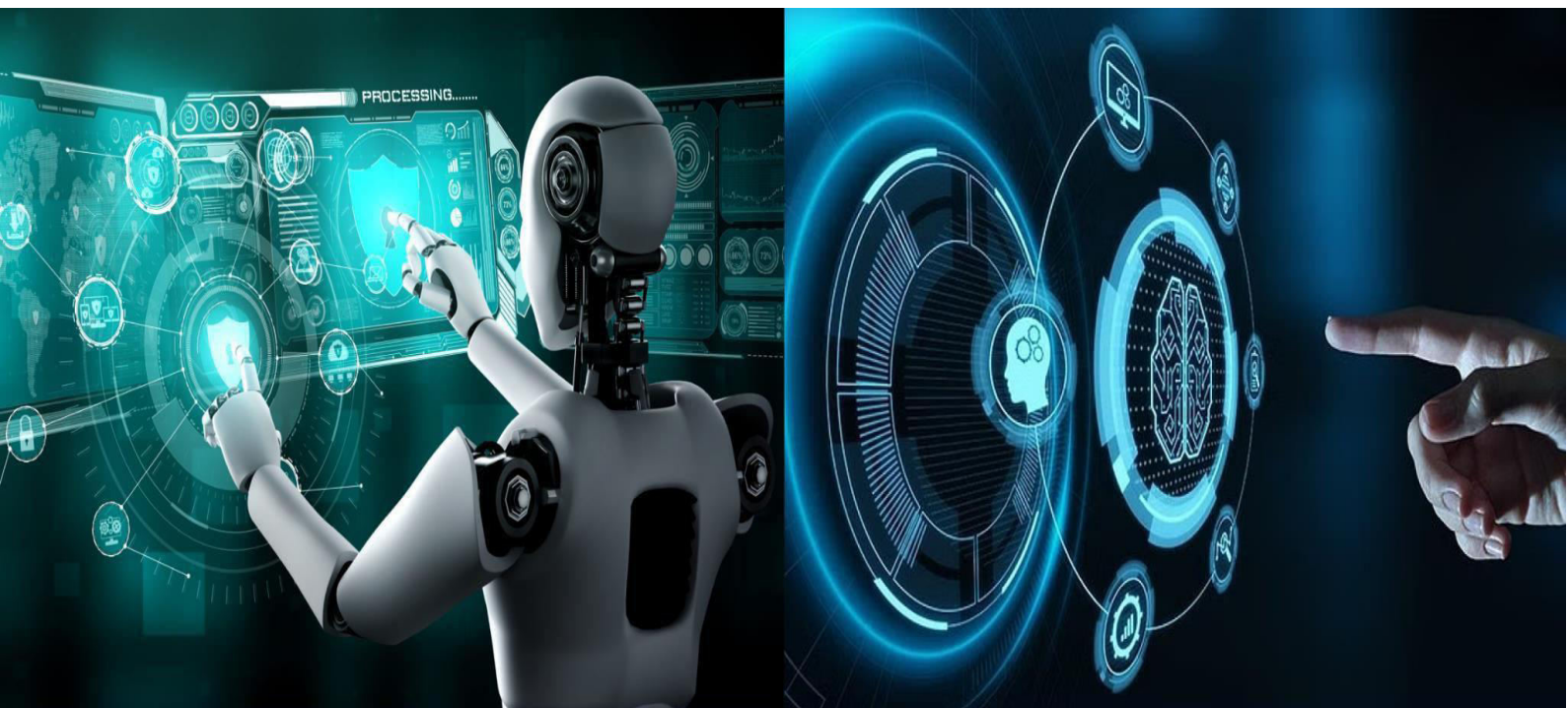




International Journal of Innovative Research in Computer and Communication Engineering

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)





International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Quantum-Resistant Cyber Security Apparatus

Santhosh G P¹, Revanth S², Prajwal³, Ravi Kumara H S⁴, Dr. Malatesh SH⁵

Student, Dept. of Computer Science and Engineering, MS Engineering College, Bengaluru, Karnataka, India^{1,2,3,4}

HOD, Dept. of Computer Science and Engineering, MS Engineering College, Bengaluru, Karnataka, India⁵

ABSTRACT: Traditional cryptographic systems are becoming more vulnerable due to the rapid growth of cyber threats and the emergence of quantum computing as a global security challenge. In order to provide extremely secure and secret data transmission, this project suggests a Quantum-Resistant Cyber Security Apparatus that combines SM4-CBC encryption with Least Significant Bit (LSB) image steganography. Using a portable and offline-compatible platform, the system, which is implemented on a Raspberry Pi 4 with a Flask-based web interface, enables users to encrypt, embed, extract, and validate secret data. Tamper detection and integrity verification are guaranteed by a secure metadata header that includes payload size, type, IV, and CRC/HMAC.

Additionally, the device incorporates a SQLite database for forensic analysis and system usage logging, as well as a 16x2 LCD display for real-time operation feedback. The system continuously maintains a PSNR > 45 dB, according to experimental results, guaranteeing that the stego-images remain visually identical to the original. This dual-layer security model is appropriate for military communication, IoT security, digital forensics, intelligence operations, and secure field environments because it offers strong confidentiality, covert transmission, and robust error detection.

KEYWORDS: Raspberry Pi 4, LSB Steganography, SM4 Encryption, Quantum-Resistant Security, Cybersecurity Appliance, Data Integrity, Flask Web Application, Image Steganography, PSNR, and CRC/HMAC Validation.

I. INTRODUCTION

Data theft, surveillance, cyber espionage, and quantum-based attacks are among the growing security risks to modern digital communication. Dual-layer protection, which allows encrypted data to be transmitted covertly, is becoming more and more necessary as encryption alone may soon become insufficient.

Based on the Raspberry Pi 4, this project presents a small and portable Quantum-Resistant Cyber Security Apparatus that combines steganography and cryptography to provide safe and covert data transmission. The system employs LSB-based steganography to embed encrypted data inside images without causing noticeable distortion and SM4, a 128-bit block cipher made to withstand both classical and quantum cryptographic threats.

Most cybersecurity setups lean on big servers, bulky equipment, or simple encryption that hackers can break into if they try hard enough. This new system goes a different way. Here's what it brings to the table:

- You can use it offline—no need to stay connected all the time. Covert data embedding
- It hides your data right inside other files, so nobody even knows it's there. Real-time LCD monitoring
- The encryption stands up to quantum attacks, not just the usual stuff. Forensic logging via SQLite
- There's an LCD that shows what's happening as it happens.
- It checks to make sure you only pull out data safely, no funny business.
- Every move gets logged with SQLite for solid forensic tracking.

So, what's the point? Well, traditional systems just don't cut it anymore. They're clunky, and their defences look old next to modern threats. That's why we built something that covers more ground, keeps things hidden, and pushes security a step ahead.

II. OBJECTIVES

1. Build SM4-CBC encryption with strong padding and proper handling of the initialization vector.
2. Create LSB-based steganography that hides encrypted data without messing up how the image looks.
3. Set up everything to run on a Raspberry Pi 4, so you can take it anywhere and use it in the field.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

4. Design a simple Flask web interface where users can upload files, embed or extract data, and download results easily.
5. Add CRC or HMAC checks to catch tampering and make sure extraction stays secure.
6. Keep the PSNR above 45 dB so the carrier image always stays sharp and clear.
7. Use a 16×2 LCD to show live status updates—like Encrypting, Embedding, Extracting, or if something goes wrong.
8. Log all timestamps, metadata, and security events to SQLite for easy audits.
9. Check storage space before embedding anything, so you never hit overflow errors.
10. Make embedding and extraction fast enough for real-time use in the field.

III. METHODOLOGY

1. System Requirement Analysis

We spotted security gaps, quantum threats, and weak spots in communication. So, we set out to design something with dual-layer protection, offline mode, integrity checks, and, honestly, it had to be easy to use.

2. SM4 Encryption Pipeline

You start with your text or image—it gets encrypted with SM4-CBC. PKCS#7 padding keeps everything lined up right. For each session, the system creates a fresh IV and sticks it in the metadata. Then, it bundles the encrypted data with some header fields.

3. Metadata Header Construction

The header packs in a magic identifier, payload type, payload length, a 16-byte IV, and a CRC/HMAC checksum.

4. LSB Embedding Process

First, the carrier image turns into a pixel array. Then, the system hides the encrypted bits and metadata in the least significant bits. After that, it checks PSNR to make sure the image quality still holds up.

5. Flask Interface Development

With the Flask interface, users can upload carrier images and secret files, encrypt data, embed it into a stego-image, extract and decrypt files, and download everything they need.

6. Raspberry Pi 4 & LCD Integration

The 16×2 LCD shows messages like:

- System Ready
- Encrypting...
- Embedding...
- Extraction Done
- Invalid Key / Error

7. Extraction & Validation

To pull the data back out, the system grabs bits from the stego image, reads the metadata, verifies CRC/HMAC, and uses SM4 decryption to rebuild the original data.

8. SQLite Logging

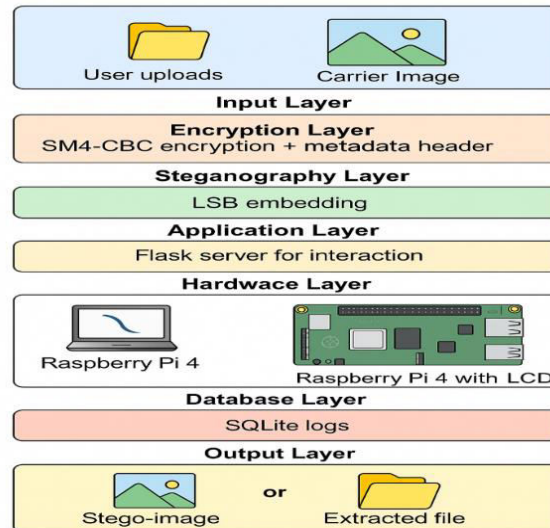
Every operation gets logged—type of operation, file details, timestamp, and validation status.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

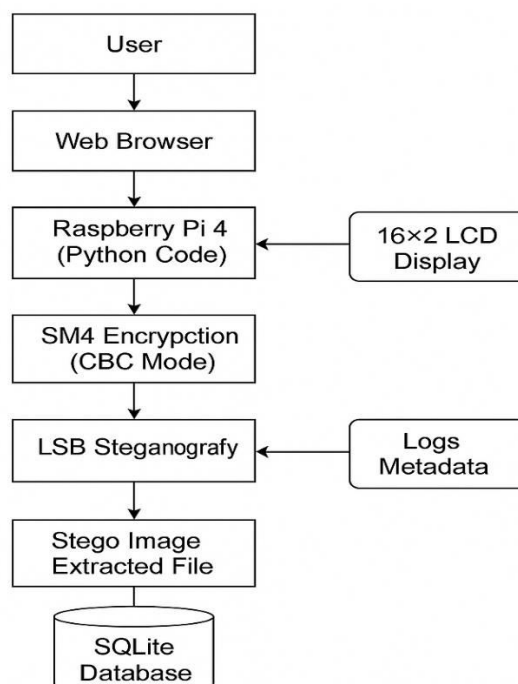
IV. SYSTEM ARCHITECTURE



Here's how it works:

1. You start by uploading your secret file and the carrier image.
2. Next, the system locks everything down with SM4-CBC encryption and adds a metadata header.
3. After that, it hides the encrypted data inside the image using LSB embedding.
4. The whole thing runs through a Flask server, so you can interact with it easily.
5. It's all set up on a Raspberry Pi 4, complete with an LCD screen.
6. Every action gets logged in an SQLite database.
7. Finally, you get either the stego-image with your hidden data or the file you extracted.

Use Case Diagram





International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

Actors:

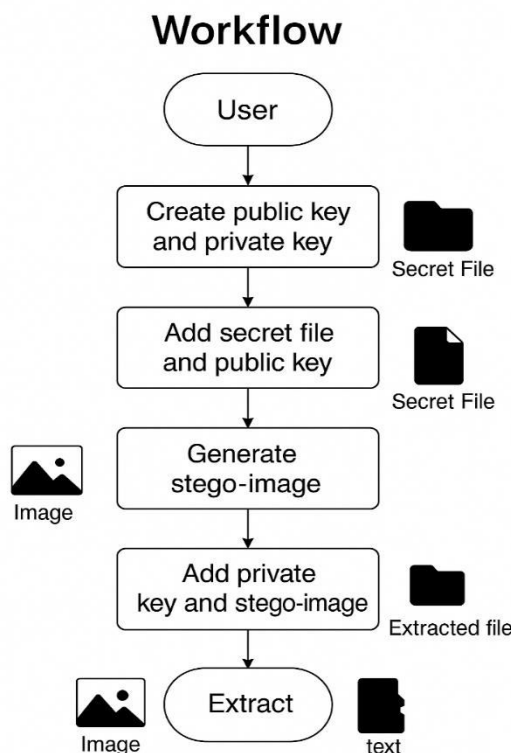
- User
- Raspberry Pi System
- Flask Application

Use Cases:

- Upload File
- Encrypt Data
- Embed Data
- Extract Data
- Validate Integrity
- View Logs

V. WORKFLOW

Here’s how the system works: it locks down sensitive data by encrypting it first, then hides it inside images using SM4 encryption and LSB steganography—all on a standalone Raspberry Pi 4. You get security, stealth, and solid data integrity, right out of the box.



Step 1: System Landing & User Entry

You land on the main page, greeted by a quick intro to the Quantum Resistance Cyber Security Apparatus. From here, you can register or log in. Only users with valid credentials move forward—no shortcuts.



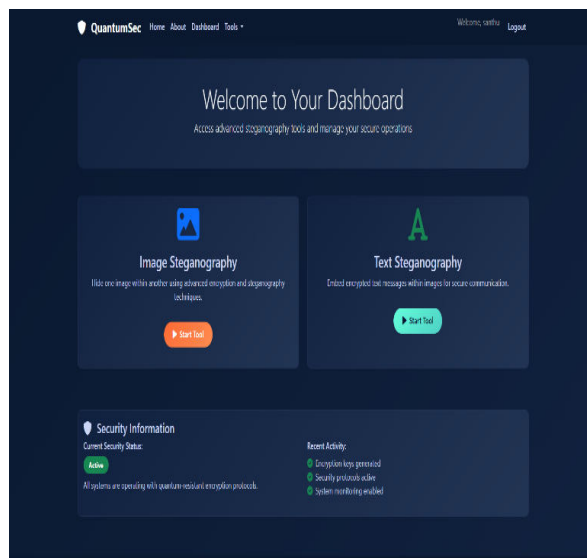
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Step 2: Secure User Dashboard

Once you're in, the dashboard lays everything out: security tools, system status, and your recent activity. You pick between Image Steganography or Text Steganography. The dashboard also shows you what's been encrypted lately, so you always know what's going on.



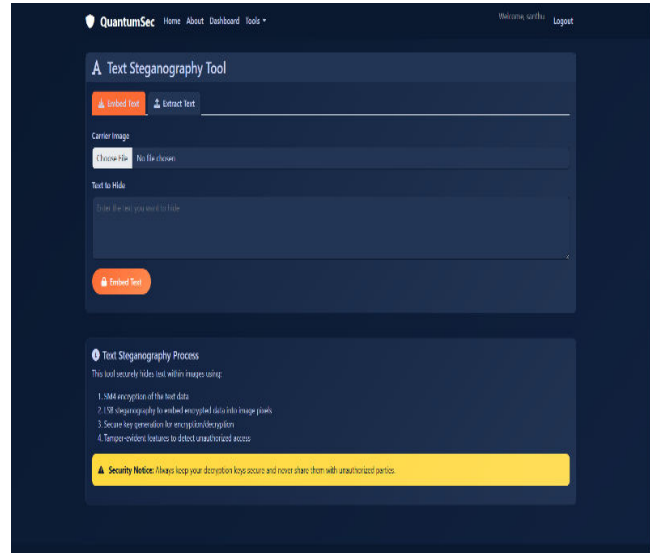
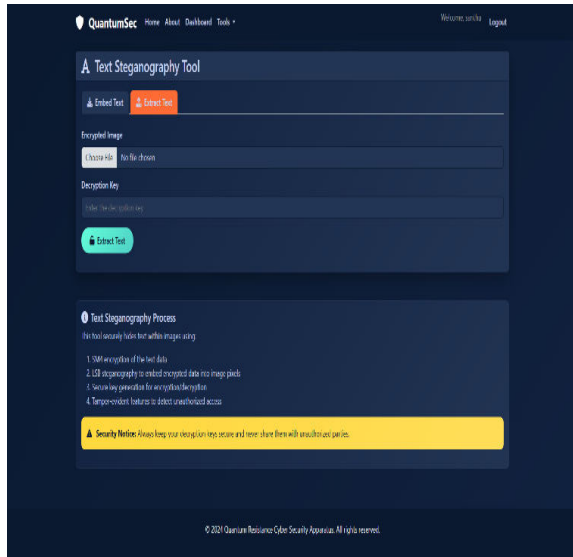
Step 3: Text Steganography – Input Selection

Ready to hide something? You grab the Text Steganography tool, upload a carrier image, and type in your secret text. That's all the system needs before it starts encrypting.



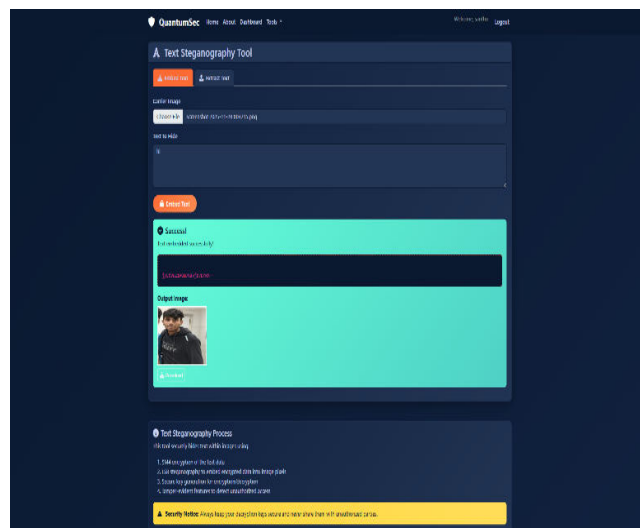
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Step 4: Encryption & Text Embedding

Your text runs through SM4 encryption. Then, the system tucks the encrypted data into the image using LSB steganography. You get a stego-image that looks just like the original—no obvious changes.



Step 5: Stego-Image Generation & Download

Once embedding is done, the system shows you the finished stego-image so you can check it yourself. Download it and send it wherever you need, safely.

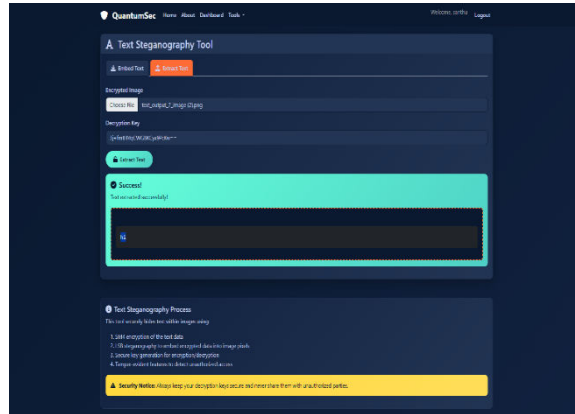
Step 6: Text Extraction & Decryption

To pull your secret out, you switch to the Extract Text option. Upload the stego-image and enter your decryption key. The system handles the rest—pulls out the hidden text and decrypts it for you.



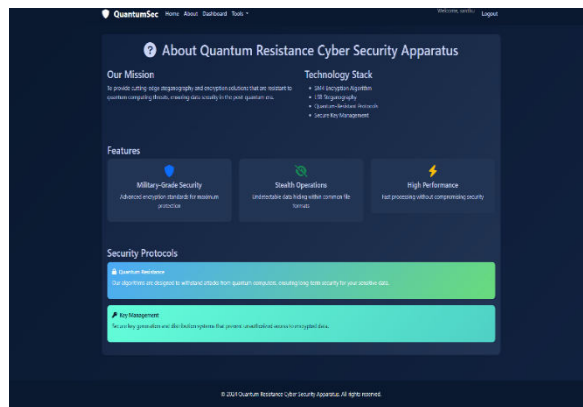
International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)



Step 7: System Information & Security Overview

Curious about the tech? The About section explains the goals, security features, and the tech stack, including quantum-resistant protocols. This part drives home how reliable and practical the system is.



VI. RESULTS

1. Functional Results

- SM4 handles encryption and decryption accurately.
- LSB steganography works well for embedding and extraction.
- Stego-images keep high quality (PSNR > 45 dB).
- LCD gives you real-time system feedback.
- Flask UI runs smoothly on different devices.
- SQLite keeps logs safe.

2. Output Examples

- Stego-images look just like the originals.
- Corrupted or tampered images get flagged with CRC/HMAC.
- Wrong keys don't get through.

3. Performance Evaluation

- Embedding and extraction take less than a second for standard images.
- System works offline, no problem.
- Integrity checks are robust and reliable.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

VII. CONCLUSION

This Quantum-Resistant Cyber Security Apparatus brings together strong SM4 encryption and stealthy LSB steganography for a secure, portable way to communicate. The Raspberry Pi keeps things mobile and works offline, which makes it great for the field. Integrity checks, logging, and live LCD updates round out the system, boosting reliability and usability. In the end, the device lives up to its purpose: quantum-resistant security, secret communication, and solid validation on extraction.

VIII. FUTURE SCOPE

There's plenty of room to grow:

- Adding post-quantum algorithms like Kyber and Dilithium.
- Adaptive steganography with DCT/DWT.
- AI to spot tampering automatically.
- A mobile app for remote use.
- Secure key storage in the cloud.
- Video steganography support.
- Multi-user authentication and roles.

REFERENCES

- [1] W. Li and J. Zhao, "An Improved Implementation of SM4 Algorithm for Secure Communication," IEEE Access, vol. 8, pp. 115920–115930, 2020.
- [2] H. Wang, L. Zhang, and X. Chen, "Quantum-Safe Cryptography: Survey and Future Research Directions," IEEE Communications Surveys & Tutorials, vol. 23, no. 3, pp. 1974–2003, 2021.
- [3] A. K. Singh and P. Kaur, "A Review on Image Steganography Techniques," IEEE International Conference on Computing, Communication and Security (ICCCS), pp. 1–6, 2018.
- [4] D. Das and N. Reddy, "Advanced Steganalysis and Detection of LSB-Based Steganography," IEEE International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 333–340, 2022.
- [5] R. Brown, T. Morley, and A. Al-Ali, "Impact of Quantum Computing on Classical Cryptographic Algorithms," IEEE Transactions on Quantum Engineering, vol. 2, pp. 1–12, 2021.
- [6] S. Ahmed and T. A. Taha, "Lightweight Cryptographic and Cybersecurity Solutions on Raspberry Pi Devices," IEEE International Conference on Smart Computing (SMARTCOMP), pp. 113–120, 2022.
- [7] Y. Murthy, K. Rao, and R. M. Rao, "Hybrid Crypto-Stego Systems for Secure Data Hiding," IEEE International Conference on Computing, Electronics & Communications Engineering (iCCECE), pp. 45–50, 2020.
- [8] S. Patel and M. Singh, "LSB Image Steganography for Confidential Data Transmission," IEEE International Conference on Advances in Computing and Communication Engineering (ICACCE), pp. 643–648, 2019.
- [9] D. O'Neil and A. Kumar, "Web-Based Secure Information Hiding Systems: A Survey," IEEE International Conference on Computational Intelligence and Communication Networks (CICN), pp. 562–567, 2021.
- [10] Raspberry Pi Foundation, Raspberry Pi 4 Model B Technical Documentation, Raspberry Pi Trading Ltd., 2020. [Online]. Available: <https://www.raspberrypi.org/documentation/>
- [11] G. S. Meshram and P. L. Ramteke, "Performance Analysis of LSB Steganography in Spatial Domain," IEEE International Conference on Inventive Systems and Control (ICISC), pp. 728–733, 2020.
- [12] National Institute of Standards and Technology (NIST), "Post-Quantum Cryptography Standardization," NIST PQC Project, 2022. [Online]. Available: <https://csrc.nist.gov/projects/postquantum-cryptography>
- [13] M. Kharrazi, H. T. Sencar, and N. Memon, "Image Steganography: Concepts and Practice," IEEE Signal Processing Magazine, vol. 31, no. 3, pp. 66–76, 2019.
- [14] B. Schneier, Applied Cryptography: Protocols, Algorithms, and Source Code in C, 20th Anniversary ed. New York, NY, USA: Wiley, 2015.
- [15] P. Yi, L. Feng, and X. Li, "Secure Data Storage and Access Control Techniques for IoT Using Lightweight Encryption," IEEE Internet of Things Journal, vol. 8, no. 5, pp. 3334–3346, 2021.
- [16] J. Fridrich, "Applications of Data Hiding in Digital Images," IEEE International Conference on Multimedia and Expo (ICME), pp. 1–6, 2018.
- [17] S. Lin and G. Sun, "Efficient SM4 Implementation and Performance Optimization on ARM Platforms," IEEE International Conference on Information Systems and Computer Aided Education (ICISCAE), pp. 315–320, 2021.



International Journal of Innovative Research in Computer and Communication Engineering (IJIRCCE)

(A Monthly, Peer Reviewed, Refereed, Scholarly Indexed, Open Access Journal)

- [18] D. Boneh and V. Shoup, A Graduate Course in Applied Cryptography, Stanford University, 2020. [Online]. Available: <https://crypto.stanford.edu/>
- [19] J. Zhang and T. Kim, "Lightweight Steganographic Approaches for Embedded Systems," IEEE Embedded Systems Letters, vol. 12, no. 2, pp. 45–50, 2020.
- [20] M. J. Atallah, "Steganographic Techniques for Covert Communication," IEEE International Conference on Information Hiding, pp. 1–10, 2019.
- [21] Dr. Malatesh S. H., S. Kattimani, P. Pallabavi, V. A. P., and D. M. G., "Intruder Detection and Protection System," International Journal of Innovative Research in Technology (IJIRT), vol. 11, no. 12, p. 6287, May 2025, ISSN: 2349-6002.



INTERNATIONAL
STANDARD
SERIAL
NUMBER
INDIA



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

 9940 572 462  6381 907 438  ijircce@gmail.com



www.ijircce.com

Scan to save the contact details