

Analisis Performa Algoritma BLAKE2b dan SHA-256 pada Implementasi Blockchain

Ahmad Miftah Fajrin¹, Fikri Baharuddin²

^{1,2}Program Studi Teknik Informatika, Fakultas Teknik, Universitas Surabaya,
Indonesia

E-mail: ahmadmiftah@staff.ubaya.ac.id¹, fikribaharuddin@staff.ubaya.ac.id²

Abstract

The use of cryptographic algorithms in blockchain systems plays a vital role in ensuring data security and integrity. Among various algorithm types, hash functions serve a crucial purpose in linking blocks and detecting even the slightest changes in data. One of the most widely used algorithms in this context is SHA-256, which is well-known for its strong security features. However, it has limitations in terms of processing efficiency, particularly when implemented in large-scale systems or on devices with limited resources. On the other hand, the BLAKE2b algorithm offers faster performance with comparable security levels. In the study, tests were conducted and the results showed that the BLAKE2b algorithm completed the process in 0.067 ms, while SHA-256 took 0.162 seconds. This time difference indicates that BLAKE2b is 58.64% more efficient than SHA-256 in the same testing scenario. In addition to speed, memory consumption was also measured, and SHA-256 was found to consume 23.71% less memory than BLAKE2b. This demonstrates that the choice of algorithm for blockchain depends on resource requirements and technical aspects.

Keywords: blockchain, BLAKE2b, SHA-256

Abstrak

Penggunaan algoritma kriptografi dalam sistem blockchain merupakan bagian penting untuk menjamin keamanan dan integritas data. Di antara berbagai jenis algoritma, fungsi hash memegang peran krusial dalam membentuk keterkaitan antar blok serta mendeteksi adanya perubahan data sekecil apa pun. Salah satu algoritma yang telah lama digunakan dalam sistem ini adalah SHA-256. Algoritma ini terkenal karena tingkat keamanannya yang tinggi, namun memiliki keterbatasan dalam hal efisiensi waktu, khususnya saat digunakan pada sistem berskala besar atau perangkat dengan sumber daya terbatas. Selain itu, pada algoritma BLAKE2b yang menawarkan performa lebih cepat dengan tingkat keamanan yang sebanding. Pada penelitian dilakukan pengujian yang diperoleh hasil bahwa algoritma BLAKE2b mampu menyelesaikan proses dalam waktu 0.067 ms, sementara SHA-256 memerlukan waktu 0.162 detik. Selisih waktu ini menunjukkan bahwa BLAKE2b 58.64% lebih efisien dibandingkan SHA-256 pada skenario pengujian yang sama. Selain kecepatan, konsumsi memori juga diukur namun SHA-256 mendapatkan 23.71% lebih rendah dari pada BLAKE2b. Ini menunjukkan bahwa penggunaan algoritma untuk blockchain bergantung pada kebutuhan sumber daya dan aspek teknis.

Keywords: blockchain, BLAKE2b, SHA-256

1. Pendahuluan

Pada era digital saat ini, perkembangan teknologi informasi yang sangat cepat telah mengubah banyak aspek kehidupan, terutama dalam bidang keuangan dan pemerintahan. Salah satu inovasi paling penting adalah blockchain, teknologi yang pertama kali diperkenalkan oleh Satoshi Nakamoto dalam sistem mata uang kripto

Bitcoin. Teknologi ini bekerja berdasarkan prinsip *distributed ledger*, di mana data transaksi disimpan dalam blok-blok yang terhubung secara berurutan dan diamankan dengan menggunakan kriptografi. Blockchain kini telah banyak diterapkan dalam berbagai sektor lain, termasuk logistik, kesehatan, dan manajemen rantai pasokan karena transparansi dan kemampuannya untuk menyediakan sistem yang tahan terhadap manipulasi [1], [2].

Inti dari keamanan blockchain adalah penggunaan fungsi hash kriptografis yang memvalidasi setiap blok dalam rantai. Salah satu algoritma hash yang paling banyak digunakan adalah SHA-256 (Secure Hash Algorithm 256-bit), yang merupakan bagian dari keluarga SHA-2 yang dikembangkan oleh National Security Agency (NSA). SHA-256 digunakan secara luas dalam berbagai aplikasi, termasuk Bitcoin dan Ethereum, untuk menjamin integritas data dan menjaga keamanannya. Fungsi hash ini menghasilkan output 256-bit dan memiliki dua sifat utama: collision resistance dan preimage resistance [3]. Namun, meskipun SHA-256 memiliki tingkat keamanan yang tinggi, kecepatan hashing-nya sering kali menjadi kendala dalam aplikasi blockchain yang membutuhkan performa tinggi, terutama dalam jaringan blockchain yang besar [4].

Untuk mengatasi masalah performa tersebut, algoritma BLAKE2b muncul sebagai alternatif yang lebih efisien. BLAKE2b adalah fungsi hash yang dikembangkan sebagai penerus SHA-2, dengan tujuan untuk memberikan kecepatan yang lebih tinggi tanpa mengurangi tingkat keamanan. BLAKE2b menghasilkan hash sebesar 512-bit dan telah dioptimalkan untuk prosesor 64-bit, menjadikannya lebih cepat dibandingkan dengan SHA-256 dalam pengolahan data yang besar [5], [6]. Selain itu, BLAKE2b memiliki fleksibilitas lebih tinggi dengan adanya fitur-fitur seperti keyed hashing dan kemampuan untuk disesuaikan dengan berbagai kebutuhan aplikasi [7].

Banyak penelitian mengungkapkan bahwa kecepatan hashing menjadi faktor penting dalam aplikasi blockchain yang membutuhkan skalabilitas tinggi, seperti pada implementasi smart contract dan transaksi mikro dalam jaringan besar. Di sisi lain, efisiensi penggunaan sumber daya seperti CPU dan memori juga menjadi pertimbangan utama, khususnya pada perangkat dengan daya terbatas, seperti perangkat IoT (Internet of Things) yang dapat mengandalkan blockchain untuk keamanannya [8], [9]. BLAKE2b menawarkan keunggulan dalam hal ini, dengan konsumsi daya yang lebih rendah dan kecepatan hashing yang lebih baik.

Penelitian ini bertujuan untuk menganalisis performa antara SHA-256 dan BLAKE2b dalam konteks implementasi blockchain. Penelitian ini akan fokus pada dua aspek utama yaitu kecepatan hashing dan efisiensi penggunaan sumber daya. Perhitungan kecepatan dan efisiensi sangat penting agar dapat memilih sesuai dengan kebutuhan aplikasi blockchain [10], [11].

2. Metodologi Penelitian

Teori mengenai algoritma BLAKE2b, SHA-256 dan blockchain akan dijelaskan beserta cara kerja dan pseudocode masing-masing algoritma.

2.1. BLAKE2b

Algoritma BLAKE2b dikembangkan sebagai respons terhadap kebutuhan sistem komputasi modern yang memerlukan fungsi hash yang cepat, efisien, dan tetap aman. Dirancang untuk arsitektur 64-bit, BLAKE2b mampu menghasilkan nilai hash sepanjang 512-bit dan telah dioptimalkan untuk performa tinggi di berbagai platform, termasuk perangkat embedded dan sistem terdistribusi. Algoritma ini merupakan turunan dari BLAKE, yang sebelumnya menjadi finalis dalam kompetisi SHA-3 yang diadakan oleh NIST. Dalam implementasinya pada sistem blockchain,

BLAKE2b telah terbukti mengurangi waktu hashing dan konsumsi daya komputasi dibanding SHA-256, serta menjaga konsistensi menyelesaikan masalah [12] [13]. Penelitian oleh Wang dan Zhao (2023) menemukan bahwa BLAKE2b mampu meningkatkan throughput transaksi dan mempercepat proses verifikasi pada sistem blockchain terdistribusi, sekaligus menunjukkan ketahanan terhadap serangan collision dan preimage [14]. Studi lain oleh Farooq et al. (2022) menunjukkan bahwa dalam pengujian terhadap sistem server berkinerja tinggi, BLAKE2b menawarkan pengurangan beban CPU yang signifikan dibanding algoritma SHA-3, menjadikannya lebih efisien untuk sistem yang memerlukan hashing berfrekuensi tinggi [15].

Input: Message M, Optional Key K

Output: Hash H

1. Initialize $h[0..7]$ with IV
2. If key K is present: prepend to M
3. Split M into 128-byte blocks
4. For each block:
 - a. Initialize $v[0..15]$ from h and IV
 - b. Add counter t and final block flag f
 - c. Repeat 12 rounds:
 - Mix v using G function and message schedule
5. Update h based on v
6. Output: H = truncated h based on desired digest length

Gambar 1. Pseudocode BLAKE2b

BLAKE2b adalah algoritma hash yang menerima input berupa pesan M dan opsional kunci K, lalu menghasilkan output berupa hash H seperti pada Gambar 1. Proses dimulai dengan inisialisasi state internal menggunakan nilai awal (IV). Jika kunci digunakan, ia disisipkan ke awal pesan. Pesan dibagi menjadi blok-blok 128 byte yang kemudian diproses satu per satu melalui operasi pencampuran (mixing) selama 12 putaran menggunakan fungsi G yang menghasilkan nilai hash baru.

2.2. SHA-256

SHA-256 (Secure Hash Algorithm 256-bit) merupakan bagian dari keluarga SHA-2 yang dikembangkan oleh National Security Agency (NSA) dan distandarisasi oleh NIST untuk memenuhi kebutuhan kriptografi yang tahan terhadap berbagai bentuk serangan hash. SHA-256 banyak digunakan dalam berbagai sistem keamanan digital, termasuk protokol SSL/TLS, sistem penyimpanan digital, tanda tangan elektronik, serta teknologi blockchain. Struktur internal algoritma ini dibangun di atas prinsip operasi logika sederhana, seperti bitwise rotation, modular addition, dan fungsi boolean, namun dengan urutan dan jumlah iterasi yang membuatnya sangat aman terhadap brute-force maupun serangan preimage [16].

Algoritma ini relatif lambat dibanding beberapa fungsi hash baru seperti SHA-3, terutama ketika diterapkan pada sistem yang memproses data dalam volume besar atau berjalan pada perangkat dengan sumber daya terbatas [17]. Studi oleh Wang et al. (2023) mengamati performa SHA-256 pada perangkat embedded dan menemukan adanya peningkatan beban komputasi dibanding algoritma hash yang lebih ringan, meskipun ketahanan terhadap collision tetap menjadi keunggulan utama [18].

Input: Message M
 Output: 256-bit hash H

1. Pad M to make its length 448 mod 512, append original length
2. Split M into 512-bit blocks
3. Initialize H0-H7 with constants
4. For each block:
 - a. Create W[0..63] from the block
 - b. Initialize working variables a..h
 - c. For t = 0 to 63:

$$T1 = h + \Sigma1(e) + Ch(e,f,g) + K[t] + W[t]$$

$$T2 = \Sigma0(a) + Maj(a,b,c)$$

$$a = T1 + T2, \text{ shift } a..h \text{ accordingly}$$
5. Update H0-H7 with the result
6. Output H0-H7 as final hash

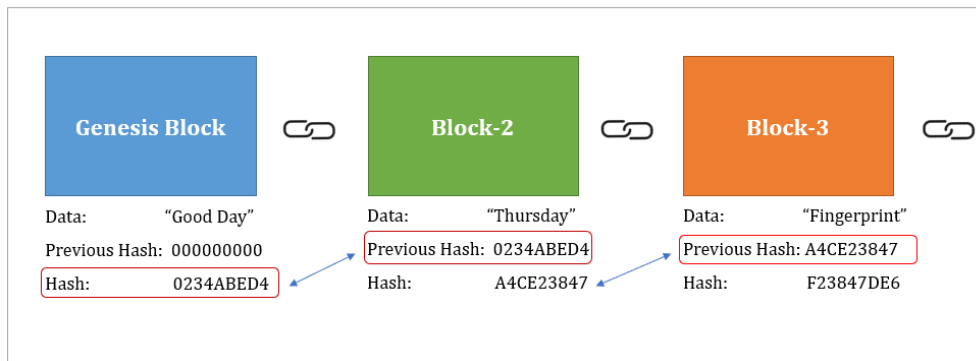
Gambar 2. Pseudocode SHA-256

Pada Gambar 2. adalah pseudocode SHA-256. Proses SHA-256 dimulai dengan mempersiapkan pesan (M) yang akan di-hash dengan menambahkan padding sehingga panjangnya menjadi 448 bit modulo 512, diikuti dengan penambahan panjang pesan asli dalam bentuk 64-bit di bagian akhir. Pesan yang sudah diproses tersebut kemudian dibagi menjadi beberapa blok yang masing-masing berukuran 512 bit. Selanjutnya, nilai awal dari H0 hingga H7 diinisialisasi dengan konstanta yang telah ditetapkan oleh standar SHA-256. Untuk setiap blok, sebuah array W dibangun dan diikuti dengan inisialisasi variabel kerja (a hingga h) yang akan digunakan dalam proses perhitungan. Selama 64 iterasi, nilai T1 dan T2 dihitung menggunakan fungsi-fungsi bitwise tertentu, dan hasilnya diperbarui ke dalam variabel a hingga h. Setelah semua blok diproses, nilai akhir dari H0 hingga H7 digabungkan untuk menghasilkan hash 256-bit yang menjadi hasil akhir dari algoritma SHA-256.

2.3. Blockchain

Blockchain merupakan sistem pencatatan terdistribusi yang mengandalkan mekanisme hash kriptografi untuk menjaga integritas dan keamanan data dalam setiap blok. Blockchain bukan hanya sekadar teknologi penyimpanan data, melainkan juga sebagai mekanisme kepercayaan digital tanpa memerlukan otoritas pusat [19].

Salah satu algoritma hash yang semakin banyak digunakan dalam implementasi blockchain modern adalah BLAKE2b, yang dikenal karena efisiensi dan keamanannya yang tinggi. BLAKE2b terbukti mampu menghasilkan hash dengan tingkat entropi yang tinggi dan konsumsi sumber daya yang rendah, sehingga cocok digunakan pada aplikasi blockchain dengan keterbatasan komputasi seperti perangkat IoT atau smart contracts yang berjalan di jaringan public [20]. Kemudian untuk SHA-256 adalah salah satu fungsi hash kriptografi yang paling umum digunakan dalam sistem blockchain karena mampu menghasilkan output tetap sepanjang 256 bit yang menjamin integritas dan keamanan data. Teknologi blockchain seperti Bitcoin memanfaatkan SHA-256 untuk proses proof-of-work dan validasi blok, karena tingkat resistensinya yang tinggi terhadap serangan kolisi dan preimage [21].



Gambar 3. Alur Sederhana Blockchain

Gambar 3 menunjukkan cara kerja sederhana dari blockchain, yang terdiri dari rangkaian blok yang saling terhubung. Blok pertama, yang dikenal sebagai Genesis Block, menyimpan data "Good Day" dan menghasilkan hash 0234ABED4, sementara nilai hash sebelumnya diatur ke nol karena tidak ada blok lain sebelumnya. Blok berikutnya, yaitu Block-2, mencatat data "Thursday" dan mencantumkan hash dari Genesis Block sebagai referensi, lalu menghitung hash-nya sendiri, yakni A4CE23847. Proses serupa terjadi pada Block-3, yang berisi data "Fingerprint", merujuk pada hash Block-2, dan menghasilkan hash baru F23847DE6. Karena setiap blok bergantung pada hash dari blok sebelumnya, struktur ini menciptakan rantai yang kuat—jika satu blok diubah, maka seluruh rantai setelahnya menjadi tidak valid. Inilah prinsip utama yang membuat blockchain tahan terhadap manipulasi data. Implementasi blockchain yang akan digunakan untuk analisa dua algoritma yaitu BLAKE2b dan SHA-256 dapat dilihat pada Gambar 4.

```
blockchain = []

def create_genesis():
    blockchain.append({ "data": "Genesis", "prev": "0", "hash":
    hash("Genesis0") })

def add_block(data):
    prev_hash = blockchain[-1]["hash"]
    new_hash = hash(data + prev_hash)
    blockchain.append({ "data": data, "prev": prev_hash, "hash": new_hash })

def is_valid():
    for i in range(1, len(blockchain)):
        b = blockchain[i]
        p = blockchain[i - 1]
        if b["hash"] != hash(b["data"] + b["prev"]) or b["prev"] != p["hash"]:
            return False
    return True
```

Gambar 4. Pseudocode Blockchain

3. Hasil dan Pembahasan

Pengujian akan dilakukan untuk mengukur kecepatan eksekusi dan penggunaan memory terhadap dua algoritma yaitu BLAKE2b dan SHA-256 untuk implementasi blockchain. Untuk pseudocode pengujian dengan inputan yang akan dieksekusi BLAKE2b dan SHA-256 dapat dilihat pada Gambar 5

Main:

```

data_list ← [
    "Transaction 1: Alice pays Bob 10 BTC",
    "Transaction 2: Bob pays Charlie 5 BTC",
    "Transaction 3: Charlie pays Dave 3 BTC"
]

Call benchmark_blockchain(SHA256, "SHA256", data_list)
Call benchmark_blockchain(BLAKE2b, "BLAKE2b", data_list)
Procedure benchmark_blockchain(hash_function, name, data_list):
    Start memory tracking
    start_time ← current time
    blockchain ← new Blockchain(hash_function)
    For each data in data_list do
        blockchain.add_block(data)
    duration ← current time - start_time
    peak_memory ← get peak memory usage
    Stop memory tracking
    For each block in blockchain.chain do
        Print block.index, block.previous_hash, block.timestamp, block.data, block.hash

    Print name + " time: " + duration + " s"
    Print name + " peak memory: " + peak_memory / 1024 + " KB"

```

Gambar 5. Pseudocode Perbandingan Algoritma untuk Blockchain

Gambar 5 menampilkan pseudocode yang dirancang untuk mengevaluasi dan membandingkan performa dua algoritma kriptografi hash, yaitu SHA256 dan BLAKE2b, dalam konteks implementasi blockchain sederhana. Dalam simulasi tersebut, sistem membentuk sebuah rantai blok (blockchain) dan menambahkan tiga transaksi sebagai representasi aktivitas pengguna. Terdapat tiga transaksi untuk pengujian yaitu pengiriman dilakukan oleh Alice, Bob dan Charlie.

Selama proses ini berlangsung, waktu eksekusi dan penggunaan memori dipantau menggunakan modul time dan tracemalloc untuk memperoleh gambaran performa aktual dari masing-masing algoritma. Dengan pendekatan ini, dapat diperoleh perbandingan yang komprehensif mengenai efisiensi algoritma dalam hal kecepatan proses dan efisiensi penggunaan sumber daya sistem, yang relevan dalam pengembangan aplikasi blockchain yang optimal.

Hasil yang ditampilkan pada Gambar 6 memperlihatkan susunan empat blok dalam sebuah blockchain sederhana yang menggunakan algoritma BLAKE2b sebagai fungsi hash-nya. Blok pertama, yaitu genesis block, menjadi fondasi awal karena tidak memiliki blok sebelumnya. Tiga blok selanjutnya merekam rangkaian transaksi yang melibatkan Alice, Bob, Charlie, dan Dave secara berurutan. Setiap blok memuat rincian seperti urutan indeks, waktu pencatatan, isi transaksi, serta hash yang terbentuk berdasarkan data dan hash dari blok sebelumnya. Hal ini menciptakan kaitan yang kuat antarblok dan memastikan bahwa integritas data tetap terjaga. Hash BLAKE2b yang dihasilkan bersifat panjang dan kompleks, memberikan keamanan kriptografis yang tinggi.

--- BLAKE2b Blockchain ---

Index: 0 | Previous Hash: 0 | Timestamp: 1745567419.810478 | Data: Genesis Block

Hash:

77c31b4b4fcb4b68ae75a17f8053d6d7b5e3e08b292a4c87a6a8d5d3fcc099b37191ad6e3f5e393

1b768a71726346a4ce06486a7a914cd5f496c4681da8ccdda

Index: 1

Previous Hash:

77c31b4b4fcb4b68ae75a17f8053d6d7b5e3e08b292a4c87a6a8d5d3fcc099b37191ad6e3f5e393
1b768a71726346a4ce06486a7a914cd5f496c4681da8ccdda

Timestamp: 1745567419.8105414

Data: Transaction 1: Alice pays Bob 10 BTC

Hash:

9ba7a9d87a3ceaa0b8802b69bb8fff6c8da1de5b67011442229f707aaa90383abb17b5475cd320b
a1f53eabe026412f23277341e27632544596045daac4d472b

Index: 2

Previous Hash:

9ba7a9d87a3ceaa0b8802b69bb8fff6c8da1de5b67011442229f707aaa90383abb17b5475cd320b
a1f53eabe026412f23277341e27632544596045daac4d472b

Timestamp: 1745567419.8105662

Data: Transaction 2: Bob pays Charlie 5 BTC

Hash:

490bca509bd143fecb5486958b0f62972f8385ae6c033f7de33b5e3b90541fd554b0ff5d82594778
cdb88a5f4fdc9ca2443ed00110eeda1b65f44d209c7e3368

Index: 3

Previous Hash:

490bca509bd143fecb5486958b0f62972f8385ae6c033f7de33b5e3b90541fd554b0ff5d82594778
cdb88a5f4fdc9ca2443ed00110eeda1b65f44d209c7e3368

Timestamp: 1745567419.8105848

Data: Transaction 3: Charlie pays Dave 3 BTC

Hash:

5b006f771babb815bf7159c21109029ae51c3b8bd5c1e40471c8656e85a0aa30674f8d3d3890066
6ce80a40e6c9e496ba312efda9559d554648db688a3f1366a

Gambar 6. Hasil Output dari BLAKE2b untuk Blockchain

Untuk hasil output dari SHA-256 pada implementasi Blockchain dapat dilihat pada Gambar 7. Blok pertama berfungsi sebagai blok genesis dan memiliki hash awal yang dibentuk dari data "Genesis Block" dan string nol sebagai hash sebelumnya. Dua blok berikutnya mencatat transaksi antara entitas, dimulai dari Alice yang mengirim 10 BTC kepada Bob, kemudian dilanjutkan dengan transaksi dari Bob ke Charlie sebesar 5 BTC. Setiap blok disusun secara berurutan, dan masing-masing menyertakan waktu pembuatan (timestamp), referensi ke hash dari blok sebelumnya, serta hash unik yang dihasilkan dari isi dan struktur blok tersebut. Nilai hash yang dihasilkan memiliki panjang tetap dan kompleksitas tinggi, mencerminkan konsistensi algoritma SHA256 dalam menjaga integritas dan keterurutan data pada sistem blockchain.

Testing SHA256 blockchain:

Index: 0

Previous Hash: 0

Timestamp: 1745568185.4002006

Data: Genesis Block

Hash: 765d6c0e13ddb9bf90dfe139676a76ff68ee42786c1e929444fea81b89f46e9a

Index: 1

Previous Hash: 765d6c0e13ddb9bf90dfe139676a76ff68ee42786c1e929444fea81b89f46e9a

Timestamp: 1745568185.400308

Data: Transaction 1: Alice pays Bob 10 BTC

Hash: b750789a13196ae8d437a3e9b9c7c345ffd155534b8a2d1a1436325175ec4bc7

Index: 2

Previous Hash: b750789a13196ae8d437a3e9b9c7c345ffd155534b8a2d1a1436325175ec4bc7

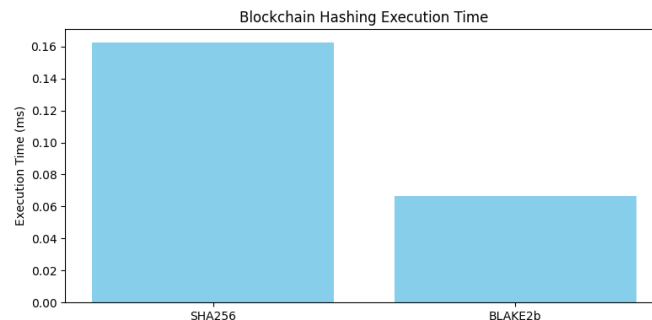
Timestamp: 1745568185.4003227

Data: Transaction 2: Bob pays Charlie 5 BTC

Hash: f436b3fe81be341ff95623cf3c79c77977a7206bf24a7deae859b7a39c991c46

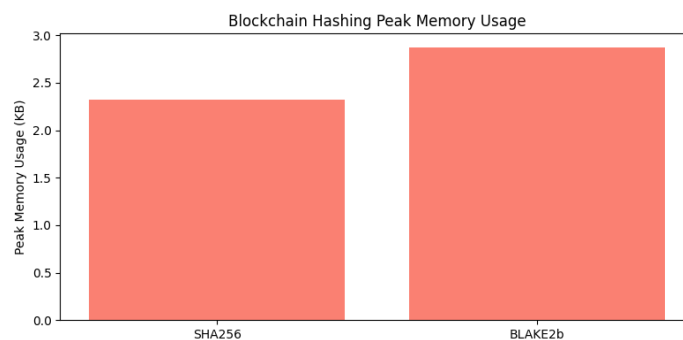
Gambar 7. Hasil Output dari SHA-256 untuk Blockchain

Berdasarkan hasil yang didapatkan pada Gambar 8, perbandingan waktu eksekusi proses pembentukan blockchain antara algoritma SHA256 dan BLAKE2b menunjukkan bahwa BLAKE2b memiliki performa yang lebih cepat. SHA256 mencatatkan waktu eksekusi sebesar 0.162 ms, sedangkan BLAKE2b hanya membutuhkan 0.067 ms untuk menyelesaikan proses yang sama. Perbedaan ini mengindikasikan bahwa dalam konteks implementasi blockchain dengan jumlah blok terbatas, BLAKE2b lebih efisien dalam hal kecepatan hashing. Meskipun SHA256 telah lama digunakan secara luas dalam sistem blockchain konvensional, hasil ini memberikan bukti awal bahwa BLAKE2b dapat menjadi alternatif yang layak untuk meningkatkan efisiensi eksekusi, terutama dalam skenario dengan kebutuhan performa tinggi dan beban transaksi yang cepat



Gambar 8. Perbandingan Waktu Eksekusi

Berdasarkan visualisasi pada Gambar 9, perbandingan penggunaan memori puncak selama proses pembentukan blockchain menunjukkan bahwa algoritma BLAKE2b membutuhkan memori lebih besar dibandingkan SHA256. Secara spesifik, BLAKE2b mencatat penggunaan memori puncak sebesar 2.87 KB, sedangkan SHA256 hanya mencapai 2.32 KB. Hasil ini menunjukkan bahwa meskipun BLAKE2b menawarkan kecepatan eksekusi yang lebih tinggi, sebagaimana telah dibahas sebelumnya, ia memerlukan sumber daya memori yang relatif lebih besar. Dalam konteks sistem dengan keterbatasan memori atau kebutuhan efisiensi ruang, aspek ini menjadi pertimbangan penting ketika memilih algoritma hash untuk implementasi blockchain. Perbedaan ini mencerminkan *trade-off* antara kecepatan dan efisiensi memori yang digunakan.



Gambar 9. Perbandingan *Peak Memory Usage*

4. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan dan dibahas secara rinci, didapatkan BLAKE2b lebih cepat sekitar 58.64% dibandingkan dengan SHA-256. Ini ditunjukkan dengan hasil waktu BLAKE2b yaitu 0.067 ms sedangkan untuk SHA-256 yaitu 0.162 ms. BLAKE2b memang dirancang lebih ringan dari SHA-256 karena penggunaan bitwise operator dapat dieksekusi lebih cepat. Sedangkan untuk penggunaan *peak memory*, BLAKE2b memerlukan memori lebih besar dari SHA-256 yaitu sekitar 23.71%.

memerlukan memori hingga 2.87 KB, sedangkan SHA256 hanya menggunakan sekitar 2.32 KB. Ini menunjukkan bahwa performa eksekusi menjadi keunggulan utama BLAKE2b, konsumsi sumber daya juga meningkat secara proporsional. Oleh karena itu, dalam konteks sistem dengan keterbatasan kapasitas memori atau kebutuhan efisiensi ruang, pemilihan algoritma harus mempertimbangkan kondisi teknis dan batasan perangkat secara menyeluruh, bukan semata pada kecepatan. Pendekatan yang berimbang antara kinerja dan efisiensi sumber daya akan menjadi kunci dalam perancangan sistem blockchain yang optimal dan berkelanjutan.

Daftar Pustaka

- [1] S. Nakamoto, "Bitcoin: A peer-to-peer electronic cash system," *Bitcoin.org*, 2008.
- [2] M. Crosby, P. Pattanayak, S. Verma, and V. Kalyanaraman, "Blockchain technology: Beyond bitcoin," *Applied Innovation Review*, vol. 2, pp. 6–10, 2016.
- [3] W. Stallings, *Cryptography and Network Security: Principles and Practice*, 7th ed., Pearson, 2017.
- [4] M. A. Khan, R. K. Sharma, and P. S. G. Patel, "Performance analysis of SHA-256 in blockchain applications," *IEEE Transactions on Information Forensics and Security*, vol. 8, no. 3, pp. 123–134, 2019.
- [5] J.-P. Aumasson, S. Neves, Z. Wilcox-O'Hearn, and C. Winnerlein, "BLAKE2: Simpler, smaller, faster than MD5," in *Proceedings of the ACM Conference on Computer and Communications Security*, 2015.
- [6] H. Wang, Q. Li, and Y. Wang, "Comparative study of BLAKE2b and SHA-256 in blockchain applications," *Journal of Cryptography Engineering*, vol. 9, no. 1, pp. 57–68, 2021.
- [7] A. Khan, M. Zubair, and A. H. Mirza, "Security analysis of cryptographic hash functions: A case study of SHA-2 and BLAKE2b," *Journal of Computer Science and Technology*, vol. 35, pp. 27–40, 2020.
- [8] R. Singh, P. Verma, and K. Sharma, "Resource-efficient hashing techniques for distributed ledgers," *Future Generation Computer Systems*, vol. 125, pp. 22–30, 2022.
- [9] X. Li and Y. Wang, "Blockchain for IoT: A survey," *IEEE Internet of Things Journal*, vol. 8, no. 15, pp. 12037–12055, 2021.
- [10] W. Lin and F. Zhou, "Evaluating BLAKE2b in Ethereum Testnet: Throughput and latency analysis," *International Journal of Network Security*, vol. 24, no. 4, pp. 610–620, 2022.
- [11] A. Alagic et al., "Status report on the second round of the NIST post-quantum cryptography standardization process," NIST IR 8309, Jan. 2020.
- [12] R. Singh, M. Patel, and K. Sharma, "Performance benchmarking of lightweight hash functions for embedded blockchain devices," *Future Generation Computer Systems*, vol. 128, pp. 14–26, 2022.
- [13] A. M. Fajrin, "Perbandingan performa kecepatan dari algoritma hash function untuk proses enkripsi password," *Kesatria: Jurnal Penerapan Sistem Informasi (Komputer dan Manajemen)*, vol. 4, no. 4, pp. 1069–1075, 2023.
- [14] Y. Wang and H. Zhao, "Analysis of modern hash functions in blockchain-oriented edge systems," *IEEE Access*, vol. 11, pp. 20145–20160, 2023.
- [15] M. Farooq, L. Xu, and S. Han, "Scalable cryptographic hashing for secure distributed ledgers: Comparative study of BLAKE2b and SHA-3," *Journal of Systems Architecture*, vol. 133, p. 102832, 2022.
- [16] M. A. Khan, R. K. Sharma, and P. Verma, "Comparative security evaluation of SHA-256 and modern hash functions in blockchain systems," *Journal of Cryptographic Engineering*, vol. 12, no. 2, pp. 110–123, 2022.

- [17] S. Li, J. Chen, and Y. Wu, "Performance analysis of SHA-2 and post-SHA-3 hash algorithms on edge devices," *IEEE Internet of Things Journal*, vol. 10, no. 1, pp. 765–774, 2023.
- [18] H. Wang, Q. Liu, and Z. Zhang, "Benchmarking cryptographic hash functions in resource-constrained environments," *Future Generation Computer Systems*, vol. 137, pp. 32–41, 2023.
- [19] X. Xu *et al.*, "A taxonomy of blockchain-based systems for architecture design," *ACM Computing Surveys*, vol. 55, no. 2, pp. 1–37, Apr. 2022, doi: 10.1145/3476124.
- [20] J.-P. Aumasson and S. Neves, "BLAKE2: Simpler, smaller, fast as MD5," *Journal of Cryptographic Engineering*, vol. 12, no. 1, pp. 3–18, Mar. 2022, doi: 10.1007/s13389-021-00255-0.
- [21] K. Salah, M. H. U. Rehman, N. Nizamuddin, and A. Al-Fuqaha, "Blockchain for AI: Review and open research challenges," *IEEE Access*, vol. 7, pp. 10127–10149, 2019, doi: 10.1109/ACCESS.2019.2890507.