JIKUM
JURNAL ILMU KOMPUTER

**E-ISSN: 9999-9999**
**Volume** 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

# Implementasi Fungsi Hash dalam Kriptografi Modern untuk Enkripsi Data Satu Arah

**Mirza Winanda[1], Serli Defrianti[2], Wulan Nabila[3], Rikarni[4], Habib Alfarizhi[5]**
[1,2,3,4,5]Universitas Malikussaleh, Indonesia
[1]serli220180040@mhs.unimal.ac.id, [2]wulan.220180018@mhs.unimal.ac.id, [3]habieb.220180151@mhs.unimal.ac.id

**ABSTRACT**

*Cryptography is the science and art used to maintain data privacy. In its development, modern cryptographic algorithms work by processing data in the form of a series of bits. One important approach in cryptography is the use of a one-way hash function (one-way hash function), which is able to change data into a fixed representation that cannot be returned to its original form.*

*This paper discusses the implementation of hash functions in modern cryptography for the one-way encryption process. The hashing application is developed using the Python programming language with the hashlib library and a Tkinter-based GUI interface. This application allows users to enter text, select a hash algorithm such as MD5, SHA-1, SHA-256, SHA-512, and HSE, and get the hash results directly. Tests were carried out on several algorithms to trigger differences in the length and complexity of the results.*

*The main characteristic of a one-way hash function is that two different messages will always produce different hash values. The test results show that each algorithm produces a unique hash value for a given input, with varying lengths depending on the algorithm used. SHA-512 produces the longest hash and is considered the most secure among other algorithms.*

**Kata kunci**: *fungsi hash, kriptografi, hashing satu arah, Python, hashlib*

## PENDAHULUAN

Sejak digunakan dalam perang dunia, kriptografi terus mengalami perkembangan. Jika dahulu kriptografi menyamarkan teks asli (plainteks) menjadi teks sandi (cipherteks) dengan keluaran berupa karakter huruf yang dikenal dengan kriptografi klasik, kini teks sandi yang dihasilkan ialah berupa mode bit yang dikenal dengan kriptografi modern. Perkembangan ini tidak lepas dari penggunaan komputer digital yang merepresentasikan data dalam bentuk biner. Kriptografi modern kini lebih banyak digunakan, karena cenderung lebih aman daripada kriptografi klasik.

Algoritma kriptografi dibagi menjadi tiga bagian berdasarkan kunci yang dipakainya yaitu Algoritma Simetris, Algoritma Asimetris dan Fungsi Hash. Salah satu metoda kriptografi yang dianggap sebagai algoritma block cipher yang terbaik dan teraman yang tersedia untuk publik sampai saat ini adalah metoda kriptografi IDEA (International Data Encryption Algorithm) dimana IDEA termasuk jenis Algoritma Simetris berdasarkan kunci yang dipakainya (Andi,2003)

Fungsi hash adalah fungsi yang menerima masukkan string yang panjangnya sembarang dan mentransformasikannya menjadi string keluaran yang panjangnya tetap (nilai hash), umumnya berukuran jauh lebih kecil daripada string masukkannya. Ide dasar dari fungsi hash adalah menghitung nilai hash dari kunci atau nilai asli, kemudian membandingkan kunci atau nilai asli dengan isi pada memori yang beralamat nomor hashnya tanpa harus memeriksa isi tabel satu per satu sehingga lebih efisien.

## METODE PENELITIAN

### Jenis Penelitian

Penelitian ini menggunakan metode penelitian rekayasa perangkat lunak (software engineering) dengan pendekatan kuantitatif deskriptif. Tujuannya adalah untuk merancang, membangun, dan menguji aplikasi hashing teks menggunakan berbagai algoritma kriptografi modern.

### Metode Pengembangan Sistem

Model pengembangan sistem yang digunakan adalah Waterfall, yang terdiri dari lima tahapan utama:

### Analisis Kebutuhan

Pada tahap ini dilakukan pengumpulan data mengenai kebutuhan pengguna terhadap sistem hashing teks. Kebutuhan utama mencakup:

a. Input teks dari pengguna
b. Pemilihan algoritma hashing (MD5, SHA-1, SHA-256, SHA-512)
c. Output berupa hasil hash
d. Penanganan kesalahan jika algoritma tidak dikenali

JIKUM
JURNAL ILMU KOMPUTER
**E-ISSN: 9999-9999**
**Volume** 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

**Perancangan Sistem**

Desain antarmuka pengguna dibuat sederhana dan interaktif menggunakan bahasa pemrograman Python dan pustaka GUI seperti Tkinter atau PyQt. Diagram alur data dan struktur menu dirancang agar mudah dipahami oleh pengguna.

**Implementasi**

Aplikasi diimplementasikan dengan menggunakan bahasa pemrograman Python. Proses hashing dilakukan dengan memanfaatkan pustaka hashlib. Pengguna dapat memilih algoritma hash melalui menu dropdown, kemudian aplikasi akan menampilkan hasil hash berdasarkan input yang diberikan.

**Pengujian**

Pengujian dilakukan dengan metode black-box testing untuk memastikan bahwa setiap fitur bekerja sesuai harapan. Uji coba dilakukan terhadap berbagai algoritma dan variasi input teks.

**Pemeliharaan**

Setelah pengujian, aplikasi dapat diperbarui berdasarkan masukan pengguna dan pengembangan selanjutnya, seperti penambahan algoritma hashing terbaru atau fitur ekspor hasil hash.

**Alat dan Bahan**

a. Bahasa Pemrograman: Python
b. Pustaka Kriptografi: hashlib
c. GUI: Tkinter/PyQt
d. Sistem Operasi: Windows

**Teknik Pengumpulan Data**

a. Observasi terhadap aplikasi hashing sejenis
b. Studi literatur mengenai algoritma hashing modern
c. Wawancara atau kuesioner untuk uji coba pengguna (jika diterapkan)

**Teknik Analisis Data**

Data dianalisis dengan pendekatan deskriptif, yaitu dengan membandingkan hasil hash dari berbagai algoritma untuk teks yang sama dan mengamati panjang hasil serta keunikan nilai hash.

## HASIL DAN PEMBAHASAN

Perancangan dan pembuatan Aplikasi Hash Kriptografi dilakukan melalui tahapan sebagai berikut:

1. Aplikasi menerima input teks dari pengguna dan memungkinkan pemilihan algoritma hash seperti MD5, SHA-1, SHA-256, SHA-512, dan HSE.
2. Aplikasi menampilkan hasil hash dari teks yang dimasukkan berdasarkan algoritma yang dipilih.
3. Disediakan informasi dasar mengenai algoritma kriptografi modern (hashing) pada dokumentasi atau panduan.
4. Proses hashing berlangsung cepat karena bersifat satu arah dan tidak tergantung pada ukuran data besar.
5. User diasumsikan memahami dasar konsep hash seperti irreversibility (tidak dapat dibalik), keunikan hash, dan perbedaan hash dengan enkripsi.
6. Aplikasi dirancang dengan antarmuka sederhana, dan menampilkan hasil hash secara langsung di layar, disesuaikan dengan algoritma yang digunakan.
7. Jika algoritma yang dipilih tidak dikenali atau tidak tersedia, aplikasi memberikan peringatan "Algoritma tidak dikenali".

**Implementasi Sistem**

Implementasi Aplikasi / program ini mencakup spesifikasi kebutuhan perangkat keras (hardware) dan spesifikasi perangkat lunak (software).

JIKUM
JURNAL ILMU KOMPUTER
**E-ISSN: 9999-9999**
**Volume** 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

**Spesifikasi Perangkat Keras dan Perangkat Lunak**

**Perangkat Keras (Hardware)**

Program ini direkomendasikan untuk dijalankan menggunakan perangkat keras dengan spesifikasi minimum sebagai berikut:

1. Prosesor: Intel atau AMD Processor.
2. Memori (RAM): Minimum 1 GB.
3. Penyimpanan (Harddisk): Minimum 250 GB ruang kosong.
4. VGA Card: Disarankan untuk tampilan grafis yang optimal.
5. Monitor: Resolusi minimum 1024 × 768 piksel.
6. Perangkat Input: Keyboard dan mouse.

**Perangkat Lunak (Software)**

Adapun perangkat lunak yang diperlukan untuk menjalankan dan mengembangkan aplikasi ini antara lain:

1. Sistem Operasi: Windows 7/8/10/11 atau Linux.
2. Bahasa Pemrograman: Python (dengan pustaka GUI seperti Tkinter atau PyQt, jika digunakan).
3. Interpreter Python: Python versi 3.x.
4. Library Tambahan: hashlib (untuk proses hashing), tkinter (jika GUI dibuat dengan ini).
5. Teks Editor / IDE: Visual Studio Code, PyCharm, atau IDE lain yang mendukung Python.

**Alur Kerja fungsi Hash**

Aplikasi ini adalah alat sederhana berbasis GUI (Graphical User Interface) yang dibuat menggunakan Python dengan modul tkinter dan hashlib untuk menghasilkan hash dari teks input menggunakan algoritma kriptografi seperti MD5, SHA-1, SHA-256, atau SHA 512. Berikut adalah alur cara kerja aplikasi ini secara rinci:

1. Inisialisasi GUI:
   a. Program membuat jendela utama (root) menggunakan tk.Tk() dengan judul " Aplikasi Hash Modern" dan ukuran 500x400 piksel.
   b. Jendela ini tidak dapat diubah ukurannya (resizable(False, False)).
   c. Elemen GUI seperti label, input teks, combobox untuk memilih algoritma, tombol, dan area hasil dibuat dan diatur posisinya menggunakan metode pack dan grid.
2. Komponen GUI:
   a. Judul: Label "Aplikasi Hash Kriptografi Modern" ditampilkan di bagian atas.
   b. Input Teks: Kolom input (entry_teks) memungkinkan pengguna memasukkan teks yang ingin di-hash.
   c. Pilihan Algoritma: Combobox (combo_algoritma) menawarkan empat opsi algoritma: MD5, SHA-1, SHA-256, dan SHA-512, dengan SHA-256 sebagai pilihan default. Tombol Hash: Tombol " Hash Sekarang" memicu proses hashing saat diklik.
   d. Area Hasil: Kotak teks (hasil_hash) menampilkan hasil hash dalam format heksadesimal.
3. Fungsi Hashing (hash_teks):
   a. Ketika tombol "Hash Sekarang" diklik, fungsi hash_teks dijalankan.
   b. Fungsi ini mengambil teks dari kolom input (entry_teks.get()) dan algoritma yang dipilih dari combobox (combo_algoritma.get()).
   c. Jika kolom input kosong, muncul peringatan menggunakan messagebox.showwarning dan proses berhenti.
   d. Teks input dienkode ke format byte menggunakan teks.encode().
   e. Berdasarkan algoritma yang dipilih, fungsi memanggil metode yang sesuai dari modul hashlib: hashlib.md5() untuk MD5 hashlib.sha1() untuk SHA-1 hashlib.sha256() untuk SHA-256 hashlib.sha512() untuk SHA-512
   f. Hasil hash dalam format heksadesimal (hexdigest()) dimasukkan ke kotak teks hasil (hasil_hash), setelah kotak teks tersebut dikosongkan terlebih dahulu.
4. Proses Hashing:
   a. Modul hashlib mengubah teks input menjadi hash satu arah, yang berarti teks asli tidak dapat dikembalikan dari hash.
   b. Setiap algoritma menghasilkan panjang hash yang berbeda: MD5: 128 bit (32 karakter heksadesimal) SHA-1: 160 bit (40 karakter) SHA-256: 256 bit (64 karakter) SHA-512: 512 bit (128 karakter)
   c. Hash yang dihasilkan bersifat unik untuk input tertentu; perubahan sekecil apa pun pada teks input akan menghasilkan hash yang berbeda.
5. Interaksi Pengguna:
   a. Pengguna memasukkan teks, memilih algoritma, lalu mengklik tombol "Hash Sekarang".
   b. Hasil hash ditampilkan di kotak teks di bawah tombol.
   c. Pengguna dapat mengulang proses dengan teks atau algoritma baru tanpa perlu menutup aplikasi.

JIKUM
JURNAL ILMU KOMPUTER
**E-ISSN: 9999-9999**
**Volume** 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx

6. Siklus Hidup Aplikasi:
   a. Aplikasi berjalan dalam loop utama (root.mainloop()), menunggu interaksi pengguna.
   b. Aplikasi berhenti ketika jendela ditutup.

Contoh Alur Penggunaan:

a. Pengguna mengetik "Hello" di kolom input.
b. Memilih "SHA-256" dari combobox. Klik tombol "Hash Sekarang". Kotak teks menampilkan: a591a6d40bf420404a011733cfb7b190d62c65bf0bcda32b57b277d9ad9f146e.
c. Jika pengguna mengubah teks menjadi "hello" dan mengklik lagi, hash baru akan muncul karena hash bersifat case-sensitive.

**Penggujian Program**

Aplikasi ini dirancang untuk melakukan proses hashing terhadap input teks menggunakan algoritma kriptografi modern. Pengujian dilakukan untuk memastikan bahwa proses hashing berjalan sesuai dengan pilihan algoritma yang tersedia.

**Langkah-langkah Penggunaan:**

1. **Masukkan Teks**

Pengguna terlebih dahulu memasukkan teks pada kolom input. Dalam pengujian ini, teks yang dimasukkan adalah `"SAMSUNG"`.

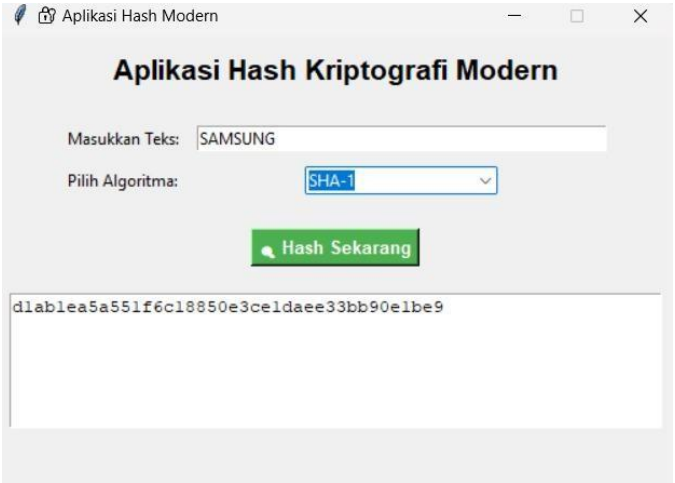
2. **Pilih Algoritma Hashing**

Aplikasi menyediakan dropdown menu untuk memilih algoritma hashing. Pada contoh ini, algoritma yang dipilih adalah **SHA-1**.

3. **Klik Tombol "Hash Sekarang"**

Setelah memilih algoritma, pengguna menekan tombol **"Hash Sekarang"** yang berwarna hijau dengan ikon kaca pembesar. Tombol ini akan memproses input teks dengan algoritma yang telah dipilih.

Hasil Hash Ditampilkan :



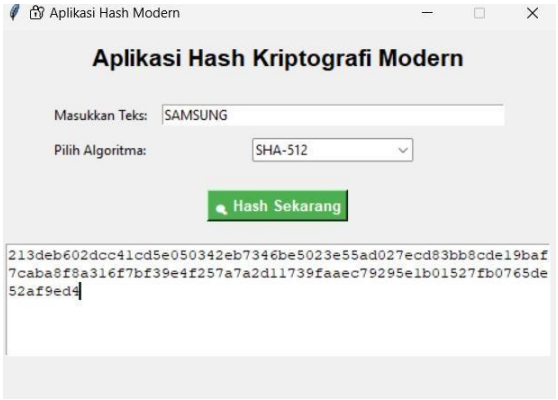

Gambar 1. proses penggunaan aplikasi dengan algoritma SHA-1

Setelah proses hashing selesai, hasil dari algoritma SHA-1 akan ditampilkan di area teks di bawah tombol. Pada pengujian ini, output hash yang dihasilkan adalah: **d1ablea5a551f6c18850e3celdaee33bb90e1be9**

contoh lain:

Pada pengujian ini, pengguna memasukkan teks **"SAMSUNG"** ke dalam kolom input. Kemudian, pengguna memilih algoritma **SHA-512** dari daftar algoritma hash yang tersedia. Setelah itu, pengguna menekan tombol **"Hash Sekarang"**, dan aplikasi berhasil menghasilkan nilai hash dari teks yang dimasukkan.

JIKUM
JURNAL ILMU KOMPUTER
**E-ISSN: 9999-9999**
**Volume** 1, **Number** 1, Mei 2025
https://doi.org/10.62671/jikum.vxix.xxxx




Gambar 2. proses penggunaan algoritma SHA-512

Hasil hashing yang ditampilkan merupakan representasi dari hasil enkripsi menggunakan algoritma SHA-512. Nilai hash ini bersifat unik dan tidak dapat dikembalikan ke bentuk semula (irreversible), sesuai dengan prinsip dasar dari algoritma kriptografi hash modern dan masih banyak contoh lainnya.

Tabel 1. Perbandingan Hasil Hash dari Algoritma yang Berbeda

| Algoritma | Panjang Output (bit) | Panjang Hash (karakter) | Hasil Hash | Kekuatan |
|---|---|---|---|---|
| **MD5** | 128 | 32 | e0c9035898dd52fc65c41454cec9c4d2 | **Lemah** |
| **SHA-1** | 160 | 40 | d1ablea5a551f6c18850e3celdaee33bb 90e1be9 | **kurang aman** |
| **SHA-256** | 256 | 64 | 2a1022b922f5038bc5bff79ff2960cd1 2cf266647b4f59121997a75c710b5b5 | **sangat aman** |
| **SHA-512** | **512** | **128** | **213deb602dcc41cd5e050342eb7346b e5023e55ad027ecd83bb8cde19baf7ca ba8f8a316f7bf39e4f257a7a2d11739fa aec79295e1b01527fb0765de52af9ed4** | **sangat kuat dan aman** |

## KESIMPULAN

Berdasarkan hasil perancangan dan implementasi aplikasi fungsi hash kriptografi modern, dapat disimpulkan bahwa fungsi hash merupakan komponen penting dalam menjaga integritas dan keamanan data melalui enkripsi satu arah. Aplikasi yang dikembangkan dengan menggunakan bahasa Python dan pustaka hashlib ini mampu menjalankan proses hashing dengan berbagai algoritma seperti MD5, SHA-1, SHA-256, dan SHA-512 secara efektif.

Pengujian menunjukkan bahwa setiap algoritma menghasilkan nilai hash yang unik dan tidak dapat dikembalikan ke bentuk semula (irreversible), sesuai dengan prinsip dasar fungsi hash. SHA-256 dan SHA-512 terbukti menghasilkan panjang hash yang lebih besar serta memiliki tingkat keamanan yang lebih tinggi dibandingkan MD5 dan SHA-1, yang kini dianggap kurang aman.

Aplikasi ini memberikan kemudahan bagi pengguna dalam mempelajari dan mempraktikkan konsep hashing melalui antarmuka yang sederhana dan interaktif. Proses hashing berlangsung cepat dan efisien tanpa memerlukan spesifikasi perangkat keras yang tinggi, sehingga dapat diimplementasikan secara luas untuk kebutuhan pendidikan maupun pengembangan sistem keamanan data.

## REFERENSI

A. E. Putra, "Fungsi hash pada kriptografi," *Makalah Struktur Diskrit*, Institut Teknologi Bandung, 2009.

A. H. Lubis, "Perbandingan algoritma kriptografi hash MD5 dan SHA-1," *Seminar Nasional Teknologi Informatika (Semantika)*, Politeknik Ganesha Medan, 2019.

Mujaddid, S. (2009). Kriptoanalisis pada fungsi hash kriptografi MD5. Makalah, Institut Teknologi Bandung.

Maryanto, B. (2008). Penggunaan fungsi hash satu-arah untuk enkripsi data. Media Informatika, 7(3), 138–146.

Prasetyo, R., & Suryana, A. (2016). Aplikasi pengamanan data dengan teknik algoritma kriptografi AES dan fungsi hash SHA-1 berbasis desktop. Jurnal SISFOKOM, 5(1), 61–65.

Rahim, I., Anwar, N., Widodo, A. M., Juman, K. K., & Setiawan, I. (2023). Komparasi fungsi hash MD5 dan SHA256 dalam keamanan gambar dan teks. Jurnal IKRAITH-Informatika, 7(2), 41–48.

RJP, Rizky MT. Diktak Kuliah Algoritma Kriptografi Modern. Program Studi Ilmu Komputer Universitas Pendidika Indonesia. 2009