New Fully Homomorphic Encryption Scheme Based On Multistage Partial Homomorphic Encryption Applied In Cloud Computing

Zainab Hikmat Mahmood
Computer Engineering Department
AL Mamoun University College
Baghdad, Iraq
zainabh.mahmood@gmail.com

Mahmood Khalel Ibrahem
College of Information Engineering
Al Nahrain University
Baghdad, Iraq
mahmoodkhalel@coie-nahrain.edu.iq

***Abstract*—Cloud computing is deemed as one of the most powerful innovations in the computing world, However its usage is still hindered by security concerns. Many types of encryption algorithms were applied in the cloud for securing the data. Data can be stored in the cloud in an encrypted form. So, a new technique called Homomorphic encryption is introduced that allows to apply specific operations on the encrypted data.**

The paper presents an overview of security issues in cloud computing and utilization of the fully homomorphic encryption technique has drawbacks of large key size and low calculation efficiency, and it is not practical for the secure cloud computing. We build up a hybrid homomorphic encryption scheme based on the GM encryption algorithm which is additively (single bit) homomorphic, and RSA algorithm which is multiplicative homomorphic. The hybridization of homomorphic encryption schemes seems to be an effective way to defeat their limitations and to benefit from their resistance against the confidentiality attacks. This hybridization of homomorphic encryption algorithm lead to increase the speed (2.9) times, reduce the computation time to 66% percentage from previous one, enhanced confidentiality of the data that is stored in the cloud by enhancing security (two layer of encryption methods used) . Since hybrid encryption is utilized high security and authentication is provided.

Keywords—homomorphic encryption; partially homomorphic encryption; hybrid homomorphic encryption; RSA, Goldwasser-Micali GM.

I. INTRODUCTION

It has been a typical practice for organizations to redistribute their online business logics to Web hosting service providers for over 10 years. Generally, databases and in addition the business logics of an organization are hosted by a third party to save the IT management time and cost. The cloud computing further pushes forward this paradigm. There are many cloud data centers which store a very large amount of information from sources and support data-centric computation. [1] Security can be a major issue for such data centers when the information they have are touchy. A data center may be attacked, compromised and an addition the capability of insider attacks .The security problems with the outsourced databases can be solved if the critical data are encrypted. Naturally it leads to the problem of how the data center can perform computation on encrypted data. [2]

Homomorphic encryption schemes, provide a solution for this stalemate: Such schemes allow for functions to evaluate encrypted data, the result of which is still encrypted data, but can be decrypted back into the result of the (logically) same function applied to the plain data.[3]

II. HOMOMORPHIC ENCRYPTION

Homomorphic encryption systems have the capability of performing computation on encrypted data without knowing the private key. These calculations, create an outcome, which is itself encrypted. The result of any computations on the encrypted data is the same as in the case of raw data [4] , [5].

Mathematically, we said that system is Homomorphic encryption if:
Enc (a) and Enc (b) can calculate Enc (f (a, b)),
where f can be: Add, multiplication, X-or [6]

If the client wants to perform computations on its data in the cloud, the secret key should be shared with the provider to decrypt the data. Sharing the key would allow access to the data from a cloud provider. So to solve this problem, the homomorphic encryption used. The client allows for cloud providers to compute the data without decrypting it. The result will be returned to the client side and it is still encrypted. So,

978-1-5386-9188-5/18/$31.00 ©2018 IEEE
DOI 10.1109/AiCIS.2018.00043

since the client is the only holder of the secret key, no one else is able to decrypt neither data nor results. [7]

The general FOUR process of the Homomorphic Encryption systems is: [4]

1) Generation of Key: the clients generate two keys: secret key K_s and public key K_p.

$$(K_s,K_p)= key\ Gen(s)$$

2) Encryption: **:** an encryption algorithm that takes the public key to encrypt the plain text (*M)* and gives the ciphertext (C).

$$C=Enc_{pk}(m)$$

3) Evaluation: applies a function *f* to a ciphertext *c* using the public key

$$C^*=Eval_{pk}(f,c)\ .$$

4) Decryption: a decryption algorithm that takes the ciphertext *c* and the secret key and recovers the plain text *M.*

$$M=Dec_{sk}(c)$$

A. Categorization of Homomorphic Encryption

We classify the algorithms based on the above mentioned properties into [8]

• Partial Homomorphic Encryption (PHE): Permits only one process on encrypted data either addition or multiplication.

• Somewhat Homomorphic Encryption (SWHE): Permits more than one process –multiplication and addition, but the number of processes is limited

• Fully Homomorphic Encryption (FHE): Permits multiple – multiplication and addition operations without a restriction on the number of operations

B. Properties of Homomorphic Encryption

A homomorphic encryption scheme must check the following properties. [4]

- Additive homomorphism (AH): A homomorphic encryption is additive if,

$$\boldsymbol{Dec_{sk}\ (Enc_{pk}(M1)+ Enc_{pk}(M2)) = M1+M2}$$

- Multiplicative homomorphism (MH): A homomorphic encryption is multiplicative if

$$\boldsymbol{Dec_{k}\ (C_k(M1) * C_k(M2)\ ) = M1*M2}$$

An algorithm is called fully homomorphic if both properties are satisfied simultaneously .

III. HOMOMORPHIC ENCRYPTION TECHNIQUES

Of the various techniques available in homomorphic encryption we shall present at least one of each category. We shall also specify the operation that is permitted in each algorithm. In the next section we compare our findings.

A. RSA – Multiplicative PHE

Rivest, Shamir and Adleman published their public key cryptosystem in 1978 [9] [10] Although it is a very basic algorithm, it is one of the most crucial building blocks of homomorphic encryption, which is why it has been included as an example of multiplicative partial homomorphic encryption technique. RSA algorithm as shown in the figure 1.

1. Generation of Key
Stage 1: Client creates a private/public keys based on choosing two random large primes number P, Q. Calculate N = P*Q, Euler's totient Ø (N) = (P-1)(Q-1).
Stage 2: Choose the encryption key E randomly Where, $1 < E < Ø(N)$ | gcd(E, Ø(N)) = 1.
Stage 3: The two generated pair keys are: the Public key (K_p)= {E,N} and the Private key (K_s)= {D,P,Q}

2. Encryption
Stage 1: To be used K_p= {E, N}.
Stage 2: Calculate cipher text $C = M^E \bmod N$, where $0 \leq M$.

3. Decryption
Stage 1: To be used K_s= {D, P, Q}.
Stage 2: Calculate: M=CD mod N to get back the original plain text message

Figure 1: RSA Cryptosystem Algorithm

Homomorphic property of the RSA.
Assume the two cipher texts, C1, C2.
$C1 = m_1^e \bmod n$
$C2 = m_2^e \bmod n$
$C1 \cdot C2 = (m_1^e \cdot m_2^e) \bmod n$

It is apparent that RSA is a basic algorithm that provides us with limited computation options. This greatly reduces actual practical applications of this algorithm. However, it is a very important algorithm because it acts as a building block and many enhanced algorithms are based on RSA or use it for some part of the implementation. It is also important to note that comparatively RSA is fast and can be feasibly implemented.

B. Goldwasser-Micali System (GM) – Additive PHE

The Goldwasser-Micali (GM) cryptosystem is an asymmetric key encryption algorithm, developed

by Shafi Goldwasser and Silvio Micali in 1982. It is an additive Homomorphic Encryption, but it can encrypt just a single bit. [10] [11]It provides data confidentiality, however, it is not efficient in terms of space complexity because in several cases the cipher text generated is many times larger than the input plain text. GM algorithm as shown in the figure 2.

1. Generation of Key
Stage 1: User generates a public/private key pair by choosing , two large random primes - P, Q. Stage 2: compute N = P.Q. Choose b € Bn | b is a quadratic non-residue modulo n and (b/n) = 1 Stage 3: the key pair is obtained as (pk, sk) Where pk = (n, b) {public key} And sk = (p, q) {private key}

2. Encryption
Input message M is made up of multiple bits m1,m2…m3 Step 1: For every bit mi, a random value yi is generated | gcd(yi,N) = 1 Step 2: compute: Ci = yi2 xmi (mod N)

3. Decryption
Input message C is made up of multiple bits c1,c2…c3 Step 1: calculate: ei = (ci/p), ∀ i € [1, t] Step 2: if ei = 1then mi = 0 else mi = 1 Output message is M = m1,m2,…m3

Figure 2: Goldwasser-Micali Algorithm

Assume we have to encrypt two bits: m1 and m2 using the Goldwasser Micali cryptosystem.
Additive:
EncGM (m1) × EncGM (m2) ≡ bm1×r12).(bm2×r22) mod n
≡ b m1+m2 (r1r2) 2 mod n
≡EncGM(m1⊕m2 , pk)

IV. PROPOSED METHOD

In this part, we will attempt to answer, the possibility to fabricate another encryption scheme which supports all homomorphic tasks from the partial encryption schemes that supports a limited number of homomorphic operations (addition or multiplication). In this paper, we have displayed the appropriate answer on whether hybrid homomorphic encryption schemes be practical. The partial homomorphic encryption schemes can support just only homomorphic property. While, the fully homomorphic encryption schemes can support all properties of homomorphic. To build a new fully homomorphic encryption scheme which supports all homomorphic operations from two partial homomorphic encryption schemes one supports only addition and other supports only multiplication operations. The hybrid homomorphic encryption scheme must preserve the algebraic structure.
In this work, the Hybrid partial encryption process is utilized for encryption the information in the cloud as shown in figure 3. The procedure is as follows:

The Algorithm Of Proposed Method
Step 1: Load original data (plaintext).
Step 2: Convert the data to binary.
Step 3: Select random large integer primes number p, q.
Step 4: Encryption by using GM cryptosystem, produce cipher text C1.
Step 5: Compute Key generation for RSA cryptosystem.
Step 6: Encryption using RSA cryptosystem, produces cipher text C2.
Step 7: Upload the encrypted data to cloud &process it (Add-multiplication)
Step 8: Decryption the cipher text C2, produces cipher text C1.
Step 9: Decryption the cipher text C1, produces original processed data.

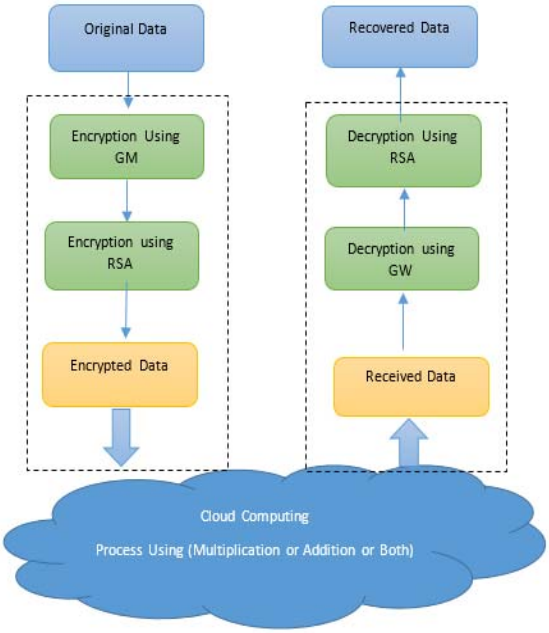

Figure 3 Proposed Method Hybrid Homomorphic Encryption scheme

V. SIMULATIONS AND ANALYSIS

In this implementation, we analyze the Hybrid Homomorphic Encryption performance. By choosing three text files of different size, which are given as inputs to the algorithms to verify the performance of the GM and RSA hybrid homomorphic encryption cryptosystems. The experiment was performed by the machine [are Intel® Core™ i7-5600U CPU @2.60GHz, and RAM is 4.00 GB]. The tables below show three different plain text sizes and the encryption and decryption time, for EHES algorithm which support one operation multiplicative process and the proposed algorithm which support two operations additive –multiplicative process with three times faster.

TABLE I. PLAIN TEXT SIZE, ENCRYPTION TIME FOR PROPOSED ALGORTHIM COMPARE WITH EHES IN REF [12]

Plaintext size in Bits	Encryption Time in second OF EHES in Ref [12]	Encryption Time in second OF Proposed Algorithm
8	0.0724	0.0245
16	0.1093	0.0693
24	0.1308	0.0825

TABLE II. CIPHERTEXT SIZE, DECRYPTION TIME FOR PROPOSED ALGORTHIM COMPARE WITH EHES IN REF [12].

Ciphertext size in Bits	Decryption Time in second Of EHES in Ref[12]	Decryption Time in second OF Proposed Algorithm
8	1.62	0.92
16	4.07	2.03
24	5.67	3.15

From Table I and II, draw two graphs, (Figure4) which represents the time taken by proposed algorithm and EHES to encrypt the different sizes of plain text and (Figure 5) represents the decryption time of different encrypted data.

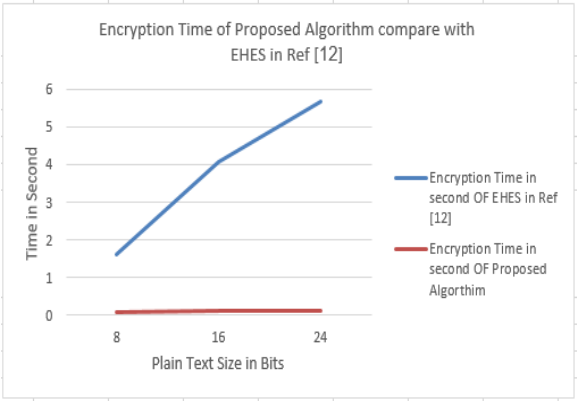

Figure 4: *Plain Text Size versus Encryption Time in Second For proposed Algorithm compare with EHES Algorithm*

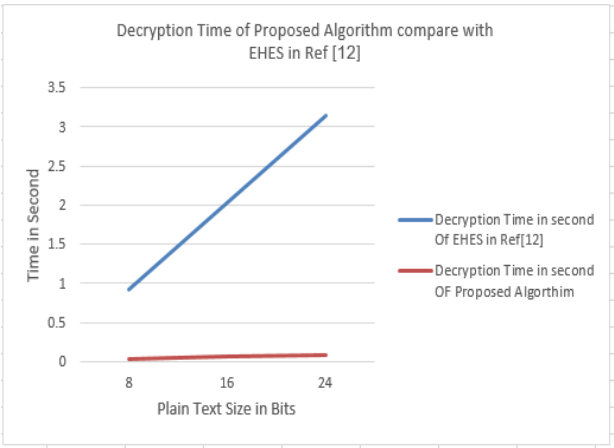

Figure 5 *: Plain Text Size versus Decryption Time in Second For proposed Algorithm compare with EHES Algorithm*

CONCLUSION

When it comes to making a decision about the application of a particular Homomorphic encryption technique, we need to take into several factors such as the operations possible, The overhead involved in terms of time and space complexity and the type data to be encrypted. To combine the advantages observed in two standalone techniques, we have come across research that aims to create a hybrid Homomorphic Encryption algorithm. As the research goes on to explain how GM and RSA techniques are used in tandem to provide enhanced data confidentiality and very secure thereby augmenting the applications of such techniques. To provide a brief idea of this research, the algorithm used is such that plain text data is encrypted with GM at the outset. This encrypted data is treated as the input data to a system which further encrypts it again with RSA. Computations are performed on this data and the decryption process follows the exact same process in the reverse manner. We achieve by combining existing techniques in such a manner is to reduce the computation time while enhancing the security level provided individually is promising area to carry out research.

REFERENCES

[1] Xidan Song , Yulin Wang, "Homomorphic Cloud Computing Scheme Based on Hybrid Homomorphic Encryption," in *IEEE International Conference on Computer and Communication (ICCC)*, Chengdu , China, 2017.

[2] Liangliang Xiao, Osbert Bastani, I-Ling Yen, "An Efficient Homomorphic Encryption Protocol for Multi-User Systems," *IACR Cryptology ePrint Archive,* p. 19, 2012.

[3] Sweta Agrawal , Aakanksha Choubey, "Survey of Fully Homomorphic Encryption and Its Potential to Cloud Computing Security," *International Journal of Advanced Research in Computer Science and Software Engineering,* vol. 4, no. 7, pp. 679 - 686, 2014.

[4] Abdellah EZZATI, Khalid EL MAKKAOUI , Abderrahim BENI HSSANE, "Homomorphic Encryption as a Solution of Trust Issues in Cloud," in *International Conference on Big Data, Cloud and Applications*, Tetuan, Morocco, 2015.

[5] Payal V. Parmar , Shraddha B. Padhar , Shafika N. Patel , Niyatee I. Bhatt , Rutvij H. Jhaveri, "Survey of Various Homomorphic Encryption algorithms and Schemes," *International Journal of Computer Applications,* vol. 91, no. 8, pp. 26 - 32, 2014.

[6] X. Yi et al., "chapter 2," in *Homomorphic Encryption and Application*, SpringerBriefs in Computer Science, 2014, pp. 27-46.

[7] YASMINA BENSITEL, RAHAL ROMADI, "SECURE DATA IN CLOUD COMPUTING USING HOMOMORPHIC ENCRYPTION," *Journal of Theoretical and Applied Information Technology,* vol. 82, no. 2, pp. 206 - 211, 2015.

[8] Monique Ogburna, Claude Turnerb, Pushkar Dahalc, "Homomorphic Encryption," *Procedia Computer Science,* vol. 20, pp. 502 - 509, 2013.

[9] Khalid El Makkaoui ; Abderrahim Beni-Hssane ; Abdellah Ezzati, "Can hybrid Homomorphic Encryption schemes be practical?," in *International Conference on Multimedia Computing and Systems (ICMCS)*, Marrakech, Morocco, 2017.

[10] Dhruva Gaidhani , Joshua Koyeerath , Neel Kudu , Prof. Mahendra Mehra, "A SURVEY REPORT ON TECHNIQUES FOR DATA CONFIDENTIALITY IN CLOUD COMPUTING USING HOMOMORPHIC ENCRYPTION," *International Journal of Advanced Research in Computer Science,* vol. 8, no. 8, pp. 389 - 394, 2017.

[11] Cezar Plesca , Mihai Togan , Cristian Lupascu, "Homomorphic Encryption Based on Group Algebras and Goldwasser-Micali Scheme," in *Innovative Security Solutions for Information Technology and Communications*, Bucharest, Romania, 2016, pp. 149 - 166.

[12] Khalid El Makkaoui ; Abdellah Ezzati ; Abderrahim Beni Hssane, "Challenges of using homomorphic encryption to secure cloud computin," in *International Conference on Cloud Technologies and Applications (CloudTech)*, Marrakech, Morocco, 2015.