KESATRIA: Jurnal Penerapan Sistem Informasi (Komputer & Manajemen)
Terakreditasi Nomor 204/E/KPT/2022 | Vol. 4, No. 4, Oktober (2023), pp. 1069-1075

Perbandingan Performa Kecepatan dari Algoritma *Hash Function* untuk Proses Enkripsi *Password*

Ahmad Miftah Fajrin
Universitas Surabaya, Indonesia
E-mail: ahmadmiftah@staff.ubaya.ac.id

Abstract

Hash Function is a crucial component in the field of cryptography. It is also utilized for authentication processes within a system. Authentication processes need to ensure both security and data processing speed. One vital component in the authentication process is the password. Password security can be enhanced through encryption using hash functions. Algorithms that can be employed include SHA-1 until SHA-3, MD5 and BLAKE2. Performance testing for execution speed and password comparison is conducted on these algorithms. The results indicate that the BLAKE2 algorithm has an average encryption time of 0.9857. Likewise, for the password comparison process, the BLAKE2 algorithm also exhibits a high speed with an average of 0.2230. As a result, the BLAKE2 algorithm is recommended for use in system authentication.

Keywords: *Hash Function, Encryption, Password*

Abstrak

Hash Function adalah salah satu komponen penting dalam bidang kriptografi. Hash function juga digunakan untuk proses authentication dalam sebuah sistem. Proses authentication harus terjamin keamanan dan kecepatan dalam pemrosesan data juga harus diperhatikan. Salah satu komponen penting dalam proses authentication adalah password. Pengamanan password dapat dilakukan dengan proses enkripsi menggunakan hash function. Algoritma yang dapat digunakan adalah SHA-1 sampai SHA-3, MD5 dan BLAKE2. Pengujian performa kecepatan dalam proses eksekusi dan komparasi password dilakukan pada algoritma tersebut. Hasil menunjukkan bahwa algoritma BLAKE2 mempunyai rata-rata waktu enkripsi yaitu 0.9857. Sedangkan untuk proses komparasi password, algoritma BLAKE2 juga memiliki kecepatan dengan rata-rata 0.2230. Algoritma BLAKE2 dapat direkomendasikan untuk digunakan untuk proses authentication pada sebuah sistem.

Kata Kunci: *Hash Function, Enkripsi, Password*

1. Pendahuluan

Hash Function adalah sebuah fungsi yang bertujuan untuk mengubah panjang dari sebuah kata atau *string*. Hash Function banyak diterapkan diberbagai bidang khususnya bidang *information security* [1]. Salah satu *concern* dalam bidang *information security* adalah *authentication. Authentication* seperti penggunaan *password* pada aplikasi harus terjamin keamanannya. Password banyak digunakan untuk mengkonfirmasi dan mengautentikasi apakah pengguna mempunyai hak akses pada sistem yang akan digunakan. Salah satu cara untuk menyamarkan *password* yaitu enkripsi dengan *hash function*. Enkripsi adalah salah satu teknik untuk menyamarkan data asli menjadi data yang sulit dibaca. *Hash Function* adalah salah satu teknik enkripsi yang berjenis *one way function* [2].

Penggunaan *hash function* untuk keamanan data sudah banyak diterapkan pada aplikasi berbasis website maupun desktop [3][4]. Algoritma *hash function* mempunyai input berupa data atau teks yang dapat menghasilkan output sesuai

ISSN: 2720-992X

dengan keinginan. Hasil output yang tercipta akan selalu sama panjangnya jika input yang diberikan sama. Semua algoritma *hash function* tidak bisa dikonversi dari output ke input [5]. Penyerang hanya bisa menebak data asli dari output. Jika terjadi perubahan pada input maka output yang dihasilkan akan beda. Ini akan menjadi dasar *integrity* dan *authentication* ketika ada data yang diubah secara *legal* maupun *illegal*.

Sampai saat ini, algoritma *hash function* seperti MD5 dan SHA *family* masih banyak digunakan [6]. Analisis MD5 dan SHA-1 dilakukan untuk mencari celah *collision* dan *length extension* yang terjadi dari hasil enkripsi [7]. MD5 mempunyai waktu eksekusi lebih cepat dari pada SHA-1 dalam hal enkripsi data [8]. Selain algoritma SHA dan MD5, blake juga digunakan untuk proses enkripsi sebuah data [9]. Akan tetapi, masih belum banyak penelitian yang membandingkan kecepatan dari proses enkripsi pada sebuah *password* yang digunakan pada proses *register* dan *login* di sebuah website. Pada penelitian ini, dilakukan analisis terhadap kecepatan enkripsi untuk proses pembuatan *password*. Tidak hanya proses pembuatan password, akan tetapi proses untuk mencocokkan data atau *select password* akan juga diukur kecepatan eksekusinya. Algoritma MD5, SHA-1 sampai SHA-3 dan BLAKE2 akan dibandingkan dalam proses tersebut. Semakin cepat waktu eksekusi maka semakin cepat waktu tunggu untuk *load* sebuah aplikasi jika memanfaatkan teknik enkripsi.

2. Metodologi Penelitian

2.1. Hash Function

Hash function adalah fungsi matematis yang mengambil input data apa pun dan menghasilkan output berupa string tetap dengan panjang tetap. Fungsi ini sering digunakan dalam bidang kriptografi dan komputer untuk keperluan seperti penyandian data, verifikasi integritas data, dan pencarian data dengan efisien. Hash function tidak mungkin untuk memulihkan input data dari *hash value* yang dihasilkan. Ini memastikan bahwa *hash value* tidak mengungkapkan informasi sensitif tentang input data [10].

Hash function dapat menghasilkan output dengan cepat. Waktu yang dibutuhkan oleh hash function harus konstan, terlepas dari ukuran input data. Ini memungkinkan penggunaan hash function dalam aplikasi yang memerlukan kinerja tinggi seperti penyandian password atau pencarian data dalam struktur data hash-based [11]. Algoritma yang termasuk Hash *Function* dan yang paling sering digunakan adalah MD5, SHA-Family dan BLAKE2.

2.2. SHA *Family*

Algoritma Secure Hash Algorithm (SHA) adalah salah satu bagian dari algoritma hash yang dikembangkan oleh badan National Security Agency (NSA) Amerika Serikat. Algoritma ini digunakan secara luas untuk menghasilkan nilai hash yang unik dari data input dengan fixed size. Ada banyak jenis yang termasuk dari algoritma SHA Family seperti SHA-1, SHA-2 dan SHA-3. National Institute of Standars and Technology (NIST) menciptakan algoritma enkripsi SHA pada tahun 1993, dan dirilis sebagai bagian dari Standar Informasi Federal (FIPS 180). Cacat dalam algoritma SHA-0 ditemukan pada tahun 1995, mendorong sejumlah perubahan dan penyempurnaan untuk membuat algoritma yang lebih kuat seperti SHA-1. Kemudian berkembang menjadi SHA-2 dan SHA-3.

2.2.1. SHA-1

SHA-1 dapat menghasilkan output hash sebesar 160-bit melalui 80 langkah proses kompresi. SHA-1 dianggap sebagai bagian dari Merkle Damgård Function, dimana pesan input dibagi menjadi sejumlah blok dan diproses secara berurutan [12]. Untuk pseudocode SHA-1 tersedia di Gambar 1.

```
Initialize variables:
h0 = 0x67452301
h1 = 0xEFCDAB89
h2 = 0x98BADCFE
h3 = 0x10325476
h4 = 0xC3D2E1F0

Pre-processing:
append the bit '1' to the message
append 0 ≤ k < 512 bits '0', so that the
resulting message length (in bits)
   is congruent to 448 ≡ −64 (mod 512)
append length of message (before pre-
processing), in bits, as 64-bit big-endian
integer

Process the message in successive 512-bit
chunks:
break message into 512-bit chunks
for each chunk
    break chunk into sixteen 32-bit big-
endian words w[i], 0 ≤ i ≤ 15

    Extend the sixteen 32-bit words into
eighty 32-bit words:
    for i from 16 to 79
        w[i] = (w[i-3] xor w[i-8] xor w[i-14]
xor w[i-16]) leftrotate 1

    Initialize hash value for this chunk:
    a = h0
    b = h1
    c = h2
    d = h3
    e = h4
```

```
Main loop:
    [26]
    for i from 0 to 79
        if 0 ≤ i ≤ 19 then
            f = (b and c) or ((not b) and d)
            k = 0x5A827999
        else if 20 ≤ i ≤ 39
            f = b xor c xor d
            k = 0x6ED9EBA1
        else if 40 ≤ i ≤ 59
            f = (b and c) or (b and d) or (c and d)
            k = 0x8F1BBCDC
        else if 60 ≤ i ≤ 79
            f = b xor c xor d
            k = 0xCA62C1D6

        temp = (a leftrotate 5) + f + e + k + w[i]
        e = d
        d = c
        c = b leftrotate 30
        b = a
        a = temp

    Add this chunk's hash to result so far:
    h0 = h0 + a
    h1 = h1 + b
    h2 = h2 + c
    h3 = h3 + d
    h4 = h4 + e

Produce the final hash value (big-endian):
digest = hash = h0 append h1 append h2
append h3 append h4
```

Gambar 1. *Pseudocode* SHA-1

2.2.2. SHA-2

SHA-2 adalah salah satu dari hash function yang dikembangkan oleh NAS, sama dengan SHA-1. SHA-2 menggunakan formula dari Merkle Damgård yang berasal dari fungsi one way compress dan atas dasar formula dari Davies–Meyer untuk special block chiper (yang dirahasiakan). SHA-2 dibangun dan telah terjadi update yang besar dari SHA-1. SHA-2 juga mempunyai variant yaitu 224, 256, 384 dan 512 bit untuk SHA-224 dan SHA-256. Pseudocode untuk SHA-2 tersedia di Gambar 2.

```
Create initial h0-h7

k[0..63] := [insert manually]

Pre-processing (Padding):
begin with the original message of length L bits
append a single '1' bit
append K '0' bits

    create a 64-entry message schedule array w[0..63] of 32-bit words
    (The initial values in w[0..63] don't matter, so many implementations zero them here)
    copy chunk into first 16 words w[0..15] of the message schedule array

    Extend the first 16 words into the remaining 48 words w[16..63]

s1 := (w[i-2] rightrotate 17) xor (w[i-2] rightrotate 19) xor (w[i-2] rightshift 10)
        w[i] := w[i-16] + s0 + w[i-7] + s1

    Initialize working variables to current hash value:
    a := h0; b := h1;c := h2;  d := h3; e := h4
    f := h5; g := h6; h := h7
```

```
    Initialize working variables to current hash value:
    a := h0; b := h1;c := h2;  d := h3; e := h4
    f := h5; g := h6; h := h7

    Compression function main loop:
    for i from 0 to 63
        S1 := (e rightrotate 6) xor (e rightrotate 11) xor (e rightrotate 25)
        ch := (e and f) xor ((not e) and g)
        temp1 := h + S1 + ch + k[i] + w[i]
        S0 := (a rightrotate 2) xor (a rightrotate 13) xor (a rightrotate 22)
        maj := (a and b) xor (a and c) xor (b and c)
        temp2 := S0 + maj
        h := g          g := f
        f := e          e := d + temp1
        d := c          c := b
        b := a          a := temp1 + temp2

    Add the compressed chunk to the current hash value:
    h0 := h0 + a    h1 := h1 + b
    h2 := h2 + c    h3 := h3 + d
    h4 := h4 + e    h5 := h5 + f
    h6 := h6 + g    h7 := h7 + h

Produce the final hash value (big-endian):
digest := hash := h0 append h1 append h2 append h3 append h4 append h5 append h6 append h7
```

Gambar 2. *Pseudocode* SHA-2

Hash function dapat menghasilkan output dengan cepat. Waktu yang dibutuhkan oleh hash function harus konstan, terlepas dari ukuran input data. Ini memungkinkan penggunaan hash function dalam aplikasi yang memerlukan kinerja tinggi seperti penyandian password atau pencarian data dalam struktur data hash-based [11]. Algoritma yang termasuk Hash *Function* dan yang paling sering digunakan adalah MD5, SHA-Family dan BLAKE2.

2.2.3. SHA-3

SHA-3 merupakan salah satu anggota dari hash yang dirilis oleh NIST pada tahun 2015. Meskipun merupakan bagian dari rangkaian standar yang sama, SHA-3 memang berbeda dari struktur SHA-1 dan SHA-2 [13]. SHA-3 terdiri dari empat fungsi hash kriptografi dan dua fungsi output (XOFs). Contoh Family dari SHA-3 adalah SHA3-224, 256, 384, dan SHA3-512.

2.3. MD5

MD5 (Message Digest Algorithm 5) adalah algoritma hash function yang menghasilkan nilai hash 128-bit dari sebuah pesan atau file. Algoritma ini dikembangkan oleh Ronald Rivest pada tahun 1991 dan digunakan secara luas untuk tujuan non-kriptografis seperti verifikasi integritas file. Enkripsi menggunakan MD5 mempunyai step sebagai berikut :

a) Penambahan bit ke input sehingga panjangnya akan berbeda dari pesan asli
b) Inisialisasi variabel awal yang akan digunakan dalam algoritma MD5
c) Pesan dibagi menjadi blok-blok yang lebih kecil dan diproses satu per satu menggunakan algoritma MD5

d) Setelah semua blok pesan diproses, hasil hash dari setiap blok digabungkan untuk menghasilkan nilai hash akhir. Hash ini biasanya direpresentasikan dalam format heksadesimal atau basis lain sesuai kebutuhan
e) Hasil hash MD5 digunakan sebagai representasi unik dari input yang diberikan.

2.4. BLAKE2

Fungsi hash yang disebut BLAKE2 dibuat dengan memodifikasi algoritme BLAKE asli. BLAKE2 dirilis pada 21 Desember 2012. Algoritma BLAKE2 dibuat dengan menggabungkan stream cipher Bernstein ChaCha dengan sedikit modifikasi yang menyebabkan salinan permutasi input beroperasi dengan exclusive or (XOR) dengan konstanta integer, dengan efek bahwa itu akan digabungkan sebelum logika ChaCha [14]. BLAKE2 terbukti dapat menghasilkan kecepatan yang lebih baik daripada SHA Family dan MD5

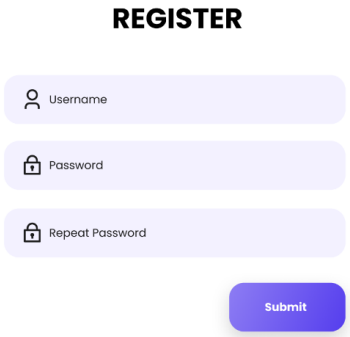
.

3. Hasil dan Pembahasan

Dilakukan pengujian terhadap lima Algoritma yaitu SHA-1 sampai SHA-3, MD5 dan BLAKE2. Pengujian dilakukan dengan cara pembuatan password yang akan dimasukkan ke dalam database MySQL. Selain itu akan dilakukan pengujian terhadap hasil *select* dari password yang tersimpan di database.

3.1. Perbandingan Performa Kecepatan untuk Enkripsi Password

Pengujian pertama dilakukan untuk mengetahui performa dari algoritma dalam hal kecepatan proses enkripsi password. Pengujian dilakukan sebanyak 10 percobaan dan akan didapatkan *average value* dari pengujian yang dilakukan. Pengujian tersebut dilakukan dengan pengisian username, password dan repeat password seperti pada Gambar 3.



Gambar 3. UI untuk Enkripsi Password

Gambar 3 terdapat username, password dan repeat password. Untuk isi username adalah “username123#” dan passwordnya adalah ”Password123#” . Hasil percobaan yang telah dilakukan terdapat pada Tabel 1

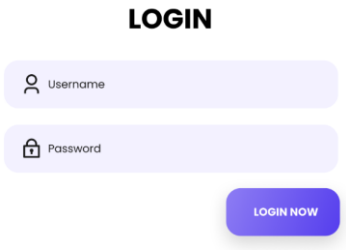
Tabel 1. Performa rata-rata waktu enkripsi dalam 10 percobaan

Algoritma	**Waktu (*ms*)**
SHA-1	1.3434
SHA-2	1.2068
SHA-3	1.0135
MD5	0.9857
BLAKE2	0.9857

Tabel 1 menunjukkan performa rata-rata waktu enkripsi terhadap lima algoritma yang dilakukan sebanyak 10 percobaan. Algoritma BLAKE2 mempunyai waktu enkripsi yang paling rendah daripada algoritma lainnya yaitu 0.9857. Sedangkan algoritma MD5 mempunyai waktu enkripsi yang paling lama yaitu 2.1914

3.2. Perbandingan Performa Kecepatan untuk Komparasi Password

Pengujian kedua dilakukan untuk mengetahui performa dari algoritma dalam hal kecepatan proses komparasi password. Pengujian dilakukan sebanyak 10 kali dan akan didapatkan nilai rata-rata dari pengujian yang dilakukan. Pengujian tersebut dilakukan dengan pengisian username, password dan repeat password seperti pada Gambar 4.



Gambar 4. UI untuk Proses Komparasi Password

Gambar 4 terdapat username dan password yang akan diuji coba sebanyak 10 kali percobaan. Untuk isi username dan password akan disamakan dengan pengujian 3.1. Hasil dari percobaan yang dilakukan tersedia di Tabel 2.

Tabel 2. Performa rata-rata waktu komparasi dalam 10 percobaan

Algoritma	Waktu (*ms*)
SHA-1	0.5873
SHA-2	0.2070
SHA-3	0.3750
MD5	0.4847
BLAKE2	0.2230

Tabel 2 menunjukkan performa rata-rata waktu komparasi password terhadap lima algoritma dan dilakukan sebanyak 10 percobaan. Algoritma BLAKE2 mempunyai waktu eksekusi yang paling rendah daripada algoritma lainnya yaitu 0.2230. Sedangkan algoritma SHA-1 mempunyai waktu enkripsi yang paling lama yaitu 0.5873

4. Kesimpulan

Pengujian telah dilakukan dengan membandingkan algoritma MD5, SHA-1, SHA-2, SHA-3 dan BLAKDE2. Hasil pengujian menunjukkan bahwa kelima algoritma ini dapat digunakan untuk proses enkripsi dan komprasi password. Dapat disimpulkan pula algoritma BLAKE2 mempunyai performa yang lebih baik daripada algoritma SHA-1, SHA-2 dan SHA-3. Terlihat dari performa kecepatan untuk enkripsi password, Algoritma BLAKE2 sangat cepat dengan mendapatkan waktu rata-rata yaitu 0.9857. Sedangkan untuk proses komparasi password, algoritma BLAKE2 juga memiliki kecepatan dengan rata-rata 0.2230. Untuk algoritma MD5 memiliki waktu enkripsi yang paling lama dengan nilai waktu 2.1914. Ini menandakan bahwa algoritma BLAKE2 sangat disarankan untuk penggunaan pada sistem *authentication* password karena kecepatannya dalam memproses data

Daftar Pustaka

[1] A. Kuznetsov, K. Shekhaniin, A. Kolhatin, D. Kovalchuk, V. Babenko, and I. Perevozova, "Performance of Hash Algorithms on GPUs for Use in Blockchain," *2019 IEEE International Conference on Advanced Trends in Information Theory, ATIT 2019 - Proceedings*, pp. 166–170, 2019, doi: 10.1109/ATIT49449.2019.9030442.

[2] P. P. Pittalia, "International Journal of Computer Science and Mobile Computing A Comparative Study of Hash Algorithms in Cryptography," *International Journal of Computer Science and Mobile Computing*, vol. 8, no. 6, pp. 147–152, 2019.

[3] R. Pamungkas and F. W. Z. Zaney, "Penerapan Hashing SHA1 dan Algoritma Asimetris RSA untuk Keamanan Data pada Sistem Informasi berbasis Web," *RESEARCH : Journal of Computer, Information System & Technology Management*, vol. 4, no. 1, p. 84, 2021, doi: 10.25273/research.v4i1.9099.

[4] A. Fathurrozi, "Penerapan Algoritma Advanced Encryption Standard (AES-256) Dengan Mode CBC DanSecure Hash Algorithm (SHA-256) Untuk Pengamanan Data File," *Journal of Information and Information Security (JIFORTY)*, vol. 2, no. 2, pp. 227–238, 2021.

[5] M. Almazrooie, A. Samsudin, A. A. A. Gutub, M. S. Salleh, M. A. Omar, and S. A. Hassan, "Integrity verification for digital Holy Quran verses u sing cryptographic hash function and compression," *Journal of King Saud University - Computer and Information Sciences*, vol. 32, no. 1, pp. 24–34, 2020, doi: 10.1016/j.jksuci.2018.02.006.

[6] A. Mohammed Ali and A. Kadhim Farhan, "A novel improvement with an effective expansion to enhance the MD5 hash function for verification of a secure E-Document," *IEEE Access*, vol. 8, pp. 80290–80304, 2020, doi: 10.1109/ACCESS.2020.2989050.

[7] Z. Al-Odat and S. Khan, "The sponge structure modulation application to overcome the security breaches for the MD5 and SHA-1 hash functions," *Proceedings - International Computer Software and Applications Conference*, vol. 1, pp. 811–816, 2019, doi: 10.1109/COMPSAC.2019.00119.

[8] M. H. Santoso, N. D. Girsang, H. Siagian, A. Wahyudi, and B. A. Sitorus, "Perbandingan Algoritma Kriptografi Hash MD5 dan SHA-1," *Seminar Nasional Teknologi Informatika*, vol. 2, no. 1, pp. 54–59, 2019.

[9] D. Upadhyay, N. Gaikwad, M. Zaman, and S. Sampalli, "Investigating the Avalanche Effect of Various Cryptographically Secure Hash Functions and Hash-Based Applications," *IEEE Access*, vol. 10, no. October, pp. 112472–112486, 2022, doi: 10.1109/ACCESS.2022.3215778.

[10] J. Katz and Y. Lindell, *Introduction to Modern Cryptography*. Chapman and Hall/CRC, 2014. doi: 10.1201/b17668.

[11] B. Schneier, "Applied Cryptography: Protocols, Algorithms, and Source Code in C. Second Edition," in *John Wiley and Sons*, 1996.

[12] Z. Al-Odat, A. Abbas, and S. U. Khan, "Randomness analyses of the secure hash algorithms, SHA-1, SHA-2 and modified SHA," in *Proceedings - 2019 International Conference on Frontiers of Information Technology, FIT 2019*, Institute of Electrical and Electronics Engineers Inc., Dec. 2019, pp. 316–321. doi: 10.1109/FIT47737.2019.00066.

[13] M. J. Dworkin, "SHA-3 Standard: Permutation-Based Hash and Extendable-Output Functions," Gaithersburg, MD, Jul. 2015. doi: 10.6028/NIST.FIPS.202.

[14] J.-P. Aumasson, S. Neves, Z. Wilcox-O'hearn, and C. Winnerlein, "BLAKE2: simpler, smaller, fast as MD5." [Online]. Available: https://blake2.net.