Informatics Bulletin, Faculty of Computers and Artificial Intelligence, Helwan University
Published Online ***Vol 6 Issue 2, July 2024***
(https://fcihib.journals.ekb.eg)

# Enhanced Data Security and Storage Efficiency in Cloud Computing: A Survey of Data Compression and Encryption Techniques

A.Abdo [1], Taghreed S. Karamany [1,2,*], Ahmed Yakoub [1]

1 Department of Information Systems, Faculty of Computers and Artificial Intelligence, Helwan University, Cairo 11795, Egypt

2 Future Academy - Higher Institute for Specialized Technological Studies, 29 Ismailia Desert Rd, El Shorouk - Cairo 6363040, Egypt

***Abstract*— Cloud computing is the most recent technology to emerge in the last several decades. Which has gained significant popularity across various organizations because of the benefits it offers, including cost reduction, resource pooling, broad network access, and ease of administration. It is a great platform for users to share data or applications on remote servers that can be processed and accessed through the Internet. However, as the demand for cloud computing grows along with the amount of data being transferred and stored in the cloud environment, ensuring data security has become a major concern. Securing data at rest and in transit is one of the most pressing issues faced by cloud computing. Therefore, many researchers have contributed to address the problem of data security in cloud computing while enabling effective data transmission by developing a variety of technologies to secure and compress cloud data, including data encryption and compression. In terms of achieving a high level of security and a high compression ratio, this survey is proposed to provide an overview of data compression and encryption techniques used to enhance data transmission and prevent unauthorized access to sensitive data. Also, we represent a comparative study between data encryption and compression algorithms, especially applied to textual data in the cloud. The comparative study encompasses the methodology, results, and limitations encountered when combining techniques in both domains. Moreover, we propose an approach to enhance security, address previous issues, and achieve higher levels of security and storage efficiency in cloud data transmission.**

*Keywords* — Information security, Encryption techniques, Data Compression, Cloud Computing.

## I. INTRODUCTION

Cloud computing is an advanced service that provides an efficient computing platform via sharing and virtualization. It improves availability, scalability, collaboration, and agility for both users and businesses [1]. Cloud computing's rapid growth in the IT industry is fuelled by significant advantages in resource storage, unlimited storage, and cost-effective methods for business continuity and scalability [2]. However, the main barrier to widespread adoption is a lack of security, specifically in data protection, authentication, and transmission [3]. Security issues in cloud computing are a constant source of concern for users due to the cloud's shared nature, which includes all layers of cloud computing.

Cloud computing encompasses various forms of services and models, which form the fundamentals of the cloud. These include Saas,Paas,Iaas [4]. SaaS acts as an end-user interface, connecting to the user and containing software packages, system software, and application-specific server storage networks. PaaS is primarily used by application developers to create cloud-based applications, whereas IaaS refers to hardware services, virtual machines, and network infrastructure. Cloud environments have several deployment models, including public, private, community, and hybrid [1, 2, 3, 4]. The public cloud is open to everyone, whereas the private cloud is only used by businesses.

### *A. Cryptography*

Cryptography refers to the process of transforming easily readable data into encrypted data that is inaccessible to unauthorized individuals. When sensitive information such as passwords, credit card numbers, and bank details is shared via the cloud, it is kept secure through encryption. Encryption involves the process of protecting information or data by using mathematical models to scramble it in such a way that only the parties who have the key to unscramble it can access it, and the original data can be retrieved by decrypting the encrypted data to make it readable (Fig. 1). Cryptography algorithms are classified into symmetric (private key) and asymmetric (public key) methods (Fig. 2, Fig. 3) [5, 6].

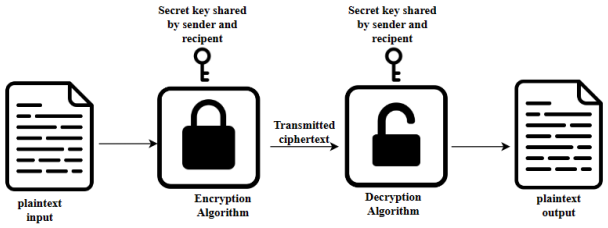



Fig. 1 Conventional encryption process

- ***Symmetric or (Private Key) Cryptography:*** is a cryptographic method that encrypts and decrypts data with the same secret key shared by both the sender and the recipient [7], as illustrated in Figure 2.

**Fig.2.**Symmetic encryption

- ***Asymmetric or (public Key) Cryptography***: is a cryptographic method that uses a public key and a private key to encrypt and decrypt data. As shown in Figure 3. Only the authorized receiver can decode the message using the private key [7].

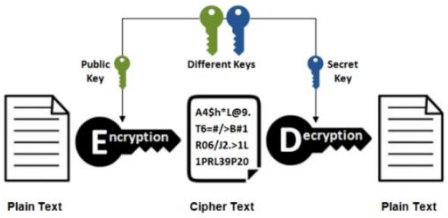



**Fig.3.** Asymmetric encryption

### *B. Data Compression*

There are two categories of compression methods: lossy compression methods and lossless compression methods. The goal of compression is to minimize the amount of original data while maximizing file transfer speed, saving storage capacity, and lowering the cost of both storage hardware and network bandwidth [6].

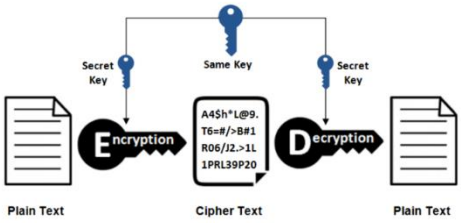
- **The lossy compression method*:*** Compression discards some of the original data. Although this reduces file size, the decompressed data does not exactly match the original data. Lossy compression is commonly used for image and audio files such as JPEG, MP3, and MPEG. [7,8,9].

- **Lossless Compression method**: implies that no data is lost during the compression process because it enables the original data to be fully restored after decompression. Among the lossless compression methods are arithmetic coding, run length encoding, Huffman encoding, Lempel-Ziv-Welch (LZW), and LZMA [7, 8, 9].

For achieving maximum protection and powerful security with high capacity, many researchers have proposed many encryption methods that focus on encrypting blocks or each byte of data with specific methods, but these methods can be vulnerable to attacks by cryptanalysts or intruders. Therefore, adding data compression not only enhances performance but also provides an additional layer of security, as the compressed data becomes unreadable in its compressed format.

## II. ENCRYPTION TECHNIQUES

This section presents some of the encryption techniques used, mainly on text data. The strength of encryption techniques lies in their ability to withstand attacks, such as brute force and




cryptanalysis, ensuring the security and privacy of sensitive information during transmission and storage. Numerous studies and researches have been conducted to ensure the security of data.

Sajjan et al. [15] conducted a survey to examine multilevel encryption used in cloud data security. Following the investigation of various ciphers, they implemented a two-layer encryption algorithm comprised of the DES and RSA ciphers. Their research concluded that multilayer encryption is more secure than single-level models.

K. R. Sajay, et al., [19] discussed cloud computing and the need for security measures within the cloud. Once a firm moves to the cloud, it may lose control over its information, and cloud security largely depends on trusted cryptography and computing. This study presented a hybrid algorithm that offers better storage and security techniques using encryption algorithms within the cloud architecture. The homographic encryption method, which facilitates secure computations, and the blowfish algorithm, which generates symmetric keys for encryption and decryption, are discussed. The study concluded that the use of encryption techniques can increase the security of information and prevent unauthorized access. However, additional measures are still necessary to fully resolve security issues within cloud storage for both small and large firms.

In [21], an improved cryptography solution was developed to enhance secure cloud computing with efficient and high-performance capabilities. The solution leveraged an improved version of the Blowfish algorithm, a symmetric encryption technique, resulting in reduced encryption time, improved execution time, and increased throughput. To address security challenges associated with symmetric key exchange algorithms, an EC asymmetric cryptography algorithm was employed to encrypt the key, mitigating risks such as key theft during transit. The proposed solution also incorporated MD5-based digital signatures to ensure data integrity. Evaluation of the solution against AES, DES, 3DES, and RSA demonstrated overall improvements, with the proposed solution exhibiting better throughput, memory usage, and execution time on average. However, it was observed that AES performed slightly faster and had higher throughput when handling larger data sizes.

Another study [22] introduced an encryption algorithm designed specifically for cloud computing systems. This algorithm was compared with existing algorithms such as DES, AES, and Blowfish. Performance evaluation was

conducted based on various parameters, including encryption time for different block sizes, the avalanche effect on plain text (64% improvement), and key strength measured against brute force cryptanalysis attacks. The results demonstrated that the algorithm outperformed existing algorithms across all evaluated parameters. Notably, the algorithm significantly reduced the time required for searching operations on encrypted data compared to the other considered algorithms.

Mohammed N. Alenezi et al. [25] analysed several encryption algorithms, including AES, Blowfish, DES, DESede, SEED, IDEA, RC2, RC4, RC6, SEED, and XTEA, which were compared and analysed based on encryption time, throughput, and CPU utilization. The results indicated that RC4, RC6, and AES performed exceptionally well in terms of encryption time and throughput. Among them, AES emerged as the preferred choice due to its superior performance and level of security. Compression Techniques

This section provides an exploration of various compression techniques, with a particular focus on text encryption. To guarantee the effectiveness of data transmission, several researches and studies have been carried out.

Kumar et al. [11] introduced an approach that included a two-factor data encryption protection mechanism based on IBE and unique customer users, as well as password-protected files stored on the cloud and issued keys, in another study. This approach improves security while requiring less storage space by incorporating LZ4 algorithms for compression and encryption. This method, in general, provides a comprehensive and effective solution for secure and efficient big data processing in cloud-based environments.

Muhammad Alif et al. [20] proposed a study that compared two lossless compression algorithms, Huffman and Lempel Ziv Welch (LZW), using various test files of different types. Evaluating space-saving and compression time, the results demonstrated LZW's superiority over Huffman in compressing.txt and.csv files, achieving average space savings of 63.85% and 77.56%, respectively. Compression speed was found to be directly proportional to the file size.

As storage space becomes limited, compression techniques have emerged to maximize storage efficiency. The study [23] focused on comparing two lossless compression algorithms, Huffman and Lempel Ziv Welch (LZW). Various test files of different file types were used to evaluate the performance of both algorithms in terms of space-saving and compression time. The results indicated that the LZW algorithm outperformed Huffman in compressing.txt and.csv files, achieving average space savings of 63.85% and 77.56%, respectively. The compression speed of each algorithm was found to be directly proportional to the file size. This comparative analysis provided valuable insights into the effectiveness of these compression techniques in maximising storage space.

Senthil and Robert [26] presented a comprehensive survey of various fundamental lossless data compression algorithms. The study included experimental results and comparisons of these algorithms on text data using statistical and dictionary-based compression techniques. Among the statistical coding techniques, this paper explored algorithms such as Shannon-Fano Coding, Huffman coding, Adaptive Huffman coding, Run Length Encoding, and Arithmetic coding. In addition, the paper examined the Lempel Ziv scheme, which is a dictionary-based technique. The Lempel Ziv scheme is divided into two families: those derived from LZ77 (LZ77, LZSS, LZH, and LZB) and those derived from LZ78 (LZ78, LZW, and LZFG). The study found that in terms of statistical compression techniques, the Arithmetic coding technique outperformed the others.

Shoukat et al. [10] proposed using the Relative Frequency of Alphabetic Letters to improve the Vigenère cipher's efficiency and security. This method introduces multiple layers of security processes to improve the confidentiality of a text message by rearranging letters according to relative frequency and performing modifications in the Vigenère cipher, followed by data compression using the lossless Huffman technique. The proposed technique generates shorter, more secure data codes than the current process, effectively improving data security in commercial, organisational, or industrial applications.

M. K. Padmapriya et al. [12] presented an experimental study that investigated the effect of data compression on amino acid cipher text using dictionary-based and entropy coding methods, achieving storage savings of 47% and 60%, respectively, without risking security. As a result, storage efficiency is doubled, making it a convincing and advantageous method for improved data storage.

Robbi, et al., [17] employed the Blowfish algorithm and the Lempel-Ziv-Welch (LZW) algorithm, respectively. Studies showed that the combination of these two algorithms produced smaller and safer ciphertext results due to compression which changes the indirect ciphertext with different patterns. The combination of Blowfish and LZW algorithms can lead to better compression security as the LZW algorithm generates a new form of strings with different patterns from the initial ciphertext, making it difficult for cryptanalysts to decrypt quickly. Additionally, the smaller size of the compressed data facilitates faster transmission without requiring significant resources. Overall, the study demonstrated positive outcomes from the combination of Blowfish and LZW algorithms for data security and compression.

M. R. Ashila et al, [18] proposed a combination of AES - Huffman code method to produce secure file encryption and minimize file size to reduce storage space and expedite the transmission of file transfers. Whereas encryption is performed first, followed by compression. To measure the level of file security, the avalanche effect (AE) and entropy after encryption and compression were measured. According to the test results, AES encryption increases file size by about 25% over the original, but when Huffman compression is used, the encrypted file size decreases by about 30%, and the compressed file size is less than the original file size. Huffman compression has been shown to improve AES encryption, but it can also improve file security, depending on AE and entropy levels.

B. Carpentieri [14] carried out research in which compression and encryption techniques were combined and applied to various digital data sets using the Calgary corpus as input. The study used four standard compression algorithms

(Huffman coding, Arithmetic coding, Lempel-Ziv-welch coding, and run length encoding) and four encryption algorithms (DES, 3DES, AES, RC4), which were tested twice. The first round started with compression and ended with encryption, while the second round started with encryption first. The findings show that the cost of encryption after compression for text data is negligible; however, compressing after encryption provides no benefit and, in fact, increases both the file and encrypted sizes. Furthermore, the study discovered that arithmetic coding almost doubled the original file size due to the randomness introduced by the encryption algorithms.

## III. Compression followed by Encryption

In this section, we provide a comprehensive overview that has concentrated on the methodology of compression followed by encryption to identify the top-performing algorithms and present the most notable outcomes achieved in this field.

Usama et al. [13] presented a method for combining data compression and encryption techniques that is both effective and secure while ensuring that neither process is compromised. To integrate key control and enable secure compression and decompression, the method employs adaptive Huffman coding, a pseudorandom keystream generator, S-Box, and a chaotic logistic map. The results show that when compared to running encryption and compression algorithms separately, the method achieves faster processing times while successfully compressing secure data. The technique's sensitivity to key and plaintext was demonstrated by security analysis, The generated ciphertexts passed all NIST randomness tests with 99% of confidence level. In terms of space savings, these techniques are comparable to standard ones.

Another study [16] by Ruchita et al. shows that compression and encryption techniques can be used to effectively secure data. Huffman with RC4 and DES proved to have the best compression ratio for text files of various sizes, reducing the file size by about half, while LZW and Arithmetic techniques also produced good results. Run-length encoding with RC4 and Caesar Cipher, on the other hand, was found to be less effective when there are fewer consecutive characters present. The authors of the study also discovered that Huffman had a higher Compression Ratio than all other approaches by nearly 50-80%, indicating that Huffman is the best compression algorithm for text compression, followed by LZW, Arithmetic, and run length. These findings have important implications for data security and provide valuable insights for researchers and practitioners working in this field.

The authors of [22] discussed the importance of security for large-sized data and proposed a method to make data or messages more secure and smaller in size. This approach used the Huffman compression technique to reduce data size and a newly developed block-type symmetric key algorithm to ensure security. The system used two private keys that were known to both the sender and the receiver but kept hidden from the outside world, resulting in a form of secret key cryptography. The system addressed security concerns and was not vulnerable to brute-force attacks due to its large key domain. However, it is currently limited to text data. While Shukur [23] clearly demonstrated through experimental results that for both AES and DES algorithms, the encryption and decryption time is greatly reduced if files are compressed and then encryption techniques are applied to them.

Ahmad Al-Smadi et al. [24] presented a practical methodology for file cryptography using the one-time pad algorithm. By controlling the data type in the encryption key, this methodology effectively addressed the encryption key management challenges associated with the Vernam algorithm. Additionally, the use of the Huffman algorithm significantly reduced the size of the output file. To enhance security, the output file is further protected with a password encrypted by the AES algorithm, making it more difficult to decrypt. Successful experiments were conducted on various file types, including txt, pdf, doc, bmp, mp4, and exe, without any loss of information. The experimental results also indicated that the time required for encryption and decryption is reduced when using a cryptographic key generated from integers, compared to a key generated from the ASCII table. The size of the file has a minimal impact on data encryption time when compression is not applied.

According to the previous sections, the most effective approach for cloud computing is to combine cryptography and compression algorithms. The following section demonstrates the major features and limitations of each previous technique.

## IV. Comparative Study

Literature survey discussed some of the common encryption algorithms which are combination of conventional (DES, 3DES, AES, Blowfish, RSA, ECC etc.) algorithms. Compression algorithms which are combination of (Huffman technique, lz4, Adaptive Huffman coding, LZMA, Arithmetic coding, Run-length encoding), Encryption followed by compression, and compression followed by encryption. This section presents some comparison between encryption and compression algorithms. Table 1 mentions comparison between encryption algorithms. According to Table 1, Blowfish, RC4, RC6, and AES give better performance rather than other algorithms. Table 2 compares performance of some compression algorithms. From Table 2, it is concluded that LZMA, arithmetic coding, LZW give better result in comparison to techniques mentioned.

Table 1 shows comparative study for all the mentioned studies in Encryption Techniques subsection.

| Reference | Data used | Algorithms | Results | Limitations |
|---|---|---|---|---|
| [15] | Text | DES and RSA | The study concluded that multilayer encryption is more secure than single-level models | ------ |
| [19] | Text | Homographic encryption method and Blowfish | Enhanced storage and security techniques in cloud architecture with a hybrid algorithm. It explored the use of homomorphic encryption for secure computations and the blowfish algorithm for generating symmetric keys for encryption and decryption. | ---------- |
| [21] | Text, image | EC, MD5 | exhibiting better throughput, memory usage, and execution time on average | It was observed that AES performed slightly faster and had higher throughput when handling larger data sizes |
| [25] | Text, image | AES, BlowFish, RC2, RC4, RC6, DES, DESede, SEED, XTEA, and IDEA. | RC4, RC6, and AES performed exceptionally well in terms of encryption time and throughput. Among these algorithms, AES emerged as the preferred choice due to its superior performance and level of security. | ------- |

Table 2 shows comparative study for all the mentioned studies in Compression Techniques subsection.

| Reference | Data used | Algorithms | Results | Limitations |
|---|---|---|---|---|
| [16] | Text | LZMA, Huffman coding | The entropy coding method saves 47% of storage space, whereas the dictionary-based coding method saves 60%. Storage efficiency has also doubled. | The level of safety is not clear. |
| [20] | various test files of different types. | Huffman and LZW compression algorithms. | The results of the study demonstrated that LZW outperformed Huffman in compressing .txt and .csv files, achieving average space savings of 63.85% and 77.56%, respectively. The study also found that compression speed was directly proportional to the file size. | ------ |
| [11] | Text | Lz4 | This approach improves security while requiring less storage space by incorporating LZ4 algorithms for compression and encryption. This method, in general, provides a comprehensive and effective solution for secure and efficient big data processing in cloud-based environments. | ------- |
| [16] | Text | LZMA, Huffman coding | The entropy coding method saves 47% of storage space, whereas the dictionary-based coding method saves 60%. Storage efficiency has also doubled. | The level of safety is not clear. |
| [20] | various test files of different types. | Huffman and LZW compression algorithms. | The results of the study demonstrated that LZW outperformed Huffman in compressing .txt and .csv files, achieving average space savings of 63.85% and 77.56%, respectively. The study also found that compression speed was directly proportional to the file size. | ---- |
| [11] | Text | Lz4 | This approach improves security while requiring less storage space by incorporating LZ4 algorithms for compression and encryption. This method, in general, provides a comprehensive and effective solution for secure and efficient big data processing in cloud-based environments. | ------- |
| [26] | Text | Shannon-Fano Coding, Huffman coding, Adaptive Huffman coding, Run Length Encoding and Arithmetic coding, LZB, LZ77, LZSS, and LZH, LZFG, LZ78 and LZW. | From the results, it was found that the arithmetic coding technique performed the best among the statistical compression techniques, showing improvements over adaptive Huffman coding, Huffman coding, Shannon-Fano coding, and run length encoding. In the dictionary-based techniques, LZB outperformed LZ77, LZSS, and LZH. Among the LZ78 family, LZFG showed significant improvements in average bits per character compared to LZ78 and LZW. | ----- |

Table 3 shows comparative study for all the mentioned studies in Encryption followed by Compression Techniques subsection.

| Reference | Data used | Algorithms | Results | Limitations |
|---|---|---|---|---|
| [10] | Text | Vigenère cipher, Huffman technique | improves the efficiency and security of the Vigenère cipher by introducing multiple layers of security processes. | the complexity of the algorithm and the additional processing time required |
| [11] | Text | IBE, lz4 | improves security while requiring less storage space. | -------- |
| [13] | Text, image | Adaptive Huffman coding, a pseudorandom keystream generator, S-Box, and a chaotic logistic map | The method outperforms running encryption and compression algorithms separately in terms of processing time. | its adaptability to different applications may also be limited and worth investigating further. |
| [18] | Text | AES then Huffman code | AES encryption increases the file size by approximately 25%, but after applying Huffman compression, the encrypted file size decreases by around 30% compared to the original file size. | AES encryption increases file size by approximately 25% over the original. |
| [17] | Text | Blowfish and LZW | The study showed that combining these two algorithms resulted in smaller and safer ciphertext due to compression, which alters the indirect ciphertext with different patterns | ---------- |
| [14] | Text | Huffman coding, Arithmetic coding, Lempel-Ziv-welch coding, and run length encoding. DES, 3DES, AES, and RC4 | the files were compressed after encryption, did not help. | compressing data after encryption did not provide any benefit |

Table 4 shows comparative study for all the mentioned studies in Compression followed by Encryption Techniques subsection.

| Reference | Data used | Algorithms | Results | Limitations |
|---|---|---|---|---|
| [16] | Text | Huffman, RC4 DES, LZW, Arithmetic techniques, Run-length encoding, Caesar Cipher | The results showed that Huffman had a nearly 50-80% higher compression ratio than all other approaches, indicating that Huffman is the best compression algorithm for text compression, followed by LZW, Arithmetic, and run length. | ------- |
| [14] | Text | Huffman coding, Arithmetic coding, Lempel-Ziv-welch coding, and run length encoding. DES, 3DES, AES, and RC4 | Text data had a negligible cost of encryption after compression. | Encrypting data after compression had minimal cost. |
| [13] | Text, image | adaptive Huffman coding, a pseudorandom keystream generator, S-Box, and a chaotic logistic map. | The results show that when compared to running encryption and compression algorithms separately, the method achieves faster processing times while successfully compressing secure data. The technique's sensitivity to key and plaintext was demonstrated by security analysis. In addition, the generated ciphertexts passed all NIST randomness tests with 99% confidence. | --------- |
| [22] | Text data | Huffman and a newly developed block-type symmetric key algorithm | The system addressed security concerns and was not vulnerable to brute-force attacks due to its large key domain | it is limited to text data only |
| [24] | txt, pdf, doc, bmp, mp4, and exe | Huffman and Vernam | The experimental results also indicated that the time required for encryption and decryption is reduced when using a cryptographic key generated from integers, compared to a key generated from the ASCII table. The size of the file has a minimal impact on data encryption time when compression is not applied. | ------ |

## V. DISCUSSION

A lot of studies and methods have been proposed by researchers to achieve the highest protection and powerful security with high capacity. However, all of these studies concentrated on developing specific techniques for encrypting data, which cryptanalysts might exploit to their advantage, Since the compressed data cannot be read in its original format, data compression will attain an additional level of protection. Applying a high level of security to data may have a detrimental impact on run time, thus it is preferable to use a hybrid strategy of improved algorithms to prevent attacks and ensure that performance is unaffected by complexity. The researches of [10, 11, 12, 13, 14, 15,16,17,18,19,27] were applied to the text data. In [14], the most provided advantage is that it provides valuable insights into the effectiveness of different compression and encryption algorithms, as well as their combinations when applied to various types of digital data. Findings suggest that compressing text data before encryption does not significantly increase the cost of encryption, whereas compressing after encryption is not beneficial and may even lead to an increase in file and encrypted sizes. Additionally, the study highlights the impact of various compression algorithms on different file sizes after encryption. While [15] concluded that multilayer encryption is more secure than single-level models. While Kumar et al. [11] enhanced the secret level of data since they provided an approach that included a two-factor data encryption protection mechanism based on IBE and unique customer users, as well as password-protected files stored on the cloud and issued keys that improves security while requiring less storage space by incorporating LZ4 algorithms for compression and encryption. In [13], the proposed technique is robust enough since they applied the method that employs adaptive Huffman coding, a pseudorandom keystream generator, S-Box, and a chaotic logistic map. The results show that when compared to running encryption and compression algorithms separately, the method achieves faster processing times while successfully compressing secure data. The technique's sensitivity to key and plaintext was demonstrated by security analysis, and the generated ciphertexts passed all NIST tests with 99% confidence in randomness. In [10], although the Vigenère cipher algorithm is not secure enough but Shoukat et al. proposed using the Relative Frequency of Alphabetic Letters to improve the Vigenère cipher's efficiency and security introducing multiple layers of security processes to improve the confidentiality of a text message by rearranging letters according to relative frequency and performing modifications in the Vigenère cipher, followed by data compression using the lossless Huffman technique. In [12] M. K. Padmapriya et al. proved by an experimental study the effect of data compression on amino acid cipher text using dictionary-based and entropy coding methods, achieving storage savings of 47% and 60%, respectively, without risking security. While Ruchita et al. [16] adopts a compression-then-encryption approach, utilizing techniques such as Huffman, RC4, DES, LZW, Arithmetic, Run-length encoding, and Caesar Cipher and indicate that Huffman provides the highest compression ratio, followed by LZW, Arithmetic, and run length. Finally, in [10] proposes a hybrid algorithm that combines homographic encryption and the Blowfish algorithm. This approach aims to enhance storage and security techniques in cloud architecture, utilizing homomorphic encryption for secure computations and the Blowfish algorithm for generating symmetric keys.

Overall, these references highlight various data, techniques, algorithms, and results related to encryption and compression. However, each approach also has its limitations, such as algorithm complexity, processing time, adaptability, and level of safety. These limitations warrant further investigation and consideration in future research. Also, this survey proposes a superior solution to address the limitations of current technologies.

## VI. CONCLUSIONS AND FUTURE WORK

This survey has highlighted the importance of considering the trade-off between data security and data compression, as it has also provided a comprehensive overview of data compression and encryption techniques and their importance in the context of cloud computing. By exploring various methods and approaches, it is evident that these techniques play a crucial role in enhancing data security and storage efficiency, as shown in the previous research. Throughout the survey, we have observed that data compression techniques can significantly reduce the size of data, resulting in reduced storage space requirements, improved transmission speeds, and accordingly, improved transmission performance. Additionally, encryption techniques provide a robust layer of security, ensuring that data remains confidential and protected from unauthorized access. Previous research has also proven that multilayer encryption is more secure than single-level models. Furthermore, this survey represented a comparative study conducted on data encryption and compression algorithms, specifically focusing on textual data in the cloud computing environment. The study encompassed the utilization of various methodologies, the analysis of results obtained, and the identification of limitations encountered when integrating techniques from both domains. In our future work, we plan to provide a framework that can enhance most of the previous security issues and achieve a higher level of security and storage efficiency for data transmission in the cloud which involves six successive stages: data encryption 1, decomposition, data encryption 2, data compression, data encryption 3, and uploading the encrypted and compressed data into the cloud.

## REFERENCES


[1] R. Mary Sheeba and R. Parameswari. (2022). Hybrid Security for Data in Cloud Computing: A Review. In Proceedings of IEMIS 2022, Volume 1, pp. 441-449.

[2] F.F. Moghaddam, O. Karimi, and M.T. Alrashdan. (2013, November). A comparative study of applying real-time encryption in cloud computing environments. In 2013 IEEE 2nd International Conference on Cloud Networking (CloudNet) (pp. 185-189).

[3] S. Gnatyuk, M. Iavich, V. Kinzeryavyy, T. Okhrimenko, Y. Burmak, and I. Goncharenko. (2020). Improved Secure Stream Cipher for Cloud Computing. In ICTERI Workshops (pp. 183-197).

[4] Y. Alemami, A.M. Al-Ghonmein, K.G. Al-Moghrabi, and M.A. Mohamed. (2023). Cloud data security and various cryptographic algorithms. *International Journal of Electrical and Computer Engineering*, 13(2), p. 1867.

[5] D.K. Sharma, N.C. Singh, D.A. Noola, A.N. Doss, and J. Sivakumar. (2022). A review on various cryptographic techniques & algorithms. Materials Today: Proceedings, 51, pp. 104-109.

[6] H. Abroshan. (2021). A hybrid encryption solution to improve cloud computing security using symmetric and asymmetric cryptography algorithms. *International Journal of Advanced Computer Science and Applications*, 12(6), pp. 31-37.

[7] M. Mua'ad and Z.A. Alqadi. (2020). Using Highly Secure Data Encryption Method for Text File Cryptography. *IJCSNS*, 20(11), p. 53.

[8] P. Kavitha. (2016). A survey on lossless and lossy data compression methods. *International Journal of Computer Science & Engineering Technology*, 7(03), pp. 110-114.

[9] M.N. Fauzan, M. Alif, and C. Prianto. (2023). Comparison of Huffman Algorithm and Lempel Ziv Welch Algorithm in Text File Compression. *IT Journal Research and Development*, 7(2), pp. 184-197.

[10] N. Shoukat, M. Azam, and I. Khan. (Oct .2022). An Improved Method of Vigenère Cipher to Securely Compress the Text by using Relative Frequenc. *International Journal of Innovative Science and Research Technology*, Vol. 7, no. 10.

[11] S. Kumar, P. Sundaresan, R. Logith, and N. Mathivanan. (2023, April). A Data Security-based Efficient Compression and Encryption for Cloud Computing. In *2023 7th International Conference on Trends in Electronics and Informatics (ICOEI)* (pp. 647-653). IEEE.

[12] M.K. Padmapriya and P.V. Eric. (2022). Effect of Data Compression on Cipher Text Aiming Secure and Improved Data Storage. In *Information and Communication Technology for Competitive Strategies (ICTCS 2021) ICT: Applications and Social Interfaces* (pp. 195-201). Singapore: Springer Nature Singapore.

[13] M. Usama, Q.M. Malluhi, N. Zakaria, I. Razzak, and W. Iqbal. (Oct. 2021). An efficient secure data compression technique based on chaos and adaptive Huffman coding. *Peer-to-Peer Networking and Applications*, vol. 14, no. 5, pp. 2651-2664, doi: 10.1007/s12083-020-00981-8.

[14] B. Carpentieri. (Mar .2018). Efficient compression and encryption for digital data transmission. *Security and Communication Networks*, vol. 2018, article 9591768, doi: 10.1155/2018/9591768.

[15] S. Bhattacharjee and S. Bansal. (2023). Simultaneous encryption and compression for securing large data transmission over a heterogeneous network. *Applied Intelligence in Human-Computer Interaction*, pp. 129-142.

[16] R. Sharma and S. Bollavarapu. (May 2015). Data Security using Compression and Cryptography Techniques. *International Journal of Computer Applications*, vol. 117, pp. 22-25, doi: 10.1007/978-981-19-0095-2_20.

[17] R. Rahim, M. Dahria, M. Syahril, and B. Anwar. (2017). Combination of the Blowfish and Lempel-Ziv-Welch algorithms for text compression. *World Trans. Eng. Technol. Educ.*, vol. 15, no. 3, pp. 292-297.

[18] M. R. Ashila, N. Atikah, D. R. I. Moses Setiadi, E. H. Rachmawanto, and C. A. Sari. (2019). Hybrid AES-Huffman Coding for Secure Lossless Transmission. In *2019 Fourth International Conference on Informatics and Computing (ICIC)*, Semarang, Indonesia, pp. 1-5, Oct. 2019, doi: 10.1109/ICIC47613.2019.8985899.

[19] S. K. R. Sajay, S. S. Babu, and Y. Vijayalakshmi. (2019). Enhancing the security of cloud data using hybrid encryption algorithm. *Journal of Ambient Intelligence and Humanized Computing*, pp. 1-10.

[20] M. N. Fauzan, M. Alif, and C. Prianto. (2023). Comparison of Huffman Algorithm and Lempel Ziv Welch Algorithm in Text File Compression. *IT Journal Research and Development*, vol. 7, no. 2, pp. 184-197.

[21] H. Abroshan. (2021). A hybrid encryption solution to improve cloud computing security using symmetric and asymmetric cryptography algorithms. *International Journal of Advanced Computer Science and Applications*, vol. 12, no. 6, pp. 31-37, doi: 10.14569/IJACSA.2021.0120604.

[22] N. Sangwan. (2012). Text encryption with Huffman compression. *International Journal of Computer Applications*, vol. 54, no. 6.

[23] W. A. Shukur, L. K. Qurban, and A. Aljuboori. (2023). Digital Data Encryption Using a Proposed W-Method Based on AES and DES Algorithms. *Baghdad Science Journal*.

[24] A. M. Al-Smadi, A. Al-Smadi, R. M. Ali Aloglah, N. Abu-Darwish, and A. Abugabah. (2021). Files cryptography based on one-time pad algorithm. *International Journal of Electrical and Computer Engineering (IJECE)*, vol. 11.

[25] M. N. Alenezi, H. Alabdulrazzaq, and N. Q. Mohammad. (2020). Symmetric encryption algorithms: Review and evaluation study. *International Journal of Communication Networks and Information Security*, vol. 12, no. 2, pp. 256-272.

[26] S. Shanmugasundaram and R. Lourdusamy. (2011). A comparative study of text compression algorithms. *International Journal of Wisdom Based Computing*, vol. 1, no. 3, pp. 68-76.

[27] Abdo, A., Karamany, T.S. and Yakoub, 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), p.101999.