Journal of Computer Science

Research Article

Unveiling the Performance: A Comparative Study of HMAC and RSA Encryption on SHA-512

Louis Leonardo and Rojali

Department of Computer Science, BINUS Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia

Article history
Received: 22-08-2024
Revised: 14-01-2025
Accepted: 06-03-2025

Corresponding Author:
Louis Leonardo
Department of Computer Science, BINUS Graduate Program-Master of Computer Science, Bina Nusantara University, Jakarta, Indonesia 11480, Indonesia
Email: louis.leonardo@binus.ac.id

Abstract: The study evaluates critical operations including token signing, verification, and decoding, as well as data transfer efficiency, utilizing the jsonwebtoken library. By benchmarking these operations, we provide insights into the trade-offs between symmetric (HMAC) and asymmetric (RSA) encryption methods, highlighting their impact on performance and security. The results serve as a guideline for developers to make informed decisions when choosing JWT algorithms for their Node.js applications, balancing security needs and application performance. Our analysis revealed that HMAC-SHA512 consistently outperforms RSA-SHA512 in terms of token creation speed, data transfer efficiency, and token size. Specifically, HMAC-SHA512 demonstrated faster token generation times, achieving rates up to ten times higher than RSA-SHA512, and produced tokens that were 256 bytes smaller on average. Furthermore, HMAC-SHA512 maintained superior data transfer rates, reaching up to 42.601 MB/s, compared to RSA-SHA512's maximum of 9.822 MB/s. Despite RSA-SHA512's robust performance in token verification and decoding, HMAC-SHA512's overall efficiency and consistency make it the preferred choice for applications requiring rapid processing and minimal latency. This study underscores the potential of HMAC-SHA512 to enhance performance in data-intensive environments, highlighting its advantages in speed and reduced data payload.

Keywords: JSON Web Token, JWT, HMAC-SHA512, RSA-SHA512, NodeJS, Encryption Performance, Data Transfer

Introduction

Web services as business processes within a platform remain popular today. Most of these services utilize the HTTP protocol, which is stateless. This means that each request made by an application is executed independently and is not related to previous requests, thus allowing the server to avoid managing prior client requests (Ady Kusuma, 2021). Web services are designed to connect various computer systems, enabling machines to interact with one another through a network (Frisca *et al*., 2023). REST is the most popular web service. This is supported by polling data indicating that 58% of IT professionals find that web services other than REST are more complex and costly. However one of the drawbacks of REST is latency, which refers to the delay in request initiation and data transfer (Halili and Ramadani, 2018).

Authentication in REST uses the OAuth method to validate transmitted data. The commonly employed OAuth method involves using JSON Web Token (JWT) (Prayogi *et al*., 2020). A JWT is a token composed of three parts: Header, Payload, and Signature, each separated by a dot. Before encrypting with JWT, it is crucial to choose the appropriate digital signature. As indicated in Table 1, each digital signature has different SHA sizes. A larger SHA number implies greater resistance to cryptographic attacks due to the larger SHA space. However, a larger SHA size also means greater resource requirements.

When evaluating HMAC (Hash-based Message Authentication Code) and RSA (Rivest-Shamir-Adleman) for JWT (JSON Web Token) implementation, one of the biggest performance challenges is the difference in computational efficiency between the two algorithms. HMAC, being a symmetric encryption algorithm, is designed for rapid processing, leveraging relatively straightforward mathematical operations that allow for quick token signing and verification. HMAC is one of the most well-known cryptographic algorithms and is used to ensure that stored or exchanged data is not altered (intentionally or unintentionally) without authorization (Backendal *et al*., 2023). This performance advantage is particularly beneficial in high-throughput environments where fast token validation is crucial. However, the need for secure key management presents a secondary challenge, as the performance benefit is only realized if the shared secret key is handled and stored securely across all interacting systems.

In contrast, RSA, an asymmetric encryption algorithm, involves more complex mathematical operations due to the use of large prime numbers for encryption and decryption (Luo *et al*., 2023). This complexity results in slower performance compared to HMAC, as both signing and verification processes require more computational resources. While RSA offers enhanced security by separating the signing and verification keys, the increased processing time can be a significant drawback for applications that require low latency and high performance. Thus, the challenge lies in balancing the security advantages of RSA with its performance costs, making it essential for developers to carefully assess their application's performance requirements and choose the algorithm that best fits their needs.

Choosing between RSA and HMAC for performance is crucial because it directly impacts the efficiency and scalability of an application. RSA's computationally intensive operations can introduce latency, potentially affecting user experience and system responsiveness, especially in high-load scenarios where frequent token processing occurs. On the other hand, HMAC's superior speed and efficiency can enhance application performance, making it a preferable choice for environments demanding rapid token validation and minimal delay.

While comparative studies of HMAC and RSA encryption algorithms have been previously conducted, this study advances the state of the art in several key ways. First, it evaluates these algorithms with SHA-512, a cryptographic hash function offering enhanced security compared to SHA-256, which is commonly used in similar studies. This investigation, therefore, provides new insights into the performance implications of a more robust hashing approach. Additionally, by testing across diverse, high-volume datasets, including Amazon Prime Users, Credit Score Users, and Netflix Movies, this study simulates real-world data complexity and transfer scenarios, making the findings directly applicable to high-throughput, data-intensive applications. Finally, by analyzing detailed metrics such as token creation and verification speed, data transfer rates, and token size differences, this research offers specific, practical recommendations for developers. For example, HMAC-SHA512's reduced token size and faster data transfer rates suggest its suitability for applications where bandwidth and processing speed are critical, while RSA-SHA512's stable verification performance may be preferred in security-sensitive contexts. These contributions make this study a valuable resource for developers and researchers seeking efficient, scalable solutions in data authentication and encryption.

Related Work

Recent studies in cryptographic performance analysis have emphasized the need to balance security and efficiency, particularly in the context of web and mobile applications. For instance, recent research has explored the comparative performance of HMAC and RSA with varying SHA hash sizes, though much of this work has centered around SHA-256, it demonstrated that while HMAC generally outperforms RSA in token creation speed, RSA's security advantages remain valuable in certain applications. Additionally, recent investigations into cryptographic algorithms for JSON Web Tokens (JWTs) have highlighted performance trade-offs that depend on factors such as data complexity, token size, and processing frequency, indicating a need for targeted algorithm selection based on specific application requirements.

Performance testing was conducted for HMAC and RSA using SHA-256 and SHA-512. According to Table 1, HMAC showed superior performance in token creation and verification compared to RSA when using both SHA-256 and SHA-512 (Ramadhoni and Santoso, 2023). This testing was performed using the National Standard Open Payment API (SNAP), which further supports the superior performance of HMAC over RSA. However, research found that RSA outperformed HMAC with SHA-256 in terms of encryption and decryption performance for image objects (Kiran and Harini, 2018).

Table 1: RSA and HMAC Performance on Create and Verify with SHA256

Type	Algorithm	Average (ms)	Received KB/sec	Sent KB/sec
Create	HMAC256	1.5	1.38	3.22
Create	RSA256	8	3.72	3.21
Verify	HMAC256	1.5	2.14	2.96
Verify	RSA256	1	2.14	5.29

JWT is the optimal method (Ramadhoni and Santoso, 2023). Other findings indicate that SHA-512 outperforms SHA-256 in performance when using the HMAC algorithm. However, the selection of digital signatures is still predominantly dominated by HMAC + SHA-256, RSASSA-PKCS1-v1_5 + SHA-256, or ECDSA + P-256 + SHA-256, and unfortunately, research on the performance of digital signature algorithms remains limited. The choice of the appropriate digital signature is crucial as it can enhance data transfer performance during client-server requests. Therefore, the researcher is interested in conducting a comparative test of HMAC and RSA algorithms with SHA-512. The study will use three variables: creation speed, verification speed, and data transfer speed from client to server, aiming to provide valuable insights for selecting digital signatures and improving efficiency in data authentication using JWT.

Methods

The proposed method includes several steps, from Data Collection, Preprocessing, Loading the dataset into jsonwebtoken library, and evaluation.

The experimental setup for this study was designed to rigorously evaluate the performance of HMAC and RSA encryption algorithms using SHA-512. We implemented HMAC-SHA512 and RSA-SHA512 to reflect industry standards and common cryptographic practices. Each algorithm's performance was assessed based on metrics that include token creation time, token verification speed, data transfer rate, and token size, ensuring a comprehensive understanding of the trade-offs between speed, efficiency, and security. Testing was conducted using the JSON Web Token (JWT) library within a Node.js environment, and each dataset was formatted as a JSON payload. The datasets used were sourced from high-volume sources, such as Amazon Prime Users, Credit Score Users, and Netflix Movies, to simulate varied and realistic application conditions. Each dataset was processed multiple times to ensure consistency in results, and performance measurements, including request-response times, were averaged across trials to account for any fluctuations in processing speed or network variability. This detailed setup allows for accurate performance comparisons and provides a robust framework for assessing HMAC and RSA's efficiency and scalability in data-intensive applications.

Data Collections

The dataset was collected from Kaggle. There are 3 dataset that will be used for the performance analysis. Amazon Prime User, Credit Score User, and Netflix Movies titles. Each dataset has thousands of records and later will be used for data transfer performance analysis, and record will be used for sign, decode, verify and size analysis.

Preprocessing

Before the data is used for performance analysis, the data row will be put as payload in JSON format. Later the payload JSON format can be used to do sign token, decode token, and verify token.

This hardware configuration features a mid-range performance setup ideal for testing cryptographic algorithms like RSA and HMAC with SHA-512. With 16GB of RAM, it offers sufficient memory for handling intensive computations. The 11th Gen Intel Core i5-1135G7 processor, with 4 cores and 8 threads running at 2.40GHz, is capable of managing computationally demanding tasks, making it a good choice for cryptographic performance testing. Running on Windows, this setup has access to a broad selection of cryptographic tools and libraries, providing a solid foundation for benchmarking RSA and HMAC with SHA-512 performance. Added that there are 2 libraries to be used during processing the data from NPM since the test will performed with javascript on expressjs, jsonwebtoken and benchmarkify. jsonwebtoken is a popular package that has been used to sign, verify, and even to decode the data of token. While benchmarkify is used to evaluate the performance.

Results and Discussion

The results reveal distinct performance patterns between HMAC-SHA512 and RSA-SHA512, which can be attributed to each algorithm's structural differences and computational demands. HMAC, as a symmetric algorithm, requires a single shared secret key and performs lighter hashing operations, allowing it to generate and verify tokens more quickly, particularly with larger datasets where rapid processing is critical. In contrast, RSA relies on asymmetric encryption, with separate public and private keys, which introduces additional computational overhead, particularly during the signing process. This structural complexity in RSA, though beneficial for security, leads to comparatively slower performance. Additionally, key sizes impact each algorithm differently. RSA's use of larger key sizes (e.g., 2048 bits) enhances security but further increases processing time, whereas HMAC maintains both speed and efficiency with smaller keys (e.g., 256 bits), balancing security with faster token creation and verification times. The superior data transfer rates observed with HMAC-SHA512 reflect its compact token size and minimal resource demands, making it a preferred choice for high-throughput applications. Conversely, RSA-SHA512's relatively stable verification performance across datasets makes it advantageous in

scenarios where consistent security is prioritized over speed. This in-depth analysis underscores the importance of selecting algorithms based on specific application requirements, balancing speed, security, and data handling needs.

The performance of token creation with the RSA-SHA512 and HMAC-SHA512 algorithms was tested on three different datasets: Amazon Prime Users, Credit Score Users, and Netflix Movies. Performance measurements were taken by recording duration, cycles, counts, average times, and rate per second (RPS). For the RSA-SHA512 algorithm, the average time to create a token on the Amazon Prime Users dataset ranged from 0.001064 to 0.001148 seconds, with an RPS ranging from 870 to 940. The Credit Score Users dataset showed similar results, with average times between 0.001079 and 0.001144 seconds and an RPS ranging from 874 to 927.

On the Netflix Movies dataset, the RSA-SHA512 algorithm exhibited a wider range of performance, with average times between 0.001052 and 0.001262 seconds and an RPS ranging from 792 to 950.

Testing on Fig. 1 with the HMAC-SHA512 algorithm yielded better results compared to RSA-SHA512. On the Amazon Prime Users dataset, the average time to create a token with HMAC-SHA512 ranged from 0.000342 to 0.000423 seconds, with an RPS ranging from 2361 to 2922. The Credit Score Users dataset also showed similar performance, with average times between 0.000346 and 0.000375 seconds and on Fig. 2 shows an RPS ranging from 2668 to 2910. This indicates that HMAC-SHA512 outperforms RSA-SHA512 in terms of token creation speed.

For the Netflix Movies dataset with HMAC-SHA512, performance on Fig. 1 was also better compared to RSA-SHA512, with average times between 0.000346 and 0.000423 seconds and Fig. 2 displays an RPS ranging from 2361 to 2922. This suggests that HMAC-SHA512 is more efficient in token creation on datasets with varying structures and sizes. The difference in performance can be attributed to the cryptographic mechanisms used by each algorithm, where HMAC-SHA512 is simpler and faster in the hashing process compared to RSA-SHA512.

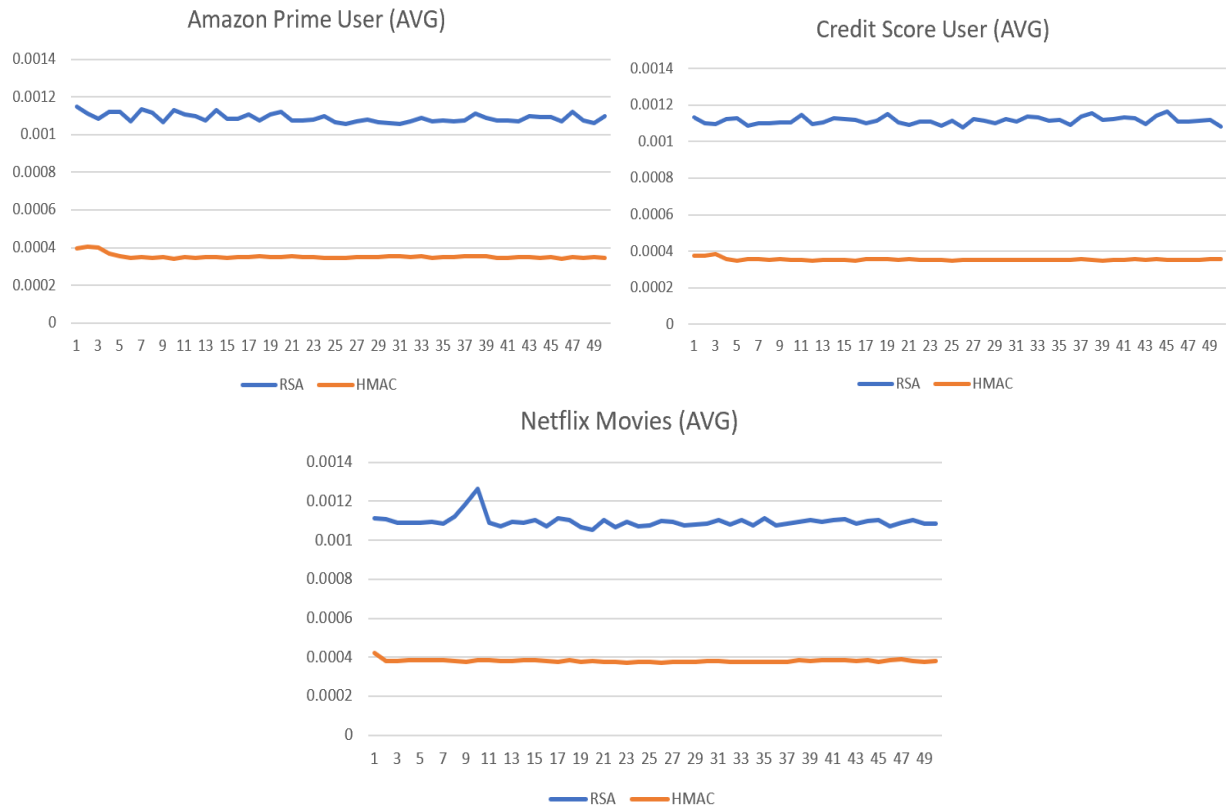

Fig. 1: Average performance on token creation

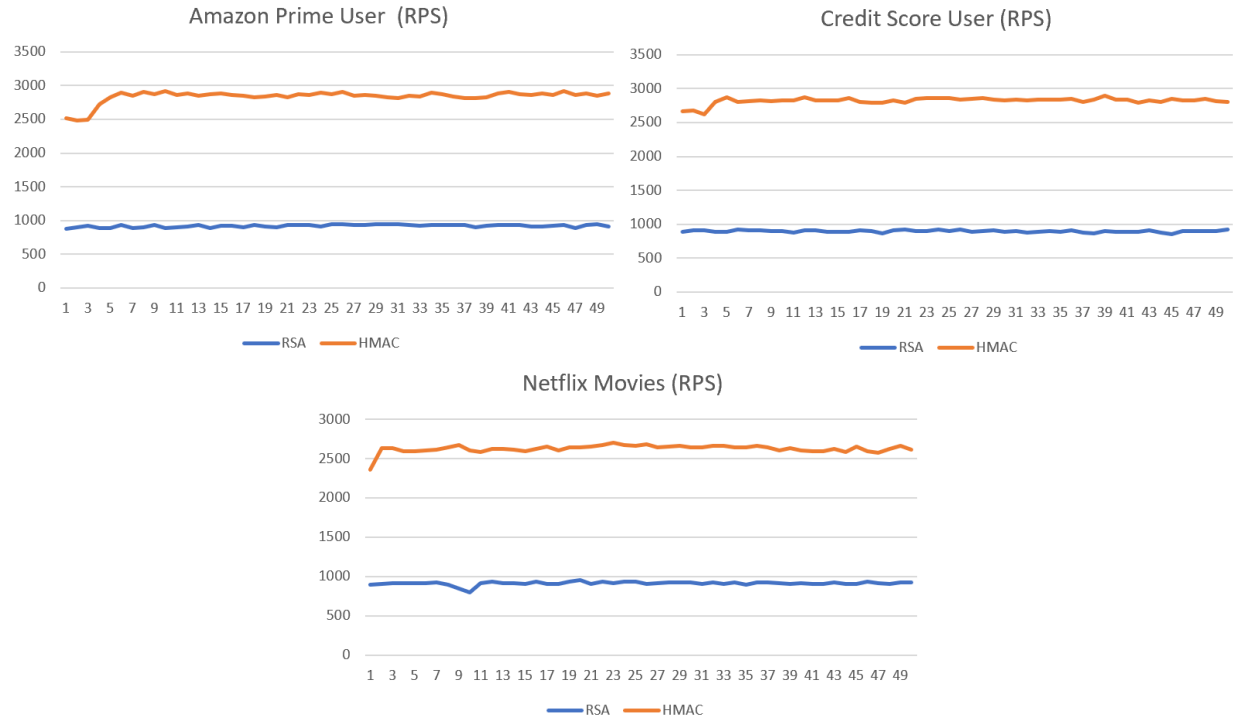

Fig. 2: RPS performance on token creation

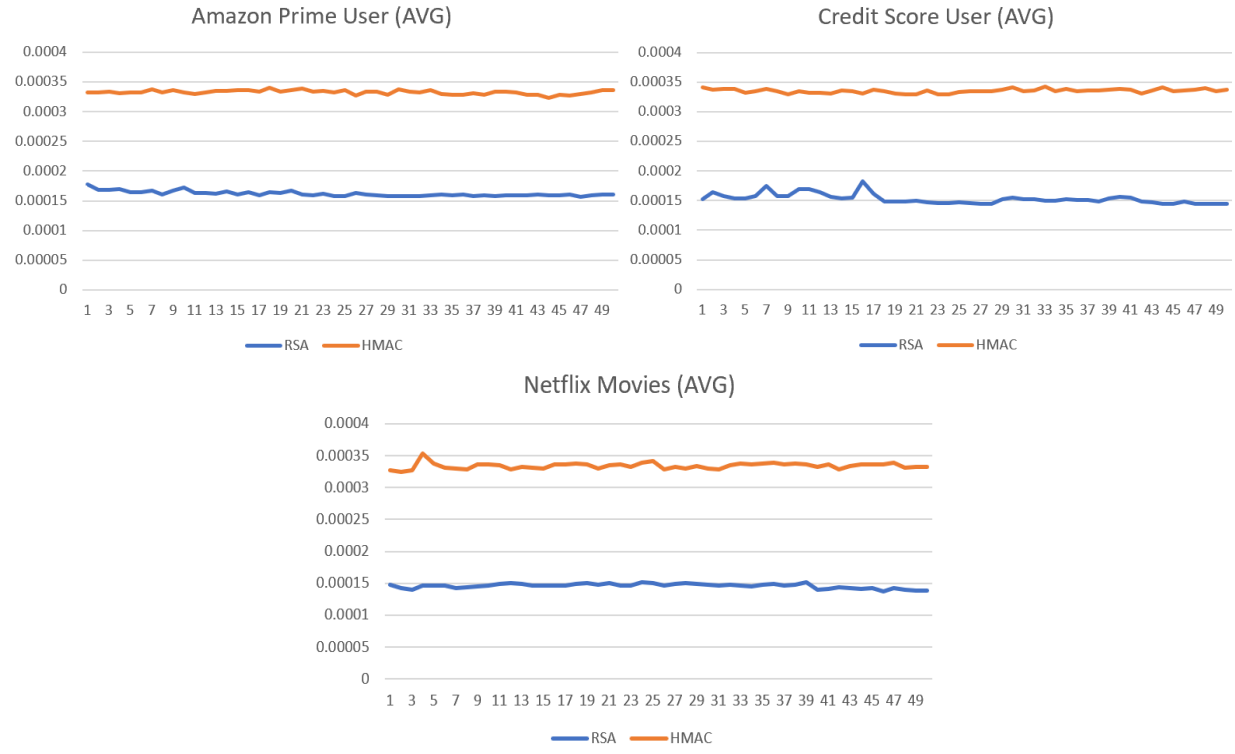

Fig. 3: Average performance on token verification

In the performance testing of token verification using the RSA-SHA512 algorithm in Fig. 3, the Amazon Prime User data showed an average verification time ranging from 0.000157 to 0.000172 seconds. The highest average RPS in Fig. 4 (requests per second) reached 6348.098, while the lowest was 5623.051. These results indicate that the RSA-SHA512 algorithm can handle a substantial verification load with relatively stable times. Variations in RPS suggest some performance fluctuations but remain within acceptable limits.

For the Credit Score User data displayed in Fig. 3, token verification with RSA-SHA512 resulted in an average verification time of approximately 0.000146 to 0.000164 seconds. In Fig. 4 the highest recorded average RPS was 6871.653, while the lowest was 5455.781. This performance is relatively consistent with previous data, demonstrating the stability of RSA-SHA512 in handling token verification across different datasets. Variations in RPS were observed but did not significantly impact overall performance.

The Netflix Movies data in Fig. 3 showed slightly better performance, with average verification times ranging from 0.00014 to 0.000149 seconds. Fig. 4 shows the highest average RPS achieved was 7151.007, while the lowest was 6667.789. This indicates that RSA-SHA512 can enhance efficiency under certain conditions, providing more optimal results compared to the previous two datasets. Performance variations were still present but remained within acceptable ranges.

In testing with HMAC-SHA512 for the Amazon Prime User data, the average verification time ranged from 0.000327 to 0.000342 seconds in Fig. 3. The highest average RPS reached 3073.475, while the lowest was 2836.013. HMAC-SHA512 demonstrated good stability in handling token verification with consistent performance in Fig. 4. However, the verification time was slower compared to RSA-SHA512, though still within efficient limits.

Finally, for the Netflix Movies data, token verification with HMAC-SHA512 showed stable performance with average verification times between 0.000327 and 0.000342 seconds. The highest recorded average RPS was 3073.475, while the lowest was 2836.013. The consistency in HMAC-SHA512's performance indicates its capability to handle token verification effectively, even though with higher verification times compared to RSA-SHA512, it still delivers reliable and stable results.

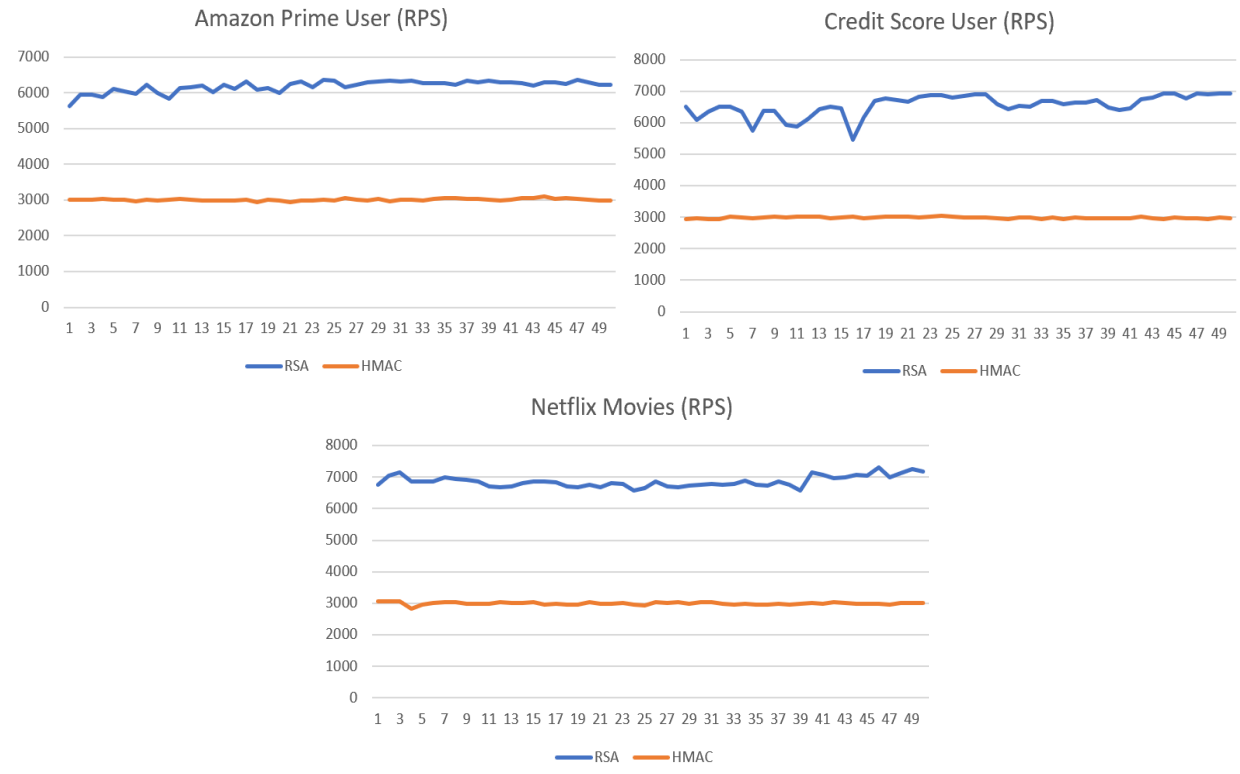

Fig. 4: RPS performance on token verification

DOI: 10.3844/jcssp.2025.2940.2950

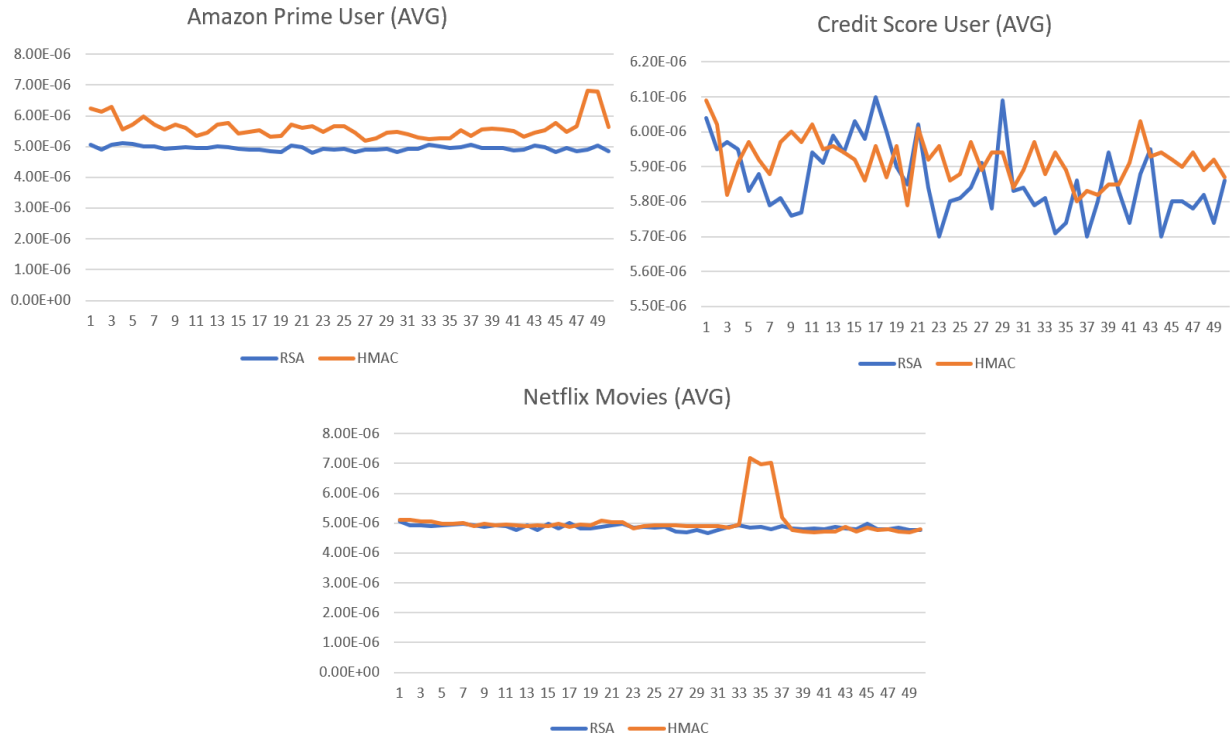

Fig. 5: Average performance on decoding token

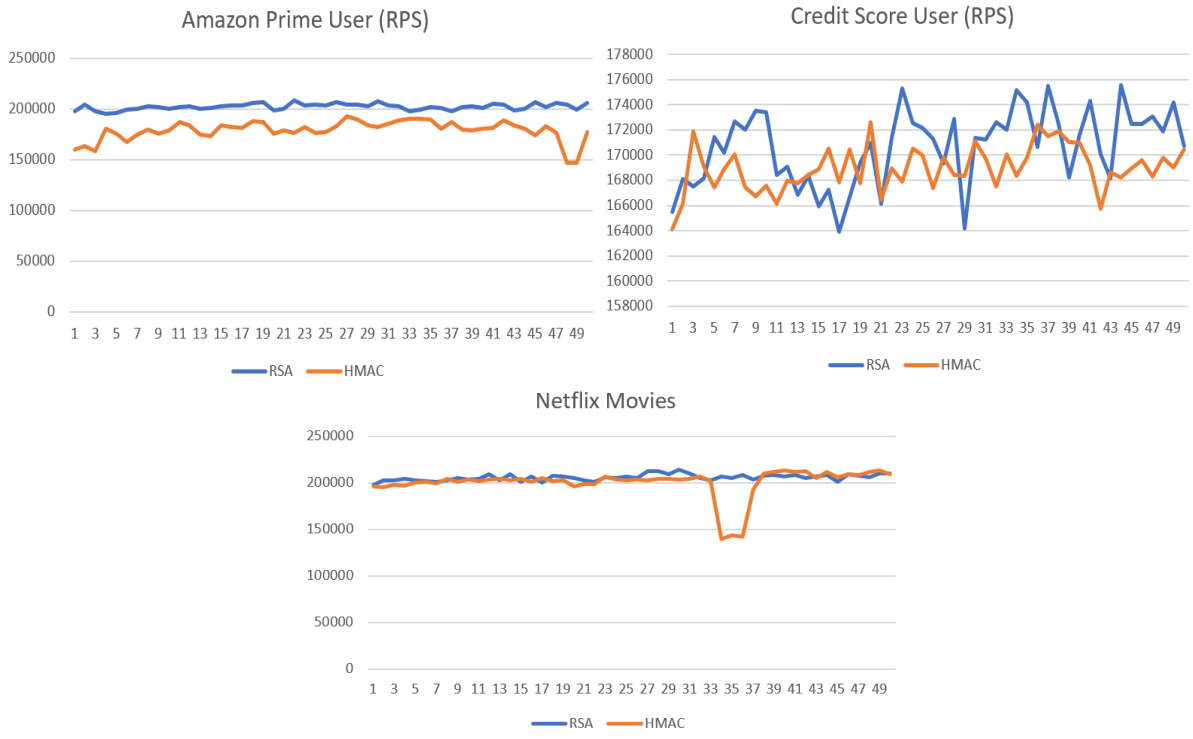

Fig. 6: RPS performance on decoding token

The performance of token decoding using the RSA-SHA512 algorithm on the Amazon Prime User data showed variable results. The average duration for the decoding process ranged from 4.9E-06 to 5.06E-06 seconds, with the number of cycles ranging from 988 to 1043 in Fig. 5. The recorded rate per second (RPS) ranged from 195558.7 to 208501.7. These results in Fig. 6 reflect the consistent performance of RSA-SHA512 in handling varying workloads, with some fluctuations possibly caused by data variability and algorithm efficiency in specific cycles.

For the Credit Score User data with the RSA-SHA512 algorithm, decoding performance showed similar variability but with slightly lower RPS values compared to the Amazon Prime User data. The average duration ranged from 5.35E-06 to 6.1E-06 seconds in Fig. 5, with the number of cycles between 820 and 968. The recorded RPS in Fig. 6 ranged from 163898.4 to 186756.7, indicating that the algorithm's performance remained fairly consistent despite the increased workload.

The decoding performance for the Netflix Movies data using the RSA-SHA512 algorithm showed relatively stable results in Fig. 5, with average durations between 4.71E-06 and 5.05E-06 seconds. The recorded cycles in Fig. 6 ranged from 1001 to 1071, with RPS values between 200068.2 and 214199.2. This indicates that the RSA-SHA512 algorithm can handle high-volume data, such as that of Netflix Movies, with fairly good efficiency.

When using the HMAC-SHA512 algorithm for the Amazon Prime User data, the token decoding performance showed greater variability in Fig. 5. The average duration ranged from 5.18E-06 to 6.82E-06 seconds, with the number of cycles between 734 and 966. The recorded RPS ranged from 146706 to 193204.4. This suggests that HMAC-SHA512 offers competitive performance but tends to be more variable compared to RSA-SHA512 in Fig. 6.

For the Credit Score User data with HMAC-SHA512 in Fig. 5, the average decoding duration ranged from 5.79E-06 to 6.09E-06 seconds, with the number of cycles between 830 and 864 based. The recorded RPS ranged from 164081 to 172633.2. This indicates that HMAC-SHA512 performs relatively steadily for medium-volume data in Fig. 6, such as Credit Score User data, with consistent efficiency despite variations in the number of cycles.

The comparison of token sizes in Table 2 using RSA and HMAC algorithms across different datasets Amazon Prime Users, Credit Score Users, and Netflix Movies highlights a significant difference in their efficiency regarding token size. For the Amazon Prime Users dataset, the RSA token size is 1099 bytes, whereas the HMAC token size is 843 bytes. This indicates that HMAC tokens are substantially smaller, reducing the size by 256 bytes. The smaller token size can enhance performance, particularly in applications where bandwidth and storage efficiency are critical.

Similarly, in the Credit Score Users dataset, RSA tokens measure 1494 bytes compared to HMAC tokens at 1238 bytes. Once again, HMAC tokens are 256 bytes smaller. This consistent size reduction suggests that HMAC is more efficient in generating compact tokens, which could lead to faster data transfer rates and reduced storage requirements. The ability to maintain smaller token sizes across different datasets further emphasizes HMAC's potential advantages in environments where minimizing data payload is crucial.

The Netflix Movies dataset shows a comparable pattern, with RSA tokens being 1107 bytes and HMAC tokens 851 bytes. The 256-byte reduction in token size when using HMAC illustrates its superiority in creating more efficient tokens. This consistency across all three datasets demonstrates that HMAC not only provides better performance in terms of speed, as previously observed, but also offers a notable advantage in token size efficiency.

The RSA algorithm in Table 3 shows a range of request-response times from 0.83482 seconds to 3.26 seconds, resulting in data transfer rates between 2.515 MB/s and 9.822 MB/s. For instance, with a data size of 8.2 MB, RSA's fastest data transfer rate is 9.822 MB/s (at a request-response time of 0.83482 seconds), while its slowest rate is 2.515 MB/s (at a request-response time of 3.26 seconds). This variation indicates that RSA's performance can fluctuate significantly depending on the conditions.

In contrast, the HMAC algorithm consistently demonstrates superior performance across the same metrics. Request-response times for HMAC range from 0.19248 seconds to 2.38 seconds, yielding data transfer rates between 3.445 MB/s and 42.601 MB/s. The highest data transfer rate for HMAC, 42.601 MB/s, is achieved at a request-response time of 0.19248 seconds, while the lowest rate, 3.445 MB/s, occurs at a request-response time of 2.38 seconds.

Table 2: Hardware and specification

Hardware	Spesification
RAM	16GB
SSD	1TB
Prosesor	11th Gen Intel Core i5-1135G7 @2.40GHz (8CPUs)
OS	Windows

Table 3: Token size comparison

Category	RSA Size (bytes)	HMAC Size (bytes)
Amazon Prime Users	1099	843
Credit Scores User	1494	1238
Netflix Movies	1107	851

The data transfer comparison between RSA and HMAC algorithms for the Credit Score User dataset in Table 4 reveals significant differences in performance across various metrics, including request-response times and data transfer rates. The RSA algorithm displays request-response times ranging from 0.83482 seconds to 3.26 seconds, resulting in data transfer rates between 2.515 MB/s and 9.822 MB/s. The highest transfer rate for RSA, 9.822 MB/s, is achieved at the shortest request-response time of 0.83482 seconds, while the lowest rate, 2.515 MB/s, corresponds to the longest request-response time of 3.26 seconds. These results indicate that RSA's performance can vary significantly based on the conditions and the data being processed.

In contrast, the HMAC algorithm demonstrates consistently superior performance. The request-response times for HMAC range from 0.19248 seconds to 2.38 seconds, with data transfer rates spanning from 3.445 MB/s to an impressive 42.601 MB/s. The highest transfer rate of 42.601 MB/s is observed at the shortest request-response time of 0.19248 seconds, whereas the lowest rate of 3.445 MB/s occurs at a request-response time of 2.38 seconds. These findings illustrate that HMAC not only performs faster on average but also maintains higher data transfer rates compared to RSA.

The performance comparison of RSA and HMAC algorithms for the Netflix Movies dataset reveals significant differences in terms of request-response times and data transfer rates in Table 5. For the RSA algorithm, request-response times range from 0.215 seconds to 5.24 seconds, with corresponding data transfer rates spanning from 1.202 MB/s to 29.302 MB/s. The highest data transfer rate of 29.302 MB/s is observed at a request-response time of 0.215 seconds, indicating RSA's potential for high-speed performance under optimal conditions. However, the lowest transfer rate of 1.202 MB/s occurs at a much longer request-response time of 5.24 seconds, highlighting RSA's variability and occasional inefficiency in handling data transfers.

On the other hand, the HMAC algorithm demonstrates consistently superior performance in Table 5. The request-response times for HMAC range from 0.182 seconds to 1.12 seconds, significantly shorter than those for RSA. Correspondingly, the data transfer rates for HMAC are notably higher, ranging from 5.625 MB/s to an impressive 34.615 MB/s. The peak transfer rate of 34.615 MB/s is achieved at the shortest request-response time of 0.182 seconds, showcasing HMAC's efficiency in data handling. Even at its lowest performance, HMAC's transfer rate of 5.625 MB/s remains competitive, further illustrating its reliability and consistency (Table 6).

Table 4: Data transfer comparison on Amazon Prime User dataset

	RSA		HMAC	
size (mb)	request-response (s)	Data Transfer (mb/s)	request-response (s)	Data Transfer (mb/s)
1.4	0.07916	17.68569985	0.06927	20.21076945
1.4	0.10124	13.82852627	0.03848	36.38253638
1.4	0.06282	22.28589621	0.04841	28.9196447
1.4	0.29146	4.803403555	0.0497	28.16901408
1.4	0.05607	24.96878901	0.05559	25.18438568
1.4	0.06917	20.23998843	0.01007	139.0268123
1.4	0.08557	16.36087414	0.0625	22.4
1.4	0.06061	23.0984986	0.05911	23.68465573
1.4	0.08557	16.36087414	0.23973	5.839903224
1.4	0.06127	22.84968174	0.04347	32.20611916

Table 5: Data transfer comparison on Credit Score User dataset

	RSA		HMAC	
size (mb)	request-response (s)	Data Transfer (mb/s)	request-response (s)	Data Transfer (mb/s)
8.2	2.01	4.07960199	1.3	6.307692308
8.2	1.77	4.632768362	0.20484	40.0312439
8.2	1.49	5.503355705	2.38	3.445378151
8.2	1.34	6.119402985	1.29	6.356589147
8.2	2.17	3.778801843	1.54	5.324675325
8.2	0.83482	9.822476702	1.21	6.776859504
8.2	2.84	2.887323944	0.81231	10.09466829
8.2	1.22	6.721311475	1.15	7.130434783
8.2	1.02	8.039215686	0.19248	42.60182876
8.2	3.26	2.515337423	1.51	5.430463576

Table 6: Data transfer comparison on Netflix Movies Title dataset

	RSA		HMAC	
size (mb)	request-response (s)	Data Transfer (mb/s)	request-response (s)	Data Transfer (mb/s)
6.3	1.43	4.405594406	0.22	28.63636364
6.3	1.12	5.625	0.222	28.37837838
6.3	0.23161	27.20089806	1.08	5.833333333
6.3	1.18	5.338983051	1.12	5.625
6.3	3	2.1	0.218	28.89908257
6.3	5.24	1.202290076	0.241	26.14107884
6.3	0.215	29.30232558	0.182	34.61538462
6.3	0.219	28.76712329	1.12	5.625
6.3	1.32	4.772727273	0.198	31.81818182
6.3	0.219	28.76712329	0.226	27.87610619

This study's findings on HMAC-SHA512 and RSA-SHA512 performance reveal important considerations for scalability in real-world applications, where data sizes and processing requirements vary widely. HMAC-SHA512's lightweight, symmetric nature enables it to handle high-throughput tasks efficiently, making it well-suited for applications with frequent token generation and verification, such as user authentication in large-scale web services. Its faster processing times and smaller token sizes contribute to reduced bandwidth usage and lower latency, offering scalable benefits in environments where rapid data transfer and minimal overhead are critical. Conversely, RSA-SHA512, with its asymmetric architecture, demonstrates robust performance in verification tasks, particularly in security-sensitive applications that prioritize strong key separation and verification stability. However, the computational demands of RSA can become a bottleneck in scenarios requiring frequent token generation, especially as data size and key length increase. In high-volume applications where RSA's security benefits are essential, scalability may require additional processing resources or hybrid approaches that balance RSA's strengths with the speed of symmetric algorithms like HMAC. This analysis underscores the importance of selecting encryption strategies that align with an application's scalability needs, ensuring that both security and performance are maintained as data requirements evolve.

Conclusion

The RSA-SHA512 algorithm's average time to create a token was consistent across the datasets, ranging from 0.001052 to 0.001262 seconds. However, its performance varied slightly more on the Netflix Movies dataset, suggesting that data complexity and structure might affect RSA's efficiency. Conversely, the HMAC-SHA512 algorithm demonstrated significantly faster token creation times, ranging from 0.000342 to 0.000423 seconds across all datasets. This consistent superiority in speed indicates HMAC-SHA512's efficiency in token creation, making it more suitable for applications requiring rapid token generation.

In terms of token verification, RSA-SHA512 showed impressive performance with average verification times ranging from 0.00014 to 0.000172 seconds across the datasets. The highest RPS was recorded for the Netflix Movies dataset at 7151.007, indicating RSA's robustness under varying conditions. HMAC-SHA512, while stable, had slower verification times ranging from 0.000327 to 0.000342 seconds, with RPS values between 2836.013 and 3073.475. This suggests that although HMAC-SHA512 is reliable, RSA-SHA512 might be better suited for applications where faster verification is critical.

For token decoding, RSA-SHA512 displayed variable yet consistent performance, with average durations between 4.9E-06 and 6.1E-06 seconds across the datasets. The RPS values for RSA ranged from 163898.4 to 214199.2, demonstrating its capability to handle high-volume data efficiently. HMAC-SHA512, while competitive, showed greater variability in decoding performance, with average durations between 5.18E-06 and 6.82E-06 seconds and RPS values from 146706 to 193204.4. This indicates that while HMAC-SHA512 is effective, RSA-SHA512 may offer more stable performance for token decoding.

When comparing token sizes, HMAC-SHA512 consistently produced smaller tokens than RSA-SHA512 across all datasets. For example, the Amazon Prime Users dataset showed a reduction from 1099 bytes (RSA) to 843 bytes (HMAC). This pattern was consistent for the Credit Score Users and Netflix Movies datasets, with HMAC tokens being 256 bytes smaller on average. The reduced token size of HMAC-SHA512 highlights its efficiency in minimizing data payload, which is advantageous for bandwidth and storage optimization.

The data transfer performance revealed that HMAC-SHA512 outperformed RSA-SHA512 significantly. For instance, in the Credit Score Users dataset, HMAC achieved data transfer rates ranging from 3.445 MB/s to 42.601 MB/s, whereas RSA's rates ranged from 2.515 MB/s to 9.822 MB/s. The Netflix Movies datas*et al*so demonstrated HMAC's superiority, with transfer rates between 5.625 MB/s and 34.615 MB/s, compared to RSA's range of 1.202 MB/s to 29.302 MB/s. These results underscore HMAC-SHA512's consistent and higher data

transfer efficiency, making it a better choice for environments requiring fast and reliable data handling.

Overall, the HMAC-SHA512 algorithm outperforms RSA-SHA512 across various metrics, including token creation speed, data transfer efficiency, and token size. While RSA-SHA512 demonstrates robust performance in token verification and decoding, HMAC-SHA512's superior efficiency and consistency make it more suitable for applications where speed and reduced data payload are critical. Therefore, for most practical purposes, especially in environments demanding high performance and minimal latency, HMAC-SHA512 is the preferred choice.

Author's Contributions

Louis Leonardo: Do computer experiments, analyze data and write papers.

Rajali: Supervising research projects, data analysis review, paper writing review, final paper review.

Ethics

The author states that this paper has never been published and there are no ethical issues in writing.

Conflict of Interest Declaration

The author states that there is no conflict of interests regarding the publication of this paper.

References

Ady Kusuma, I. G. N. (2021). Perancangan Simple Stateless Autentikasi Dan Otorisasi Layanan Rest-Api Berbasis Protokol Http. *Jurnal Manajemen Informatika Dan Sistem Informasi*, *4*(1), 78–87. https://doi.org/10.36595/misi.v4i1.325

Backendal, M., Bellare, M., Günther, F., & Scarlata, M. (2023). When Messages Are Keys: Is HMAC a Dual-PRF? *Advances in Cryptology -- CRYPTO 2023*, 661–693. https://doi.org/10.1007/978-3-031-38548-3_22

Frisca, G. S., Kosasi, S., Wijaya, T., Laipaka, R., & David, D. (2023). Pemanfaatan Web Service dalam Sistem Layanan Gereja Katolik Paroki Mrpd. *NARATIF : Jurnal Nasional Riset Aplikasi Dan Teknik Informatika*, *5*(1), 54–70. https://doi.org/10.53580/naratif.v5i1.194

Halili, F., & Ramadani, E. (2018). Web Services: A Comparison of Soap and Rest Services. *Modern Applied Science*, *12*(3), 175. https://doi.org/10.5539/mas.v12n3p175

Kiran, K. V. V. N. L. S., & Harini, N. (2018). Evaluating Efficiency of HMAC and Digital Signatures to Enhance Security in IoT. *International Journal of Pure and Applied Mathematics, 119*(12), 13991–13997.

Luo, Z. J., Liu, R., Mehta, A., & Ali, M. L. (2023). Demystifying the RSA Algorithm: An Intuitive Introduction for Novices in Cybersecurity. *ArXiv:2308.02785v2*. https://doi.org/10.48550/arXiv.2308.02785

Prayogi, A. A., Niswar, M., Indrabayu, & Rijal, M. (2020). Design and Implementation of REST API for Academic Information System. *IOP Conference Series: Materials Science and Engineering*, *875*(1), 012047. https://doi.org/10.1088/1757-899x/875/1/012047

Ramadhoni, M., & Santoso, H. (2023). Performance Comparison between Signature Cryptography: A Case Study on SNAP Indonesia. *Sinkron*, *8*(4), 2327–2335. https://doi.org/10.33395/sinkron.v8i4.12819