Full Paper

Implementasi Algoritma Argon2 untuk Pengamanan Password pada Sistem Jaringan Client-Server

Fernandoz Lim[1], Samuel Candra[2], Novelia Safitri[3], Khairunnisa Saputri[4], Alfana Hasbi Shiddieqy[5], Vincent Tan[6], Muhammad Iqbal Farizki Abdillah[7]

[1,2,3,4,5,6,7]Program Studi Teknik Komputer, Institut Teknologi Batam, Indonesia

Informasi Artikel

Terbit: Juli 2026

Kata Kunci:

Argon2id
Password Hashing
Client-Server
Brute-Force .
Keamanan

ABSTRAK

Penelitian ini membahas implementasi algoritma password hashing Argon2id sebagai metode penguatan keamanan autentikasi pada sistem client–server. Argon2, pemenang Password Hashing Competition (PHC) tahun 2015, dirancang sebagai memory-hard function yang secara intensif memanfaatkan memori dan CPU untuk menghambat serangan brute-force dan GPU cracking berbasis paralel. Dalam studi ini, algoritma diimplementasikan pada proses enkripsi dan verifikasi kata sandi, serta dianalisis berdasarkan tingkat keamanan, kompleksitas komputasi, dan efektivitasnya dibandingkan dengan algoritma hash konvensional seperti MD5 dan SHA-256. Pengujian performa (Benchmarking) menunjukkan bahwa Argon2id membutuhkan waktu rata-rata 202,2238 ms, yang secara signifikan lebih lambat (sekitar 2.229 kali lebih lambat dari MD5 dan 1.111 kali dari SHA-256), sehingga efektif memberikan penalti waktu untuk mitigasi serangan. Selain itu, Avalanche Effect (Uji Kualitas Kriptografi) menghasilkan tingkat perubahan bit sebesar 34,39%, yang tergolong "Sangat Baik" dan membuktikan kemampuan difusi serta ketahanan terhadap analisis pola data. Implementasi Argon2id terbukti menghasilkan hash yang kuat dan tahan terhadap serangan brute-force berbasis paralel, sehingga meningkatkan keamanan autentikasi serta memberikan perlindungan lebih baik terhadap risiko kebocoran kata sandi dalam lingkungan sistem modern.

Corresponding Author:

Fernandoz Lim,
Email: Ferandlim.oz@gmail.com

1. PENDAHULUAN

Dalam era digital yang semakin maju, keamanan data dan sistem autentikasi menjadi aspek yang sangat penting karena hampir seluruh aktivitas manusia kini bergantung pada teknologi informasi, mulai dari transaksi keuangan, komunikasi daring, hingga pengelolaan data pribadi. Meningkatnya volume dan nilai data digital telah menyebabkan ancaman seperti pencurian identitas, *brute-force* attack, dan kebocoran data semakin kompleks, sehingga dibutuhkan penerapan algoritma kriptografi yang kuat untuk melindungi informasi [1].

Salah satu teknik utama dalam kriptografi adalah fungsi hash, yaitu metode yang mengubah data input menjadi string karakter dengan panjang tetap yang bersifat unik dan tidak dapat dikembalikan ke bentuk aslinya. Sifat ini menjadikannya ideal digunakan untuk penyimpanan kata sandi, tanda tangan digital, dan verifikasi integritas data [2]. Fungsi hash berperan penting dalam memastikan integritas dan autentikasi data, karena setiap perubahan kecil pada input akan menghasilkan nilai hash yang sepenuhnya berbeda. Dalam praktiknya, algoritma hash telah berkembang dari generasi awal seperti *MD5* dan *SHA-1* menuju algoritma yang lebih aman seperti *SHA-256* [3]. George menjelaskan bahwa meskipun algoritma lama seperti Bcrypt dapat beradaptasi dengan peningkatan daya pemrosesan, algoritma tersebut hanya mempertimbangkan biaya waktu komputasi, sehingga diperlukan algoritma yang lebih mutakhir untuk mencegah ancaman kriptografi yang luas. Salah satu yang paling mutakhir adalah Argon2, pemenang Password Hashing Competition (PHC) tahun 2015 [4].

Argon2 dirancang dengan konsep *memory-hard* function yang memanfaatkan sumber daya CPU dan memori secara intensif. Tujuannya adalah memperlambat proses *brute-force* yang dilakukan penyerang menggunakan perangkat keras paralel seperti GPU. Algoritma ini memiliki varian *Argon2id* yang menggabungkan ketahanan terhadap serangan *side-channel* dan serangan berbasis memori, menjadikannya standar rekomendasi terkini. Penggunaan algoritma modern seperti *Argon2id* terbukti lebih efektif dalam meningkatkan lapisan keamanan autentikasi pada sistem jaringan *client–server* karena mampu memitigasi risiko peretasan kata sandi secara langsung [5].

Penelitian ini bertujuan untuk mengimplementasikan algoritma *Argon2id* sebagai mekanisme pengamanan password. Fokus utama penelitian mencakup pengujian fungsionalitas sistem, analisis perbandingan performa (*Benchmarking*) terkait biaya waktu (*time cost*) terhadap algoritma konvensional *MD5* dan *SHA-256*, serta evaluasi kualitas enkripsi melalui uji *Avalanche Effect*. Dengan demikian, penelitian ini diharapkan dapat memberikan bukti empiris mengenai efektivitas *Argon2id* dalam menciptakan sistem autentikasi yang tidak hanya fungsional, tetapi juga memiliki standar keamanan kriptografi yang tinggi.

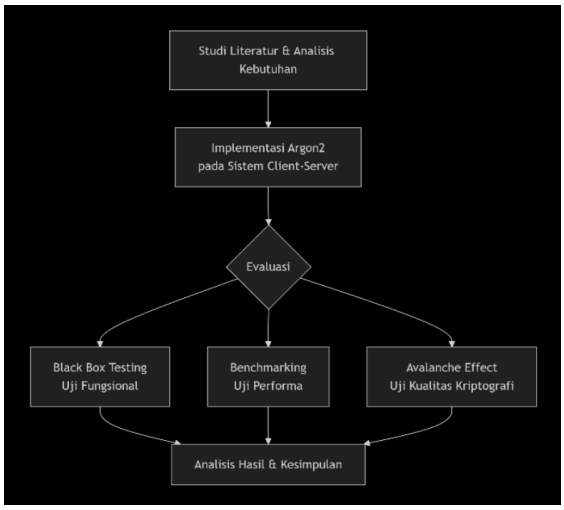
Dalam perkembangan kriptografi, algoritma SHA-512 telah menjadi standar yang luas digunakan untuk menjamin integritas data karena panjang bitnya yang besar dan resistensinya terhadap tabrakan data (collision). Namun, meskipun SHA-512 menawarkan tingkat keamanan yang tinggi dalam skenario hashing statis, algoritma ini tidak dirancang sebagai memory-hard function, sehingga tetap rentan terhadap serangan brute-force yang memanfaatkan akselerasi perangkat keras seperti GPU atau ASIC. Namun, Tippe dan Berne menekankan bahwa efektivitas Argon2 sangat bergantung pada konfigurasi parameter yang tepat; penggunaan memori yang sesuai dengan rekomendasi standar dapat mengurangi tingkat kompromi akun secara signifikan dibandingkan dengan penggunaan SHA-256 [6]. Guna mengatasi celah keamanan tersebut, penelitian ini mengimplementasikan Argon2id sebagai solusi mutakhir. Argon2id menggabungkan ketahanan terhadap serangan side-channel dari Argon2i dengan efisiensi melawan serangan time-memory tradeoff dari Argon2d, sehingga memberikan proteksi yang lebih tangguh dibandingkan algoritma hashing konvensional dalam mengamankan kredensial pengguna pada arsitektur client-server.

Keamanan autentikasi dalam era digital menjadi prioritas utama untuk melindungi data sensitif dari ancaman brute-force dan kebocoran data. Salah satu metode yang umum digunakan adalah algoritma SHA-512, yang menghasilkan output 512 bit melalui 80 kali proses looping untuk menjaga integritas data. Meskipun SHA-512 memiliki struktur kompleks, sifatnya yang cepat dan statis membuatnya rentan terhadap serangan sniffing serta eksploitasi perangkat keras paralel. Guna mengatasi keterbatasan tersebut, penelitian ini mengimplementasikan Argon2id sebagai mekanisme pengamanan kata sandi yang lebih tangguh. Berbeda dengan algoritma konvensional, Argon2id dirancang sebagai memory-hard function yang memberikan penalti biaya komputasi tinggi bagi penyerang. Penggunaan Argon2id terbukti lebih efektif dalam memitigasi risiko peretasan karena fleksibilitas konfigurasinya yang mampu menghambat penggunaan GPU secara masif. Dengan mengintegrasikan karakteristik ini, sistem jaringan client-server memperoleh perlindungan mutakhir yang memenuhi standar kriptografi modern.

Argon2 dirancang dengan konsep memory-hard function yang memanfaatkan sumber daya CPU dan memori secara intensif. Tujuannya adalah memperlambat proses brute-force yang dilakukan penyerang menggunakan perangkat keras paralel seperti GPU. Algoritma ini memiliki varian Argon2id yang menggabungkan ketahanan terhadap serangan side-channel dan serangan berbasis memori, menjadikannya standar rekomendasi terkini.

2. METODE PENELITIAN

Penelitian ini menguji keamanan penggunaan algoritma Argon2 untuk melindungi password pada sistem client-server. Metode pengujian dilakukan dalam tiga tahap utama. Pertama, *Black Box Testing* digunakan untuk memastikan semua fungsi aplikasi berjalan sesuai harapan tanpa melihat kode programnya [7]. Kedua, dilakukan *Benchmarking* untuk membandingkan kecepatan dan efisiensi Argon2 melawan algoritma lain [8]. Terakhir, pengujian *Avalanche Effect* dilakukan untuk membuktikan kualitas enkripsi, yaitu melihat apakah perubahan kecil pada password akan menghasilkan kode acak yang berubah total [9]. Hasilnya menunjukkan bahwa Argon2 memiliki performa yang kuat dan memenuhi standar keamanan kriptografi.



Gambar 1. *Flowchart Sistem Agron2 Dalam Pengamanan Password*

2.1. *Black Box Testing* (Uji Fungsional)

Pengujian fungsionalitas sistem dilakukan menggunakan pendekatan *Black Box Testing* untuk memvalidasi integritas alur autentikasi keamanan tanpa perlu mengintervensi kode program internal. Prosedur pengujian dirancang dalam dua skenario kritis. Skenario pertama berfokus pada proses registrasi pengguna, di mana pengujian dilakukan dengan mendaftarkan akun baru dan dilanjutkan dengan inspeksi langsung terhadap tabel basis data. Indikator keberhasilan utama pada tahap ini adalah tersimpannya kata sandi dalam format hash enkripsi algoritma *Argon2id* yang ditandai dengan prefiks identifier argon2 dan bukan dalam bentuk teks asli (*plain text*). Skenario kedua adalah pengujian autentikasi login yang mencakup pengujian positif dan negatif. Pada pengujian positif, sistem diuji dengan memasukkan kredensial yang valid untuk memastikan mekanisme verifikasi hash mampu mencocokkan input dengan data terenkripsi dan memberikan hak akses. Sebaliknya, pengujian negatif dilakukan dengan memasukkan kombinasi kata sandi yang salah untuk memverifikasi bahwa sistem mampu menolak akses secara efektif.

2.1. *Benchmarking* (Uji Performa)

Tahap selanjutnya adalah pengujian performa (*Benchmarking*) yang bertujuan untuk mengukur beban komputasi (computational cost) dari algoritma *Argon2id* dibandingkan dengan algoritma hashing standar, yaitu *MD5* dan *SHA-256*. Metode komparasi ini dirancang untuk membuktikan efektivitas *Argon2id* dalam memberikan penalti waktu (*time delay*) guna memitigasi serangan *brute-force*. Prosedur pengujian dilakukan dengan langkah-langkah sebagai berikut:

1. Penyiapan Lingkungan: Ketiga algoritma dijalankan pada perangkat keras (*server environment*) yang sama untuk menjaga konsistensi variabel pengujian.
2. Penetapan Sampel: Menggunakan satu string input standar (misalnya "12345678") sebagai objek uji bagi ketiga algoritma.
3. Iterasi Pengujian: Proses hashing dijalankan sebanyak 10 kali iterasi (*looping*) berturut-turut. Hal ini dilakukan untuk mendapatkan data yang stabil dan meminimalisir bias akibat fluktuasi kinerja CPU sesaat.
4. Pengukuran Metrik: Waktu eksekusi (execution time) untuk setiap iterasi dicatat dalam satuan milidetik (ms).
5. Analisis Data: Hasil pencatatan dari 10 iterasi tersebut kemudian dirata-ratakan (*average*) untuk mendapatkan nilai waktu eksekusi final. Selain itu, dihitung pula "Rasio Kelambatan" (Delay Ratio), yaitu perbandingan relatif antara kecepatan *Argon2id* terhadap algoritma dasar (*MD5*) untuk mengkuantifikasi peningkatan keamanan secara matematis.

2.1. *Avalanche Effect* (Uji Kualitas Kriptografi)

Tahap akhir penelitian ini menerapkan pengujian *Avalanche Effect* untuk mengevaluasi kualitas difusi algoritma *Argon2id*. Prinsip utama pengujian ini adalah memverifikasi bahwa perubahan minor pada input harus menghasilkan perubahan mayor pada output guna mencegah serangan analisis pola. Prosedur pengujian dilakukan dengan memproses dua sampel string yang memiliki tingkat kemiripan tinggi, yakni "password" dan "password1", menggunakan parameter *Argon2id* yang identik untuk menghasilkan dua hash berbeda. Kedua hasil hash tersebut kemudian dikomparasi menggunakan prinsip *Hamming Distance* untuk mengukur seberapa jauh perbedaan struktur bit antar keduanya. Perhitungan dilakukan dengan membandingkan jumlah bit yang berubah terhadap total panjang bit hash dan dinyatakan dalam bentuk persentase. Semakin tinggi persentase

perubahan yang dihasilkan, semakin baik kemampuan algoritma dalam mengacak pola data dan menyembunyikan korelasi antara input dan output.

3. HASIL DAN ANALISIS

Hasil pengujian komprehensif terhadap penerapan algoritma *Argon2id* sebagai mekanisme pengamanan data otentikasi telah dilakukan. Evaluasi dilakukan berdasarkan tiga metode utama yang telah ditetapkan, yaitu validitas fungsional sistem, membandingkan antar hashing, dan kualitas kriptografi. Data yang disajikan berikut ini bertujuan untuk membuktikan hipotesis bahwa *Argon2id* mampu memberikan peningkatan keamanan yang signifikan dibandingkan metode hashing konvensional tanpa mengorbankan fungsionalitas sistem secara keseluruhan.

3.1. Implementasi dan Pengujian Fungsional

Tahap implementasi ini berfokus pada penerapan keamanan password menggunakan algoritma *Argon2id* dan Java sebagai bahasa pemograman utama yang diterapkan pada website. Konfirgurasi algoritma ini disesuaikan dengan parameter keamanan yang telah ditentukan untuk menyeimbangkan keamanan dan performa server.

3.1.1. Program Registrasi

Algoritma *Argon2id* diterapkan pada saat pengguna melakukan pendaftaran. Setelah pengguna melakukan pendaftaran, maka data yang telah diterima akan disimpan di database yang dimana saat ini menggunakan MYSQL. Kode program sebagai berikut

```
            passwordHash = argon2.hash(3, 65536, 2, passwordChars);
            String sql = "INSERT INTO users (username, email, password_hash) VALUES (?,
?, ?)";

            try (Connection conn = DatabaseUtil.getConnection();
                 PreparedStatement pstmt = conn.prepareStatement(sql)) {

                pstmt.setString(1, username);
                pstmt.setString(2, email);
                pstmt.setString(3, passwordHash);

                pstmt.executeUpdate();
                request.setAttribute("message", "Registrasi berhasil! Silakan login.");
                request.getRequestDispatcher("login.jsp").forward(request, response);

            }
            }
```

Gambar 2. *Kode Program Registrasi*

Proses diawali dengan memanggil fungsi *argon2.hash* untuk mengonversi kata sandi pengguna *(plain text)* menjadi format terenkripsi. Konfigurasi algoritma ditetapkan dengan parameter spesifik, yaitu 3 iterasi waktu, penggunaan memori 65.536 KB (64 MB), dan 2 *thread paralel* guna menghasilkan hash yang memiliki resistensi tinggi terhadap serangan *brute-force*. Selanjutnya, penyimpanan data ke dalam tabel users dilakukan menggunakan mekanisme PreparedStatement. Pendekatan ini dipilih tidak hanya untuk mencegah serangan *SQL Injection*, tetapi juga untuk memastikan bahwa nilai yang disuntikkan ke kolom *password_hash* adalah murni hasil enkripsi. Setelah eksekusi kueri *INSERT* berhasil, sistem memberikan umpan balik berupa pesan sukses dan secara otomatis mengarahkan pengguna ke halaman *login*, menandakan bahwa akun telah terbentuk dengan standar keamanan yang ditetapkan. Hasil Pendaftaran sebagai berikut.

Gambar 3. *Hasil Insert Database pada MySQL*

Hal ini terbukti bahwa password yang didaftarkan sudah berhasil di hash dengan algoritma *Argon2id*. Saat pendaftaran berlangsung password yang awalnya hanya text biasa diolah atau dienkripsi menjadi teks yang tidak biasa dimana akan sulit dibobol dari serangan siber.

3.1.2. Program Login

Berbeda dengan sistem konvensional yang melakukan pencocokan kata sandi langsung di dalam kueri SQL, implementasi *Argon2id* mengharuskan proses verifikasi dilakukan di sisi aplikasi . Hal ini dikarenakan setiap hash Argon2 memiliki salt yang unik, sehingga input yang sama tidak akan menghasilkan hash yang sama jika dienkripsi ulang secara manual. Alur logika autentikasi dibangun dengan tahapan sebagai berikut:

1. Sistem mencari data pengguna berdasarkan username.
2. Sistem mengambil string hash yang tersimpan di kolom password_hash.
3. Fungsi verifikasi (argon2.verify) membandingkan password inputan pengguna dengan hash dari database.

Berikut adalah implementasi kode program untuk proses verifikasi login.

```
String sql = "SELECT password_hash FROM users WHERE username = ?";
        try (Connection conn = DatabaseUtil.getConnection();
             PreparedStatement pstmt = conn.prepareStatement(sql)) {
            pstmt.setString(1, username);
            try (ResultSet rs = pstmt.executeQuery()) {
                if (rs.next()) {
                    String storedHash = rs.getString("password_hash");
                    if (argon2.verify(storedHash, passwordChars)) {
                        HttpSession session = request.getSession();
                        session.setAttribute("username", username);
                        response.sendRedirect("index.jsp");
                        return;
                    } else {
                        request.setAttribute("error", "Username atau password salah.");
                        request.getRequestDispatcher("login.jsp").forward(request,
response);
                    }
                } else {
                    request.setAttribute("error", "Username atau password salah.");
                    request.getRequestDispatcher("login.jsp").forward(request,
response);
                }
            }
```

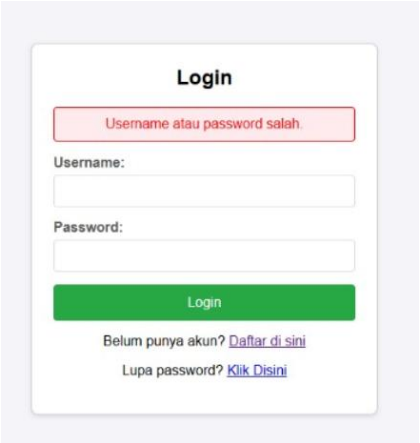
Gambar 4. *Kode Program Login*

Proses autentikasi dimulai dengan mengeksekusi kueri SQL untuk mengambil string password_hash dari basis data yang sesuai dengan username inputan pengguna. Perlu ditekankan bahwa sistem hanya mengambil data terenkripsi, bukan kata sandi asli. Inti dari keamanan proses ini terletak pada pemanggilan fungsi argon2.verify(storedHash, passwordChars). Fungsi ini secara otomatis mendekode hash yang tersimpan untuk mengekstrak salt unik dan parameter algoritma (iterasi dan memori) yang digunakan saat registrasi, kemudian menerapkannya pada kata sandi yang baru dimasukkan. Jika hasil komputasi tersebut cocok (true), sistem akan menginisialisasi sesi baru (HttpSession) sebagai tanda otorisasi akses yang sah. Sebaliknya, jika tidak cocok, sistem akan menolak permintaan login, memastikan bahwa verifikasi kredensial dilakukan sepenuhnya secara matematis di sisi server tanpa pernah membandingkan teks asli (*plain text*).

Untuk memvalidasi logika program di atas, dilakukan pengujian antarmuka dengan dua pengujian. Pengujian pertama adalah pengujian login dengan kredensial valid. Sebagaimana ditunjukkan pada Gambar 3.4, sistem berhasil memverifikasi kecocokan hash dan memberikan hak akses kepada pengguna untuk memasuki halaman utama (dashboard).

Gambar 5. *Hasil Login dan Memasuki Halaman Utama (Dashboard)*

Pengujian kedua adalah pengujian keamanan dengan memasukkan kata sandi yang salah. Hasil pengujian pada Gambar 3.5 memperlihatkan bahwa sistem secara responsif menolak permintaan akses dan menampilkan notifikasi kesalahan tanpa memberikan detail spesifik, sesuai dengan prinsip keamanan untuk mencegah enumerasi akun.



Gambar 6. *Hasil Sistem Responsif Penolakan*

3.2. Pengujian Perbandingan *Benchmarking*

Pengujian performa (*Benchmarking*) dilakukan untuk mengukur beban komputasi (computational cost) yang dihasilkan oleh algoritma *Argon2id* dibandingkan dengan algoritma hashing konvensional (*MD5* dan *SHA-256*). Pengujian dilakukan sebanyak 10 kali iterasi (run) menggunakan sampel input yang sama ("12345678") untuk mendapatkan nilai rata-rata waktu eksekusi yang valid dan konsisten.

Tabel 1.*Rekapitulasi Pengukuran Waktu Eksekusi Ketiga Algoritma*

Iterasi Ke-	*MD5* (ms)	*SHA-256* (ms)	*Argon2id* (ms)
1	0.1473	0.1074	189.7604
2	0.0773	0.2054	225.9636
3	0.0806	0.2718	191.0782
4	0.0806	0.2718	191.0782
5	0.0806	0.2718	191.0782
6	0.0670	0.0846	206.8024
7	0.1241	0.1542	184.8197
8	0.1523	0.1651	211.8703
9	0.0624	0.1564	217.6254
10	0.0353	0.1307	212.1619
Rata-Rata	**0.0907 ms**	**0.1819 ms**	**202.2238 ms**

Berdasarkan data rata-rata yang tersaji pada Tabel 4.1, terdapat disparitas performa yang sangat signifikan antara algoritma hashing konvensional dengan *Argon2id*. Algoritma *MD5* mencatat waktu eksekusi rata-rata tercepat sebesar 0,0907 ms, diikuti oleh *SHA-256* dengan 0,1819 ms. Sementara itu, implementasi *Argon2id* membutuhkan waktu rata-rata 202,2238 ms untuk menyelesaikan satu kali proses hashing.
Secara matematis, hasil ini menunjukkan bahwa *Argon2id* bekerja sekitar 2.229 kali lebih lambat dibandingkan *MD5* dan 1.111 kali lebih lambat dibandingkan *SHA-256*. Dalam konteks keamanan siber, kesenjangan waktu eksekusi ini memiliki implikasi krusial sebagai berikut:

1. **Peningkatan Biaya Serangan (*Cost of Attack*):** Kecepatan tinggi pada *MD5* dan *SHA-256* merupakan kerentanan fatal (*vulnerability*) karena memungkinkan penyerang melakukan serangan *brute-force* dengan kecepatan jutaan percobaan per detik. Dengan menerapkan *Argon2id*, sistem secara efektif memaksa

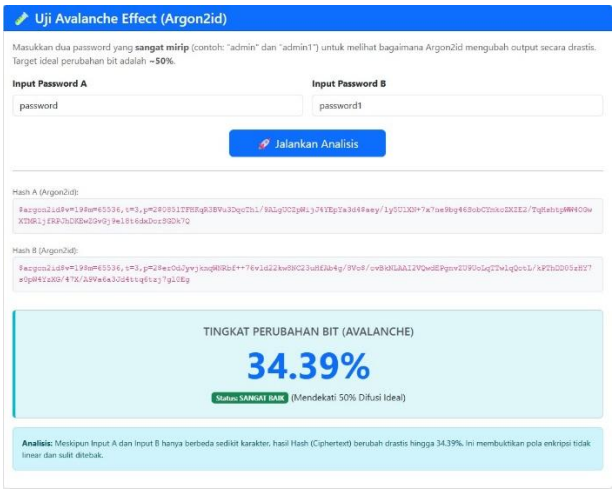
penyerang untuk menghabiskan waktu ~200 milidetik untuk setiap tebakan password. Hal ini meningkatkan "biaya waktu" serangan hingga ribuan kali lipat, menjadikan upaya peretasan secara paksa menjadi tidak efisien dan memakan waktu yang sangat lama.

2. **Keseimbangan Keamanan dan Fungsionalitas:** Meskipun *Argon2id* jauh lebih lambat bagi mesin, durasi rata-rata 202 ms masih berada di bawah ambang batas persepsi respons manusia (umumnya 200-500 ms) [10]. Artinya, pengguna sah yang melakukan login tidak akan merasakan penundaan (*lag*) yang mengganggu, namun sistem tetap memiliki pertahanan yang kokoh terhadap serangan mesin otomatis.

Hasil pengujian ini membuktikan hipotesis penelitian bahwa *Argon2id* mampu memberikan perlindungan yang jauh lebih unggul dibandingkan algoritma standar tanpa mengorbankan kenyamanan pengguna aplikasi

3.3. Evaluasi kualitas kriptografi

Pengujian kualitas kriptografi dilakukan untuk memverifikasi properti difusi pada algoritma *Argon2id*, sebuah karakteristik yang mensyaratkan bahwa perubahan minor pada input harus menghasilkan perubahan mayor pada struktur output (*ciphertext*). Proses pengujian dilaksanakan dengan membandingkan dua string masukan yang memiliki tingkat kemiripan tinggi, yakni *'password'* sebagai *Input A* dan *'password1'* sebagai *Input B*. Kedua sampel tersebut diproses menggunakan konfigurasi parameter *Argon2id* yang identik untuk memastikan validitas perbandingan.



Gambar 7. *Hasil Komparasi Visual dan Perhitungan Persentase*

Berdasarkan Gambar 4.6, hasil pengujian menunjukkan tingkat perubahan bit (*Avalanche Effect*) sebesar 34,39%. Meskipun kedua input hanya memiliki selisih satu karakter (penambahan angka "1"), algoritma *Argon2id* menghasilkan dua string hash yang memiliki struktur bit yang sangat berbeda.

Alat pengujian mengkategorikan hasil 34,39% ini dalam status "Sangat Baik". Hal ini mengindikasikan implikasi keamanan sebagai berikut:

1. **Non-Linearitas:** Tidak ditemukan korelasi linear antara input dan output. Penyerang tidak dapat memprediksi perubahan pada hash hanya dengan mengetahui perubahan kecil pada password asli.
2. **Ketahanan terhadap Analisis Diferensial:** Tingkat perubahan yang signifikan ini menyulitkan serangan *differential cryptanalysis*, di mana penyerang mencoba mencari pola kunci dengan membandingkan perbedaan input dan output.

Dengan demikian, terbukti bahwa implementasi *Argon2id* pada sistem ini memenuhi standar kualitas kriptografi yang baik dalam hal pengacakan dan penyembunyian pola data.

4. KESIMPULAN

Penelitian ini berhasil mencapai tujuan yang telah ditetapkan dalam Bab Pendahuluan, yaitu mengimplementasikan algoritma *password hashing* ***Argon2id*** pada sistem autentikasi *client–server* dan menganalisis efektivitasnya dalam menanggulangi serangan *brute-force* berbasis paralel. Hasil dan Pembahasan secara konsisten membuktikan bahwa *Argon2id*, sebagai *memory-hard function*, memberikan penalti waktu eksekusi yang tinggi (rata-rata 202,2238 ms), yang merupakan strategi mitigasi serangan yang jauh lebih unggul dibandingkan algoritma konvensional (MD5 dan SHA-256). Kualitas kriptografi Argon2id juga terverifikasi "Sangat Baik" melalui pengujian *Avalanche Effect* dengan tingkat perubahan bit sebesar 34,39%. Dengan demikian, **terjadi kecocokan antara tujuan yang diajukan di awal penelitian dan hasil akhir yang dicapai**, di mana implementasi Argon2id terbukti mampu meningkatkan keamanan autentikasi dan memberikan perlindungan data kata sandi yang kuat.

Prospek pengembangan hasil penelitian ini dapat difokuskan pada analisis mendalam terhadap konfigurasi parameter Argon2id (seperti *memory cost*, *time cost*, dan *parallelism*) untuk mendapatkan titik optimal keamanan dan kinerja di berbagai lingkungan *server* yang berbeda. **Prospek penerapan penelitian selanjutnya** adalah mengintegrasikan *Argon2id* tidak hanya pada autentikasi *client–server* biasa, tetapi juga pada sistem yang lebih sensitif seperti sistem perbankan digital, *blockchain*, atau *Zero-Knowledge Proofs*, di mana ketahanan terhadap serangan berbasis *time-memory tradeoff* menjadi kritikal, sehingga penelitian ini dapat berkontribusi pada standar keamanan data di masa depan.

DAFTAR PUSTAKA

[1] M. A. S. A. Toras Pangidoan Batubara, "Perbandingan Algoritma Kriptografi Hash Sha 256 dan Sha 512 Untuk Keamanan Sandi," vol. 3, no. 1, pp. 6–10, 2025.

[2] J. Hutagalung, P. S. Ramadhan, S. J. Sihombing, and P. Korespondensi, "KEAMANAN DATA MENGGUNAKAN SECURE HASHING ALGORITHM (SHA) - 256 DAN RIVEST SHAMIR ADLEMAN (RSA) PADA DIGITAL SIGNATURE IMPLEMENTATION OF SECURE HASHING ALGORITHM (SHA) -256 AND RIVEST SHAMIR ADLEMAN (RSA) ON DIGITAL SIGNATURE," vol. 10, no. 6, 2023, doi: 10.25126/jtiik.2023107319.

[3] N. T. Jehian *et al.*, "PENGEMBANGAN SISTEM KEAMANAN DATA BERBASIS WEB MENGGUNAKAN KOMBINASI ALGORITMA CHACHA20-POLY1305 DAN ARGON2," vol. 13, no. 3, 1958.

[4] A. T. George, "Argon2 : The Secure Password Hashing Function," vol. 3, no. 1, pp. 81–84, 2021, doi: 10.5281/zenodo.5091700.

[5] D. K. Alex Biryukov, Daniel Dinu, "Argon2 : new generation of memory-hard functions for password hashing and other applications," no. July 2015, 2016, doi: 10.1109/EuroSP.2016.31.

[6] P. T. and M. P. Berner, "Real-World Software".

[7] L. C. Jain, H. S. Behera, J. K. Mandal, and D. P. Mohapatra, *Computational Intelligence in Data Mining - Volume 3*, vol. 3, no. December. 2014.

[8] S. Eum, "Optimized Implementation of Argon2 Utilizing the Graphics Processing Unit," 2023.

[9] A. F. W. and S. E. Tavares, "ON THE DESIGN OF S-BOXES," pp. 523–534, 1986.

[10] R. B. Miller, "Response time in man-computer conversational transactions".