Studi Literatur: Analisis Perbandingan dan Implementasi Algoritma Hash (MD5, SHA-2, SHA-3, dan SHA-512) untuk Keamanan Data dan Otentikasi

1. Pendahuluan

Perkembangan teknologi komunikasi dan informasi telah menjadikan data sebagai aset yang sangat penting. Seiring dengan meningkatnya nilai data, tantangan keamanan dalam jaringan dan sistem informasi juga turut meningkat. Insiden keamanan seperti peretasan kata sandi, pencurian data, dan akses tidak sah telah menjadi ancaman serius bagi organisasi, yang dapat mengakibatkan kerugian finansial dan reputasi.

Untuk mengatasi tantangan ini, otentikasi pengguna dianggap sebagai langkah keamanan krusial. Kriptografi, khususnya fungsi hash, memainkan peran sentral dalam melindungi data sensitif seperti kata sandi. Fungsi hash digunakan untuk menyimpan representasi data yang tidak dapat dibalik (satu arah), sehingga kata sandi asli tidak disimpan dalam bentuk plaintext.

Namun, banyak algoritma hash yang lebih tua (seperti MD5 dan SHA-1) kini dianggap tidak aman dan rentan terhadap serangan, misalnya rainbow table attack. Oleh karena itu, penelitian terus berlanjut untuk menemukan metode penyimpanan kata sandi yang lebih aman. Studi literatur ini bertujuan untuk menganalisis dan membandingkan beberapa algoritma hash kunci (MD5, SHA1, SHA2, SHA3, dan SHA-512) serta teknik pengamanan tambahan seperti salting, berdasarkan tinjauan penelitian terdahulu untuk menilai performa, keamanan, dan tren aplikasinya.

2. Konsep Dasar Kriptografi

Fungsi Hash (Hash Function)

Fungsi hash adalah fungsi matematis satu arah yang mengonversi data dengan ukuran acak menjadi sebuah string data dengan ukuran tetap, yang disebut nilai hash atau *message digest*. Secara konseptual, ini bisa ditulis sebagai **h = H(M)**, di mana M adalah pesan (data input dengan ukuran acak) dan h adalah nilai hash (output dengan ukuran tetap). Sifat utamanya adalah *one-way*, yang berarti secara praktis tidak mungkin untuk meregenerasi data input asli hanya dari nilai hash-nya. Fungsi hash umumnya digunakan untuk memastikan integritas data dan untuk penyimpanan kata sandi yang aman.

Beberapa algoritma hash yang umum dibahas meliputi:

- **MD5 (Message Digest 5)**: dikembangkan oleh Rivest, menghasilkan nilai hash 128-bit. Algoritma ini telah terbukti rentan terhadap serangan kolisi (*collision attacks*) sehingga tidak lagi dianggap aman untuk penyimpanan kata sandi. *Serangan kolisi* adalah kondisi ketika dua input data yang berbeda menghasilkan nilai hash yang sama persis. Ini berbahaya karena memungkinkan penyerang mengganti data tanpa terdeteksi, sebab sistem menganggap kedua data tersebut identik secara hash.
- **SHA-1 (Secure Hash Algorithm 1)**: dipublikasikan pada tahun 1995, menghasilkan hash 160-bit. Seperti MD5, SHA-1 juga tidak lagi dianggap tahan terhadap serangan kolisi. Secara teknis, kelemahan SHA-1 berasal dari struktur internalnya yang mirip dengan MD5, sehingga

serangan diferensial dapat digunakan untuk menemukan pasangan input yang menghasilkan hash identik dengan usaha komputasi yang lebih sedikit.

- **SHA-2 (Secure Hash Algorithm 2)**: merupakan keluarga fungsi hash yang mencakup SHA-224, SHA-256, SHA-384, dan SHA-512. SHA-512 secara spesifik menerima pesan dengan ukuran acak dan menghasilkan *message digest* 512-bit. Hingga saat ini, SHA-2 masih dianggap aman dan tahan terhadap serangan kolisi. Secara teknis, SHA-2 dibangun di atas struktur **Merkle–Damgård**, yaitu arsitektur berulang yang memproses blok data satu per satu. Rumusan dasar dari proses berantai (chaining) Merkle-Damgård ini adalah **H_i = f(H_{i-1}, M_i)**, di mana H_i adalah hash internal saat ini, f adalah fungsi kompresi, H_{i-1} adalah hasil hash dari blok sebelumnya, dan M_i adalah blok pesan (data) saat ini. Meskipun strukturnya mirip dengan MD5 dan SHA-1, ukuran blok dan fungsi kompresinya diperkuat untuk mencegah serangan yang berhasil pada pendahulunya.
- **SHA-3 (Secure Hash Algorithm 3)**: standar terbaru yang dikembangkan melalui kompetisi NIST dan didasarkan pada algoritma Keccak. SHA-3 dikembangkan untuk mengatasi kekhawatiran teoretis pada struktur SHA-2. Perbedaan mendasar SHA-3 adalah arsitekturnya menggunakan **Sponge Construction**, bukan Merkle–Damgård. Model *sponge* menyerap data input dan kemudian memeras output hash dengan cara yang fleksibel, membuatnya tahan terhadap serangan *length extension* maupun kolisi yang mungkin terjadi pada struktur lama.

Salting (Teknik Garam)

Salting adalah teknik di mana sebuah nilai acak dan unik (*salt*) ditambahkan ke kata sandi pengguna sebelum proses hash dilakukan. *Salt* ini kemudian disimpan bersama dengan hash kata sandi di dalam basis data. Tujuan utama salting adalah untuk mempersulit proses peretasan kata sandi, terutama terhadap serangan *rainbow table* dan *brute force*, dengan memastikan bahwa dua pengguna dengan kata sandi yang sama akan memiliki nilai hash yang berbeda. Secara teknis, *rainbow table* adalah database besar yang berisi jutaan hasil hash yang sudah dihitung sebelumnya untuk berbagai kombinasi kata sandi umum. Dengan menambahkan *salt* unik pada setiap kata sandi sebelum hashing Rumusan dasar dari teknik ini adalah **HashedPassword = H(Password + Salt)**, di mana H adalah fungsi hash (seperti SHA-512) dan + adalah operasi penggabungan (concatenation). Dengan 'salt' yang unik, hasil HashedPassword akan selalu unik, sehingga tabel tersebut menjadi tidak berguna, karena hasil hash untuk setiap pengguna akan berbeda. Hal ini memaksa peretas untuk membangun *rainbow table* baru untuk setiap *salt*, yang secara komputasi sangat mahal dan tidak efisien.

3. Tinjauan Penelitian Terdahulu

Berikut adalah tinjauan dari tiga penelitian relevan yang berfokus pada perbandingan dan penerapan algoritma *hash* untuk keamanan:

Peneliti & Tahun	Metode / Algoritma	Tujuan Penelitian	Hasil & Temuan	Kelemahan / Keterbatasan
(Natho et al., 2024)	Perbandingan MD5, SHA1, SHA2, dan SHA3. Menggunakan analisis kecepatan hash dan analisis keamanan dengan serangan Hashcat terhadap kata sandi lemah & kuat.	Membandingkan efektivitas berbagai fungsi hash untuk menentukan metode penyimpanan kata sandi yang paling aman.	**Kecepatan:** MD5 paling cepat, diikuti SHA1, SHA2, dan SHA3 paling lambat. **Keamanan:** SHA3 paling aman dengan tingkat keberhasilan serangan terendah	Tidak menguji algoritma yang dikombinasikan dengan teknik *salt*. Keamanan masih bisa ditembus meskipun SHA3 paling kuat.

			(20% pada kata sandi kuat). MD5 & SHA1 (35%) dan SHA2 (30%) lebih rentan.	
(Rasyada, 2022)	Perbandingan SHA-512 vs SHA-256 pada otentikasi JSON Web Token (JWT) untuk RESTful Web Service. Kinerja diukur berdasarkan kecepatan respons (ms) dan ukuran data (kb).	Menganalisis kinerja (kecepatan dan ukuran) algoritma SHA-512 pada otentikasi JWT.	**Kecepatan:** SHA-512 sedikit lebih cepat (rata-rata 512.8 ms) dibandingkan SHA-256 (515.55 ms). **Ukuran:** SHA-512 menghasilkan token lebih besar (0.75 kb) dibanding SHA-256 (0.72 kb) karena ukuran hash 512-bit vs 256-bit.	Fokus pada performa (kecepatan/ukuran) dalam konteks JWT, tidak membahas ketahanan terhadap serangan kriptanalisis.
(Aranuwa et al., 2022)	Model hibrida yang mengintegrasikan SHA-512 dengan teknik *salting*, diimplementasikan dengan Python dan diuji pada LAN yang diekspos ancaman.	Merancang model keamanan data yang lebih baik untuk mengatasi kelemahan enkripsi mode tunggal.	Model hibrida (SHA-512 + Salt) terbukti efisien dan tangguh terhadap serangan. Menunjukkan peningkatan kinerja tinggi (recital 97.6%) dibanding teknik non-hibrida.	Tidak membandingkan SHA-512+Salt dengan algoritma lain seperti SHA-3+Salt; fokus hanya pada pembuktian model hibrida.

4. Analisis dan Sintesis

Dari tinjauan ketiga penelitian tersebut, beberapa pola, tren, dan celah penelitian (*research gap*) dapat diidentifikasi:

- **Tren Evolusi Algoritma:** Terdapat pergeseran dari algoritma lama seperti MD5 dan SHA-1 yang terbukti rentan, menuju standar yang lebih modern dan aman seperti SHA-2 (termasuk SHA-512) dan SHA-3.

- **Perbandingan Performa (Kecepatan vs Keamanan):** Terdapat *trade-off* antara kecepatan dan keamanan. Studi oleh (Natho et al., 2024) menunjukkan bahwa algoritma yang lebih baru dan lebih aman (SHA-3) cenderung lebih lambat secara komputasi. Kelambatan ini sering dianggap positif karena membuat serangan *brute force* menjadi lebih mahal dan lambat bagi peretas.

- **Performa Kontekstual (Aplikasi Spesifik):**
    - Untuk penyimpanan kata sandi, keamanan adalah prioritas utama. (Natho et al., 2024) menempatkan SHA-3 sebagai yang paling aman terhadap serangan Hashcat.

- Untuk otentikasi API (JWT), kecepatan respons penting. (Rasyada, 2022) menunjukkan SHA-512 sedikit mengungguli SHA-256 dalam kecepatan, menjadikannya pilihan yang baik untuk *web service* yang membutuhkan keamanan tinggi tanpa mengorbankan performa.

- **Celah Penelitian (Research Gap) – Pentingnya Salting:**
  Algoritma hash yang kuat saja tidak cukup. (Natho et al., 2024) membuktikan bahwa bahkan SHA-3 masih dapat diretas (keberhasilan 20%) dan merekomendasikan penggunaan *salt*. Studi oleh (Aranuwa et al., 2022) mengisi celah ini dengan mengusulkan model hibrida (SHA-512 + Salting) yang terbukti sangat efektif.

5. Arah dan Peluang Penelitian

Berdasarkan analisis dan sintesis di atas, beberapa peluang penelitian di masa depan dapat diidentifikasi:

1. **Model Hibrida Lanjutan:**
   Melanjutkan penelitian (Aranuwa et al., 2022) membandingkan SHA-512 + Salt dan SHA-3 + Salt untuk menentukan standar baru keamanan database.
2. **Kriptografi Ringan (Lightweight Cryptography):**
   Meneliti fungsi hash yang ringan namun tetap aman untuk perangkat IoT dengan sumber daya terbatas.
3. **Perbandingan dengan Algoritma Slow Hash:**
   Meneliti kombinasi SHA-3 + Salt dibandingkan algoritma *slow hash* seperti Argon2, bcrypt, atau scrypt dalam skenario serangan nyata.

6. Kesimpulan

Studi literatur ini telah menganalisis dan membandingkan berbagai algoritma fungsi hash (MD5, SHA-1, SHA-2, SHA-3, dan SHA-512) berdasarkan penelitian terbaru. Temuan utama menunjukkan bahwa algoritma lama seperti MD5 dan SHA-1 sudah usang dan tidak aman. Terdapat *trade-off* antara kecepatan dan keamanan, di mana SHA-3 terbukti paling aman namun paling lambat.

Untuk aplikasi seperti otentikasi *web service* (JWT), SHA-512 menawarkan keseimbangan sangat baik antara keamanan 512-bit dan kecepatan tinggi. Temuan penting lainnya adalah bahwa keamanan kata sandi modern tidak hanya bergantung pada algoritma, tetapi juga pada metode implementasinya. Kombinasi algoritma hash kuat (seperti SHA-512 atau SHA-3) dengan teknik salting merupakan pendekatan yang jauh lebih tangguh dan direkomendasikan untuk keamanan data yang efektif.

7. Daftar Pustaka

1. Aranuwa, F., Olubodun, F., & Akinwumi, D. (2022). Hybridized model for data security based on Security Hash Analysis (SHA 512) and salting techniques. *International Journal of Network Security & Its Applications*, *14*(2), 31–39.
2. Natho, P., Somsuphaprungyos, S., Boonmee, S., & Boonying, S. (2024). Comparative study of password storing using hash function with MD5, SHA1, SHA2, and SHA3 algorithm. *International Journal of Reconfigurable and Embedded Systems (IJRES)*, *13*(3), 502–511.
3. Rasyada, N. (2022). SHA-512 algorithm on Json Web Token for Restful Web Service-based authentication. *Journal of Applied Data Sciences*, *3*(1), 33–43.