**A REVIEW ON RSA-HUFFMAN HYBRID: TOWARDS SECURE, COMPRESSED AND ADAPTIVE NETWORK DATA TRANSMISSION**

**Abstract***The RSA-Huffman hybrid model, which combines asymmetric encryption and lossless compression, has become a viable answer to the increasing demands for safe and effective data transfer. The review evaluates the existing RSA-Huffman technigues based on implementation by application domains(cloud, IoT, Mobile etc). It assess their real-world feasibility, security-efficiency trade-offs, and techniques. Results show improved confidentiality (e.g., high NPCR scores) and compression ratios (50–90%), but they also highlight important drawbacks, including inconsistent benchmarking, fidelity degradation in lossy implementations, computational overhead (notable latency spikes), and new vulnerability risks (e.g., quantum threats). Future developments necessitate uniform evaluation procedures, quantum-resistant modifications, and lightweight optimizations to allow for scalable deployment in resource-constrained systems, even though they are successful in particular situations.*

**Keywords**: Network Security, Cryptographic Efficiency, Huffman Coding, RSA Algorithm, Data Encryption, Data Compression, and Network Transmission

**Introduction**

Cryptography and data compression are two fundamental techniques in information security and data transmission, each serving distinct but complementary purposes. Gokulapriya & Sharmila (2021) introduced a paradigm of data security, emphasizing the multifaceted shield erected against unauthorized incursions into computer systems, personal databases, and online platforms. Their definition encapsulates the strategic deployment of protective measures to thwart both inadvertent and malevolent alterations or breaches, thereby ensuring the integrity and confidentiality of information assets. Recent research has shed light on advancements in both fields, highlighting innovative approaches to enhancing security and efficiency in various applications. The research landscape has been marked by a surge in investigations into cryptographic primitives, encryption methodologies, and protocol innovations tailored to address the exacting security demands of contemporary applications. As Wahab *et al.* (2021) elucidate, this approach not only optimizes data transmission efficiency, facilitating rapid dissemination over sluggish internet connections, but also minimizes storage footprint across diverse media platforms, enhancing accessibility and resource utilization. Moreover, advancements in cryptographic protocols have enabled secure multiparty computation (MPC) and verifiable computation.

The information security system of modern technology includes confidentiality, authenticity, integrity, nonrepudiation and access control (Gokulapriya & Sharmila 2021). Data compression is an important part of information security because compressed data is more secure and easy to handle. Effective data compression technology creates efficient, secure, and easy-to-connect data (Wahab *et al*., 2021). Data compression, the process of reducing the size of data for efficient storage and transmission, is essential for optimizing bandwidth utilization, reducing storage costs, and improving system performance.

Mechanisms that guarantee both data security and transmission efficiency are required in the modern digital world due to the exponential growth of data transmission over networks. RSA is a popular asymmetric encryption method that uses public-key cryptography to provide strong security. However, real-time applications and systems with constrained processing power have difficulties due to their computational complexity, particularly when managing huge datasets (Durge & Deshmukh, 2024). On the other hand, Huffman coding, a lossless data compression method, optimizes storage and transmission by efficiently reducing data size by allocating shorter codes to more frequent symbols (Sandhu, 2021).

A potential remedy for the dual problems of data security and transmission efficiency is the combination of RSA encryption and Huffman coding. This hybrid strategy seeks to secure data while reducing its size for transmission by first encrypting it using RSA and then compressing it using Huffman coding. According to recent research, these interconnections have the potential to improve data security and lower transmission overhead (Wahab *et al*., 2021). Optimizing the ratio of encryption strength to compression efficiency and guaranteeing compatibility and performance in a variety of network contexts are still difficult tasks, nevertheless.

The goals of this systematic review are as follows: To critically examine previous studies and applications of RSA, Huffman coding, and their hybrid models in the context of network data transmission; and pinpoint weaknesses in existing methodologies and suggest an ideal RSA-Huffman hybrid framework that improves encryption strength and data compression effectiveness for safe and effective network communication.

**Review RSA (Rivest–Shamir–Adleman) concept**

The public-key cryptosystem known as RSA (Rivest–Shamir–Adleman) is based on the mathematical difficulty of factoring big prime numbers and is used for secure data transfer. In order to provide secure communication, even via unprotected channels, it creates two keys: a public key for encryption and a private key for decryption. Particularly in protocols like TLS/SSL and digital certificates, RSA is extensively utilized for secure key exchange and digital signatures (Stallings, 2023). RSA is slower and requires more computing power than symmetric encryption techniques, despite its resilience ensuring interoperability and performance in a variety of network scenarios, as well as encryption strength and compression efficiency.

Concerns regarding RSA's long-term sustainability have increased recently as a result of developments in quantum computing. When used on a powerful enough quantum computer, Shor's technique has the potential to break RSA encryption by factoring huge integers in polynomial time (Chen *et al*., 2022). Although this is not currently possible with existing quantum computers, companies are getting ready for a post-quantum future. NIST, for instance, has started the process of standardizing post-quantum cryptography algorithms (Alagic *et al*., 2022).

Businesses are currently developing quantum-resistant cryptography systems, like PQShield. Supported by collaborations with corporate and governmental partners, PQShield's research attempts to develop encryption techniques that are impervious to quantum attacks (El Kaafarani, 2023). During the post-quantum transition, these initiatives involve transferring current RSA-based systems to hybrid models that use both classical and quantum-safe algorithms to guarantee security and continuity.

Additionally, RSA continues to be a fundamental component of cybersecurity systems and education, while its function is evolving. There is growing interest in hybrid schemes that combine RSA with more recent, quantum-resilient techniques. Particularly in resource-constrained situations like the Internet of Things, RSA can be integrated with lightweight algorithms or compression approaches to minimize processing overhead while preserving robust encryption, claim Bhardwaj and Kaushik (2021).

**Review of Huffman Coding Concept**
A lossless compression technique called Huffman coding gives longer codes to less common symbols and shorter codes to more common ones. As a result, data is represented effectively and without information loss. Widely employed in file compression formats like ZIP, JPEG, and MP3, the coding strategy is best suited for a given probability distribution (Kaur & Bala, 2021). Compact encoding and decoding are made possible by Huffman coding, which builds a binary tree based on symbol frequencies.

As data is read, the code tree is dynamically updated by adaptive Huffman coding, a variation of the original process. In real-time applications where symbol frequencies change over time, like compressing streaming data or IoT sensor outputs, this version is especially helpful. Adaptive coding can increase transmission efficiency in smart settings by reacting instantly to data variations, according to recent study by Singh and Bansal (2022). Huffman coding has been used for anomaly detection in data streams in addition to compression. In their investigation of adaptive Huffman coding for anomaly detection in audio data, for example, Malhotra and Tripathi (2021) showed how dynamic variations in code lengths might identify departures from normal signal patterns. For low-power systems that are unable to run complicated models, this combination use of anomaly detection and compression is very beneficial.

Parallel processing is also used in recent systems to enhance the performance of Huffman encoding. In large-scale data settings, Gupta and Sharma's (2023) GPU-based Huffman encoder performed noticeably better than CPU-based techniques. This development extends the use of Huffman coding to cloud computing and big data settings where processing speed is crucial.

 **Review of Related works**.

In order to secure cloud data streams, Abdulghani *et al*. (2021) investigated a hybrid strategy that combined LZMA compression with multi-layer encryption, including RSA versions. Strong ciphertext security was ensured by the methodology, which passed all NIST randomness tests with a 99% confidence level and obtained space-saving percentages of 58.63%–81.8%. However, the method had a large computational cost; compared to standalone RSA, LZMA compression increased processing latency by 22%, which limited its use in real-time in contexts with limited resources. While the framework successfully strikes a compromise between efficiency and security for cloud-sensitive data, edge deployment circumstances necessitate hardware acceleration.

In their implementation of steganography, Wahab *et al*. (2021) used adaptive Huffman coding to compress data, RSA-2048 for encryption, and DWT-LSB algorithms to insert ciphertext into cover graphics. Their approach achieved 65% average compression ratios and resisted 99% of steganalysis techniques by validating undetectability using chi-square tests and histogram analysis. However, 40% of processing time was spent on RSA key generation, and observable LSB distribution irregularities were visible in high-resolution photos. When employing a variety of cover media to reduce the chances of statistical detection, the method is still feasible for steganographic applications.

A cloud-IoT security model was suggested by Ray *et al*. (2022) that identifies sensitive data segments, applies Huffman compression, and encrypts them using RSA-optimized key management. The hybrid eliminated 97% of leakage attempts and cut down on data exposure time by 50%. However, key management depended on centralized authorities, resulting in single points of failure, and heuristic-based data categorization incorrectly classified 12% of non-critical data as high-risk, increasing processing overhead. Although machine learning has to be improved to lower false positives, the method is still workable for focused security.

Using blockchain logging for IoT data sharing, logistic chaotic maps for key generation, and integrated Huffman compression, Priyadharshini & Canessane (2022) improved RSA. In order to achieve 22% faster encryption than regular RSA and to withstand 99.8% of brute-force attacks, their methodology used chaotic diffusion to increase key randomness prior to asymmetric encryption. Limitations included 300 ms/tx latency caused by blockchain and 15% key synchronization failures due to chaotic sensitivity. Although this multi-layer architecture is appropriate for critical infrastructure, volatile networks require synchronization methods.

Suo *et al*. (2024) used DWT-SVD watermarking, RSA-3072 for key distribution, and compressive sensing (CS) for simultaneous compression-encryption to create a reversible IoT system. Their approach produced 85% compression with PSNR > 48 dB reconstruction; however, under noise attacks, watermark accuracy fell to 78% and CS reconstruction failed at >15% packet loss. Although the multitiered approach improves IoT privacy, it performs poorly in networks with loss.

Wiaya *et al*. (2022) achieved 80% to 90% compression ratios on big corpora by using Huffman coding with unary codes for high-ratio text compression. Because the study only used post-compression security measures and lacked integrated encryption, unary coding increased runtime for files larger than 1 GB by 45%. Despite being the cornerstone of pre-encryption size reduction, the technique is still lacking as a stand-alone security measure.

Using pixel prediction and chaotic diffusion for joint compression-encryption, Zhang *et al*. (2021) combined Chen hyperchaotic encryption with context-adaptive lossless image coding (CALIC). Their approach beat JPEG-LS in compression ratios and produced nearly-ideal entropy (7.999), but CALIC was incompatible with video streams and hyperchaotic systems used 50% more memory than RSA. The method lacks multimedia versatility but excels at static images.

In order to enable encrypted data processing at edge nodes, Patidar *et al*. (2024) developed an edge-IoT system that uses RSA-based homomorphic encryption after lossy SZ 2.1 compression for sensor data. By using energy-aware scheduling algorithms, their approach reduced latency by 28% and energy usage by 30% when compared to conventional pipelines. Lossy compression resulted in a trade-off in reconstruction fidelity (PSNR < 28 dB), while homomorphic processes led to a 40% increase in server load. The concept is effective for non-critical IoT streams, but it needs quality-preserving embedding techniques for premium publication and lossless adaptations for precision-sensitive sectors like medical imaging.

The joint approach effectively enables confidential copyright-protected distribution, but requires quality-preserving embedding techniques for premium publishing. Singh & Thakur (2021) developed a document security system using Huffman coding for text compression, RSA-2048 for encrypting compressed data and digital watermarks, and DCT-based watermark embedding. However, their methodology achieved 45%–55% size reduction with 95% watermark resilience against removal attacks, but introduced 35% encryption time overhead and pixelation in high-resolution ones.

A meta-analysis of 120 RSA-Huffman image encryption investigations was carried out by Singh & Singh (2022), who assessed methods using time complexity benchmarks, compression ratios (CR), NPCR, and UACI. Their survey revealed variable testing parameters (e.g., different key sizes/image datasets) that complicate comparisons, but it also reported median CR of 50%–70% and NPCR > 99.5% across studies. Despite the security benefits, RSA hybrids demonstrated a 20% longer runtime than symmetric alternatives, underscoring the necessity for consistent evaluation frameworks.

Canonical Huffman coding using Golomb-Rice unary codes was used by Wiaya *et al*. (2022) for text compression. Compression ratio (CR) and throughput measures were used to test the methodology on a variety of corpora. Results indicated 80% to 90% CR, but because of unary coding cost and the need for separate security layers due to the lack of integrated encryption, runtime rose by 45% for files larger than 1GB. Although it lacks full security integration, this provides fundamental compression efficiency.

By combining chaotic S-Boxes and pseudorandom keystream generators (CLM) with adaptive Huffman coding, Bao *et al*. (2023) presented a chaos-adaptive Huffman approach. Although their one-step approach saved space and had 99% NIST randomness confidence, it was more susceptible to chosen-plaintext attacks because security was mostly dependent on plaintext sensitivity. The concept showed real-time viability, but for wider adoption, plaintext-agnostic improvements are needed.

In a hybrid AES-RSA architecture created by Oo & Soe (2021), text data was encrypted using AES before RSA protected the AES key. Huffman compression was then used after encryption. Their Java-based solution resulted in a 30%–40% reduction in bandwidth, however the dual encryption layers caused a 25% latency. Due to the complexity of key management, hierarchical security works well for LAN data transfers but not so well for real-time IoT.

For image steganography, Pandimurugan *et al*. (2023) used RSA encryption in conjunction with Huffman-RLE-DWT compression. Their approach achieved PSNR > 42 dB in stego-images by embedding compressed ciphertext into quantized DWT coefficients. Reconstruction fidelity differed greatly throughout image textures, and the multi-stage pipeline resulted in a 50% increase in processing time. Although the method requires computational optimization for video applications, it strikes a compromise between security and perceptual quality.

Characters are converted into elliptic curve points (Xm, Ym) prior to encryption and compression in Khaing Oo & Soe's (2024) ECC-based text encryption with Huffman compression. Their approach produced 80% to 90% CR post-compression, while expanding the text size to 4x before compression. The inefficiency of point mapping for Unicode text and the lack of empirical validation outside of artificial datasets were among its drawbacks. Although it shows promise in terms of bandwidth efficiency, multilingual support and practical testing are necessary.

Latha (2023) used logistic maps for diffusion and Haar wavelets for dimensionality reduction to create HWCD, a wavelet-based image compression with chaotic encryption. SSIM > 0.92 and PSNR > 40 dB were demonstrated in the results; however, wavelet processing had a 25% higher latency than DCT-based techniques. The method works well for biological imaging, however it needs GPU acceleration to operate in real time.

By adding adder-rotation operations to Blowfish's F-function and utilizing ECC for key exchange, Gupta *et al*. (2024) optimized a Blowfish-ECC hybrid for the Internet of Things. Their method showed 10% throughput degradation at >1 MB payloads, but also lowered execution time by 15% and memory utilization by 20% when compared to solo RSA. Although the approach makes resource-constrained security effective, multimedia IoT requires hardware acceleration.

In their paper A hybrid approach to secure and compress data streams in cloud computing environment, Abdo *et al*. (2024) presented a strategy that used Huffman coding and RSA encryption to improve cloud data security and compression. Their approach, which focused on the twin goals of security and transmission overhead reduction, used RSA to encrypt critical data before compressing it using Huffman coding. Although specific numerical outcomes, such as compression ratios or throughput increases, were not given, the results demonstrated increased encryption reliability and data size reduction. The authors came to the conclusion that this two-layer approach successfully strikes a balance between the requirements for data confidentiality and communication effectiveness. Nevertheless, the study does not provide comparative analysis with other hybrid techniques or comprehensive performance measures.

An overview of corporate governance studies in India was provided by Abhilash *et al*. in 2023. A bibliometric analysis that provided valuable insights into research mapping and trend analysis pertinent to future cryptographic studies, despite not being specifically focused on encryption. By identifying methodological gaps and growth areas in governance research, their bibliometric methodology examined a large dataset of articles. These findings could potentially improve risk models and policy goals, so indirectly guiding hybrid cryptography frameworks. Specific encryption approaches were not explored, despite the fact that their findings showed an increasing focus on cybersecurity and digital transformation in organizational networks. The importance of trend analysis in creating current research agendas was emphasized in their conclusion. Its lack of technical research in data security models or cryptography is a significant drawback.

Durge and Deshmukh (2024) presented a hybrid RSA-AES encryption technique for safe data transfer in cloud contexts in Securing cloud data: A hybrid encryption strategy with RSA and AES for increased security and performance. In order to combine the benefits of both public and private key cryptosystems, the authors created a system that encrypts data after it has been encoded using both RSA and AES. Although particular statistical evaluations were not included, their model increased computing efficiency and security. They came to the conclusion that this kind of hybrid strategy enhances security against unwanted access while enhancing processing efficiency. The lack of quantifiable benchmarks for delay, encryption speed, or compression ratio was one of the study's limitations.

In their work Secure storage of data using hybrid cryptography, Elamir and Elamir (2024) suggested a method that used the RSA and AES algorithms to secure data that was saved. The approach provided a layered defense mechanism by combining symmetric and asymmetric encryption to safeguard data while it was at rest. Although there were no quantifiable performance measures offered, the results showed increased confidentiality and durability against threats. The authors came to the conclusion that dual encryption has a lot to offer in terms of storage and cloud security. However, the study's main flaw was that it lacked testing in real-time systems or comparative evaluation.

In Hybrid Cryptography Approach, Gour *et al*. (2024): To improve the security and authenticity of data during transmission, a cryptographic model utilizing AES, ECC, and DSA was designed for secure data transfer using block cipher techniques. The model supported confidentiality, integrity, and authenticity by integrating key exchange mechanisms and block cipher techniques. Although they did not reveal any performance metrics, such as encryption time or storage savings, their results demonstrated increased versatility for secure communication. The authors came to the conclusion that complex network security issues are addressed by multi-method integration. However, it was difficult to verify their assertions because there were no implementation data or benchmarks available.

In their study Weak-key distinguishers for AES, Grassi *et al*. (2019) investigated weak-key vulnerabilities in AES. Their study demonstrated that specific weak keys could lessen AES's resistance to differential and linear assaults using theoretical modeling and cryptanalysis. The results showed trends that may be used to lower the quality of encryption in certain situations. In order to preserve the resilience of encryption, their conclusion underlined the significance of safe key generation. Although the study is important for pointing out possible vulnerabilities, it is limited by the absence of workable mitigation techniques or compatibility with alternative encryption methods.

In Cloud security utilizing hybrid cryptography techniques, Kumar *et al*. (2021) introduced a multilevel cloud security paradigm that included symmetric (DES) and asymmetric (RSA) encryption. By employing dual encryption, their suggested methodology sought to protect data in cloud settings while it is in transit and at rest. Although the approach demonstrated theoretical resilience to typical assaults, empirical performance criteria including storage efficiency, computing burden, and encryption time were not specified. The authors came to the conclusion that cloud systems' security environment is improved by integrating RSA and DES. Nevertheless, the study did not examine the integration of Huffman coding or compare it with contemporary compression techniques.

Rahman and Hamada (2020) presented a hybrid lossless compression model that integrated key-based security elements into Huffman coding in their Burrows–Wheeler transform-based lossless text compression using keys and Huffman coding. They preprocessed the text using the Burrows-Wheeler transform, then enhanced data integrity with Huffman encoding and a key mechanism. Although precise percentages were not revealed, their research demonstrated superior compression ratios when compared to standard Huffman alone. The authors came at the conclusion that combining compression and transformation improves transmission and storage efficiency. The primary flaw was that there was no encryption to protect the confidentiality of the data, which made it better suited for situations involving merely compression.

Rahman and Hamada's (2021) work on Lossless text compression using GPT-2 language model and Huffman coding expanded on their previous work in which they used GPT-2 to predict and encode common patterns before implementing Huffman compression. The compression performance was better with our AI-assisted approach than with conventional statistical models. Though no specific numerical improvements were revealed, their results demonstrated higher compression efficiency, especially for structured text. The result highlighted AI's potential to improve data minimization strategies. Nevertheless, the technique ignores security, which means it is not appropriate for jobs involving the combination of encryption and compression.

Rehman *et al*. (2021) presented a Hybrid AES-ECC model for cloud data security, emphasizing the combination of AES and elliptic curve cryptography (ECC) for improved data secrecy. They used ECC for secure key exchange and AES for quick encryption in their model. Although there were no precise timing or throughput measurements, the results showed increased attack resistance and interoperability with cloud infrastructures. They came to the conclusion that the hybrid approach preserves scalability while bolstering cloud security. The lack of compression techniques was a major flaw that prevented storage efficiency from being addressed.

Yusuf and Miyim (2022) created a system that combines AES, ElGamal encryption, and Huffman coding in their work Hybrid cryptosystem and the Huffman coding for efficiency in a communication channel. Their objective was to offer encryption in addition to effective data transfer via unprotected routes. The results demonstrated improved performance in terms of communication security and transmission size reduction, albeit precise numbers were not provided. They stressed the usefulness of hybrid models in real-time systems in their conclusion. However, the study was devoid of implementation specifics and measures that might be compared to conventional methods.

In their research, "Secure file storage on cloud using hybrid cryptography," Satheesh Kumar *et al*. (2023) employed RSA and AES for encryption and steganography for concealment. Before being uploaded to AWS S3 storage, files were first encrypted and then concealed using steganographic techniques. Their strategy used multiple layers of protection to guarantee confidentiality and covert transmission. They came to the conclusion that cloud file security is enhanced by multi-method integration. One drawback, though, was the absence of measurements pertaining to compression and quantified performance indicators.

Bhawane and Ahuja (2024) created a storage encryption model that combined symmetric and asymmetric cryptographic approaches in their paper Secured data storage on the cloud using hybrid cryptography. To guarantee data integrity and secrecy in cloud services, they employed AES and RSA. The solution was created to reduce risks and stop unwanted access when using the cloud. Although the approach was well-designed, it lacked performance metrics like compression gain or processing time. This restricts how their suggested remedy may be evaluated practically.

A technique that uses Huffman encoding to compress data tokens that have already been encrypted was proposed by Hamidu *et al*. (2025) for modified RSA-AES encrypted token compression in secure banking transactions. For transmission across secure banking channels, the model uses Huffman compression after dual encryption using RSA and AES. The results showed that packet size was significantly reduced, increasing network efficiency without sacrificing data security. The authors came to the conclusion that financial systems are more protected and perform better thanks to this three-layered approach. However, the system's inability to scale to big data settings was shown to be caused by integration complexity and processing cost.

An effective secure data compression method based on chaos and adaptive Huffman coding was presented by Usama *et al*. in 2021. Their approach used adaptive Huffman coding with chaotic maps to improve compression efficiency and data security. In comparison to conventional techniques, the results demonstrated enhanced security metrics and a compression ratio of 65%. They came to the conclusion that data security and compression efficiency are successfully balanced by this hybrid strategy. The possible complexity of key management as a result of the dependence on chaotic maps is a recognized vulnerability.

Laurentinus *et al*. (2020) compared the effectiveness of AES and RSA when used in conjunction with the Huffman algorithm to compress SMS messages. Their methodology especially examined the two encryption algorithms on SMS messages when combined with Huffman compression. The findings showed that RSA+Huffman outperformed AES+Huffman in terms of compression efficiency, achieving 24.8% as opposed to 17.35%. They came to the conclusion that RSA and Huffman improve SMS compression effectiveness. The main shortcoming is that RSA has a larger computational overhead than AES, which may affect how well devices with limited resources function.

Wahab *et al*. (2021) used RSA encryption and Huffman coding to create a high capacity and safe image steganography method. Their approach used a steganography technique to conceal data in photographs by fusing Huffman coding and RSA encryption. The outcomes showed little discernible distortion and a strong embedding capacity. They came to the conclusion that the method guarantees effective and safe image data concealment. One drawback is that different image sizes and types may affect how effective the procedure is.

Khan & Tran (2024) presented a thorough approach to data compression and decompression that included DES and AES for increased security with Huffman coding, LZW compression, and run-length encoding. These many compression methods were integrated with DES and AES encryption in their methodology. Significant data size reduction was accomplished while preserving security. They came to the conclusion that compression and security are successfully balanced by this multi-layered method. One limitation mentioned was the difficulty of combining several methods, which could impact processing time.

Huffman coding was reviewed by Tian *et al*. (2020), who concentrated on making it run extremely well on contemporary GPU architectures. As part of their process, Huffman coding was specifically optimized for GPU performance. On NVIDIA GPUs, the results indicated an encoding throughput speedup of increase to 6.8x. They came to the conclusion that compression performance is much enhanced by GPU-optimized Huffman coding. The approach's hardware reliance, which restricts its application to computers lacking appropriate GPUs, is a major flaw.

HE is all you need: Compressing FHE ciphertexts using additive HE" was first presented by Mahdavi *et al*. (2023) (also known as Akhavan Mahdavi *et al*. 2023) after their publication. Their approach used additive homomorphic encryption to create a compression method for Fully Homomorphic Encryption (FHE) ciphertexts. The findings showed remarkable compression ratios (up to 90% or 99%), which considerably lower the cost of communication in networks with high latency. They came to the conclusion that the technique provides a workable way to transmit FHE ciphertexts effectively. The main drawback is that it is only applicable to FHE schemes, which restricts its generalizability to other encryption techniques.

Duda and Niemiec (2023) investigated asymmetric numeral system (ANS)-based encryption and lightweight compression. Their approach combines encryption and data compression using ANS. The results added encryption capabilities while achieving compression rates similar to arithmetic coding. They came to the conclusion that ANS offers a productive way to encrypt and compress data at the same time. One vulnerability was found to be the security level's possible incompatibility with specialized encryption methods like RSA.

AbdelWahab *et al*. (2021) developed an effective method for data hiding that combines lossy and lossless compression steganography techniques with RSA cryptography. Their approach combined lossy and lossless compression methods with RSA encryption for steganographic applications. Improved data security and concealment ability were demonstrated by the results. They came to the conclusion that data can be securely concealed using a hybrid technique. The use of lossy compression was identified as a problem since it may cause data degradation and impact the quality of hidden information.

Yin *et al*. (2021) looked into asymmetric-alphabet channels using multi-channel Huffman codes. Their approach investigated how these specific codes were designed. In some situations, multi-channel codes can perform better than single-channel codes, according to the results. They came to the conclusion that, for some communication channels, multi-channel Huffman coding may be advantageous. The intricacy of code development was identified as a drawback that would restrict practical application.

**Table 1 SUMMARY OF RELATED WORK**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| S/N | Authors | Experiments | Strengths | Weaknesses  |
| 1 | Abdulghani *et al*. (2021) | Combined LZMA compression with multi-layer encryption (incl. RSA). Tested with NIST randomness tests and measured space-saving. | Strong ciphertext security (passed all NIST tests @99% confidence), high space-saving (58.63%-81.8%). | High computational cost (22% latency increase vs. standalone RSA), limits real-time use in resource-constrained contexts. |
| 2 | Wahab *et al*. (2021) | Used adaptive Huffman coding, RSA-2048, and DWT-LSB steganography. Validated undetectability (chi-square, histogram). | 65% avg compression ratio, resisted 99% of steganalysis techniques. | 40% processing time spent on RSA key gen, observable LSB irregularities in high-res photos. |
| 3 | Ray *et al*. (2022) | Suggested cloud-IoT model: sensitive data identification, Huffman compression, RSA encryption. Measured leakage attempts & exposure time. | Eliminated 97% leakage attempts, reduced data exposure time by 50%. | Centralized key management (SPOF), heuristic classification had 12% false positives (non-critical→high-risk), increased overhead. |
| 4 | Priyadharshini & Canessane (2022) | Enhanced RSA with blockchain, logistic chaotic maps, Huffman compression. Tested encryption speed & brute-force resistance. | 22% faster encryption than regular RSA, withstood 99.8% brute-force attacks. | Blockchain latency (300 ms/tx), 15% key sync failures due to chaotic sensitivity. |
| 5 | Singh & Thakur (2021) | Document security: Huffman compression, RSA-encrypted watermarks, DCT embedding. Tested size reduction & watermark resilience. | 45%-55% size reduction, 95% watermark resilience against removal. | 35% encryption time overhead, pixelation in high-res documents post-reconstruction. |
| 6 | Suo *et al*. (2024) | Reversible IoT: CS compression-encryption, RSA-3072, DWT-SVD watermarking. Measured compression, PSNR, watermark accuracy. | 85% compression, PSNR > 48 dB reconstruction. | Watermark accuracy fell to 78% under noise; CS reconstruction failed at >15% packet loss. |
| 7 | Wiaya *et al*. (2022) | Huffman + unary codes for text compression. Tested CR & throughput on large corpora. | 80%-90% compression ratios. | 45% runtime increase for >1GB files (unary cost), lacked integrated encryption, requires separate security. |
| 8 | Zhang *et al*. (2021) | Combined CALIC (image coding) with Chen hyperchaotic encryption. Compared to JPEG-LS, measured entropy & memory. | Beat JPEG-LS compression, near-ideal entropy (7.999). | Incompatible with video, hyperchaotic systems used 50% more memory than RSA, lacks multimedia versatility. |
| 9 | Patidar *et al*. (2024) | Edge-IoT: Lossy SZ 2.1 compression + RSA-based homomorphic encryption. Tested latency, energy, PSNR, server load. | Reduced latency 28%, energy usage 30% vs. conventional pipelines. | Lossy trade-off (PSNR < 28 dB), homomorphic processes increased server load by 40%. |
| 10 | Singh & Singh (2022) | Meta-analysis of 120 RSA-Huffman image encryption studies. Assessed CR, NPCR, UACI, time complexity. | Reported median CR 50%-70%, NPCR > 99.5%. | Variable testing parameters complicate comparisons; RSA hybrids 20% slower than symmetric. |
| 11 | Bao *et al*. (2023) | Chaos-adaptive Huffman: Chaotic S-Boxes/keystream + adaptive Huffman. Tested space-saving, NIST randomness. | One-step space-saving, 99% NIST randomness confidence. | Susceptible to chosen-plaintext attacks (security dependent on plaintext sensitivity). |
| 12 | Oo & Soe (2021) | Hybrid AES-RSA (AES encrypts data, RSA protects key) + Huffman post-encryption. Java implementation, measured bandwidth/latency. | 30%-40% bandwidth reduction. | 25% latency due to dual encryption layers, complex key management; not suitable for real-time IoT. |
| 13 | Pandimurugan *et al*. (2023) | Image stego: RSA + Huffman-RLE-DWT compression, embedding in DWT coeffs. Measured PSNR, processing time. | PSNR > 42 dB in stego-images. | 50% processing time increase, reconstruction fidelity varied significantly with image texture. |
| 14 | Khaing Oo & Soe (2024) | ECC-based text enc: Char → ECC points (Xm, Ym) + Huffman compression. Tested CR. | 80%-90% CR post-compression. | Text size expanded 4x before compression; inefficient Unicode mapping; lack of empirical validation outside artificial datasets. |
| 15 | Latha (2023) | HWCD: Haar wavelets (compression) + logistic maps (encryption diffusion). Measured SSIM, PSNR, latency. | SSIM > 0.92, PSNR > 40 dB. | 25% higher latency than DCT-based methods; requires GPU for real-time. |
| 16 | Gupta *et al*. (2024) | Optimized Blowfish-ECC hybrid for IoT (adder-rotation in F-function, ECC key exchange). Benchmarked time, memory, throughput. | 15% faster execution, 20% lower memory vs. solo RSA. | 10% throughput degradation at >1 MB payloads; requires HW acceleration for multimedia IoT. |
| 17 | Abdo *et al*. (2024) | Hybrid Huffman + RSA for cloud data. Focused on security & transmission overhead reduction. | Demonstrated increased encryption reliability and data size reduction. | No specific numerical outcomes (CR, throughput), no comparative analysis with other hybrids, limited performance measures. |
| 18 | Abhilash *et al*. (2023) | Bibliometric analysis of corporate governance studies in India (indirect relevance). | Identified cybersecurity/digital transformation trends; insights for risk models/policy. | Not focused on encryption/compression; lacks technical research in data security/cryptography. |
| 19 | Durge & Deshmukh (2024) | Hybrid RSA-AES for cloud data transfer. | Claimed increased computing efficiency and security. | No quantifiable benchmarks (delay, speed, CR). |
| 20 | Elamir & Elamir (2024) | Hybrid RSA+AES for data-at-rest security. | Claimed increased confidentiality and durability. | No performance measures, no testing in real-time systems, no comparative evaluation. |
| 21 | Gour *et al*. (2024) | Cryptographic model (AES, ECC, DSA) for secure transfer. | Claimed increased versatility for secure communication (confidentiality, integrity, authenticity). | No performance metrics (time, savings), no implementation data/benchmarks. |
| 22 | Grassi *et al*. (2019) | Investigated weak-key vulnerabilities in AES (theoretical modeling/cryptanalysis). | Highlighted potential vulnerabilities impacting encryption quality. | No practical mitigation techniques; not integrated with other methods/compression. |
| 23 | Kumar *et al*. (2021) | Multilevel cloud security: DES + RSA. | Claimed theoretical resilience to common attacks. | No empirical performance criteria (burden, efficiency, time); no Huffman integration/comparison. |
| 24 | Rahman & Hamada (2020) | Lossless text compression: BWT + Key-based Huffman. | Superior compression vs. standard Huffman (no % given), improved data integrity. | No encryption for confidentiality; suited for compression-only. |
| 25 | Rahman & Hamada (2021) | Lossless text compression: GPT-2 prediction + Huffman coding. | Higher compression efficiency vs. statistical models (structured text). | Ignores security; not suitable for encryption+compression. |
| 26 | Rehman *et al*. (2021) | Hybrid AES-ECC for cloud data security (AES encryption, ECC key exchange). | Claimed increased attack resistance, cloud interoperability, scalability. | Lacked compression techniques; did not address storage efficiency. |
| 27 | Usama *et al*. (2021) | Hybrid: Chaos-based encryption + Adaptive Huffman compression. | Secure entropy, greater compression ratio (suitable for lightweight/IoT). | Complexity; challenge of precise chaos key management. |
| 28 | Yusuf & Miyim (2022) | Hybrid: AES + ElGamal + Huffman for communication. | Claimed improved security & transmission size reduction. | No specifics/numbers; lacked implementation details/comparative measures. |
| 29 | Satheesh Kumar *et al*. (2023) | RSA+AES encryption + Steganography for cloud storage (AWS S3). | Multi-layer protection (confidentiality, covert transmission). | No compression/quantified performance indicators. |
| 30 | Bhawane & Ahuja (2024) | Hybrid AES+RSA for secured cloud data storage. | Aimed to ensure integrity/secrecy, reduce risks. | No performance metrics (gain, time). |
| 31 | Hamidu *et al*. (2025) | Modified RSA-AES encrypted token compression with Huffman for banking. | Significantly reduced packet size, increased network efficiency. | Integration complexity, processing cost limits scalability to big data. |
| 32 | Laurentinus *et al*. (2020) | Compared AES+Huffman vs RSA+Huffman for SMS compression. | RSA+Huffman achieved better compression (24.8%) than AES+Huffman (17.35%). | RSA has higher computational overhead than AES (impacts resource-constrained devices). |
| 33 | Khan & Tran (2024) | Multi-compression (Huffman, LZW, RLE) + Multi-encryption (DES, AES). | Significant data size reduction, preserved security. | Complexity of combining methods impacts processing time. |
| 34 | Tian *et al*. (2020) | Optimized Huffman coding for GPUs. | Up to 6.8x encoding throughput speedup on NVIDIA GPUs. | Hardware-dependent (requires suitable GPU). |
| 35 | Mahdavi *et al*. (2023) | Compressed FHE ciphertexts using additive HE. | High compression ratios (up to 90%/99%), reduced comms cost in high-latency nets. | Only applicable to FHE schemes (not generalizable). |
| 36 | Duda & Niemiec (2023) | Asymmetric Numeral System (ANS) for encryption + lightweight compression. | Compression similar to arithmetic coding + added encryption capabilities. | Security level may not match specialized methods (e.g., RSA). |
| 37 | AbdelWahab *et al*. (2021) | Data hiding: Lossy/lossless compression stego + RSA. | Improved security and concealment ability. | Lossy compression may cause data degradation/impact hidden info quality. |
| 38 | Yin *et al*. (2021) | Multi-channel Huffman codes for asymmetric-alphabet channels. | Potential performance benefits over single-channel codes in specific situations. | Complexity of code development restricts practical application. |

**Critiques of the review of related studies**

(i).Methodological Fragmentation and Validation Gaps

A major flaw in the literature is the lack of standardized empirical validation. While research such as Singh & Singh (2022) expressly mentioned inconsistent testing parameters (e.g., varying key sizes and datasets), more than 40% of the examined works (Abdo *et al*., 2024; Elamir & Elamir, 2024; Gour *et al*., 2024) did not give measurable performance indicators. Comparative benchmarks against baselines (e.g., standalone RSA/AES or newer alternatives such as ChaCha20) failed to support claims of "enhanced security" or "improved compression". This obscures true efficacy; for example, Durge & Deshmukh's (2024) hybrid RSA-AES architecture claimed "increased computing efficiency" without latency/throughput data, While Wiaya *et al*. (2022) found compression rates of 80-90%, they ignored entropy tests, leaving security integrity unproven. Such methodological inadequacies jeopardize reproducibility and cross-study analysis, especially when niche approaches (Mahdavi *et al*.'s FHE compression; Zhang *et al*.'s hyperchaotic-CALIC) assert domain-specific superiority without interoperability testing.

 (ii) Practical Scalability and Resource Trade-offs

The evaluated works frequently undervalue real-world deployment restrictions, particularly in IoT/edge situations. Hybrid techniques based on asymmetric encryption (e.g., RSA in Laurentinus *et al*., 2020; Wahab *et al*., 2021) include prohibitive overheads—22-40% latency spikes and key-generation bottlenecks—that directly contradict their applicability for "resource-constrained" or "real-time" use cases. Patidar *et al*. (2024) demonstrated a paradox: their edge-IoT system cut energy by 30% but decreased reconstruction fidelity (PSNR <28 dB) and increased server load by 40%, making it unsuitable for medical/industrial use. Similarly, blockchain-augmented (Priyadharshini & Canessane, 2022) and multi-algorithm frameworks (Khan & Tran, 2024; Pandimurugan *et al*., 2023) added cascading inefficiencies: 50% slower processing, synchronization problems, and hardware dependencies (e.g., GPU requirements in Latha, 2023). These compromises illustrate a crucial imbalance between academic innovation and operational feasibility, in which security improvements (e.g., 99.8% brute-force resistance) are offset by unsustainable resource requirements.

(iii) Security-Compression Integration Weaknesses

Fundamental difficulties continue in the balance between security robustness and compression efficiency. Techniques that prioritized compression (Wiaya *et al*., 2022; Rahman & Hamada, 2020-2021) frequently disregarded encryption completely, leaving data vulnerable to interception—a serious weakness in cloud/IoT contexts. Security-centric models (Grassi *et al*., 2019; Kumar *et al*., 2021) neglected compression, resulting in higher storage/bandwidth costs. Even integrated techniques have flaws: chaos-based systems (Bao *et al*., 2023; Abdulghani *et al*., 2021) achieved high NIST randomness (99%) but were vulnerable to plaintext-based attacks or key-management complexity. Lossy compression hybrids (AbdelWahab *et al*., 2021; Singh & Thakur, 2021) reduced data fidelity (pixelation and 35% encryption delays), whereas hardware-accelerated approaches (Tian *et al*., 2020) compromised generality. Duda & Niemiec (2023) and Oo & Soe (2021) found that "simultaneous" compression-encryption can impair both, with ANS security behind dedicated ciphers and ECC point-mapping inflating pre-compression data by 4×. This highlights an ongoing trade-off: optimizations for one domain (e.g., Mahdavi *et al*.'s 90% FHE compression) rarely translate holistically without jeopardizing confidentiality, integrity, or adaptability.

**Conclusion**

**Current hybrid compression-encryption approaches demonstrate promising theoretical benefits for data security and efficiency but remain hampered by crippling trade-offs between robustness, performance, and practicality particularly in IoT/edge environments—due to inconsistent validation, asymmetric encryption overheads (e.g., 22–40% latency spikes), hardware dependencies, and security vulnerabilities in integrated designs.** To enable real-world adoption, future research must prioritize: **standardized cross-platform benchmarking** (NIST randomness tests, PSNR/SSIM for fidelity, resource-constrained throughput/latency) to ensure reproducible and comparable results; **lightweight, hardware-agnostic co-design** leveraging adaptive algorithms (e.g., chaos-resilient Huffman variants) and portable acceleration to eliminate deployment bottlenecks; and **proactive integration of post-quantum cryptography** (e.g., lattice-based schemes) with neural or entropy-aware compression to future-proof solutions against emerging threats. Only through rigorous, holistic frameworks that balance these dimensions can scalable secure data ecosystems be achieved.

**References**

AbdelWahab, O. F., Hussein, A. I., Hamed, H. F. A., Kelash, H. M., & Khalaf, A. A. M. (2021). Efficient combination of RSA cryptography, lossy, and lossless compression steganography techniques to hide data. Procedia Computer Science, 182, 5–12.

Abdo, A., Al-Habashna, A., & Al-Dubai, A. (2024). A hybrid approach to secure and compress data streams in cloud computing environment. Journal of King Saud University - Computer and Information Sciences, 36(3), 101999.

Abdulghani, H. A., Nijdam, N. A., Collen, A., & Konstantas, D. (2021). A hybrid approach to secure and compress data streams in cloud computing. *Journal of King Saud University - Computer and Information Sciences*, 34(10), 9012–9023.

Abhilash, A., Shenoy, S. S., & Shetty, D. K. (2023). Overview of corporate governance research in India: A bibliometric analysis. Cogent Business & Management, 10(1), 2182361.

Akhavan Mahdavi, R., Diaa, A., & Kerschbaum, F. (2023). HE is all you need: Compressing FHE ciphertexts using additive HE. arXiv preprint arXiv:2303.09043.

**Bao, W., Zhu, C., & Chen, Y. (2023).** A secure and efficient image compression-encryption scheme using chaotic S-Box and DNA operations. IEEE Access, 11, 10383-10396.

Bhardwaj, D., & Kaushik, S. (2021). A hybrid RSA and lightweight cryptography approach for secure IoT communication. *Materials Today: Proceedings*, 47, 4695–4700.

Bhawane, M., & Ahuja, S. (2024). Secured data storage on the cloud using hybrid cryptography. International Journal for Research in Applied Science and Engineering Technology, 12(4), 65674.

Duda, J., & Niemiec, M. (2023). Lightweight compression with encryption based on asymmetric numeral systems. *International Journal of Applied Mathematics and Computer Science*, *33*(1).

Durge, R. S., & Deshmukh, V. M. (2024). Securing cloud data: A hybrid encryption approach with RSA and AES for enhanced security and performance. Journal of Integrated Science and Technology, 13(3), 1060.

El Kaafarani, A. (2023). Quantum-safe cryptography: Preparing for the future. *Journal of Cybersecurity Practice and Research*, 2(1), 10–20.

Elamir, M., & Elamir, M. (2024). Secure storage of data using hybrid cryptography. GIJET, 10(2), 634.

Gokulapriya, P. L. G. B. J., & Sharmila, S. S. S. (2021). A study of cryptography encryption and compression techniques. Journal of Critical Reviews, 8(2), 1126–1131.

Gour, A., Malhi, S. S., Singh, G., & Kaur, G. (2024). Hybrid cryptographic approach: For secure data communication using block cipher techniques. E3S Web of Conferences, 556, 01048.

Grassi, L., Leander, G., Rechberger, C., Tezcan, C., & Wiemer, F. (2019). Weak-key distinguishers for AES. Cryptology ePrint Archive, Report 2019/852.

Gupta, R., & Sharma, A. (2023). Parallel Huffman coding implementation using CUDA for large-scale data compression. *International Journal of Computer Applications*, 185(32), 1–7.

Gupta, R., Silakari, S., & Khan, S. A. (2024). A hybrid encryption approach for efficient and secure data transmission in IoT devices. *Journal of Engineering and Applied Science*, 71, Article 138.

Hamidu, M., Sarjiyus, O., & Manga, I. (2025). Huffman Encoding for Modified RSA-AES Encrypted Token Compression in Secure Banking Transactions. *Journal of Science Innovation and Technology Research*. 11(3), 98–128.

Kaur, G., & Bala, A. (2021). Compression algorithms and their performance analysis: A case study of Huffman coding. *International Journal of Information Technology*, 13(3), 785–792.

Khaing Oo, K., & Soe, Y. N. (2024). Huffman compression technique in the context of ECC for enhancing security and channel bandwidth utilization. *International Journal of Trend in Scientific Research and Development*, 3(6), 834–838.

Khan, S., & Tran, T. (2024). A comprehensive methodology for data compression and decompression utilizing Huffman coding, LZW compression, and run-length encoding, integrated with data encryption standard (DES) and advanced encryption standard (AES) for enhanced security. Indian Journal of Cryptography and Network Security (IJCNS), 4(2), 7–13.

Kumar, S., Karnani, G., Gaur, M. S., & Mishra, A. (2021). Cloud security using hybrid cryptography algorithms. In 2021 2nd International Conference on Intelligent Engineering and Management (ICIEM) (pp. 597–600). IEEE.

Latha, H. R. (2023). HWCD: A hybrid approach for image compression using wavelet, encryption using confusion, and decryption using diffusion. *Journal of Intelligent Systems*, 32(1), Article 20229056.

Laurentinus, D., Suryanegara, M., & Suryanegara, M. (2020). Performance comparison of RSA and AES to SMS messages compression using Huffman algorithm. Procedia Computer Science, 179, 100–107.

Malhotra, R., & Tripathi, N. (2021). Anomaly detection in audio streams using adaptive Huffman coding. *Procedia Computer Science*, 190, 678–685.

Oo, K. K., & Soe, Y. N. (2021). Encryption data measurement and data security of hybrid AES and RSA algorithm. *International Journal of Trend in Scientific Research and Development*, 3(6), 834–838.

Pandimurugan, V., Sathish Kumar, L., Amudhavel, J., & Sambath, M. (2023). Hybrid compression technique for image hiding using Huffman, RLE and DWT. *Materials Today: Proceedings*, 47, 1128–1133.

Patidar, S., Jindal, R., & Kumar, N. (2024). A secure and energy-efficient edge computing improved SZ 2.1 hybrid algorithm for handling IoT data stream. *Multimedia Tools and Applications*, 83(25), 83629–83660.

Priyadharshini, K., & Canessane, A. (2022). Security in data sharing for blockchain-intersected IoT using novel chaotic-RSA encryption. *International Journal of Information Security and Privacy*, 16(2), 1–15.

Rahman, M. A., & Hamada, M. (2020). Burrows–Wheeler transform based lossless text compression using keys and Huffman coding. Symmetry, 12(10), 1654.

 Rahman, M. A., & Hamada, M. (2021). Lossless text compression using GPT-2 language model and Huffman coding. SHS Web of Conferences, 102, 04013.

Ray, S., Mishra, K. N., & Dutta, S. (2022). Sensitive data identification and security assurance in cloud and IoT-based networks. *International Journal of Computer Network and Information Security*, 14(5), 11–27.

Satheesh Kumar, A., Anfah, K., Hariharan, T., Parveen, R., Mahmud, S., & Sharma, S. (2023). Secure file storage on cloud using hybrid cryptography. International Journal of Advanced Research, 11(April), 01–05.

Singh, A. K., Thakur, S., Jolfaei, A., Srivastava, G., Elhoseny, M. D., & Mohan, A. (2021). Joint encryption and compression-based watermarking technique for security of digital documents. *ACM Transactions on Internet Technology*, 21(1), 1–20.

Singh, K. N., & Singh, A. K. (2022). Towards integrating image encryption with compression: A survey. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 18(3), 1–21.

Singh, M., & Bansal, H. (2022). Enhancing real-time data compression using adaptive Huffman coding in IoT networks. *Wireless Personal Communications*, 127(3), 2233–2247.

Stallings, W. (2023). *Cryptography and network security: Principles and practice* (8th ed.). Pearson.

Suo, Z., Xia, C., Jiang, D., Peng, H., Tong, F., & Chen, X. (2024). Multitiered reversible data privacy protection scheme for IoT based on compressive sensing and digital watermarking. *IEEE Internet of Things Journal*, 11(7), 11809–11823.

Tian, J., Di, S., Yu, X., Rivera, C., Zhao, K., Jin, S., Feng, Y., Liang, X., Tao, D., & Cappello, F. (2021). Optimizing error-bounded lossy compression for scientific data on GPUs. In 2021 IEEE International Conference on Cluster Computing (CLUSTER) (pp. 283–293). IEEE.

Usama, M., Malluhi, Q. M., Zakaria, N., Razzak, I., & Iqbal, W. (2021). An efficient secure data compression technique based on chaos and adaptive Huffman coding. Peer-to-Peer Networking and Applications, 14, 2651–2664.

Wahab, A. A., *et al*. (2021). Implementation of Huffman encoding for modified RSA-AES encrypted token compression in secure banking transactions. *International Journal of Computer Science and Engineering*, 9(3), 45–52.

Wahab, O. F. A., Khalaf, A. A. M., Hussein, A. I., & Hamed, H. F. A. (2021). Hiding data using efficient combination of RSA cryptography and compression steganography techniques. *IEEE Access*, 9, 31805–31815.

Wiaya, B. A., Siboro, S., Brutu, M., & Lase, Y. K. (2022). Application of Huffman algorithm and unary codes for text file compression. *SinkrOn*, 7(3), 1000–1007.

Yin, H. H. F., Wang, X., Ng, K. H., Lai, R. W. F., Ng, L. K. L., & Ma, J. P. K. (2021). On multi-channel Huffman codes for asymmetric-alphabet channels. arXiv preprint arXiv:2105.03606.

 Yusuf, A. Y., & Miyim, A. M. (2022). Hybrid cryptosystem and the Huffman coding for efficiency in a communication channel. Nigerian Journal of Computing, Engineering and Technology, 1(1), 44–56.

Zhang, M., Tong, X., Wang, Z., & Chen, P. (2021). Joint lossless image compression and encryption scheme based on CALIC and hyperchaotic system. *Entropy*, 23(8), 1096.