Enhanced Fully Homomorphic Encryption Using Integer-based DGHV Scheme

Zainab H. Mahmood [1,*], Mahmood K. Ibrahem [2]
[1] Department of Information and Communications Engineering, College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
[2] Department of Systems Engineering, College of Information Engineering, Al-Nahrain University, Baghdad, Iraq
Email: [1] zainabh.mahmood@gmail.com, [2] mahmoodkhalel@coie-nahrain.edu.iq
*Corresponding Author

***Abstract*—Homomorphic encryption (HE) was introduced to enable untrusted parties to apply computation over encrypted data without decrypting the message. It makes a promising future to solve security challenges with Cloud Computing. One of these challenges, still HE faces complex issues such as large-time computation and huge ciphertext size, limiting its usability in real-world applications. In this paper, we take into account many vital homomorphic encryption schemes, all of them is depend on computing over integers by converting the message to a binary format, then the ciphertext is calculated to every single bit of plaintext. The research contribution suggested a new symmetric Fully Homomorphic Encryption scheme that operates on integer numbers without the need for binary conversion. It encrypts the message using a prime secret key. The proposed system is achieved a great improvement over DGHV system. The scheme significantly reduces ciphertext size and execution time compared to the DGHV system. It's approximately 23.8 times faster than DGHV, and generates single Ciphertext for whole Message: Unlike traditional schemes that create ciphertexts for every bit of the message. The security hardness of proposed scheme is based on the greatest common divisor (GCD) problem, a well-known mathematical problem used in cryptography. We present a detailed explanation of these schemes showing their efficient techniques and compare it with the proposed algorithm in terms of ciphertext size of algorithm, and execution time and analyze their security them finally draw several conclusions.**

Keywords*—*Data Privacy of Cloud Computing, Fully Homomorphic Encryption, DGHV, GCD

I. INTRODUCTION

Homomorphic Encryption (HE) is a fresh subject of cryptographic studies introduced by Rivest, Adleman, and Dertouzos [1] in 1978 to assist people in maintaining their data confidentiality and privacy by enabling untrusted parties to process computations over encrypted information. So, in real-world applications such as cloud-based computing and privacy-preserving search, data aggregation in wireless network sensor scenarios, electronic voting, spam filters, etc. [2]-[7], The HE becomes an essential need. HE may be either symmetrical or asymmetrical [8]-[9]. And similarly, to other encryption schemes, it has three main algorithms: Key Generation, Encryption, and Decryption, in a homomorphic encryption scheme we add an Evaluation algorithm that applies arithmetic operation over the ciphertexts without seeing the original messages [10]-[11].

Homomorphic encryption can be classified into three categories [12]-[14] Partial Homomorphic Encryption (PHE) which supports only one process either addition or multiplication, Somewhat Homomorphic Encryption (SWHE) which supports more than one process multiplication and addition, but the number of processes is limited. Fully Homomorphic Encryption (FHE) supports multiple–multiplication and addition operations without a restriction on the number of operations. The first and most important problem in the whole homomorphic encryption is that the ciphertext format must be conserved to be decrypted successfully after an evaluation phase [15]-[17]. The second crucial problem is to support an infinite number of processes, so the ciphertext size should be constant. Otherwise, increasing the size of the ciphertext will demand more resources and the number of functions will be reduced [18].

The contribution of this paper develops a new fully homographic encryption scheme based on the FHE integer algorithm which is called (Enhancement of DGHV Scheme over Integer), the above special FHE algorithm is selected because the theory is extremely simpler, the key size is lower, smaller ciphertext size and less execution time for comparable performance to other FHE scheme. The remainder of the paper is structured as follows. In Section II: the background information and related work are presented. In Section III: the suggested technique (enhancing the fully homomorphic encryption scheme) Section IV describes the suggested FHE encryption algorithm. Implementation and performance result.

Several homomorphic ideas have been introduced in the literature: An encryption technique is called Fully Homomorphic Encryption FHE if it supports arbitrary computation on ciphertexts. Such a system makes it possible to build programs for any desirable features that can be executed on encrypted inputs to produce a result of encryption. Since such a program never needs to decrypt its inputs, without displaying its inputs and inner state, it can be run by an untrusted party [19]. In 2009 Craig Gentry submitted the first full homomorphic scheme over an ideal lattice [20]. The lattices became more common among cryptography scientists after Gentry's study. After that, some works such as Smart and Vercauteren in 2010 [21] worked merely on enhancing the Gentry ideal lattice-based FHE system [22]. Then, in 2010 Van Dijk-Gentry-Halevi-Vaikuntanathan introduced the FHE scheme over integers based on the Approximate-GCD problems [23]. The conceptual simplicity is the main motivation behind the model. After that, in 2011 Brakerski and Vaikuntanathan suggested another FHE system whose hardness is based on Ring Learning with Error (RLWE) problems, the suggested

scheme promises some characteristics of efficiency. And at last, López-Alt, et al. in 2012 presented an NTRU-like FHE for its promising effectiveness and standardization characteristics. NTRU-Encrypt is one of the earliest attempts based on the Lattice problem. So, fully homomorphic encryption schemes can be categorized under four main FHE families [24]-[25]: (1) Ideal lattice-based by Gentry in 2009, (2) Over integers by Van Dijk et al. in 2010, (3) (R)LWE-based by Brakerski and Vaikuntanathan in 2011, and (4) NTRU-like by López-Alt et al. in 2012, as shown in Fig. 1.

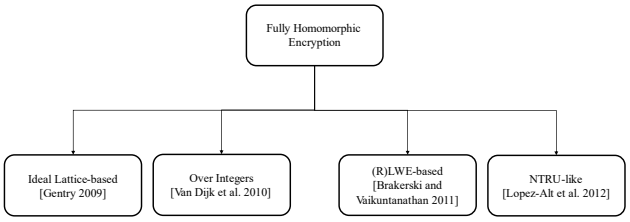

Fig. 1. Main classification of fully homomorphic encryption schemes

A. Description of Dijk-Gentry-Halevi-Vaikuntanathan Scheme Fully Homomorphic Encryption Over the Integers

In 2010, Marten van Dijk, Craig Gentry, Shai Halevi, and Vino Vaikuntanathan introduced a second fully homomorphic encryption scheme called (the DGHV scheme) [26], the main contribution of this scheme is its operations are performed on integers using elementary modular arithmetic computation instead of Euclidean lattices over a polynomial ring. The DGHV scheme description is as follows:

DGHV Fully homomorphic encryption algorithm	
Key generation(λ)	**Step 1:** pick a secrete key p as an odd η bit integer from interval $p \in [2^{1-\eta}, 2^{\eta})$.
Encryption (p, m)	**Step 1:** The plaintext space in this scheme is $m = Z_2$,means convert the message to binary format $m \epsilon \{0,1\}$ **Step 2:** Calculate ciphertext: $c = pq + 2r + m$ for each message bit. **Step 3:** Chose the integers q, r at random in some other prescribed intervals, such that $2r$ is smaller than $p/2$ in absolute value, $r \approx 2^{\sqrt{\eta}}$ and generate $q \approx (2^{\eta})^3$
Decryption (p, c)	$Output = (c \bmod p) \bmod 2$ $= c - p * \lfloor c/p \rceil \bmod 2.$

Where this scheme encrypts each bit from the message and it is considered complex because it is generating some bits to hide one bit.

- **Homomorphic properties**:
  Let $c1 = pq1 + 2r1 + m1$,
  $c2 = pq2 + 2r2 + m2$
- **Additive Homomorphic properties:**
  $c1 + c2 = (q1 + q2)p + 2(r1 + r2)$
  $+ (m1 + m2)$
  $[(c1 + c2) \bmod p] \bmod 2 \equiv m1\ XOR\ m2\ (mod2)$
- **Multiplicative Homomorphic properties:**
  $c1.c2 = (q1q2p + 2q1r2 + q1m2 + 2q2r1 + q2m1)$
  $p + 2(2r1r2 + r1m2 + r2m1) + m1m2$
  $[(c1.c2 \bmod p) \bmod 2] \equiv m1\ AND\ m2\ (mod2)$

B. A Simple Fully Homomorphic Encryption Scheme

A Simple Fully Homomorphic Encryption Scheme (SDC Scheme) was presented by Jian Li, Danjie Song, Sicong Chen and Xiaofeng Lu in 2012 [27]-[28]. the scheme derived from the Gentry cryptosystem to ensure privacy in cloud storage. The description of Simple Fully Homomorphic Encryption scheme is as follows:

A Simple Fully Homomorphic Encryption Scheme	
Key generation(p)	**Step 1:** Generate a prime odd bit integer p as private key Step 2: Generate q which is a random number
Encryption (p, m)	**Step 1:** Convert the plain text to binary format $m \epsilon \{0,1\}$ **Step 2:** Calculate ciphertext $c = m + p + r * p * q,$ where: r: is a random number of $R - bit$
Decryption (p, c)	$m = (c \bmod p)$
Retrieval(c)	$R = (ci - c\ index) \bmod q$

At the client-side, the client encrypts the Keywords m index, so $c\ index = m\ index + p + r * p * q$,then send it to the server. After receiving the c index, the server reads the ciphertexts and computes: $R = (ci - c\ index) \bmod q,$ once $R = 0,$ that is mean the ciphertext retrieval succeeds, and ci is the desired result. Suppose $c1 = m1 + p + r1 * p * q, c2 = m2 + p + r2 * p * q.$

- **Additive Homomorphism:**
  $c3 = c1 + c2 = (m1 + m2) + (r1 + r2)$
  $* p * q + 2p.$
  $m3 = c3 \bmod p = m1 + m2.$
- **Multiplicative Homomorphism:**
  $c4 = c1 * c2 = m1 * m2 + (m1 + m2 + p)$
  $p + r1\,(p + m + r2)\,p\,q + r2.$

C. Nuida-Kurosawa FHE Scheme

Nuida and Kurosawa proposed in 2015 [29], an FHE scheme over the integers with message space ZQ where Q is a prime [30]. The degree of the decryption circuit is smaller than the previous scheme and the scheme is more efficient. This is a conversion of the Boolean circuit and arithmetic circuit. We mainly study the symmetric version of the Nuida-Kurosawa FHE scheme and describe it as follows.

Nuida-Kurosawa FHE Scheme	
Key generation(λ)	**Step 1:** Select an odd number $p \epsilon [2^{\lambda-1}, 2^{\lambda})$ **Step 2:** Set p as the secret key.
Encryption (p, m)	**Step 1:** Given a message m $\in \mathbb{Z}Q$ **Step 2:** Calculate the ciphertext as: c = pq + Qr + m; **Step 3:** Where the integers q, r are chosen at random in some other prescribed intervals, such that $Qr < \lfloor p/2 \rfloor$.
Decryption (p, c)	m = (c mod p) mod Q.

- **Additive homomorphic property:**
  $(c1 + c2) mod\ p\ mod\ Q = m1 + m2$
- **Multiplicative homomorphic property:**
  $(c1 \cdot c2)\ mod\ p\ mod\ Q = m1 \cdot m2$

D. Proposed Algorithm

After looking in [DGHV, SDC and Nuida-Kurosawa] fully homomorphic encryption schemes, we take note that the message previous encrypted as one bit, as either *0 or 1*, and get a huge cipher text size for one character. where the plaintext characters converted to binary format (8-bit) then encrypts each bit to perform one ciphertext for each bit that is mean eight ciphertext for each character in the plaintext, Our proposed encryption scheme encrypts the integer number as whole character using: $c = m + 2r\,p + p.q$ without converting it to binary format and gets one ciphertext instead of 8-ciphertext.We modified on DGHV encryption equation and there is other method also try to encrypt the plaintext without converting it to binary format [31].

E. Algorithm of Proposed Algorithm (Symmetric Encryption)

a) Generation of Key
Step 1: Client generates a prime big integer p as private key **Step 2:** Chose $q \in Zn$, big integer number. **Step 3:** Chose $r \in Zn$, small integer number.
b) Encryption
Step 1: A Input message m without converting it to binary forma, where $m \in [0, p-1]$ **Step 2:** Calculate the encryption of plaintext $C = m + 2\,r.p + p.q$
c) Decryption
Step 1: Input message C (one cipher text for one character) **Step 2:** Decrypt the cipher text C using the same privet key p $m = C\ mod\ p$ where $m < p$
c) Evaluation
Let C_1 the ciphertext for m_1 and C_2 the ciphertext for m_2 $C1 = m1 + 2\,r1\,p + p.q$ $C2 = m2 + 2r2\ p + p.q$

- **Additive homomorphic property: A homomorphic encryption support addition process if,**
  $Dec[\,sk\,, (Enc\ sk\ (m1) + Enc\ sk\ (m2))]$
  $= m1 + m2$
- **Multiplicative homomorphic property: A homomorphic encryption support multiplication if**
  $Dec[\,sk\,, (C\ sk\ (m1) * C\ sk\ (m2))] = m1 * m2$

If both properties are satisfied simultaneously then the algorithm is called fully homomorphic. The proposed algorithm scheme can be described as follows:

- KeyGen (λ):
  Chose a big prime integer randomly and set it as a secret key p.
- Encryption phase (Sk, m):
  The message $m \in [0, p-1]$ encrypted by *private key p using* $C = m + 2\,r\,p + p.q$

Where the noise r is a random integer (adding value of noise r is to hide the private key p) and choosing q as a constantan big integer (represent duplicated number of P).

- Evaluate phase $(Sk, m1 \ldots mn, c1, \ldots, cn)$: after applying the process either addition or multiplication to n ciphertexts ci, then the decryption of the resultant ciphertext ci, lead to the same result if the addition and multiplication process applied to n input mi .

Decrypt phase (Sk, C): the ciphertext decrypted C using the private key P using: $C : m = c\ mod\ p$
The mathematical proof that the proposed algorithm scheme supports both additive and multiplicative homomorphism properties: Suppose, $c1 = m1 + 2\,r1\,p + p.q$, $c2 = m2 + 2\,r2\,p + p.q$

- **Additive Homomorphism:**
  $c3 = c1 + c2 = (m1 + m2) + (r1 + r2)\ p + 2\,p.q$
  $m3 = (c1 + c2)\ mod\ p = m1 + m2$
- **Multiplicative Homomorphism:**
  $c4 = c1.c2 = m1.m2 + (m1 + m2 + p.q)p.q + r1\,(m2 + r2 + p.q)\ p + r2\,(m1 + p.q)p.m4 = (c1.c2)\ mod\ p = m1.m2.$

F. The Proposed Method Example

a) Generation of Key	
➢ Let a prime secret key p =1307645657. ➢ Chose q ∈ Zn, q =4259427989. ➢ Select two integers $r1 = 12328989585$ and $r2 = 2236953862$, randomly. ➢ Let the messages $m1 = 70$ and $m2 = 45$	
b) Encryption	
Now calculate ciphertext c1 for first message $m1\ c1 = m1 + r1\,p + p.q$ $c1 = 70 + 12328989585 * 1307645657 + 1307645657 * 4259427989$ $c1 = 21691772197143576188.$ Then calculate ciphertext c2 for second message $m2$: $c2 = m2 + r2\,p + p.q$ $c2 = 45 + 2236953862 * 1307645657 + 1307645657 * 4259427989$ $c2 = 8494965513673771152$	
c) Evaluation	
Additive Homomorphic Encryption Scheme Let c3 represent the addition of two encrypted messages: $c3 = c1 + c2$ = 21691772197143576188 +8494965513673771152 = 30186737710817347340	**Multiplication Homomorphic Encryption Scheme** Let $c4$ represent the multiplication of two encrypted messages: $c4 = c1.c2$ = 21691772197143576188* 8494965513673771152 = 184270856745202207168739002881588528576
e) Decryption	
Now decrypt c3: $m3 = c3\ mod\ p\ m3$ =30186737710817347340 mod 1307645657 m3=115 This is equal to $m1 + m2$ (i.e., 70 + 45=115)	Now decrypt $c4$: $m4 = c4\ mod\ p\ m4$ =184270856745202207168739002881588528576 mod 1207645633 $m4$= 3150, this is equal to $m1 * m2$ (*i.e.*, 70 *454 = 3150).

II. Results and Discussion

In this section, the proposed algorithm compared with the rest schemes in terms of security, execution time and ciphertext size.

A. Execution Time

As shown in Table 1, the result reveals that the execution time of the proposed scheme is less than the time required for DGHV, SDC, and SAM schemes for the same parameters p, q, r, and the same size of the message. By comparison with

reference [8], the results show that the proposed method is much faster than the previous methods and the execution time of the proposed scheme is very fast. Fig. 2 displays the execution time for the proposed method and Fig. 3 display the comparison between the fully homomorphic encryption schemes in term of execution time. The proposed algorithm ciphertext size for addition vs multiplication process shown in Fig. 4.

Table 1. Comparison between the proposed algorithm & REF. [8] in term of execution time

Length of the message	Execution Time in (ms)			
	The proposed algorithm	DGHV-FHE algorithm	*SDC-FHE* algorithm	SAMFHE algorithm
12 bytes	4.69	1118	1180	1007
1.4 Kbyte	20.93	124182	1283068	20818
2.8 Kbyte	27.003	6715148	4425899	72901

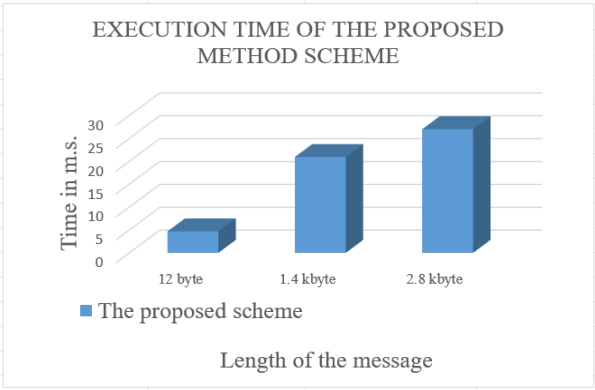

Fig. 2. The proposed method execution time

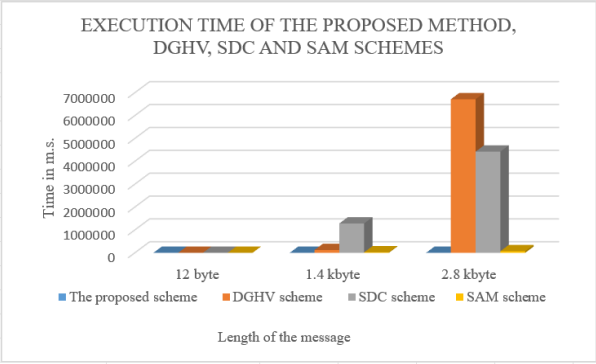

Fig. 3. Comparison between the proposed method, and other schemes in term of execution time

B. *Ciphertext size*

The main contribution of this proposed system is a reduction in ciphertext size it encrypts the whole message as an integer number and generates one ciphertext, instead of converting the plaintext to binary format and encrypts every bit individually to produce eight ciphertexts to every byte in plaintext. The proposed algorithm ciphertext size in Table 2.

Table 2. The proposed algorithm ciphertext size

Length of the message (plaintext)	The generated ciphertext	
	Addition process	Multiplication process
12 Byte	14 bytes	22 bytes
1.4 K byte	1.64 Kbyte	2.57 Kbyte
2.8 Kbyte	3.28 K byte	5.15 Kbyte

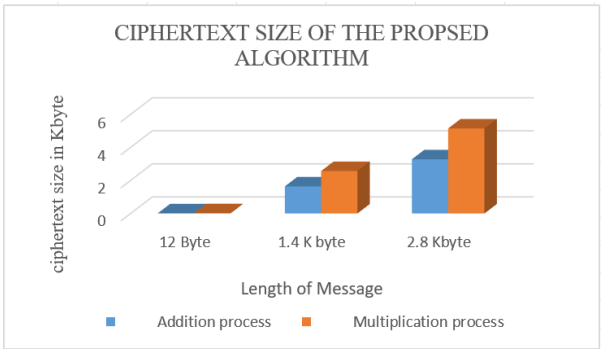

Fig. 4. The proposed algorithm ciphertext size for addition vs multiplication process

C. *Security*

As mentioned in the literature review section and Fig.1 the security strength of the DGHV scheme is based on the (approximate GCD) problem, The robustness in the security of the proposed algorithm depends on the selection of a large prime number as a secret key compared to the selection of any number by selecting the appropriate parameters for the scheme, and it is proven to be resistant to brute force attacks with at least 2λ time and other types of attacks. This is explained by the fact that the prime number itself costs the third party, where it is first important to check that the number is prime, and then the prime number should be checked on the encryption equation, which requires several attempts to break the code. Also, the prime number provides a unique probability of decrypting the ciphertext.

III. CONCLUSION

Information security is undoubtedly the biggest concern in the world. And yet one of the solutions is homomorphic encryption that provides data processing privacy and security in unreliable environments such as grid computing and cloud. Van Dijk fully homomorphic encryption scheme defines a simpler scheme than the Gentry scheme, but both of them are limited to encrypting every bit of the message individually. This work extends the DGHV scheme to enable operations with integers of arbitrary size of plaintext without transforming them into bits.it is a symmetric scheme with a very big odd integer private key. The evaluation of the proposed algorithm is based on performance in terms of execution time and memory requirements. Which is less execution time needed and less ciphertext size. The proposed algorithm presents a global performance improvement which is 23.8 times faster than the DGHV scheme.

REFERENCES

[1] M. L. Gaid and S. A. Salloum, "Homomorphic encryption," *The International Conference on Artificial Intelligence and Computer Vision*, vol. 1377, pp. 634-642, 2021, https://doi.org/10.1007/978-3-030-76346-6_56.

[2] S. L. Nita and M. I. Mihailescu, "Homomorphic Encryption," *Advances to Homomorphic and Searchable Encryption*, pp. 27-88, 2023, https://doi.org/10.1007/978-3-031-43214-9_3.

[3] W. Yang, S. Wang, H. Cui, Z. Tang, and Y. Li, "A Review of Homomorphic Encryption for Privacy-Preserving Biometrics," *Sensors*, vol. 23, no. 7, p. 3566, 2023, https://doi.org/10.3390/s23073566.

[4] A. Ali, B. A. S. Al-Rimy, F. S. Alsubaei, A. A. Almazroi, and A. A. Almazroi, "HealthLock: Blockchain-Based Privacy Preservation Using Homomorphic Encryption in Internet of Things Healthcare

Applications," *Sensors*, vol. 23, no. 15, p. 6762, 2023, https://doi.org/10.3390/s23156762.

[5] Y. Ameur and S. Bouzefrane, "Handling security issues by using homomorphic encryption in multi-cloud environment," *Procedia Computer Science*, vol. 220, pp. 390-397, 2023, https://doi.org/10.1016/j.procs.2023.03.050.

[6] C. Aguilar-Melchor, S. Fau, C. Fontaine, G. Gogniat and R. Sirdey, "Recent Advances in Homomorphic Encryption: A Possible Future for Signal Processing in the Encrypted Domain," *IEEE Signal Processing Magazine*, vol. 30, no. 2, pp. 108-117, 2013, https://doi.org/10.1109/MSP.2012.2230219.

[7] R. Podschwadt, D. Takabi, P. Hu, M. H. Rafiei and Z. Cai, "A Survey of Deep Learning Architectures for Privacy-Preserving Machine Learning with Fully Homomorphic Encryption," *IEEE Access*, vol. 10, pp. 117477-117500, 2022, https://doi.org/10.1109/ACCESS.2022.3219049.

[8] B. Seth, S. Dalal, and R. Kumar, "Hybrid Homomorphic Encryption Scheme for Secure Cloud Data Storage," *Recent Advances in Computational Intelligence*, vol. 823, pp. 71-92, 2019, https://doi.org/10.1007/978-3-030-12500-4_5.

[9] F. Thabit, O. Can, S. Alhomdy, G. H. Al-Gaphari, and S. Jagtap, "A Novel Effective Lightweight Homomorphic Cryptographic Algorithm for data security in cloud computing," *International Journal of Intelligent Networks*, vol. 3, pp. 16-30, 2022, https://doi.org/10.1016/j.ijin.2022.04.001.

[10] I. Amorim and I. Costa, "Leveraging Searchable Encryption through Homomorphic Encryption: A Comprehensive Analysis," *Mathematics*, vol. 11, no. 13, p. 2948, 2023, https://doi.org/10.3390/math11132948.

[11] A. V. Kumar, K. Bhavana and C. Yamini, "Fully Homomorphic Encryption for Data Security Over Cloud," *2022 6th International Conference on Electronics, Communication and Aerospace Technology*, pp. 782-787, 2022, https://doi.org/10.1109/ICECA55336.2022.10009404.

[12] S. Nithya, V. Seethalakshmi, G. Vetrichelvi, M. Siva Sangari, and G. Basavaraj, "A Survey on Private Keyword Sorting and Searching Homomorphic Encryption," *Homomorphic Encryption for Financial Cryptography*, pp. 259-275, 2023, https://doi.org/10.1007/978-3-031-35535-6_12.

[13] P. Chaudhary, R. Gupta, A. Singh and P. Majumder, "Analysis and Comparison of Various Fully Homomorphic Encryption Techniques," *2019 International Conference on Computing, Power and Communication Technologies (GUCON)*, pp. 58-62, 2019, https://ieeexplore.ieee.org/abstract/document/8940577.

[14] Y. Shoukry *et al*., "Privacy-aware quadratic optimization using partially homomorphic encryption," *2016 IEEE 55th Conference on Decision and Control (CDC)*, pp. 5053-5058, 2016, https://doi.org/10.1109/CDC.2016.7799042.

[15] N. Naqvi, A.T. Abbasi, R. Hussain, et al., "Multilayer Partially Homomorphic Encryption Text Steganography (MLPHE-TS): A Zero Steganography Approach," *Wireless Personal Communications*, vol. 103, pp. 1563-1585, 2018, https://doi.org/10.1007/s11277-018-5868-1.

[16] X. Sun, P. Zhang, J. K. Liu, J. Yu and W. Xie, "Private Machine Learning Classification Based on Fully Homomorphic Encryption," in *IEEE Transactions on Emerging Topics in Computing*, vol. 8, no. 2, pp. 352-364, 2020, https://doi.org/10.1109/TETC.2018.2794611.

[17] A. A. Ahmed, M. M. Madboly, and S. K. Guirguis, "Securing Data Transmission and Privacy Preserving Using Fully Homomorphic Encryption," *International Journal of Intelligent Engineering & Systems*, vol. 16, no. 1, pp. 277-289, 2023, https://doi.org/10.22266/ijies2023.0228.25.

[18] X. Sun, F. R. Yu, P. Zhang, W. Xie, and X. Peng, "A survey on secure computation based on homomorphic encryption in vehicular ad hoc networks," *Sensors*, vol. 20, no. 15, p. 4253, 2020, https://doi.org/10.3390/s20154253.

[19] K. Munjal and R. Bhatia, "A systematic review of homomorphic encryption and its contributions in the healthcare industry," *Complex Intelligent Systems*, vol. 9, pp. 3759-3786, 2023, https://doi.org/10.1007/s40747-022-00756-z.

[20] R. Huang, Z. Li, and J. Zhao, "A Verifiable Fully Homomorphic Encryption Scheme," *Security, Privacy, and Anonymity in Computation, Communication, and Storage*, vol. 11611, pp. 412-426, 2019, https://doi.org/10.1007/978-3-030-24907-6_31.

[21] C. Gentry, "Fully homomorphic encryption using ideal lattices," *Proceedings of the forty-first annual ACM symposium on Theory of computing*, pp. 169-178, 2009, https://doi.org/10.1145/1536414.1536440.

[22] N. P. Smart and F. Vercauteren, "Fully Homomorphic Encryption with Relatively Small Key and Ciphertext Sizes," *Public Key Cryptography – PKC 2010*, vol. 6056, pp. 420-443, 2010, https://doi.org/10.1007/978-3-642-13013-7_25.

[23] P. -C. Chen, T. -H. Kuo and J. -L. Wu, "A Study of the Applicability of Ideal Lattice-Based Fully Homomorphic Encryption Scheme to Ethereum Blockchain," *IEEE Systems Journal*, vol. 15, no. 2, pp. 1528-1539, 2021, https://doi.org/10.1109/JSYST.2021.3064053.

[24] C. Bonte, I. Iliashenko, J. Park, H. V. Pereira, N. P. Smart, "Final: Faster fhe instantiated with ntru and lwe," *International Conference on the Theory and Application of Cryptology and Information Security*, vol. 13792, pp. 188-215, 2022, https://doi.org/10.1007/978-3-031-22966-4_7.

[25] J. H. Cheon and D. Stehlé, "Fully Homomorphic Encryption over the Integers Revisited," *Advances in Cryptology -- EUROCRYPT 2015*, vol. 9056, p. 513, 536, 2015, https://doi.org/10.1007/978-3-662-46800-5_20.

[26] C. Gentry, A. Sahai, and B. Waters, "Homomorphic Encryption from Learning with Errors: Conceptually-Simpler, Asymptotically-Faster, Attribute-Based," *Advances in Cryptology – CRYPTO 2013*, vol. 8042, 2013, pp. 75-92, https://doi.org/10.1007/978-3-642-40041-4_5.

[27] K. Kluczniak, "NTRU-v-um: Secure Fully Homomorphic Encryption from NTRU with Small Modulus," *Proceedings of the 2022 ACM SIGSAC Conference on Computer and Communications Security*, pp. 1783-1797, 2022, https://doi.org/10.1145/3548606.3560700.

[28] L. Dong, S. Chen, Y. Cheng, Z. Wu, C. Li, H. Wu, "Measuring economic activity in China with mobile big data," *EPJ Data Science*, vol. 6, no. 29, pp. 1-17, 2017, https://doi.org/10.1140/epjds/s13688-017-0125-5.

[29] Y. Li, "Review on Nuida-Kurosawa Fully Homomorphic Encryption in Client-server Computing Scenario," *International Journal of Electronics and Information Engineering*, vol. 11, no. 1, pp. 16-24, 2019, http://ijeie.jalaxy.com.tw/contents/ijeie-v11-n1/ijeie-2019-v11-n1-p16-24.pdf.

[30] J. Zhao, R. Huang, and B. Yang, "Efficient GSW-Style Fully Homomorphic Encryption over the Integers," *Security and Communication Networks*, vol. 2021, pp. 1-13, https://doi.org/10.1155/2021/8823787.

[31] Z. H. Mahmood and M. K. Ibrahem, "Hardware Implementation of an Encryption for Enhancement DGHV," *Iraqi Journal of Information and Communication Technology*, vol. 2, no. 2, pp. 44-57, 2019, https://doi.org/10.31987/ijict.2.2.69.