Jurnal Komputer Antartika
Volume 4 Nomor 2 2026 Halaman 61-68
ISSN (Online) 2988-5272
DOI: https://doi.org/10.70052/jka.v4i2.1377

Double Layer Encryption Caesar Cipher dan AES-128 untuk Pengamanan Data Teks *Web*

Siboy Arisona[1], Nanda Aulia Ramadlani[2*], Muhlis Tahir[3]

[1,2,3] Universitas Trunojoyo Madura
Jl. Raya Telang, PO BOX 2, Kamal, Bangkalan, Madura, Jawa Timur, Indonesia

e-mail korespondensi: nanda2210aulia@gmail.com

Submit: 26-05-2026 | Revisi: 05-06-2026 | Terima: 14-06-2026 | Terbit online: 27-06-2026

Abstrak - Keamanan data teks pada aplikasi web menjadi perhatian penting karena tingginya risiko penyadapan dan pencurian informasi digital. Penggunaan algoritma enkripsi tunggal masih memiliki kelemahan terhadap teknik kriptanalisis modern sehingga diperlukan metode pengamanan yang lebih kuat. Penelitian ini bertujuan membangun sistem pengamanan data teks berbasis *web* menggunakan metode *double layer encryption* dengan menggabungkan algoritma Caesar Cipher dan AES-128. Proses enkripsi dilakukan dalam dua tahap, yaitu *plaintext* dienkripsi menggunakan Caesar Cipher kemudian dilanjutkan dengan AES-128 untuk menghasilkan *ciphertext* yang lebih kompleks. Hasil penelitian menunjukkan bahwa kombinasi kedua algoritma mampu meningkatkan keamanan data karena *ciphertext* lebih sulit dianalisis dibandingkan penggunaan algoritma tunggal. Sistem yang dibangun juga mampu menjalankan proses enkripsi dan dekripsi dengan baik serta memberikan perlindungan data yang lebih baik.

Kata Kunci : AES-128, Caesar Cipher, Double Layer Encryption, Keamanan Data

Abstract *- Text data security in web-based applications has become important due to the increasing risk of interception and digital information theft. The use of a single encryption algorithm still has weaknesses against modern cryptanalysis techniques to modern cryptanalysis techniques; therefore, stronger security methods are required. This study aims to develop a web-based text security system using a double layer encryption method by combining Caesar Cipher and AES-128. The encryption process is carried out in two stages, where plaintext is encrypted using Caesar Cipher and then re-encrypted using AES-128 to produce more complex ciphertext. The results show that the combination of both algorithms improves data security because the ciphertext becomes more difficult to analyze compared to a single algorithm approach. The developed system is also capable of performing encryption and decryption processes properly while providing better protection for digital data.*

Keywords : AES-128, Caesar Cipher, Double Layer Encryption, Data Security

1. Pendahuluan

Pertukaran informasi pada aplikasi web berkembang sangat pesat dan memungkinkan pengiriman data secara cepat serta mudah diakses. Namun, kondisi tersebut juga meningkatkan risiko keamanan data karena informasi sensitif dapat diintersepsi oleh pihak yang tidak berwenang [1], [2]. Oleh sebab itu, diperlukan mekanisme keamanan yang mampu menjaga kerahasiaan dan integritas data selama proses pertukaran informasi digital berlangsung. Salah satu teknologi yang banyak digunakan untuk mengatasi permasalahan tersebut adalah kriptografi, yaitu teknik pengamanan data melalui proses enkripsi dan dekripsi menggunakan kunci tertentu [3], [4]. Salah satu algoritma kriptografi klasik yang masih sering digunakan sebagai dasar pembelajaran keamanan data adalah Caesar Cipher. Algoritma ini bekerja menggunakan teknik substitusi dengan menggeser karakter *plaintext* berdasarkan nilai kunci tertentu [1], [5]. Caesar Cipher memiliki keunggulan pada proses komputasi yang sederhana dan cepat, namun tingkat keamanannya relatif rendah karena rentan terhadap serangan *brute force* maupun analisis frekuensi [5], [6]. Untuk meningkatkan keamanan data, algoritma modern seperti Advanced Encryption Standard (AES-128) banyak digunakan karena memiliki tingkat keamanan tinggi melalui proses transformasi blok dan operasi matematis yang kompleks [7], [8]. AES-128 juga dinilai efisien dan banyak diterapkan pada sistem pengamanan data digital maupun aplikasi web [9], [10].

Beberapa penelitian sebelumnya telah menerapkan kombinasi algoritma kriptografi untuk meningkatkan keamanan data. Penelitian oleh Nuraeni et al. [11] menunjukkan bahwa penggabungan Caesar Cipher dan AES mampu meningkatkan kompleksitas *ciphertext* pada proses pengamanan data. Penelitian lain juga menerapkan konsep superenkripsi pada pengamanan file digital menggunakan AES-128 untuk menghasilkan sistem keamanan

yang lebih kuat [12]. Selain itu, Alatawi [13] menjelaskan bahwa penggunaan hybrid cryptographic cipher dapat meningkatkan keamanan komunikasi digital melalui kombinasi beberapa metode enkripsi, sedangkan Nasrullah [14] menyatakan bahwa AES-128 efektif digunakan pada sistem keamanan data berbasis *web*. Metode pengamanan data menggunakan algoritma kriptografi juga telah diterapkan pada berbagai jenis data digital, seperti pengamanan citra, dokumen, serta data penjualan [10], [15]. Penelitian lain menunjukkan bahwa penggunaan AES dengan berbagai ukuran kunci mampu meningkatkan perlindungan data digital secara signifikan [8]. Selain itu, penerapan Caesar Cipher pada sistem keamanan teks masih relevan digunakan sebagai lapisan awal enkripsi karena memiliki proses komputasi yang ringan [6].

Berdasarkan penelitian sebelumnya, kombinasi algoritma klasik dan modern dinilai mampu meningkatkan keamanan data digital. Namun, implementasi *double layer encryption* menggunakan Caesar Cipher dan AES-128 pada sistem pengamanan data teks berbasis *web* masih belum banyak dikembangkan. Oleh karena itu, penelitian ini bertujuan membangun sistem keamanan data teks berbasis *web* menggunakan metode *double layer encryption* untuk meningkatkan kompleksitas *ciphertext* serta meminimalkan risiko kebocoran data pada jaringan digital.

2. Metode Penelitian

Penelitian ini menggunakan metode pengembangan sistem keamanan data berbasis web dengan menerapkan teknik double layer encryption menggunakan algoritma Caesar Cipher dan AES-128. Sistem dirancang untuk melakukan proses enkripsi dan dekripsi data teks secara berlapis guna meningkatkan keamanan informasi pada aplikasi web. Tahapan penelitian ini dilaksanakan secara sistematis melalui tiga sub-proses utama, yaitu perancangan sistem, implementasi algoritma, dan pengujian proses enkripsi serta dekripsi.

Gambar 1. Tahapan Penelitian

Gambar 1 menunjukkan tahapan penelitian yang dilakukan dalam pengembangan sistem pengamanan data teks berbasis web menggunakan metode *double layer encryption*. Penelitian diawali dengan analisis kebutuhan untuk mengidentifikasi kebutuhan sistem dan permasalahan keamanan data yang akan diselesaikan. Selanjutnya dilakukan perancangan sistem yang meliputi desain antarmuka, alur proses, dan mekanisme penyimpanan data. Tahap implementasi dilakukan dengan menerapkan algoritma Caesar Cipher sebagai lapisan enkripsi pertama dan AES-128 sebagai lapisan enkripsi kedua. Kedua algoritma kemudian diintegrasikan menjadi mekanisme *double layer encryption*. Setelah implementasi selesai, dilakukan pengujian sistem untuk memastikan proses enkripsi dan dekripsi berjalan sesuai kebutuhan. Tahap terakhir adalah analisis hasil pengujian untuk mengevaluasi keberhasilan sistem dalam meningkatkan keamanan data teks.

2.1 Perancangan Sistem

Sistem ini dirancang menggunakan arsitektur *client-server* berbasis web. Antarmuka pengguna (*front-end*) dibangun menggunakan HTML dan CSS untuk memfasilitasi input plaintext serta visualisasi hasil enkripsi/dekripsi. Bagian pemrosesan (*back-end*) menggunakan bahasa pemrograman PHP untuk mengeksekusi logika kriptografi. Data teks yang telah diubah menjadi *ciphertext* akhir kemudian akan dikirim secara aman untuk disimpan ke dalam basis data MySQL, lengkap dengan informasi penanda waktu (*timestamp*) dan identitas pengirim. Keamanan awal sistem juga dirancang menggunakan modul otentikasi login berbasis enkripsi kata sandi untuk membatasi akses pengguna.

2.2 Implementasi Algoritma

Implementasi pengamanan data menggunakan pendekatan kombinasi dua lapis secara berurutan (*sequential double-layer encryption*). Lapisan pertama menerapkan Caesar Cipher sebagai penyandi awal yang bertugas mengacak susunan karakter plaintext berdasarkan nilai pergeseran (*shift value*) tertentu. Selanjutnya, hasil keluaran dari lapisan pertama (*intermediate ciphertext*) langsung dijadikan input bagi lapisan kedua, yaitu algoritma modern AES-128. AES-128 akan mengolah teks tersebut dalam blok data berukuran 128-bit

menggunakan kunci simetris 16 karakter melalui serangkaian transformasi matematis yang kompleks, sehingga menghasilkan berkas *ciphertext* akhir yang memiliki tingkat keamanan ganda.

2.3 Pengujian Proses Enkripsi dan Dekripsi

Pengujian sistem dilakukan menggunakan metode *black-box testing* untuk memvalidasi fungsionalitas fungsional utama dari modul kriptografi. Skenario pengujian disusun dengan menguji sistem menggunakan variasi muatan plaintext, variasi nilai pergeseran Caesar Cipher, dan variasi kata kunci AES-128. Pengujian difokuskan pada dua aspek, yakni uji keberhasilan pemulihan data (proses dekripsi dengan kunci yang benar) dan uji ketahanan kegagalan sistem (proses dekripsi dengan simulasi kunci atau nilai pergeseran yang salah).

2.4 Caesar Cipher

Pada penelitian ini, proses Caesar Cipher dilakukan menggunakan representasi karakter berbasis ASCII. Karakter *plaintext* terlebih dahulu dikonversi ke bentuk desimal menggunakan tabel ASCII sebelum dilakukan proses pergeseran karakter. Penggunaan kode ASCII bertujuan agar proses enkripsi dapat diterapkan pada karakter yang dapat ditampilkan (*printable character*) [11]. Himpunan karakter ASCII yang digunakan pada penelitian ini ditunjukkan pada Gambar 2.

Gambar 2. Karakter Printable ASCII

Gambar 2 menampilkan tabel referensi karakter *printable* ASCII dari nilai desimal 32 (karakter spasi) hingga nilai desimal 127. Karakter yang digunakan pada penelitian ini berada pada rentang desimal 32 sampai 127 sehingga total karakter yang digunakan sebanyak 94 karakter. Penggunaan seluruh rentang ini memastikan bahwa setiap input teks berupa huruf kapital, huruf kecil, angka, spasi, maupun simbol khusus pada papan ketik standar dapat dipetakan secara konsisten oleh sistem tanpa menyebabkan hilangnya data (*data loss*).

Proses enkripsi Caesar Cipher dilakukan menggunakan metode pergeseran karakter berbasis ASCII sebagaimana dijelaskan pada penelitian sebelumnya [1], [2]. Secara matematis, proses enkripsi Caesar Cipher dituliskan pada persamaan (1).

$$C_i = 32 + ((P_i + k) mod\ N)\ (1)$$

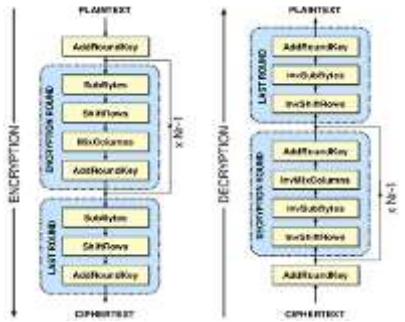
Sedangkan proses dekripsi Caesar Cipher dituliskan pada persamaan (2).

$$P_i = 32 + ((C_i - k) mod\ N)\ (2)$$

Pada persamaan (1) dan persamaan (2), C_i menyatakan *ciphertext* indeks ke-i, P_i menyatakan *plaintext* indeks ke-i, k merupakan nilai pergeseran karakter, sedangkan N adalah jumlah karakter yang digunakan dalam himpunan ASCII (N = 94) [1], [11].

2.5 Advanced Encryption Standard (AES-128)

Advanced Encryption Standard (AES-128) merupakan algoritma kriptografi modern berbasis *symmetric key* dengan panjang kunci 128-bit [7], [8]. Algoritma ini bekerja melalui beberapa tahapan transformasi seperti *Key Expansion*, *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* untuk menghasilkan *ciphertext* yang memiliki tingkat keamanan tinggi. Alur proses enkripsi dan dekripsi pada AES-128 ditunjukkan pada Gambar 3.

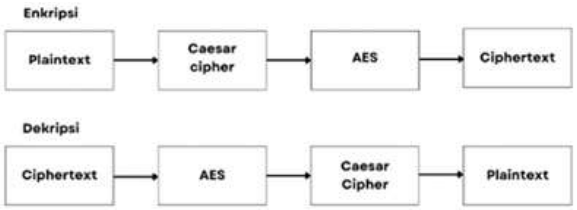


Gambar 3. Alur Enkripsi dan Dekripsi AES-128

Gambar 3 menjabarkan struktur internal algoritma AES-128. Pada blok enkripsi sebelah kiri, data diproses melalui 10 putaran (*rounds*). Sembilan putaran awal mengeksekusi tahapan *SubBytes* (substitusi bita lewat S-Box), *ShiftRows* (pergeseran baris state), *MixColumns* (pengacakan kolom), dan *AddRoundKey* (XOR dengan round key), sementara putaran terakhir mengabaikan tahap *MixColumns*. Pada blok dekripsi sebelah kanan, Gambar 3 menunjukkan alur pembalikan arah dengan memanfaatkan fungsi inversi (*InvSubBytes, InvShiftRows*, *InvMixColumns*) secara berurutan dari bawah ke atas demi memulihkan data. Pada penelitian ini, AES-128 digunakan sebagai lapisan kedua setelah proses Caesar Cipher selesai dilakukan. Penggunaan AES-128 bertujuan meningkatkan kompleksitas *ciphertext* sehingga data menjadi lebih sulit dianalisis maupun diretas menggunakan teknik serangan sederhana [12].

2.6 Double Layer Encryption

Metode *double layer encryption* pada penelitian ini dilakukan dengan menggabungkan Caesar Cipher dan AES-128 secara berurutan. Proses dimulai ketika pengguna memasukkan *plaintext* dan kunci melalui antarmuka aplikasi. *Plaintext* terlebih dahulu dienkripsi menggunakan Caesar Cipher untuk menghasilkan *ciphertext* tahap pertama [4], [11]. Selanjutnya, hasil *ciphertext* tersebut diproses kembali menggunakan AES-128 untuk menghasilkan *ciphertext* akhir yang memiliki tingkat keamanan lebih tinggi [2], [12]. Pada proses dekripsi, sistem melakukan tahapan kebalikan dimulai dari dekripsi AES-128 kemudian dilanjutkan dengan dekripsi Caesar Cipher hingga diperoleh *plaintext asli*. Alur proses *double layer encryption* ditunjukkan pada Gambar 4.



Gambar 4. Alur Double Layer Encryption

Gambar 4 memperlihatkan diagram blok skema kerja metode pengamanan berlapis yang diajukan. Sisi enkripsi menunjukkan aliran data linear dari *plaintext* menuju blok Caesar Cipher, menghasilkan ciphertext tahap pertama, yang kemudian diproses kembali menggunakan algoritma AES-128 untuk memproduksi *ciphertext* akhir. Sebaliknya, diagram dekripsi pada Gambar 4 memperlihatkan bahwa *ciphertext* akhir wajib dibongkar terlebih dahulu oleh algoritma AES sebelum diteruskan ke unit dekripsi Caesar Cipher untuk menghasilkan *plaintext* asal yang valid. Pengujian sistem dilakukan menggunakan beberapa variasi *plaintext* dan kunci untuk memastikan proses enkripsi dan dekripsi berjalan dengan baik. Hasil pengujian digunakan untuk menganalisis keberhasilan sistem dalam meningkatkan keamanan data teks berbasis web.

3. Hasil dan Pembahasan

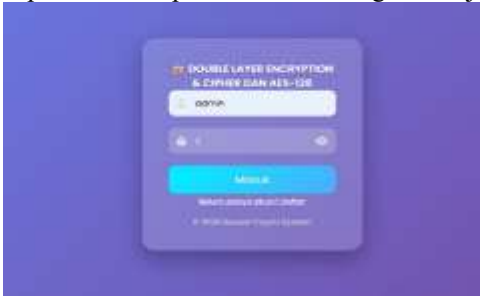
Pada penelitian ini berhasil dikembangkan sistem pengamanan data teks berbasis *web* menggunakan metode *double layer encryption* dengan kombinasi algoritma Caesar Cipher dan AES-128. Sistem dibangun menggunakan bahasa pemrograman PHP dengan antarmuka berbasis HTML dan terintegrasi dengan database untuk penyimpanan *ciphertext*. Hasil penelitian meliputi implementasi sistem, proses enkripsi dan dekripsi, serta analisis keamanan data yang dihasilkan.

3.1. Hasil Implementasi Sistem

Hasil implementasi menunjukkan bahwa sistem mampu menjalankan proses utama, yaitu input data, enkripsi, penyimpanan *ciphertext*, dan dekripsi data teks dengan baik. Sistem dirancang agar pengguna dapat melakukan proses pengamanan data melalui antarmuka berbasis *web* secara mudah dan terintegrasi.

3.1.1 Antarmuka Login

Sebelum mengakses fitur enkripsi, pengguna diwajibkan melakukan proses login menggunakan username dan password. Halaman login berfungsi sebagai lapisan keamanan awal agar hanya pengguna tertentu yang dapat mengakses sistem enkripsi dan dekripsi data. Tampilan antarmuka login ditunjukkan pada Gambar 5.

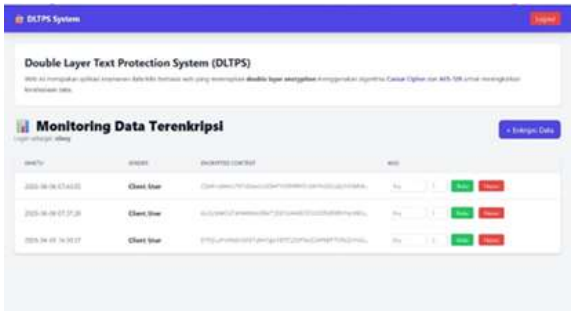


Gambar 5. Tampilan Antarmuka Login

Gambar 5 menampilkan desain halaman autentikasi log masuk (*login interface*) dari sistem yang dikembangkan dengan identitas aplikasi "Double Layer Encryption & Cipher AES-128". Antarmuka ini memuat kolom masukan akun pengguna (*username*) dan kata sandi (*password*). Perancangan visual ini berfungsi sebagai gerbang validasi otorisasi di sisi klien, memastikan bahwa hak akses ke halaman pengelolaan data terenkripsi terlindungi dari manipulasi pihak luar.

3.1.2 Tampilan Dashboard

Sistem menyediakan halaman dashboard yang menampilkan data terenkripsi yang telah tersimpan di dalam database. Dashboard memuat informasi *ciphertext*, waktu penyimpanan, dan identitas pengguna sehingga memudahkan proses pengelolaan data terenkripsi. Tampilan dashboard sistem ditunjukkan pada Gambar 6.



Gambar 6. Tampilan Dashboard Data Terenkripsi

Gambar 6 menunjukkan tampilan utama pusat pemantauan data terenkripsi (Dashboard Monitoring DLTPS System). Berdasarkan Gambar 6, antarmuka ini menyajikan tabel dinamis yang merangkum log aktivitas pengamanan data, yang terdiri dari kolom Waktu (*timestamp*), Pengirim (*Sender*), Konten Terenkripsi (*Encrypted Content*), serta kolom Aksi. Dashboard ini mempermudah administrator atau pengguna terdaftar untuk memantau muatan data rahasia yang tersimpan di server database serta menyediakan tombol pintas menuju form kunci dekripsi data.

3.1.3 Proses Input dan Enkripsi Data

Pada proses enkripsi, pengguna memasukkan *plaintext*, nilai pergeseran Caesar Cipher, dan kunci AES-128. Setelah tombol enkripsi ditekan, sistem akan melakukan proses enkripsi secara berlapis dimulai dari Caesar Cipher kemudian dilanjutkan menggunakan AES-128. Tampilan proses input dan enkripsi data ditunjukkan pada Gambar 7.

Gambar 7. Proses Input dan Enkripsi Data

Gambar 7 menampilkan formulir input enkripsi data pada sisi pengguna (*Client-Server Data Encryption*). Form ini memuat area input teks "Pesan Rahasia" (dalam contoh diisi kata "sistem"), kolom parameter "Caesar Shift" (skala pergeseran rentang nilai 0-25, diisi nilai 3), serta kolom parameter "AES Key" (wajib diisi 16 karakter, diisi "128bitkey_example"). Gambar ini mengilustrasikan kemudahan pengguna dalam mengunggah teks mentah dan mengonfigurasi dua lapis variabel parameter kunci sebelum dikirim ke sistem web.

3.1.4 Proses Dekripsi Data

Proses dekripsi dilakukan menggunakan urutan kebalikan dari proses enkripsi, yaitu dekripsi AES-128 terlebih dahulu kemudian dilanjutkan dengan Caesar Cipher untuk memperoleh *plaintext asli*. Pengguna harus memasukkan kunci yang sesuai agar data dapat dikembalikan ke bentuk semula. Tampilan proses dekripsi data ditunjukkan pada Gambar 8.

Gambar 8. Proses Dekripsi Data

Gambar 8 menampilkan halaman notifikasi sukses eksekusi pembongkaran sandi pada alamat berkas decrypt_action.php. Ketika sistem menerima kombinasi parameter kunci AES dan pergeseran Caesar yang valid, sistem akan menampilkan pesan konfirmasi hijau "Dekripsi Berhasil!" dan mencetak kembali isi "Pesan Asli: sistem". Gambar ini membuktikan keandalan logika algoritma yang diimplementasikan dalam mengembalikan berkas ciphertext menjadi plaintext secara utuh.

3.2. Hasil Pengujian Sistem

Tabel 1. Skenario Pengujian Sistem

No	Skenario Pengujian	Input
1	Enkripsi dengan kunci yang benar	Plaintext, Caesar Shift, AES Key valid
2	Dekripsi dengan kunci yang benar	Ciphertext, Caesar Shift, AES Key valid
3	Dekripsi dengan AES Key salah	Ciphertext, Caesar Shift benar, AES Key salah
4	Dekripsi dengan Caesar Shift salah	Ciphertext, AES Key benar, Caesar Shift salah

Tabel 1 menunjukkan skenario pengujian yang digunakan untuk mengevaluasi fungsionalitas sistem enkripsi dan dekripsi. Pengujian dilakukan menggunakan kombinasi kunci yang benar dan salah untuk memastikan sistem mampu menjaga keamanan data. Berdasarkan Tabel 1, keberhasilan sistem ditentukan oleh kemampuan menghasilkan ciphertext saat proses enkripsi dan mengembalikan plaintext secara utuh saat proses dekripsi menggunakan parameter yang sesuai.

Pengujian pertama dilakukan menggunakan kunci AES yang tidak sesuai. Hasil pengujian menunjukkan bahwa *ciphertext* tidak dapat dikembalikan menjadi *plaintext asli* sehingga data tetap aman dari akses yang tidak sah. Hasil pengujian kesalahan kunci AES ditunjukkan pada Gambar 9.

Gambar 9. Pengujian Kesalahan Kunci AES

Gambar 9 memperlihatkan penanganan galat (*error handling*) oleh sistem melalui kotak dialog peringatan berbasis JavaScript (*alert browser*) yang memunculkan notifikasi "Gagal! Key atau Shift salah.". Fenomena pada Gambar 9 ini terjadi akibat dimasukkannya parameter kunci AES yang tidak cocok pada saat request dekripsi dijalankan di server proyek_skj.test, sehingga memblokir akses terhadap data teks.

Pengujian berikutnya dilakukan menggunakan nilai pergeseran Caesar Cipher yang salah. Hasil dekripsi menghasilkan karakter acak sehingga *plaintext asli* tidak dapat diperoleh secara benar. Hasil pengujian kesalahan Caesar Cipher ditunjukkan pada Gambar 10.

Gambar 10. Pengujian Kesalahan Caesar Cipher

Gambar 10 memvisualisasikan kondisi pengujian kegagalan pada lapisan pertama, di mana kunci AES yang

dimasukkan sudah benar namun nilai pergeseran Caesar Cipher salah. Sistem mengindikasikan status sukses "Dekripsi Berhasil!" karena proses dekripsi AES berhasil dilakukan, namun luaran teks yang dihasilkan berupa deretan huruf acak "Pesan Asli: rhrsdl". Gambar ini membuktikan efektivitas skema *double layer*, di mana kesalahan pada salah satu variabel kunci tetap mampu melindungi kerahasiaan isi pesan asli.

Tabel 2. Hasil Pengujian Sistem

No	Skenario	Hasil aktual
1	Enkripsi data dengan kunci valid	Ciphertext terbentuk
2	Dekripsi dengan Caesar Shift dan AES Key valid	Plaintext kembali
3	Dekripsi dengan AES Key salah	Dekripsi gagal
4	Dekripsi dengan Caesar Shift salah	Plaintext tidak sesuai

Tabel 2 memperlihatkan hasil pengujian sistem pada beberapa skenario penggunaan kunci. Berdasarkan hasil pengujian, seluruh fungsi utama sistem berjalan sesuai dengan rancangan. Proses enkripsi dan dekripsi berhasil dilakukan ketika menggunakan kombinasi kunci yang benar, sedangkan penggunaan AES Key atau nilai Caesar Shift yang tidak sesuai menyebabkan plaintext tidak dapat dipulihkan secara benar. Hasil tersebut menunjukkan bahwa mekanisme *double layer encryption* yang diterapkan mampu memberikan perlindungan tambahan terhadap akses data oleh pihak yang tidak berwenang.

3.3. Pembahasan dan Analisis

Sistem menerapkan dua lapisan enkripsi yang memiliki karakteristik berbeda. Caesar Cipher digunakan sebagai algoritma klasik untuk melakukan transformasi awal berbasis pergeseran karakter, sedangkan AES-128 digunakan sebagai algoritma modern untuk meningkatkan kompleksitas *ciphertext* [11], [12].

Berdasarkan hasil pengujian pada Tabel 2, proses dekripsi hanya berhasil apabila nilai Caesar Shift dan AES Key yang digunakan sesuai dengan parameter saat proses enkripsi. Ketidaksesuaian salah satu parameter menyebabkan data tidak dapat dikembalikan ke bentuk aslinya. Hal ini menunjukkan bahwa penerapan *double layer encryption* memberikan tingkat keamanan yang lebih baik dibandingkan penggunaan Caesar Cipher secara tunggal karena memerlukan dua lapisan validasi kunci yang berbeda.

Pengujian dilakukan untuk memastikan sistem mampu menjalankan proses enkripsi dan dekripsi dengan benar berdasarkan kombinasi kunci yang diberikan pengguna. Pengujian juga dilakukan untuk mengetahui pengaruh kesalahan kunci terhadap hasil dekripsi data.

Secara matematis, proses enkripsi Caesar Cipher menggunakan persamaan berikut:

$$C_i = 32 + ((P_i - 32 + k) mod\ 94)$$

Pada persamaan tersebut, C_i menyatakan ciphertext indeks ke-i, P_i menyatakan plaintext indeks ke-i, sedangkan k merupakan nilai pergeseran karakter.

Contoh proses enkripsi menggunakan *plaintext* "SISTEM" dengan nilai pergeseran k =3 menghasilkan transformasi karakter sebagai berikut:

Karakter S memiliki nilai ASCII 83 sehingga:

$C_i = 32 + ((83 - 32 + 3) mod\ 94) = 86 \rightarrow V$

Karakter I memiliki nilai ASCII 73:

$C_i = 32 + ((73 - 32 + 3) mod\ 94) = 76 \rightarrow L$

… (dst.)

Hasil akhir proses Caesar Cipher menghasilkan *ciphertext*: **SISTEM → VLVWHP**

Ciphertext tersebut kemudian diproses kembali menggunakan AES-128 melalui tahapan *SubBytes*, *ShiftRows*, *MixColumns*, dan *AddRoundKey* untuk menghasilkan *ciphertext* akhir yang lebih kompleks dan sulit dianalisis [7], [8].

Berdasarkan hasil implementasi dan pengujian, metode *double layer encryption* mampu meningkatkan keamanan data teks karena proses dekripsi hanya dapat dilakukan apabila kedua kunci dimasukkan secara benar. Kombinasi Caesar Cipher dan AES-128 juga menghasilkan *ciphertext* yang lebih sulit diprediksi dibandingkan penggunaan satu algoritma saja.

4. Kesimpulan

Berdasarkan hasil penelitian yang telah dilakukan, sistem pengamanan data teks berbasis *web* menggunakan metode *double layer encryption* berhasil diimplementasikan dengan menggabungkan algoritma Caesar Cipher dan AES-128. Sistem mampu menjalankan proses enkripsi dan dekripsi data teks dengan baik melalui dua tahapan pengamanan secara berurutan. Hasil pengujian menunjukkan bahwa *plaintext* hanya dapat dikembalikan ke bentuk semula apabila kedua kunci dimasukkan dengan benar. Penggunaan Caesar Cipher

sebagai lapisan awal dan AES-128 sebagai lapisan kedua berhasil meningkatkan kompleksitas *ciphertext* sehingga data menjadi lebih sulit dianalisis dibandingkan penggunaan satu algoritma enkripsi saja. Selain itu, implementasi berbasis *web* memudahkan proses pengelolaan data terenkripsi melalui antarmuka yang sederhana dan terintegrasi dengan database.

Berdasarkan hasil analisis, metode *double layer encryption* dapat menjadi alternatif dalam meningkatkan keamanan data teks pada aplikasi web. Penelitian selanjutnya dapat dikembangkan dengan menambahkan algoritma kriptografi lainnya, menerapkan sistem keamanan pada file multimedia, atau mengintegrasikan teknologi keamanan tambahan seperti hashing dan autentikasi multi-faktor untuk meningkatkan perlindungan data digital.

Referensi

[1] M. Hidayat, M. Tahir, A. Sukriyadi, A. Sulton, and A. History, "Penerapan kriptografi caesar chiper dalam pengamanan data," vol. 2, no. 3, pp. 35–41, 2023, doi: https://doi.org/10.56127/jukim.v2i03.619.

[2] G. Ramasamy, M. Gurupriya, G. L. Sridhar, U. Aditya, and N. Shreyas, "Enhancing Database Security through Multi - Layered Cryptographic Techniques," vol. 3, pp. 532–540, 2025, doi: 10.5220/0013901400004919.

[3] M. H. Alfirdaus, M. Tahir, and N. E. Dewanti, "Perancangan Aplikasi Enkripsi Deskripsi Mengunakan Metode Caesar Chiper Berbasis Web," vol. 2, no. 2, 2023, doi: https://doi.org/10.55606/jtmei.v2i2.1628.

[4] A. Hermawan, A. Halim, D. Susilawati, and I. A. Putri, "Implementasi Algoritma Advance Encryption Standard dan Caesar Cipher pada Pesan Terenkripsi," vol. 5, no. 1, 2023, doi: 10.36499/jinrpl.v5i1.6714.

[5] S. F. Nabila, Z. Vonna, and S. A. Hasibuan, "Implementasi Algoritma Caesar Cipher untuk Enkripsi dan Dekripsi Pesan," pp. 32–41, 2026, doi: https://doi.org/10.62383/polygon.v4i1.925.

[6] R. Maruli, T. Pasaribu, R. Dewi, and I. Parlina, "Implementasi Kombinasi Algoritma Rinjdael dengan Caesar Cipher pada Pengamanan Dokumen Digital," 2022, doi: https://doi.org/10.56211/helloworld.v1i1.11.

[7] M. B. Aryanto, M. Tahir, S. I. Devita, and Z. N. Mustofa, "Implementasi Enkrip Dan Dekrip File Menggunakan Metode Advance Encryption Standard (AES-128)," vol. 3, no. 1, 2023, doi: https://doi.org/10.55606/juisik.v3i1.434.

[8] P. D. Dwiyansah, F. Fathurrohman, N. Alfikry, and J. Sunupurwa, "A Comparative Analysis of the Advanced Encryption Standard (AES) 128- , 192- , and 256-Bit Algorithms in Digital Data Security," vol. 6, no. 02, pp. 176–182, 2026, doi: 10.58471/jms.v6i02.

[9] I. Asih, R. Simbolon, I. Gunawan, I. O. Kirana, R. Dewi, and S. Solikhun, "Penerapan Algoritma AES 128-Bit dalam Pengamanan Data Kependudukan pada Dinas Dukcapil Kota Pematangsiantar," vol. 1, no. 2, pp. 54–60, 2020, doi: 10.47065/josyc.v1i2.112.

[10] M. Azhari, D. I. Mulyana, F. J. Perwitosari, and F. Ali, "Jurnal Pendidikan Sains dan Komputer Implementasi Pengamanan Data pada Dokumen Menggunakan Algoritma Kriptografi Advanced Encryption Standard (AES) Jurnal Pendidikan Sains dan Komputer," vol. 2, no. 1, pp. 163–171, 2022, doi: https://doi.org/10.47709/jpsk.v2i01.1390.

[11] F. Nuraeni, Y. H. Agustin, and A. E. Purnama, "IMPLEMENTASI CAESAR CIPHER AND ADVANCED ENCRYPTION STANDARD (AES) PADA PENGAMANAN DATA," vol. 22, no. 2, pp. 187–194, 2020, doi: https://doi.org/10.33557/jurnalmatrik.v22i2.949.

[12] F. Nuraeni, M. F. Amrulloh, A. Mulyani, and D. Kurniadi, "Implementasi Superenkripsi DSA dan Aes 128 Bit Dalam Pengamanan File Surat Digital," pp. 601–613, 2025, doi: 10.33364/algoritma/v.22-1.1832.

[13] M. N. Alatawi, "A Hybrid Cryptographic Cipher Solution for Secure Communication in Smart Cities," vol. 10, no. 5, pp. 776–791, 2023, doi: 10.22247/ijcna/2023/223423.

[14] A. H. Nasrullah, "Secure Web-Based File Encryption Using AES-128," vol. 6, no. 2, pp. 146–155, 2025, doi: https://doi.org/10.59562/jessi.v6i2.8436.

[15] J. P. Azanuddin, Suardi Yakub, "Implementasi Keamanan Citra Menggunakan Algoritma AES-128 Dengan Aplikasi Client-Server," vol. 7, pp. 51–61, 2022, doi: http://dx.doi.org/10.30645/jurasik.v7i1.415.