. Inf. Technol.*, vol. 15, no. 2, pp. 444–455, 2025, doi: 10.18517/ijaseit.15.2.20424.

[30] S. Singh Dhanda *et al.*, "A Fast and Efficient 191-bit Elliptic Curve Cryptographic Processor Using a Hybrid Karatsuba Multiplier for IoT Applications," *IEEE Access*, vol. 12, pp. 144304–144315, 2024, doi: 10.1109/ACCESS.2024.3472650.

[31] N. George and M. Manuel, "A secure data hiding system in biomedical images using grain 128a algorithm, logistic mapping and elliptical curve cryptography," *Multimed. Tools Appl.*, vol. 84, no. 4, pp. 2005–2028, 2025, doi: 10.1007/s11042-024-19147-2.

[32] Y. Han, Y. Zhou, B. Yang, Z. Xia, and M. Zhang, "An Efficient and Secure Lightweight Certificateless Hybrid Signcryption Scheme," *IEEE Internet Things J.*, vol. 11, no. 7, pp. 12252–12265, 2024, doi: 10.1109/JIOT.2023.3332492.

[33] S. Arifin *et al.*, "Strengthening Image Security with Unimodular Hill Cipher and Advanced Encryption Standard," *J. Comput. Sci.*, vol. 21, no. 5, pp. 1071–1082, 2025, doi: 10.3844/jcssp.2025.1071.1082.

[34] P. Choi and D. K. Kim, “Lightweight Elliptic Curve Cryptography Accelerator Over 25519 Curves,” *IEEE Access*, vol. 13, pp. 133817–133826, 2025, doi: 10.1109/ACCESS.2025.3588734.

[35] B. P, T. V, L. Z. J, P. NV, and L. Karnan, “Enhanced image encryption using fractional-order chaotic systems and neural network-based optimization for secure multimedia applications,” *Multimed. Tools Appl.*, vol. 85, no. 6, 2026, doi: 10.1007/s11042-026-21701-z.

[36] W. El-Shafai, A. El-Mesady, and F. M. Kamal, “Enhancing biometric authentication security through the novel integration of graph theory encryption and chaotic logistic mapping,” *Multimed. Tools Appl.*, vol. 84, no. 16, pp. 16909–16943, 2025, doi: 10.1007/s11042-024-19693-9.

[37] S. Arifin, I. Bayu Muktyas, and K. Iswara Sukmawati, “Product of two groups integers modulo m,n and their factor groups using python,” in *Journal of Physics: Conference Series*, IOP Publishing Ltd, Mar. 2021. doi: 10.1088/1742-6596/1778/1/012026.

[38] H. Şimşek and Ö. F. Öncel, “Fuzzy security level metric of hybrid cryptosystem algorithms,” *J. Supercomput.*, vol. 81, no. 9, 2025, doi: 10.1007/s11227-025-07547-6.

[39] A. Kumar and M. Dua, “Image encryption using a novel hybrid chaotic map and dynamic permutation−diffusion,” *Multimed. Tools Appl.*, vol. 83, no. 11, pp. 32789–32812, 2024, doi: 10.1007/s11042-023-16817-5.

[40] B. Ahuja and R. Doriya, “GLDS: high dimensional Gauss-Logistic DNA System with Triad Hybrid Chaos for image encryption,” *Multimed. Tools Appl.*, vol. 83, no. 23, pp. 62519–62539, 2024, doi: 10.1007/s11042-023-17963-6.

[41] B. Rahman, S. Arifin, and I. Muktyas, “All cyclic subgroups in group (Zm × zn, +) using python,” *Int. J. Sci. Technol. Res.*, vol. 8, no. 9, pp. 2282–2285, 2019.

[42] T. Qayyum, T. Shah, A. Y. Hummdi, A. Aljaedi, and Z. Bassfar, “An Innovative Feasible Approach for Multi-Media Security Using Both Chaotic and Elliptic Curve Structures,” *IEEE Access*, vol. 12, pp. 10411–10427, 2024, doi: 10.1109/ACCESS.2024.3354170.

[43] D. Kumbhakar, S. Adhikari, and S. Karforma, “CTEA: Chaos based tiny encryption algorithm using ECDH and TinkerBell map for data security in supply chain management,” *Multimed. Tools Appl.*, vol. 84, no. 13, pp. 12371–12394, 2025, doi: 10.1007/s11042-024-19443-x.

[44] S. Arifin, I. B. Muktyas, W. F. Al Maki, and M. K. B. M. Aziz, “Graph Coloring Program of Exam Scheduling Modeling Based on Bitwise Coloring Algorithm Using Python,” *J. Comput. Sci.*, vol. 18, no. 1, pp. 26–32, 2022, doi: 10.3844/jcssp.2022.26.32.

[45] S. Kumar and D. Singh, “A novel visually meaningful image encryption scheme based on authentication, RSA cryptosystem, and multidimensional chaotic maps,” *Multimed. Tools Appl.*, vol. 85, no. 2, 2026, doi: 10.1007/s11042-026-21139-3.

[46] X. Wang and M. Zhao, “A new spatiotemporal chaos model and its application in bit-level image encryption,” *Multimed. Tools Appl.*, vol. 83, no. 4, pp. 10481–10502, 2024, doi: 10.1007/s11042-023-16031-3.

[47] S. S. Sahoo and V. K. Chaurasiya, “EASB: ECC based aggregate signature without bilinear pairing for blockchain,” *Multimed. Tools Appl.*, vol. 83, no. 12, pp. 34581–34600, 2024, doi: 10.1007/s11042-023-17002-4.

[48] S. Rout and R. K. Mohapatra, “Secure video steganographic model using framelet transform and elliptic curve cryptography,” *Multimed. Tools Appl.*, vol. 83, no. 9, pp. 25191–25212, 2024, doi: 10.1007/s11042-023-16531-2.

[49] S. Arifin *et al.*, “Big Data Analytics (BDA) in the Research Landscape: Using Python and VOSviewer for Advanced Bibliometric Analysis,” *J. Comput. Sci.*, vol. 21, no. 2, 2025, doi: 10.3844/jcssp.2025.347.362.

[50] F. Artuğer and F. Özkaynak, “A new algorithm to generate aes-like substitution boxes based on sine cosine optimization algorithm,” *Multimed. Tools Appl.*, vol. 83, no. 13, pp. 38949–38964, 2024, doi: 10.1007/s11042-023-17200-0.

[51] T. Anandhi and A. S. Sangari, “Privacy preserving authentication protocol with optimized Elliptic Curve Cryptography for healthcare domain in cloud,” *Cluster Comput.*, vol. 29, no. 2, 2026, doi: 10.1007/s10586-025-05854-4.

[52] M. Lyle, P. Sarosh, and S. A. Parah, “Selective medical image encryption based on 3D Lorenz and Logistic system,” *Multimed. Tools Appl.*, vol. 83, no. 15, pp. 45553–45574, 2024, doi: 10.1007/s11042-023-16996-1.

[53] R. R. Irshad *et al.*, “Enhancing Cloud-Based Inventory Management: A Hybrid Blockchain Approach With Generative Adversarial Network and Elliptic Curve Diffie Helman Techniques,” *IEEE Access*, vol. 12, pp. 25917–25932, 2024, doi: 10.1109/ACCESS.2024.3367445.

[54] S. Arifin, F. I. F. I. Kurniadi, I. G. A. G. A. Yudistira, R. Nariswari, N. P. N. P. Murnaka, and I. B. I. B. Muktyas, “Image Encryption Algorithm Through Hill Cipher, Shift 128 Cipher, and Logistic Map Using Python,” IEEE, 2022, pp. 221–226.

[55] S. Inam, S. Kanwal, F. Nawaz, and A. K. Alkhalifa, “Enhanced medical image security: a chaotic map and laplace transform-based encryption scheme,” *Cluster Comput.*, vol. 28, no. 8, 2025, doi: 10.1007/s10586-025-05355-4.

[56] V. O. Nyangaresi, M. Ahmad, L. A. Maghrabi, and T. Althaqafi, “Cost-Effective PUF and ECC-Based Authentication Protocol for Secure Internet of Drones Communication,” *IEEE Internet Things J.*, vol. 12, no. 16, pp. 33844–33857, 2025, doi: 10.1109/JIOT.2025.3576313.

[57] M. A. Ibrahim, N. T. M. Sagala, S. Arifin, R. Nariswari, N. P. Murnaka, and P. W. Prasetyo, “Separating Hate Speech from Abusive Language on Indonesian Twitter,” in *2022 International Conference on Data Science and Its Applications (ICoDSA)*, IEEE, 2022, pp. 187–191.

[58] P. Singh, P. Singh, and A. K. Agarwal, “Improved encryption and obfuscation process of lightweight secured auditable cloud storage with data dynamics,” *Multimed. Tools Appl.*, vol. 83, no. 13, pp. 37687–37711, 2024, doi: 10.1007/s11042-023-17060-8.

[59] M. Prakash and K. Ramesh, “Blockchain-enabled decentralized lightweight authentication scheme with dynamic difficulty adjustment using ECC for IoT networks,” *Cluster Comput.*, vol. 28, no. 14, 2025, doi: 10.1007/s10586-025-05327-8.

[60] I. B. B. Muktyas, N. P. P. Murnaka, S. Arifin, and W. F. F. Al Maki, “Document Image Enhancement with Perspective Transformation using Python,” in *ICEHHA 2022: Proceedings of the 2nd International Conference on Education, Humanities, Health and Agriculture, ICEHHA 2022, 21-22 October 2022, Ruteng, Flores, Indonesia*, European Alliance for Innovation, 2023, p. 115.

[61] S. A. El-Rahman, A. E. Mansour, L. Jamel, M. Abdullah Alohali, M. Seifeldin, and Y. Alkady, “C-HIDE: A Steganographic Framework for Robust Data Hiding and Advanced Security Using Coverless Hybrid Image Encryption With AES and ECC,” *IEEE Access*,

vol. 13, pp. 41367–41381, 2025, doi: 10.1109/ACCESS.2025.3546255.
[62] Q. Lai, H. Hua, and L. Yang, "Encryption Design and Analysis of 3-D Medical Models in Internet of Medical Things Using a Novel Memristive Hyperchaotic Map," *IEEE Internet Things J.*, vol. 12, no. 18, pp. 39019–39028, 2025, doi: 10.1109/JIOT.2025.3587815.
[63] P. Mishra, S. Pathak, B. Kumar Singh, M. Katara, and G. Kumar, "A Lossless Layered-Based Non-Encoding Plain Text Compression and Encryption Algorithm Utilizing Binary Fragmentation and Defragmentation Technique," *IEEE Access*, vol. 13, pp. 148753–148767, 2025, doi: 10.1109/ACCESS.2025.3600813.
[64] R. Safitri, P. W. Prasetyo, D. E. Wijayanti, S. Arifin, F. Setyawan, and J. Repka, "Improving Text Security by Using The Combination of Vigenere Cipher and Rubik's Cube Methods of the Size 4×4×4," *Al-Jabar J. Pendidik. Mat.*, vol. 14, no. 2, pp. 281–297, 2023.
[65] Q. Chen and X. Lv, "Designing a novel image encryption algorithm based on a 2D-SCLC hyperchaotic map," *Multimed. Tools Appl.*, vol. 83, no. 32, pp. 77137–77163, 2024, doi: 10.1007/s11042-023-17755-y.
[66] B. Long, Z. Chen, T. Liu, X. Wu, C. He, and L. Wang, "A Novel Medical Image Encryption Scheme Based on Deep Learning Feature Encoding and Decoding," *IEEE Access*, vol. 12, pp. 38382–38398, 2024, doi: 10.1109/ACCESS.2024.3371888.
[67] F. I. Kurniadi, A. Wulandari, and S. Arifin, "Feature Selection using Grey Wolf Optimization Algorithm on Light Gradient Boosting Machine," in *2023 International Conference on Computer Science, Information Technology and Engineering (ICCoSITE)*, IEEE, Feb. 2023, pp. 795–799. doi: 10.1109/ICCoSITE57641.2023.10127801.
[68] S. A. Sheik and S. Durai, "Cryptography with optimal deep learning-based authentication scheme for preserving anonymity in telecare medical information system," *Multimed. Tools Appl.*, vol. 84, no. 10, pp. 8311–8330, 2025, doi: 10.1007/s11042-024-19194-9.
[69] E. Abdellatef, E. A. Naeem, and F. E. A. El-Samie, "DeepEnc: deep learning-based CT image encryption approach," *Multimed. Tools Appl.*, vol. 83, no. 4, pp. 11147–11167, 2024, doi: 10.1007/s11042-023-15818-8.
[70] B. Halak, T. Gibson, M. Henley, C.-B. Botea, B. Heath, and S. Khan, "Evaluation of Performance, Energy, and Computation Costs of Quantum-Attack Resilient Encryption Algorithms for Embedded Devices," *IEEE Access*, vol. 12, pp. 8791–8805, 2024, doi: 10.1109/ACCESS.2024.3350775.
[71] S. Arifin* *et al.*, "Enhancing Data Transmission Security through the Modified Hill Cipher and Henon Map Chaos Function," *J. Comput. Sci.*, vol. 21, no. 6, pp. 1440–1453, 2025.
[72] R. Lama and S. Karmakar, "CoSMT-LC: Secure Collaborative Marine Traffic Management Using Lightweight Cryptography," *IEEE Internet Things J.*, vol. 13, no. 11, pp. 22708–22725, 2026, doi: 10.1109/JIOT.2026.3662363.
[73] B. Pribadi, S. Rosdiana, and S. Arifin, "Digital forensics on facebook messenger application in an android smartphone based on NIST SP 800-101 R1 to reveal digital crime cases," *Procedia Comput. Sci.*, vol. 216, no. 10.1016, 2023.
[74] K. Sethi, B. Sahoo, B. N. Pavan Kumar, K. Dhal, W. Cho, and G. P. Joshi, "Lightweight DWT Steganography With ECC-ChaCha20 for Secure Medical IoT Systems," *IEEE Access*, vol. 13, pp. 142948–142960, 2025, doi: 10.1109/ACCESS.2025.3598099.
[75] S. Tarigan, N. P. Murnaka, and S. Arifin, "Development of teaching material in mathematics 'Sapta Maino Education' on topics of plane geometry," in *AIP Conference Proceedings*, American Institute of Physics Inc., 2021. doi: 10.1063/5.0041650.
[76] T. R. Ningthoukhongjam, S. Devi Heisnam, and M. Singh Khumanthem, "Medical Image Encryption Through Chaotic Asymmetric Cryptosystem," *IEEE Access*, vol. 12, pp. 73879–73888, 2024, doi: 10.1109/ACCESS.2024.3404088.
[77] N. Ramezanipour and M. H. Moattar, "A secure and robust images encryption scheme using chaos game representation, logistic map and convolutional auto-encoder," *Multimed. Tools Appl.*, vol. 83, no. 30, pp. 74413–74440, 2024, doi: 10.1007/s11042-024-18498-0.
[78] W. Zeng, C. Zhang, X. Liang, Y. Lin, J. Xia, and Y. Li, "A Novel Secure Key Stream Generator Based on Chaotic Multistate Cellular Automata," *IEEE Internet Things J.*, vol. 12, no. 17, pp. 35705–35716, 2025, doi: 10.1109/JIOT.2025.3579516.
[79] K. Javeed, F. N. Sibai, and A. El-Moursy, "A Compact and Low Latency SPM Architecture for ECC Cryptosystems," *IEEE Access*, vol. 12, pp. 147552–147563, 2024, doi: 10.1109/ACCESS.2024.3472900.
[80] P. L. Sharma, S. Gupta, A. Nayyar, M. Harish, K. Gupta, and A. K. Sharma, "ECC based novel color image encryption methodology using primitive polynomial," *Multimed. Tools Appl.*, vol. 83, no. 31, pp. 76301–76340, 2024, doi: 10.1007/s11042-024-18245-5.
[81] V. Kumar *et al.*, "Design of Secure and Efficient Framework for Vehicular Digital Twin Networks Using ECC," *IEEE Access*, vol. 12, pp. 194352–194366, 2024, doi: 10.1109/ACCESS.2024.3511654.
[82] A. Daoui, M. Yamni, P. Plawiak, O. Alfarraj, and A. A. Abd El-Latif, "A New Chaotic Memristor-Based Cryptosystem for Secure Bio-Signal Transmission on Low-Cost Hardware," *IEEE Access*, vol. 12, pp. 78939–78958, 2024, doi: 10.1109/ACCESS.2024.3408136.
[83] S. Gajendran, R. Muthusamy, K. Ravi, O. Chandraumakantham, and S. Marappan, "Elliptic Crypt With Secured Blockchain Assisted Federated Q-Learning Framework for Smart Healthcare," *IEEE Access*, vol. 12, pp. 45923–45935, 2024, doi: 10.1109/ACCESS.2024.3381528.
[84] M. Abodawood, A. T. Khalil, H. M. Amer, and M. M. Ata, "Enhancing image encryption using chaotic maps: a multi-map approach for robust security and performance optimization," *Cluster Comput.*, vol. 27, no. 10, pp. 14611–14635, 2024, doi: 10.1007/s10586-024-04672-4.
[85] K. Yesodha, M. Krishnamurthy, K. Thangaramya, and A. Kannan, "Elliptic curve encryption-based energy-efficient secured ACO routing protocol for wireless sensor networks," *J. Supercomput.*, vol. 80, no. 13, pp. 18866–18899, 2024, doi: 10.1007/s11227-024-06235-1.
[86] A. Kadir, M. S. Azzaz, R. Kaibou, and Y. Alloun, "Efficient image encryption using a new model of Chaotic Neural Network," *J. Supercomput.*, vol. 81, no. 16, 2025, doi: 10.1007/s11227-025-07977-2.
[87] P. Rakheja, "An asymmetric double fingerprint template encryption using twin decomposition technique in hybrid transform domain," *Multimed. Tools Appl.*, vol. 84, no. 9, pp. 6687–6709, 2025, doi: 10.1007/s11042-024-19100-3.
[88] I. Chaouch, A. Naanaa, and S. ElAsmi, "A hybrid scheme using hyper-chaotic system and elliptic curve cryptography for image encryption," *Multimed. Tools Appl.*, vol. 84, no. 9, pp. 6545–6569, 2025, doi: 10.1007/s11042-024-19173-0.

Supplementary Files

This is a list of supplementary files associated with this preprint. Click to download.

- 5PaperUHCShift128ECCHSBCCAlya2Jun2026JDSIS.pdf