Solving Time Complexity Issues in Storage Area Network using Enhanced Homomorphic Encryption

Asimini-Hart K. Roseline, Nuka D. Nwiabu, Onate E. Taylor, and V.I.E Anireh
Department of Computer Science
Rivers State University (RSU), Port Harcourt, Rivers State, Nigeria.
Contact: +234 8064134113
Corresponding author email: kalalali.asimini-hart1@ust.edu.ng
DOI: 10.56201/ijcsmt.v10.no6.2024.pg73.82

Abstract

In the era of cloud computing and big data, ensuring secure and efficient data storage is paramount. Storage Area Networks (SAN) play a critical role in providing high-speed data access; however, they face significant challenges related to time complexity in encryption processes. Traditional encryption methods often compromise performance, leading to increased latency and inefficiencies. This study addresses these issues by proposing an Enhanced Homomorphic Encryption (EHE) model that integrates a camouflage process and a substitution-based key generation method. The EHE model not only enhances data security but also optimizes encryption and decryption times. Experimental results reveal that EHE significantly outperforms both Rivest-Shamir-Adleman (RSA) and standard Homomorphic Encryption (HE) techniques, achieving a processing time reduction of 11 seconds compared to 12 seconds for RSA and 250 seconds for HE. This research underscores the need for advanced encryption methodologies in modern computing environments and recommends further exploration of EHE's applications in various data security contexts.

1. Introduction

In today's digital landscape, the proliferation of cloud computing and data-intensive applications necessitates robust and efficient data security solutions. Storage Area Networks (SAN) serve as critical infrastructures designed to provide high-speed access and sharing capabilities among multiple devices and servers [16]. These networks are engineered to optimize data storage and retrieval, thereby enhancing input/output performance for enterprise applications [13]. However, as the volume of sensitive data increases, so does the demand for advanced security measures to protect against unauthorized access and data breaches.

Technically, SAN offers secured, scalable and redundant storage solutions by employing cryptosystem, access control, and disaster recovery strategies [9]. However, in the context of cloud computing, it enables virtualized storage resources that can be dynamically allocated and managed, thereby providing flexibility and scalability for cloud-based applications and services [2]. The different types of storage network include Direct Access Storage (DAS), Network-Attached Storage (NAS), Cloud Storage, and High Storage.

In recent years, various database programs have gained publicity with security issues taken precedence. Majority of these security activities are focused on cloud-based big data system and its application. However, as the need for data protection increases, so does the advancement of cryptographic algorithms. The distinct design of database implementations, as well as their emphasis on rapid resource distribution, has emphasized the need for more complex, modem, and high-performance encryption algorithms. Homomorphic encryption is a new method for safeguarding raw data while promoting Privacy- Preserving Data Processing (PPDP) Systems.

This cryptosystem has emerged as a transformative approach in the field of cryptography, enabling computations on encrypted data without the need for decryption (Belland *et al*., 2017). This characteristic is particularly beneficial in cloud environments, where data privacy is paramount. Despite its advantages, the implementation of homomorphic encryption is often hindered by significant computational overhead, leading to increased encryption and decryption times [15].

This study focuses on developing an Enhanced Homomorphic Encryption (EHE) model that addresses this time complexity issues within SAN environments. By leveraging a camouflage process and a substitution-based algorithm, our proposed method enhances both the security and efficiency of data encryption. Through comprehensive experimentation and analysis, we aim to demonstrate the practical applicability of EHE in securing data while minimizing latency. This paper will explore existing literature on homomorphic encryption, outline the methodology employed in our research, and present the results of our performance evaluations, ultimately contributing to the ongoing discourse on cryptographic advancements in data security.

2. Related Works

Homomorphic Encryption is a feature that an Encryption technique exhibits, some calculation on the data. It is an advanced cryptographic method that enables computation on encrypted data without first decrypting it [3]. Researchers focused their discussions on different security and privacy challenges in cloud computing environments. Their discussion was on the different approaches to addressing the challenges, obstacles and solutions to systems to have a trust and worthy Storage Area Network environment for the Storage Area Network users [10].

Smart and Vercauteren [14] introduce a fully homomorphic encryption scheme with limited key and encrypted text sizes. This structure presented by these researchers is premised on Gentry's ideal lattices construction. It creates a completely homomorphic scheme in a certain manner. Public and private keys comprise of two large entities for a somewhat homomorphic scheme, most of which are exchanged between the public and the private keys: and the chip text comprises one large integer.

Researchers conducted a study on "Secure cloud computing: Benefits, Risks, and Controls." The authors concentrated on cloud security risk, management as a crucial step to establishing safe cloud infrastructure. As a standard guide, they provided an overview of cloud computing rewards and privacy concerns to assist management in the implementation of cloud processes, procedures, and control [5].

Authors in their paper proposed the implementation of the RSA algorithm where a legitimate user can have unique access to data. Even if an intruder obtains the data, he/she will not be able to unlock it and recover any content of it [12].

Authors in their paper “Security Threats and Countermeasures in Cloud Computing” discussed various security threats and the countermeasures in the Storage Area Network computing are discussed in detail to showcase their values in different aspects to enhance the security in Storage Area Network environment to safeguard the data that are stored [17].

[8] conducted research on the Simple Fully Homomorphic Encryption Scheme Available in Cloud Computing. The algorithm was based on the Gentry cryptosystem and used only modular arithmetic. The suggested method can preserve confidentiality in an unreliable third-party cloud. Nevertheless, no safety evaluation was conducted to showcase the veracity of the SDC scheme.

[7] in their paper titled “Homomorphic Hybrid Encryption for Cloud Computing” presented a key generation from the Paillier cryptosystem by using a private and public key which is followed by Elgamal Algorithm. The proposed methodology achieves a secure data over the Storage Area Network by using the renewed hybrid algorithm achieves high efficiency and protection of a data by the Homomorphic Encryption algorithm rule done with the Homomorphic Elgamal Encryption and Homomorphic Paillier Encryption.

[1] researched on "A Searchable Encryption of CP-ABE Scheme in Cloud Storage." They investigated the Cipher-text-policy attribute-based encryption algorithm and analyzed the relevance of security assurances and success regulation in cloud computing.

[18] wrote a paper titled "Using Fully Homomorphic Encryption to Secure Cloud Computing." This paper discusses how homomorphic encryption can be used to encrypt data in a cloud, as well as the ability to perform necessary arithmetic operations on the data that was encrypted. The spacing between both the client and the physical location of his data creates a barrier because this data can be accessed by a third party, compromising the client's data's privacy. The distance between the client and the physical location of his data acts as a barrier since a third party can access it, hence, compromising the client's data privacy.

[6] proposed a hybrid homomorphic crypto algorithm that integrates Public Key Encryption (PKE) and somewhat homomorphic encryption (SHE), lowering the storage costs for homomorphic applications.

[11] discussed sharing of services in data, compared it with the user requirement choice of parameters and surveyed it by using different cryptography techniques of the algorithm in the Storage Area Network environment. These authors did not carry out a comparative analysis on the various cryptographic algorithms.

A survey of homomorphic encryption techniques for cloud computing data security was conducted by [4]. Their study gives an overview of homomorphic encryption and the latest condition in this field. The study also mentions problems in homomorphic encryption and provided even more guidelines for future researchers. The study provides an overview of homomorphic encryption technologies for cloud computing data security.

3. Methods

Our study employed a constructive research method for carrying out the research construct, with a strong focus on experimental design and performance analysis. We explored and analyzed the existing standards of Fully Homomorphic Encryption (FHE) schemes by conducting a comprehensive literature review. Our system employed camouflaged encrypted text and camouflaged plaintext. The encryption sub-algorithm uses the secret key, plaintext and random error. The camouflage process can be applied to the original message before encryption or after decryption. The encryption algorithm generates the ciphertext depending on the camouflage process. Similarly, the generated ciphertext can be decrypted using the secret key as by reversing the applied camouflage process, to obtain the original plaintext.

Further, we utilized the Substitution Homomorphic Encryption Box (SHE-box) and the random selection method as a 2-step procedure required for generating crypto-keys. Furthermore, the proposed algorithm allows for performing homomorphic addition and multiplication on the ciphertext. Finally, we implemented our model using python programming language.

Proposed Algorithm: The proposed algorithm is the addition of camouflage and Substitution Homomorphic Encryption Box for key generation and Fully Homomorphic Encryption (FHE) was used in place of Partial Homomorphic Encryption (PHE).
Here, the pseudocode of two different variants of the proposed, enhanced homomorphic encryption model for preserving privacy is given as follows:

Variant 1: Camouflaged Encrypted Text

The variant 1, named Camouflaged encrypted text, a camouflage process is applied to the encrypted text C_M, obtained as a resultant after encrypting theoriginal plaintext M. After applying the camouflage process over encrypted text C_M, the camouflaged ciphertext, C_{cam} can be obtained.

Sub-Algorithm1: $\boldsymbol{KeyGen}$: Key Generation Algorithm

```
i.   Generate 256 random prime numbers as $P_i$, where $P_i \in [2^1, 2^\lambda]$,
     $i \in (0, \ldots 255)$ $\lambda$, a security parameter
     $\in \mathbb{N}$ and $\mathbb{N} \in [1, n]$.
ii.  Generate substitution box SHE -Box, by filling the matrix of size, $16\times 16$
     using $P_i$.
iii. Return the secret key as $S_{sk}$ using the random selection method.
     a. Generate a random number, $R$, and convert it into its hexadecimalform $R_{hex}$
     b. Select the row and column of SHE -Box using LSB and MSB of
        $R_{hex}$.
```

```
    c. Select the secret key, $S_{sk}$.
```

Sub-Algorithm2: *Encrypt*: Encryption Algorithm

```
i.   Input the plaintext/ message as $M$, where $M \in \mathbb{R}, \mathbb{Z}$.
ii.  Encrypt $M$ to get ciphertext as $C_M$, where $C_M \leftarrow M \times S_{sk}$.
iii. Apply the Camouflage process (CP) to generate camouflaged ciphertextas $C_{cam}$ accordingly.
iv.  Camouflage process (CP):
     a. Find the next prime number as $P_{Next}$, to the ciphertext $C_M$ and calculate $d$ i.e., the difference of $P_{Next}$ and $C_M$.
     b. Find the random error, $(Err)$
     c. $C_{cam} \leftarrow \quad + \quad$.
```

Sub-Algorithm3: *Decrypt*: Decryption Algorithm

```
i.   Apply the Reverse Camouflage process $(CP_{rev})$ to generate originalplaintext, M.
     Reverse Camouflage process $(CP_{rev})$:
     a. $P_{Next} \leftarrow C_{cam} - Err$
     b. $C_M \leftarrow P_{Next} - d$.
ii.  Input $C_M$ and divide it by $S_{sk}$.
iii. Output the Original message, $M$.
```

Sub-Algorithm4: *Eval*: Evaluation Algorithm

```
Generate two camouflaged ciphertexts as $C_{cam}$ and $C_{cam}'$ for two differentplaintexts as $M$ and $M'$ respectively, using an Encryption algorithm (Encrypt).
i.   Evaluate the $C_{cam}$ and $C`_{cam}$ using evaluating function as $F_{eval}$ and it consists of algebraic addition or multiplication operations.
ii.  Calculate the result of an applied operation, as $C_{eval}$ using $f_{eval}$.
     $C_{eval} \leftarrow C_{cam} + C_{cam}'$ or $C_{eval} \leftarrow C_{cam} \times C_{cam}'$
iii. Also, find the result of the applied operation as $M_{eval}$ over the different plaintext, $M$ and $M'$ using $f_{eval}$.
```

$M_{eval} \leftarrow M + M' \text{ or } M_{eval} \leftarrow M \times M'$

Variant 2: Camouflaged Plaintext

Variant 2, named Camouflaged plaintext, as camouflage process is applied to the original plaintext M, and then resulted in camouflage plaintext, $M_{cam}$is encrypted using a specified encryption algorithm, to generate the camouflaged ciphertext, C_{cam}.

Sub-Algorithm1: *KeyGen*: Key Generation Algorithm

```
i.   Generate 256 random prime numbers as $P_i$, where $P_i \in [2^1, 2^\lambda]$, $i \in (0, \ldots 255)$. $\lambda$, a security parameter $\in \mathbb{N}$ and $\mathbb{N} \in [1, n]$.
ii.  Generate substitution box SHE -Box, by filling the matrix of size, $16\times 16$ using $P_i$.
iii. Return the secret key as $S_{sk}$ using the random selection method.
     a. Generate a random number, $R$, and convert it into its hexadecimalform $R_{hex}$
     b. Select the row and column of SHE -Box using LSB and MSB of $R_{hex}$.
Select the secret key, $S_{sk}$.
```

Sub-Algorithm2: *Encrypt*: Encryption Algorithm

```
i.   Input the plaintext/ message as $M$, where $M \in \mathbb{R}, \mathbb{Z}$.
ii.  Apply the Camouflage process (CP) to generate camouflaged plaintext as $M_{cam}$ accordingly.
iii. Camouflage process (CP):
     a. Find the next prime number as $P_{Next}$, to message $M$ and calculate $d$ i.e., the difference of $P_{Next}$ and .
     b. Find the random error ($Err$ )
     c. $M_{cam} \leftarrow \quad + \quad$.
iv.  Encrypt $M_{cam}$ to get camouflaged ciphertext as $C_{cam}$, where, $C_{cam} \leftarrow M_{cam} \times S_{sk}$.
v.   Output the ciphertext, $C_{cam} \leftarrow M_{cam} \times S_{sk}$.
```

Sub-Algorithm3: $\boldsymbol{Decrypt}$: Decryption Algorithm

```
i.   Input $C_{cam}$ and divide it by $S_{sk}$ and obtain the $M_{cam}$.
ii.  Apply the Reverse Camouflage process ($CP_{rev}$) to generate original plaintext, M.
iii. Reverse Camouflage process ($CP_{rev}$):
     a. $P_{Next} \leftarrow M_{cam} - Err$
     b. $M \leftarrow P_{Next} - d$.
iv.  Output the Original plaintext $M$.
```

Sub-Algorithm4: $\boldsymbol{Eval}$: Evaluation Algorithm

```
Generate two camouflaged ciphertexts as $C_{cam}$ and $C'$ for two different plaintexts as $M$ and $M'$ respectively, using an Encryption algorithm (Encrypt).
i.   Evaluate the $C_{cam}$ and $C`_{cam}$ using evaluating function as $F_{eval}$ and it consists of algebraic addition or multiplication operations.
ii.  Calculate the result of the applied operation, as $C_{eval}$ using $f_{eval}$.
     a. $\leftarrow C_{cam} + C'$ or $C_{eval} \leftarrow C_{cam} \times C'_{cam}$
iii. Also, find the result of the applied operation as $M_{eval}$ over the differentplaintext, $M$ and $M'$ using $f_{eval}$.
     a. $M_{eval} \leftarrow M + M'$ or $M_{eval} \leftarrow M \times M'$
iv.  If $C_{eval} = M_{eval}$, for respective $f_{eval}$ then homomorphism is achieved, or else homomorphism fails.
```

4. Results

Table 1 shows the result of implementation carried out in time complexity using three (3) different variables which includes Rivest Shamir Adleman (RSA), Homomorphic Encryption (HE) and the proposed system which is the Enhanced Homomorphic Encryption.

Time complexity aids to evaluate the efficiency of an algorithm, measuring how the runtime of an algorithm grows as the size of the input increases. It gives insight into how scalable an algorithm is when handling large data sets which is vital in determining the practicability of an algorithm to the real-world application. This also aids the optimization of system resources leading to faster execution and more efficient programs.

Table 1 Time Complexity Table

File Size (MB)	RSA (s)	Percentage Difference(%)	HE (s)	Percentage Difference	EHE (S)	Percentage Difference (%)
10	0.5	5.0	5.0	50.0	0.4	4.0
50	1.5	3.0	20.0	40.0	1.2	2.4
100	3.0	3.0	50.0	50.0	2.5	2.5
500	8.0	1.6	150.0	30.0	6.0	1.2
1000	20.0	2.0	400.0	40.0	15.0	1.5

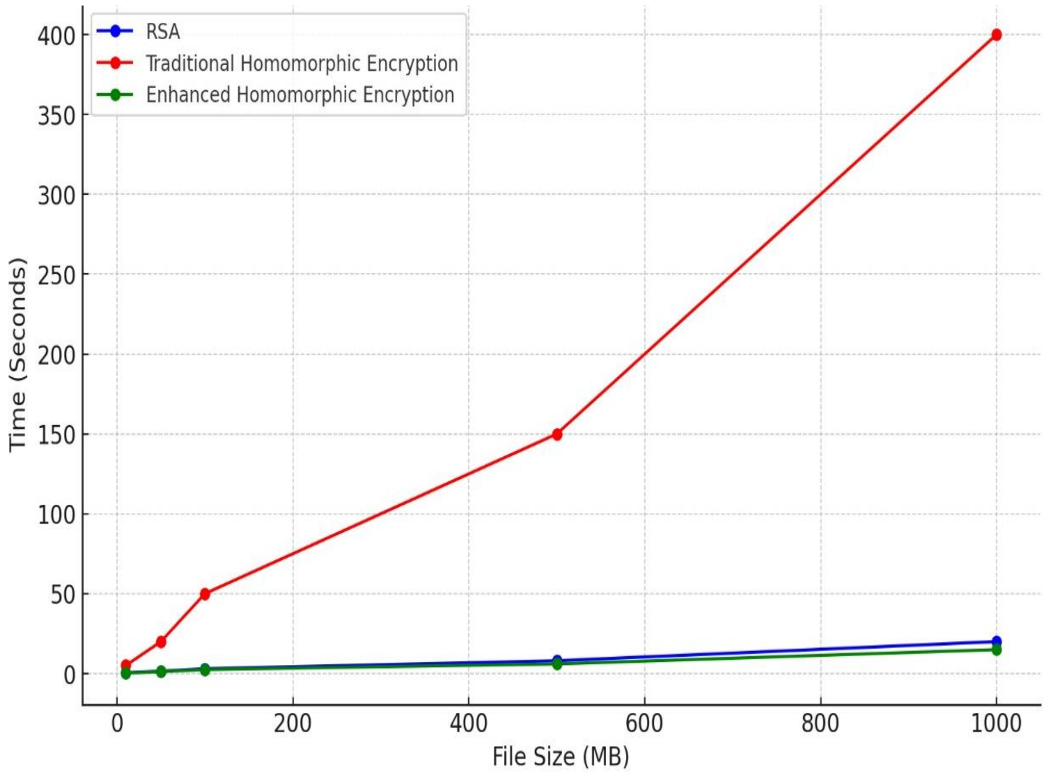

Figure 1: Graph of Time Complexity of RSA, HE and EHE

5. Conclusion

This study successfully developed an Enhanced Homomorphic Encryption (EHE) model aimed at addressing the time complexity issues prevalent in Storage Area Networks (SAN). By integrating advanced techniques such as a camouflage process and a substitution-based key generation method, the EHE model not only enhances data security but also significantly improves encryption and decryption efficiency. The experimental results confirm that EHE outperforms traditional encryption methods, including Rivest-Shamir-Adleman (RSA) and standard Homomorphic Encryption (HE), demonstrating a marked reduction in processing times.

The findings highlight the necessity for innovative cryptographic solutions that can meet the demands of modern data storage environments, particularly as data privacy and security concerns continue to escalate. The EHE model provides a promising framework for secure data handling in cloud computing contexts, where performance and confidentiality are critical.

Future research should aim to explore the scalability of the EHE model in various application scenarios and assess its effectiveness in real-world data security challenges. Additionally, further investigations into hybrid encryption techniques and their potential integration with the EHE model could yield valuable insights into enhancing both security and performance. By continuing to refine and expand upon these methodologies, we can contribute to the development of more efficient and secure data storage solutions that align with the evolving landscape of digital information management.

References

[1] An-Ping, X., Gan, Q., He, X., & Zhao,Q. (2013) A searchable encryption of CP-ABE scheme in cloud storage, 1*0th International Computer Conference on Wavelet Active Media Technology and Information Processing (ICCWAMTIP),* 345-349.

[2] Armbrust, M., Fox, A., Griffith, R., Joseph, A.D., Katz, R., Konwinski, A., Lee, G., Patterson, D., Rabkin, A., Stoica, I., & Zaharia, M. (2010). A view of cloud computing. *Communications of the ACM,* 53(4), 50-58.

[3] Belland, M., Xue, W., Kurdi, M., & Chu, W. (2017). Somewhat Homomorphic Encryption. Retrieved from https://courses.csail.mit.edu/6.857/2017/ project/22 .pdf on the 10th of August 2020.

[4] Biksham, V., & Vasumathi, D. (2017). Homomorphic Encryption Techniques for securing Data in Cloud Computing: A Survey. *International Journal of Computer Applications,* 160 (6), 1 - 6 .

[5] Carroll, M., Van der, M., Alta, D., & Kotze, P. (2011). Secure cloud computing: Benefits, risks and controls. 1-9, 10.1109/ISSA.2011.6027519.

[6] Cheon, J. H., Kim, A., Kim, M., and Song, Y. (2017). Homomorphic Encryption for Arithmetic of Approximate Numbers. *In Advances in Cryptology – ASIACRYPT 2017, (Lecture Notes in Computer Science),* 10624, 409–437.

[7[Daniya, R., Deepak, P., & Kamal V. (2016). Homomorphic Hybrid Encryption for Cloud Computing, *International Journal of Advanced Research in Computer and Communication Engineering,* 5 (5), 502 -511.

[8] Jian, L., Danjie, S., Chen, S., & Lu, X. (2012). A Simple Fully Homomorphic Encryption Scheme Available in Cloud Computing, *IEEE 2nd International Conference,* 01, 214 - 217.

[9] Johnson, T., Smith, L. & Davis, M. (2015). Modern approaches to storage networking. *Journal of Network Infrastructure,* 15(4), 120-135.

[10] Joshi, R., & Renuka, k. (2015). Network Security with Cryptography, *International Journal of Computer Science and Mobile Computing,* 4(1), 201-204.

[11] Kranti, M., Chaudhari, P., & Megha, V. (2015). A Survey on Various Cryptographic Algorithms for Security Enhancement, *International Journal of Computer Applications,* 110(7), 112-119.

[12] Parsi, K. (2012). Data Security in Cloud Computing using RSA Algorithm, *International Journal of Research in Computer and Communication technology,* 1(4), 1112-1119.

[13] Sadasivarao, K., Kumar, S. & Reddy, P. (2003). Defining storage networks in modern systems. *Network Technology,* 8(2), 55-70.

[14] Smart, N., & Vercauteren, F. (2010). Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes. *In: Public Key Cryptography - Proceedings of the 13th International Conference on Practice and Theory in Public Key Cryptography*

[15] Takabi, H., & Joshi, J. (2010). Security and Privacy Challenges In Cloud Computing Environment, Security & Privacy, IEEE, 8(6), 71-79.

[16] Tanenebaum, A.S. & Wetherall, O.J. (2011). *Computer networks* (5th ed.). Pearson Education.

[17] Vahid, A., & Seyed, R. (2012). Security Threats and Countermeasures In Cloud Computing, *International Journal of Application or Innovation in Engineering & Management,* 1 (2), 657-665.

[18] Zhen-Zhong M. (2014). Research of Information Retrieval in the Cloud Computing Environment. *7th International Conference on Intelligent Computation Technology and Automation.* 32-41.