Jumistik
E-ISSN: 2964-3953 P-ISSN: 2964-3888 Jurnal Manajemen Informatika, Sistem Informasi dan Teknologi Komputer

Vol.3.No.2 Desember 2024. Hal 270-276
Https://ojs.amiklps.ac.id

Analisis Kinerja Algoritma MD5, SHA-256, dan Base62 dalam Sistem Pemendekan URL

Novin Ardian Yulianto [1*], Mujib Ridwan[2], Achmad Teguh Wibowo[3]
[1,2,3]Sistem Informasi, Fakultas Sains dan Teknologi, UIN Sunan Ampel Surabaya
E-mail: 09020621039@student.uinsby.ac.id[1*], mujibrw@uinsa.ac.id[2], atw@uinsa.ac.id[3]

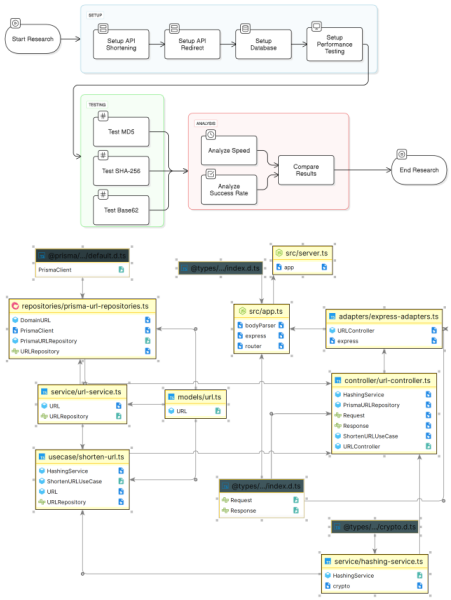
INFO ARTIKEL

Sejarah Artikel
Diterima : 22/11/2024
Direvisi : 25/11/2024
Diterbitkan : 01/12/2024

***Corresponding author**
09020621039@student.uinsby.ac.id

DOI: 10.70247/jumistik.vi2.113
Https://ojs.amiklps.ac.id

GRAPHICAL ABSTRACT



ABSTRACT

This study aims to compare the performance of three algorithms used in URL shortening systems: MD5, SHA-256, and Base62. These algorithms are commonly employed to generate efficient and secure short URLs in various online services. The testing was conducted to evaluate the speed, memory usage efficiency, and the uniqueness level of hash outputs produced by each algorithm. The experiments were carried out by building a URL shortening system based on Node.js and Express and performing performance testing using K6 to measure response times and throughput for each algorithm. The results show that Base62 excels in speed, while SHA-256 and MD5 offer higher uniqueness and security. However, increased memory usage and computational time for SHA-256 make it less efficient compared to Base62 in scenarios with high request volumes.

Keywords: Base62, MD5, SHA-256 , Performance, Uniqueness, URL Shortener.

ABSTRAK

Penelitian ini bertujuan untuk membandingkan kinerja tiga algoritma yang digunakan dalam sistem pemendekan URL: MD5, SHA-256, dan Base62. Algoritma-algoritma ini umumnya digunakan untuk menghasilkan URL pendek yang efisien dan aman di berbagai layanan online. Pengujian dilakukan untuk mengevaluasi kecepatan, efisiensi penggunaan memori, dan tingkat keunikan output hash yang dihasilkan oleh masing-masing algoritma. Percobaan dilakukan dengan membangun sistem pemendekan URL berbasis Node.js dan Express serta melakukan pengujian performa menggunakan K6 untuk mengukur waktu respon dan throughput dari masing-masing algoritma. Hasilnya menunjukkan bahwa Base62 unggul dalam hal kecepatan, sementara SHA-256 dan MD5 menawarkan keunikan dan keamanan yang lebih tinggi. Namun, peningkatan penggunaan memori dan waktu komputasi untuk SHA-256 membuatnya kurang efisien dibandingkan dengan Base62 dalam skenario dengan volume permintaan yang tinggi

Kata kunci: Base62, MD5, SHA-256, Performa, Keunikan, Pemendek URL,

PENDAHULUAN

Penggunaan layanan pemendekan URL secara luas telah menjadi praktik yang ada di mana-mana di berbagai platform dan aplikasi online. Layanan ini memainkan peran penting dalam mengubah URL yang panjang dan berat menjadi alternatif yang ringkas dan mudah diingat yang

mudah dibagikan dan dilacak oleh pengguna. [1] Namun, pemilihan algoritma hashing yang tepat untuk menghasilkan URL pendek ini merupakan pertimbangan penting bagi pengembang dan penyedia layanan. Pemilihan algoritme hashing dapat secara langsung memengaruhi kinerja, keamanan, dan keunikan tautan yang dihasilkan, yang secara signifikan memengaruhi efisiensi dan efektivitas sistem pemendekan URL secara keseluruhan. [2] Pengembang harus secara hati-hati mengevaluasi trade-off dan kesesuaian algoritma hashing yang berbeda, seperti MD5, SHA-256, dan Base62, untuk menentukan opsi yang paling tepat untuk persyaratan aplikasi pemendekan URL spesifik mereka. Faktor-faktor seperti kecepatan, penggunaan memori, dan keunikan output hash yang dihasilkan oleh masing-masing algoritma dapat secara signifikan memengaruhi kinerja dan efektivitas platform pemendek URL. Dengan demikian, keputusan ini membutuhkan pemahaman menyeluruh tentang karakteristik kinerja dan trade-off yang terkait dengan setiap algoritma hashing untuk memastikan efisiensi dan keamanan yang optimal dari sistem pemendekan URL. [3]

Meskipun MD5, SHA-256, dan Base62 adalah algoritma hashing yang umum digunakan dalam sistem pemendekan URL, literatur yang ada tentang kinerja komparatif mereka dalam konteks ini terbatas. [4] Algoritme- algoritme ini menawarkan berbagai tingkat kecepatan, penggunaan memori, dan keunikan output hash, masing-masing dengan kelebihan dan kekurangannya. Kurangnya penelitian yang komprehensif ini meninggalkan kesenjangan dalam pemahaman kita tentang kesesuaian dan pertukaran algoritme ini untuk aplikasi pemendekan URL. Mengatasi kesenjangan ini sangat penting, karena pilihan algoritma hashing dapat secara signifikan berdampak pada efisiensi dan efektivitas sistem pemendekan URL secara keseluruhan. Saat ini, para pengembang dan penyedia layanan sering kali mengandalkan bukti anekdot atau preferensi pribadi saat memilih algoritma hashing untuk platform pemendek URL mereka, tanpa pemahaman yang jelas tentang karakteristik kinerja spesifik dan trade-off dari opsi yang tersedia. Kurangnya data empiris ini dapat menyebabkan keputusan yang tidak optimal yang dapat membahayakan kecepatan sistem, efisiensi memori, atau keunikan dan keamanan URL pendek yang dihasilkan. Untuk menjembatani kesenjangan ini, penelitian ini bertujuan untuk memberikan perbandingan menyeluruh berbasis data dari algoritme MD5, SHA-256, dan Base62 dalam konteks sistem pemendekan URL, memeriksa kinerja mereka dalam hal kecepatan, penggunaan memori, dan keunikan output hash. Dengan menawarkan wawasan ini, penelitian ini dapat memandu para pengembang dan penyedia layanan dalam memilih algoritme hashing yang paling tepat untuk kebutuhan spesifik mereka, memastikan kinerja dan efisiensi yang optimal dari aplikasi pemendek URL mereka

Penelitian ini bertujuan untuk mengatasi kesenjangan yang ada dalam literatur penelitian dengan melakukan analisis kinerja yang komprehensif terhadap algoritme MD5, SHA-256, dan Base62 dalam konteks sistem pemendekan URL. Penelitian ini secara menyeluruh memeriksa dan membandingkan kecepatan, efisiensi penggunaan memori, dan keunikan output hash yang dihasilkan oleh masing-masing algoritme ini. Dengan memberikan wawasan yang berharga ini, penelitian ini berusaha untuk memandu para pengembang dan penyedia layanan dalam memilih algoritme hashing yang paling tepat untuk sistem pemendekan URL mereka. Pemilihan harus didasarkan pada pertimbangan yang cermat terhadap persyaratan, keuntungan, dan trade-off spesifik yang terkait dengan masing-masing algoritma, sehingga memastikan kinerja, efisiensi, dan keamanan yang optimal dari sistem pemendekan URL

TINJAUAN PUSTAKA

1. Pemendekan URL

Pemendekan URL adalah proses mengubah URL panjang menjadi URL yang lebih pendek tanpa kehilangan fungsi utama dari tautan tersebut. URL pendek sering digunakan untuk mempermudah berbagi tautan di media sosial, pesan teks, atau email. Teknologi ini juga memungkinkan pelacakan performa tautan, seperti jumlah klik atau lokasi pengguna. Loan dan Shah (2020) [1] menyebutkan bahwa pemendekan URL juga berperan dalam menghemat ruang karakter dalam konteks platform dengan batasan karakter, seperti Twitter.

2. NodeJs

Node.js adalah platform runtime JavaScript yang berjalan di sisi server, menggunakan mesin V8 milik Google Chrome. Node.js memungkinkan pengembangan aplikasi berbasis jaringan yang cepat, ringan, dan skalabel. Dalam konteks penelitian ini, Node.js digunakan untuk membangun API pemendekan URL, karena kemampuannya menangani banyak koneksi secara simultan dengan pendekatan event-driven non-blocking I/O [3].

3. ExpressJs

Express.js adalah framework minimalis untuk Node.js yang menyediakan fitur lengkap untuk pengembangan web dan API. Framework ini mendukung arsitektur modular dengan middleware, sehingga cocok untuk membangun aplikasi RESTful seperti sistem pemendekan URL. Kuznetsov et al. (2021) [2] menyebutkan bahwa penggunaan Express.js mempermudah pengelolaan rute dan pengolahan data untuk aplikasi berbasis API.

4. K6

K6 adalah alat pengujian performa berbasis open-source yang digunakan untuk mengukur beban dan skalabilitas aplikasi. Dalam penelitian ini, K6 digunakan untuk menguji kinerja algoritma hashing pada sistem pemendekan URL. Alat ini dapat mensimulasikan ribuan pengguna virtual dan memberikan metrik performa seperti throughput, waktu respons, dan jumlah permintaan yang berhasil.

5. Base62

Base62 adalah algoritma encoding yang menggunakan 62 karakter alfanumerik (0-9, a-z, A-Z) untuk menghasilkan string pendek dan unik. Base62 banyak digunakan dalam sistem pemendekan URL karena efisiensinya dalam mengurangi panjang string URL tanpa kehilangan keterbacaan. Menurut Parmar dan Jit (2021) [4], Base62 memiliki kecepatan tinggi tetapi tidak memiliki fitur keamanan seperti algoritma hashing lainnya.

6. MD5

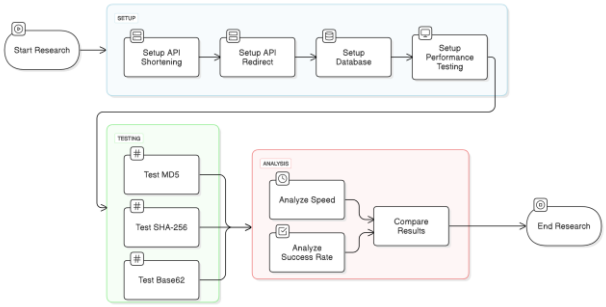
MD5 (Message Digest Algorithm 5) adalah algoritma hashing yang menghasilkan output dengan panjang tetap 128-bit. Algoritma ini cepat tetapi kurang aman karena rentan terhadap serangan tabrakan (collision attack). MD5 cocok untuk aplikasi yang tidak memerlukan tingkat keamanan tinggi, seperti pemendekan URL sederhana.

7. SHA-256

SHA-256 (Secure Hash Algorithm 256-bit) adalah algoritma hashing dalam keluarga SHA-2 yang menawarkan tingkat keamanan lebih tinggi dibandingkan MD5. Algoritma ini menghasilkan output dengan panjang tetap 256-bit, menjadikannya lebih tahan terhadap serangan tabrakan. Namun, SHA-256 membutuhkan waktu komputasi lebih lama dibandingkan MD5 .

METODOLOGI PENELITIAN

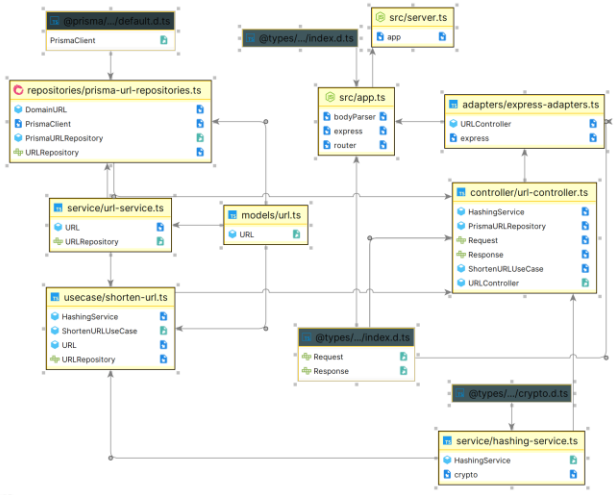
Analisis kinerja algoritme MD5, SHA-256, dan Base62 dalam sistem pemendekan URL dilakukan melalui serangkaian eksperimen yang terdefinisi dengan baik. Eksperimen ini dirancang untuk mengevaluasi secara sistematis kecepatan, efisiensi penggunaan memori, dan keunikan output hash yang dihasilkan oleh masing- masing algoritma.



Gambar 1. Metode Penelitian

Implementasi Sistem Pemendekan URL

Sistem pemendek URL diimplementasikan menggunakan Node.js dan Express.js, dengan ORM Prisma dan PostgreSQL sebagai basis data. Sistem ini mengikuti pendekatan arsitektur bersih, yang merupakan pola desain perangkat lunak yang mengatur aplikasi ke dalam lapisan-lapisan yang berbeda dan digabungkan secara longgar, seperti presentasi, aplikasi, dan infrastruktur. Pola arsitektur ini membantu memisahkan perhatian dari berbagai komponen, membuat sistem lebih mudah dipelihara, dapat diuji, dan dapat diskalakan. Dengan mendefinisikan dengan jelas tanggung jawab setiap lapisan dan mendorong pemisahan yang jelas, pendekatan arsitektur bersih memungkinkan tim pengembangan untuk melakukan perubahan dan peningkatan pada sistem secara lebih efisien, tanpa menimbulkan efek samping yang tidak diinginkan atau mengorbankan integritas aplikasi secara keseluruhan. Selain itu, pola arsitektur bersih mendukung modularisasi sistem, memungkinkan pengembangan dan pengujian independen dari masing-masing komponen, yang selanjutnya meningkatkan fleksibilitas, kemampuan pengujian, dan skalabilitas aplikasi pemendek URL.



Gambar 2. Implementasi Sistem Pemendekan URL

a. API Pemendekan URL

API pemendekan URL menyediakan titik akhir RESTful API yang menerima URL panjang dari

pengguna. [5] API kemudian menghasilkan URL pendek yang sesuai dengan menggunakan algoritme hashing yang dipilih. Fungsionalitas ini memungkinkan pengguna untuk mengirimkan URL mereka yang panjang dan berat dan menerima URL pendek yang ringkas dan unik yang kemudian dapat mereka bagikan dan distribusikan di berbagai platform dan aplikasi online. URL pendek yang dihasilkan dirancang agar mudah diingat dan dilacak, sehingga memberikan cara yang lebih nyaman bagi pengguna untuk berbagi dan mengelola tautan mereka di berbagai lingkungan digital.

b. Layanan Pengalihan URL

Layanan pengalihan URL menyediakan fungsionalitas inti dari sistem pemendekan URL. Komponen ini bertanggung jawab untuk menerima permintaan ke URL pendek yang dihasilkan dan mengarahkan pengguna ke URL panjang asli yang sesuai. Ketika pengguna mengakses URL pendek, layanan pengalihan akan mencari pemetaan antara URL pendek dan URL panjang asli dalam database, dan kemudian mengeluarkan respons pengalihan HTTP ke browser pengguna, menavigasikannya dengan mulus ke tujuan yang dituju. Proses pengalihan ini merupakan bagian penting dari sistem pemendekan URL, yang memungkinkan pengguna untuk mengakses konten asli melalui URL pendek yang ringkas dan mudah dibagikan. [6]

c. Basis data

Basis data PostgreSQL digunakan untuk menyimpan pemetaan antara URL pendek yang dihasilkan dengan URL panjang aslinya. Prisma ORM, alat Pemetaan Objek-Relasional tingkat lanjut, digunakan untuk memfasilitasi interaksi yang efisien dan aman dengan basis data dari dalam aplikasi Node.js. [7] Prisma menyediakan lapisan abstraksi yang mulus, yang memungkinkan tim pengembangan untuk melakukan operasi basis data, seperti membuat, mengambil, memperbarui, dan menghapus pemetaan URL, dengan cara yang lebih intuitif dan aman untuk tipe, sehingga meningkatkan pemeliharaan dan keandalan sistem pemendekan URL secara keseluruhan.

Pengujian Kinerja

Untuk mengevaluasi performa dari algoritma hashing yang berbeda, para peneliti menggunakan alat pengujian beban K6. [8] K6 adalah alat pengujian beban open-source yang berpusat pada pengembang yang memungkinkan pelaksanaan pengujian beban yang dapat diskalakan dan didistribusikan. Pengujian kinerja melibatkan langkah-langkah kunci berikut ini:

Layanan Pengalihan URL

Skrip K6 dirancang untuk mensimulasikan sejumlah besar permintaan pengguna secara bersamaan ke API pemendekan URL. Beban secara bertahap ditingkatkan untuk menguji sistem dan mengamati karakteristik kinerja setiap algoritma hashing di bawah kondisi beban tinggi [9].

Pengujian Kinerja

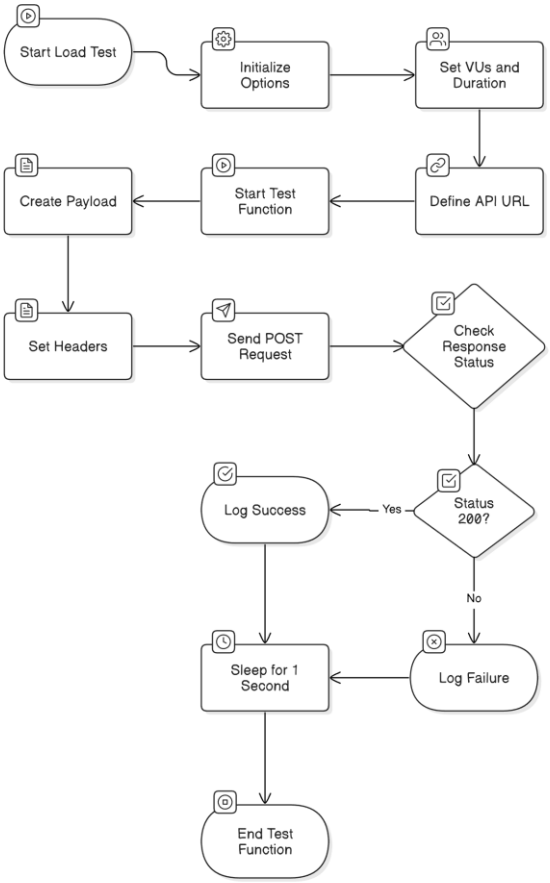
Metrik kinerja berikut ini dikumpulkan selama pengujian beban :

a. Waktu Tanggapan Permintaan

Waktu yang dibutuhkan untuk menghasilkan URL pendek sebagai respons terhadap permintaan pengguna, yang merupakan metrik penting untuk mengevaluasi kecepatan dan daya tanggap sistem pemendekan URL. Waktu yang diperlukan untuk menghasilkan URL pendek, karena pengalaman pengguna dan kinerja sistem secara keseluruhan sangat dipengaruhi oleh kecepatan sistem dalam merespons permintaan pemendekan URL. [10]

b. Throughput

Throughput, yang merepresentasikan jumlah permintaan pembuatan URL pendek yang berhasil yang dapat ditangani oleh sistem per detik. Metrik ini sangat penting untuk mengevaluasi kapasitas dan skalabilitas keseluruhan sistem pemendekan URL, karena secara langsung berdampak pada kemampuan sistem untuk melayani sejumlah besar pengguna secara bersamaan dan menangani lalu lintas bervolume tinggi tanpa mengorbankan pengalaman pengguna. [11]



Gambar 3. *Throughput*

HASIL DAN PEMBAHASAN

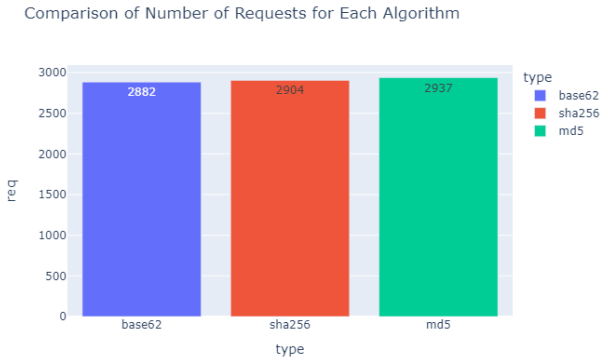
Hasil penelitian ini memberikan wawasan yang berharga tentang karakteristik kinerja algoritme MD5, SHA-256, dan Base62 ketika digunakan dalam sistem pemendekan URL dalam lingkungan Docker. Eksperimen ini dirancang untuk menilai efisiensi dan keandalan algoritme- algoritme ini dalam kondisi yang terkendali, dengan menggunakan RAM 8GB, 50 pengguna virtual, dan durasi pengujian selama 1 menit. Metrik kinerja yang dianalisis meliputi jumlah permintaan, permintaan yang berhasil, permintaan yang gagal, iterasi, pengguna virtual, dan permintaan rata-rata per detik, yang menawarkan evaluasi komprehensif terhadap perilaku algoritme dan kesesuaiannya untuk aplikasi pemendekan URL.

Data yang disajikan dalam tabel yang disediakan menunjukkan bahwa algoritma MD5 mengungguli dua algoritma lainnya, SHA-256 dan Base62, dalam hal jumlah permintaan yang diproses dan jumlah permintaan yang berhasil. Algoritma MD5 menangani volume permintaan tertinggi di 2.937 dan memiliki jumlah permintaan sukses terbesar di 2.234, yang mengindikasikan kinerjanya yang lebih unggul dibandingkan dengan algoritma SHA-256 dan Base62 dalam kondisi yang diuji.

Tabel 1. Metrik Kinerja

Algoritma	**Base62**	**SHA256**	**MD5**
Jumlah Request	2882	2904	2937
Jumlah Request Sukses	2047	2126	2234
Jumlah Request Gagal	835	778	703
Iterasi	2882	2904	2937
Pengguna Virtual	50	50	50
Rata - rata Request per Detik	47	47	49

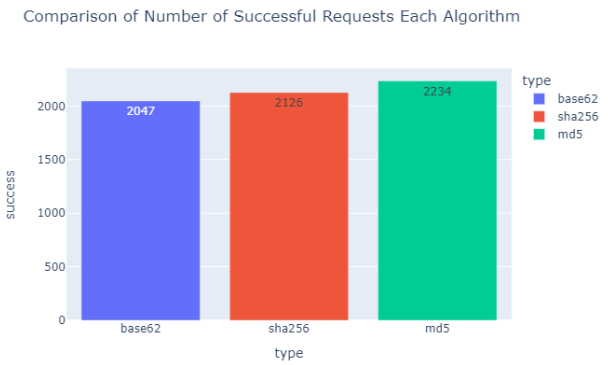
Analisis komparatif dari algoritma mengungkapkan beberapa temuan menarik. Dalam hal jumlah total permintaan, MD5 tampil paling tinggi dengan 2.937 permintaan, diikuti oleh SHA-256 dengan 2.904 permintaan dan Base62 dengan 2.882 permintaan [3]. Hal ini menunjukkan bahwa algoritma MD5 mampu memproses volume permintaan yang sedikit lebih tinggi dibandingkan dengan dua algoritma lainnya. Volume permintaan yang lebih tinggi untuk MD5 ini dapat dikaitkan dengan proses hashing yang relatif lebih sederhana dibandingkan dengan algoritma SHA-256 yang lebih kompleks, yang memungkinkannya untuk menangani jumlah permintaan yang sedikit lebih banyak dalam waktu dan sumber daya yang diberikan. Namun, perbedaan volume permintaan di antara ketiga algoritme tersebut relatif kecil, yang menunjukkan bahwa semuanya menunjukkan kinerja yang kuat dalam sistem pemendekan URL dalam kondisi yang diuji.



Gambar 4. Analisis Komparatif

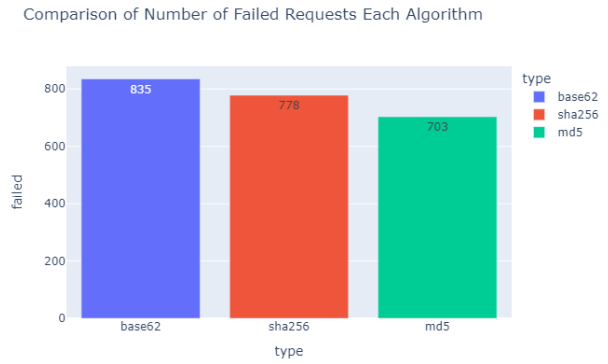
Gambar pertama membandingkan jumlah total permintaan yang diproses oleh masing-masing algoritma. Ini menunjukkan bahwa algoritma MD5 menangani jumlah permintaan tertinggi yaitu 2.937, diikuti oleh SHA-256 dengan 2.904 dan Base62 dengan 2.882. Hal ini menunjukkan bahwa algoritma MD5 mampu memproses volume permintaan yang sedikit lebih besar, yang mungkin disebabkan oleh proses hashing yang lebih sederhana dibandingkan dengan algoritma SHA-256 yang lebih intensif secara komputasi. Perbedaan yang relatif kecil dalam total

volume permintaan di antara ketiga algoritme, bagaimanapun juga, menunjukkan bahwa semuanya menunjukkan kinerja yang kuat dalam sistem pemendekan URL dalam kondisi yang diuji.



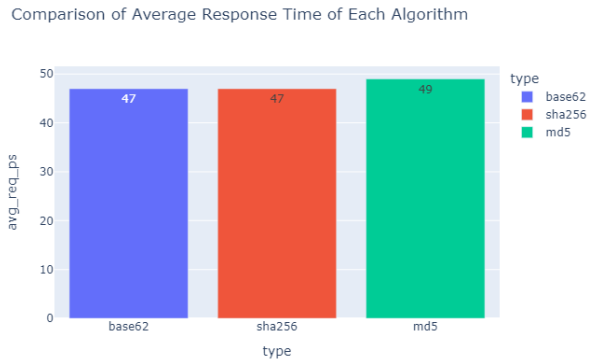
Gambar 5. Analisis Komparatif

Gambar kedua menyajikan perbandingan terperinci dari permintaan yang berhasil untuk setiap algoritme. Metrik kinerja utama ini sangat penting karena secara langsung mencerminkan keandalan dan keefektifan algoritme dalam sistem pemendekan URL. Hasilnya menunjukkan bahwa algoritme MD5 memiliki jumlah permintaan yang berhasil paling banyak, yaitu 2.234, yang merupakan 76% dari total permintaan. Hal ini menunjukkan bahwa algoritma MD5 mungkin merupakan pilihan yang paling sesuai dan menguntungkan untuk aplikasi pemendekan URL yang memprioritaskan kecepatan, throughput, dan efisiensi sebagai persyaratan utamanya. Tingkat keberhasilan yang tinggi dari algoritma MD5 menunjukkan kemampuannya untuk secara konsisten dan andal memproses permintaan pemendekan URL dalam jumlah besar, menjadikannya pilihan yang berpotensi menarik untuk aplikasi dengan kebutuhan yang didorong oleh kinerja.



Gambar 6. Analisis Komparatif

Gambar ketiga membandingkan jumlah permintaan yang gagal untuk setiap algoritme. Metrik ini memberikan wawasan tentang tingkat kesalahan dan ketahanan algoritma. Data menunjukkan bahwa algoritma MD5 memiliki jumlah permintaan yang gagal paling rendah yaitu 703, dibandingkan dengan 778 untuk SHA-256 dan 835 untuk Base62. Tingkat kegagalan yang lebih rendah untuk algoritma MD5 ini menunjukkan bahwa algoritma ini menunjukkan ketahanan dan keandalan yang lebih baik dibandingkan dengan dua algoritma lainnya dalam kondisi yang diuji. Tingkat kesalahan yang relatif lebih rendah dari algoritme MD5 semakin memperkuat potensi kecocokannya untuk sistem pemendekan URL yang memprioritaskan kinerja dan keandalan sebagai persyaratan utama, karena tingkat kegagalan yang lebih rendah dari permintaan yang gagal kemungkinan besar akan menghasilkan layanan pemendekan URL yang lebih stabil dan efisien.



Gambar 7. Analisis Komparatif

Gambar keempat membandingkan waktu respons rata-rata untuk setiap algoritme. Metrik ini memberikan wawasan tentang efisiensi dan daya tanggap algoritme dalam kondisi pengujian. Data menunjukkan bahwa ketiga algoritme menunjukkan waktu respons rata-rata yang relatif sama, dengan hanya sedikit perbedaan yang terlihat. Meskipun algoritme MD5 memiliki waktu respons rata-rata yang sedikit lebih cepat dibandingkan dengan SHA-256 dan Base62, perbedaannya tidak terlalu besar, menunjukkan bahwa ketiga algoritme tersebut menunjukkan daya tanggap dan efisiensi yang sebanding dalam memproses permintaan di dalam sistem pemendekan URL

KESIMPULAN DAN SARAN

Kesimpulan

Analisis kinerja algoritma MD5, SHA-256, dan Base62 dalam sistem pemendekan URL memberikan gambaran yang beragam. Meskipun algoritme MD5 menunjukkan volume permintaan dan tingkat keberhasilan tertinggi secara keseluruhan, perbedaan kinerja di antara ketiga algoritme tersebut relatif kecil. Dalam konteks pemendekan URL, di mana kecepatan dan throughput sering kali sangat penting, algoritme MD5 mungkin merupakan pilihan yang lebih disukai karena kinerjanya yang sedikit lebih baik dalam hal jumlah permintaan dan permintaan yang berhasil.

Namun, penting untuk mempertimbangkan potensi pertukaran dalam hal keamanan, karena algoritme MD5 dikenal kurang aman dibandingkan dengan algoritme SHA-256. AlgoritmaMD5 lebih rentan terhadap serangan tabrakan, di mana dua input yang berbeda dapat menghasilkan nilai hash yang sama, yang berpotensi membahayakan integritas sistem.

Pada akhirnya, pilihan algoritme untuk sistem pemendekan URL akan bergantung pada persyaratan dan prioritas spesifik dari aplikasi, menyeimbangkan faktor-faktor seperti kinerja, keamanan, dan keandalan. Jika keamanan adalah hal yang paling penting, algoritma SHA-256 mungkin merupakan pilihan yang lebih baik, meskipun metrik kinerjanya sedikit lebih rendah dalam penelitian ini. Sebaliknya, jika kecepatan dan throughput adalah pendorong utama, algoritma MD5 dapat menjadi pilihan yang layak, asalkan langkah-langkah keamanan yang tepat tersedia untuk mengurangi kerentanan yang diketahui.

Saran

Penelitian lebih lanjut dan pengujian di dunia nyata mungkin diperlukan untuk menentukan algoritme yang paling sesuai untuk kasus penggunaan pemendekan URL yang diberikan, dengan mempertimbangkan kebutuhan dan batasan unik aplikasi

UCAPAN TERIMA KASIH

Terima kasih penulis ucapkan kepada seluruh pihak yang telah membantu dalam penyelesaian penelitian ini, sehingga penelitian ini dapat terselesaikan. Semoga penelitian ini dapat bermanfaat bagi banyak orang

DAFTAR PUSTAKA

[1] F. A. Loan and U. Y. Shah, "The decay and persistence of web references," *Emerald Publishing Limited*, vol. 36, no. 2, pp. 157–166, 2020, doi: 10.1108/dlp-02-2020-0013.

[2] A. Kuznetsov, I. Oleshko, V. Tymchenko, K. Lisitsky, M. Rodinko, and A. Kolhatin, "Performance Analysis of Cryptographic Hash Functions Suitable for Use in Blockchain," vol. 13, no. 2, pp. 1–15, 2021, doi: 10.5815/ijcnis.2021.02.01.

[3] F. Wang *et al.*, "An Experimental Investigation Into the Hash Functions Used in Blockchains," *Institute of Electrical and Electronics Engineers*, vol. 67, no. 4, pp. 1404–1424, 2020, doi: 10.1109/tem.2019.2932202.

[4] M. Parmar and H. Jit, "Comparative Analysis of Secured Hash Algorithms for Blockchain Technology and Internet of Things," *Science and Information Organization*, vol. 12, no. 3, 2021, doi: 10.14569/ijacsa.2021.0120335.

[5] P. K. -, S. C. -, S. M. K. -, and S. S. -, "A Web-based Platform that Enables Small Businesses to Set Up an Online Store and Sell their Products to Customers," vol. 5, no. 3, 2023, doi: 10.36948/ijfmr.2023.v05i03.2919.

[6] U. Naseem, K. Musiał, P. Eklund, and M. Prasad, "Biomedical Named-Entity Recognition by Hierarchically Fusing BioBERT Representations and Deep Contextual-Level Word-Embedding," 2020, doi: 10.1109/ijcnn48605.2020.9206808.

[7] K. Revathi, T. Tamilselvi, B. Dhanwanth, and M. Dhivya, "Auto JSON: An Automatic Transformation Model for Converting Relational Database to Non-relational Documents," *Science and Information Organization*, vol. 14, no. 3, 2023, doi: 10.14569/ijacsa.2023.0140377.

[8] Y. Nakatani, "Structured Allocation-Based Consistent Hashing With Improved Balancing for Cloud Infrastructure," *Institute of Electrical and Electronics Engineers*, vol. 32, no. 9, pp. 2248–2261, 2021, doi: 10.1109/tpds.2021.3058963.

[9] V. Nair and D. Song, "Multi-Factor Credential Hashing for Asymmetric Brute-Force Attack Resistance," 2023, doi: 10.1109/eurosp57164.2023.00013.

[10] S. Zhuang, J. H. Wang, P. Zhang, and J. Wang, "Understanding the latency to visit websites in China: An infrastructure perspective," *Elsevier BV*, vol. 169, pp. 107102–107102, 2020, doi: 10.1016/j.comnet.2020.107102.

[11] K. MacMillan, T. Mangla, J. Saxon, and N. Feamster, "Measuring the performance and network utilization of popular video conferencing applications," 2021, doi: 10.1145/3487552.3487842.