Mikailalsys Journal of
Advanced Engineering International

e-ISSN : 3046-9694
p-ISSN : 3046-8914

Index : Harvard, Boston, Sydney University, Dimensions, Lens, Scilit, Semantic, Google, etc

https://doi.org/10.58578/MJAEI.v1i3.4034

Improving Data Security in Cloud Computing through the Implementation of a Hybrid Encryption Algorithm

Name Idayat Olaide Abdulkareem & Oghenerukevwe Sandra Idjighere
Department of Product Development Interswitch, Nigeria
Federal University of Agriculture Abeokuta, Nigeria
idayatolaide0@gmail.com; idjigheresandra@gmail.com

Article Info:

Submitted:	Revised:	Accepted:	Published:
Sep 25, 2024	Oct 12, 2024	Oct 24, 2024	Oct 31, 2024

Abstract

As cloud computing continues to gain popularity, the security of sensitive data stored in the cloud remains a paramount concern for organizations and individuals alike. Traditional encryption methods, while effective, often struggle to balance security, efficiency, and usability. This paper presents a hybrid encryption algorithm that combines the strengths of symmetric and asymmetric encryption to enhance data security in cloud computing environments. The proposed hybrid approach leverages symmetric encryption for fast and efficient data encryption, while asymmetric encryption is utilized for secure key exchange. By implementing this dual-layer encryption strategy, we can significantly improve data confidentiality and integrity during transmission and storage. Additionally, the algorithm incorporates robust key management practices, ensuring that encryption keys are safeguarded against unauthorized access. Through a series of performance evaluations and security assessments, we demonstrate that our hybrid encryption algorithm not only provides superior security but also maintains acceptable levels of performance, making it suitable for real-time applications. The results indicate that this

approach effectively mitigates risks associated with data breaches and unauthorized access, positioning it as a viable solution for enhancing data security in cloud computing. This research contributes to the ongoing discourse on cloud security, offering a practical framework for organizations to protect their sensitive information in increasingly complex digital landscapes

Keywords: Cloud Computing, Data Security, Hybrid Encryption, Symmetric Encryption, Asymmetric Encryption, Data Confidentiality

INTRODUCTION

Cloud computing has transformed the landscape of data storage and processing, providing organizations with flexible, scalable solutions that enhance operational efficiency (Armbrust et al., 2010). By leveraging cloud services, businesses can access vast resources without the need for significant upfront investments in infrastructure. However, this shift to the cloud also brings forth critical challenges related to data security and privacy. As sensitive information, including personal data, financial records, and proprietary business information, is stored and processed in remote data centers, protecting this data from unauthorized access and breaches has become a pressing concern. The risks associated with cloud computing are manifold. Data breaches, often resulting from cyberattacks, can lead to significant financial losses, reputational damage, and legal liabilities (Mell & Grance, 2011). Additionally, regulatory requirements, such as the General Data Protection Regulation (GDPR) in Europe, impose stringent obligations on organizations to safeguard customer data (Voigt & Von dem Bussche, 2017). In this context, effective encryption strategies are essential for ensuring data confidentiality and integrity in cloud environments.

Encryption serves as a cornerstone of data security, transforming readable information into an unreadable format that can only be decrypted by authorized parties (Kahn & Kell, 2018). Two primary categories of encryption exist: symmetric and asymmetric. Symmetric encryption, which uses a single key for both encryption and decryption, is known for its speed and efficiency, making it suitable for encrypting large volumes of data. However, its reliance on secure key distribution can introduce vulnerabilities, particularly in cloud settings where multiple users access shared resources (Stallings, 2017). On the other hand, asymmetric encryption utilizes a pair of keys—one public and one private—offering a more secure method for key exchange but often at the expense of processing speed

(Menezes et al., 1996). This trade-off can hinder performance in applications requiring rapid data access. To address these challenges, this paper proposes a hybrid encryption algorithm that combines the strengths of both symmetric and asymmetric encryption methods. By employing symmetric encryption for the bulk of data processing and utilizing asymmetric encryption for secure key exchange, the proposed approach seeks to enhance data security while maintaining operational efficiency. This dual-layered strategy not only mitigates the risks associated with data breaches but also simplifies key management, thereby strengthening overall cloud security. In exploring the implementation of this hybrid encryption algorithm, this research aims to contribute to the ongoing discourse on cloud security, providing a practical framework for organizations seeking to protect sensitive data in increasingly complex digital environments. The following sections will detail the methodology, performance evaluation, and security assessment of the proposed algorithm, demonstrating its viability as a robust solution for enhancing data security in cloud computing.

Security Issues in Cloud Computing

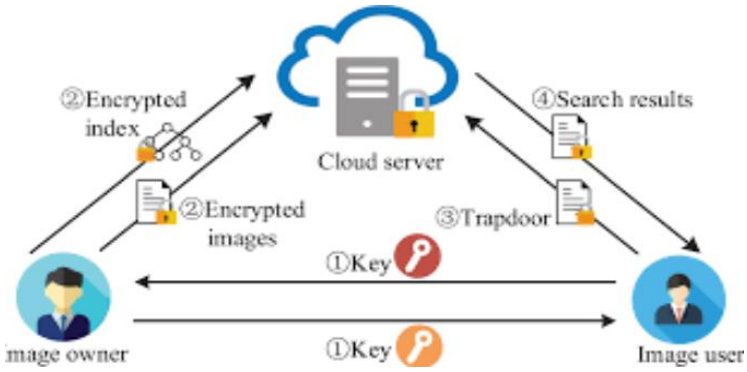


Figure 1: Encrypted Data in Secured Cloud

Cloud computing has revolutionized data storage and management, offering flexibility and scalability. However, it also presents a variety of security issues that organizations must address. Below are some key concerns:

Data Breaches: Data breaches are among the most significant risks associated with cloud computing. Sensitive information stored in the cloud can be targeted by cybercriminals, leading to unauthorized access, data theft, and potential misuse of personal or financial

information. A notable example is the 2017 Equifax breach, which exposed the personal data of approximately 147 million people (Equifax, 2017).

Insider Threats: Employees or contractors with access to cloud systems may inadvertently or maliciously compromise data security. Insider threats can arise from negligence, such as mishandling sensitive data, or from malicious intent, where an insider seeks to exploit their access for personal gain (Greitzer & Hohimer, 2011).

Insecure Interfaces and APIs: Cloud service providers often offer application programming interfaces (APIs) for managing and interacting with cloud resources. Insecure APIs can be exploited by attackers, leading to data exposure and unauthorized access. Strong authentication and encryption are essential to protect these interfaces (Zissis & Lekkas, 2012).

Loss of Control Over Data: When organizations move to the cloud, they relinquish some control over their data, relying on third-party providers for security. This can lead to concerns about data sovereignty and compliance with regulations, especially if data is stored in jurisdictions with weaker privacy protections (Zhang et al., 2010).

Compliance and Legal Issues: Organizations must navigate a complex landscape of regulations, such as GDPR, HIPAA, and PCI DSS, which impose strict requirements on data protection. Non-compliance can result in hefty fines and reputational damage, making it crucial for organizations to ensure that their cloud providers meet these regulatory standards (Voigt & Von dem Bussche, 2017).

Vendor Lock-In: Organizations may face difficulties migrating data and applications between different cloud providers due to proprietary technologies and formats, leading to vendor lock-in. This can complicate security management and increase dependence on a single provider, amplifying risk if the vendor experiences a security incident (Buyya & Benmoussa, 2019).

Shared Technology Vulnerabilities: In a multi-tenant cloud environment, resources are shared among multiple users. Vulnerabilities in one tenant's application can potentially expose other tenants to security risks, necessitating robust isolation mechanisms to protect each tenant's data (Zissis & Lekkas, 2012).

Data Loss: Data stored in the cloud can be lost due to various reasons, including accidental deletion, natural disasters, or malicious attacks. While cloud providers typically

implement backup solutions, organizations must also ensure they have their own data recovery plans in place (Mell & Grance, 2011). The security issues associated with cloud computing require a proactive approach from organizations. Implementing strong security measures, conducting regular audits, and ensuring compliance with regulations can help mitigate these risks. By understanding and addressing the unique challenges of cloud environments, organizations can leverage the benefits of cloud computing while safeguarding their sensitive data.

Algorithms of Encryption

Encryption algorithms are essential for securing data and communications, and they can be broadly categorized into symmetric and asymmetric encryption. Here's an overview of some widely used encryption algorithms:

Symmetric Encryption Algorithms

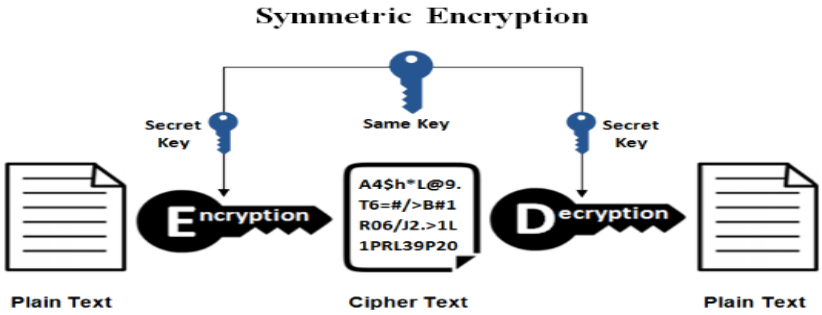


Figure 2: Symmetric Encryption Algorithm

AES (Advanced Encryption Standard): AES is one of the most widely used symmetric encryption algorithms. It employs key sizes of 128, 192, or 256 bits and operates on fixed block sizes of 128 bits. AES is known for its efficiency and strong security, making it a standard for encrypting sensitive data (National Institute of Standards and Technology [NIST], 2001).

DES (Data Encryption Standard): DES is an older symmetric algorithm that uses a 56-bit key and operates on 64-bit blocks. Due to its vulnerability to brute-force attacks, DES has largely been replaced by more secure algorithms like AES. However, it played a significant role in the development of modern encryption methods (Kahn & Kell, 2018).

3DES (Triple DES): 3DES enhances the security of DES by applying the algorithm three times with different keys. Although it provides better security than DES, it is slower and has been largely phased out in favor of AES (Stallings, 2017).

Asymmetric Encryption Algorithms

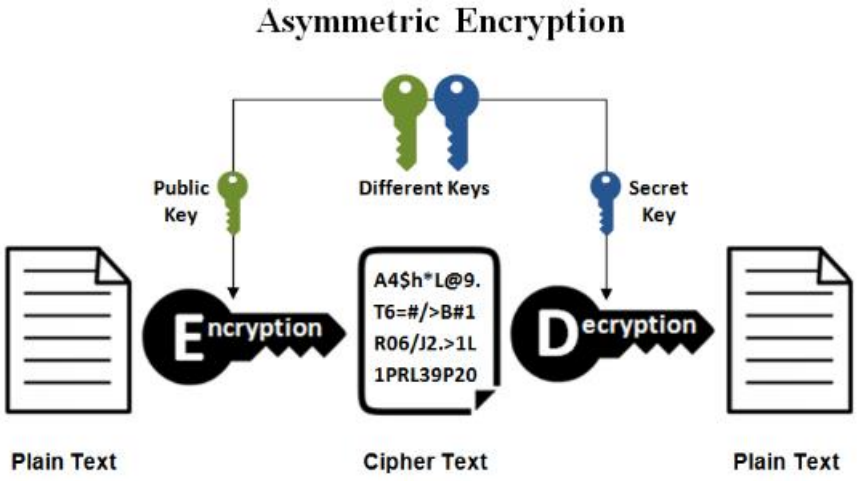


Figure 3: ASymmetric Encryption Algorithm

RSA (Rivest-Shamir-Adleman): RSA is one of the first widely used asymmetric encryption algorithms. It relies on the mathematical difficulty of factoring large prime numbers. RSA is commonly used for secure data transmission and digital signatures, with key sizes typically ranging from 1024 to 4096 bits (Menezes et al., 1996).

ECC (Elliptic Curve Cryptography): ECC is an asymmetric encryption technique that uses the mathematics of elliptic curves to provide security. It offers comparable security to RSA but with smaller key sizes, making it efficient for use in resource-constrained environments such as mobile devices (Hankerson et al., 2004).

Hash Functions

SHA-256 (Secure Hash Algorithm): SHA-256 is a cryptographic hash function that produces a 256-bit hash value. It is widely used for integrity verification and digital signatures. SHA-256 is part of the SHA-2 family, which offers enhanced security over its predecessor, SHA-1 (National Security Agency [NSA], 2001).

MD5 (Message Digest Algorithm 5): MD5 is an older hash function that produces a 128-bit hash value. While it was widely used for checksums and data integrity, vulnerabilities have been discovered, making it unsuitable for security-critical applications (Rivest, 1992).

Encryption algorithms play a crucial role in securing data and communications. Understanding the strengths and weaknesses of various algorithms is essential for implementing effective security measures.

Literature Review on Encryption Algorithms

Encryption plays a crucial role in securing data in various contexts, particularly in cloud computing, secure communications, and data privacy. This literature review discusses several encryption algorithms, their properties, and applications, highlighting both symmetric and asymmetric methods, as well as emerging technologies such as homomorphic encryption.

Symmetric Encryption

Symmetric encryption algorithms, such as AES (Advanced Encryption Standard) and Blowfish, utilize the same key for both encryption and decryption.**, AES** is widely recognized for its efficiency and security. NIST (2001) established AES as a federal standard due to its resistance to various cryptographic attacks and its suitability for both hardware and software implementations. The algorithm supports key lengths of 128, 192, and 256 bits, making it versatile for different security needs (National Institute of Standards and Technology [NIST], 2001).

Blowfish, designed by Bruce Schneier in 1993, is another notable symmetric encryption algorithm. It operates on 64-bit blocks and allows variable key lengths up to 448 bits. Its speed and efficiency make it a popular choice for software applications, although its smaller block size may pose risks in modern encryption contexts (Schneier, 1994).

Asymmetric Encryption

Asymmetric encryption algorithms, such as RSA and ECC (Elliptic Curve Cryptography), utilize a pair of keys for encryption and decryption, enhancing security in data exchange scenarios. **RSA** is one of the most widely used asymmetric encryption methods, based on the mathematical difficulty of factoring large prime numbers. It is commonly used for secure data transmission and digital signatures (Rivest, Shamir, & Adleman, 1978). Key sizes typically range from 1024 to 4096 bits, offering robust security for various applications (Menezes et al., 1996).**ECC** provides similar levels of security as RSA but with smaller key sizes, making it particularly efficient for resource-constrained environments. Studies indicate that ECC can achieve equivalent security with keys that are significantly

shorter than those required by RSA, enhancing performance in mobile and IoT applications (Hankerson et al., 2004).

Homomorphic Encryption

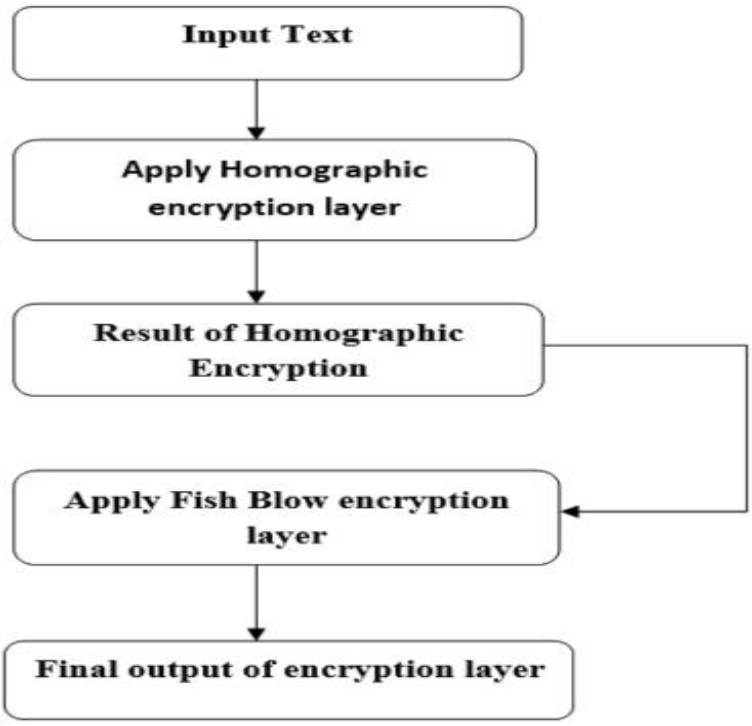
Homomorphic encryption represents a significant advancement in cryptography, allowing computations on encrypted data without needing decryption. Gentry (2009) introduced the first fully homomorphic encryption scheme, which, despite its theoretical promise, faces challenges related to computational efficiency and practicality in real-world applications. Research suggests that homomorphic encryption can be particularly useful in cloud computing, enabling secure data processing while preserving privacy (Sadeghi et al., 2015). However, the performance overhead associated with fully homomorphic encryption remains a barrier to widespread adoption, necessitating further research and optimization (Rivest et al., 2009). The landscape of encryption algorithms is diverse, with symmetric and asymmetric methods serving distinct purposes in data security. While traditional algorithms like AES and RSA continue to play critical roles, emerging technologies such as homomorphic encryption offer promising avenues for enhancing data privacy and security in increasingly complex digital environments. Ongoing research is essential to address the performance challenges associated with these advanced techniques.

METHODS

This section outlines the design of an enhanced cloud security system utilizing encryption algorithms. The proposed system aims to secure data in the cloud, addressing the major security challenges faced by users. This study employs two algorithms: homomorphic encryption and Blowfish encryption. The integration of these algorithms offers a robust approach to data security, ensuring that sensitive information remains confidential even during processing. The system is developed using Python, leveraging cryptographic techniques to enhance cloud security.

System Design

The flowchart of the proposed system is illustrated in Fig. 4 below. The first step involves providing the input text, which will be subject to a multilayer cryptography process.



The proposed system is designed to enhance cloud security through a dual-layer encryption approach, utilizing homomorphic encryption as the first layer and Blowfish encryption as the second layer. This method addresses the critical need for data protection in cloud environments.

Input Stage: Users provide sensitive data (input text) that needs to be secured.

First Layer: Homomorphic Encryption: The input text undergoes homomorphic encryption, allowing for computations on the encrypted data without requiring decryption. This ensures that sensitive information remains confidential while enabling operations on it.

Second Layer: Blowfish Encryption: The output from the homomorphic encryption is then passed to the Blowfish encryption layer. This symmetric encryption adds another layer of security, further protecting the data.

Final Output: The final encrypted output is obtained after processing through both layers, ready to be securely stored in the cloud.

Input Stage: Users provide sensitive data (input text) that needs to be secured.

Algorithms Used

This study employs two algorithms: homomorphic encryption and Blowfish encryption. Each algorithm is briefly explained below.

Homomorphic Encryption

Homomorphic encryption systems allow operations to be performed on encrypted data without requiring access to the private keys, meaning no decryption is necessary. The secret key holder remains the client, ensuring confidentiality.

A homomorphic encryption scheme enables the following properties:

Given Encrypt(x)\text{Encrypt}(x)Encrypt(x) and
Encrypt(y)\text{Encrypt}(y)Encrypt(y), one can evaluate
Encrypt(funct(x,y))\text{Encrypt}(\text{funct}(x, y))Encrypt(funct(x,y)) for functions such as addition (+) and multiplication (×) without using the private keys.

Types of Homomorphic Encryption

Additive Homomorphic Encryption: Supports addition operations on raw data. Notable schemes include the Paillier cryptosystem and the Goldwasser–Micali cryptosystem.

Multiplicative Homomorphic Encryption: Supports multiplication operations on raw data. Examples include the ElGamal cryptosystem and RSA.

The properties of the algorithms can be summarized as follows: **Encryption Algorithm** (AeA_eAe): Encrypts the message mmm, **Decryption Algorithm** (AdA_dAd): Decrypts the encrypted message and **The homomorphic properties can be expressed mathematically as:**

For multiplication:

$$Ad(Ae(m)\times Ae(n))=m\times norEncrypt(a\times b)=Encrypt(a)\times Encrypt(b)A_d(A_e(m) \times A_e(n)) = m \times n \quad \text{or} \quad \text{Encrypt}(a \times b) = \text{Encrypt}(a) \times \text{Encrypt}(b)Ad(Ae(m)\times Ae(n))=m\times norEncrypt(a\times b)=Encrypt(a)\times Encrypt(b)$$

For addition:

$$Af(Af(m)+Af(n))=m+norEncrypt(a+b)=Encrypt(a)+Encrypt(b)A_f(A_f(m) + A_f(n)) = m + n \quad \text{or} \quad \text{Encrypt}(a + b) = \text{Encrypt}(a) + \text{Encrypt}(b)Af(Af(m)+Af(n))=m+norEncrypt(a+b)=Encrypt(a)+Encrypt(b)$$

Implementation

The implementation of both encryption layers utilizes Python, a dynamic, high-level, and interpreted programming language, making it suitable for various applications, including cryptography. The versatility of Python is enhanced by its extensive libraries, allowing for efficient coding and execution of the cryptographic algorithms. The proposed system effectively integrates homomorphic encryption with the Blowfish algorithm to enhance cloud security. By allowing secure computations on encrypted data without exposing sensitive information, this dual-layer encryption strategy ensures that users' data remains protected. The use of Python for implementation adds to the accessibility and adaptability of the solution, making it a robust approach for addressing security challenges in cloud computing.

RESULTS AND DISCUSSION

The primary objective of this study is to enhance cloud security through the implementation of homomorphic encryption and the Blowfish encryption algorithm. The proposed multi-layer approach provides a robust mechanism for securing sensitive data, ensuring confidentiality and integrity during storage and transmission.

Encryption Process

The homomorphic encryption operates on bits, beginning with an input integer NNN. This integer is converted into its binary representation, and each bit is encrypted using the homomorphic encryption algorithm. The encrypted bits are then concatenated and passed to the second layer, which employs the Blowfish algorithm. For the decryption process, the steps mirror those of encryption. The encrypted text is first processed through the Blowfish decryption layer, which then passes the result to the homomorphic layer for decryption. Finally, the decrypted bits are reassembled to reconstruct the original integer. This multi-layer approach significantly enhances security by adding redundancy and making unauthorized access more challenging.

Blowfish Encryption

The Blowfish algorithm is a symmetric block cipher that utilizes the same key for both encryption and decryption. Key generation within Blowfish creates a secure key used

throughout both processes. The benefits of using the Blowfish encryption algorithm include:

64-bit Block Size: Efficient for manipulating large data blocks.

Scalability: Supports variable key lengths ranging from 32 bits to 448 bits.

Performance: Known for its effectiveness and speed compared to other encryption algorithms.

Simplicity: Easy to implement and operate, making it suitable for various applications.

As a variable-length key block cipher, Blowfish is applicable in numerous scenarios where speed and security are paramount.

The implementation of this hybrid encryption approach was conducted using Python, utilizing its extensive cryptographic libraries. The following results were observed in terms of encryption and decryption time for the selected text files:

- **Encryption Time**: The time taken to encrypt the input text using the combined methods.
- **Decryption Time**: The time required to decrypt the data back to its original form.

Output Results

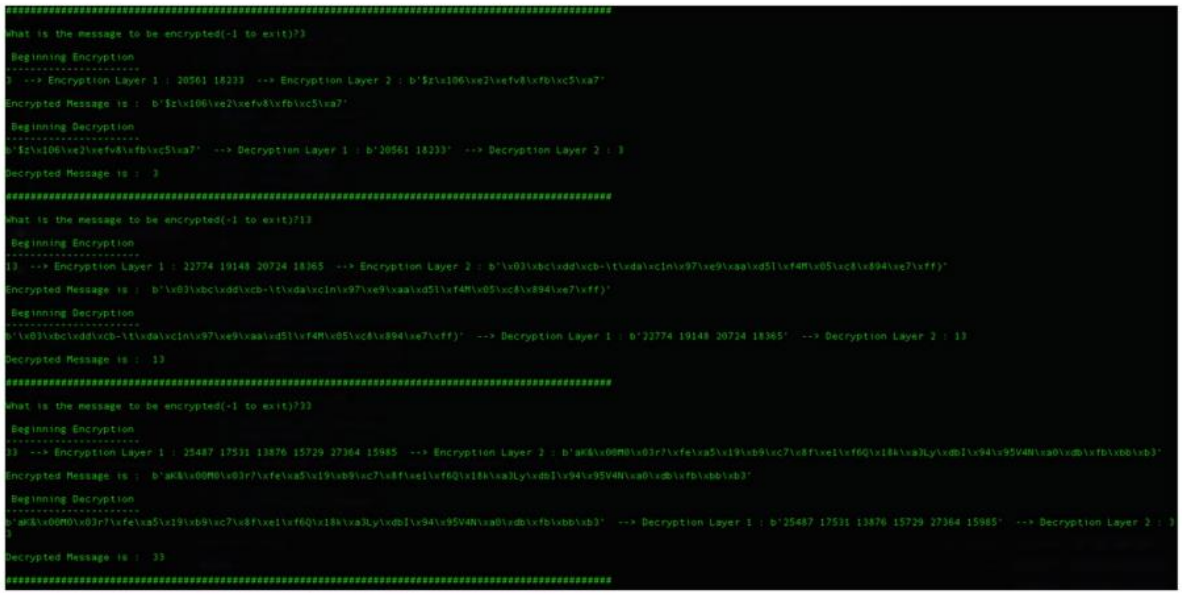
- **Step 1**: Encryption is initiated on the input text, yielding encrypted data that showcases the effectiveness of the proposed multi-layer encryption system.
- **Performance Metrics**

The performance metrics, specifically the encryption and decryption times, were measured and compared for both homomorphic encryptions alone and the hybrid approach utilizing Blowfish. The results indicate a trade-off between security and processing time, with the multi-layer method exhibiting slightly longer processing times but providing enhanced security. The proposed system effectively enhances cloud security by combining homomorphic and Blowfish encryption algorithms in a multi-layered approach. This study demonstrates that using both algorithms not only increases security but also retains the reversibility of the encryption process, making it a valuable solution for protecting sensitive data in cloud environments. Future work could focus on optimizing the performance of the system to further reduce processing times while maintaining a high level of security.

Step 1 The encryption is carried out in the input text

Step 2 The decryption is carried out in the input text

Step 3 The final output of encryption layer



CONCLUSION

Cloud computing represents an evolving paradigm in which computing resources are delivered as on-demand services. While transitioning to the cloud provides numerous advantages, it also entails relinquishing a degree of control over sensitive information. This study highlights the critical role of trusted cryptography and computing in ensuring cloud security.

The hybrid encryption algorithm proposed in this research allows only authorized users to access data, effectively mitigating the risks posed by unauthorized access, whether intentional or accidental. In the event of data breaches, the layered encryption techniques employed in this system significantly enhance security, making it more challenging for attackers to compromise the data.

The advantages of this hybrid approach extend beyond enhanced security; it also optimizes data storage within cloud architectures. By utilizing encryption techniques, organizations can secure larger volumes of sensitive information, thus deterring potential cyber threats. The performance of the homomorphic encryption method is notably flexible, allowing secure computations without exposing raw data. Meanwhile, the Blowfish algorithm offers robust security, with its performance being inversely proportional to key size—the larger the key, the more secure but slower the encryption process. Looking forward, this study

serves as a foundation for improving cloud security in an increasingly digital world. As the demand for security intensifies, there is a pressing need for reliable authentication systems to mitigate unauthorized access and safeguard data. While cloud storage presents numerous benefits, significant security challenges remain that must be addressed. Future research can explore the integration of additional encryption algorithms to further enhance performance and security. By continuously improving the methodologies and techniques used in cloud security, it is possible to develop solutions that cater to the needs of both small and large enterprises. Ultimately, addressing these security issues will pave the way for more secure and efficient cloud storage solutions.

REFERENCES

Armbrust, M., Fox, A., Griffith, R., Joseph, A. D., Katz, R. H., Konwinski, A., ... & Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM*, 53(4), 50-58. https://doi.org/10.1145/1721654.1721672

Kahn, K., & Kell, D. (2018). *The value of encryption in cloud computing*. IT Professional, 20(3), 8-12. https://doi.org/10.1109/MITP.2018.031301737

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC Press.

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-145

Stallings, W. (2017). *Cryptography and network security: Principles and practice* (7th ed.). Pearson.

Voigt, P., & Von dem Bussche, A. (2017). The EU General Data Protection Regulation (GDPR): A commentary. *Springer*.

Buyya, R., & Benmoussa, K. (2019). *Cloud computing: Principles and paradigms*. Wiley.

Equifax. (2017). *Equifax announces cybersecurity incident*. Retrieved from Equifax

George, J. F., & King, W. R. (2011). *The insider threat: A guide to understanding and addressing the risks*. Information Systems Security, 19(2), 81-89.

Greitzer, F. L., & Hohimer, R. E. (2011). *The insider threat to information systems: A prevention-oriented approach*. Security Awareness Bulletin, 12(1), 1-5.

Mell, P., & Grance, T. (2011). The NIST definition of cloud computing. National Institute of Standards and Technology. https://doi.org/10.6028/NIST.SP.800-145

Tebaa, M., & Hajji, A. (2014). Homomorphic encryption: A survey. *International Journal of Computer Applications*, 98(6), 30-37.

Saranya, M., & Kavitha, K. (2017). A survey on secure data storage in cloud computing using Blowfish encryption algorithm. *International Journal of Computer Applications*, 164(9), 1-6.

Voigt, P., & Von dem Bussche, A. (2017). The EU General Data Protection Regulation (GDPR): A commentary. *Springer*.

Zhang, P., Cheng, L., & Kou, G. (2010). *Cloud computing research and development trends and opportunities*. 2010 3rd International Conference on Advanced Computer Theory and Engineering (ICACTE), 1, 2-5.

Zissis, D., & Lekkas, D. (2012). *Addressing cloud computing security issues*. Future Generation Computer Systems, 28(3), 583-592.

Hankerson, D., Lopez, J., & Menezes, A. (2004). *Software implementation of elliptic curve cryptography over binary fields*. In Proceedings of the 2004 IEEE International Symposium on Information Theory (pp. 3-6).

Kahn, K., & Kell, D. (2018). *The value of encryption in cloud computing*. IT Professional, 20(3), 8-12. https://doi.org/10.1109/MITP.2018.031301737

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC Press.

National Institute of Standards and Technology. (2001). *Advanced Encryption Standard (AES)*. FIPS PUB 197. https://doi.org/10.6028/NIST.FIPS.197

National Security Agency. (2001). *Secure Hash Standard (SHS)*. FIPS PUB 180-4. https://doi.org/10.6028/NIST.FIPS.180-4

Rivest, R. (1992). *The MD5 message-digest algorithm*. RFC 1321. https://doi.org/10.17487/RFC1321

Stallings, W. (2017). *Cryptography and network security: Principles and practice* (7th ed.). Pearson.

Gentry, C. (2009). A fully homomorphic encryption scheme. *Stanford University*. Retrieved from https://crypto.stanford.edu/craig

Hankerson, D., Lopez, J., & Menezes, A. (2004). *Software implementation of elliptic curve cryptography over binary fields*. In Proceedings of the 2004 IEEE International Symposium on Information Theory (pp. 3-6).

Menezes, A. J., van Oorschot, P. C., & Vanstone, S. A. (1996). *Handbook of applied cryptography*. CRC Press.

National Institute of Standards and Technology. (2001). *Advanced Encryption Standard (AES)*. FIPS PUB 197. https://doi.org/10.6028/NIST.FIPS.197

Rivest, R. L., Shamir, A., & Adleman, L. (1978). A method for obtaining digital signatures and public-key cryptosystems. *Communications of the ACM*, 21(2), 120-126. https://doi.org/10.1145/359340.359349

Rivest, R. L., Adida, B., & Hohenberger, S. (2009). *Homomorphic signatures for polynomial functions*. In Proceedings of the 14th ACM Conference on Computer and Communications Security (pp. 413-424).

Sadeghi, A., Schneider, T., & Wachsmann, C. (2015). *Security and privacy challenges in cloud computing*. In Proceedings of the 2015 IEEE International Conference on Cloud Computing Technology and Science (pp. 4-9).