Bol. Soc. Paran. Mat.
©SPM – E-ISSN-2175-1188
SPM: www.spm.uem.br/bspm

(3s.) **v. 2026 (44) 11** : 1–8.
ISSN-0037-8712
doi:10.5269/bspm.81209

# Improving the Efficiency of Partial Homomorphic Encryption RSA and Developing it into a Hybrid System

Nibras Hadi Jawad

ABSTRACT: Previous research has carried out several investigations into the mechanics of encryption and the approaches that might speed up computations. It became necessary to develop these algorithms and leverage other technologies to achieve greater efficiency and performance in encryption systems by improving RSA encryption technology. RSA is a type of partial homogeneous encryption mechanism that enables and accelerates multiplication on encrypted data that contains large numbers and requires significant computational effort. Therefore, the system was improved to be more efficient by using the Extended Residual System (RNSBE), with a time reduction of up to 75% compared to the previous system. A hybrid RSA-Pa system was also introduced, combining the characteristics of the RSA algorithm and the Paillier algorithm, resulting in a strong and robust hybrid.

Keywords: Partial homomorphic encryption, Homomorphic RSA, RNS, base extension, Paillier.

## Contents



## 1. Introduction

Currently, Due to the swift increase in internet usage and the necessity to access resources provided by cloud networks, which facilitates many processes and procedures for users and helps save time and cost, cyber concerns about maintaining privacy and the importance of data security and preventing unauthorized access have increased. It has become necessary to provide solutions to the privacy problem. One proposed solution is the fast public key RSA (Rivest-Shamir-Adleman) algorithm [1]. This well-known algorithm lies in its strength in encrypting data on the difficulty of calculating the discrete logarithm, especially for large numbers. This provides security and protection for encrypted data. RSA is a type of partial homomorphic encryption algorithm, which supports only multiplication operations. Homomorphic RSA allows operations to be performed on encrypted data without the need to decrypt it, enhancing data privacy in insecure environments. Such technology can be used to classify health systems [2,3], and in electronic voting [4], providing users with greater confidence and security by protecting their privacy. Despite the advantages offered by homomorphic RSA [5,6], the issue of computational cost remains due to its handling of large exponents for security. This poses a barrier to its use. One effective method for reducing the computational cost of large numbers and increasing their speed, while maintaining efficiency, is the use of parallel calculations. This approach divides a number into several smaller components

2020 *Mathematics Subject Classification:* 68P25, 68P27.
Submitted January 05, 2026. Published March 13, 2026

based on a specific base and calculates all parts simultaneously, significantly enhancing speed. One of these systems is the Base Extensions Residue Number System (BERNS) [7,8,9], which is based on two specific bases and utilizes a collection of lesser residues to represent a large integer in relation to a set of pairwise co-prime moduli. Hybrid systems are efficient and effective systems that rely on combining the characteristics of different systems to generate a more efficient system. For example, the RSA system is a robust system capable of repelling attacks, while the Paillier system is complex. These two algorithms are similar in characteristics and of a homogeneous type. Therefore, combining such systems results in significantly higher efficiency and superior performance [10].

Researchers in this field have conducted numerous experiments and studies to contribute to improving the efficiency of the algorithm, as suggested by A.E. Mesioye et al. [11] A modified CRT (MCRT-RSA) is created to speed up homomorphic RSA decryption, where the developed decryption algorithms are for traditional RSA, CRT-RSA, and MCRT-RSA. MCRT-RSA was faster than RSA and CRT-RSA but used more memory.

The Researchers K. El Makkaoui, and et al [12], proposed Cloud-RSA modulus and exponents are modified to create variants. The first variant encrypts and decrypts using the Chinese remainder theorem (CRT) with a modulus of two or more primes. In contrast, the second variation employs a modulus of n=$p^r$ $q^s$ for $r \geq 2$ and $s \geq 1$, decrypting using Hensel lifting and CRT. H. Touil, and et al [13], in this work, was construct a crypto-system that can calculate encrypted data, notably integers, using two encryption algorithms: additive homomorphic using paillier method and multiplicative homomorphic using RSA. This ensures cloud calculation security. While R. Abid, and et al. [14], who used the Chinese Remainder Theorem with a homomorphic RSA (HE-CRT-RSA) algorithm to solve the speed. Multiple keys are used for efficient communication and security. With the proposed approach, performance improves with reduced decryption time. The suggested HE-CRT-RSA is 3–4% faster than Rivest-Shamir-Adleman. P. Krishnadoss, and et al [15], was proposed method optimizes speed by storing encrypted repeated element values. Many experiments were run on datasets with paragraphs of varied sizes to test this strategy. All paragraph sizes show that the proposed dynamic RSA technique beats classic RSA in encryption, decryption, and execution time. C. Gilbert, and et al [16], a geometric encryption method is proposed to improve multiparty cloud compute efficiency and security with used RSA and paillier. According to experimental results, optimized homomorphic algorithm is more suited to big data, healthcare, and finance because they reduce computational complexity and ciphertext sizes. N. H. Jawad, & S. Abdulhadi [17], Using the Hybrid-Position-Residues number system (HPR) that is based on the Base Extensions Residue Number System, the authors are able to lower the cost of calculations for the Brakerski-Fan-Vercauteren (BFV) algorithm while preserving the efficiency of the method.

The following is the structure of the subsequent sections that will be included in this paper: Within the second section, implementing the proposed HRSA-BR system with an overview of some fundamentals. Additionally, the third section Proposing a Hybrid System that Combines RSA and Paillier Algorithms (RSA-Pa). In the fourth section, the examining results. The conclusion is presented in the fifth section of the work.

## 2. Implementing BERNS in Homomorphic RSA (HRSA-BR) Proposed System

The fundamental concept of this research is to encrypt data with homomorphic RSA by employing a technique that is founded on the notion of homomorphism in order to process the data without having to resort to decryption. Due to the fact that RSA security is dependent on the utilization of a large modulo (N) value, the execution of these operations necessitates a substantial amount of time and processing resources. For this reason, the proposed system makes use of an expanded residual numbers system in order to solve the problem of time-consuming procedures. Take a look at Fig. 1.

Fig. 1 illustrates the general working mechanism. The user encrypts and decrypts the data according to the proposed HRSA-BR or other system, and the other party does not need to use any key. After the user encrypts the data, they send the encrypted data to the server for processing and operation, then return it to the user, who then decrypts the encrypted result to obtain the final, unencrypted result.

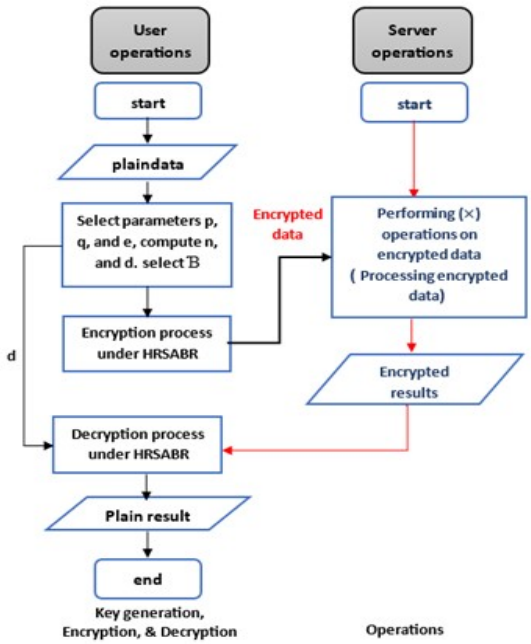


Figure 1: General Proposed System

### 2.1. Homomorphic RSA Algorithm

Homomorphic RSA is a partial symmetric encryption algorithm that supports multiplication only. Its strength problem stems from the difficulty of prime factorizing large integers, which requires significant computational time and effort. Algorithm 1 explains its operation [18]. Homomorphic RSA uses asymmetric keys (public key and private key) for encryption. Unlike standard RSA, the data owner performs both encryption and decryption. The server only processes the data and does not handle explicit data. Algorithm 1 explains its working steps.

**Algorithm 1** Homomorphic RSA [1,5]

**Input:** prim numbers (p, q), m (message), e
**Compute:** N=p*q, $(\phi(N))$, d=$(e)^{-1}$ $mod$ $(\phi(N))$
**Public key:** (N, e)
**Privet key:** $((\phi(N))$, d)
**Output:** $E(m)$, m

**Begin**
Encrypting $m$ by
$E(m) = m^e \mod N$
Return Result $c' = E(m)$
Decryption $c'$
$m = (c')^d \mod N$
Return Result $m$
**End**

Table 1: The Notations List.

| Symbol | Definition |
|---|---|
| $\phi(N)$ | Euler of N =(p-1) *(q-1) |
| LMC | Least Common Multiple |
| $\lambda(N)$ | $LMC(\phi(N))$ |
| $\prod_{i=1}^{t} \mu_i$ | The product of pairwise co-prime modules, that length t. |
| B | Set of pairwise co-prime moduli (bases) |
| $<m>_B$ | The message is represented by RNS with one set. |
| $<m>_a \vert b$ | The message is represented by BERNS, by 2 sets of bases (a is concatenated with b) |
| $c'$ | Ciphertext |
| $E(m)$ | Encryption message |
| $D(m)$ | Decryption message |
| $[x]_a$ | x mod a |

### 2.2. Base Extensions Residue Number System (BERNS)

RNS is a set of pairwise co-prime moduli that divides huge large number into a set of infinitesimal numbers based on certain bases $B = \{\mu_1, \mu_2, \ldots, \mu_t\}$, where $t$ is the number of bases and $\gcd(\mu_i, \mu_j) = 1$. Must be $m < BP$, and $BP = \prod_{i=1}^{t} \mu_i$, and $m_1 \cdot m_2 < BP$. Represent $m$ in RNS as $\langle m \rangle_B = \langle m_1, \ldots, m_t \rangle_B$ where $m_i \equiv m \pmod{\mu_i}$. BERNS [19] used two sets of bases $B_a$ and $B_b$, where $B_b$ is an extension of $B_a$, $\langle B \rangle_{a|b} = \langle \mu_{a1}, \ldots, \mu_{at_a}, \ \mu_{b1}, \ldots, \mu_{bt_b} \rangle_{a|b}$, where $a|b$ concatenation and $B_a > B_b$. $m$ represented as $\langle m \rangle_{a|b} = \langle m_{a1}, \ldots, m_{at_a}, \ m_{b1}, \ldots, m_{bt_b} \rangle_{a|b}$. Extract $m$ from the BERNS formula based on Chinese Remainder Theorem (CRT), represented as $m = \left( \sum_{i=1}^{n_a} ([\, m_{a,i}\, B_{a,i}^{-1}\, ]_{m_{a,i}})\, B_{a,i} \right) - r(m)\, B_a$, where $r(m) = \left\lfloor \sum_{i=1}^{t_a} \frac{[\, m_{a,i}\, B_{a,i}^{-1}\, ]_{\mu_{a,i}}}{\mu_{a,i}} \right\rfloor$, where it is always $r(m) < t$. See BE example in [5] and for further understanding, read the research. For a better understanding of the symbols used in the research paper, see Table 1.

### 2.3. Homomorphic RSA with BERNS (HRSA-BR) Proposed System

HRSA-BR proposes increasing the efficiency and speed of homomorphic RSA operations by converting encoded data into the BERNS system: $\langle m \rangle_{a|b} = \langle m_{a1}, \ldots, m_{at_a}, \ m_{b1}, \ldots, m_{bt_b} \rangle_{a|b}$. To perform a homomorphic multiplication on the encrypted data (let's say m1, m2), it must be $BP > m1 * m2$ when using RNS, that is, it requires m1*m2 times multiplication operations. Using RNS when retrieving numbers to their normal format can create a bottleneck, negating the speed gained during performing operations and slowing data retrieval. But when used, BERNS just uses $BP > m1$, that is, it requires m1 times multiplication operations, where $m1 > m2$. Base Extensions greatly reduce the multiplication load on encrypted data, thus reducing the time consumed. Since RSA security relies on large numbers, and BERNS is more efficient with large numbers but its performance decreases with small numbers, using BERNS with homomorphic RSA is very beneficial.

### 2.4. Homomorphic Multiplication Operations on HRSA-BR

Homomorphic RSA supports multiplication; the application of homomorphic multiplication is simple and becomes more efficient when using BERNS, as Algorithm 2 labelalg:Homomorphic Multiplication RSA demonstrates the homomorphic multiplication process without of BERNS. The homomorphic multiplication process using RSA is straightforward and simple, but it requires more time and effort due to the large mod used to achieve the required level of security.

On the other hand, when using BERNS in homomorphic RSA processing, the encrypted data x and y are converted to BERNS format using a pair of bases. The number is represented by several small numbers

**Algorithm 2** Homomorphic Multiplication RSA

**Input:** $c1 = E(m1)$, $c2 = E(m2)$
**Output:** Encrypted result $c' = E(m1 * m2)$

**Begin**
Multiplication c1, c2 $m$ by
$c' = (c1 * c2)\ mod\ N$
Return Result $c'$
**End**

based on the bases $B_a$ and $B_b$, allowing for parallel and simultaneous processing. After the conversion, $\langle x \rangle_{a|b}$ is directly multiplied by $\langle y \rangle_{a|b}$ to obtain the result, $\langle z \rangle_{a|b}$. The remaining task is to restore the text to its original state z, which is done using Base Extensions transformations under CRT.

## 3. Proposing a Hybrid System that Combines RSA and Paillier Algorithms (RSA-Pa)

The proposal is to combine the RSA and Paillier partial homomorphic encryption algorithms techniques to create a robust and secure algorithm that leverages the unique features of both.

### 3.1. Paillier homomorphic encryption Algorithm

Paillier is a partially homogeneous encryption algorithm that encrypts text using a public key, allowing operations to be performed on its encrypted text. It only supports addition, hence its partial nature. Its characteristics are similar to those of the RSA algorithm, with the difference being the specific operations it supports [20]. This scheme works by generating public keys for encrypting plaintext and private keys for decryption, encryption, decryption, and performing addition operations on the encrypted text.

**Algorithm 3** Paillier Homogeneous Encryption Algorithm [20]

**Input:** prim numbers (p, q), m (message), g, r
**Compute:** N=p*q, $(\phi(N))$, $\lambda(N)$, u=$(g)^{\lambda(N)}\ mod\ N^2$, L(u)= $\frac{u-1}{N}$, $\delta = L(u)^{-1}$
**Public key:** (N, g)
**Privet key:** $(\lambda(N), \delta)$
**Output:** m

**Begin**
Encrypting $m$ by
$E(m) = g^m * r^N mod\ N^2$
Return Result $E(m) = c'$
Decryption $c'$
$m = L((c')^{\lambda(N)} \mod N^2) * \delta \mod N^2$
Return Result m
**End**

The public key is generated primarily by choosing two prime integers of a large size (p and q) to compute N= p*q. Then, the calculation of $LMC(\phi(N))$. the selection of the number $g \in \mathbb{Z}^*_{N^2}$ is used. By using the invers function L(u), ensure g is divided by N. u is $g^{\lambda(N)} \mod N^2$. the public key= (N, g), and the privet key = $(\lambda(N), L(u)^{-1})$. To encryption message m select any $r \in \mathbb{Z}^*_N$ and GCD (r, N) =1. See Algorithm 3.

### 3.2. The Proposing Algorithms (RSA-Pa)

The proposed algorithm combines RSA and Paillier, giving both algorithms enhanced strength and robustness. They work together seamlessly. The process resembles cryptography, encapsulating the plaintext

with two layers of protection, thus increasing its security and making it more difficult to detect. Three items for encryption keys are used instead of two, and three items for decryption keys are used instead of two, which is inherently complex. Initially, the calculations begin with the RSA encryption process, and then the encryption is completed using Paillier.

**Algorithm 4** The Proposing Algorithms (RSA-Pa)

**Input:** prim numbers (p, q), m (message),e, g, r
**Compute:** N=p*q, ($\phi(N)$), $\lambda(N)$, d=$(e)^{-1}\ mod\ (\phi(N))$, u=$(g)^{\lambda(N)}\ mod\ N^2$, L(u)= $\frac{u-1}{N}$, $\delta = L(u)^{-1}$
**Public key:** (N,e, g)
**Privet key:** ($\lambda(N)$, $\delta$, d)
**Output:** m

**Begin**
Encrypting $m$ by
$E(m) = g^{(m^e \mod N)} mod\ N^2 * (r^N mod\ N^2)$
Return Result $E(m) = c'$
Decryption $c'$
m = $(L((c')^{\lambda(N)}\ mod\ N^2) * \delta\ mod\ N)^d\ mod\ N$
Return Result m
**End**

Calculate $c' = g^{(m^e \mod N)} mod : N^2 * (r^N mod : N^2)$ , and for decryption $c'$ to retrieve $m = (L((c')^{\lambda(N)} \mod N^2) * \delta \mod N)^d \mod N$. the operations can perform by the proposed are addition and multiplication. To calculate d, where using $\lambda(N)$ instead of $\phi(N)$ to reduce computational cost, the both $\lambda(N)$ and $\phi(N)$ gives the same results. See algorithm 4. The plaintext space of RSA-Pa is $0 < m < N$, and the ciphertext space is $0 < c < N$. In the proposed RSA-Pa algorithm, the use of BERNS is advantageous because it handles enormous numbers and multiples, thus significantly reducing computational time and effort. This RSA-Pa proposal has not yet implemented BERNS, but future projects may incorporate it.

## 4. Examining results

The results of the work in this paper are divided into three main areas, as follows:

1. The application of the proposed HRSA-BR system yielded good results in increasing the computational speed of encrypted texts using homomorphic RSA on the numbers with size 1024 bits. See Table 2. Using RNS offers several advantages, including distributing the workload of homogeneous operations across parallel processors according to specific principles. This improves memory efficiency because it deals with fractions of numbers rather than large numbers, leading to faster execution by reducing calculation times. Table 2 shows that the time taken to implement decreased by 75% (Note the total time in seconds in Table 2)and the Complexity of multiplication too. Security is maintained by complicating the numbers and converting them to a different format. Such systems are also energy-efficient, making them ideal for resource-constrained applications.

2. The hybrid possesses RSA-Pa both the multiplicative property of RSA and the homomorphic property of Paillier, which supplies the additive property. This type of system is used for specific applications; the hybrid is designed to mix the two kinds of systems. See Table 3. Hybrid systems offer additional security advantages. The system became more robust due to its multiple encryption processes, where it was encrypted first using RSA and then secondly using Paillier. Both encryptions operate simultaneously. In this case, more than one key was used (public key: (N, e, g); private key: ($\lambda(N)$, $\delta$, d)), making it more difficult to obtain the keys. Although the workload on them increases, there remains a need to balance security, performance, and advanced functionality to get the best.

Table 2: comparison between RSA and HRSA-BR with constant parameters - For a Number of k Bases

| **Measures** | **RSA** | **HRSA-BR** | **RSA-Pa** |
|---|---|---|---|
| **Bit numbers of public key** | 1024 | 1024 | 1024 |
| **Bit numbers of privet key** | 1024 | 1024 | 1024 |
| **plaintext space** | $0 < m < N$ | $0 < m < \prod_{i=1}^{t} \mu_i$ | $0 < m < N$ |
| **ciphertext space** | $0 < c' < N$ | $0 < c' < \prod_{i=1}^{t} \mu_i$ | $0 < c' < N$ |
| **Total time in second** | 0.4771238 | 0.119280 | 0.521044 |
| **Complexity of multiplication** | $O(N^2)$ | $O(k)$ | $O(N^3)$ |
| **Calculations Load** | Load | Load-free | Load |
| **Suitability** | for small numbers | for large numbers | for small numbers |
| **Memory efficiency** | Deling with large numbers | small parts of numbers | Deling with large numbers |
| **Safety** | Good | Good | Very good |

Table 3: comparison between RSA, Paillier and RSA-Pa

| **Measures** | **RSA** | **Paillier** | **RSA-Pa** |
|---|---|---|---|
| **Bit numbers of ciphertext** | 1024 | 1024 | 1024 |
| **Key generation** | 0.100 | 0.1041 | 0.1041 N |
| **Encryption** | 0.117 | 0.22901 | 0.270100 |
| **Decryption** | 0.36542 | 0.10969 | 0.136844 |
| **Entropy in bits** | 1028.3 | 2190.1 | 2241.7 |
| **Randomness ratio** | 98.4% | 99.5% | 99.7% |
| **Block Frequency** | 0.403 | 0.887 | 0.906 |
| **Longest Run** | 0.230 | 0.691 | 0.813 |
| **Operations** | * | + | +,* |
| **Complexity** | $O(N^2)$ | $O(N^2 logN)$ | $O(N^3)$ |
| **Safety** | Good | Good | Very good |

All time measurements are in seconds. The desirable p-value is equal to or more than 0.01. Table 3 shows that the hybrid system takes slightly longer than the two individual systems, but this is not significant due to the small difference in time compared to the safety it provides. The solution to the time and speed problem uses BERNS, which doubles the efficiency of the proposed hybrid system.

## 5. Conclusion

The objective of this research is to provide an improvement to a homomorphic encryption system that combines homomorphic RSA and BERNS. The goal of this improvement is to improve the efficiency of multiplication calculations that are carried out on homomorphic encoded data. Because it places a strong emphasis on parallel processing, this method reduces the amount of memory that is utilized while simultaneously reducing the amount of time it takes to execute the program. By doing so, it guarantees that the whole security of RSA is maintained. An RSA-Pa hybrid system was also proposed, utilizing the power of the RSA algorithm in conjunction with the computational intricacy of the Paillier method, which further enhanced the strength and security of the encryption system.

## References

1. S. Nisha, & M. Farik, "RSA public key cryptography algorithm", A Review. International Journal of Scientific and Technological Research, 6, 187–191, 2017.

2. L. F. Katran, E. N. AlShemmary, & W. A. M. Al-Jawher, "Integrating Swin Transformer with Fuzzy Gray Wolve Optimization for MRI Brain Tumor Classification", International Journal of Intelligent Engineering & Systems, 17(6), 2024.

3. L. F. Katran, E. N. AlShemmary, & W. A. Al-Jawher, "Enhanced MRI classification through the integration of a Swin-S Transformer and Db4 wavelet transform", In AIP Conference Proceedings, Vol. 3264, No. 1, p. 040024. AIP Publishing LLC, 2025.

4. J. Ye, L. Wang, Z. Wang, Z. Zhang, Z. Xu, & J. Zhao, "An Electronic Voting Scheme with Privacy Protection", Procedia Computer Science, 243, 1248–1256, 2024.

5. D. Chandravathi, & P. V. Lakshmi, "Privacy preserving using extended Euclidean algorithm applied to RSA-homomorphic encryption technique", International Journal of Innovative Technology and Exploring Engineering, 8(10), 3175–3179, 2019.

6. N. H. Jawad, "FHE Cryptographic Systems with Using Chaotic Secret Key Generation", Boletim da Sociedade Paranaense de Matemática, vol. 43, ISSN 0037-8712, 2025.

7. F. J. Taylor, "Residue arithmetic: a tutorial with examples", Computer, 17(05), 50–62, 1984.

8. N. H. Jawad, & S. Abdulhadi, "Efficient Brakerski-Fan-Vercauteren Algorithm Using Hybrid-Position-Residues Number System", International Journal of Mathematics & Computer Science, 20(2), 2025.

9. N. H. Jawad, & S. Abdulhadi, "Multiple approaches to Convert RNS to Decimal Numbers", Journal of Al-Qadisiyah for Computer Science and Mathematics, 15(3), Page 108, 2023.

10. S. J. Mohammed, & D. B. Taha, "Performance evaluation of RSA, ElGamal, and Paillier partial homomorphic encryption algorithms", In 2022 International Conference on Computer Science and Software Engineering (CSASE), pp. 89–94. IEEE, 2022.

11. A. E. Mesioye, F. T. Ibharalu, S. A. Onashoga, O. M. Olayiwola, "Hensel Lifting: A tool for fast decryption process in RSA cryptosystem", Mathematical Association of Nigeria, 127–134, 2019.

12. K. El Makkaoui, A. Beni-Hssane, & A. Ezzati, "Speedy Cloud-RSA homomorphic scheme for preserving data confidentiality in cloud computing", Journal of Ambient Intelligence and Humanized Computing, 10(12), 4629–4640, 2019.

13. H. Touil, N. El Akkad, & K. Satori, "Homomorphic method additive using Paillier and multiplicative based on RSA in integers numbers", In International Conference on Big Data and Internet of Things, pp. 153–164. Cham: Springer International Publishing, 2021.

14. R. Abid, C. Iwendi, A. R. Javed, M. Rizwan, Z. Jalil, J. H. Anajemba, & C. Biamba, "RETRACTED ARTICLE: An optimised homomorphic CRT-RSA algorithm for secure and efficient communication", Personal and Ubiquitous Computing, 27(3), 1405–1418, 2023.

15. P. Krishnadoss, P. T. Krishnan, N. Paramasivam, D. S. Kesavan, & A. T. Raagav, "Dynamic Approach for Time Reduction in RSA Algorithm through Adaptive Data Encryption and Decryption", International Journal of Intelligent Engineering & Systems, 17(4), 2024.

16. C. Gilbert, & M. A. Gilbert, "Homomorphic Encryption Algorithms for Secure Data Computation", International Research Journal of Advanced Engineering and Science, Volume 10, 2025.

17. N. H. Jawad, & S. Abdulhadi, "Efficient Brakerski-Fan-Vercauteren Algorithm Using Hybrid-Position-Residues Number System", International Journal of Mathematics & Computer Science, 20(2), 2025.

18. D. Chandravathi, & P. V. Lakshmi, "Enhanced homomorphic encryption technique using RSA algorithm with multiple keys", International Journal of Advanced Trends in Computer Science and Engineering, 2019.

19. K. Bigou & A. Tisserand, "RNS modular multiplication through reduced base extensions", 25th IEEE International Conference on Application-Specific Systems, Architectures and Processors (ASAP), IEEE, 2014.

20. P. Paillier, "Public-key cryptosystems based on composite degree residuosity classes", In International Conference on the Theory and Applications of Cryptographic Techniques, pp. 223–238. Berlin, Heidelberg: Springer Berlin Heidelberg, 1999.

*Nibras Hadi Jawad,*
*Department of Mathematics,*
*University of Al-Qadisiyah,*
*Al-Qadisiyah, Iraq.*
*E-mail address:* nibras.hadi@qu.edu.iq