KOMIK (Konferensi Nasional Teknologi Informasi dan Komputer)
Volume 9, Nomor 1, November 2025
DOI: 10.30865/komik.v9i1.9565

ISSN 2597-4645 (media online)
ISSN 2597-4610 (media cetak)
Page: 11-18

Analisis Perbandingan Algoritma Advanced Encryption Standard (AES) dan Twofish Dalam Mengamankan Dokumen Teks

Aditya Prayoga Abdillah

Fakultas Ilmu Komputer dan Teknologi Informasi, Teknik Informatika, Universitas Budi Darma, Medan, Indonesia
Email: yogapoki64@gmail.com
*) yogapoki64@gmail.com

ABSTRAK- Perkembangan teknologi informasi yang pesat menuntut adanya sistem keamanan andal untuk melindungi dokumen teks yang berisi informasi sensitif. Penelitian ini membandingkan efisiensi dan tingkat keamanan dua algoritma kriptografi simetris terkemuka, yaitu Advanced Encryption Standard (AES) dan Twofish, dalam mengamankan file teks. Implementasi meliputi proses enkripsi file teks menjadi *cipherteks* dan dekripsi kembali menjadi *plaintext* menggunakan kunci tertentu. Pengujian kinerja difokuskan pada analisis kecepatan enkripsi dan dekripsi, ukuran file *cipherteks*, serta kualitas keamanan menggunakan parameter *Peak Signal-to-Noise Ratio* (PSNR). Hasil penelitian menunjukkan bahwa AES cenderung unggul dalam kecepatan proses, sementara Twofish menunjukkan konsistensi yang lebih baik dalam menjaga kualitas data selama pengujian PSNR. Dengan demikian, penelitian ini memberikan pemahaman yang jelas mengenai kelebihan dan kekurangan masing-masing algoritma kriptografi sebagai acuan penting dalam pemilihan metode enkripsi yang tepat untuk pengamanan data digital.

Kata Kunci: Kriptografi, AES, Twofish, Enkripsi, Keamanan Dokumen Teks, Perbandingan Algoritma, Kecepatan, PSNR, Kualitas Data.

ABSTRCT- The rapid development of information technology demands a reliable security system to protect text documents containing sensitive information. This study compares the efficiency and security level of two leading symmetric cryptographic algorithms, namely Advanced Encryption Standard (AES) and Twofish, in securing text files. The implementation includes the process of encrypting text files into ciphertext and decrypting them back into plaintext using a specific key. Performance testing focuses on analyzing encryption and decryption speed, ciphertext file size, and security quality using the Peak Signal-to-Noise Ratio (PSNR) parameter. The results show that AES tends to be superior in processing speed, while Twofish shows better consistency in maintaining data quality during PSNR testing. Thus, this study provides a clear understanding of the advantages and disadvantages of each cryptographic algorithm as an important reference in selecting the right encryption method for digital data security.

Keywords: AES, Twofish, Encryption, Text Document Security, Algorithm Comparison, Speed, PSNR, Data Quality.

1. PENDAHULUAN

Kemajuan pesat dalam teknologi informasi dan komunikasi telah menciptakan kemudahan yang signifikan dalam pengaksesan dan pertukaran data melalui berbagai media komunikasi. Namun, kemudahan ini secara langsung memberikan dampak dan tantangan serius terhadap keamanan informasi atau pesan yang ditransmisikan maupun disimpan. Dalam lingkungan yang terbuka, file digital, terutama yang memuat informasi sensitif, rentan terhadap berbagai ancaman seperti pengintaian, manipulasi, atau pencurian oleh pihak yang tidak bertanggung jawab[1]. Untuk meminimalisasi risiko tersebut dan menjaga kerahasiaan (*confidentiality*) serta integritas data, diperlukan suatu metode pengamanan data yang andal. Salah satu solusi fundamental dan paling efektif adalah penggunaan kriptografi [2].

Kriptografi adalah ilmu dan seni dalam menyandikan data, yaitu mengubah data asli (*plaintext*) menjadi bentuk tersandi (*ciphertext*) yang tidak dapat dibaca, menggunakan algoritma dan kunci tertentu. Hanya pihak yang memiliki kunci dekripsi yang sah yang dapat mengembalikan *ciphertext* ke *plaintext* semula. Dalam klasifikasi kriptografi modern, algoritma kunci simetri (*symmetric key*) merupakan pilihan populer karena kecepatan pemrosesan datanya yang tinggi, menjadikannya ideal untuk enkripsi data dalam jumlah besar [3].

Dua algoritma kriptografi modern berbasis *cipher block* dengan kunci simetri yang menjadi perhatian utama adalah Algoritma Rijndael dan Algoritma Twofish. Algoritma Rijndael, yang kemudian ditetapkan sebagai Advanced Encryption Standard (AES) oleh NIST, menggunakan kombinasi proses substitusi dan permutasi melalui sejumlah putaran (*rounds*) yang dikenakan pada tiap blok data yang dienkripsi atau didekripsi. Setiap putaran menggunakan sub-kunci yang berbeda. Algoritma Rijndael dirancang untuk mendukung panjang kunci 128, 192, dan 256 *bit* [4].

Sementara itu, Algoritma Twofish beroperasi pada blok *plaintext* berukuran 128 *bit*. Twofish menggunakan struktur sejenis Feistel dalam 16 putaran dengan mengadopsi teknik *whitening* baik pada *input* maupun *output* proses enkripsi. Selain itu, Twofish menggunakan fungsi-fungsi kompleks untuk membangkitkan sub-kunci, sehingga meningkatkan keamanannya. Algoritma ini juga mendukung panjang kunci yang fleksibel, yaitu 128, 192, dan 256 *bit* [5].

Mengingat perbedaan mendasar pada arsitektur internal, seperti struktur Rijndael yang bukan merupakan jaringan Feistel murni dan struktur Twofish yang berbasis Feistel, serta perbedaan pada mekanisme penyandian dan pembuatan kunci, penting untuk membandingkan performa kedua algoritma ini ketika diterapkan pada jenis data yang sama, yaitu dokumen teks. Perbandingan ini akan memberikan wawasan krusial mengenai efisiensi komputasi,

kecepatan, dan ketahanan keamanan yang dimiliki oleh masing-masing algoritma, yang pada akhirnya dapat menjadi acuan dalam pemilihan algoritma enkripsi yang paling optimal untuk pengamanan file teks [6].

Batasan masalah dalam penelitian ini ditetapkan agar fokus penelitian tetap terarah, yaitu hanya menggunakan data berbentuk file berekstensi .txt dan bukan karakter yang diinput secara manual, dengan algoritma kriptografi yang digunakan terbatas pada Algoritma Rijndael dan Algoritma Twofish dalam proses enkripsi dan dekripsi file. Penelitian ini juga tidak mencakup pembahasan mengenai enkripsi dan dekripsi pada komunikasi client-server atau transmisi data, serta seluruh pengujian yang dilakukan menggunakan panjang kunci tetap, yaitu 128 bit.

2. METODE PENELITIAN

Metodologi penelitian ini dirancang untuk menganalisis dan membandingkan kinerja algoritma AES dan Twofish secara objektif dalam konteks pengamanan file teks. Penelitian ini menggunakan pendekatan kuantitatif komparatif, di mana dua variabel utama (AES dan Twofish) diuji pada *input* yang identik dan hasilnya dibandingkan berdasarkan serangkaian metrik kinerja dan keamanan.

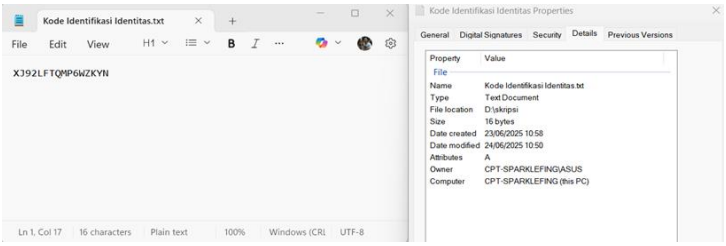
2.1 Kerangka Kerja Penelitian

Kerangka kerja penelitian ini dimulai dengan mengumpulkan data berupa file teks (*.txt*) sebagai *plaintext*. Selanjutnya, dilakukan proses implementasi dan pengujian. *Plaintext* dienkripsi menggunakan algoritma AES dan Twofish, keduanya menggunakan kunci 128-*bit* yang sama. Hasil enkripsi berupa *ciphertext* kemudian dicatat waktu pemrosesannya dan dianalisis ukurannya. Setelah itu, *ciphertext* didekripsi kembali untuk memverifikasi kebenaran dan integritas data [7]. Hasil dari tahapan ini dibandingkan untuk mengevaluasi algoritma mana yang lebih unggul berdasarkan parameter yang telah ditetapkan. Implementasi algoritma dalam penelitian ini dibangun menggunakan bahasa pemrograman Python [8].

2.2 Samplel Data

Sampel data yang digunakan dalam penelitian ini berupa tiga jenis file teks (*.txt*) yang memuat berbagai informasi untuk mewakili skenario penggunaan nyata. Ketiga sampel file teks tersebut adalah:

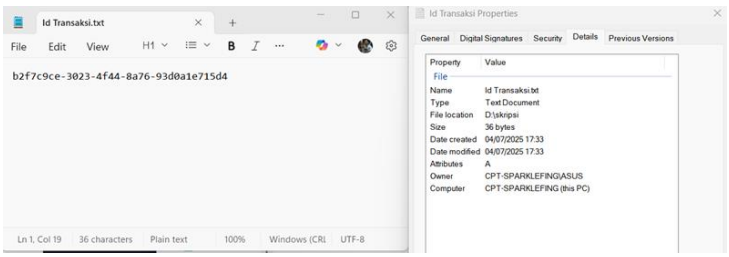
1. File Teks Kode Identitas Pribadi



Gambar 1. File Teks Kode Identitas Pribadi

Plaintext (teks asli): XJ92LFTQMP6WZKYN Ini adalah data yang ingin diamankan. Panjangnya 16 karakter = 16 byte (128 bit).

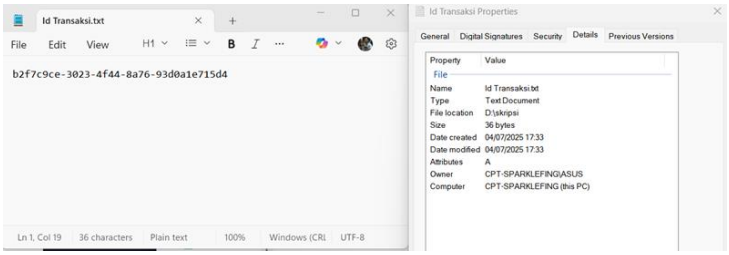
2. File Teks ID Transaksi



Gambar 2. File Teks ID Transaksi

Plaintext (teks asli): b2f7c9ce-3023-4f44-8a76-93d0a1e715d4 Ini adalah data yang ingin diamankan. Panjangnya 36 karakter = 36 byte (288 bit).

3. File Teks *Payload Login Token*



Gambar 3. File Teks *Payload Login Token*

Plaintext (teks asli): "alg": "HS256","typ": "JWT","payload": sub": "user_00123","name": "PW Dev","iat": 1720101000,"exp": 1720104600 Ini adalah data yang ingin diamankan. Panjangnya 110 karakter = 110 byte (880 bit).

Tujuan penggunaan sampel data yang beragam ini adalah untuk menguji performa kedua algoritma terhadap *input* dengan ukuran dan struktur konten yang berbeda, memastikan hasil perbandingan yang lebih beragam dan objektif. Seluruh pengujian dilakukan dengan panjang kunci 128-*bit* yang disamakan untuk kedua algoritma[9].

2.3 Indikator Pengujian

Untuk membandingkan performa AES dan Twofish, digunakan lima parameter pengujian utama:

1. Kecepatan Enkripsi dan Dekripsi: Mengukur waktu komputasi yang dibutuhkan algoritma (dalam detik) untuk menyelesaikan proses enkripsi (*plaintext* ke *ciphertext*) dan dekripsi (*ciphertext* ke *plaintext*). Kecepatan sangat memengaruhi efisiensi sistem.
2. Ukuran File (*Ciphertext*): Mengamati perubahan ukuran file hasil enkripsi. Meskipun secara teori *cipher block* tidak mengubah ukuran, pengukuran ini penting untuk analisis efisiensi penyimpanan.
3. *Peak Signal-to-Noise Ratio* (PSNR): Digunakan sebagai metrik kualitatif keamanan untuk mengukur tingkat perbedaan atau *noise* yang dihasilkan antara *plaintext* asli dan *ciphertext*. Nilai PSNR yang lebih rendah umumnya menunjukkan perubahan data yang lebih besar dan *randomness* yang lebih tinggi, mengindikasikan keamanan yang lebih kuat [10].
4. *Bit Error Rate* (BER): Mengukur tingkat kesalahan bit yang terjadi antara *plaintext* dan *ciphertext*. BER yang rendah menunjukkan algoritma lebih stabil dan memiliki tingkat kesalahan yang lebih rendah dalam mengamankan data.
5. *Payload Capacity*: Mengukur kecepatan pemrosesan kapasitas *payload* (data yang dienkripsi) dalam satuan *size/s* untuk menilai efisiensi algoritma dalam menangani volume data.
6. *Brute Force Attack*: Pengujian sederhana untuk mengukur waktu yang dibutuhkan kedua algoritma untuk menemukan kunci yang benar pada percobaan ke-1234, sebagai indikator komparatif terhadap kerentanan serangan.

3. HASIL DAN PEMBAHASAN

Analisis dan penerapan metode dilakukan terhadap implementasi AES dan Twofish pada tiga sampel file teks. *Output* dari penelitian ini berupa data perbandingan performa, yang kemudian dianalisis berdasarkan metrik pengujian yang telah ditetapkan [11].

3.1 Pengujian dan Perbandingan Algortima AES dan *Twofish*

a. Gunalan huruf kecil dan abjed untuk penomoran list.
b. Seting 5 mm untuk bagian kiri menjorok kedalam.
c. Jika lebih dari 1 level penomoran gunakan penomoran angka untuk list selanjutnya:
    a) Gunakan penomoran angka.
    b) Selanjutnya

3.1.1 Pengujian Kecepatan Enkripsi dan Dekripsi

Pengujian kecepatan menunjukkan performa yang sangat cepat untuk kedua algoritma dengan perbedaan waktu yang relatif kecil.

Tabel 1. Kecepatan Enkripsi dan Dekripsi

Algoritma	File Teks	Waktu Enkripsi (detik)	Waktu Dekripsi (detik)
AES	Sampel 1	0,004791	0,002012
AES	Sampel 2	0,001755	0,001815
AES	Sampel 3	0,001612	0,001770
Twofish	Sampel 1	0,001493	0,001505
Twofish	Sampel 2	0,002191	0,001670
Twofish	Sampel 3	0,001517	0,001614

Secara umum, performa keduanya berimbang dengan perbedaan yang sangat tipis. Twofish sedikit lebih unggul pada beberapa kasus enkripsi (terlihat pada Sampel 1 dan 3), sementara AES menunjukkan waktu yang lebih stabil dan konsisten. Ini mengindikasikan bahwa kedua algoritma sama-sama efisien untuk pengamanan dokumen teks dalam skala kecil hingga menengah.

3.1.2 Pengujian *Payload Capacity*

Pengujian Payload capacity mengukur efisiensi algoritma dalam memproses volume data (ukuran/size per detik).

Tabel 2. *Payload Capacity*

Algoritma	File Teks	Proses Payload	Kecepatan (Size/s)
AES	Sampel 1	100%	400,76
AES	Sampel 2	100%	400,76
AES	Sampel 3	100%	400,76
Twofish	Sampel 1	100%	4.451,61
Twofish	Sampel 2	100%	4.451,61
Twofish	Sampel 3	100%	4.451,61

Hasil pengujian menunjukkan Twofish sangat unggul dalam kecepatan pemrosesan *payload*. Twofish rata-rata 4.451,61 *size/s*, sekitar sebelas kali lebih cepat dibandingkan AES yang hanya 400,76 *size/s*. Keunggulan signifikan ini menunjukkan bahwa Twofish jauh lebih efisien dan optimal dalam menangani kapasitas *payload* besar dibandingkan AES.

3.1.3 Pengujian Bit Error Rate (BER)

BER mengukur tingkat *error* bit antara *plaintext* dan *ciphertext*, di mana nilai yang lebih rendah menunjukkan stabilitas dan tingkat kesalahan yang lebih rendah.

Tabel 3. Bit Error Rate (BER)

Algoritma	File Teks	Rata–rata BER	BER Maksimum
AES	Sampel 1	22,89%	59,38%
AES	Sampel 2	15,30%	25,35%
AES	Sampel 3	6,19%	8,52%
Twofish	Sampel 1	14,30%	48,44%
Twofish	Sampel 2	5,95%	17,36%
Twofish	Sampel 3	2,66%	6,93%

Berdasarkan hasil pengujian BER, algoritma Twofish cenderung lebih baik dibandingkan AES. Twofish menghasilkan nilai rata-rata BER dan BER maksimum yang lebih rendah di semua sampel. Hal ini menyimpulkan bahwa Twofish lebih stabil dan memiliki tingkat kesalahan yang lebih rendah dibandingkan AES dalam mengamankan dokumen teks.

3.1.4 Pengujian Peak Signal-to-Noise Ratio (PSNR)

PSNR digunakan untuk menilai kualitas keamanan kualitatif dengan mengukur perubahan bit data. PSNR umumnya diterapkan pada citra, tetapi dalam konteks ini digunakan untuk menilai sejauh mana *noise* yang dihasilkan oleh algoritma. Nilai yang lebih tinggi menunjukkan kualitas data yang lebih baik (noise lebih sedikit), sedangkan nilai yang lebih rendah menunjukkan data yang lebih terdistorsi (*random*).

Tabel 4. Peak Signal-to-Noise Ratio (PSNR

Algoritma	File Teks	Desibel (dB)	Keterangan
AES	Sampel 1	NaN	AES error
AES	Sampel 2	Nan	AES error
AES	Sampel 3	28.6463	AES sedikit lebih baik
Twofish	Sampel 1	≈ 28.00	Twofish Stabil
Twofish	Sampel 2	NaN	Twofish error
Twofish	Sampel 3	28.0448	Twofish sedikit kecil

Pada pengujian kualitas data (menggunakan sampel data 128 *byte*), AES sedikit lebih baik dengan nilai PSNR 28,6463 *dB* dibandingkan Twofish sebesar 28,0448 *dB*. Perbedaan ini sangat tipis (*marginal*), sehingga secara praktis keduanya masih sama-sama stabil. Namun, munculnya nilai *NaN* (Not a Number) pada beberapa ukuran data menunjukkan adanya *error* perhitungan pada metode pengujian yang perlu diperbaiki pada studi selanjutnya.

3.1.5 Pengujian *Brute Force Attack*

Pengujian ini mengukur waktu yang dibutuhkan untuk menemukan kunci pada percobaan ke-1234.

Tabel 5. *Brute Force Attack*

Algoritma	File Teks	Percobaan Ke-	Waktu Diperlukan (detik)
AES	Sampel 1	1234	0,0615
AES	Sampel 2	1234	0,0235
AES	Sampel 3	1234	0,0255
Twofish	Sampel 1	1234	0,0175
Twofish	Sampel 2	1234	0,0089
Twofish	Sampel 3	100%	4.451,61

Dalam konteks pengujian *brute force*, Twofish relatif lebih cepat dalam proses pencarian kunci dibandingkan dengan AES. Hal ini menunjukkan efisiensi Twofish dalam operasi *key expansion* dan putaran komputasi. Namun, perlu dicatat bahwa pengujian ini dilakukan pada iterasi kunci yang rendah (ke-1234), dan kedua algoritma pada kunci 128-*bit* secara teoritis masih sangat aman terhadap serangan *brute force* standar.

Berdasarkan rangkaian pengujian, AES dan Twofish memiliki keunggulan yang saling melengkapi.

a. Keunggulan Twofish: Twofish menunjukkan keunggulan signifikan dalam menangani kapasitas payload, menjadi lebih dari 11 kali lebih cepat dari AES. Selain itu, Twofish juga lebih unggul dalam stabilitas data, ditunjukkan dengan nilai BER yang lebih rendah dan konsisten.
b. Keunggulan AES: AES menunjukkan keunggulan pada konsistensi waktu enkripsi-dekripsi, serta sedikit lebih unggul (walaupun tipis) dalam menjaga kualitas data (PSNR) pada sampel tertentu.

Secara keseluruhan, Twofish unggul dalam kinerja berorientasi throughput (Payload Capacity) dan stabilitas (BER), sementara AES unggul dalam konsistensi waktu proses. Keduanya terbukti efektif dan layak digunakan untuk mengamankan dokumen teks, dengan pemilihan algoritma yang disesuaikan dengan prioritas kebutuhan pengguna: kecepatan pemrosesan data besar (throughput) atau konsistensi waktu proses .

3.2 Implementasi

Tahap implementasi dilakukan untuk merealisasikan rancangan sistem pengamanan dokumen teks menggunakan algoritma AES dan Twofish. Proses ini mencakup pembuatan program berbasis Python dengan antarmuka GUI (Graphical User Interface) yang memungkinkan pengguna untuk melakukan enkripsi dan dekripsi dokumen teks secara mudah [12].

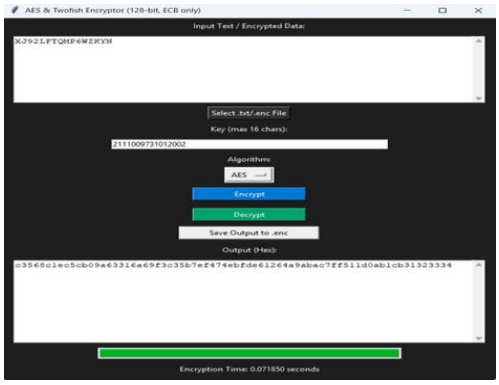
Implementasi diawali dengan pembuatan modul enkripsi dan dekripsi untuk masing-masing algoritma. Modul AES menggunakan pustaka PyCryptodome, sedangkan modul Twofish diimplementasikan dengan pustaka twofish yang telah dimodifikasi untuk menerima kunci dan teks dalam format UTF-8.

Antarmuka GUI dibangun menggunakan pustaka Tkinter, yang menyediakan tombol untuk:

a. Memilih file teks yang akan diamankan.
b. Memasukkan kunci enkripsi/dekripsi.
c. Memilih algoritma yang akan digunakan (AES atau Twofish).
d. Menampilkan hasil proses enkripsi dalam bentuk ciphertext dan menyimpannya ke file.
e. Menampilkan hasil dekripsi untuk mengembalikan file ke bentuk aslinya.

Tahap implementasi ini juga melibatkan pengujian setiap fungsi, mulai dari pemilihan file, pembacaan data, proses enkripsi, penyimpanan ciphertext, hingga proses dekripsi untuk memastikan bahwa data yang dihasilkan sesuai dengan plaintext awal. Seluruh kode program diintegrasikan sehingga pengguna dapat melakukan perbandingan antara hasil dan waktu proses dari kedua algoritma secara langsung melalui aplikasi.

Untuk menjalankan program pertama pastikan dulu file python terbuka, lalu tekan Ctrl + F5 -> program akan langsung jalan.



Gambar 4. Proses Enkripsi AES

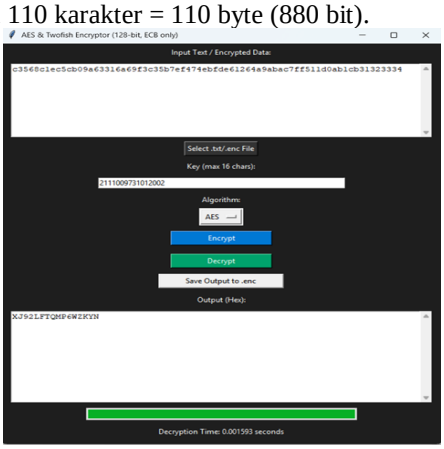
a. Input
   1. Plaintext (teks asli): XJ92LFTQMP6WZKYN Ini adalah data yang ingin diamankan. Panjangnya 16 karakter = 16 byte (128 bit), pas dengan ukuran blok AES-128.
   2. Kunci (Key): 2111009731012002 Key ini juga 16 karakter (16 byte = 128 bit), sehingga cocok untuk AES-128.

b. Pemilihan Algoritma
   Pada bagian Algorithm, user memilih AES. Mode operasi yang digunakan adalah ECB (Electronic Codebook), sesuai judul aplikasi ("AES & Twofish Encryptor (128-bit, ECB only)"). ECB membagi plaintext menjadi blok-blok 128-bit, lalu setiap blok dienkripsi secara independen.

c. Proses Enkripsi AES
   Langkah internal AES-128 (dengan key 128-bit) terdiri dari 10 ronde transformasi:
   1. AddRoundKey (Ronde Awal) Plaintext di-XOR dengan kunci utama.
   2. Ronde 1 – 9 SubBytes: Setiap byte plaintext diganti dengan nilai lain sesuai tabel substitusi (S-box), ShiftRows: Baris-baris matriks state digeser dengan jumlah tertentu. MixColumns, Setiap kolom di campur menggunakan operasi matriks. AddRoundKey, Hasil state di-XOR dengan kunci ronde.
   3. Ronde 10 (Final Round) Sama seperti ronde biasa, tetapi tanpa MixColumns. Hasil akhir dari ronde ke-10 inilah yang menjadi ciphertext.

d. Output
   Ciphertext(Hex): "c3568c1ec5cb09a63316a69f3c35b7ef474ebfde61264a9abac 7ff511d0ab1cb31323334". Ini adalah hasil enkripsi AES dalam bentuk heksadesimal.
   Panjangnya 32 byte (64 digit hex) karena ada kemungkinan padding tambahan (plaintext 16 byte + padding sesuai skema yang digunakan di library PyCryptodome). Dari output terlihat ada bagian 31323334 di akhir, yang merupakan ASCII untuk "1234" ini kemungkinan hasil padding.

e. Waktu Enkripsi Encryption Time: 0.071850 seconds → Ini adalah waktu komputasi yang dibutuhkan program untuk menyelesaikan enkripsi plaintext dengan AES-128 ECB.Plaintext (teks asli): "alg": "HS256","typ": "JWT","payload": sub": "user_00123","name": "PW Dev","iat": 1720101000,"exp": 1720104600 Ini adalah data yang ingin diamankan. Panjangnya 110 karakter = 110 byte (880 bit).



Gambar 5. Proses Dekripsi AES

a. Input Chiperteks
   Di bagian atas kotak teks terlihat data terenkripsi (ciphertext) dalam format heksadesimal: "c3568clec5cb09a63316a69f3c35b7ef474ebfde61264a9abac7ff 511d0ab1cb31323334", Ini adalah hasil enkripsi dari plaintext sebelumnya menggunakan algoritma AES dengan mode ECB.

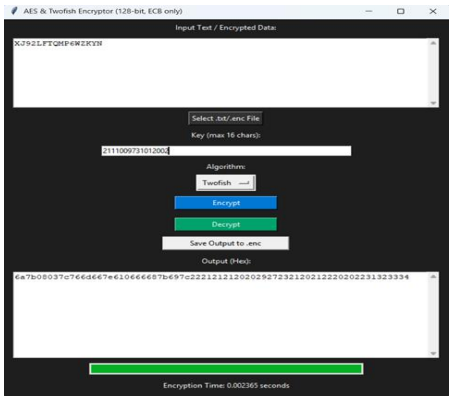
b. Kunci Enkripsi/Dekripsi Key yang digunakan: "2111009731012002" Key ini berjumlah 16 karakter (128-bit), sesuai dengan AES-128. Dalam AES, kunci yang sama digunakan baik untuk proses enkripsi maupun dekripsi (algoritma simetris).

c. Algoritma yang Dipilih Dari dropdown Algorithm, dipilih AES. Sehingga program akan menjalankan AES decryption sesuai input ciphertext dan kunci.

d. Proses Dekripsi Ciphertext hex diubah dulu ke dalam bentuk biner (byte array). AES menjalankan Inverse Cipher: AddRoundKey (dibalik), Inverse ShiftRows, Inverse SubBytes, Inverse MixColumns (kecuali ronde terakhir), Proses ini dilakukan untuk 10 ronde (karena AES-128). Output hasil dekripsi berupa plaintext asli.

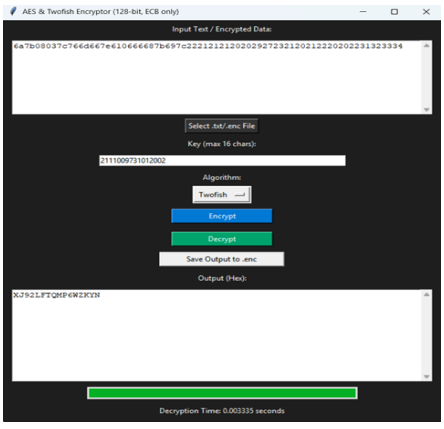
e. Output Hasil dekripsi ditampilkan pada bagian Output (Hex): "XJ92LF TQMP6WZKYN" Ini adalah plaintext asli sebelum dilakukan enkripsi.

f. Waktu Dekripsi Program juga menampilkan lama waktu proses dekripsi: Decryption Time: 0.001593 seconds, Artinya proses dekripsi berlangsung sangat cepat (sekitar 1,5 milidetik).



Gambar 6. proses Enkripsi Twofish

a. Input Plainteks
Teks asli yang akan dienkripsi adalah: “XJ92LFTQMP6WZKYN” Panjangnya 16 karakter (128 bit), sesuai dengan block size Twofish.
b. Kunci yang digunakan: 2111009731012002 Panjang kunci ini 16 karakter (128 bit). Twofish mendukung kunci 128, 192, dan 256 bit jadi ini termasuk kunci 128 bit. Kunci Enkripsi/Dekripsi Key yang digunakan: “2111009731012002” Key ini berjumlah 16 karakter (128-bit), sesuai dengan AES-128. Dalam AES, kunci yang sama digunakan baik untuk proses enkripsi maupun dekripsi (algoritma simetris).
c. Pemilihan Algoritma Dari pilihan algoritma, Anda memilih Twofish (mode ECB, 128-bit). ECB (Electronic Codebook) artinya setiap blok plaintext sepanjang 128 bit dienkripsi langsung dengan kunci, tanpa chaining antar blok.
d. Proses Enkripsi Internal Twofish Secara garis besar, Twofish melakukan langkah-langkah berikut:
    1. Key Whitening (Pre-Whitening) Plaintext di-XOR dengan sebagian kunci sebelum masuk ke ronde utama.
    2. 16 Ronde Feistel Network Twofish adalah algoritma berbasis Feistel structure. Setiap ronde, plaintext dibagi 2 bagian (kiri dan kanan). Fungsi F digunakan untuk memproses setengah blok menggunakan: S-box yang dibangkitkan dari kunci (bukan statis). MDS (Maximum Distance Separable) matrix untuk difusi. Bagian hasil F di-XOR dengan bagian lainnya, lalu ditukar (Feistel swap).
    3. Post-Whitening Setelah 16 ronde selesai, hasilnya kembali di-XOR dengan subkunci terakhir. Proses Dekripsi Ciphertext hex diubah dulu ke dalam bentuk biner (byte array). AES menjalankan Inverse Cipher: AddRoundKey (dibalik), Inverse ShiftRows, Inverse SubBytes, Inverse MixColumns (kecuali ronde terakhir), Proses ini dilakukan untuk 10 ronde (karena AES-128). Output hasil dekripsi berupa plaintext asli.
e. *Output* (Ciphertext dalam Hex) Hasil enkripsi yang muncul: “6a7b08037c766d 667e610666687b 697c222121202927321202122020223132333 4” Ini adalah ciphertext dalam format heksadesimal, Jika diubah ke biner, panjangnya tetap 128 bit (16 byte), Bentuk heksadesimal inilah yang disimpan / bisa dipakai untuk dekripsi.
f. Waktu Enkripsi Proses enkripsi selesai dalam: 0.002365 detik, Sangat cepat untuk plaintext 1 blok (16 byte).



Gambar 7. proses Enkripsi Twofish

a. Input Ciphertext Cipherteks dalam bentuk heksadesimal dimasukkan: 6a 7b 08 03 7c 76 6d 66 7e 61 06 68 67 8c 22 21 ... Ini adalah hasil enkripsi dari plainteks sebelumnya.
b. Key Expansion (Ekspansi Kunci)
Kunci "2111009731012002" (16 karakter, 128-bit) diproses untuk menghasilkan: Subkey Whitening (kunci untuk XOR awal/akhir), Round Keys (kunci untuk tiap ronde), S-box Keys (digunakan dalam fungsi g). Pada Twofish dengan kunci 128-bit → total ada 40 subkeys + 4 S-box keys.

c. Inverse Rounds (16 ronde dekripsi)
   Dekripsi dilakukan dengan kebalikan enkripsi: Setiap blok (128-bit = 16 byte) ciphertext dipecah menjadi 4 word (32-bit), Dilakukan whitening awal (XOR dengan subkey), Kemudian diproses 16 ronde dekripsi dengan urutan round keys terbalik, Fungsi utama yang digunakan adalah Feistel structure dengan fungsi g() (memakai S-box berbasis key), Setelah semua ronde selesai dilakukan whitening akhir.
d. Output
   Hasil dari dekripsi berupa plainteks asli: "XJ92LFTQMP6WZKYN" yang merupakan teks awal sebelum dienkripsi.
e. Waktu Eksekusi
   Dekripsi berlangsung sangat cepat, hanya 0.003335 detik, karena blok hanya 16 byte dan algoritma Twofish efisien.

4. KESIMPULAN

Perbandingan performa menunjukkan bahwa Twofish memiliki keunggulan signifikan dalam efisiensi *throughput* (*Payload Capacity*, sekitar 11 kali lebih cepat dari AES) dan stabilitas data (nilai *Bit Error Rate* lebih rendah). Sementara itu, AES unggul dalam konsistensi waktu proses enkripsi dan dekripsi yang lebih stabil. Kedua algoritma aman (kunci 128 *bit*) dan merupakan *cipher block* simetris sehingga tidak mengubah ukuran file hasil enkripsi secara signifikan. Pemilihan algoritma bergantung pada prioritas: Twofish optimal untuk volume data besar, dan AES untuk konsistensi waktu. Proses implementasi kedua algoritma pada pemrograman Python berhasil dilakukan melalui tiga tahapan kunci: Persiapan Data (*plaintext* dipastikan berukuran blok 16 *byte* dengan penambahan *padding*), Enkripsi Inti (AES: 10 putaran non-Feistel; Twofish: 16 putaran Feistel dengan *Whitening*), dan Dekripsi menggunakan kunci yang sama secara terbalik. Hal ini membuktikan kedua algoritma fungsional dan efektif untuk pengamanan file teks.

REFERENCES

[1] R. Agus, G. Gultom, and A. Info, "Jurnal Pertahanan," 2017.
[2] A. Pratama, S. Kom, L. A. M, and S. Si, "16 + 1 (," 2021.
[3] I. Journal, C. Applications, and E. H. No, "A Symmetric Key Cryptographic Algorithm," vol. 1, no. 15, pp. 1–4, 2010.
[4] S. Geraldi and U. M. Nusantara, "Tabel 2.1. Jumlah Putaran Berdasarkan Ukuran (1 word = 32 bit)," no. November, pp. 6–18, 2001.
[5] D. Nim, "Algoritma Twofish Sebagai Finalis AES dan Metode Kriptanalisisnya," 2011.
[6] E. E. Awal *et al.*, "Jurnal Mahasiswa Ilmu Komputer (JMIK) Jurnal Mahasiswa Ilmu Komputer (JMIK)," vol. 03, no. 01, pp. 1–6, 2022.
[7] I. Shulhan, "Analisis Perbandingan Antara Algoritma Rijndael Dan Algoritma Twofish Dalam Penyandian Teks," vol. 03, 2018.
[8] I. Herianto, "ANALISIS PERBANDINGAN PERFORMANSI ALGORITMA Advance Encryption Standard (AES) dan Twofish PADA BLOK CIPHER," 2015.
[9] R. K. Muhammed *et al.*, "Comparative Analysis of AES, Blowfish, Twofish, Salsa20, and ChaCha20 for Image Encryption," 2024.
[10] A. Z. Hussain and A. Ali, "Medical image encryption using multi chaotic maps," vol. 21, no. 3, pp. 556–565, 2023, doi: 10.12928/TELKOMNIKA.v21i3.24324.
[11] K. Assa-agyei and F. Olajide, "A Comparative Study of Twofish , Blowfish , and Advanced Encryption Standard for Secured Data Transmission," vol. 14, no. 3, pp. 393–398, 2023.
[12] I. Mukmin, "Algoritma Twofish : Kinerja dan Implementasinya Sebagai Salah Satu Kandidat Algoritma AES (Advanced Encryption Standard)," *Informatika*, no. C, pp. 3–4, 2013.