Digital Zone
Jurnal Teknologi Informasi dan Komunikasi

Volume 17 Issue 1 Year 2026 | Page 105-116 | e-ISSN: 2477-3255 | ISSN: 2086-4884
Received: 18-11-2025 | Revised: 30-04-2026 | Accepted: 26-05-2026

Analytical Study of Hash Algorithm Performance for Smart Grid Data Security

Muhammad Ridwan[1], Rusydi Umar[2], Muhammad Kunta Biddinika[3], Puguh Prasetyo[4]
[1,2,3,4] Universitas Ahmad Dahlan, Yogyakarta, Indonesia, 55191
E-mail:ridwan211221@gmail.com[1],rusydi@mti.uad.ac.id[2],muhammad.kunta@mti.uad.ac.id[3], puguh.prasetyo@pmat.uad.ac.id[4]
*Corespondence: ridwan211221@gmail.com

Abstract: *Data integrity is a critical requirement in Smart Grid systems due to the continuous exchange of high-frequency transactional data across distributed environments. Cryptographic hash functions play a key role in ensuring data authenticity and preventing unauthorized modification. This study presents a comparative analytical evaluation of three widely used hash algorithms SHA-256, SHA3-256, and BLAKE2b using simulated Smart Grid data. The evaluation focuses on computational efficiency, avalanche effect sensitivity, and demonstrative security properties under controlled experimental conditions. The experimental results show that all algorithms maintain strong diffusion characteristics, with avalanche effect values close to the theoretical ideal, indicating reliable sensitivity to input changes. In terms of computational performance, BLAKE2b demonstrates lower execution time and more stable performance compared to SHA-256 and SHA3-256 within the tested environment. However, the observed differences are relatively small and should be interpreted cautiously. This study contributes by providing a contextual evaluation of hash algorithm performance in relation to Smart Grid data characteristics, highlighting the balance between efficiency and security sensitivity. Nevertheless, the experiments are conducted using a limited synthetic dataset and a software-based runtime environment, and the security tests are demonstrative in nature. Therefore, the findings should be considered as preliminary analytical insights rather than direct recommendations for real-world Smart Grid deployment.*

Keywords: *Smart Grid, Cryptographic Hash Functions, SHA-256, SHA3-256, BLAKE2b, Avalanche Effect, Data Integrity, Hashing Performance*

1. Introduction

Smart Grid systems represent a modern transformation of electrical infrastructure by integrating power networks with information and communication technology (ICT) to enable real-time monitoring, automated control, and efficient energy distribution. However, this digital integration also increases the exposure of critical infrastructure to cybersecurity threats, particularly in the context of data integrity and authentication. Unlike conventional data systems, Smart Grid environments generate high-frequency, structured, and continuously transmitted data, such as energy consumption records, operational logs, and user-level transactions. These data are often processed in near real-time across distributed architectures. As a result, even minor data manipulation can propagate rapidly and affect system reliability, billing accuracy, and operational stability [1]. Therefore, ensuring data integrity in Smart Grid systems requires not only secure cryptographic mechanisms but also efficient algorithms capable of supporting low-latency processing. Cryptographic hash functions are widely used to maintain data integrity by producing fixed-length outputs that are highly sensitive to input changes. Key properties such as avalanche effect, collision resistance, and computational efficiency are essential in determining the suitability of a hash function for practical applications. In this context, widely used algorithms

such as SHA-256, SHA3-256, and BLAKE2b have been extensively studied in the literature. Previous research has conducted comparative benchmarking of these algorithms, primarily focusing on general-purpose datasets and evaluating performance metrics such as execution time and basic security properties. While these studies provide useful insights, they do not fully address the specific operational characteristics of Smart Grid systems. In particular, existing works rarely consider how hash algorithm performance interacts with Smart Grid requirements such as high-frequency data processing, latency sensitivity, and distributed system constraints. Based on the state-of-the-art review, a clear research gap can be identified[2]. Existing studies predominantly evaluate cryptographic hash algorithms in generic computational environments and rarely examine their suitability within Smart Grid-inspired data processing scenarios[3]. Furthermore, previous works often focus on isolated performance or security metrics without considering their combined implications for data integrity verification in latency-sensitive and continuously operating environments[4]. First, most benchmarking studies are conducted in generic computational contexts without incorporating domain-specific data characteristics. Second, prior evaluations often analyze performance and security metrics separately, rather than examining their combined impact on system-level requirements. Third, the relevance of these results to Smart Grid applications remains unclear due to the lack of contextual evaluation using data structures that resemble real Smart Grid transactions. The novelty of this study lies in the contextual evaluation of SHA-256, SHA3-256, and BLAKE2b using Smart Grid-inspired transactional data and multiple evaluation indicators, including computational efficiency, avalanche effect behavior, and demonstrative integrity validation[5]. Unlike conventional benchmarking studies that primarily emphasize generic performance measurements, this work attempts to relate cryptographic characteristics to Smart Grid operational considerations, thereby providing a more application-oriented analytical perspective. The main contribution of this research is the provision of a comparative analytical assessment of three widely used cryptographic hash algorithms within a Smart Grid simulation environment. The study also provides an integrated interpretation of efficiency, diffusion behavior, and demonstrative security characteristics while explicitly acknowledging the limitations of the experimental scope[6]. Based on these gaps, this study aims to provide a contextual evaluation of cryptographic hash algorithms within a Smart Grid simulation environment. Specifically, this research compares SHA-256, SHA3-256, and BLAKE2b based on hashing time efficiency, avalanche effect sensitivity, and demonstrative security properties. By integrating performance metrics with application-oriented considerations, this study seeks to provide a more relevant analytical perspective for Smart Grid data integrity. To improve clarity and readability, the research objectives are defined as follows: To evaluate the computational efficiency of SHA-256, SHA3-256, and BLAKE2b using Smart Grid simulation data. To analyze the avalanche effect of each algorithm as a measure of sensitivity to input changes. To compare the performance and security characteristics of the algorithms within the context of Smart Grid data processing requirements. This study is limited to a controlled experimental environment using simulated Smart Grid data and software-based implementation. It does not include real-world deployment, network level simulation, or hardware-based validation. Therefore, the results should be interpreted as analytical insights rather than direct implementation recommendations

2. Research Method

This study uses a quantitative experimental approach to analyze the performance and security of three cryptographic hash algorithms, namely SHA-256, SHA3-256, and BLAKE2b. This approach was chosen for its ability to provide objective measurements of hashing speed, collision resistance, demonstrative-scale preimage resistance, and the avalanche effect. The selection of these three algorithms is based on their relevance in modern data security discussions. SHA-256 was chosen as a mature industry-standard baseline [7][8]. SHA3-256 was selected as the modern NIST standard with a different internal structure based on Keccak [9][2]. Lastly,

https://doi.org/10.31849/digitalzone.v17i1.30369

BLAKE2b was chosen for its reputation in the literature for high computational efficiency, often reported to be faster than SHA-256 [10][11]. The experiment was conducted using simulated data representing a Smart Grid system, allowing full control over variables resembling real electrical energy data such as Meter ID, User ID, Time, and Power Consumption (kWh). All testing processes were implemented using Python 3 in Google Colab, a platform chosen for its stability, open-source accessibility, and full support for essential data analysis libraries such as hashlib, pandas, numpy, and matplotlib.

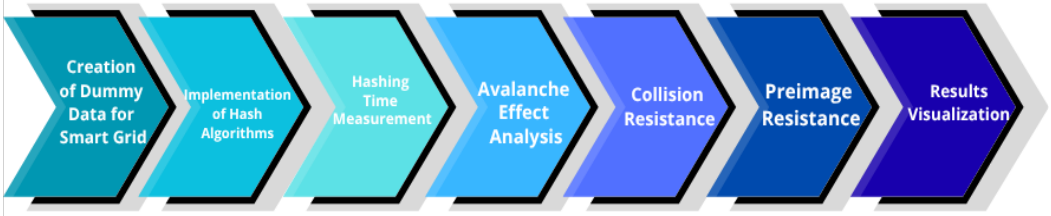


Figure 1. Method Flowchart

The research stages are structured modularly into seven main interconnected work blocks that reflect practical implementation in the Google Colab notebook to maintain a logistical workflow and reproducibility. All experiments are conducted in the Google Colab environment within a single runtime session without restarting[12]. This is done to ensure that the three algorithms are tested under identical computing conditions, so that the performance comparison results are consistent and not affected by environmental changes[13]. Each hashing time measurement is repeated five times within the same runtime session, and then the mean and standard deviation values are calculated to obtain a more accurate and representative performance estimate. This approach is very important because Google Colab runs on a virtualization infrastructure, where hardware specifications, such as CPU and memory allocation, can vary between sessions. This study adopts a quantitative experimental approach to evaluate the computational performance and security-related characteristics of three cryptographic hash algorithms: SHA-256, SHA3-256, and BLAKE2b. The evaluation focuses on four primary metrics: hashing time efficiency, avalanche effect, collision behavior (demonstrative), and preimage resistance (demonstrative). The experimental design is structured to ensure consistency, reproducibility, and fair comparison across all algorithms under identical conditions[14]. All experiments were conducted using Python 3.10 in a Google Colab environment. The computational runtime utilizes a virtualized CPU environment (Intel Xeon class processor) with approximately 2–4 vCPUs and 12–16 GB RAM. The primary libraries used include: hashlib for cryptographic hashing pandas and numpy for data manipulation time for performance measurement matplotlib for visualization All experiments were executed within a single runtime session without interruption to minimize variability caused by dynamic resource allocation in cloud-based environments[15]. The dataset used in this study consists of 2,000 rows of simulated Smart Grid data. Each record represents a simplified abstraction of energy consumption transactions and includes the following attributes: Meter_ID User_ID Timestamp Energy Consumption (kWh) Status (Normal, Anomaly, Error) Location Each record is transformed into a structured string input by concatenating selected attributes (Meter_ID, User_ID, Timestamp, and Consumption). The resulting input strings have an average length of approximately 40–60 characters, ensuring consistent input size across all hashing operations. It is important to note that this dataset is synthetic and designed to emulate structural characteristics of Smart Grid data rather than replicate real world communication dynamics such as network latency, distributed nodes, or streaming behavior.

Three widely used cryptographic hash algorithms were evaluated:

SHA-256 part of the SHA-2 family, producing a 256-bit digest, standardized by NIST.

SHA-3 (Keccak) the latest SHA standard, based on sponge construction, designed to resist advanced cryptanalysis.
BLAKE2 a modern and efficient hash function, optimized for speed while maintaining strong cryptographic resistance.
All three were implemented using Python's built-in libraries: hashlib for SHA-256 and SHA-3, and pyblake2 for BLAKE2.Experimental Design Three types of experiments were performed:
Avalanche Effect Test Input: data samples from the smart grid dataset.
Process: a one-bit modification was introduced into the original input (e.g., replacing one digit).
Measurement: the difference between the original hash and the modified hash was quantified using the Hamming Distance.
Formula:

$$HD(h1, h2) = \sum_{i=1}^{n} (b1i \oplus b2i)$$

where b1i and b2i are the two hash outputs, and n=256n = 256n=256 bits.
The avalanche percentage is given by:

$$PAE = HD \div 256 \times 100\%$$

Hash Consistency Test Input: identical data samples. Process: each record was hashed repeatedly using the same algorithm. Goal: verify that identical inputs always produce identical outputs. Performance Test (Computation Time) The execution time of each hashing operation was recorded in milliseconds. Comparison was made across the three algorithms. The experiments were implemented in Python (3.10) using Google Colab as the computational environment. The libraries employed were:
hashlib → for SHA-256 and SHA-3
pyblake2 → for BLAKE2
pandas and numpy → for data handling
matplotlib → for data visualization
Key functions included:

```
# Hashing function
def hash_data(algorithm, text):
    if algorithm == "sha256":
        return hashlib.sha256(text.encode()).hexdigest()
    elif algorithm == "sha3":
        return hashlib.sha3_256(text.encode()).hexdigest()
    elif algorithm == "blake2":
        return hashlib.blake2b(text.encode()).hexdigest()

# Hamming Distance for avalanche effect
def hamming_distance(h1, h2):
    b1 = bin(int(h1, 16))[2:].zfill(len(h1)*4)
    b2 = bin(int(h2, 16))[2:].zfill(len(h2)*4)
    return sum(c1 != c2 for c1, c2 in zip(b1, b2))
```

The implementation phase involves applying three cryptographic hash algorithms (SHA-256, SHA3-256, and BLAKE2b) to the Smart Grid dataset. Each data record is transformed into a unique input string by concatenating key attributes, including Meter_ID, User_ID, Timestamp, and Energy Consumption[10]. This approach ensures that each record has a distinct and consistent representation across all hashing processes[16]. The evaluation of algorithm performance is based on three primary metrics: (1) average hashing time, which reflects computational efficiency, (2) avalanche effect, which measures sensitivity to minor input changes, and (3) standard deviation of avalanche values, which indicates the consistency of bit diffusion[17]. To ensure reliability, each hashing experiment is repeated five times within the same runtime environment. The mean and standard deviation of execution time are

https://doi.org/10.31849/digitalzone.v17i1.30369

calculated to reduce the influence of computational variability in the cloud-based system[11]. Hashing time is measured using the Python time library by calculating the total execution duration required to process all 2,000 data records for each algorithm. This metric is particularly relevant in Smart Grid systems, where large volumes of data must be processed with low latency. The avalanche effect is evaluated by generating pairs of input messages that differ by a single bit. For each algorithm, 500 iterations are performed. In each iteration, the output hashes are compared at the bit level, and the percentage of differing bits is calculated. An avalanche value close to 50% indicates strong cryptographic diffusion. Collision testing is conducted by examining the uniqueness of hash outputs across all input records. The number of duplicate hashes is counted to identify potential collisions[18], However, this test is limited in scope and is intended only as a conceptual demonstration rather than a full evaluation of collision resistance. Similarly, a preimage resistance test is performed using a limited brute-force approach, where up to 1,000,000 candidate inputs are tested against a target hash. This procedure is designed to illustrate the one-way property of hash functions but does not constitute a comprehensive security evaluation[19]. All experimental results, including hashing time and avalanche effect values, are visualized using charts and tables generated with Python libraries. These visualizations support comparative analysis across algorithms under identical input conditions, ensuring a fair and consistent evaluation framework.

3. Results and Discussion

The initial stage of the experiment is the creation of the Smart Grid simulation dataset[20]. In accordance with the methodology, 2,000 unique rows of data have been successfully generated. Table 1 below presents a portion of the raw dataset, which includes crucial variables such as Meter_ID, User_ID, Timestamp, Consumption_kWh, Status, and Location. This dataset serves as the foundation and input for all hash analysis tests conducted in this study.

Table 1. Smart Grid Data 2000 (5)

Meters ID	User ID	Consumption Wh	Status	locationi
1001	U001	2.1	Anomali	Surabaya
1002	U002	02.06	Normal	Makassar
1003	U003	2.12	Anomali	Medan
1004	U004	1.0	Error	Palembang
1005	U005	1.95	Error	Malang

Next, the dataset is processed to implement the three hash algorithms. As shown in Figure 9, this implementation was successful[21]. Each row of input data was converted into a unique string and processed by the three hash algorithms separately. To ensure the consistency and reliability of the results, each hashing process was repeated five times within the same runtime session. The results from each trial were then used to calculate the mean and standard deviation, providing a more accurate and representative estimation of the algorithm performance[22]. Each row of input data generated a unique hash with a fixed length, which was stored in new columns: hash SHA256, hash SHA3-256, and hash BLAKE2b[10]. The success of this stage validates the hash functions as a foundation for subsequent performance and security testing, and serves as a reference for the discussion regarding hashing time efficiency, bit-level sensitivity (avalanche effect), and the demonstrative concepts of collision and preimage resistance. Thus, this approach enables a comprehensive and quantitative analysis of the performance of the three algorithms in maintaining the integrity of Smart Grid data.

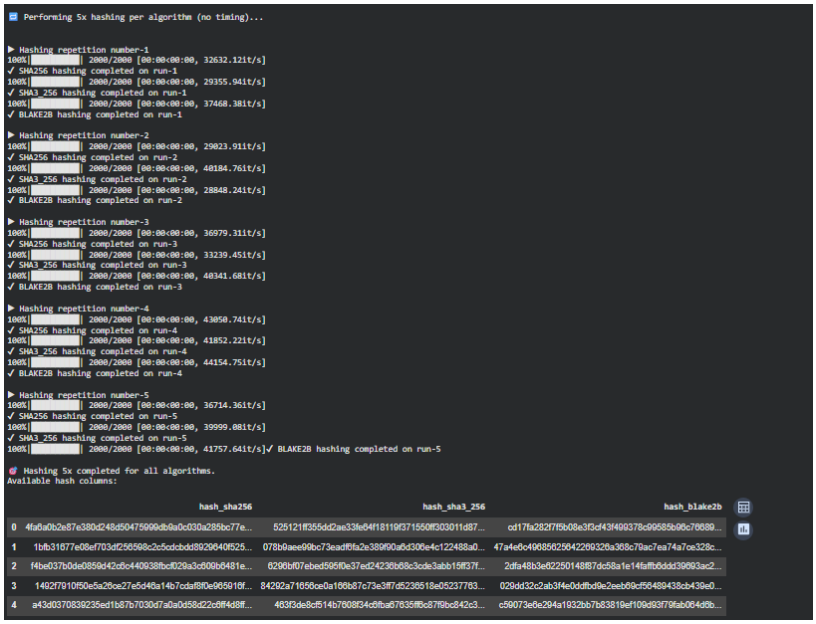
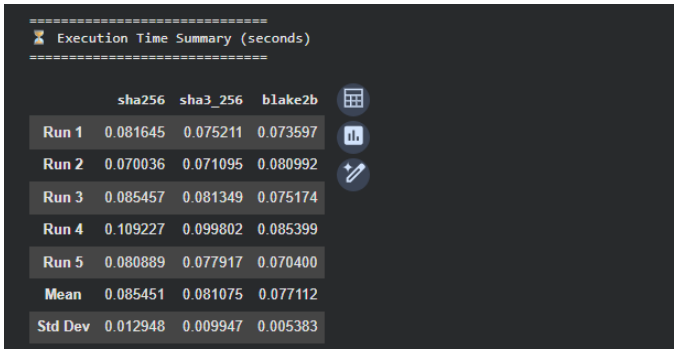


Figure 2. Hash Results of the Three Algorithms

The first and one of the most crucial analyses for Smart Grid applications is computational efficiency, measured as the total hashing time. Each hashing process was repeated five times to ensure the consistency and reliability of the results, and the mean (Mean) and standard deviation (Std Dev) were calculated for each algorithm. The measured hashing times for processing 2000 rows of Smart Grid data are presented in Figure 10 and visualized in Figure 11. For SHA-256, the hashing times across five consecutive trials were 0.081645 seconds, 0.070036 seconds, 0.085457 seconds, 0.109227 seconds, and 0.080889 seconds, with a mean of 0.085451 seconds and a standard deviation of 0.012948 seconds. For SHA3-256, the five trials resulted in 0.075211 seconds, 0.071095 seconds, 0.081349 seconds, 0.099802 seconds, and 0.077917 seconds, with a mean of 0.081075 seconds and a standard deviation of 0.009947 seconds. Meanwhile, BLAKE2b yielded 0.073597 seconds, 0.080992 seconds, 0.075174 seconds, 0.085399 seconds, and 0.070400 seconds, with a mean of 0.077112 seconds and a standard deviation of 0.005383 seconds. These results show a clear performance difference: BLAKE2b is the fastest algorithm, with the lowest average hashing time and the smallest variation, indicating high consistency. SHA3-256 ranks in the middle, while SHA-256 has the highest average time and the largest variation.



	sha256	sha3_256	blake2b
Run 1	0.081645	0.075211	0.073597
Run 2	0.070036	0.071095	0.080992
Run 3	0.085457	0.081349	0.075174
Run 4	0.109227	0.099802	0.085399
Run 5	0.080889	0.077917	0.070400
Mean	0.085451	0.081075	0.077112
Std Dev	0.012948	0.009947	0.005383

Figure 3. Hashing Time Measurement Results

https://doi.org/10.31849/digitalzone.v17i1.30369

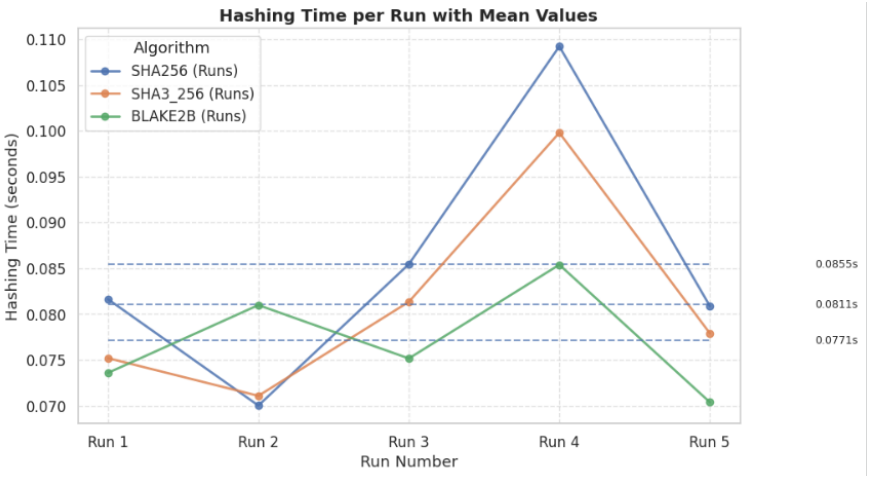


Figure 4. Hashing Time Measurement Graph

The primary security metric evaluated is the Avalanche Effect, which measures how well a single-bit change in the input can randomize the hash output. Results from 500 iterations of testing show that all three algorithms perform very well and are secure, with average values approaching the ideal 50%, as also identified in [10].

	Algorithm	Average Avalanche (%)
0	sha256	50.160938
1	sha3_256	49.607813
2	blake2b	49.933984

Figure 5. Avalanche Effect Comparison Results

Figure 12 present the avalanche effect results obtained from 500 testing iterations. SHA-256 produced an average avalanche effect of 50.160938%, while SHA3-256 and BLAKE2b produced average values of 49.607813% and 49.933984%, respectively. All measured values remained very close to the theoretical ideal of 50%, indicating strong diffusion characteristics and effective sensitivity to input changes.

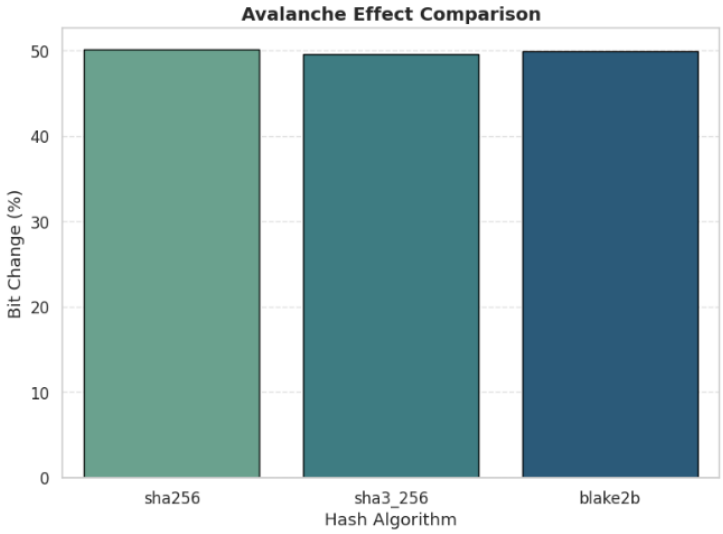


Figure 6. Avalanche Effect Comparison Chart

Specifically, using the precise data from the latest experiments, SHA-256 recorded an average value of 50.160938%, SHA3-256 recorded 49.607813%, and BLAKE2b recorded 49.933984%. The next security tests focused on resistance against collision and preimage attacks. The Collision Resistance Test (Figure 14) shows that for 2000 unique Smart Grid data rows, all three algorithms (SHA-256, SHA3-256, and BLAKE2b) each produced 2000 unique hashes, with 0 duplicates (collisions) found for all of them.

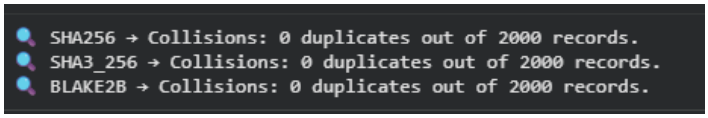


Figure 7. Collision Resistance Results

The Preimage Resistance Test, which is demonstrative according to the methodology, also showed positive results. A limited brute-force attempt (testing 1,000,000 inputs) failed to find the preimage (original input) for the target hash of all three algorithms.

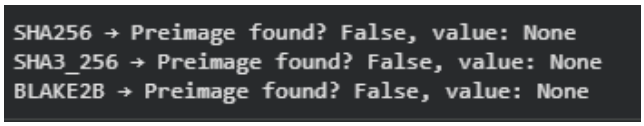


Figure 8. Preimage Resistance Results

Figure 8 presents the results of the demonstrative preimage resistance test conducted using a limited brute-force approach. For each evaluated algorithm (SHA-256, SHA3-256, and BLAKE2b), no preimage was found after 1,000,000 brute-force attempts. The output for all three algorithms shows "Preimage found? False" and "value: None", indicating that no matching original input was identified during the testing process. The same outcome was observed consistently across the three evaluated hash algorithms.

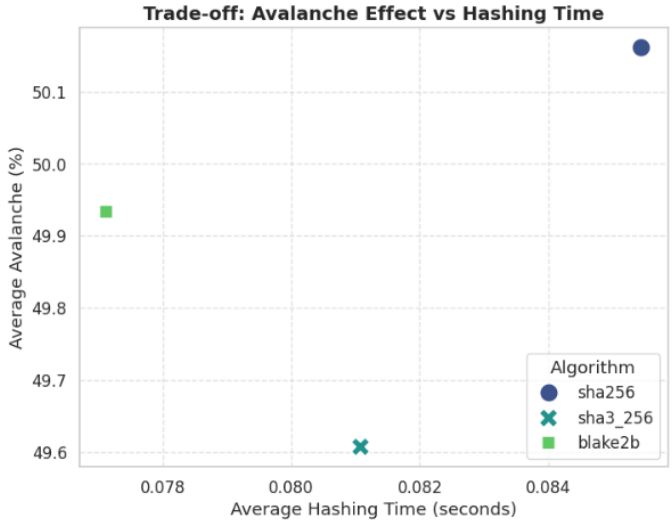


Figure 9. Trade-off: Avalanche vs Hashing Time

Figure 16 illustrates the relationship between average hashing time and average avalanche effect for the three evaluated algorithms. SHA-256 recorded an average hashing time of 0.085451 seconds and an average avalanche effect of 50.160938%. SHA3-256 recorded an average hashing time of 0.081075 seconds and an average avalanche effect of 49.607813%. BLAKE2b recorded an average hashing time of 0.077112 seconds and an average avalanche effect of 49.933984%. The figure shows the distribution of the three algorithms based on the two evaluation metrics, namely computational efficiency and avalanche effect performance.

https://doi.org/10.31849/digitalzone.v17i1.30369

	Algorithm	Average Avalanche (%)	Average Hashing Time (seconds)	Std Dev Time (seconds)	Relative Efficiency (%)
0	sha256	50.160938	0.085451	0.012948	0.000000
1	sha3_256	49.607813	0.081075	0.009947	5.120949
2	blake2b	49.933984	0.077112	0.005383	9.757979

Figure 10. Final Results Summary

Figure 10 summarizes the experimental results obtained from all evaluation stages. For SHA-256, the average hashing time was 0.085451 seconds with a standard deviation of 0.012948 seconds and an average avalanche effect of 50.160938%. SHA3-256 achieved an average hashing time of 0.081075 seconds with a standard deviation of 0.009947 seconds and an average avalanche effect of 49.607813%. BLAKE2b achieved an average hashing time of 0.077112 seconds with a standard deviation of 0.005383 seconds and an average avalanche effect of 49.933984%.

3.1 Discussion

The benchmarking results indicate that BLAKE2b achieved the lowest average hashing time among the evaluated algorithms, followed by SHA3-256 and SHA-256. Across the repeated experimental runs, BLAKE2b also exhibited the lowest standard deviation, suggesting relatively consistent execution performance within the testing environment. These findings are generally consistent with previous studies that reported favorable computational efficiency for BLAKE-family algorithms while maintaining cryptographic functionality comparable to SHA-based alternatives [10],[11]. From a Smart Grid perspective, computational efficiency is an important consideration because integrity verification may be performed continuously on large volumes of operational data. Lower hashing latency may contribute to reducing processing overhead and improving responsiveness in data-intensive environments. However, the performance differences observed in this study remain relatively small and were obtained using a software-based experimental setup with simulated Smart Grid data. Therefore, the results should be interpreted as comparative observations within the scope of the experimental environment rather than as definitive evidence of superiority in real-world Smart Grid deployments. The avalanche-effect analysis demonstrates that all three algorithms exhibit strong diffusion properties. The average avalanche values obtained were 50.160938% for SHA-256, 49.607813% for SHA3-256, and 49.933984% for BLAKE2b. All values remain close to the theoretical ideal of 50%, indicating that minor modifications to the input data produce substantial changes in the resulting hash output. Although BLAKE2b achieved the value closest to the theoretical threshold in this experiment, the differences among the evaluated algorithms are relatively small. Consequently, these results should not be interpreted as evidence of substantial cryptographic superiority but rather as confirmation that all three algorithms provide effective sensitivity to input modifications and are suitable for integrity-verification purposes. The collision and preimage experiments further illustrate the integrity-preserving characteristics of cryptographic hash functions. No collisions were observed within the evaluated dataset, and the demonstrative preimage tests did not successfully recover the original inputs. Nevertheless, these evaluations should be regarded as conceptual demonstrations rather than formal cryptographic validation. The dataset size and experimental scope are insufficient to establish collision resistance or preimage resistance in the cryptographic sense, as such properties are supported primarily by theoretical design and extensive community analysis rather than small-scale experimental verification. Comparison with previous studies indicates that the present findings generally support existing literature regarding the efficiency advantages often associated with BLAKE-family algorithms. However, unlike many generic benchmarking studies, this work evaluates algorithm behavior using Smart Grid-inspired transactional data. This contextual approach provides additional insight into the relationship between cryptographic performance and data-integrity requirements in continuously operating environments. Even so, the results remain limited to the characteristics of the simulated

dataset and should not be generalized directly to operational Smart Grid infrastructures. Several limitations should be acknowledged. First, the experiments were conducted using a synthetic dataset consisting of 2,000 Smart Grid-inspired records rather than real operational Smart Grid data. Second, the evaluation was performed within a Google Colab environment, which may introduce variability in runtime performance. Third, the collision and preimage experiments were designed for demonstration purposes and do not constitute comprehensive cryptographic security assessments. Future research should consider larger datasets, real-world Smart Grid environments, hardware-level benchmarking, communication-overhead analysis, and broader cybersecurity evaluation scenarios to further validate and extend the findings of this study

4. Conclusions

This study successfully conducted an experimental analytical investigation to compare the performance and security of three cryptographic hash algorithms SHA-256, SHA3-256, and BLAKE2b within the context of Smart Grid simulation data[10]. Based on quantitative testing of 2000 data rows, BLAKE2b proved to be the highest-performing algorithm, recording an average time of 0.077112 seconds with a standard deviation of 0.005383, indicating high consistency, and a relative efficiency of 9.757979% compared to SHA-256. The Avalanche Effect of BLAKE2b reached 49.933984%, closest to the ideal 50% value, demonstrating high sensitivity to input changes and optimal bit-level security. SHA3-256, despite being a newer hash standard, showed an average time of 0.081075 seconds (Std Dev 0.009947) and an Avalanche Effect of 49.607813%, making it slower and further from the ideal value compared to BLAKE2b. SHA-256, used as the baseline, recorded an average time of 0.085451 seconds (Std Dev 0.012948) and an Avalanche Effect of 50.160938%, remaining secure and sensitive to data changes but with lower computational efficiency than BLAKE2b. The experimental results indicate that BLAKE2b achieves the lowest average hashing time with relatively stable performance across repeated runs, while SHA3-256 and SHA-256 exhibit slightly higher execution times. In terms of avalanche effect, all three algorithms produce values close to the theoretical ideal of 50%, confirming strong diffusion characteristics and sensitivity to input changes. These findings suggest that the evaluated algorithms maintain fundamental cryptographic properties required for data integrity. However, the observed performance differences occur at a relatively small scale and should be interpreted cautiously. The experiments are conducted using a limited synthetic dataset (2,000 records) in a cloud-based environment, and the collision and preimage tests are demonstrative in nature. Therefore, these results do not constitute a comprehensive evaluation of cryptographic security nor a direct representation of real-world Smart Grid deployment conditions. Within the scope of this study, BLAKE2b demonstrates a favorable balance between computational efficiency and diffusion consistency. Nevertheless, this observation should be considered as a preliminary comparative indication rather than a definitive recommendation for Smart Grid implementation. The practical selection of a hash algorithm depends on various factors, including system architecture, hardware constraints, communication overhead, and integration with broader cybersecurity mechanisms. Future research is recommended to extend this work by incorporating real-world Smart Grid datasets, larger-scale distributed environments, and hardware-level performance evaluation. Additionally, further investigation into network latency, throughput, and resilience under cyberattack scenarios would provide a more comprehensive understanding of the applicability of cryptographic hash functions in Smart Grid systems

References

[1] T. L. Imam Riadi, Rusydi Umar, “Telematika Smart Payment Application Security Optimization from Cross-Site Scripting (XSS) Attacks Based on Blockchain Technology,” *Telematika*, vol. 14, no. 2, pp. 74–85, 2021, doi:

https://doi.org/10.35671/telematika.v14i2.1221.

[2] M. S. L. Muhammad Hakim Sadiq, Abdul Wahid, "Implementation of One-Time Password and SHA-3 Algorithm on the Lab Inventory Website of the Department of Informatics and Computer Engineering," *IOTA*, vol. 05, 2025, doi: https://doi.org/10.31763/iota.v5i2.914.

[3] S. F. Muhammad Waseem, Muhammad Adnan Khan Intisar Ali Sajjad and Pierluigi Siano , Arman Goudarzi, "Incorporation of Blockchain Technology for Different Smart Grid Applications : Architecture , Prospects , and Challenges," *MDPI*, vol. 16, 2023, doi: https://doi.org/10.3390/en16020820.

[4] M. A. Muhammad Tanveer, Abd Ullah Khan, Habib Shah, Ahmed Alkhayyat, Shehzad Ashraf Chaudhry, "ARAP-SG : Anonymous and Reliable Authentication Protocol for Smart Grids," *IEEE Access*, vol. 9, pp. 143366–143377, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3121291.

[5] E. A.-C. Mohsen Hatami, Qian Qu Yu Chen, Javad Mohammadi, Erik Blasch, "ANCHOR-Grid: Authenticating Smart Grid Digital Twins Using Real-World Anchors," *MDPI*, vol. 25, pp. 0–25, 2024, doi: https://doi.org/10.3390/s25102969.

[6] Y. B. Hind EL Makhtoum, "An improved IOT Authentication Process based on Distributed OTP and Blake2," *IJWMT*, vol. 11, no. 5, pp. 1–8, 2021, doi: https://doi.org/10.5815/ijwmt.2021.05.01.

[7] S. P. ANGGRAENI, "PENERAPAN ALGORITMA KRIPTOGRAFI SHA-256 DAN TEKNOLOGI BLOCKCHAIN UNTUK KEAMANAN CITRA DIGITAL," *UNISSULA*, vol. 11, no. 1, pp. 1–14, 2025, [Online]. Available: https://repository.unissula.ac.id/40125/.

[8] S. Adilah, Rumani, Marisa, and Paryasto, "Implementasi Kriptosystem menggunakan metode Algoritma ECC dengan Fungsi Hash SHA-256 pada sistem ticketing online," *Univ. Telkom*, vol. 4, no. 3, pp. 4138–4146, 2017, [Online]. Available: https://openlibrarypublications.telkomuniversity.ac.id/index.php/engineering/article/view/5471.

[9] V. N. Maulidya, "Implementasi Algoritma SHA-3 dan Mceliece dengan kode hamming untuk otentikasi dokumen berbasis Digital Signature," *UIN Malang*. 2025, [Online]. Available: http://etheses.uin-malang.ac.id/76345/.

[10] J. S. G. Sinaga, Nehemia Sitorus, and Steven Lukas Samosir, "Analisis Kinerja Algoritma Hash pada Keamanan Data: Perbandingan Antara SHA-256, SHA-3, dan Blake2," *J. QUANCOM QUANTUM Comput. J.*, vol. 2, no. 2, pp. 9–16, Dec. 2024, doi: https://doi.org/10.62375/jqc.v2i2.432.

[11] Z. Abdullah Jasim and A. Kadhim Hadi, "Optimizing Blockchain Network Performance Using Blake3 Hash Function in POS Consensus Algorithm," *IEEE Access*, vol. 13, no. March, pp. 44760–44774, 2025, doi: https://doi.org/10.1109/ACCESS.2025.3546723.

[12] A. H. T. ZENG ZENG, MEIYA DONG, WEIWEI MIAO, MINGMING ZHANG, "A Data-Driven Approach for Blockchain-Based Smart Grid System," *IEEE Access*, vol. 9, pp. 70061–70070, 2021, doi: https://doi.org/10.1109/ACCESS.2021.3076746.

[13] Y. Guo, Z. Wan, and X. Cheng, "When blockchain meets smart grids: A comprehensive survey," *High-Confidence Comput.*, vol. 2, no. 2, pp. 0–1, 2022, doi: https://doi.org/10.1016/j.hcc.2022.100059.

[14] T. Cioara, C. Pop, R. Zanc, I. Anghel, M. Antal, and I. Salomie, "Smart grid management using blockchain: Future scenarios and challenges," *Proc. - RoEduNet IEEE Int. Conf.*, vol. 2020-Decem, 2020, doi: https://doi.org/10.1109/RoEduNet51892.2020.9324874.

[15] A. Agung Aryasatya Daniswara and I. K. Dwi Nuryana, "Data Preprocessing Pola Pada Penilaian Mahasiswa Program Profesi Guru," *J. Informatics Comput. Sci.*, vol. 05, pp. 97–100, 2023, doi: https://doi.org/10.26740/jinacs.v5n01.p97-100.

[16] O. P. Hafizh Fianto Putra, Wirawan, "Penerapan Blockchain dan Kriptografi untuk Keamanan Data pada Jaringan Smart Grid," *Jural Tek. its*, vol. 8, no. 1, 2019, doi: https://doi.org/10.12962/j23373539.v8i1.38525.

[17] F. B. Fajrin, Ahmad Mifta, "Analisis Performa Algoritma BLAKE2b dan SHA-256 pada Implementasi Blockchain," *KESATRIA*, vol. 6, no. 2, pp. 451–460, 2025, doi: https://doi.org/10.30645/kesatria.v6i2.588.

[18] J. Sugier, "Reducing Power of BLAKE3 implementations with dedicated FPGA resources," *Int. J. Electron. Telecommun.*, vol. 71, no. 3, pp. 1–7, 2025, doi: https://doi.org/10.24425/ijet.2025.153622.

[19] Nilda Aulia, "Prediksi Harga Ethereum Berdasarkan Informasi Blockchain Menggunakan Metode Long Short Term Memory," *Univ. Islam Indones.*, no. 9, pp. 1689–1699, 2020, [Online]. Available: https://library.uii.ac.id/repositories/#gsc.tab=0.

[20] D. H. Sinaga, R. Rifai, O. Sasue, and H. D. Hutahaean, "Pemanfaatan Energi Terbarukan Dengan Menerapkan Smart Grid Sebagai Jaringan Listrik Masa Depan," *zetroem*, vol. 03, pp. 11–17, 2021, doi: https://doi.org/10.36526/ztr.v3i1.1251.

[21] G. Aldabbagh, O. Bamasag, L. Almasari, R. Alsaidalani, A. Redwan, and A. Alsaggaf, "Blockchain for Securing Smart Grids," *Int. J. Comput. Sci. Netw. Secur.*, vol. 21, no. 4, pp. 255–263, 2021, doi: https://doi.org/10.22937/IJCSNS.2021.21.4.31.

[22] I. Riadi, R. Umar, I. Busthomi, and A. Wirawan, "Block-hash of blockchain framework against man-in-the- middle attacks," *Register*, vol. 8, no. January, pp. 1–9, 2022, doi: https://doi.org/10.26594/register.v8i1.2190.