UTM UNIVERSITI TEKNOLOGI MALAYSIA
INTERNATIONAL JOURNAL OF INNOVATIVE COMPUTING
ISSN 2180-4370
Journal Homepage : https://ijic.utm.my/

Cryptography for Blockchain System: Comparative Analysis of Cryptographic Algorithms

Woon Zi Jian[1] & Hazinah Kutty Mammi[2]
Faculty of Computing
Universiti Teknologi Malaysia
81310 UTM Johor Bahru, Johor, Malaysia
Email: woonjian@graduate.utm.my[1]; hazinah@utm.my[2]

Submitted: 13/11/2024. Revised edition: 20/5/2025. Accepted: 20/5/2025. Published online: 27/5/2025
DOI: https://doi.org/10.11113/ijic.v15n1.500

***Abstract*—This study aims to investigate the best cryptographic algorithm for Blockchain Networks. The rapid adoption of blockchain technology and cryptocurrencies has generated worries regarding their long-term security, specifically concerning the cryptographic algorithms used to protect these systems. Despite well-established cryptographic technologies, blockchain is not inherently secure and requires a comprehensive assessment to defend against cybersecurity threats and vulnerabilities. Therefore, this research has been conducted and aims to discover the best cryptographic algorithms for cryptocurrency performance among the SHA-256, Keccak-256, SHA-512, and Keccak-512. Experiments are conducted and the conclusion can be made by analysing the results, and the best cryptographic algorithm is selected for the cryptocurrency blockchain system.**

***Keywords*—Blockchain Network, Cybersecurity, Cryptographic Algorithm**

I. INTRODUCTION

In recent years, blockchain technology has garnered much attention as a promising solution to many problems plaguing traditional centralized systems. At the heart of blockchain technology is a decentralized, secure network maintained by cryptographic algorithms. Cryptography, the science of secure communication in the presence of adversaries, is an essential component of blockchain systems because it guarantees transaction confidentiality, authenticity, and integrity [1].

For example, Ethereum is a decentralized, open-source blockchain platform that has garnered immense popularity over the past few years due to its ability to support the creation of decentralized applications (dApps) and smart contracts. Miners solve intricate mathematical problems to validate transactions and add them to the Ethereum blockchain as part of the proof-of-work consensus mechanism [2]. However, the security and efficacy of the Ethereum network are heavily dependent on the cryptographic algorithms used to protect it.

Selecting a cryptographic algorithm is crucial for ensuring the security and performance of blockchain systems. For assuring the integrity of the blockchain, a cryptographic algorithm must be robust, efficient, and attack-resistant. Concerns regarding efficacy and security can make developing and adopting new cryptographic algorithms difficult [3].

II. PROBLEM BACKGROUND

The rapid adoption of blockchain technology and cryptocurrencies has generated worries regarding their long-term security, specifically concerning the cryptographic algorithms used to protect these systems. Many cryptographic methods now employed in blockchain systems, including Ethereum, are susceptible to computer assaults.

As computing technology evolves, creating and deploying cryptography algorithms immune to attacks by these machines becomes increasingly crucial. However, integrating cryptography into existing blockchain systems can be difficult, and implementing these algorithms in various cryptocurrencies must be evaluated.

Cryptographic algorithms are essential for the security and privacy of blockchain systems, such as the Ethereum platform. To ensure the integrity of transactions on its blockchain, Ethereum employs Keccak-256 cryptographic algorithm that securing the network and resistant to attacks.

Despite numerous kinds of research on cryptographic algorithm performance in blockchain systems, there is a

significant gap that exists regarding comparisons between the specific cryptographic algorithms employed by Ethereum–namely SHA-256, SHA-512, Keccak-256, and Keccak-512. Most studies have focused on various algorithms without focusing on any particular blockchain platform. This research must be carried out to better comprehend the efficiency and robustness of these cryptographic algorithms within Ethereum, one of the leading blockchain platforms for widespread adoption.

A complete analysis of cryptography algorithms is presented as a solution to this challenge. Therefore, the analysis is conducted to analyse the performance of various cryptographic algorithms, such as SHA-256 cryptography, Keccak-256 cryptography, SHA-512 cryptography, and Keccak-512 cryptography to identify the most suitable algorithm for secure and efficient transactions. Although numerous studies have explored cryptographic algorithms in general blockchain systems, few have directly compared the specific algorithms used in Ethereum within a controlled simulation environment. Existing literature often lacks focus on Ethereum's context or omits important metrics such as CPU load and memory usage. Therefore, this research fills a unique gap by analyzing the performance of SHA and Keccak algorithm variants using detailed resource metrics specifically in the context of Ethereum blockchain simulations.

This study differs from past works by conducting an experimental analysis of cryptographic algorithms using practical performance indicators, enabling better decision-making for blockchain system design and optimization.

III. LITERATURE REVIEW

A cryptographic algorithm that can protect a blockchain system and resistant to all attack may be quite challenging. Therefore, numerous related studies have been published about the performance of the cryptographic algorithms.

A. Investigating The Efficiency of Cryptographic Algorithms In Online Transactions

Lamprecht, C., *et al.* [4] had investigating the efficiency of cryptographic algorithms in online transactions. The algorithms involved are MD2, MD4, MD5, RIPEMD128, RIPEMD160, SHA, SHA-0, SHA-1, SHA-256, SHA-512, and Tiger. The tools used in this experiment was Java Cryptix Libraries, Java, Linux Fedora Core 2, PC, and Trusted Services Integration Kit (TSIK). The experiment is conducted in a virtual machine and the Java coding was taken from the Java Cryptrix Libraries.

B. Performance Analysis of Cryptographic Hash Function Suitable for Use in Blockchain

Another research done by Kuznetsov, A., *et al.* [5] they are doing performance analysis of cryptographic hash function suitable for use in blockchain. The algorithms tested are GOST 34.311, STRIBOG256, STRIBOG512, KECCAK-256, KECCAK-512, SHA-256, SHA-512, RIPEMD160, BLAKE2B, Whirlpool, and they tested in their own machine and using HashCat software.

Cryptocurrencies rely on cryptographic algorithms to secure their blockchain networks and safeguard transactions and data from unauthorised access or tampering. Cryptographic algorithms perform fundamental cryptographic operations such as encryption, decryption, hashing, and digital signature generation. Below are some examples of cryptographic algorithms that can be implement into current blockchain systems.

x

1) SHA-256

Secure Hash Algorithm 256 (SHA-256) is a cryptographic hash function that generates a fixed-size output (256 bits) from any length input message. Fig. 1 shows the structure of SHA-256. A hash function generates a "fingerprint" of a message or data for authentication. The length of the hash code generated by the algorithm determines the resistance of the hash code to brute-force attacks. SHA-256 is extensively employed in applications requiring digital signatures, authentication, message integrity, and data protection [8].

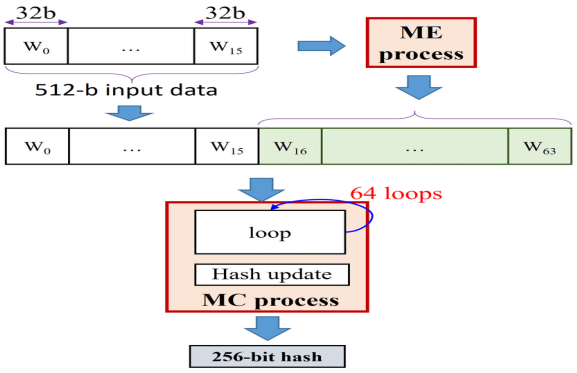

Fig. 1. Structure of SHA-256 [10]

2) SHA-512

SHA-512 is a cryptographic hash function with a fixed output capacity of 512 bits. The structure of SHA-512 is shown in Fig 2. It belongs to the Secure Hash Algorithm family, alongside SHA-1, SHA-2, and SHA-3. By using an extended message schedule and larger hash values, SHA-512 is intended to be more secure than its predecessor, SHA-256. It is commonly used in digital signatures, message authentication codes, and other applications where data integrity is crucial [7].

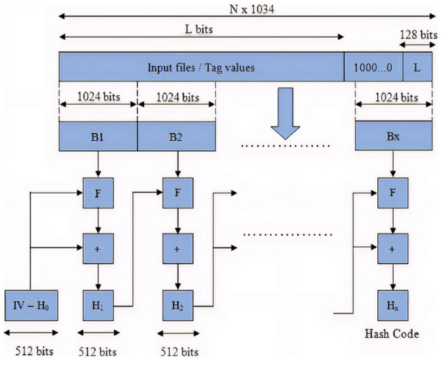

Fig. 2. Structure of SHA-512 [11]

3) Keccak-256

Keccak-256 is precisely engineered to defend against every recognized attack with utmost security. The structure of Keccak-256 is shown in Fig 3.This hash function belongs to the esteemed SHA-3 family, which is lauded for its impeccable performance. By accepting input messages of diverse lengths, Keccak-256 flawlessly churns fixed-length outputs that are 256 bits long [13]. This algorithm efficiently executes both hashing and encryption functions by employing a sponge-based mechanism in its design. A significant feature of this technique is that it absorbs an input message within a state using the sponge structure before compressing said content into output results.

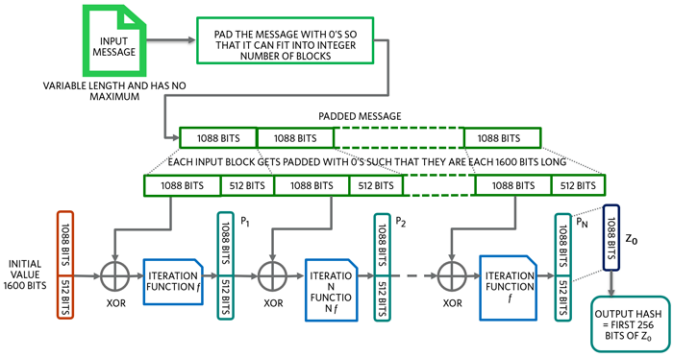

Fig. 3. Structure of Keccak-256 [9]

4) Keccak-512

Keccak-512 algorithm utilizes a sponge construction inherited from well-known hash functions like SHA-256 and SHA-3. Fig 4 shows the structure of Keccak-512. Part of a larger group of this kind, the Keccak hash function family-Keccak -512 cemented its place in cryptographic history for being one of the most secure algorithms available [6]. Keccak-512's design is intended to counteract various attacks, such as collision attacks, preimage attacks, and second preimage attacks. Moreover, it boasts of being immune to length extension attacks leveraged to produce new messages with an identical hash value to an existing one.

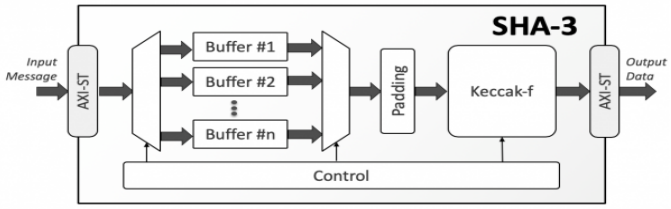

Fig. 4. Structure of Keccak-512 [1]

While prior studies such as Lamprecht *et al.* (2006) and Kuznetsov *et al.* (2021) have explored the performance of cryptographic algorithms in blockchain environments, many of them focus primarily on theoretical comparisons or hashing speed. These studies often omit critical system-level performance data such as CPU usage, memory consumption, and throughput, which are vital in real-world deployments. In contrast, this study provides a comprehensive experimental evaluation of four widely used cryptographic hash algorithms—SHA-256, SHA-512, Keccak-256, and Keccak-512—based on multiple performance metrics. The goal is to offer practical insights for developers and blockchain system designers by providing reproducible performance results under varying data loads.

IV. RESEARCH FRAMEWORK

This research consists of three phases. For Phase 1, a literature review studying existing cryptographic algorithms and their characteristics was conducted. Next, the second phase was to design, develop, and test the experiment performed in Phase 2. Lastly, the experiment was conducted, and its results were analysed and discussed in Phase 3. The workflow of the overall research framework is illustrated in Fig. 5.

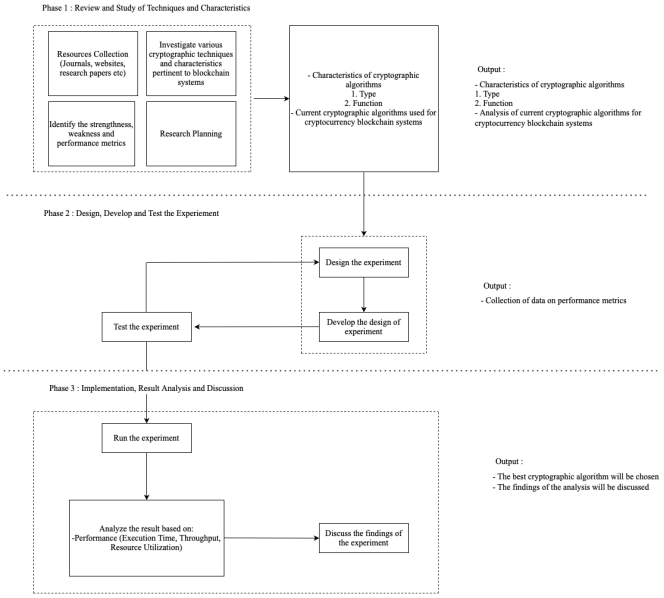

Fig. 5. Workflow of the Overall Research Framework

A. Phase 1: Review and Study of Techniques and Characteristics

In Phase 1, information related to the research topic is gathered, and these resources are obtained through journals, websites, and articles. Then, the helpful information helps to investigate various cryptographic algorithms and characteristics pertinent to blockchain systems. This phase entails collecting data on different cryptographic algorithms and their applications in cryptocurrency and comprehending their strengths, weaknesses, and performance metrics. The overall planning for this research is also done in this phase to ensure the following process can always be on track. At the end of Phase 1, the first research objective is achieved.

B. Phase 2: Design, Develop, and Test the Experiment

Phase 2 focused on designing and developing the proposed method. Based on the results from Phase 1, the experiment is

designed to fulfil the problem background, research objectives, and scope of this research. The evaluation criteria and metrics will be defined to assess the cryptographic algorithms' performance and security based on cryptocurrency. Then, the proposed method was developed, and a simple experiment was tested. If the experiment fails, the experiment will be redesigned until it succeeds. The second objective is achieved at the end of Phase 2.

C. Phase 3: Implementation, Result Analysis and Discussion

In Phase 3, the experiment was running based on the tested experiment from Phase 2. Then, the data was collected, and the collected data were analysed in this final phase, and the results obtained from the evaluations and experiments were interpreted. The performance of different cryptographic algorithms for cryptocurrency was compared, including the strengths and weaknesses identified during the review phase. The results will be discussed, conclusions will be drawn, and recommendations for selecting the best cryptographic algorithms for cryptocurrency have been made. At the end of this phase, objective three in this research was achieved and led to the end of Phase 3.

V. EXPERIMENT DESIGN

The flowchart for the whole experiment design is shown below in Fig. 6.

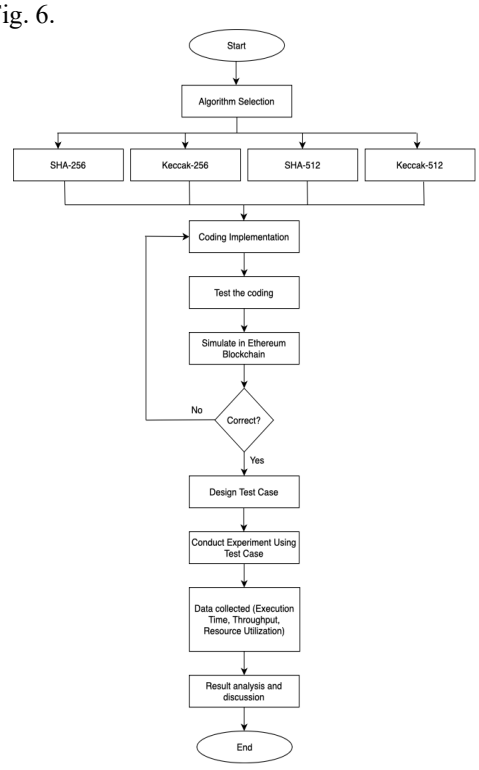

Fig. 6. Experiment Design Flowchart

The experiment starts by selecting cryptographic algorithms suitable for cryptocurrency blockchain environments. Cryptographic algorithms such as SHA-256, Keccak-256, SHA-512, and Keccak-512 were chosen due to their prominent roles in various applications worldwide.

Then, the coding of these cryptographic algorithms needs to be implemented using Microsoft Visual Studio Code as our development environment platform because it provides us with a sense of independence coupled with Java programming language, wherein its ease of use translates into seamless integration with other existing systems. Before evaluating their performance within blockchain ecosystems, we test these algorithms' accuracy and reliability to ensure minimum errors occur in the experiment later. Within the experiment, metrics such as execution speed, throughput, and resource utilization were used to test the performance of the algorithms.

After that, all the experiment is conducted. The test cases are run based on the chosen cryptographic methods and the transaction data size. Data for each success metric is collected, and each algorithm's strengths and weaknesses are identified.

Lastly, the data based on performance compares the cryptographic algorithms. The results are examined to determine the cryptocurrency blockchain system's best algorithm. The experiment's design structured the test of the correctness and speed of the algorithms that can be added to the blockchain systems.

VI. RESULT AND DISCUSSION

The experiment tested three data sets: 1MB, 5MB, and 10MB JSON files. The overall experimental result for SHA-256, SHA-512, Keccak-256, and Keccak-512 cryptographic algorithms on 1MB is shown in the Table I.

TABLE I. OVERALL PERFORMANCE DATA FOR SHA-256, SHA-512, KECCAK-256 AND KECCAK-512 ON HASHING 1MB JSON FILE

Algorithm	Average Hashing Speed (MB/s)	Average Hashing Time (ms)	CPU Load (Before)	CPU Load (After)	Memory Usage (Before)	Memory Usage (After)	Throughput (MB)
SHA-256	194.108	5.07	0.16	0.16	1.49	1.53	193.06
Keccak-256	486.113	2.02	0.26	0.19	7.5	7.68	484.53
SHA-512	284.405	3.46	0.17	0.19	1.49	1.57	282.98
Keccak-512	244.688	4.00	0.25	0.18	7.5	7.68	244.69

Table I shows the performance data of SHA-256, SHA-512, Keccak-256, and Keccak-512 when hashing a 1MB JSON file in terms of average hashing speed, average hashing time, average CPU load before and after, average memory usage before and after and throughput. Based on the performance data in Table I, the performance trends and comparisons for the algorithms SHA-256, SHA-512, Keccak-256, and Keccak-512 when hashing a 1MB JSON file are discussed below. Also, graphs were constructed to have a better picture to explain the data.

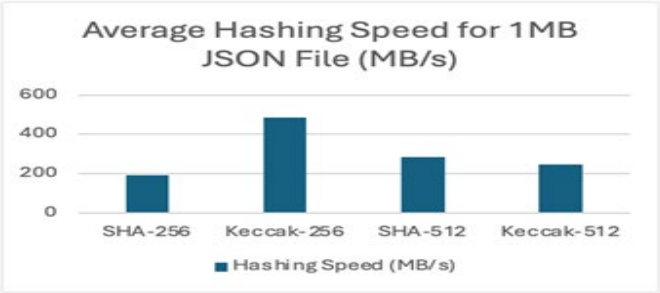

Fig. 7. Average Hashing Speed for 1MB JSON File

Fig. 7 shows the hashing speeds of SHA-256, SHA-512 Keccak-256 and Keccak-512 for a 1MB JSON File. By comparing the hashing speeds, Keccak-256 stands out as the most efficient, with a speed of 486.113 MB/s surpassing the other algorithms by a significant margin. Following closely is SHA-512, with a speed of 284.405 MB/s, Keccak-512 at 244.688 MB/s. SHA-256 lags with the slowest average hashing speed of 194.108 MB/s. This suggests that when it comes to processing speed for 1MB JSON files Keccak-256 proves to be the algorithm.

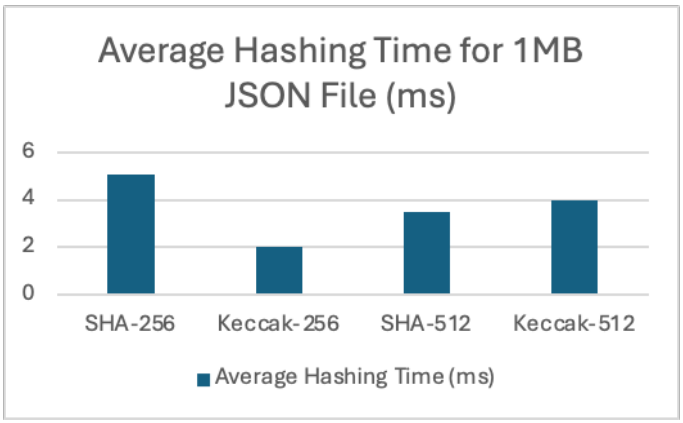

Fig. 8. Average Hashing Time for 1MB JSON File

Fig. 8 illustrates the average hashing time for SHA-256, SHA-512, Keccak-256, and Keccak-512 for 1MB JSON File. In terms of hashing time, Keccak-256 is the fastest, with an average of 2.02 milliseconds. SHA-512 follows with an average of 3.46 milliseconds and Keccak-512 at 4.00 milliseconds. SHA-256 takes the longest, with an average time of 5.07 milliseconds. This further confirms that Keccak-256 is the most efficient algorithm in terms of processing time for the given data size.

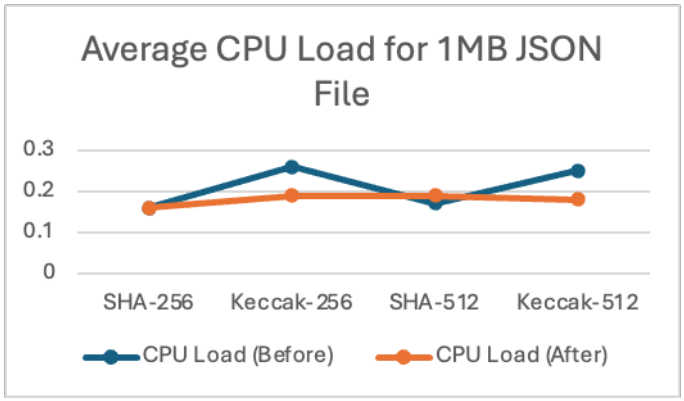

Fig. 9. Average CPU Load for 1MB JSON File

Fig. 9 illustrates the average CPU load for SHA-256, SHA-512, Keccak-256, and Keccak-512 for 1MB JSON File. By evaluating CPU load, SHA-256 and SHA-512 show minimal impact on CPU usage, with both algorithms starting and ending at around 0.16 and 0.17 respectively before hashing and after hashing. Keccak-256 starts with a higher CPU load of 0.26 before hashing but drops to 0.19 after hashing. Keccak-512 also starts with a higher load of 0.25 and drops to 0.18 after hashing. This indicates that while Keccak algorithms may initially use more CPU resources, their efficiency reduces load post-hashing.

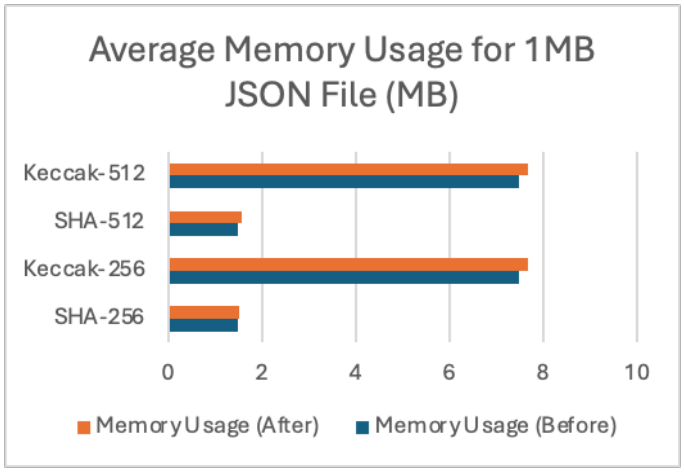

Fig. 10. Average Memory Usage for 1MB JSON File

Fig. 10 shows the bar graph of average memory usage for SHA-256, SHA-512, Keccak-256, and Keccak-512 for 1MB JSON File. For memory usage, both SHA-256 and SHA-512 start with 1.49 MB and slightly increase to 1.53 MB and 1.57 MB respectively after hashing. Keccak-256 and Keccak-512 start with a higher memory usage of 7.5 MB, which increases to 7.68 MB after hashing. This suggests that while Keccak algorithms are faster, they require more memory compared to SHA algorithms.

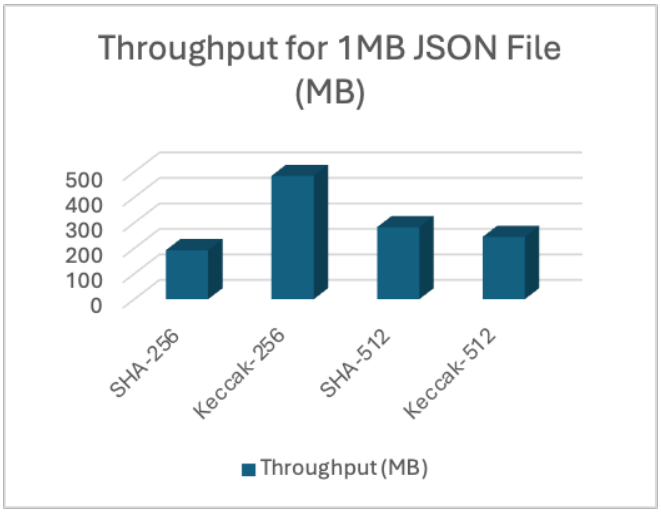

Fig. 11. Throughput for 1MB JSON File

Fig. 11 presents the graph of the average hashing speed for SHA-256, SHA-512, Keccak-256, and Keccak-512 for 1MB JSON File. In terms of throughput, which combines speed and efficiency, Keccak-256 leads again with a throughput of 484.53 MB, significantly higher than the other algorithms. SHA-512 follows with 282.98 MB, then Keccak-512 with 244.69 MB, and SHA-256 with the lowest throughput of 193.06 MB. This further reinforces the conclusion that Keccak-256 is the most efficient algorithm overall for 1MB JSON files, providing the highest throughput with lower execution time.

Based on this experiment, the result shows that SHA-256 has an average hashing time of 5.07 milliseconds and a speed of 194.108 MB/s, with minimal impact on system resources, making it suitable for 1MB JSON files. SHA-512 performs faster, with a 3.46 millisecond hashing time and a speed of 284.405 MB/s. It has a slightly higher memory usage post-operation but is optimized for 64-bit processors, making it efficient for 1MB JSON files. Keccak-256 demonstrates exceptional performance with a 2.02 millisecond hashing time and a speed of 486.113 MB/s due to its higher absorption rate. It utilizes CPU resources and slightly increases memory usage, making it ideal for high-performance tasks. Keccak-512 balances performance and security, with a 4-millisecond hashing time and a speed of 244.688 MB/s. It efficiently manages CPU resources with consistent memory usage, making it suitable for scenarios prioritizing security.

Comparing SHA-256 and SHA-512, SHA-512 is faster (284.405 MB/s vs. 194.108 MB/s) and more efficient for 1MB JSON files, despite slightly higher memory usage. Keccak-256 and Keccak-512 show that Keccak-256 is faster (486.113 MB/s vs. 244.688 MB/s) and more efficient for high-performance applications, while Keccak-512 is better for security-focused tasks. The results and trends for hashing 5MB and 10MB JSON files are the same. Hence, it can help to conclude that Keccak-256 is the best algorithm in terms of execution time and throughput, while SHA-256 is the best algorithm in terms of resource utilization.

VII. CONCLUSION

This research compared the performance of four cryptographic algorithms—SHA-256, SHA-512, Keccak-256, and Keccak-512—in a simulated blockchain framework. The results obtained with varied file sizes were always consistent, with Keccak-256 providing the optimal hashing speed, execution time, and throughput. Keccak-256 had a hashing speed of 486.113 MB/s and the least average hashing time of 2.02 milliseconds. Nevertheless, it had marginally higher memory consumption than SHA algorithms.

Conversely, SHA-256 and SHA-512 had negligible CPU and memory consumption and were thus better suited to low-resource environments at the expense of speed. Keccak-512 offered a good trade-off between performance and security and was hence apt for applications requiring high-security levels.

The results conclude that Keccak-256 is the best-performing algorithm among the algorithms considered when speed and throughput are the objectives. This presents useful information to blockchain developers and system designers interested in selecting optimal cryptographic algorithms.

Shortcomings in the present study are its reliance on simulations and a constrained testing setting. The performance might differ in real deployments or under fluctuating network loads. Future research should investigate larger data sizes, testnet integrations, and performance on heterogeneous hardware architectures. Further security evaluation would also be warranted to accompany performance testing.

This paper contributes to the field by providing a practical, system-oriented comparison of commonly used cryptographic algorithms, using experimental data that extends beyond theoretical metrics often emphasized in previous studies.

ACKNOWLEDGMENT

We would like to thank members of the Computer Science Department of Universiti Teknologi Malaysia for their helpful feedback and support.

CONFLICTS OF INTEREST

The author(s) declare(s) that there is no conflict of interest regarding the publication of this paper.

REFERENCES

[1] Habib, G., Sharma, S., Ibrahim, S., Ahmed, I., Qureshi, S., & Ishfaq, M. (2022). Blockchain technology: Benefits, challenges, applications, and integration of blockchain technology with cloud computing. *Future Internet*, *14*(11), 341. https://doi.org/10.3390/fi14110341.

[2] Frankenfield, J. (2023). Decentralized applications (dApps). investopedia. https://www.investopedia.com/terms/d/decentralized-applications-dapps.asp.

[3] Guo, H., & Yu, X. (2022). A survey on blockchain technology and its security. *Blockchain: Research and Applications*, *3*(2), 100067.

[4] Lamprecht, C., Van Moorsel, A., Tomlinson, P., & Thomas, N. (2006). Investigating the efficiency of cryptographic algorithms

in online transactions. *International Journal of Simulation: Systems, Science & Technology*, *7*(2), 63–75.

[5] Kuznetsov, A., Oleshko, I., Tymchenko, V., Lisitsky, K., Rodinko, M., & Kolhatin, A. (2021). Performance analysis of cryptographic hash functions suitable for use in blockchain. *International Journal of Computer Network & Information Security*, *13*(2), 1–15.

[6] Bertoni, G., Daemen, J., Peeters, M., & Van Assche, G. (2009). The road from Panama to Keccak via RadioGatún. *Dagstuhl Seminar Proceedings. Schloss Dagstuhl-Leibniz-Zentrum für Informatik.*

[7] Khaishagi, Z. (n.d.). Cryptography: Explaining SHA-512. Medium. https://medium.com/@zaid960928/cryptography-explaining-sha-512-ad896365a0c1.

[8] Rachmawati, D., Tarigan, J. T., & Ginting, A. B. C. (2018). A comparative study of Message Digest 5(MD5) and SHA256 algorithm. *Journal of Physics: Conference Series, 978*.

[9] Analog Devices. (n.d.). Back to basics: Secure hash algorithms. analog devices. https://www.analog.com/en/technical-articles/back-to-basics-secure-hash-algorithms.html.

[10] Tran, Thi Hong & Hoai Luan, Pham & Nakashima, Yasuhiko. (2021). A High-Performance Multimem SHA-256 Accelerator for Society 5.0. *IEEE Access*, 1–1. 10.1109/ACCESS.2021.3063485.

[11] P G, Shynu & Rk, Nadesh & Menon, Varun & Parameswaran, Venu & Abbasi, Mahdi & Khosravi, Mohammad. (2020). A secure data deduplication system for integrated cloud-edge networks. *Journal of Cloud Computing Advances Systems and Applications, 9*, 1–12. 10.1186/s13677-020-00214-6.

[12] CAST, Inc. (n.d.). SHA-3. CAST, Inc. https://www.cast-inc.com/security/encryption-primitives/sha-.

[13] Ranadive, S. S., Sawant, H. S., & Pinjarkar, J. E. (2022). Secure file storage on cloud computing using cryptographic algorithm. *International Journal for Research in Applied Science and Engineering Technology*.

\