

Computational Time Evaluation of AES and RSA for Text Message Security

Qotrunnada Shobaha Zahirah*,
Fachrur Rozi

Mathematics Study Program, Faculty of
Science and Technology, Maulana Malik
Ibrahim State Islamic University of
Malang, Indonesia

Article History

Received: 25 April 2026
Revised: 17 August 2026
Accepted: 20 October 2026
Published: 31 October 2026



Copyright I 2026 by Authors, Published by
JRMM Group. This is an open access
article under the CC BY-SA License.



Abstract. This study analyzes reported encryption and decryption times for AES-128, direct RSA, and hybrid AES–RSA on text messages of 50, 75, 100, 150, and 200 characters, with 50 repetitions per condition. Aggregate timings and Kruskal–Wallis statistics are interpreted using a mathematical workload model. With fixed parameters, AES data-processing work grows with message length, whereas hybrid RSA key wrapping operates on a fixed 16-byte AES key. The absence of a direct-RSA measurement at 200 characters cannot be attributed to a specific cause because the RSA configuration is undocumented. AES has lower reported mean times than hybrid encryption for the tested messages. However, the substantial disparity between direct-RSA and hybrid decryption times prevents causal conclusions about relative performance. The analysis clarifies the scope of the observed results and the limitations imposed by unavailable implementation details and trial-level measurements.

Keywords: AES-128; computation time; hybrid encryption; RSA; RSA-OAEP; text message security

Abstrak. Penelitian ini menganalisis waktu enkripsi dan dekripsi yang dilaporkan untuk AES-128, RSA langsung, serta skema hibrida AES–RSA pada pesan teks sepanjang 50, 75, 100, 150, dan 200 karakter, dengan 50 pengulangan setiap kondisi. Ringkasan waktu dan statistik Kruskal–Wallis ditafsirkan melalui model matematis beban komputasi. Dengan parameter tetap, beban pemrosesan AES bertambah seiring panjang pesan, sedangkan RSA pada skema hibrida mengenkripsi kunci AES tetap sepanjang 16 byte. Ketidadaan pengukuran RSA langsung untuk 200 karakter belum dapat diatribusikan pada penyebab tertentu karena konfigurasi RSA tidak terdokumentasi. Rata-rata waktu AES lebih rendah daripada waktu skema hibrida pada pesan yang diuji. Namun, perbedaan besar antara waktu dekripsi RSA langsung dan hibrida tidak mendukung kesimpulan kausal mengenai kinerja relatifnya. Analisis memperjelas cakupan temuan serta keterbatasan akibat tidak tersedianya rincian implementasi dan data setiap pengulangan.

Kata kunci: AES-128; enkripsi hibrida; keamanan pesan teks; RSA; RSA-OAEP; waktu komputasi

1. Introduction

The rapid advancement of information and communication technology has increased the use of text-based messaging, including systems that handle personal and financial information. Encryption can protect confidentiality in communications exposed to eavesdropping [1], [2]. Advanced Encryption Standard (AES) is a symmetric block cipher suitable for bulk data, whereas RSA is a public-key cryptosystem that can protect small secret values, including symmetric keys [1], [3].

Hybrid AES–RSA encryption uses AES for the message and RSA for the AES key. Previous applications have explored such combinations in cloud and Internet of Things (IoT) systems [4], [5]. Recent research has also integrated AES–RSA with blockchain-based smart-environment security [6] and compared RSA-based hybrid schemes with ECC-based alternatives under several performance metrics [7]. However, a performance comparison must distinguish the different workloads of RSA applied to the whole message and RSA applied only to a fixed-length key. Moreover, microsecond-level timing measurements are sensitive to implementation details and experimental conditions. Accordingly, the present analysis combines the reported experimental summaries with a workload model to explain the expected dependence of computational effort on input size.

This study examines aggregate experimental results for AES-128, direct RSA, and hybrid AES–RSA applied to short

text messages. The analysis addresses the interpretation of mean times, the payload limit of a single RSA encryption operation, and the distinction between symmetric data processing and asymmetric key wrapping. Because only group summaries and Kruskal–Wallis statistics are available, statistical interpretation is restricted to what can be supported without individual timing observations.

Contributions. The article (i) compares the reported mean times of the three methods across the five tested message lengths; (ii) formulates a block-processing and fixed-key-wrapping workload model; (iii) uses the RSA-OAEP payload limit to examine, without presuming its cause, the missing direct-RSA observation at 200 characters; and (iv) reports conditional effect-size estimates and identifies the limits of the available aggregate data. The contribution is analytical rather than the design of a new cipher or a security proof.

2. Literature Review

2.1. Cryptography and Fundamental Concepts

Cryptography is a field of study concerned with securing information by transforming data into a form that is unintelligible to unauthorized parties. The term cryptography is derived from the Greek words *kryptos* (hidden) and *graphein* (writing), which literally means “hidden writing” [1]. Cryptographic mechanisms can support confidentiality, integrity,

authentication, and non-repudiation when the appropriate encryption, authentication, and signature components are combined [1].

A cryptographic system consists of several fundamental components, namely plaintext, ciphertext, key, and encryption/decryption algorithms. Mathematically, the encryption and decryption processes can be expressed as:

$$C = E_K(P), \quad P = D_K(C)$$

where P denotes plaintext, C denotes ciphertext, and K represents the cryptographic key [1].

Based on key usage, cryptographic algorithms are generally classified into symmetric and asymmetric cryptography. Symmetric cryptography uses a single key for both encryption and decryption, while asymmetric cryptography employs a pair of keys, namely public and private keys [1].

2.2. Number Theory in RSA Cryptography

Modern cryptographic algorithms, particularly RSA, rely heavily on number theory concepts such as divisibility, modular congruence, prime numbers, greatest common divisor (GCD), and Euler's totient function [8], [9]. These concepts form the mathematical foundation for generating secure public and private keys in RSA [1].

Two integers a and b are said to be congruent modulo m if:

$$a \equiv b \pmod{m} \iff m \mid (a - b)$$

This property enables arithmetic operations in modular systems, which are essential in RSA encryption [9].

Euler's totient function $\varphi(n)$ is defined as the number of positive integers less than n that are relatively prime to n . If $n = p \cdot q$ with p and q being prime numbers, then [8]:

$$\varphi(n) = (p - 1)(q - 1)$$

Furthermore, Euler's theorem states that if $\gcd(a, n) = 1$, then:

$$a^{\varphi(n)} \equiv 1 \pmod{n}$$

which is relevant to the correctness of modular-exponentiation operations but does not, by itself, prove RSA encryption security [8].

2.3. Advanced Encryption Standard (AES)

The Advanced Encryption Standard (AES) is a widely used symmetric block cipher for securing digital data [1]. AES operates on fixed block sizes of 128 bits and supports key lengths of 128, 192, or 256 bits.

The AES encryption process consists of several transformation rounds, including SubBytes, ShiftRows, MixColumns, and AddRoundKey. These transformations provide strong diffusion and confusion properties, contributing to the security of properly implemented AES-based schemes [10].

Mathematically, AES represents data in a 4×4 state matrix [1]:

$$S = \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix}$$

AES offers high efficiency and fast computation, especially for large data sizes. However, its main limitation lies in key distribution, as both sender and receiver must securely share the same secret key [5], [11].

2.4. Rivest–Shamir–Adleman (RSA) Algorithm

RSA is an asymmetric cryptographic algorithm that uses a pair of keys: a public key for encryption and a private key for decryption [1]. The security of correctly padded RSA schemes depends on well-specified computational assumptions; the hardness of factoring large integers motivates the use of RSA [3].

The key generation process in RSA involves the following steps:

$$n = p \cdot q \quad (1)$$

$$\varphi(n) = (p - 1)(q - 1) \quad (2)$$

$$e : \gcd(e, \varphi(n)) = 1 \quad (3)$$

$$d \equiv e^{-1} \pmod{\varphi(n)} \quad (4)$$

The textbook RSA encryption and decryption operations are expressed as [3]:

$$C \equiv M^e \pmod{n}, \quad M \equiv C^d \pmod{n}$$

RSA is used for public-key encryption and digital signatures; its particular role depends on the protocol and cryptographic configuration [3].

2.5. Hybrid Cryptography AES–RSA

Hybrid cryptography combines symmetric and asymmetric algorithms to achieve both efficiency and security [12]. In this approach, AES is used to encrypt the actual data, while RSA is used to encrypt the AES key.

The hybrid encryption process can be expressed as:

$$C_{text} = E_{K_{AES}}(P) \quad (5)$$

$$C_{key} = E_{(e,n)}(K_{AES}) \quad (6)$$

The decryption process is given by:

$$K_{AES} = D_{(d,n)}(C_{key}) \quad (7)$$

$$P = D_{K_{AES}}(C_{text}) \quad (8)$$

This approach significantly improves efficiency since RSA is only used for encrypting small-sized keys, while AES handles large data encryption [5], [13].

2.6. Text Message Security

Text messages are one of the most widely used forms of digital communication but are vulnerable to attacks such as sniffing and man-in-the-middle attacks [4]. Therefore, robust encryption mechanisms are required to ensure data security.

In a hybrid AES–RSA system, text messages are encrypted using AES, while the AES key is secured using RSA:

$$C_D = E_{K_s}(P), \quad C_K = RSA_{Enc}(K_s)$$

This approach enhances both security and efficiency compared to single-algorithm systems [2], [5]. However, confidentiality alone does not guarantee message integrity or sender authentication; those properties require a suitable authenticated-encryption or signature mechanism [14], [15].

3. Methods

3.1. Study design and timing procedure

The experimental comparison covered AES-128, direct RSA, and hybrid AES–RSA for text messages of 50, 75, 100, 150, and 200 characters. Fifty repetitions were reported for each condition. The application used Python and Streamlit, with elapsed time measured using `time.perf_counter()`. The underlying measurements and the implementation responsible for generating them are unavailable for independent verification. Therefore, the analysis is based on reported aggregate statistics rather than on reconstructed timing observations.

For a message encoded in bytes, let $m = |\text{UTF8}(P)|$ be its byte length. Importantly, m need not equal the number of characters, particularly for non-ASCII text. Let $T_{a,s}^{(r)}(m)$ denote the elapsed time for algorithm a , stage $s \in \{\text{enc}, \text{dec}\}$, and repetition r . The sample mean is

$$\bar{T}_{a,s}(m) = \frac{1}{50} \sum_{r=1}^{50} T_{a,s}^{(r)}(m). \quad (9)$$

The elapsed-time values analyzed in this study are the recorded sample means, expressed in seconds; no additional experimental measurements are introduced.

3.2. Reported experimental conditions

The documented study design is summarized in Table 1. The hardware platform, operating system, software versions, RSA modulus length, padding and hash settings, AES mode and initialization-vector policy, and precise timing boundaries were not specified. These omissions limit reproducibility and prevent confirmation that all cryptographic operations were measured under comparable conditions.

Table 1: Reported experimental conditions

Condition	Recorded specification
Symmetric cipher	AES-128 (16-byte key)
Message lengths	50, 75, 100, 150, 200 characters
Replications	50 per condition
Application	Python with Streamlit
Time measurement	<code>time.perf_counter()</code>

3.3. Hybrid encryption and decryption

Let K_A be a 128-bit AES key, K_{pub} and K_{priv} an RSA key pair, and P a message. The hybrid encryption output comprises the encrypted text and wrapped key:

$$C_{\text{text}} = E_{\text{AES}}(K_A, P), \quad (10)$$

$$C_{\text{key}} = E_{\text{RSA}}(K_{\text{pub}}, K_A), \quad (11)$$

$$C_{\text{total}} = (C_{\text{text}}, C_{\text{key}}). \quad (12)$$

Decryption performs

$$\hat{K}_A = D_{\text{RSA}}(K_{\text{priv}}, C_{\text{key}}), \quad (13)$$

$$P' = D_{\text{AES}}(\hat{K}_A, C_{\text{text}}). \quad (14)$$

Correct functional operation requires $P' = P$. The AES mode, nonces/IVs, padding, and authentication tag (if applicable) must also be transmitted or otherwise synchronized.

The mode of operation, associated metadata, and authentication mechanism were not documented; therefore, the security properties of the implemented scheme cannot be fully assessed. Encryption alone does not establish integrity or authentication [14], [15].

3.4. Algorithmic flow

The two stages of the hybrid procedure are illustrated in Figure 1. This representation emphasizes that the plaintext is processed by AES, whereas RSA is applied to the short AES key rather than the text itself.

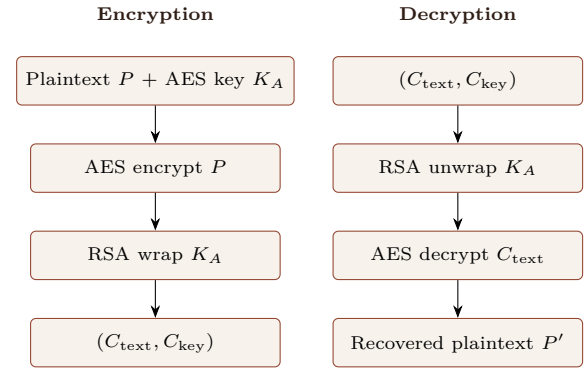


Figure 1: Schematic operations of hybrid AES–RSA encryption and decryption.

3.5. Mathematical workload model

AES operates on 128-bit (16-byte) blocks. Let $b(m)$ denote the number of block-cipher invocations needed for a message of m bytes under a fixed mode and padding policy. For example, $b(m) = \lceil m/16 \rceil$ in an unpadded counter-mode data path; padding may add a final block in other modes. Thus, for a fixed AES key size and implementation,

$$b(m) = \Theta\left(\left\lceil \frac{m}{16} \right\rceil\right), \quad T_{A,s}(m) = a_s b(m) + c_s \quad (15)$$

models the dominant block processing and fixed setup overhead, with a_s, c_s implementation-dependent. Hence the asymptotic data-processing workload is $\Theta(m)$ as m grows, not a guarantee of strictly increasing wall-clock time for very short inputs.

For hybrid AES–RSA, the RSA input is the 16-byte key K_A , not P . Let $R_{\text{enc}}(16)$ and $R_{\text{dec}}(16)$ be the costs of RSA key wrapping and unwrapping for fixed RSA parameters. A component-level timing decomposition is

$$T_{H,\text{enc}}(m) = T_{A,\text{enc}}(m) + R_{\text{enc}}(16) + \delta_{\text{enc}}(m), \quad (16)$$

$$T_{H,\text{dec}}(m) = R_{\text{dec}}(16) + T_{A,\text{dec}}(m) + \delta_{\text{dec}}(m), \quad (17)$$

where $\delta_s(m)$ denotes surrounding overhead, such as object creation, serialization, and measurement setup. Under fixed key size and padding, the *RSA computation itself* has no payload-length dependency. The total hybrid runtime can still fluctuate with m because AES and the surrounding code execute additional work. Equations (16)–(17) are workload predictions, not a numerical decomposition of the recorded timings. A recent RSA–AES chat-application study reports separate measurements for AES message processing and RSA session-key wrapping under a different cryptographic configuration [16].

3.6. Direct RSA input-size restriction

The largest plaintext supported by a *single* RSA encryption depends on the modulus and encoding scheme. For RSAES-OAEP, RFC 8017 gives [15]

$$L_{\max} = k - 2h_{\text{len}} - 2 \text{ bytes}, \quad (18)$$

where $k = \lceil \text{bitlen}(n)/8 \rceil$ is the RSA modulus length in bytes and h_{len} is the hash digest length in bytes. For example, 2048-bit RSA with OAEP-SHA-256 gives $256 - 64 - 2 = 190$ bytes, while OAEP-SHA-1 gives $256 - 40 - 2 = 214$ bytes. The examples illustrate the dependence of the limit on cryptographic parameters and do not establish the settings used in the reported experiment. If a plaintext exceeds $L_{\max}$, it may be divided into $q = \lceil m/L_{\max} \rceil$ blocks and processed separately, although a properly specified hybrid encryption scheme is generally preferable for large messages. For fixed RSA parameters, the direct-RSA operation count scales with q .

3.7. Statistical analysis

Kruskal–Wallis statistics H were reported for the three methods at message lengths of 50, 75, 100, and 150 characters. For $k = 3$ groups, the reference chi-squared distribution has $k - 1 = 2$ degrees of freedom. Significance values printed as 0.000 to three decimal places are reported as $p < 0.001$, rather than as exact zeros. With a nominal total sample size of $N = 3 \times 50 = 150$, the following descriptive effect-size estimator is calculated from each reported H :

$$\varepsilon_H^2 = \frac{H - k + 1}{N - k} = \frac{H - 2}{147}. \quad (19)$$

Interpretation of the Kruskal–Wallis test requires independent observations. Without trial-level data, independence, ties, and the dependence structure of repetitions cannot be assessed. Consequently, the reported p values and derived effect sizes are interpreted conditionally, and no additional inferential tests are performed. A repeated or paired measurement design would instead require an analysis that accommodates the corresponding dependence structure.

4. Results and Discussion

4.1. Functional validation

A reported functional check recovered the tested plaintext after hybrid encryption and decryption, $P' = P$ in (14). Such a round-trip test demonstrates recovery for that input; it does not establish security, correctness for every message, or interoperability across implementations.

4.2. Reported computational time

The recorded sample means, expressed in seconds, are presented in Table 2. Missing entries are shown with dashes and are not treated as zeros.

For comparable reported conditions (50–150 characters), AES has the lowest mean time and direct RSA the largest mean decryption time. The hybrid timings are between those

of AES and direct RSA for these conditions, but this comparison alone is insufficient to establish general superiority under matched hardware, cryptographic parameters, and timing boundaries. In particular, AES encryption ranges from 0.000064 to 0.000096 seconds and is not monotonically increasing with message length. At such short durations, fixed software overhead and measurement noise can dominate the cost of processing a few additional AES blocks.

The missing direct-RSA entries at 200 characters should not be interpreted as an algorithm failure. As shown in (18), a single RSA-OAEP operation has a modulus- and hash-dependent byte capacity. For example, a 200-byte ASCII message exceeds the 190-byte limit of RSA-2048/OAEP-SHA-256, but not the 214-byte limit of RSA-2048/OAEP-SHA-1. The available documentation does not identify the RSA modulus, padding, character encoding, or any error trace. The cause of the missing measurements therefore remains undetermined.

4.3. Interpretation of hybrid processing times

Equations (16) and (17) imply that the work performed by RSA on the 16-byte AES key is independent of the text length when its parameters are fixed. Only the AES data path and surrounding overhead need increase with the message size. The aggregate hybrid timings do not permit a numerical decomposition into AES and RSA components. Component-level measurements would be required to estimate the separate contributions from key wrapping, data encryption, key unwrapping, and data decryption.

A further implementation question emerges from Table 2: direct-RSA decryption is approximately 0.049–0.050 seconds, whereas *the entire hybrid decryption* is around 0.0017–0.0025 seconds. Both workflows are described as involving a private-key RSA operation. The asymmetry may reflect differences in RSA key size, key generation inside the timing boundary, library functions, caching, or what each timer includes. The available metadata cannot distinguish among these explanations. Therefore, the results do not establish that the RSA private-key operation itself is faster within the hybrid procedure.

4.4. Kruskal–Wallis results and effect size

Table 3 presents the reported Kruskal–Wallis statistics and the corresponding ε_H^2 estimates from (19). Values of p rounded to 0.000 in the results are expressed as $p < 0.001$. The effect sizes are algebraically derived from H and the reported nominal sample sizes, without imputing individual timings.

Under the assumptions required for the test, these results indicate differences in the group distributions. They do not, by themselves, identify every significantly different pair, verify the accuracy of the timers, or establish that the algorithms differ only because of the encryption primitive. Pairwise post-hoc analysis, matched hardware parameters, and independent or blocked repeated-trial data would be needed for stronger conclusions. The reported data also omit a comparable three-group test at 200 characters.

Table 2: Mean elapsed times (seconds), 50 repetitions per condition

Length (characters)	AES enc.	AES dec.	RSA enc.	RSA dec.	Hybrid enc.	Hybrid dec.
50	0.000096	0.000061	0.001227	0.048863	0.001311	0.001724
75	0.000075	0.000041	0.001357	0.049731	0.000899	0.002503
100	0.000064	0.000047	0.001268	0.049054	0.000952	0.001733
150	0.000075	0.000077	0.001250	0.050240	0.001511	0.001672
200	0.000068	0.000044	–	–	0.001397	0.001698

Abbreviations: Enc., encryption; dec., decryption. Dashes indicate unavailable direct-RSA measurements at 200 characters.

Table 3: Reported Kruskal–Wallis statistics and derived effect sizes

Length	Stage	H	p	ϵ_H^2
50	Enc.	95.537	< 0.001	0.636
50	Dec.	132.460	< 0.001	0.887
75	Enc.	120.481	< 0.001	0.806
75	Dec.	132.472	< 0.001	0.888
100	Enc.	118.890	< 0.001	0.795
100	Dec.	132.457	< 0.001	0.887
150	Enc.	117.738	< 0.001	0.787
150	Dec.	132.475	< 0.001	0.888

4.5. Measurement uncertainty and statistical limitations

The available summaries do not include trial-level measurements or Shapiro–Wilk statistics. Thus, sample distributions, normality assessments, uncertainty intervals, and pairwise comparisons cannot be independently reconstructed. In particular, the absence of raw measurements prevents a check of whether the same inputs or experimental runs were shared across methods. The statistical results are therefore interpreted as reported, subject to the independence assumption stated in Section 3.7.

For elapsed times of approximately 0.00004–0.00010 seconds, runtime initialization, operating-system scheduling, garbage collection, and user-interface activity can be substantial relative to the operation being measured. High timer resolution alone does not ensure accurate benchmarking. A controlled follow-up study would record the complete hardware and software environment, define the timed operation boundaries, use warm-up iterations and repeated trials, and report distributions or confidence intervals. Separate timing of AES data processing and RSA key wrapping would help explain the observed decryption disparity. Larger payloads would also be informative, with RSA input-size constraints documented explicitly.

4.6. Discussion and scope of the contribution

The study illustrates a well-known engineering trade-off: symmetric cryptography processes bulk messages efficiently, whereas public-key cryptography enables a recipient to recover a protected symmetric key without an initially shared AES secret. The mathematical results in (15)–(18) explain why the hybrid RSA cost need not grow with message length while the AES data-processing workload does. These results represent an explicit analytical interpretation of existing cryptographic techniques, not a new cryptographic construction. Security is not measured by execution time, and stronger security than AES alone cannot be inferred without

specifying the threat model, key management, authentication, and cryptographic parameters.

5. Conclusion

The reported comparison covers AES-128, direct RSA, and hybrid AES–RSA on messages of 50–200 characters, using 50 repetitions per condition. For the reported 50–150-character cases, AES shows the lowest mean times and direct RSA the longest decryption times. However, the data do not support an unqualified speed or security ranking because timing boundaries and cryptographic configurations are undocumented. The unreported RSA measurement at 200 characters is consistent with an implementation-dependent single-operation size limit but is not sufficient to establish its cause. The mathematical analysis shows that AES has a block-processing workload proportional to message size, while RSA in hybrid encryption processes a fixed 16-byte key for fixed parameters.

The absence of trial-level data and full implementation metadata limits independent reproduction and attribution of the observed timing differences. The particularly large disparity between direct RSA and hybrid RSA decryption calls for controlled component-level measurements. Future experiments should include larger payloads, explicit timing boundaries, and documented authenticated-encryption and key-handling configurations.

CRedit Authorship Contribution Statement

Qotrunnada Shobaha Zahirah: Conceptualization, Methodology, Software, Validation, Formal Analysis, Investigation, Data Curation, Writing–Original Draft Preparation.
Fachrur Rozi: Supervision, Validation, Writing–Review and Editing.

Declaration of Competing Interest

The authors declare no conflict of interest.

Funding and Acknowledgments

This study did not receive any external funding. The authors thank the Mathematics Study Program, Faculty of Science and Technology, Maulana Malik Ibrahim State Islamic University of Malang for facilities and support.

Data and Code Availability

The aggregate elapsed-time data and Kruskal–Wallis statistics used in this analysis are reported in Tables 2 and 3. Individual trial-level measurements and the implementation used to produce the recorded timings are not provided with this article. Independent verification of the recorded measurements and reanalysis at the trial level are therefore not possible from the available material.

References

- [1] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th. Pearson, 2017.
- [2] Y. M. Walle, “Hybrid rsa–aes based software defined network to improve the security of manet,” *Open Information Science*, vol. 8, no. 1, 2024.
- [3] A. J. Menezes, P. C. Van Oorschot, and S. A. Vanstone, *Handbook of Applied Cryptography*. CRC Press, 1996.
- [4] R. Akter, M. A. R. Khan, F. Rahman, S. J. Soheli, and N. J. Suha, “Rsa and aes based hybrid encryption technique for enhancing data security in cloud computing,” *International Journal of Computational and Applied Mathematics & Computer Science*, vol. 3, pp. 60–71, 2023. DOI: [10.37394/232028.2023.3.8](https://doi.org/10.37394/232028.2023.3.8)
- [5] Q. Chang, T. Ma, and W. Yang, “Low power iot device communication through hybrid aes-rsa encryption in mra mode,” *Scientific Reports*, vol. 15, no. 1, pp. 1–25, 2025. DOI: [10.1038/s41598-025-98905-0](https://doi.org/10.1038/s41598-025-98905-0)
- [6] S. A. Al-Abassy, “An AI-based blockchain framework with AES–RSA hybrid encryption for secure data sharing in smart environments,” *Frontiers in Blockchain*, vol. 9, 2026, Art. no. 1935632. DOI: [10.3389/fbloc.2026.1935632](https://doi.org/10.3389/fbloc.2026.1935632) Available online.
- [7] S. J. Saydahd, R. K. Muhammed, S. A. Hassan, and A. M. Al-addin, “A comparative performance evaluation of hybrid encryption techniques using ECC, RSA, AES, and ChaCha20 for secure data transmission,” *Iraqi Journal of Industrial Research*, vol. 12, no. 2, pp. 157–172, 2025. DOI: [10.53523/ijoirVol12I2ID598](https://doi.org/10.53523/ijoirVol12I2ID598) Available online.
- [8] K. H. Rosen, *Elementary Number Theory and Its Applications*, 7th. Pearson Education, 2021.
- [9] W. Stein, *Elementary Number Theory: Primes, Congruences, and Secrets*. Open Textbook, 2017.
- [10] A. S. Mabururi, “Data security system of text messaging based on android mobile devices using advanced encryption standard dynamic s-box,” *Journal of Soft Computing Exploration*, vol. 1, no. 1, pp. 39–46, 2020. DOI: [10.52465/joscex.v1i1.10](https://doi.org/10.52465/joscex.v1i1.10)
- [11] K. K. Khaing Oo and Y. N. Soe, “Encryption data measurement and data security of hybrid aes and rsa algorithm,” *International Journal of Trend in Scientific Research and Development (IJT-SRD)*, vol. 3, no. 6, pp. 834–838, 2019. Available online.
- [12] A. Mitali, S. Ayesha, K. Kaynat, A. Tuika, and P. Kudal, “Encryption and decryption (file cryptography with aes and rsa for mobile based on android),” *International Journal of Computer Science and Mobile Computing*, vol. 10, no. 4, pp. 132–138, 2021. DOI: [10.47760/ijcsmc.2021.v10i04.018](https://doi.org/10.47760/ijcsmc.2021.v10i04.018)
- [13] A. M. Rasheed and R. M. S. Kumar, “Efficient lightweight cryptographic solutions for enhancing data security in healthcare systems based on iot,” *Frontiers in Computer Science*, vol. 7, no. April, pp. 1–22, 2025. DOI: [10.3389/fcomp.2025.1522184](https://doi.org/10.3389/fcomp.2025.1522184)
- [14] M. Dworkin, “Recommendation for block cipher modes of operation: Methods and techniques,” National Institute of Standards and Technology, Tech. Rep. NIST SP 800-38A, 2001. DOI: [10.6028/NIST.SP.800-38A](https://doi.org/10.6028/NIST.SP.800-38A) Available online.
- [15] K. Moriarty, B. Kaliski, J. Jonsson, and A. Rusch, “Pkcs #1: Rsa cryptography specifications version 2.2,” Internet Engineering Task Force, Request for Comments 8017, 2016. DOI: [10.17487/RFC8017](https://doi.org/10.17487/RFC8017) Available online.
- [16] Nurdin, S. A. D. Ghani, N. Salman, Suryani, and M. Oktavianus, “Hybrid RSA–AES message encryption in a progressive web app (PWA) chat application,” *Indonesian Journal of Enterprise Architecture*, vol. 3, no. 3, pp. 99–111, 2026. DOI: [10.66314/ijea.v3i3.1226](https://doi.org/10.66314/ijea.v3i3.1226) Available online.