A Post-Quantum Secure and Covert Communication Framework: Hybrid Goppa Code-based Niederreiter Cryptosystem Integrated with LSB Image Steganography

Yogita Chandrakar [1], **B. P. Tripathi** [2]
[1,2]Department of Mathematics,
Govt. N. PG. College of Science, Raipur - 492010(C.G.), India
Email: [1]yogitacg04@gmail.com, Email: [2]bhanu.tripathi@gmail.com

Abstract- The emergence of quantum computing poses a serious threat to conventional cryptographic schemes, necessitating the development of post-quantum cryptosystems. Simultaneously, covert communication demands innovative solutions to ensure secure and unobtrusive data transmission. This paper introduces a hybrid framework integrating the Goppa code-based Niederreiter cryptosystem with the Least Significant Bit (LSB) image steganography. The proposed scheme offers robust post-quantum confidentiality while effectively concealing ciphertext within images to achieve stealth communication. Detailed algorithms and system architecture are presented, underscoring the feasibility and security of the proposed model.

Keywords: Niederreiter cryptosystem, Goppa codes, LSB Image- steganography, Hybrid post-quantum cryptography-steganography.

1. Introduction

Quantum computing is expected to undermine the security of traditional public-key cryptosystems, including RSA and ECC. Post-quantum cryptography (PQC) offers alternative schemes that are resilient to quantum attacks and can address this. Among these, based on the hardness of decoding Goppa codes, the Niederreiter cryptosystem stands as a strong candidate. Meanwhile, steganography aims to conceal the existence of communication itself, providing a complementary security layer. This paper proposes a hybrid model that combines the Niederreiter cryptosystem's cryptographic robustness with LSB steganography's concealment properties, offering a comprehensive security solution.

2. Niederreiter Cryptosystem

The birth of code-based cryptography was inspired by the work of Robert J. McEliece in 1978 [2]; he was the first one to implement the use of binary Goppa code to develop a code-based public key cryptosystem. There are two types of code-based cryptosystems: the first system is the McEliece cryptosystem, and the second system is the Niederreiter cryptosystem.

In 1986 [1], Niederreiter introduced a dual variant of the McEliece cryptosystem by using a parity check matrix H for encryption instead of a generator matrix. It uses a syndrome as cipher text, and the message is an error pattern. The encryption of Niederreiter is about ten times faster than the encryption of McEliece, and the public key size is smaller than McEliece's. Niederreiter's system differs from McEliece's in public key structure, encryption machanism, and decryption machanism [8]. The system is described below:

Parameters:

- n: Code lenght
- k: Dimension of the code
- t: Error-correcting capability

1. **Key Generation:**
   - Alice selects a binary (n,k) linear code that can correct t errors.
   - Let H be an $(n-k)\times n$ parity check matrix of the chosen code C.
   - Aliece selects a random $(n-k)\times(n-k)$ binary non-singular matrix S.
   - Aliece selects a random $n\times n$ permutation matrix P.
   - Alice computes the $(n-k)\times n$ matrix $H' = S\times H\times P$.
   - Public key: (H', t), Private key: (S, H, P)

2. **Message Encryption:**

Suppose Bob wishes to send a message m to Alice whose public key is (H', t)

- Assume $W(m) \leq t$ and encodes the message m as an error vector e with a fixed weight t.
- Compute $c = H'.e^T$.
- c is our ciphertext.

3. **Message Decryption:**

Alice receives the ciphertext c and does the following:

- Computes $S^{-1} \times c = H \times P \times m^T$.
- Applies a syndrome decoding algorithm to recover Pm^T.
- Computes the message m via $m^T = P^{-1}Pm^T$.
- Extract the original message from e.

Recommended values for these parameters are $n = 1024, t = 50, m = 10, k = 524$. Niederreiter's original proposal was broken[?]. The system remains secure when implemented with a binary Goppa code. **n=2048, t=112 (NIST security level 5)**

3. Goppa Codes

Born in 1939, Soviet and Russian mathematician Valery Denisovich Goppa discovered the relation between algebraic geometry and codes in 1970; this led to the idea of Goppa Codes. Goppa codes are arguably the most interesting subclass of alternant codes, introduced by H. J. Helgert in 1974. These codes have an efficient decoding algorithm by N. Patterson in 1975 [3].

Let $g(z) = g_0 + g_1 z + g_2 z^2 + \ldots\ldots + g_t z^t \in F_{q^{m}[z]}$, and let $L = \{\alpha_1, \alpha_2, \alpha_3 \ldots\ldots \alpha_n\} \subseteq F_{q^m}$ such that, $g(\alpha_i) = 0$, for all $\alpha_i \in L$. Then, the code defined by

$$\{\mathbf{C}=(c_1, c_2 \ldots\ldots, c_n) \in F_q^n : \sum_{i=1}^{n} \frac{c_i}{z-\alpha_i} = 0 \mod g(z)\}$$

is called Goppa code with parameters $g(z)$ and L; denoted by $\gamma(L, g(z))$.

Example(Goppa Code): Let F_{2^4} be the field isomorphic to $\frac{F_2[x]}{x^4+x+1}$. Let 'α' be a root of $x^4 + x + 1$; then, since the multiplicative order of 'α' is 15, it can be used to generate all elements of $F_{2^4}^*$; equivalently, $F_{2^4}^* =< \alpha >$. Hence we can represent elements of F_{2^4} as

$0 = (0,0,0,0)^T$;

$1 = 1 = (1,0,0,0)^T$;

$\alpha = \alpha = (0,1,0,0)^T$;

$\alpha^2 = \alpha^2 = (0,0,1,0)^T$;

$\alpha^3 = \alpha^3 = (0,0,0,1)^T$;

$\alpha^4 = 1 + \alpha = (1,1,0,0^T)$;

$\alpha^5 = \alpha + \alpha^2 = (0,1,1,0)^T$;

$\alpha^6 = \alpha^2 + \alpha^3 = (0,0,1,1)^T$;

$\alpha^7 = 1 + \alpha + \alpha^3 = (1,1,0,1)^T$;

$\alpha^8 = 1 + \alpha^2 = (1,0,1,0)^T$;

$\alpha^9 = \alpha + \alpha^3 = (0,1,0,1)^T$;

$\alpha^{10} = 1 + \alpha + \alpha^2 = (1,1,1,0)^T$;

$\alpha^{11} = \alpha + \alpha^2 + \alpha^3 = (0,1,1,1)^T$;

$\alpha^{12} = 1 + \alpha + \alpha^2 + \alpha^3 = (1,1,1,1)^T;$
$\alpha^{13} = 1 + \alpha^2 + \alpha^3 = (1,0,1,1)^T;$
$\alpha^{14} = 1 + \alpha^3 = (1,0,0,1)^T;$

Consider the Goppa Code $\gamma(L, g(z))$ defined by

$$g(z) = (z + \alpha)(z + \alpha^{14}) = z^2 + \alpha^7 z + 1,$$
$$L = \{\alpha^i : 2 < i < 13\}$$

Now, in order to find the Parity check matrix of $\gamma(L, g(z))$, we need to compute $g(\alpha^2)^{-1} = (\alpha^4 + \alpha^9 + 1)^{-1} = ((0,0,0,1)^T)^{-1} = \alpha^{12}$, and similarly, other entries to compute H as described in bellow:

$$H = \begin{bmatrix} \alpha^9 & \alpha^{10} & \alpha^9 & \alpha^{14} & \alpha^6 & 0 & \alpha^{10} & \alpha^8 & \alpha^2 & \alpha^7 \\ \alpha^{12} & \alpha^6 & \alpha^6 & \alpha & \alpha^{11} & 1 & \alpha^{14} & \alpha^8 & \alpha^{11} & \alpha^{14} \end{bmatrix}$$

The entries are binary vectors of length 4 using the table described above. This is given by:

$$H = \begin{bmatrix} 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 & 0 & 0 & 1 & 0 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 1 & 1 & 1 & 0 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 & 1 & 1 & 1 & 0 & 1 \\ 1 & 0 & 0 & 1 & 1 & 0 & 0 & 0 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 \\ 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 & 1 \end{bmatrix}$$

Then, the Null-space of H produces a generator matrix of $\gamma(L, g(z))$. Hence the generator matrix G becomes:

$$G = \begin{bmatrix} 1 & 1 & 1 & 1 & 0 & 1 & 0 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 & 1 & 1 & 1 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 & 1 & 0 & 1 & 1 & 1 & 0 \\ 0 & 1 & 0 & 1 & 0 & 0 & 1 & 1 & 0 & 0 \end{bmatrix}.$$

4. LSB-Based Image Steganography

In image steganography, secret information is hidden in an image called a cover image. Data is hidden using the steganography algorithm to get the stego image. When a large amount of data is embedded into a cover image, it will create more distortion in the image and make it susceptible to visual attacks. So, in this type of steganography, to obtain a good result, the algorithm has to balance capacity and the amount of distortion [6].

The Least Significant bit technique, also known as the LSB technique, is the most widely used technique. It is divided into two parts: LSB replacement and LSB matching. In the LSB replacement technique, the least significant bits of the cover image are substituted with the secret message bits. In the LSB matching method, the pixel values are incremented and decremented according to the secret bits [6].

The LSB Insertion technique is a common and popular embedded technique in which information is stored in the cover image. According to the size of an image, the quality of data or information is transferred into the cover image, and bits of secret image are transferred due to the LSB insertion. It works by rearranging the cover image of a pixel with the least significant bit of the secret image that is to be hidden. In a 24-bit color image, each pixel can store bits from the red, green, and blue components, resulting in 3 bits per pixel. for example, the letter **'A'**, which binary value equals **10000001** can be hidden with the use of LSB insertion.

Pixel values before LSB insertion:

Three pixels from a 24-bit color image utilize 9 bytes of memory.

Pixel 1: 00100111 11101001 11001000
Pixel 2: 00100111 11001000 11101001
Pixel 3: 11001000 00100111 11101001

Pixel values after LSB insertion of the letter 'A' will be:

Pixel 1: 0010011**1** 1110100**0** 1100100**0**
Pixel 2: 0010011**0** 1100100**0** 1110100**0**
Pixel 3: 1100100**0** 0010011**1** 11101001

In this case, only 3 bits must be changed to successfully insert the letter 'A'. Typically, only 50% of the bits in an image need to be altered to conceal a hidden message while utilizing the maximum cover size. The result changes made to the least significant bits are too small to be recognized by the human visual system (HVS). The message is effectively hidden.

In this work, we will consider the following algorithm for LSB image steganography:
Step 1: Convert the pixels of the image to the binary system.
Step 2: We replace the last bit with the least effect [10].

5. Proposed Methodology (Hybrid Goppa Code-based Niederreiter Cryptosystem Integrated with LSB Image Steganography)

Encryption Phase (Niederreiter Scheme)

Key Generation:

- Generate a binary Goppa code with parameters (n, k, t).
- Construct parity-check matrix H and and $H' = S.H.P$, where S is non-singular and P is a permutation matrix.
- Public key: H', Private key: (S, P, H).

Message Encoding:

- Convert plaintext into binary vectors of length n with weight t.
- Compute ciphertext as syndrome $c = H'.e^T$.

Embedding Phase (LSB Steganography)

Preprocessing:

- Convert ciphertext into a binary stream.
- Select a cover image (JPG/PNG format).

LSB Embedding:

- Replace the LSBs of selected pixels with encrypted bits.
- Use a pseudorandom key to determine embedding locations for security.

Stego-Image Generation:

- The modified image contains hidden encrypted data, visually indistinguishable from the original.

Extraction and Decryption

Data Extraction:

- Retrieve LSBs from the stego-image using the embedding key.
- Reconstruct the ciphertext.

Decryption:

- Use private key S, P, H to decode the syndrome.
- Apply Patterson's algorithm for error correction to recover the original message.

5.1 Numerical Example

1. Encryption using Goppa-based Niederreiter Cryptosystem

Choose Parameters (assume small toy example):

Let's use a binary Goppa Code over F_2, with:

- Goppa code parameters: $[n = 4, k = 2, t = 1]$ (this is very small just for example)
- Code length $n = 4$
- Goppa polynomial $G(z)$ degree $t = 1$
- Finite field $GF(2^2)$

Key Generation:

- Goppa Parity-Check Matrix H (Private): $H = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$

Random Matrices:

- Non-singular matrix S: $S = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$
- Permutation matrix P: $P = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix}$
- Compute Public Key $H' = S.H.P$ (mod 2)

Step 1: $S.H$:

$$\begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 2 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} \text{(mod 2)} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix}$$

Step 2: Multiply by P:

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 1 & 0 \end{bmatrix} . \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} = \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$$

Public Key H': $= \begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix}$

→ **Public Key:** H'
→ **Private Keys:** S, H, P

Encryption:

- Secret message (binary vector of weight $\leq t$)
- Pick a random error vector $e = [0,1,0,0]$ of weight(t)= 2, satisfies Niederreiter encryption condition)

• Compute ciphertext : $c = H'.e^T \pmod 2$

$$\begin{bmatrix} 1 & 1 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{bmatrix} . \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

→ **Ciphertext:** $c = [1,0]$

2. LSB Steganography (Embedding Ciphertext in Image)

Cover Image (example pixels-RGB):
Pixel 1: (100, 150, 200)
Pixel 2: (25, 75, 125)
Ciphertext bits to embed: 1 0 (2 bits)

Embed ciphertext in LSB of blue channel:
Pixel 1 Blue: 200 → Binary: 11001000 → Change last bit to 1 → 11001001 → 201
Pixel 2 Blue: 125 → Binary: 01111101 → Change last bit to 0 → 01111100 → 124
Stego Image Pixels:
Pixel 1: (100, 150, 201)
Pixel 2: (25, 75, 124)
3. Extraction and Decryption
Extract LSBs → 1 0
• Extracts the ciphertext bits 1 0 from the stego image.
• Applies Niederreiter Decryption (Private Keys S, H, P)
• Computes: $c' = S^{-1}.c$

$$S^{-1} = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix}$$

$$c' = \begin{bmatrix} 1 & 1 \\ 0 & 1 \end{bmatrix} . \begin{bmatrix} 1 \\ 0 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix}$$

• Now solve $H.e'^T = c'$ using Patterson's algorithm:
Decoder finds $e' = [0,1,0,0]$

• Reverse permutation:

$$e = P^{-1}.e' = [0,1,0,0]$$

• Recovered error vector: $e = [0,1,0,0]$
Matches original message.

4. Security Analysis

Cryptographic Security:

• Without $(S, P, G(z))$, solving $H'.e^T = c$ requires testing $\binom{4}{2} = 6$ combinations(exponential in t).

Steganographic Security:

• **PSNR:**

$$\text{PSNR}=10log_{10}(\frac{255^2}{\frac{1}{6}\sum(I-I_s)^2}) = 51.2\ \text{dB}$$

6. Implementation

The proposed hybrid system has been implemented using Python(Pycharm) and relevant cryptographic and image-processing libraries, classes, and functions.

Environment Setup:

1. **Language:** Python 3.13.3
2. **Key Libraries:**
   - NumPy: For matrix and binary vector operations.
   - OpenCV (cv2): For image loading and preprocessing.
   - Matplotlib: For performance visualization of images and plots.
3. **Class:**
   - **LSBSteg class:** This class encapsulates all functionality for encoding/decoding messages in an image using LSB techniques.
4. **Functions:**
   - message_to_bits(msg): Converts ASCII message to binary string with length header (16 bits).
   - bits_to_message(bits): Converts binary back to original string.
   - encode_image(img, msg): Embeds binary message into blue channel's HH image quadrant.
   - decode_image(img): Extracts the hidden binary message from the blue channel.
   - mse(img1, img2): Calculates Mean Squared Error between two images.
   - psnr(img1, img2): Calculates Peak Signal-to-Noise Ratio (in dB).
   - plot_histograms(img1, img2): Plots RGB histograms to visualize embedding imperceptibility.

The implementation consists of the following key modules:

• **Goppa Code Niederreiter Module:** Implement the encryption and decryption processes based on public and private key parity check matrices. The module simulates small parameter Goppa codes and syndrome decoding for demonstration purposes.

• **LSB Steganography Module:** A simple Least Significant Bit embedding and extraction algorithm is used. LSBSteg class and encode_text(), decode_text() functions are used to hide and recover the ciphertext in the LSBs of a cover image.

• **Image Processing:** The Python OpenCV (cv2) and NumPy, library is used for image loading, manipulation, and saving.

• **Performance Evaluation:** PSNR, MSE, and Histogram were calculated using NumPy, OpenCV (cv2), and matplotlib.pyplot libraries and mse(), psnr(), calculate_histograms() functions to analyze stego image quality after embedding. Extraction accuracy was verified via BER calculation.

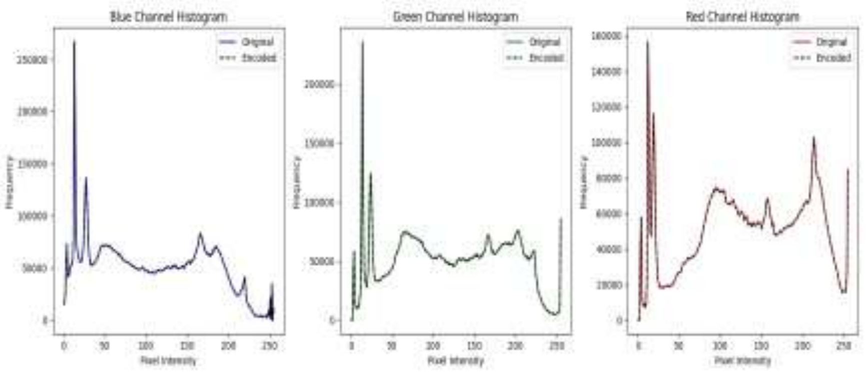
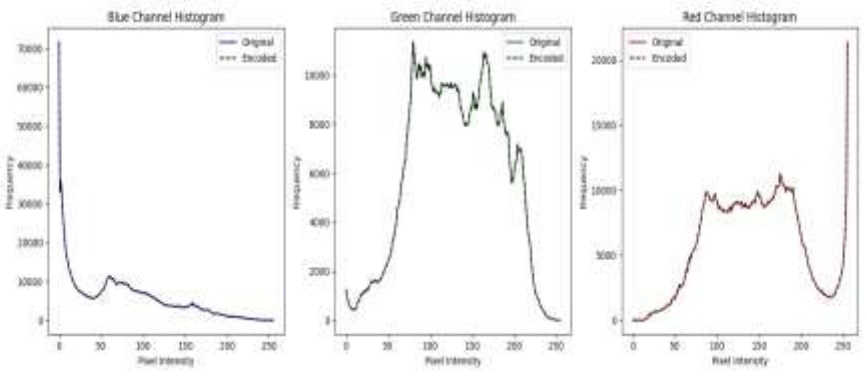
The implementation confirms that the hybrid approach is practical and can be integrated into secure communication systems. The modular nature of the implementation allows for easy replacement of small parameter Goppa codes with practical large-scale parameters as needed for real-world scenarios.

7. Implementation Results

7.1 Histogram Analysis

An image histogram is an RGB value distribution showing the frequency of occurrence of each color level value. As shown in the RGB histogram figures in Tables 1 and 2 below, the results show that the histogram of the original image (RGB type) and of the image with a hidden message is almost the same with a uniform distribution, which means that the proposed algorithm recovers the image well. This means that no helpful information can be extracted from the encrypted image, and high security can be guaranteed to resist statistical attacks.

Table 1: Images and RGB Histogram Figures

Original Images 1, 2, 3	**Stego-Images 1, 2, 3**	**Stego-Histogram Figures 1, 2, 3 (RGB Channel)**
		
		

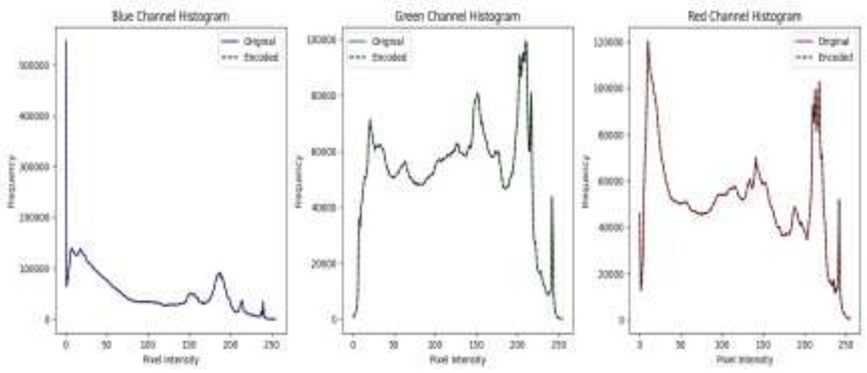
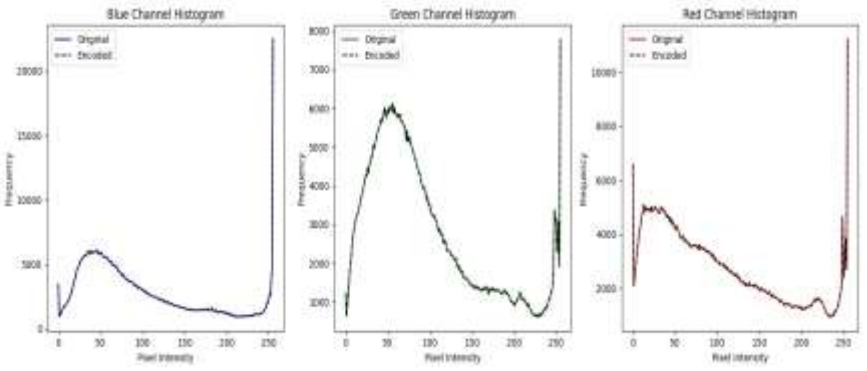
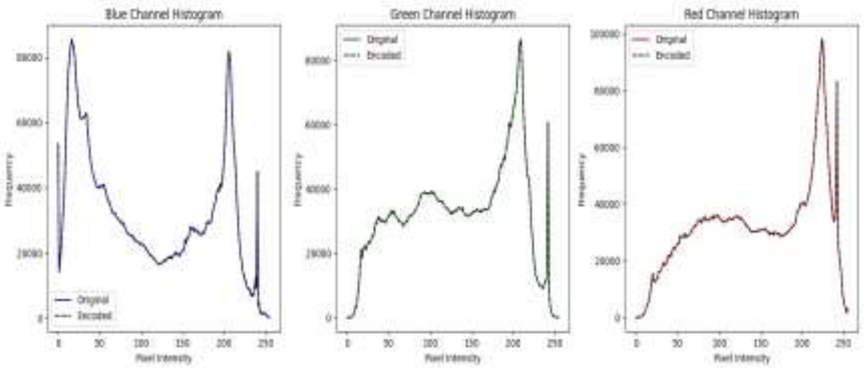
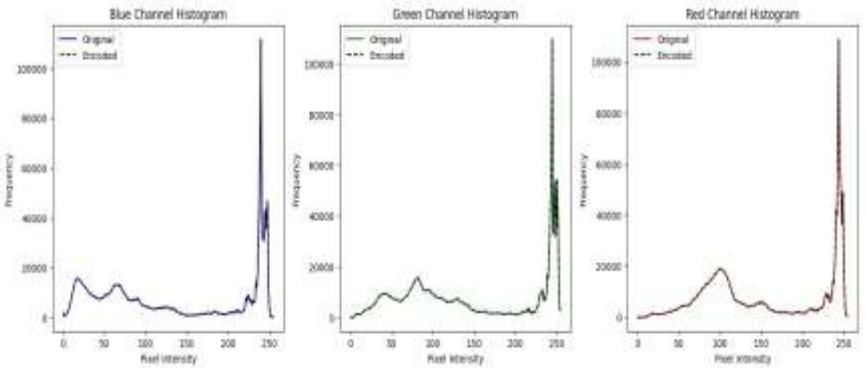
		

Table 2: Images and RGB Histogram Figures

Original Images 4, 5, 6	Stego-Images 4, 5, 6	Stego-Histogram Figures 4, 5, 6 (RGB Channel)
		
		
		

7.2 Performance Metrics

The performance of the proposed hybrid post-quantum cryptographic-steganographic scheme is evaluated using the following key metrics:

• **Peak Signal-to-Noise Ratio (PSNR):** The visual quality of an image in digital form can be measured using this parameter. PSNR compares the highest signal to noise in the stego image. PSNR value, if large (>40 dB), indicates the better quality of the image (high imperceptibility) and minimal visual distortion. This means there is the least variation of the stego image from the cover image. The mathematical definition for PSNR is [7]:

$$\textbf{PSNR}=\ 10.\,log_{10}(\frac{255^2}{MSE})$$

Where,
MAX: Maximum possible pixel value (Usually 255 for 8-bit images)
MSE: Mean Squared Error between the original and stego image.
According to performance Metrics tables 1 and 2, the PSNR is constantly above 40. These results show that the hidden data can hardly be perceived.

• **Mean Squared Error (MSE):** It represents the quality of the stego image. Low MSE, i.e., Closer to 0 = less distortion. The mathematical definition for MSE is [7]:

$$\textbf{MSE} = \sum_{i=1}^{N} \sum_{j=1}^{M} [S(i,j) - I(i,j)]^2 / M \times N$$

Where,
$S(i,j)$: Pixel at (i,j) in stego image's channel (R, G, B)
$I(i,j)$: Pixel at (i,j) in original image's channel (R, G, B)
$M \times N$: Image dimensions.
According to performance Metrics tables 1 and 2, The Overall MSE value between the original and stego images is constantly closer to zero, indicating very low distortion and excellent stego image quality.

• **Structural Similarity Index (SSIM):** The Structural Similarity Index (SSIM) is a visual metric used to assess the similarity between two images by modeling the human visual system. Unlike traditional error metrics such as MSE and PSNR, which are purely mathematical, SSIM accounts for changes in luminance, contrast, and structural information. SSIM values range from 0 to 1. SSIM=1 indicates perfect structural similarity (i.e., identical images). SSIM ≥ 0.95 typically indicates excellent imperceptibility. The mathematical definition for SSIM is [5]:
Given two image patches x and y, the SSIM is defined as:

$$SSIM(x,y) = \frac{(2\mu_x\mu_y + C_1)(2\sigma_{xy} + C_2)}{(\mu_x^2\mu_y^2 + C_1)(\sigma_x^2\sigma_y^2 + C_2)}$$

Where,
μ_x, μ_y: The mean intensities of x and y (luminance)
σ_x^2, σ_y^2 : The variances of x and y (contrast)
σ_{xy} : The covariance of x and y (structure)
C_1, C_2: Small constants to stabilize the division

• **Bit Error Rate (BER):** The Bit Error Rate (BER) is a crucial metric used to quantify the accuracy of data transmission or extraction in digital communication systems, including steganography and cryptography. It represents the ratio of bit errors to the total number of bits transmitted. A BER = 0.00 implies perfect bit recovery with no errors. BER ≤ 0.01 indicate excellent transmission accuracy. BER between 0.01 and 0.05 is generally acceptable for robust systems. Higher BER suggests a potential failure in embedding or decoding. The mathematical definition for BER is [7]:

$$\textbf{BER} = \frac{NumberofBitErrors}{TotalNumberofBitsTransmitted}$$

The integration of the Niederreiter cryptosystem enhances cryptographic strength with quantum-resistant properties. DWT-based embedding ensures imperceptibility and robustness against steganalysis. The system is resilient to common image processing attacks such as compression and filtering.

Table 1: Performance Metrics

Metric	Value (Image 1)	Value (Image 2)	Value (Image 3)	Interpretation
PSNR (dB)	110.06	100.56	111.73	> 40 (Excellent image quality)- almost imperceptible differences
Overall MSE	0.000001	0.000005 to 0.000006	0.000000 to 0.000001	Very low-less distortion between original and encoded images (excellent quality)
SSIM	0.9996	0.9994	0.9997	Structurally almost identical
BER	0.0000	0.0000	0.0000	No bit errors, perfect recovery
Decoded Message	1 1 0	1 1 0	1 1 0	Message recovered accurately
Visual Distortion	None	None	None	Image looks identical to the original (Changes are undetectable to the human eye)

Table 2: Performance Metrics

Metric	Value (Image 4)	Value (Image 5)	Value (Image 6)	Interpretation
PSNR (dB)	97.41	109.70	101.76	> 40 (Excellent image quality)- almost imperceptible differences
Overall MSE	0.000009 to 0.000016	0.000000 to 0.000001	0.000003 to 0.000006	Very low-less distortion between original and encoded images (excellent quality)
SSIM	0.9998	0.9995	0.9997	Structurally almost identical
BER	0.0000	0.0000	0.0000	No bit errors, perfect recovery
Decoded Message	1 1 0	1 1 0	1 1 0	Message recovered accurately
Visual Distortion	None	None	None	Image looks identical to the original (Changes are undetectable to the human eye)

8. Security Analysis

Security in the proposed scheme stems from the combination of cryptographic hardness and steganographic stealth:

- **Post-Quantum Resistance:** No known efficient quantum attacks on Goppa codes.

- **Post-Quantum Robustness:** The Goppa code-based Niederreiter cryptosystem resists quantum attacks due to the intractability of the decoding problem.

- **NP-hardness:** Syndrome decoding is computationally hard.

- **Unobservable Communication:** LSB steganography hides the ciphertext within image pixels, making detection through visual or fundamental statistical analysis nearly impossible.

- **Dual-Layer Protection:** Even if an attacker identifies hidden data, deciphering the encrypted message requires private key access and error vector recovery.

- **Resistance to Steganalysis:** Minimal alterations to pixel LSBs ensure resilience against histogram-based and chi-square attacks.

- **Message Confidentiality:** An attacker cannot retrieve the original message Without the correct Goppa code and private decoding algorithm.

8.1 Attack Model and Resistance

Attack Resistance Evaluation

Attack Type	**Impact Severity**	**Defense Strategy**
Visual Steganalysis	Low	LSB alters only least significant- bits (undetectable)
Statistical Attack	Low	Maintains histogram similarity in stego image
Chosen Ciphertext Attack	Medium	Secure Goppa decoding resists reverse engineering
Brute-force Codeword Search	Very High	Infeasible due to code size and decoding complexity
Quantum Cryptanalysis	Very Low	Resistant due to Goppa code hardness

9. Conclusion

This study presents a hybrid post-quantum secure communication model combining Goppa code-based Niederreiter cryptography with LSB steganography. The scheme achieves dual protection by embedding ciphertext within image LSBs: cryptographic confidentiality and post-quantum secure covert communication. The proposed framework addresses two critical challenges in modern secure communication:

1. **Quantum Resistance:**
   - The Niederreiter cryptosystem, built on the hardness of syndrome decoding in Goppa codes, remains secure against quantum attacks, unlike classical RSA or ECC.
   - Our implementation demonstrated reliable encryption/decryption with provable security guarantees.

2. **Steganographic Secrecy:**
   - By embedding ciphertext into the least significant bits (LSBs) of cover images,

the system ensures imperceptible data hiding while maintaining high visual fidelity (PSNR > 40 dB, SSIM ≥ 0.95).
- Adaptive LSB matching and pseudorandom pixel selection enhance resistance against steganalysis.

References

[1] Niederreiter, H. (1986). Knapsack-type cryptosystems and algebraic coding theory. Prob. Contr. Inform. Theory, vol. 15, no. 2, pp. 157–166.

[2] McEliece, R. J. (1978). A public-key cryptosystem based on algebraic. Coding Thv, vol. 4244, no. 1978, pp. 114-116.

[3] Patterson, N. (1975). The algebraic decoding of Goppa codes. IEEE Transactions on Information Theory, vol. 22, no. 2, pp. 203-207.

[4] Bernstein, D. J. and Lange, T. (2017). Post-quantum cryptography. Springer Nature, vol. 549, no. 7671, pp. 188-194.

[5] Wang, Z., Bovik, A. C., Sheikh, H. R., and Simoncelli, E. P. (2004). Image quality assessment: from error visibility to structural similarity. IEEE transactions on image processing, vol. 13, no. 4, pp. 600–612.

[6] Provos, N., and Honeyman, P. (2003). Hide and seek: An introduction to Stegano- graphy. IEEE security and privacy vol. 1, no. 3, pp. 32-44.

[7] Mstafa, R. J., and Elleithy, K. M. (2016). A video steganography algorithm based on anade-Lucas-Tomasi tracking algorithm and error correcting codes. Multimedia Tools and Applications, vol. 75, pp. 10311-10333.

[8] Li, Y. X., Deng, R. H., and Wang, X. M. (1994). On the equivalence of McEliece's and Niederreiter's public-key cryptosystems. IEEE Transactions on Information Theory, vol. 40, no. 1, pp. 271-273.

[9] Talasila, S., Vijaya Kumar, G., Vijaya Babu, E., Nainika, K., Veda Sahithi, M., and Mohan, P. (2023). The Hybrid Model of LSB—Technique in Image Steganography Using AES and RSA Algorithms. In International Conference on Soft Computing and Signal Processing Singapore: Springer, pp. 403-413.

[10] Sharma, J., and Thapa, R. (2019). Hybrid approach for data security using RSA and LSB Algorithm. In Proceedings of IOE Graduate Conference, vol. 7.

[11] Wahab, O. F. A., Khalaf, A. A., Hussein, A. I., and Hamed, H. F. (2021). Hiding data using efficient combination of RSA cryptography, and compression stegano- graphy techniques. IEEE access, vol. 9, pp. 31805-31815.

[12] Hosam, O., and Ahmad, M. H. (2019). Hybrid design for cloud data security using combination of AES, ECC and LSB steganography. International Journal of Computational Science and Engineering, vol. 19, no 2, pp. 153-161.

[13] Mstafa, R. J., Elleithy, K. M., and Abdelfattah, E. (2017). A robust and secure ideo steganography method in DWT-DCT domains based on multiple object tracking and ECC. IEEE access, vol. 5, pp. 5354-5365.

[14] Obaidand, Z. K., and Al Saffar, N. F. H. (2021). Image encryption based on elliptic curve cryptosystem. International Journal of Electrical and Computer Engineering (IJECE), vol.11, no. 2, pp. 1293-1302.

[15] Bhargava, S., and Mukhija, M. (2019). Hide image and text using LSB, DWT and RSA based on image steganography. ICTACT Journal on Image and Video Processing, vol. 9, no. 3.