Enhancing the Security of Data Transmission in Networks by Integrating Error Correction Mechanisms with Advanced Cryptographic Protocols

Dr. Punitha K
Associate Professor, SCOPE
Vellore Institute of Technology,Chennai
punitha.k@vit.ac.in

Tanmay Manoj Wani
School of Computer Science and Engineering
Vellore Institute of Technology,Chennai
tanmaymanoj.wani2021@vitstudent.ac.in

Patel Smit Mineshkumar
School of Computer Science and Engineering
Vellore Institute of Technology,Chennai
smitmineshkumar.patel2021@vitstudent.ac.in

Khadadiya Chinkal Mukeshkumar
School of Computer Science and Engineering
Vellore Institute of Technology,Chennai
chinkal.khadadiya2021@vitstudent.ac.in

I. ABSTRACT

In today's digital world, secure data communication over networks becomes an essential part of people's work and life, from streaming services to transmitting sensitive telecommunications through satellite; the greater dependence on data networks makes data protection from misuse and access into unauthorized hands more relevant. Data breach can bring about serious consequences such as monetary losses as well as harm to individuals and organizations.

Error correction and encryption are the critical tools for securing data transmission. Simple and effective methods of error correction, apart from the one presented above, are Hamming codes, Low-Density Parity-Check codes, and Reed-Solomon codes, designed to reduce the errors during transmission, while symmetric-key-based confidentiality algorithms like DES and AES have been devised to offer strict confidentiality. Still, challenges face these techniques-mainly system complexity, latency, and compatibility problems-and other aspects applicable to cryptographic solutions, such as key management and scalability, become major concerns in modern interconnected systems.

This project addresses these new challenges by integrating advanced error correction mechanisms with robust cryptographic protocols that enhance the integrity and security of the data during the transference process by developing a MATLAB framework. It has to present a general solution that protects transmitted data through equilibriums of trade-offs between system complexity, performance, and security.

This work will attempt to design a secure and effective model for data transfer based on the prevailing algorithms meant for evaluation and deduced customized improvement. The proposed framework considers improving performance, compliance with regulations, and compatibility with heterogeneous network environments, making better security possible over data transfers in modern communication systems.

II. Introduction

Various ways exist to secure the data and its integrity. The two most common methods are error correction and encryption of data. Techniques of error correction are those that detect and correct errors occurring during the transmission due to noise or interference in the communication channel [6]. Data encryption is a method that gets data scrambled so that unauthorized entry is not possible [7]. This combines two techniques into a significant solution that can be used to secure data transmission in networks [8]. With the application of error correction initially, which guarantees the correctness and completeness of the data during transmission, and finally encrypting, which would protect access to it, data can now be communicated securely over the network. Again, though such systems depend on error correction and encryption techniques chosen, details in implementation might make all the difference. Also, security, performance and complexity will have trade offs to be considered very carefully while devising such systems. Hence, our project focuses on determining the best combination of the encryption algorithm and the error correction technique to determine the most secure model. The security model to be evolved in MATLAB will make it assess the balance that can be achieved best with regard to security, cost, and efficiency through various parameters like the threat models, data sensitivity, performance requirements, and regulatory compliance.

Challenges: Several challenges and issues are associated with the problem statement, including:

- **Selection of Error Correction and Data Encryption Techniques:** Choosing the correct error-correction and encryption techniques is crucial for success in the security model. With many algorithms, and each type having its pros and cons, choosing the appropriate combination is a very important consideration of what is best suited to the specific need of the system.
- **Regulatory Compliance:** These designs would need to consider factors such as regulatory compliance with HIPAA or GDPR. This may further add complexity to the system, adding standards it has to meet with both security and performance.
- **Performance Impact:**Any kind of error correction and encryption techniques may impair system performance. The best balance between security and performance must be achieved to ensure that the system is managed in an efficient manner without losing on its security-related features.
- **Complexity:** The system complexity of multiple error correction and encryption algorithms becomes quite high. Added complexity may lead to increased costs, longer development time, and potential errors. Conversely, the system will be open to attacks and break-ins if proper protection is not provided.
- **Attack Mitigation:** Because there is also a variation of attacks against encryption algorithms, such as brute force, the system must be regularly updated and tested in a manner to ensure that the defenses provide robustness and resilience.

III. LITERATURE REVIEW

Although there are several widely used methods of error correction and data encryption, most of the models tend to use only one algorithm each, which makes them vulnerable to advanced attacks. Here is a list of some of the widely used algorithms and techniques used in different network applications:

Error Correction Techniques:

- **Forward Error Correction (FEC):** It embeds redundant data into the message to be transmitted for loss or corruption recovery. This category has examples such as Reed-Solomon codes, which are known for its robustness in noisy/disturbed environments. Other techniques that are included in this category are Convolution codes and LDPC [11].
- **Automatic Repeat Request (ARQ):** ARQ re-transmits corrupted data during transmission. It can be used alongside FEC to further enhance security and data integrity.
- **Checksum:** The technique adds a checksum value to the message, which is utilized by the receiver to verify integrity of the data by comparing the calculated checksum with the received checksum value.
- **Cyclic Redundancy Check (CRC):** A more advanced version of the checksum, CRC divides the message into blocks, each generating a checksum, which the receiver then compares with the calculated checksum to detect errors.

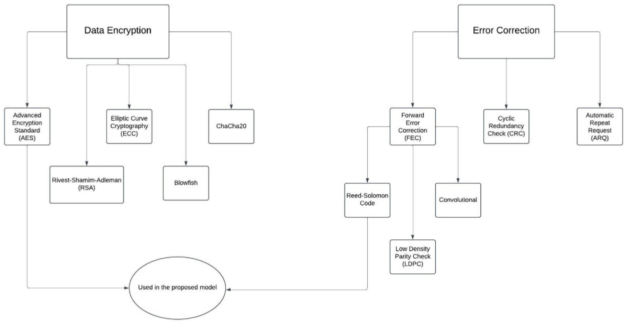
Data Encryption
Error Correction
Advanced Encryption Standard (AES)
Elliptic Curve Cryptography (ECC)
ChaCha20
Rivest-Shamim-Adleman (RSA)
Blowfish
Forward Error Correction (FEC)
Cyclic Redundancy Check (CRC)
Automatic Repeat Request (ARQ)
Reed-Solomon Code
Convolutional
Low Density Parity Check (LDPC)
Used in the proposed model

Fig. 1. Flowchart of Algorithm

The most significant challenge when trying to combine data encryption with error correction is that techniques for encrypting data are susceptible to attacks that target methods of error correction. For instance, if an attack compromises an error correction algorithm like Hamming, the associated encryption algorithm (e.g., Blowfish) could become susceptible to attack as well.

Data Encryption Algorithms:

- **Advanced Encryption Standard (AES):** AES is more secure than Data Encryption Standard (DES) due to its ability to support key sizes of 128, 192, or 256 bits.
- **Rivest-Shamir-Adleman (RSA):** In RSA, the public key is used for encryption, and anyone can encrypt the data, but decryption can only be performed using the private key, which is kept secret by the owner.
- **Elliptic Curve Cryptography (ECC):** ECC provides the same security level as RSA but uses smaller key sizes, making it more efficient in terms of memory and computational requirements.
- **ChaCha20:** A fast and secure symmetric cipher algorithm used for encryption.
- **Blowfish:** A symmetric encryption algorithm known for its strong security, suitable for applications requiring high performance and low memory usage.

Combinations of Encryption and Error Correction Techniques:

- **Encryption followed by Error Correction:** In this approach, the encrypted data is protected from unauthorized users before error correction is applied. However, this can cause some error patterns to be neglected, reducing the effectiveness of error detection.
- **Error Correction followed by Encryption:** Here, error detection and correction are applied first, improving error handling. However, a compromised encryption key could make the original data vulnerable to unauthorized access.
- **Encryption and Error Correction Simultaneously:** This optimized approach combines both error correction and data encryption, leading to improved performance.

However, it demands more processing resources and results in a complex implementation.

- **Hybrid Encryption and Error Correction:** This technique protects against both errors and unauthorized access but is complex to implement and may require special hardware. For example, using a block cipher for encryption combined with Reed-Solomon code for error correction.

In conclusion, the best combination of encryption and error correction really depends on the system-specific requirements. Although such a method offers protection at several levels, it usually complicates matters and causes an increase in bandwidth usage, and therefore efficiency down, making such systems rather uneasy to design.

IV. Proposed Methodology

Our proposed methodology will use error correction and data encryption methodologies so that the efficiency of data communication in satellite networks is guaranteed through both integrity and confidentiality: steps with different emphases on different sides of the procedure.

1) Choose Error Correction Techniques to Implement The first step would be the most in-depth research to choose appropriate error correction techniques best along with the encryption algorithm selected and compatible with the environment of a network. Error correction methods can be Forward Error Correction, Automatic Repeat Request, and other error correction methods; they must be selected according to their ability to handle transmitted data with successful corrections. It hence demands an understanding of how the error correction mechanism goes about interacting with the encryption mechanism, especially when the latter employs the same key. For instance, a key-based error-correcting mechanism that is known to be vulnerable to recovery attacks can compromise the security system if applied in a cryptosystem with an encryption algorithm vulnerable to such attacks. Hence, error correction techniques have to be particularly chosen for robustness against such vulnerabilities, and algorithms working on separate keys or being more immune to attacks may have to be employed.
2) Select an Encryption Method: Once the error correction method is chosen, the choice of encryption technique appropriate to the specific type of data that needs to be transmitted must be decided by the user. This depends on the level of security requirements, the class of data one will be transferring, and the computation powers on the satellite network. Different algorithms like AES, RSA, and ECC have different capabilities in terms of security level and computational complexity. For example, AES is more expensive in terms of resources, though it does present robust encryption; while algorithms, such as ECC, have a smaller size to generate keys, which makes them more suitable in limited-computational resource environments. Thus, the choice of encryption technique balances the necessity to provide security over data with the constraints within the satellite network, bridging both security and performance.
3) Integrate the techniques to form a combined approach: With these two techniques selected, the next step would be to interlace them in such a way as to optimize the overall security of data transmission. The two techniques can be applied either sequentially or simultaneously. For example, the encryption algorithm can be applied first, and then the error correction technique. Alternatively, both techniques can be implemented concurrently. The requirement is that when encryption meets an error correction method, then both methods must protect the data against unlawful access and the integrity of data integrity with the error correction method will be maintained during transmission. The amalgamation should reduce the vulnerabilities of both techniques, making the whole system as secure as possible. This phase may also fine-tune the combination so it's both effective as well as efficient in satellite communication.
4) Testing the Network and Simulating on MATLAB: Once the error correction and encryption techniques are integrated, they need to be experimented upon in all aspects so that the system works as intended. Simulations must be performed using MATLAB or similar software where scenarios regarding noise, interference, and security threats can be created. The testing stage involves various elements, including the integrated technique complexity, ability to resist real-world threats, system overall performance, and compliance of the network with applicable regulatory standards (like HIPAA, and GDPR). Running these simulations allows possible weaknesses in the integrated system to be indicated and necessary adjustments made before the deployment to improve security, efficiency, and compliance.
5) Continuous Monitoring: The final one sets up a system that continues to monitor the performance and security of the network over time. Real-time monitoring tools help track issues or vulnerabilities that might arise in the operation of the system. Continuous monitoring is also key in ensuring that any error in data transmission is identified and corrected immediately, hence preventing a loss or corruption of data. It also updates the error correction and encryption techniques regularly to keep ahead of changing threats. Whenever new vulnerabilities or types of attacks appear, the system needs to be updated according to the threat. However regular testing and monitoring ensure the systems comply with regulatory standards and the network overall remains strong and reliable throughout its lifecycle.

In summary, the method follows a systematic approach; namely first data is encrypted with a private key, and thereafter encoded with an error correction algorithm. At the receiver's end the data is decoded correct transmission errors and then decrypted using the private key that was used for encryption.

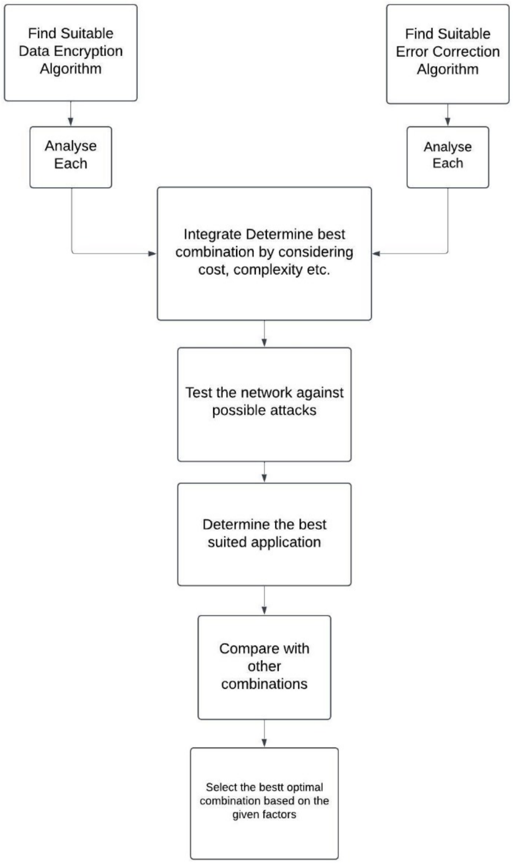
Find Suitable Data Encryption Algorithm
Find Suitable Error Correction Algorithm
Analyse Each
Analyse Each
Integrate Determine best combination by considering cost, complexity etc.
Test the network against possible attacks
Determine the best suited application
Compare with other combinations
Select the bestt optimal combination based on the given factors

Fig. 2. Flowchart of Algorithm

This allows secure data to be maintained during the period of transmission and prevent access by unauthorized parties as well as errors in transmission.

Justification:

The proposed method is an integration of error correction and data encryption in such a way that it provides an all-inclusive system that will be secure for the transmission of data. Error correction techniques are significantly used in the maintenance of data integrity due to errors that may result from the presence of noise or interference during the transmission process. Since these techniques ensure error-free data, the chances of breaches and the exposure of vulnerabilities in the communication system are minimal.

At the same time, encryption methods secure data confidentiality by allowing access only to intended users. Because encrypted data becomes a secured form that no one outside will be able to access, it cannot be intercepted or modified by such malicious persons.

These two techniques, work on the methodology basis in bringing a robust and reliable system that can handle commonly found transmission errors while protecting against security threats. In this way, the double-layered process strengthens the core processes of communication, making sure that data is accurate, secure, and resilient against possible attacks. Thus, it represents an ideal approach for critical applications like satellite network communication.

V. Implementation

The proposed methodology was implemented using the Python programming language. The reasons behind this were its flexibility and libraries used for simulation and analysis. It is heavily based on error correction codes due to their operation at lower layers of the network and direct accuracy with regard to data that is being sent across different networks. This methodology focuses on finding combinations of error correction codes that will yield maximum accuracy by compensating for the flaws of an individual one.

1) Bit Error Rate (BER): BER quantifies the probability of a single bit being received incorrectly during transmission. Lower BER values indicate higher accuracy. All three codes—Hamming, Reed-Solomon, and Turbo—contribute to reducing BER, but their effectiveness varies based on transmission conditions and implementation.
2) Frame Error Rate (FER):FER measures the probability of an entire frame (or block) of data being received with errors. This metric is crucial in applications requiring high data integrity. Reed-Solomon and Turbo codes outperform Hamming codes in reducing FER, particularly in noisy or interference-prone environments.
3) Overhead:Overhead refers to the additional bits introduced by error correction codes for redundancy. Minimizing overhead is important for efficient bandwidth utilization. Among the codes: Hamming codes introduce the least overhead. Reed-Solomon codes have moderate overhead. Turbo codes have the highest overhead due to their complex encoding structure.
4) Latency: Latency accounts for the time delay introduced by the encoding and decoding processes. Lower latency is preferred for real-time or time-sensitive communication.Hamming codes have the lowest latency due to their simplicity. Reed-Solomon codes have moderate latency. Turbo codes exhibit the highest latency because of their iterative processing.

VI. OBSERVATION

1) **Reed-Solomon + AES + RSA**

   This combination utilizes Reed-Solomon error correction, RSA encryption, and AES encryption to achieve secure and reliable data transmission. Reed-Solomon (RS) error correction operates on data blocks, adding redundant information that enables the receiver to detect and rectify transmission errors. In this methodology, RS incorporates error correction bytes into the plaintext prior to encryption, ensuring recovery of the original data even with transmission faults.

   Advantages and Limitations: Reed-Solomon stands out with its precision in detecting errors at the bit and frame levels. This hybrid model efficiently combines

AES for encrypting large volumes of data and RSA for secure key exchange. On the downside, assessing BER and FER requires additional signaling in the data stream. RS-based approaches introduce complexity and hybrid encryption schemes can significantly increase computational costs.

Performance Characteristics:

a) Bit Error Rate (BER): The combination of Reed-Solomon error correction with AES and RSA encryption significantly reduces the BER compared to using these techniques individually. Reed-Solomon's ability to detect and correct multiple errors within a block ensures improved reliability in transmission. However, the actual BER values are influenced by the coding parameters, signal-to-noise ratio (SNR), and channel conditions, requiring simulations or real-world tests for precise evaluation.

b) Frame Error Rate (FER): The Frame Error Rate, which measures the likelihood of an entire data frame being erroneous, is also reduced by the error correction capabilities of Reed-Solomon. By addressing errors across blocks, this method minimizes frame-level transmission errors. FER performance depends on factors like block size, the nature of transmission noise, and modulation techniques.

c) Latency: The latency in this hybrid approach is influenced by the computational complexity of encoding and decoding processes in Reed-Solomon and the encryption mechanisms of AES and RSA. These combined techniques lead to higher processing times, especially during encoding, decoding, and key exchange. While this added latency may impact real-time systems, it ensures improved security and error correction.

d) Overhead: The use of Reed-Solomon codes introduces overhead in the form of redundant data added for error correction. This redundancy increases the overall size of the transmitted data, which, in turn, requires more bandwidth. However, the added overhead is a trade-off for achieving higher reliability and error resilience, making it essential for applications where data integrity is critical.

2) **Checksum, Reed-Solomon, and AES-GCM** This combination employs a checksum for basic error detection, Reed-Solomon for advanced error correction, and AES-GCM for encryption and authentication. The checksum ensures data integrity at a fundamental level, while Reed-Solomon corrects errors even in challenging conditions. AES-GCM provides robust encryption and authentication, ensuring both security and integrity.

Performance Metrics:

a) Bit Error Rate (BER): The combination of checksum and Reed-Solomon coding enhances error detection, reducing the BER significantly. The checksum identifies errors during transmission, while Reed-Solomon corrects them, ensuring a lower rate of bit-level errors. However, the effectiveness depends on the transmission channel and environmental conditions.

b) Frame Error Rate (FER): The FER benefits from this combination as errors detected by the checksum and corrected by Reed-Solomon minimize the chance of entire frames being corrupted. This makes the approach suitable for scenarios requiring high data reliability. However, FER improvements rely on the robustness of the error correction and the quality of the communication channel.

c) Latency: The inclusion of Reed-Solomon and AES-GCM introduces processing delays due to the computational requirements of encryption, decryption, and error correction. While the checksum has minimal impact on latency, the combination of these techniques can lead to increased transmission times, especially for real-time or large-scale applications.

d) Overhead: The addition of a checksum and Reed-Solomon coding increases overhead due to the inclusion of error detection and correction data. AES-GCM adds further overhead by incorporating encryption and authentication tags. While this results in larger data sizes, the trade-off is enhanced security, integrity, and error resilience, which are vital for critical applications.

This combination provides enhanced error detection and encryption, achieving a balance between efficiency and security. While the overhead and latency trade-offs need to be managed, the technique is suitable for systems requiring a mix of robust encryption and reliable error correction.

RESULT AND CONCLUSION

The combination of Checksum, Reed-Solomon, and AES-GCM provides a straightforward approach to error detection and strong error correction capabilities. It is ideal for scenarios where simplicity, speed, and security are paramount, such as in real-time applications. On the other hand, Reed-Solomon, AES with RSA offers more precise error detection and efficient error correction, albeit with increased complexity. This second combination is suitable for systems that require detailed error analysis and secure key exchanges, but it may not be as fast as the first one.

When prioritizing fast processing and low latency with a satisfactory level of security, the first combination (Checksum, Reed-Solomon, and AES-GCM) is recommended. It excels in environments like video streaming, messaging applications, and large file transfers, where real-time processing is essential. In contrast, the second combination (Reed-Solomon, AES, and RSA) is preferable in systems that require more comprehensive error analysis, such as satellite communications, financial

Attribute	**RS + AES + RSA**	**RS + AES-GCM**
Encryption Algorithm	AES (symmetric) + RSA (asymmetric)	AES-GCM (symmetric with authenticated encryption)
Error Correction	Reed-Solomon (RS)	Reed-Solomon (RS)
Encryption Speed	Slower, due to RSA's computational intensity	Faster, as AES-GCM is optimized for speed
Decryption Speed	Slower, RSA requires more processing for key operations	Faster, AES-GCM provides efficient authenticated decryption
Security Level	High (asymmetric RSA for secure key exchange)	High (AES-GCM provides data confidentiality and integrity)
Authentication	Not inherently provided; requires additional MAC or HMAC	Built-in authentication with GCM (Authenticated Encryption)
Error Detection	Reed-Solomon error detection	Reed-Solomon error detection
Error Correction Capability	Strong, Reed-Solomon corrects errors in noisy environments	Strong, same Reed-Solomon-based error correction
Overhead	Higher overhead due to RSA key size and additional security metadata	Lower overhead since AES-GCM is more compact with combined encryption/authentication
Latency	Higher due to RSA's computational demand	Lower, AES-GCM reduces latency through faster processing
Suitable for Real-Time Use	Limited due to higher latency and overhead	Good, as AES-GCM offers low latency and high speed
Best-Suited Applications	Applications requiring secure key exchange and non-repudiation, e.g., secure financial transactions, medical record transfers, satellite communication	Applications needing low-latency, real-time security, e.g., video streaming, instant messaging, large file transfers
Complexity	Higher complexity due to dual encryption algorithms (AES + RSA)	Lower complexity, single symmetric encryption with authenticated encryption
Resource Requirements	High (RSA needs significant processing power and memory)	Moderate, AES-GCM is lightweight and efficient on resources
Quantum Resistance	Moderate (RSA is vulnerable, AES is resistant with longer keys)	Good (AES-GCM is generally more resistant than RSA, particularly for symmetric encryption needs)

TABLE I

Comparison of RS + AES + RSA and RS + AES-GCM Features

transactions, and the secure transmission of medical records and broadcasting systems.

An alternative powerful combination would be **Reed-Solomon + AES + RSA + Turbo**. This integrated approach, often used in high-reliability domains like wireless and satellite communications, secure file transfer protocols, and digital rights management (DRM), ensures error-free and secure data transmission. In satellite communication, for instance, it offers robust protection for sensitive data during transmission and safeguards it against unauthorized access. However, this combination is typically reserved for environments where precise error correction is critical, such as in satellite communications, due to its high overhead and latency. Therefore, its use is often limited to situations where the accuracy and security of data are non-negotiable.

In conclusion, combining error correction and encryption techniques forms a powerful solution for securing data transmission. By integrating error-correction algorithms like Reed-Solomon with encryption protocols like AES and RSA, we create a secure and reliable system that ensures both data accuracy and confidentiality. This combination not only enhances the reliability of digital communication but also fortifies the system against data loss and unauthorized access. As the demand for secure and accurate communication grows, leveraging these combined techniques will be pivotal in ensuring the integrity and protection of sensitive data across diverse applications.

References

1) A. Couvreur, M. Lequesne, On the security of subspace subcodes of reed–solomon codes for public key encryption, IEEE Transactions on Information Theory 68 (1) (2021) 632–648.
2) S. Mahmood, S. M. Mohsin, S. M. A. Akber, Network security issues of data link layer: An overview, in: 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2020, pp. 1–6. doi:10.1109/ iCoMET48670.2020.9073825.
3) A. M. Abdelaziz, E. Abdelwanees, A. D. Elbayoumy, Securing the space data link communication protocol of earth observation satellites (2019). doi:10.1109/ICICIS46948.2019.9014846.
4) N. C, Survey on network security with cryptography, https://www.ijser.org/researchpaper/Survey-on-Network-Securitywith- Cryptography.pdf (2019).
5) I. B. Djordjevic, Physical-layer security and quantum key distribution, Springer, 2019.
6) G. Yang, L. Dai, Z. Wei, Challenges, threats, security issues and new trends of underwater wireless sensor networks, Sensors 18 (11) (2018) 3907.
7) G. Yang, L. Dai, Z. Wei, Challenges, threats, security issues and new trends of underwater wireless sensor networks, Sensors 18 (11) (2018) 3907.
8) Z. Chen, L. Yin, Y. Pei, J. Lu, Codehop: physical layer error correction and encryption with ldpc-based code hopping, Science China Information Sciences 59 (2016) 1–15.
9) O. S. Younes, Securing arp and dhcp for mitigating link layer attacks, S¯adhan¯a 42 (2017) 2041–2053.
10) B. Tahir, S. Schwarz, M. Rupp, Ber comparison between convolutional, turbo, ldpc, and polar codes, in: 2017 24th

International Conference on Telecommunications (ICT), 2017, pp. 1–7. doi:10.1109/ICT.2017.7998249.

11) J. M. Hamamreh, M. Yusuf, T. Baykas, H. Arslan, Cross mac/phy layer security design using arq with mrc and adaptive modulation, in: 2016 IEEE Wireless Communications and Networking Conference, IEEE, 2016, pp. 1–7.

12) W. H. Jeong, B.-G. Yeo, K.-H. Kim, S.-H. Park, S.-W. Yang, J.-S. Lim, K.-S. Kim, Performance analysis of the encryption algorithms in a satellite communication network based on h-arq, The Journal of The Institute of Internet, Broadcasting and Communication 15 (1) (2015) 45–52.

13) L. Ning, L. Kanfeng, L. Wenliang, D. Zhongliang, A joint encryption and error correction method used in satellite communications (2014). doi:10.1109/CC.2014.6825260.

14) N. Islam, Z. Shahid, W. Puech, Denoising and error correction in noisy aes-encrypted images using statistical measures (2016). doi:https://doi.org/10.1016/j.image.2015.11.003. URL https://www.sciencedirect.com/science/article/pii/S0923596515002003

15) M. Dener, O. F. Bay, Teenysec: a new data link layer security protocol for wsns, Security and Communication Networks 9 (18) (2016) 5882–5891.

16) A. M. Abukari, An enhanced error detection and correction scheme for enterprise resource planning (erp) data storage, Journal of Advances in Computer science and Maths 36 (2021). doi: 10.9734/jamcs/2021/v36i930405. URL https://journaljamcs.com/index.php/JAMCS/article/view/1602

17) E. Bertino, Data security and privacy concepts (2016). doi: 10.1109/COMPSAC.2016.89. URL https://ieeexplore.ieee.org/document/7552042

18) B. A. Forouzan, Data communications and networking (mcgrawhill) (2007).

19) W. Stallings, Data and Computer Communications (2007).

20) S. Mahmood, S. M. Mohsin, S. M. A. Akber, Network security issues of data link layer: An overview, in: 2020 3rd International Conference on Computing, Mathematics and Engineering Technologies (iCoMET), 2020, pp. 1–6. doi:10.1109/ iCoMET48670.2020.9073825.

21) D. Purwanto, Optimization of data security system control with crc (cyclic redundancy check) algorithm, Budapest International Research and Critics Institute-Journal (BIRCI-Journal) 4 (3) 4635– 4642.

22) O. Aitsab, R. Pyndiah, Performance of reed-solomon block turbo code, in: Proceedings of GLOBECOM'96. 1996 IEEE Global Telecommunications Conference, Vol. 1, 1996, pp. 121–125 vol.1. doi:10.1109/GLOCOM.1996.594345.