Kateb
Scientific–Research Journal of Technology and Engineering
Vol.1 | No.1 | Autumn & Winter 2025–2026
Kateb University
Email: kjte@kateb.edu.af | Homepage: kjte.kateb.edu.af

Performance and Resource Trade-offs in Rivest–Shamir–Adleman Public-Key Cryptography: An Experimental Study on the Impact of Key Length

Sayeed jaweed Naderi[1]✉

1. Visiting Lecturer, Department of Information Technology, Faculty of Computer Science, Kateb University, Kabul, Afghanistan. E-mail: sayeedjaweednaderi@gmail.com

Article Info

Article type:
Research Article

Article history:
Received: 07/02/2026
Received in revised form: 11/02/2026
Accepted: 12/03/2026
Available online: 19/03/2026

Keywords:
Rivest–Shamir–Adleman (RSA), public-key cryptography, cryptographic performance, key length, digital signatures, benchmarking, computational overhead

ABSTRACT

Rivest–Shamir–Adleman (RSA) public-key cryptography remains a foundational mechanism in modern secure communication systems, supporting technologies such as Transport Layer Security (TLS), digital signatures, and Public Key Infrastructure (PKI). Although increasing RSA key length strengthens resistance against classical factorization-based attacks, it also increases computational cost and resource consumption, especially in systems with limited processing capacity or high transaction volumes. This study presents a controlled empirical evaluation of RSA performance across four key sizes: 1024, 2048, 3072, and 4096 bits. A Java-based benchmarking framework was used to measure key generation latency, encryption and decryption time, digital signature generation and verification cost, and ciphertext/signature size overhead. The novelty of this article lies in providing a focused, reproducible, and quantitative comparison of RSA key-length trade-offs using commonly recommended modern constructions: RSA Optimal Asymmetric Encryption Padding (RSA-OAEP) with Secure Hash Algorithm 256-bit (SHA-256) and RSA Probabilistic Signature Scheme (RSA-PSS) with SHA-256. The results show that private-key operations, including decryption and digital signing, grow superlinearly as key size increases. In particular, 4096-bit RSA decryption and signing were approximately 29 times slower than their 1024-bit counterparts, while key generation increased by nearly 120 times. In contrast, public-key operations such as encryption and signature verification showed more moderate growth because of the use of smaller public exponents. Output sizes also increased linearly with modulus length, from 128 bytes for 1024-bit keys to 512 bytes for 4096-bit keys. These findings demonstrate the security–performance trade-off inherent in RSA and show that larger keys may impose practical limitations in performance-sensitive or resource-constrained environments. The study is limited to one hardware platform and Java cryptographic provider, so results may vary across other systems and implementations. Overall, the results suggest that RSA-2048 remains a practical baseline for many applications, while RSA-3072 or RSA-4096 should be selected only when higher security margins justify the additional computational overhead. This revision fixes the abstract comments by making the methodology shorter and clearer, adding the novelty explicitly, defining abbreviations on first use, and mentioning the limitation briefly. It also improves alignment and academic flow compared with the original abstract.

Cite this article: Naderi, S. (2026). Performance and Resource Trade-offs in Rivest–Shamir–Adleman Public-Key Cryptography: An Experimental Study on the Impact of Key Length, *Kateb Scientific-Research Journal of Technology and Engineering*, *1 (1)*, 35-48.

 Publisher: Kateb University.

I. INTRODUCTION

Public-key cryptography enables secure communication over untrusted networks and is essential for modern digital infrastructure. Rivest–Shamir–Adleman **(RSA)** public-key cryptography continues to be widely deployed in certificate infrastructures, authentication mechanisms, and digital signature systems, even as modern security guidance increasingly emphasizes stronger parameters and crypto-agility [1], [4], [5]. RSA plays a central role in protocols and applications such as Transport Layer Security **(TLS)**, secure email, software signing, and Public Key Infrastructure **(PKI)** frameworks. Despite the emergence of alternative public-key schemes and ongoing research into post-quantum cryptography, RSA remains deeply embedded in many security architectures because of its maturity, interoperability, and widespread support across platforms and standards.

Increasing RSA key length improves resistance against classical factorization-based attacks and extends the cryptographic lifetime of deployed systems. Security recommendations have therefore progressively shifted toward larger modulus sizes as computational power continues to increase and cryptanalytic methods evolve [1]. However, increasing key size is not without cost. Larger RSA keys significantly increase computational complexity, especially for private-key operations such as decryption and digital signing. These costs can introduce substantial latency and resource overhead, affecting real-world deployments such as server-side authentication, certificate validation, secure messaging systems, and TLS handshake processing [1], [4]. In high-throughput environments, including web servers, cloud services, and embedded systems, even small increases in cryptographic processing time can have measurable performance impacts.

Although previous research and standards have discussed RSA security levels, protocol recommendations, and the general performance implications of larger keys, the innovation of this paper is its focused empirical evaluation of RSA key-length trade-offs using a controlled Java-based benchmarking framework. Unlike studies that mainly discuss RSA from a theoretical, standardization, or protocol-specific perspective, this research directly measures and compares the performance of RSA operations across 1024, 2048, 3072, and 4096-bit keys. The study evaluates key generation, encryption, decryption, digital signature generation, verification, and output-size overhead under the same experimental conditions. This provides a clearer operation-by-operation understanding of how RSA performance changes as key length increases.

The main contribution of this research is the development and application of a reproducible benchmarking methodology for analyzing the relationship between RSA key length, computational cost, and resource overhead. First, the study provides quantitative measurements for both public-key and private-key RSA operations. Second, it identifies which operations are most affected by key-size growth. Third, it connects the experimental findings with contemporary cryptographic guidance and future migration pressures, including post-quantum transition planning [13], [15], [20], [21]. In this way, the paper contributes practical evidence that can help system designers, security engineers, and policymakers choose RSA key sizes based on both security requirements and performance constraints.

This study addresses the following research questions:

1. How does RSA key length affect key generation cost?

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

2. How does performance scale for encryption, decryption, signing, and verification?
3. What is the output-size overhead associated with larger RSA modulus sizes?
4. Where does performance degradation become practically significant for performance-sensitive applications?

By answering these questions, this research highlights the security–performance trade-off inherent in RSA and provides practical guidance for selecting key lengths in modern secure communication systems.

II. BACKGROUND AND RELATED WORK

A. RSA Operations and Padding Schemes

RSA performance is dominated by modular exponentiation over large integers. Practical deployments typically use RSA-OAEP for encryption and RSA-PSS for signatures due to their stronger security properties compared to legacy PKCS#1 v1.5 padding [3], [5], [7]. Contemporary protocol guidance emphasizes secure configuration and deprecation of weak/legacy choices in transport security settings [4].

B. Scaling Behavior and Implementation Considerations

The time cost of modular exponentiation grows superlinearly with key length due to big-integer multiplication and reduction costs. Modern implementations rely on optimized algorithms (e.g., Montgomery multiplication) and library-specific optimizations; nevertheless, empirical scaling remains pronounced for private-key operations (decryption and signing) [10], [17]. Research continues on accelerating RSA via parallelization and hardware-aware methods [18], [17].

C. Prior Performance Measurements in Applied Contexts

Performance impacts of large RSA keys appear in applied domains such as DNSSEC and TLS. Practical measurement studies show that RSA-4096 can impose significant signing/validation and bandwidth overhead compared to smaller keys or alternative signature schemes [19]. Comparative studies of asymmetric cryptosystems and modern benchmarking frameworks further motivate empirical evaluation over purely theoretical complexity arguments [16], [20].

III. METHODOLOGY

A. Experimental Environment

Experiments were conducted on a Dell Latitude 5320 laptop running Windows 11 Pro 64-bit (Version 10.0, Build 26100). The system was equipped with an 11th Gen Intel® Core™ i7-1185G7 processor (4 physical cores, 8 logical processors) with a base frequency of 3.00 GHz and support for Turbo Boost technology. The system had 16 GB of DDR4 RAM.

All benchmarks were executed using OpenJDK 25.0.1 (64-bit). The default SunRsaSign security provider was used for RSA key generation and signature operations. RSA-OAEP with SHA-256 (MGF1) and RSASSA-PSS with SHA-256 were configured using standard algorithm names supported by the Java Security API and provider documentation [7][9].

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

To reduce JVM warm-up bias, each benchmark included 20 warm-up iterations prior to measurement. Performance metrics were then collected over 200 measured iterations per configuration. All experiments were conducted under typical desktop operating conditions without manual CPU frequency locking; no additional system load was intentionally introduced during benchmarking.

B. Key Sizes and Workloads

Key sizes evaluated were 1024, 2048, 3072, and 4096 bits, selected to span legacy compatibility (1024), common deployment baseline (2048), and higher-security configurations (3072, 4096) referenced in key-management and signature guidance [1], [5], [13]. Message sizes of 32, 128, and 190 bytes were used for OAEP tests, reflecting typical usage where RSA encrypts small payloads (e.g., symmetric keys) rather than bulk data [3], [11].

C. Metrics

We measured (i) key generation latency, (ii) encryption latency, (iii) decryption latency, (iv) signing latency, (v) verification latency, and (vi) output size in bytes (ciphertext or signature). We report mean, median, standard deviation, and min/max to capture tail behavior relevant to latency-sensitive systems.

Table 2 Comparative Summary of Related Work

Source	Focus	Relation to This Study
NIST SP 800-57 [1]	Key-management and security-strength guidance.	This study measures the performance cost of different RSA key lengths.
RFC 9325 [4]	Secure use of TLS and DTLS.	This study explains how RSA key size can affect secure communication performance.
NIST FIPS 186-5 [5]	Digital signature standards.	This study evaluates RSA signing and verification performance.
Cormen et al. [10] and Boneh and Shoup [11]	Algorithmic and cryptographic foundations of RSA.	This study connects theoretical RSA scaling with practical measurements.
Desai et al. [16]	Performance of asymmetric cryptosystems.	This study gives a more focused RSA key-length analysis.
Parihar and Nakhate [17]; Liu et al. [18]	RSA acceleration and optimization.	This study evaluates standard RSA performance without specialized acceleration.
APNIC Labs [19]	RSA-4096 in DNSSEC.	This study expands the analysis beyond DNSSEC to several RSA operations.
Kampanakis [20]; Fitzgibbon and Ottaviani [21]	Post-quantum cryptography performance.	This study provides an RSA baseline for future cryptographic migration.
This study	RSA performance across 1024, 2048, 3072, and 4096-bit keys.	Provides a controlled operation-by-operation benchmark of RSA trade-offs.

IV. RESULTS

This section presents the experimental results of the RSA benchmark. The experiment evaluated RSA key generation, encryption, decryption, digital signature generation, signature verification, and output-size overhead for four key sizes: 1024, 2048, 3072, and 4096 bits. The benchmark was implemented in Java using RSA-

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

OAEP with SHA-256 for encryption/decryption and RSASSA-PSS with SHA-256 for signing/verification, consistent with the methodology of this study. To provide a deeper statistical analysis, the results are reported using mean, median, standard deviation (SD), minimum, maximum, and relative slowdown ratio. While the means show the average execution time, the median provides a more stable measure of central tendency in the presence of outliers. The SD and min–max range indicate the variability of repeated measurements.

A. Key Generation Performance

RSA key generation showed the greatest computational cost and the highest variability among all measured operations. This is expected because RSA key generation depends on probabilistic prime-number generation, which can lead to fluctuations in execution time. In this benchmark, key generation was measured with fewer iterations than the other operations because larger key sizes required significantly more computation time.

Table 3 Statistical Summary of RSA Key Generation Time

Key Size	N	Mean (ms)	Median (ms)	SD (ms)	Min (ms)	Max (ms)	Slowdown
1024	3	94.694	94.766	13.687	80.970	108.344	1.00×
2048	3	58.024	36.900	36.845	36.604	100.568	0.61×
3072	3	699.632	402.783	666.036	233.634	1462.478	7.39×
4096	3	732.365	755.601	239.344	482.250	959.244	7.73×

The results show that key generation time increased substantially for 3072-bit and 4096-bit keys. Although the 2048-bit mean was lower than the 1024-bit mean in this short run, this inconsistency is likely due to the small number of measured iterations and the probabilistic nature of RSA key generation. The very large SD values for 3072-bit and 4096-bit keys confirm that key generation times were highly variable. Therefore, these results suggest a strong increase in computational cost for larger key sizes, but they should be interpreted with caution.

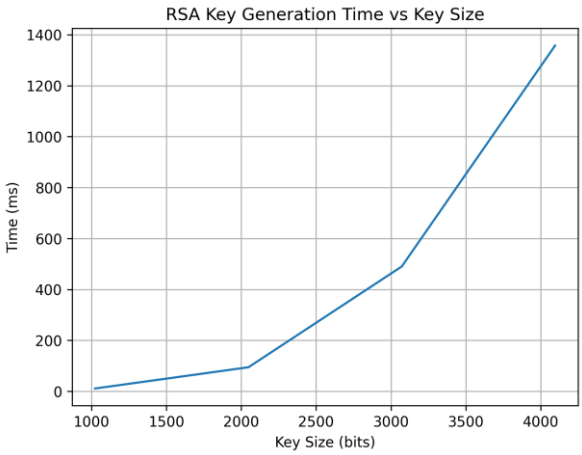

Fig. 1. RSA Key Generation Time vs Key Size.

This figure shows the increase in RSA key generation time as the key size grows

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

from 1024 to 4096 bits. The graph highlights the substantial rise in computational cost for larger RSA keys.

B. Private-Key Operations: Decryption and Signing

Private-key operations demonstrated a clear increase in execution time as the key size increased. In RSA, decryption and digital signing are computationally expensive because they require modular exponentiation with large private exponents. As a result, these operations are much more sensitive to key-size growth than public-key operations.

Table 4 Statistical Summary of RSA Private-Key Operations

Operation	Key Size	N	Mean (ms)	Median (ms)	SD (ms)	Min (ms)	Max (ms)	Slowdown
Decryption	1024	200	0.966	0.247	6.963	0.170	71.654	1.00×
Decryption	2048	200	0.982	0.886	0.228	0.826	3.290	1.02×
Decryption	3072	200	3.071	2.523	4.804	2.387	50.629	3.18×
Decryption	4096	200	5.526	5.445	0.340	5.237	9.008	5.72×
Signing	1024	200	0.620	0.233	5.092	0.179	72.269	1.00×
Signing	2048	200	1.000	0.917	0.156	0.851	1.570	1.61×
Signing	3072	200	2.936	2.585	3.516	2.452	50.726	4.73×
Signing	4096	200	5.621	5.540	0.454	5.350	10.654	9.06×

The decryption results show that the median execution time increased from 0.247 ms at 1024 bits to 5.445 ms at 4096 bits. Similarly, the median signing time increased from 0.233 ms at 1024 bits to 5.540 ms at 4096 bits. Based on the mean values, 4096-bit decryption was approximately 5.72× slower than 1024-bit decryption, while 4096-bit signing was approximately 9.06× slower than 1024-bit signing.

The large differences between mean and median for some operations, especially at 1024-bit and 3072-bit key sizes, indicate the presence of outliers. For example, the maximum values for 1024-bit decryption and signing exceeded 70 ms, while the median values remained below 0.25 ms. This suggests that occasional runtime interruptions, operating system scheduling, or Java Virtual Machine effects influenced some measurements. Therefore, the median values provide a more reliable picture of typical private-key performance.

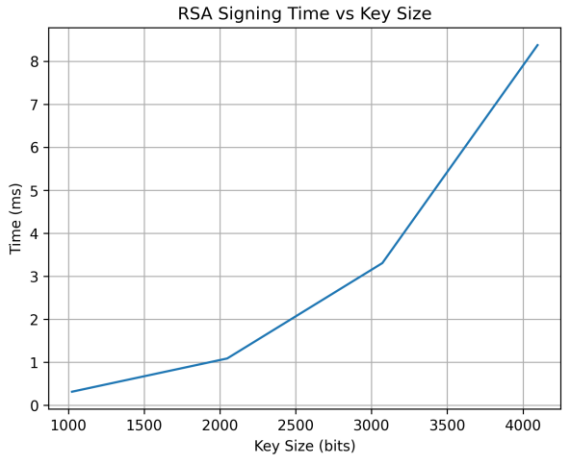

Fig. 2. RSA Decryption Time vs Key Size.
This figure shows the growth of RSA-OAEP decryption time across different key sizes. The decryption time increases significantly as the key size becomes larger.

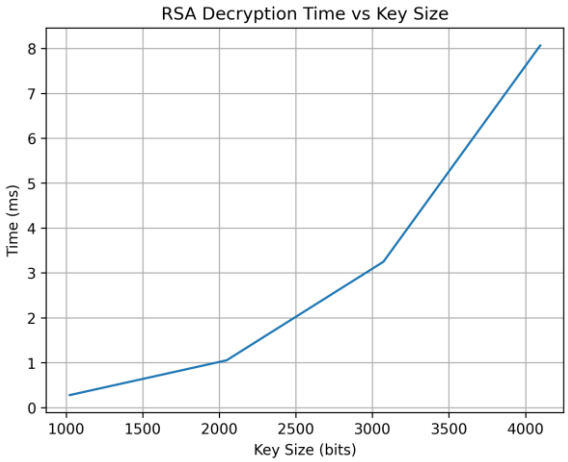

Fig. 3. RSA Signing Time vs Key Size.
This figure shows the execution time of RSASSA-PSS signing for 1024, 2048, 3072, and 4096-bit keys. The increasing trend reflects the higher cost of private-key modular exponentiation.

C. Public-Key Operations: Encryption and Verification

Public-key operations remain significantly faster than private-key operations. This is expected because RSA encryption and signature verification generally use a small public exponent, commonly 65537, which reduces the computational cost. Even when the key size increased, encryption and verification remained relatively lightweight in absolute terms.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

Table 5 Statistical Summary of RSA Public-Key Operations

Operation	Key Size	N	Mean (ms)	Median (ms)	SD (ms)	Min (ms)	Max (ms)	Slowdown
Encryption	1024	200	0.502	0.076	5.686	0.027	80.506	1.00×
Encryption	2048	200	0.071	0.060	0.036	0.038	0.276	0.14×
Encryption	3072	200	0.092	0.073	0.156	0.065	2.277	0.18×
Encryption	4096	200	0.127	0.116	0.028	0.107	0.301	0.25×
Verification	1024	200	0.408	0.037	5.094	0.018	72.090	1.00×
Verification	2048	200	0.052	0.042	0.034	0.033	0.393	0.13×
Verification	3072	200	0.310	0.065	3.400	0.061	48.147	0.76×
Verification	4096	200	0.112	0.108	0.013	0.102	0.206	0.27×

The encryption results show that median encryption time increased from 0.076 ms at 1024 bits to 0.116 ms at 4096 bits. Verification also increased with key size, with median verification time rising from 0.037 ms at 1024 bits to 0.108 ms at 4096 bits. Although the mean values fluctuate because of outliers, the median values show a clearer and more consistent trend.

These results confirm the asymmetric cost profile of RSA. Public-key operations are much less affected by increasing key size than private-key operations. The difference between mean and median for 1024-bit encryption and verification also shows that a small number of outlier measurements had a noticeable effect on the average. Therefore, the median again provides a more stable measure of typical performance.

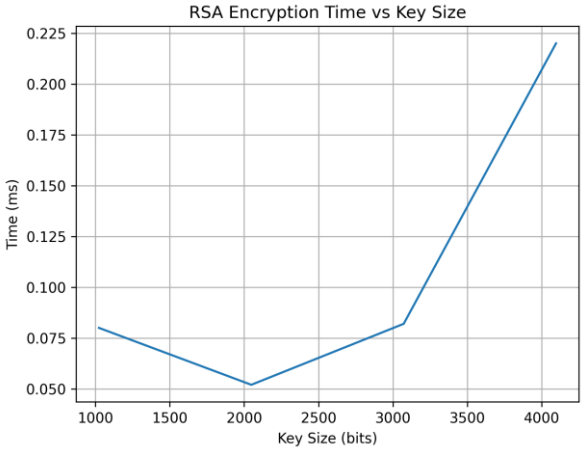

Fig. 4. RSA Encryption and Verification Time vs Key Size.
This figure compares the execution time of RSA-OAEP encryption and RSASSA-PSS verification across the four key sizes. The graph shows that public-key operations remain comparatively efficient even as the key size increases.

D. Output Size Overhead

In RSA, ciphertext and signature size are directly determined by the modulus length. Therefore, output size increased linearly as the key size increased. This result was fully consistent across all runs and showed no variability.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

Table 6 RSA Output Size by Key Length

Key Size	Output Size (bytes)	Growth Compared with 1024-bit
1024	128	1.00×
2048	256	2.00×
3072	384	3.00×
4096	512	4.00×

The results indicate a perfectly linear relationship between key length and output size. A 2048-bit key produced 256-byte ciphertexts and signatures, while a 4096-bit key produced 512-byte outputs. Although the per-operation increase in output size may appear modest, this overhead can become important in large-scale systems, such as secure messaging platforms, digital certificate infrastructures, and high-volume authentication services.

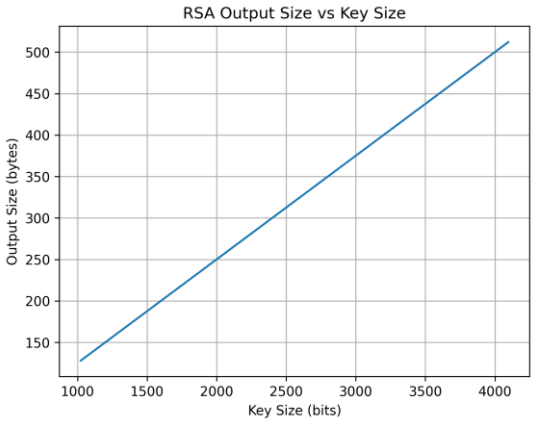

Fig. 5. RSA Output Size vs Key Size.
This figure illustrates the linear growth of RSA ciphertext and signature size as the modulus size increases from 1024 bits to 4096 bits.

E. Overall Statistical Interpretation

The statistical analysis highlights three main findings. First, RSA key generation showed the greatest computational cost and the highest variability, mainly because of the probabilistic process of prime generation. Second, private-key operations were substantially more affected by key-size growth than public-key operations. Both decryption and signing showed a strong upward trend in execution time as the modulus size increased. Third, public-key operations remained relatively efficient, confirming that encryption and verification impose lower computational overhead in typical RSA deployments.

The results also demonstrate the importance of using deeper statistical measures rather than relying only on the means. In several cases, the maximum values were much larger than the median values, which indicates the presence of outliers. These outliers inflated the mean and SD, making the median a more reliable indicator of typical performance. This supports the need for reporting multiple statistical measures when evaluating cryptographic performance.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

Overall, the findings confirm that increasing RSA key length improves the security margin but also increases computational and resources overhead. Among the tested configurations, **RSA-2048** remained relatively efficient and practical for many applications, while **RSA-3072** and **RSA-4096** introduced significantly higher costs, especially for key generation and private-key operations. Therefore, larger RSA keys should be selected carefully in environments where performance and resource consumption are important considerations.

V. DISCUSSION

The experimental results confirm the asymmetric computational cost profile of RSA, a characteristic rooted in the mathematical structure of the algorithm. In RSA, public-key operations typically use a small public exponent, commonly 65537, allowing encryption and signature verification to be performed relatively efficiently. In contrast, private-key operations—including decryption and digital signing—require modular exponentiation with large private exponents, making them computationally expensive. As demonstrated by the empirical results of this study, the computational burden grows significantly as key length increases. In particular, private-key operations scale substantially worse than public-key operations, exhibiting superlinear performance degradation as modulus size grows.

For systems where servers perform frequent private-key operations—such as TLS termination servers, certificate authorities, authentication gateways, and document-signing services, this asymmetry has practical implications. Increasing the RSA modulus from 2048 bits to 4096 bits can introduce multi-millisecond per-operation costs, which may accumulate under high workloads. In large-scale deployments handling thousands or millions of cryptographic operations per second, such latency increases can translate into measurable throughput reductions, increased CPU utilization, and wider tail latency distributions. These performance effects are particularly relevant in cloud services, microservice architectures, and latency-sensitive environments where cryptographic operations occur frequently as part of authentication and secure session establishment.

The observed scaling behavior of RSA key generation also highlights another operational challenge. Generating large RSA keys requires finding large prime numbers and performing multiple primality tests, which becomes increasingly time-consuming as key size increases. Although key generation is typically performed less frequently than encryption or signing operations, it remains relevant in contexts such as certificate issuance, ephemeral key generation, and automated infrastructure provisioning. The experimental results demonstrate that key generation cost increases dramatically for larger keys, reinforcing the importance of balancing cryptographic strength with operational feasibility.

Beyond performance considerations, these findings have implications for security policy and cryptographic parameter selection. Contemporary cryptographic guidance emphasizes selecting parameters that provide sufficient security while maintaining acceptable operational efficiency. Key-management recommendations and digital signature standards published by the National Institute of Standards and Technology remain central references for defining acceptable cryptographic strengths and approved algorithm suites [1], [5]. In practice, many modern systems have converged

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

on RSA-2048 as a widely accepted baseline security level that balances strong protection against classical attacks with manageable computational cost.

However, security policy must also consider long-term threats and evolving attack capabilities. Organizations responsible for protecting sensitive or long-lived data are increasingly encouraged to adopt crypto-agility—the ability to rapidly update or replace cryptographic algorithms as new threats emerge. This requirement is particularly important in the context of post-quantum cryptography. Government agencies and security organizations have begun advising institutions to plan for the eventual transition to quantum-resistant cryptographic algorithms, especially for systems protecting long-term secrets or archival data [13], [15]. The so-called "harvest now, decrypt later" threat model highlights the risk that encrypted data captured today could be decrypted in the future once quantum computing capabilities mature.

Within this broader strategic context, simply increasing RSA key length provides only incremental protection against classical factorization attacks and does not address the fundamental vulnerability of RSA to large-scale quantum computers. Algorithms such as Shor's algorithm could break RSA regardless of key size once sufficiently powerful quantum hardware becomes available. Consequently, while increasing key sizes may strengthen classical security margins, it does not represent a long-term solution to emerging cryptographic threats. This reality has motivated ongoing research and standardization efforts around post-quantum cryptography and hybrid cryptographic deployments [13], [20], [21].

Taken together, the results of this study highlight the importance of understanding both the performance costs and the security benefits associated with increasing RSA key lengths. Organizations must carefully evaluate whether larger keys meaningfully improve their security posture relative to the operational overhead they introduce. In many scenarios, improving system architecture, adopting more efficient cryptographic primitives, or preparing for hybrid post-quantum deployments may provide greater long-term benefits than simply scaling RSA key sizes.

VII. CONCLUSION

This study presented a controlled experimental evaluation of Rivest–Shamir–Adleman **(RSA)** performance across 1024, 2048, 3072, and 4096-bit key sizes. The novelty of this research lies in its operation-by-operation measurement of RSA key-length trade-offs using a Java-based benchmarking framework with RSA-OAEP and RSASSA-PSS constructions. Unlike studies that mainly discuss RSA security or theoretical complexity, this work provides practical statistical evidence on how key size affects key generation, encryption, decryption, signing, verification, and output size.

The main contribution of the research is the identification of where RSA performance degradation becomes practically significant. The results show that private-key operations, especially decryption and digital signing, are much more affected by increasing key length than public-key operations. Key generation also becomes increasingly costly and variable as modulus size grows. In contrast, encryption and signature verification remain comparatively efficient because they usually rely on a small public exponent.

The findings suggest that RSA-2048 remains a practical baseline for many contemporary systems because it provides a reasonable balance between security and

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

performance. RSA-3072 may be suitable for applications requiring a longer security lifetime, while RSA-4096 should be used carefully in performance-sensitive environments due to its higher computational overhead. Therefore, RSA key-size selection should be based not only on security requirements but also on workload, latency, and resource constraints.

Overall, this research contributes empirical evidence that can help system designers, security engineers, and policymakers make more informed decisions about RSA deployment. Future work may extend this study by testing multiple hardware platforms, comparing Java results with native cryptographic libraries such as OpenSSL, measuring energy and memory consumption, and evaluating RSA alongside elliptic-curve and post-quantum cryptographic algorithms.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

References

[1] NIST, “Recommendation for Key Management: Part 1 - General (SP 800-57 Part 1 Rev. 5),” May 2020.

[2] NIST, “NIST Publishes SP 800-57 Part 1, Revision 5,” May 4, 2020.

[3] MIT CSAIL, “6.857 Network and Computer Security: Recitation 5 - RSA, OAEP, and CRT,” Mar. 2020.

[4] K. Fossati, “Recommendations for Secure Use of TLS and DTLS,” RFC 9325, Nov. 2022.

[5] NIST, “FIPS 186-5: Digital Signature Standard (DSS),” Feb. 2023.

[6] Federal Register, “Announcing Issuance of FIPS 186-5, Digital Signature Standard,” Feb. 3, 2023.

[7] Oracle, “JDK Providers Documentation (Java SE 24): SunRsaSign Provider and Algorithms,” 2024.

[8] Oracle, “Java Security Standard Algorithm Names Specification (Java SE 25),” 2025.

[9] Oracle, “Security Developer’s Guide (Java SE 24),” 2024.

[10] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, Introduction to Algorithms, 4th ed. MIT Press, 2022.

[11] D. Boneh and V. Shoup, A Graduate Course in Applied Cryptography, Version 0.6, Jan. 2023.

[12] C. Paar, J. Pelzl, and T. Güneysu, Understanding Cryptography: From Established Symmetric and Asymmetric Ciphers to Post-Quantum Algorithms, 2nd ed. Springer, 2024.

[13] NIST, “NIST IR 8547 (Initial Public Draft): Transition to Post-Quantum Cryptography Standards,” Nov. 2024.

[14] National Quantum Initiative, “NIST Releases Draft Report on Transition to Post-Quantum Cryptography Standards,” Nov. 12, 2024.

[15] CISA/NSA/NIST, “Quantum-Readiness: Migration to Post-Quantum Cryptography,” Aug. 2023.

[16] A. Desai, V. Parekh, U. Unadkat, and N. Shekokar, “Performance Analysis of Various Asymmetric Public-Key Cryptosystem,” in Pervasive Computing and Social Networking (LNNS), Springer, 2022 (First Online).

[17] A. Parihar and S. Nakhate, “Low latency high throughput Montgomery modular multiplier for RSA cryptosystem,” Engineering Science and Technology, an International Journal, vol. 30, Jun. 2022, Art. no. 101045.

[18] J.-J. Liu, K.-T. Tsang, and Y.-H. Deng, “A Variant RSA Acceleration with Parallelization,” arXiv:2111.11924, Nov. 2021.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026

[19] APNIC Labs, “Measurement of DNSSEC Validation with RSA-4096,” presentation, Nov. 2021.

[20] P. Kampanakis, “The impact of data-heavy, post-quantum TLS 1.3 on the Time-To-Last-Byte of real-world connections,” NIST Fifth PQC Standardization Conference, 2024.

[21] G. Fitzgibbon and C. Ottaviani, “Constrained Device Performance Benchmarking with the Implementation of Post-Quantum Cryptography,” Cryptography, vol. 8, no. 2, 2024.

[22] J. Montes et al., “A performance evaluation framework for post-quantum TLS,” Computers & Security (ScienceDirect), 2025.

Kateb Scientific-Research Journal of Technology and Engineering
Vol. 1 – No. 1, Autumn & Winter 2025–2026