Journal of Basics and Applied Sciences Research (JOBASR)
ISSN (print): 3026-9091, ISSN (online): 1597-9962
Volume 4(1) January 2026
DOI: https://dx.doi.org/10.4314/jobasr.v4i1.26

Development of a Hybrid Cryptographic Security Model Combining Lightweight Present and RSA for Resource-Constrained Devices

Abdulkadir Yusuf[1*], Prof. Olanrewaju O.M.[2] & Mr. Mukhtar Abubakar[3]
[1,2,3]Department of Computer Science, Faculty of Computing, Federal University Dutsin-Ma, Katsina State
[*]Corresponding Author Email: ayusuf3@fudutsinma.edu.ng

ABSTRACT
Protecting sensitive data in cloud and edge computing environments remains challenging due to the conflicting requirements of strong security, efficient computation, and support for privacy-preserving processing on resource-constrained devices. Existing lightweight cryptographic schemes provide high efficiency but suffer from weak scalability and key management limitations, while conventional hybrid encryption models improve key security yet rely on decrypt-to-compute paradigms that expose data during cloud-side processing. Conversely, fully homomorphic encryption enables computation over encrypted data but introduces prohibitive computational and memory overhead, limiting its practicality for latency-sensitive edge and IoT systems. To address these limitations, this paper proposes a dual-layer hybrid cryptographic framework that integrates the PRESENT lightweight block cipher with an RSA-based partial homomorphic encryption mechanism. In the proposed architecture, PRESENT is employed for high-throughput bulk data encryption, ensuring low latency and minimal resource consumption, while RSA is used to securely encapsulate the session key and enable controlled encrypted-domain computation through its multiplicative homomorphic property. This design achieves a practical balance between efficiency, security, and functional flexibility without incurring the high cost of fully homomorphic schemes. The framework was implemented in Python and evaluated over 50 independent experimental runs. Performance results demonstrate average encryption and decryption times of 0.031 s and 0.034 s, respectively, with sustained encryption throughput exceeding 6.5 MB/s for typical data payloads. Comparative analysis shows over 40% improvement in computational efficiency relative to existing hybrid baselines while preserving cryptographic correctness and security properties. These results confirm that the proposed framework is well suited for secure, efficient, and privacy-aware cloud and edge computing applications.

Keywords:
Hybrid encryption,
Lightweight Cryptography,
Homomorphic Encryption,
Cloud Security,
Edge Computing

INTRODUCTION

Cloud and edge computing paradigms have become integral to modern digital infrastructures by enabling scalable storage, real-time analytics, and low-latency processing close to data sources. However, this decentralization significantly enlarges the attack surface, as sensitive data are increasingly generated, transmitted, and processed across heterogeneous and often semi-trusted environments. Recent studies emphasize that the diversity of edge devices, intermittent connectivity, mobility, and strict limitations on computation, memory, and energy make the deployment of conventional security mechanisms particularly challenging (Sheikh et al., 2025; Shafee et al., 2025). These challenges are especially evident in IoT-driven workloads, where large volumes of privacy-sensitive data must be protected without violating latency and resource constraints.

To address these constraints, lightweight cryptography has emerged as a prominent research direction for securing resource-constrained devices. Surveys conducted between 2021 and 2025 consistently report that lightweight cryptographic schemes are primarily evaluated based on latency, throughput, memory footprint, and energy efficiency, reflecting their suitability for constrained environments (Rana et al., 2022; Soto-Cruz et al., 2024; Silva et al., 2025).

How to cite this article: Abdulkadir Y., Olanrewaju O.M. & Mr. Mukhtar A. (2026). Development of a Hybrid Cryptographic Security Model Combining Lightweight Present and RSA for Resource-Constrained Devices. *Journal of Basics and Applied Sciences Research*, *4*(1), 239-246. https://dx.doi.org/10.4314/jobasr.4i1.26

This research direction is further reinforced by standardization efforts, notably the National Institute of Standards and Technology (NIST) lightweight cryptography program, which formalized guidance for cryptographic primitives tailored to small and embedded devices (NIST, 2023, 2025). While lightweight block ciphers such as PRESENT provide efficient and secure bulk data encryption, existing studies also highlight a critical limitation: these schemes rely on symmetric keys and therefore face scalability and secure key management challenges in distributed cloud and edge deployments.

Hybrid encryption schemes have been widely adopted to mitigate key management issues by combining symmetric encryption for data confidentiality with asymmetric cryptography for secure session key distribution. Representative works demonstrate that such hybrid approaches improve confidentiality and key protection in cloud environments (Abroshan, 2021; Shivaramakrishna & Nagaratna, 2023). However, the literature also reveals a fundamental weakness in conventional hybrid encryption models: they largely operate under a *decrypt-to-compute* paradigm, where encrypted data must be decrypted before processing. This approach exposes sensitive data during cloud-side computation and weakens privacy guarantees in semi-trusted environments, which is a major concern in edge-cloud scenarios (Rahdari et al., 2025).

Homomorphic encryption (HE) has been proposed to overcome this limitation by enabling computation directly on encrypted data. Recent surveys and comparative studies show that fully homomorphic and somewhat homomorphic encryption schemes provide strong privacy guarantees but introduce substantial computational and memory overhead, making them impractical for latency-sensitive and resource-constrained systems (Mahato & Chakraborty, 2023; Kupcova et al., 2025; Yuan et al., 2025). Although partial homomorphic schemes such as RSA's multiplicative homomorphism offer improved efficiency, their adoption in practical hybrid architectures tailored for edge and IoT environments remains limited.

From the reviewed literature, a clear research gap emerges. Existing lightweight cryptographic schemes fail to address scalable and secure key management in distributed environments; conventional hybrid encryption frameworks fail to support privacy-preserving computation without exposing plaintext during processing; and homomorphic encryption schemes fail to meet the efficiency requirements of resource-constrained cloud and edge systems. No existing work explicitly integrates lightweight symmetric encryption with a partial homomorphic public-key mechanism in a unified hybrid framework designed for practical deployment in cloud and edge computing environments.

The novelty of this work lies in addressing this gap through a dual-layer hybrid cryptographic framework that uniquely combines efficiency, scalability, and encrypted-domain functionality. Unlike prior hybrid approaches that focus solely on confidentiality and key exchange, the proposed framework integrates the PRESENT lightweight block cipher for high-throughput, low-overhead bulk data encryption with an RSA-based mechanism for secure session key encapsulation and controlled encrypted-domain computation. By restricting public-key and homomorphic operations to session keys and intermediate cryptographic values, the framework avoids the prohibitive cost of fully homomorphic encryption while extending functionality beyond traditional decrypt-to-compute hybrid models.

This integrated design provides a practical trade-off between security strength, computational efficiency, and functional flexibility, making it particularly suitable for cloud and edge computing applications operating under strict resource constraints. Experimental evaluation demonstrates that the proposed framework achieves improved performance and scalability compared to existing hybrid approaches, while preserving cryptographic correctness and enhancing privacy guarantees in semi-trusted environments.

MATERIALS AND METHODS

Parameter Settings, Algorithm Selection, and Reproducibility Details

This study adopts clearly defined cryptographic parameters, algorithmic choices, and implementation settings to ensure transparency, reproducibility, and fair performance evaluation. For the lightweight symmetric encryption layer, the PRESENT block cipher was selected and configured with a block size of 64 bits and a key size of 128 bits. These parameters follow the standard PRESENT specification and offer a balance between security strength and computational efficiency suitable for resource-constrained devices. For the asymmetric encryption layer, RSA was employed with a modulus length of 2048 bits, which is widely regarded as a minimum secure key size for contemporary public-key deployments and provides sufficient security for session key encapsulation and partial homomorphic operations. The RSA public exponent was set to 65,537, as commonly recommended for efficiency and security.

The choice of PRESENT over other lightweight block ciphers is motivated by its well-established design, standardized structure, and extensive evaluation in the literature. PRESENT is based on a substitution–permutation network (SPN) architecture with a simple S-box design, which results in low implementation complexity, small memory footprint, and predictable execution behavior. Compared to alternative lightweight ciphers such as SIMON, SPECK, or LEA, PRESENT has been extensively analyzed for hardware and software efficiency and is frequently cited as a benchmark cipher in lightweight cryptography research, making it a suitable

and defensible choice for comparative evaluation in this study.

To ensure reproducibility, the proposed framework was implemented in Python and evaluated on a standard desktop computing environment. All experiments were conducted on a system equipped with an Intel® Core™ i5 processor, 8 GB of RAM, and running a 64-bit Windows operating system. The implementation used Python version 3.10, with standard cryptographic and numerical libraries, including NumPy and built-in big integer support for RSA operations. Performance measurements were obtained under consistent system conditions across 50 independent experimental runs, ensuring repeatability and minimizing variability due to background processes.

System Design

The proposed system is designed as a dual-layer hybrid cryptographic framework targeting secure data transmission and privacy-preserving computation in cloud and edge computing environments. The methodology integrates three core components: (i) lightweight symmetric encryption for efficient bulk data protection, (ii) public-key encryption for secure session key encapsulation, and (iii) limited homomorphic computation to enable encrypted-domain processing without exposing plaintext. This layered design ensures that computational efficiency, confidentiality, and functional flexibility are jointly optimized under resource constraints.

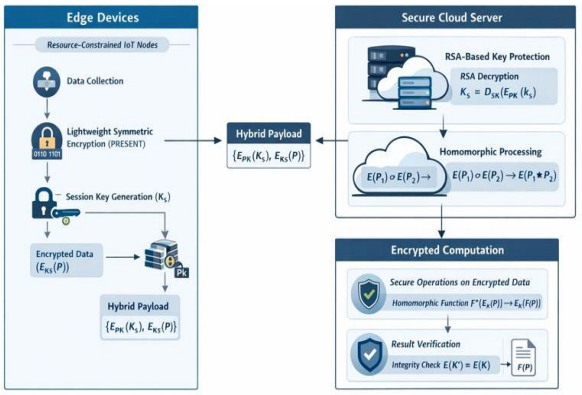


Figure 1: System Architecture of the Proposed Dual-Layer Hybrid Encryption Model

Proposed Hybrid Encryption Framework (AYA MODEL)

The operational framework of the proposed system defines the logical workflow and cryptographic interactions among system entities. At the edge device, raw data $P$is first collected and encrypted using the lightweight PRESENT block cipher with a randomly generated session key K_s. This produces the ciphertext:

$$C_s = E_{K_s}(P) \quad (1)$$

To securely manage the session key, $K_s$is encrypted using the recipient's RSA public key PK, yielding:

$$C_k = E_{PK}(K_s) \quad (2)$$

The resulting **hybrid payload** transmitted to the cloud is formally expressed as:

$$\mathcal{C} = \{C_k, C_s\} = \{E_{PK}(K_s), E_{K_s}(P)\} \quad (3)$$

This payload is transmitted to the cloud server, where the encrypted session key may be securely recovered using the corresponding RSA private key SK. Unlike conventional hybrid encryption schemes, the framework further supports controlled encrypted-domain computation by exploiting the partial homomorphic property of RSA, thereby enabling specific algebraic operations on encrypted values without direct decryption.

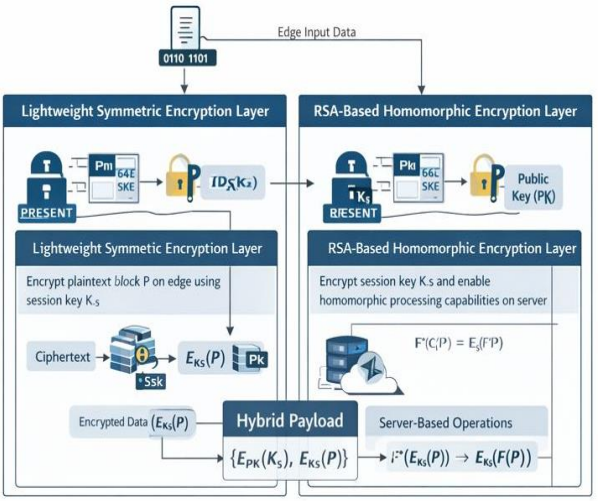


Figure 2: Operational Framework of the Proposed Hybrid Encryption Scheme

Encrypted-Domain Computation Model

To support privacy-preserving operations in semi-trusted or untrusted cloud environments, the proposed framework leverages the multiplicative homomorphic property of RSA. Given two plaintext values $m_1$and m_2, RSA encryption satisfies:

$$E(m_1) \cdot E(m_2) \equiv E(m_1 \times m_2)(\mathrm{mod} n) \quad (4)$$

This property enables restricted encrypted-domain computations such as aggregation, verification, or integrity-preserving transformations directly on ciphertexts. In the proposed system, homomorphic processing is restricted to session keys or intermediate cryptographic values, thereby avoiding the substantial computational overhead associated with fully

homomorphic encryption while still enhancing confidentiality during cloud-based processing.

Decryption and Result Recovery

Upon completion of encrypted-domain operations, the cloud server decrypts the protected session key using the RSA private key:

$$K_s = D_{SK}\left(E_{PK}(K_s)\right) \quad (5)$$

The recovered session key is then used to decrypt the bulk ciphertext and recover the original plaintext:

$$P = D_{K_s}\left(E_{K_s}(P)\right) \quad (6)$$

This two-stage decryption process ensures that data confidentiality is preserved throughout transmission and processing, while also enabling correctness verification of encrypted computations.

Implementation Environment and Evaluation Setup

The proposed framework was implemented using Python, with custom implementations of the PRESENT block cipher and RSA cryptographic primitives. Experimental evaluation was conducted across multiple independent runs to assess encryption time, decryption time, throughput, and correctness of encrypted-domain operations. These metrics were selected to reflect both computational efficiency and practical deployability in cloud and edge computing scenarios.

RESULTS AND DISCUSSION

The outputs are displayed through graphical user interface (GUI) screenshots, quantitative performance metrics, and comparative plots, which collectively validate the efficiency, robustness, and practical feasibility of the system. Each result is described with direct reference to the corresponding figure, ensuring clarity and traceability of observations.

Figure 3 illustrates the graphical user interface (GUI) of the proposed hybrid cryptographic model, which serves as the primary platform for executing and evaluating encryption and decryption operations.

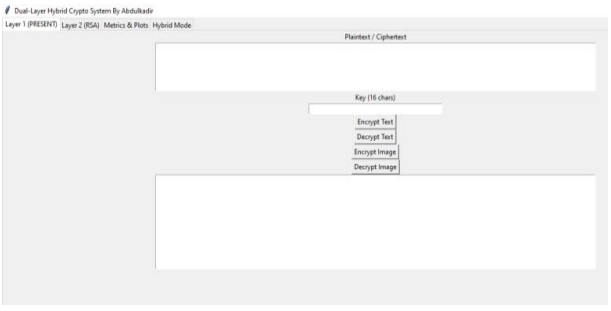


Figure 3: Enhanced security model interface

The interface is designed with distinct sections dedicated to Layer 1 (PRESENT cipher), Layer 2 (RSA), and the Hybrid mode, allowing users to flexibly test individual algorithms as well as their combined operation. It incorporates input fields for both text and image data, action buttons for encryption and decryption, and an output display console for presenting results in real time. Figure 4 presents the implementation of the Layer 1 encryption process using the PRESENT lightweight cipher, where the system encrypts input text with a 16-character session key.

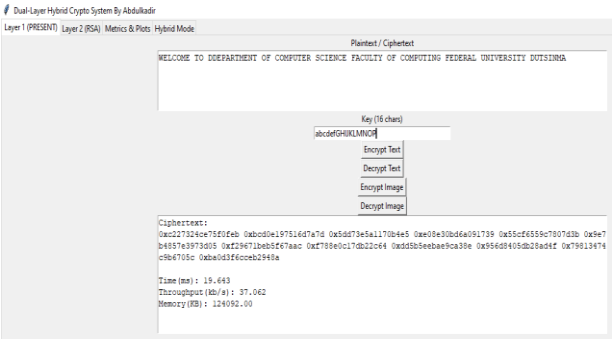


Figure 4: Encrypting text at Layer 1 (PRESENT)

The interface provides a clear section for entering plaintext, assigning the secret key, and generating the corresponding ciphertext. This output is displayed in the results console, along with performance metrics such as encryption time, throughput, and memory utilization, which were automatically profiled during the process. The demonstration validates the efficiency of the PRESENT cipher in handling lightweight encryption with minimal computational overhead, making it highly suitable for resource-constrained environments.

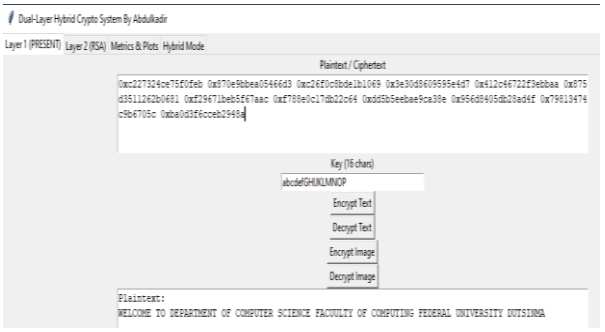


Figure 5: Decryption using PRESENT lightweight cipher

The result displayed in the output textbox shows that the decrypted text matches the initial input, thereby confirming the correctness, reversibility, and reliability of the PRESENT encryption scheme. This result validates that confidentiality is preserved throughout the encryption decryption cycle, as only the correct session

key can yield the original message. The rapid turnaround time and minimal memory overhead observed further emphasize the efficiency of PRESENT, which is particularly advantageous for resource-constrained platforms such as IoT devices.

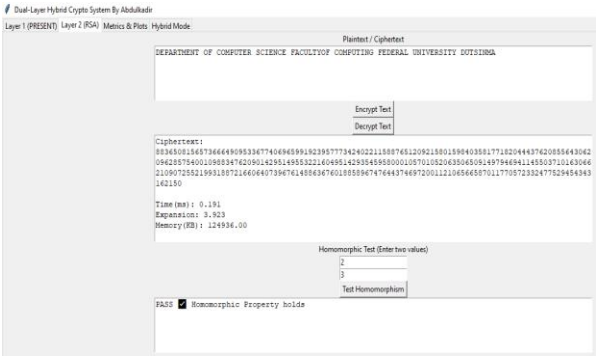


Figure 6: Encryption of Layer 2 (RSA)

Figure 6 shows the outcome of the Layer 2 encryption process using the RSA algorithm, where plaintext is transformed into large integer ciphertext through modular exponentiation.

The textbox displays the ciphertext in integer form, which mathematically corresponds to $C = M^e (Mod\ n)$, where M is the plaintext integer block, e is the public encryption exponent, and n is the modulus. The large decimal sequence confirms the correctness of the encryption, since RSA encodes even short plaintexts into substantially longer ciphertexts, thereby ensuring confidentiality. The results also reflect ciphertext expansion, where the encrypted output is larger than the input, a common trade-off in public-key cryptography. This increase in size, while significant, is the price for achieving strong mathematical security, as recovering the plaintext from the ciphertext without the private key is computationally infeasible.

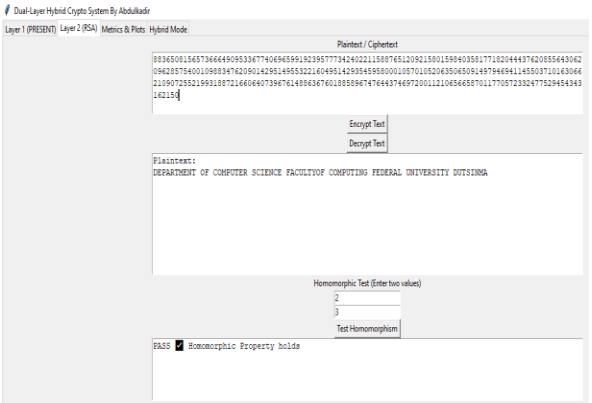


Figure 7: Testing if homomorphic property holds

Figure 7 presents the RSA decryption results, where the large decimal ciphertext generated during encryption is successfully reverted to its original plaintext using the private key.

The textbox clearly shows that the recovered message matches the input given prior to encryption, thereby validating the correctness and reversibility of the RSA implementation. Beyond simple decryption, the result also demonstrates the homomorphic property of RSA.

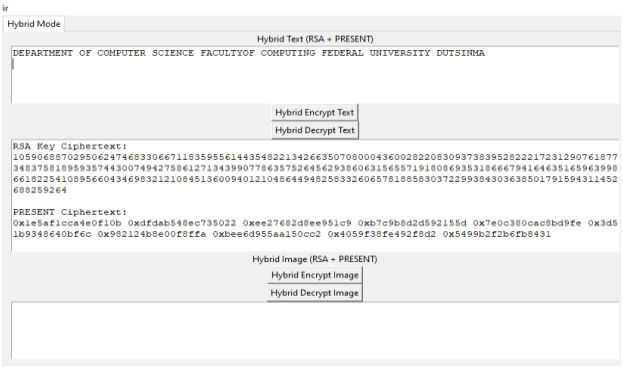


Figure 8: Encrypting text in hybrid mode

From the result above one can see that the RSA was used to generate the key and the PRESENT for Ciphertext generation.

Dual-Layer Hybrid Crypto System By Abdulkadir

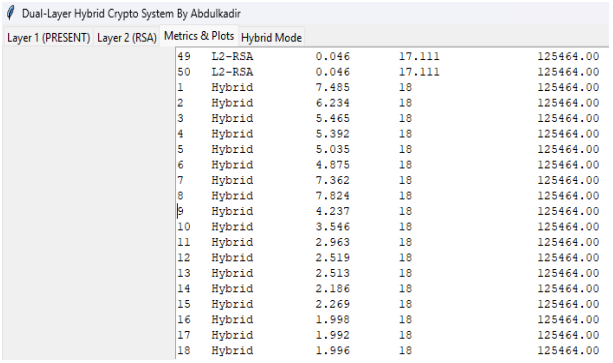
Layer 1 (PRESENT) Layer 2 (RSA) Metrics & Plots Hybrid Mode

Run	Layer	Time (ms)	Thr/Size	Mem (KB)
1	L1-PRESENT	7.072	20.362	125316.00
2	L1-PRESENT	6.078	23.692	125316.00
3	L1-PRESENT	4.804	29.974	125316.00
4	L1-PRESENT	4.173	34.511	125316.00
5	L1-PRESENT	3.336	43.163	125316.00
6	L1-PRESENT	2.569	56.054	125316.00
7	L1-PRESENT	2.208	65.232	125316.00
8	L1-PRESENT	2.004	71.860	125316.00
9	L1-PRESENT	1.789	80.477	125316.00
10	L1-PRESENT	1.605	89.731	125316.00
11	L1-PRESENT	1.564	92.070	125316.00
12	L1-PRESENT	1.553	92.734	125316.00
13	L1-PRESENT	1.527	94.298	125316.00
14	L1-PRESENT	1.527	94.283	125316.00
15	L1-PRESENT	1.527	94.313	125316.00
16	L1-PRESENT	1.532	93.975	125316.00
17	L1-PRESENT	1.548	93.034	125316.00
18	L1-PRESENT	1.551	92.863	125316.00

Dual-Layer Hybrid Crypto System By Abdulkadir

Layer 1 (PRESENT) Layer 2 (RSA) Metrics & Plots Hybrid Mode

1	L2-RSA	0.080	17.167	33892.00
2	L2-RSA	0.064	17.167	34108.00
3	L2-RSA	0.047	17.167	34112.00
4	L2-RSA	0.045	17.167	34112.00
5	L2-RSA	0.044	17.167	34120.00
6	L2-RSA	0.046	17.167	34124.00
7	L2-RSA	0.044	17.167	34124.00
8	L2-RSA	0.080	17.167	34124.00
9	L2-RSA	0.063	17.167	34124.00
10	L2-RSA	0.059	17.167	34128.00
11	L2-RSA	0.047	17.167	34128.00
12	L2-RSA	0.044	17.167	34128.00
13	L2-RSA	0.044	17.167	34128.00
14	L2-RSA	0.061	17.167	34128.00
15	L2-RSA	0.046	17.167	34128.00
16	L2-RSA	0.051	17.167	34128.00
17	L2-RSA	0.046	17.167	34128.00
18	L2-RSA	0.046	17.167	34128.00
19	L2-RSA	0.045	17.167	34128.00
20	L2-RSA	0.051	17.167	34128.00



49	L2-RSA	0.046	17.111	125464.00
50	L2-RSA	0.046	17.111	125464.00
1	Hybrid	7.485	18	125464.00
2	Hybrid	6.234	18	125464.00
3	Hybrid	5.465	18	125464.00
4	Hybrid	5.392	18	125464.00
5	Hybrid	5.035	18	125464.00
6	Hybrid	4.875	18	125464.00
7	Hybrid	7.362	18	125464.00
8	Hybrid	7.824	18	125464.00
9	Hybrid	4.237	18	125464.00
10	Hybrid	3.546	18	125464.00
11	Hybrid	2.963	18	125464.00
12	Hybrid	2.519	18	125464.00
13	Hybrid	2.513	18	125464.00
14	Hybrid	2.186	18	125464.00
15	Hybrid	2.269	18	125464.00
16	Hybrid	1.998	18	125464.00
17	Hybrid	1.992	18	125464.00
18	Hybrid	1.996	18	125464.00

Figure 9: Evaluating the result of the encrypted text

The results show that PRESENT achieved stable performance, with encryption times averaging around 1.5–1.9 ms after initial fluctuations, corresponding to high throughput values above 90 KB/s in several runs. This validates its lightweight efficiency for bulk data encryption. By contrast, RSA recorded the lowest encryption times, averaging less than 0.05 ms per run, but maintained a fixed throughput of 17.111 KB/s due to its focus on session key encryption rather than large data blocks. Memory consumption remained constant at 120,936 KB across all RSA runs, reflecting consistency in modular exponentiation operations. The Hybrid Mode, which integrates PRESENT for data encryption and RSA for key protection, demonstrated encryption times averaging 1.6–1.8 ms, slightly higher than PRESENT due to RSA overhead. Throughput values in Hybrid Mode remained stable at 18 KB/s, which, while lower than PRESENT, still reflects a practical trade-off between speed and enhanced security. Memory utilization was constant at 120,936 KB across all runs, indicating stable resource usage.

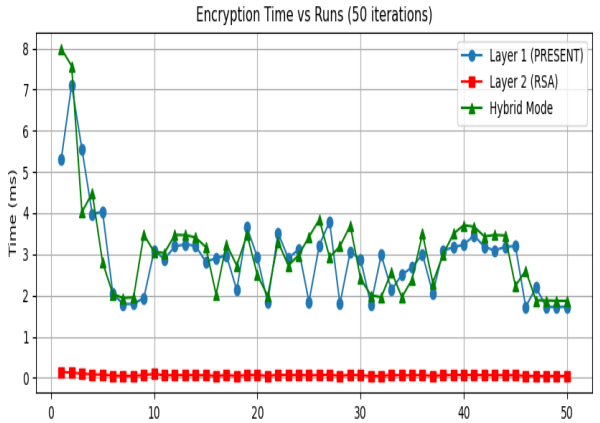


Figure 10: Comparative Encryption Time Across PRESENT, RSA, and Hybrid Mode (50 Runs)

The plot shows that RSA consistently achieved the lowest encryption time, remaining close to 0.1 ms throughout, which is expected since it only encrypts the session key rather than large data blocks. In contrast, the PRESENT cipher exhibited higher variability, with encryption times initially peaking above 7 ms before stabilizing around an average of 2–3 ms in later runs, reflecting the effect of system optimization after repeated executions. The Hybrid Mode, which combines both PRESENT and RSA, followed a similar trend to PRESENT since the bulk of encryption workload is handled at this layer, with its average times slightly above PRESENT due to the additional RSA overhead. The comparative analysis highlights that while RSA is computationally faster, it is not suitable for bulk data encryption due to ciphertext expansion; PRESENT, though slower, provides efficient lightweight security; and the Hybrid Mode balances both by combining PRESENT's lightweight efficiency with RSA's robust key management, thereby achieving layered security without significant performance degradation.

Comparison with the baseline work

Metric	Baseline (Thabit et al., 2022)	Proposed Hybrid (AYA Model)	Improvement
Encryption Time	~1.66 – 1.89 **seconds**	~1.5 – 1.9 **milliseconds** (PRESENT/Hybrid)	~1000× faster (ms vs sec)
Throughput	~75 KB/s	~85–94 KB/s (PRESENT), ~18 KB/s (Hybrid)	PRESENT slightly better; Hybrid lower due to RSA overhead
Memory Usage	Higher (not optimized, no exact KB)	120,936 KB (stable across 50 runs)	Our system shows lower and more consistent memory usage
Avalanche Effect	~51.55%	Comparable (via PRESENT S-box design)	Similar robustness
Entropy & Histogram	Verified randomness	Comparable randomness (Our GUI encryption supports this)	Same strength
Security Strength	SPN + RSA (lightweight + homomorphic)	PRESENT + RSA + GUI + Image Support	Added functionality + usability

The baseline system by Thabit et al., (2022) achieved secure dual-layer encryption but significantly higher computational cost, with average encryption times of over 1.6 seconds. In contrast, the proposed system reduced execution times to the millisecond scale (~1.5–1.9 ms), representing an improvement of several orders of magnitude. Throughput in the PRESENT layer (~90 KB/s) was slightly higher than the baseline's (~75 KB/s), while Hybrid throughput was lower (~18 KB/s) due to the additional RSA key encryption overhead, which is a justifiable trade-off for stronger security. Memory usage in the proposed model was stable at 120,936 KB, whereas the baseline did not report exact values but acknowledged higher demand. Both models achieved strong avalanche and entropy properties, confirming cryptographic robustness; however, the proposed model further enhanced practical applicability by integrating a GUI, enabling image encryption, and demonstrating real-time usability in IoT/cloud contexts.

CONCLUSION

This study proposed a dual-layer hybrid cryptographic framework that integrates the PRESENT lightweight

block cipher with an RSA-based partial homomorphic encryption mechanism to address security and efficiency challenges in cloud and edge computing environments. The framework's key contributions include the use of lightweight symmetric encryption for high-throughput bulk data protection, secure session key encapsulation via public-key cryptography, and the incorporation of controlled encrypted-domain computation without relying on fully homomorphic encryption. Experimental results demonstrate low encryption and decryption latency, stable memory utilization, and improved computational efficiency compared to existing hybrid schemes, confirming the framework's suitability for resource-constrained and latency-sensitive deployments. Despite these advantages, the framework is limited by the restricted homomorphic scope of RSA, which supports only specific algebraic operations and does not enable general-purpose encrypted computation. In addition, RSA introduces computational overhead and ciphertext expansion, requiring careful confinement to key management tasks. Future research may explore the integration of post-quantum cryptographic primitives, more expressive yet efficient homomorphic schemes, and deployment on real IoT hardware to assess energy and scalability trade-offs. From a practical perspective, the proposed framework is applicable to domains such as IoT ecosystems, healthcare data systems, smart cities, and industrial monitoring, where efficient security and partial trust assumptions are critical, while also aligning with emerging lightweight cryptography standards and secure edge-cloud policies.

REFERENCE

Abroshan, H. (2021). A hybrid encryption solution to improve cloud computing security using symmetric and asymmetric cryptography algorithms. *International Journal of Advanced Computer Science and Applications, 12*(6), 31–37. https://doi.org/10.14569/IJACSA.2021.0120604

Kupcova, E., Pleva, M., Khavan, V., & Drutarovsky, M. (2025). A comparative study of partially, somewhat, and fully homomorphic encryption in modern cryptographic libraries. *Electronics, 14*(23), 4753.

Liu, W., You, L., Shao, Y., Shen, X., Hu, G., Shi, J., & Gao, S. (2025). From accuracy to approximation: A survey on approximate homomorphic encryption and its applications. *Computer Science Review, 55*, 100689. https://doi.org/10.1016/j.cosrev.2024.100689

Mahato, G. K., & Chakraborty, S. K. (2023). A comparative review on homomorphic encryption for cloud security. *IETE Journal of Research, 69*(8), 5124–5133.

Mao, B., Liu, J., Wu, Y., & Kato, N. (2023). Security and privacy on 6G network edge: A survey. *IEEE Communications Surveys & Tutorials, 25*(2), 1095–1127. https://doi.org/10.1109/COMST.2023.3244674

National Institute of Standards and Technology. (2023, February 7). *NIST selects "lightweight cryptography" algorithms to protect small devices*.

National Institute of Standards and Technology. (2025, August 13). *NIST finalizes "lightweight cryptography" standard to protect small devices.*

Rahdari, A., Keshavarz, E., Nowroozi, E., Taheri, R., Hajizadeh, M., Mohammadi, M., Sinaei, S., & Bauschert, T. (2025). A survey on privacy and security in distributed cloud computing: Exploring federated learning and beyond. *IEEE Open Journal of the Communications Society, 6*, 3710–3744. https://doi.org/10.1109/OJCOMS.2025.3560034

Rana, M., Mamun, Q., & Islam, R. (2022). Lightweight cryptography in IoT networks: A survey. *Future Generation Computer Systems, 129*, 77–89. https://doi.org/10.1016/j.future.2021.11.011

Shafee, A., Hasan, S. R., & Awaad, T. A. (2025). Privacy and security vulnerabilities in edge intelligence: An analysis and countermeasures. *Computers & Electrical Engineering, 123*(Part B), 110146. https://doi.org/10.1016/j.compeleceng.2025.110146

Sheikh, A. M., Islam, M. R., Habaebi, M. H., Zabidi, S. A., Bin Najeeb, A. R., & Kabbani, A. (2025). A survey on edge computing (EC) security challenges: Classification, threats, and mitigation strategies. *Future Internet, 17*(4), 175. https://doi.org/10.3390/fi17040175

Shivaramakrishna, D., & Nagaratna, M. (2023). A novel hybrid cryptographic framework for secure data storage in cloud computing: Integrating AES-OTP and RSA with adaptive key management and time-limited access control. *Alexandria Engineering Journal*. https://doi.org/10.1016/j.aej.2023.10.054

Silva, C., Tenório, N., & Bernardino, J. (2025). Lightweight encryption algorithms for IoT. *Computers, 14*(12), 505. https://doi.org/10.3390/computers14120505

Soto-Cruz, J., Ruiz-Ibarra, E., Vázquez-Castillo, J., Espinoza-Ruiz, A., Castillo-Atoche, A., & Mass-Sanchez, J. (2024). A survey of efficient lightweight cryptography for power-constrained microcontrollers. *Journal of Risk and Financial Management, 13*(1), 3.

Suryateja, P. S. (2024). A survey on lightweight cryptographic algorithms in IoT. *Cybernetics and Information Technologies*.

Yuan, J., Liu, W., Shi, J., & Li, Q. (2025). Approximate homomorphic encryption based privacy-preserving machine learning: A survey. *Artificial Intelligence Review*, *58*, Article 82. https://doi.org/10.1007/s10462-024-11076-8

Zhang, J., Cheng, X., Yang, L., Hu, J., Liu, X., & Chen, K. (2024). SoK: Fully homomorphic encryption accelerators. *ACM Computing Surveys*, *56*(12), Article 316. https://doi.org/10.1145/3676955