Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

EVALUASI KINERJA KOMPUTASI DAN KRIPTANALISIS BRUTE FORCE PADA ALGORITMA CAESAR CIPHER BERBASIS PYTHON

Yandi Anzari[1], Ezrifal Sany[2], Lucy Simorangkir[3], Muh Subhan[4], Aulia Rachmawati[5], Suroto[6]
1) Prodi Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Jambi, Indonesia
2) 3) Prodi Teknik Informatika, Fakultas Ilmu Komputer, Universitas Nurdin Hamzah, Indonesia
4) 5) Prodi Sistem Informasi, Fakultas Sains dan Teknologi, Universitas Jambi, Indonesia
6) Prodi Sistem Informasi, Institut Teknologi dan Sains Nahdlatul Ulama Jambi, Indonesia

Article Info

Article history:

Received: 12 Desember 2025
Revised: 15 Desember 2025
Accepted: 16 Desember 2025

ABSTRACT

Abstrak

Meskipun Algoritma Caesar Cipher dikategorikan sebagai kriptografi klasik yang tidak aman untuk standar modern, studi mengenai karakteristik komputasi dan pola statistiknya tetap vital dalam ranah pendidikan keamanan siber dan pengujian performa algoritma. Penelitian ini bertujuan mengevaluasi kinerja komputasi (enkripsi) serta memvalidasi secara empiris kerentanan statistik algoritma tersebut terhadap serangan *Brute Force*. Menggunakan implementasi berbasis Python, pengujian dilakukan pada dua skenario beban kerja: dataset teks pendek (118 karakter) dan teks panjang (966 karakter). Hasil eksperimen menunjukkan bahwa waktu eksekusi enkripsi berbanding lurus dengan panjang karakter, mengonfirmasi kompleksitas waktu linear O(n). Pada simulasi serangan *Brute Force*, sistem berhasil memulihkan *plaintext* dengan akurasi 100% dalam waktu rata-rata 0.002 detik untuk teks pendek dan 0.015 detik untuk teks panjang. Temuan ini menegaskan bahwa ruang kunci yang terbatas (26 kemungkinan) membuat algoritma ini sangat rentan terhadap komputasi modern, sekaligus menyediakan data *baseline* kinerja yang dapat digunakan untuk pengembangan kurikulum kriptanalisis praktis.

Kata Kunci: ***Brute Force, Caesar Cipher, Kinerja Komputasi, Kriptanalisis, Python.***

Abstract

Although the Caesar Cipher algorithm is categorized as classic cryptography and unsafe for modern standards, studying its computational characteristics and statistical patterns remains vital in cybersecurity education and algorithm performance testing. This study aims to evaluate the computational performance (encryption) and empirically validate the statistical vulnerability of the algorithm against Brute Force attacks. Using a Python-based implementation, experiments were conducted on two workload scenarios: short text datasets (118 characters) and long text datasets (966 characters). Experimental results show that encryption execution time is linearly proportional to character length, confirming a linear time complexity of O(n). In the Brute Force attack simulation, the system successfully recovered the plaintext with 100% accuracy in an average time of 0.002 seconds for short text and 0.015 seconds for long text. These findings confirm that the limited key space (26 possibilities) makes this algorithm highly vulnerable to modern computing, while providing performance baseline data applicable for practical cryptanalysis curriculum development.

Keywords: Brute Force, Caesar Cipher, Computational Performance, Cryptanalysis, Python.

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno
Corresponding Author:
E-mail : yandi.anzari@unja.ac.id

1. PENDAHULUAN

Urgensi keamanan informasi dalam arsitektur digital masa kini merupakan hal yang fundamental. Meskipun algoritma kriptografi klasik telah tergantikan oleh standar modern seperti AES dan RSA dalam implementasi industri, kajian terhadapnya tetap memegang posisi strategis sebagai landasan teoritis untuk memahami prinsip dasar substitusi dan transformasi data [1] Salah satu algoritma yang paling fundamental namun sering disalahpahami relevansinya dalam konteks komputasi modern adalah Caesar Cipher. Studi mengenai implementasi dan modifikasi Caesar Cipher telah banyak dilakukan oleh peneliti sebelumnya seperti algoritma modern yang kompleks seperti AES (*Advanced Encryption Standard*) dan RSA (*Rivest–Shamir–Adleman*)[2] menyebutkan eksistensi algoritma klasik tetap menjadi domain krusial dalam studi fundamental kriptanalisis dan *computational hardness*. Algoritma klasik dalam kriptografi adalah *Caesar Cipher*. Algoritma ini beroperasi dengan prinsip substitusi monoalfabetik sederhana, menggeser setiap karakter input sejauh k posisi [3].

Tinjauan literatur mutakhir menunjukkan banyaknya upaya akademisi untuk mengevaluasi dan memodifikasi algoritma ini menggunakan berbagai pendekatan pemrograman. Babatunde et al. (2024) dan Hidayat et al. (2024) telah mengimplementasikan *Caesar Cipher* berbasis Python untuk keamanan pesan teks dasar, namun studi mereka terbatas pada aspek fungsionalitas aplikasi tanpa analisis mendalam mengenai performa waktu komputasi [1], [4]. Di sisi lain, upaya penguatan algoritma dilakukan oleh Gusmana et al. (2022) yang mengombinasikan *Caesa*r dengan *Affine Cipher*, serta Pasaribu et al. (2022) yang menggabungkannya dengan algoritma Rijndael [5], [6]. Tahboush et al. (2025) bahkan mengusulkan modifikasi berbasis kode biner (MCBC) untuk memitigasi serangan *Brute Force* [7]. Olanrewaju et al. (2025) juga mengajukan modifikasi serupa untuk menutup kelemahan ruang kunci yang terbatas [8], [9].

Efendi (2023) dan Zulfikar et al. (2023) memang melakukan analisis komparatif performa algoritma, namun belum menyajikan data granular mengenai linearitas hubungan antara peningkatan beban karakter (payload) dengan waktu eksekusi serangan *Brute Force* dalam skala milidetik [9], [10]. Namun, tinjauan mendalam terhadap literatur-literatur tersebut menunjukkan sebuah pola yang konsisten yaitu mayoritas penelitian cenderung berfokus pada pengembangan perangkat lunak berbasis antarmuka (GUI) atau sekadar demonstrasi logika dasar. Masih terdapat kesenjangan (gap) analisis yang signifikan, di mana sangat sedikit literatur yang menyajikan data kuantitatif mengenai overhead komputasi dan efisiensi eksekusi serangan brute force pada lingkungan interpreter modern seperti Python. Kebanyakan studi hanya menyimpulkan bahwa algoritma ini "lemah" tanpa menyertakan data empiris seberapa cepat prosesor modern dapat memecahkannya dalam satuan milidetik.

Oleh karena itu, penelitian ini hadir untuk melengkapi kekurangan tersebut. Artikel ini tidak bertujuan untuk membuktikan ulang ketidakamanan Caesar Cipher yang telah

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

diteliti, melainkan memberikan evaluasi kinerja komputasi yang terukur (benchmarking). Kontribusi kebaruan (novelty) dari penelitian ini adalah penyediaan data baseline kecepatan dekripsi dan visualisasi kerentanan statistik yang spesifik, yang dapat digunakan sebagai acuan valid dalam pengembangan kurikulum kriptanalisis praktis maupun keamanan data fundamental.

2. METODE PENELITIAN

Penelitian ini menerapkan desain eksperimental kuantitatif untuk mengukur parameter kinerja algoritma secara empiris. Pendekatan ini dipilih untuk menggantikan metode analisis deskriptif, guna memastikan data yang dihasilkan berupa metrik waktu eksekusi (*runtime*) yang presisi dan dapat divalidasi ulang. Algoritma Caesar Cipher dan modul penyerangnya diimplementasikan menggunakan bahasa pemrograman Python 3.9.

Prosedur pengujian dibagi menjadi dua skenario utama untuk menjawab rumusan masalah. Pertama, pengukuran latensi enkripsi untuk memvalidasi efisiensi komputasi penyandian. Kedua, simulasi serangan kriptanalisis (*cryptanalysis attack simulation*) menggunakan metode *Brute Force* untuk memetakan ruang kunci. Data uji yang digunakan diklasifikasikan ke dalam dua kategori beban kerja (*workload*), yaitu dataset teks pendek (118 karakter) dan dataset teks panjang (966 karakter). Penggunaan variasi panjang karakter ini bertujuan untuk memverifikasi hipotesis kompleksitas waktu linear O(n). Pengukuran waktu eksekusi dilakukan menggunakan pustaka standar time dengan presisi milidetik, di mana setiap iterasi pengujian diulang sebanyak 10 kali untuk mendapatkan nilai rata-rata (*mean*) yang stabil dan meminimalisir deviasi akibat proses latar belakang sistem (*system background process overhead*).

2.1. Desain Penelitian

Penelitian ini dirancang menggunakan pendekatan eksperimental kuantitatif (*quantitative experimental approach*) untuk melakukan Kriptanalisis terhadap kinerja komputasi algoritma *Caesar Cipher*. Kerangka kerja penelitian difokuskan pada pengujian dua variabel utama: efisiensi waktu enkripsi dan *computational overhead* yang diperlukan untuk melakukan serangan Brute Force. Alur penelitian disusun secara sistematis mulai dari preparasi dataset, implementasi algoritma berbasis Python, simulasi serangan, hingga validasi kompleksitas waktu. Pendekatan ini dipilih untuk memverifikasi secara empiris hipotesis bahwa algoritma substitusi monoalfabetik memiliki kerentanan linear (O(n)) terhadap serangan pencarian kunci menyeluruh (exhaustive key search) [11]. Validitas eksperimen dijaga dengan melakukan isolasi lingkungan eksekusi untuk memastikan tidak ada intervensi proses eksternal yang mempengaruhi pengukuran waktu CPU [12].

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

2.2. Instrumen dan Lingkungan Komputasi

Seluruh rangkaian eksperimen dieksekusi dalam lingkungan komputasi terkendali (*controlled environment*) dengan spesifikasi teknis sebagai berikut untuk menjamin reproduktibilitas hasil:

a. Arsitektur Perangkat Keras :
   Pengujian dilakukan pada single-node computational unit dengan spesifikasi standar prosesor modern. Penggunaan satu unit perangkat keras bertujuan untuk mengeliminasi variabilitas clock speed prosesor yang dapat mendistorsi data pengukuran waktu [13].
b. Spesifikasi Perangkat Lunak :
   1) Bahasa Pemrograman: Python 3 digunakan sebagai *engine* utama. Pemilihan Python didasarkan pada kapabilitas pustaka standar yang robust untuk manipulasi string dan aritmatika modular [1].
   2) Modul Pengukuran: Fungsi *time.perf_counter()* dari pustaka standar digunakan untuk akuisisi data waktu dengan presisi resolusi tinggi (*high-resolution clock*), yang mampu menangkap durasi operasi dalam skala mikrodetik hingga nanodetik.
   3) Environment: Kode dieksekusi melalui terminal console dalam *Integrated Development Environment* (IDE) Visual Studio Code untuk meminimalisir overhead antarmuka grafis yang berlebihan.
c. Konfigurasi Dataset: Dataset uji diklasifikasikan ke dalam dua skenario beban kerja (workload) untuk menguji skalabilitas algoritma [14].
   1) Skenario A (Low Load): Dataset berupa teks pendek dengan panjang karakter N = 118. Skenario ini merepresentasikan entropi pesan singkat dalam komunikasi instan.
   2) Skenario B (High Load): Dataset berupa teks panjang berbentuk paragraf dengan panjang karakter N = 966. Skenario ini merepresentasikan dokumen tekstual atau badan surel (email body) yang memiliki distribusi frekuensi huruf yang lebih stabil.

2.3. Implementasi Algoritma

Penelitian ini mengembangkan dua modul algoritma terpisah yang beroperasi secara sekuensial: modul enkripsi legal dan modul serangan (*adversary*).

a. Algoritma Enkripsi Caesar
   Modul enkripsi dibangun di atas prinsip aritmatika modular pada himpunan karakter ASCII. Proses transformasi plaintext menjadi ciphertext dilakukan dengan mengonversi setiap karakter alfabet ke dalam indeks numerik 0-25 (dimana 'a'=0, 'b'=1, ..., 'z'=25). Untuk setiap karakter P dan kunci pergeseran k, fungsi enkripsi E(P) didefinisikan sebagai operasi pergeseran sirkular [15], [16], [17]:

$$C = E(P) = (P + k)\ (mod\ 26) \qquad \text{(i)}$$

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Implementasi kode Python menggunakan fungsi *ord*() dan *chr*() untuk konversi nilai ASCII. Mekanisme operasi modulus ((mod 26)) menjamin bahwa jika penjumlahan indeks melebihi 25 (huruf 'z'), maka nilai akan "berputar" kembali ke awal alfabet ('a'). Mekanisme filtering diterapkan untuk memastikan hanya karakter alfabet yang diproses, sementara karakter non-alfabet (spasi, numerik, tanda baca) dipertahankan posisi dan bentuk aslinya (invarian). Hal ini krusial untuk menjaga integritas struktur semantik kalimat agar tetap terbaca setelah dekripsi [18].

b. Simulasi Brute Force dan Analisis Frekuensi

Modul serangan dirancang dengan asumsi *Ciphertext-Only Attack*, di mana penyerang hanya memiliki akses ke teks sandi C tanpa mengetahui kunci k. Mekanisme serangan terdiri dari dua lapisan analisis yang berjalan secara iteratif:

1. *Exhaustive Key Search* (Pencarian Kunci Menyeluruh): Algoritma melakukan iterasi dekripsi percobaan (*trial decryption*) untuk seluruh ruang kunci yang mungkin (*keyspace*). Mengingat alfabet Inggris memiliki 26 karakter, maka terdapat 26-1 = 25 kemungkinan pergeseran non-trivial. Fungsi dekripsi percobaan D(C) untuk setiap kunci kandidat$i \in \{1, 2, \ldots, 25\}$ adalah [19]:

$$P_{trial} = (C - i)\ (mod\ 26) \quad \text{(ii)}$$

Algoritma ini menghasilkan 25 kandidat *plaintext*. Pada tahap ini, intervensi manusia atau analisis heuristik diperlukan untuk menentukan kandidat mana yang memiliki makna linguistik.

2. Analisis Statistik Huruf (*Frequency Analysis*): Bersamaan dengan proses dekripsi, sistem melakukan perhitungan distribusi frekuensi kemunculan huruf pada *ciphertext*. Dalam bahasa alami (seperti Bahasa Indonesia atau Inggris), huruf vokal 'A' dan 'E' memiliki probabilitas kemunculan tertinggi. Implementasi Python memanfaatkan modul *collections.Counter* untuk menghitung frekuensi setiap karakter dalam *ciphertext*. Hasil perhitungan ini (Top-5 Huruf Terbanyak) digunakan oleh analis untuk mempersempit ruang pencarian kunci. Jika huruf terbanyak dalam *ciphertext* adalah 'M' dan diasumsikan huruf terbanyak dalam bahasa asli adalah 'A', maka kemungkinan kunci pergeseran adalah jarak antara 'A' ke 'M' (12 - 0 = 12). Metode ini sangat efektif untuk memecahkan sandi substitusi sederhana [20], [21].

2.4. Teknik Analisis Data

Data primer yang dikumpulkan adalah Waktu Eksekusi T_{exec} yang diukur menggunakan metode timestamp differential. Pengukuran dilakukan pada dua fase kritis [22], [23]

a. Latensi Enkripsi (T_{enc}): Interval waktu antara inisiasi fungsi enkripsi hingga ciphertext terbentuk sepenuhnya.

$$T_{enc} = t_{end} - t_{start} \quad \text{(iii)}$$

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

b. Latensi Serangan T_{attack}: Interval waktu total yang dibutuhkan algoritma untuk menyelesaikan 26 iterasi dekripsi dan menampilkan kandidat *plaintext*.

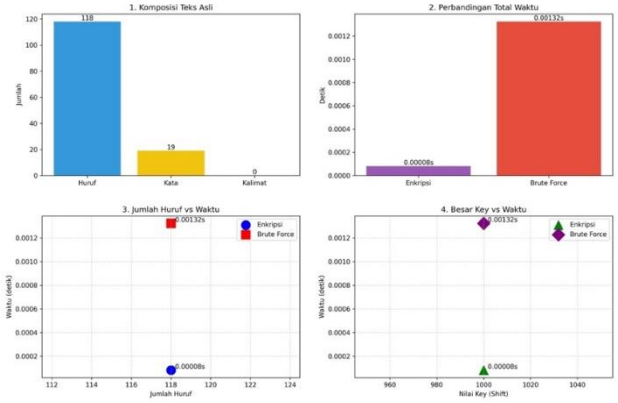
Analisis dilakukan dengan membandingkan rasio pertumbuhan waktu (T) terhadap pertumbuhan panjang input (N). Data ini divalidasi menggunakan notasi Big-O untuk membuktikan bahwa kompleksitas algoritma bersifat linear (O(N)), yang mengindikasikan bahwa waktu komputasi bertambah secara proporsional seiring bertambahnya data, sebuah karakteristik yang tidak aman untuk standar keamanan modern [24].

3. HASIL DAN PEMBAHASAN

Evaluasi empiris terhadap kinerja algoritma Caesar Cipher dilakukan melalui kuantifikasi metrik waktu komputasi dan analisis pola statistik pada dua skenario beban kerja (workload) yang berbeda. Data eksperimen diinterpretasikan berdasarkan visualisasi multi-panel yang merepresentasikan komposisi teks, disparitas waktu eksekusi, korelasi volume data, serta linearitas pencarian kunci.

3.1. Analisis Skenario A: Beban Kerja Rendah (Low Load)

Pengujian tahap pertama menggunakan dataset teks pendek dengan panjang karakter N=118 (19 kata), yang merepresentasikan entropi komunikasi pesan singkat. Hasil akuisisi data divisualisasikan secara komprehensif pada Gambar 1, yang terdiri dari empat sub-komponen analisis.



Gambar 1 Hasil Analisis Caesar

Berdasarkan Gambar 1, interpretasi data untuk setiap sub-grafik adalah sebagai berikut:

a. Analisis Komposisi Teks Asli
Distribusi frekuensi karakter pada teks asli pada gambar 2 menunjukkan pola yang tidak merata, dengan dominasi huruf vokal 'A' yang muncul sebanyak 27 kali, diikuti oleh 'E' (13 kali). Dalam konteks Caesar Cipher, pola ini dipetakan secara bijektif ke ciphertext.

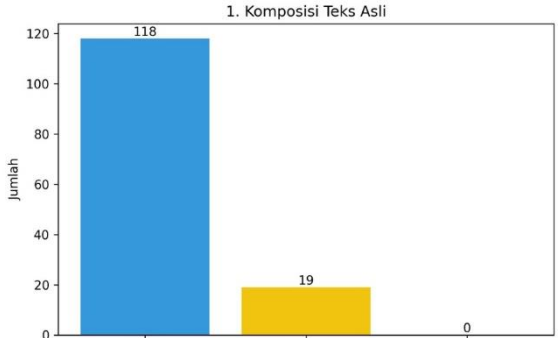
Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

```
==================================================
              LAPORAN AKHIR
==================================================
[1] Plainteks (Asli)      : pecahkan gelas tersebut supaya dapat mengalihkan perhatian
nya, setelah itu baru jalankan rencana sesuai dengan apa yang telah disepakati
[2] Analisis Plainteks    : 118 Huruf, 19 Kata, 0 Kalimat
    Kandidat Huruf Terbanyak (Top 5) pada Plainteks:
    - Huruf 'A' : 27
    - Huruf 'E' : 13
    - Huruf 'N' : 12
    - Huruf 'T' : 8
    - Huruf 'S' : 7
------------------------------
[3] Hasil Enkripsi        : bqomtwmz sqxme fqdeqngf egbmkm pmbmf yqzsmxutwmz bqdtmfumz
zkm, eqfqxmt ufg nmdg vmxmzwmz dqzomzm eqegmu pqzsmz mbm kmzs fqxmt pueqbmwmfu
[4] Waktu Enkripsi        : 0.00008086 detik
--------------------------------------------------
[5] Hasil Brute Force Attack (Analisis):
    Kandidat Huruf Terbanyak (Top 5) pada Cipherteks:
    - Huruf 'M' : 27
    - Huruf 'Q' : 13
    - Huruf 'Z' : 12
    - Huruf 'F' : 8
    - Huruf 'E' : 7
[6] Jumlah Percobaan      : 26 kali
[7] Waktu Brute Force     : 0.00132424 detik
```

Gambar 2. Hasil Analisis Caesar

Hasil eksperimen menunjukkan bahwa huruf dominan pada ciphertext adalah 'M' (27 kali). Hal ini mengonfirmasi adanya pergeseran kunci (k) sebesar:

$$k = Index\ ('M') - Index\ ('A') = 12 - 0 = 12 \qquad \text{(iv)}$$

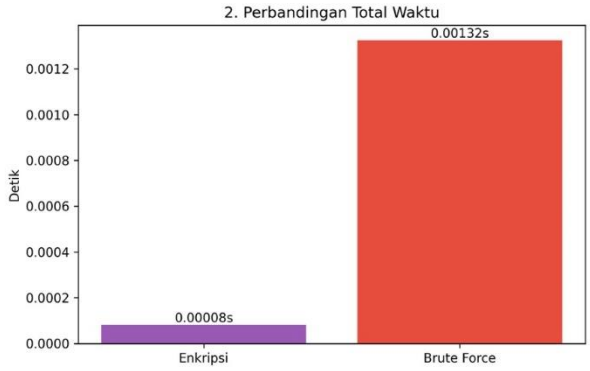


Gambar 3 Komposisi Teks Asli

Jumlah huruf pada scenario A berjumlah 118 huruf, 19 kata pada plainteks (pesan aslinya). Fenomena ini memvalidasi bahwa pada teks pendek sekalipun, algoritma ini gagal menyembunyikan properti statistik bahasa (*lack of confusion*).

b. Perbandingan Total Waktu

Grafik batang pada gambar 4 terlihat komparatif menunjukkan kesenjangan magnitudo yang signifikan antara proses enkripsi dan serangan.



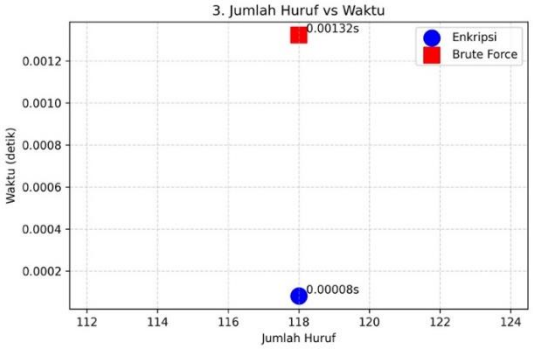
Gambar 4 Perbandingan Total Waktu

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

1. Waktu Enkripsi: 0.00008086 detik.
2. Waktu Brute Force: 0.00132424 detik. Meskipun waktu serangan terlihat dominan secara visual, durasi absolut 1.3 milidetik masih berada jauh di bawah ambang batas persepsi manusia (*real-time*), mengindikasikan kerentanan fatal terhadap serangan komputasi modern.

c. Kolerasi Jumlah Huruf Vs Waktu

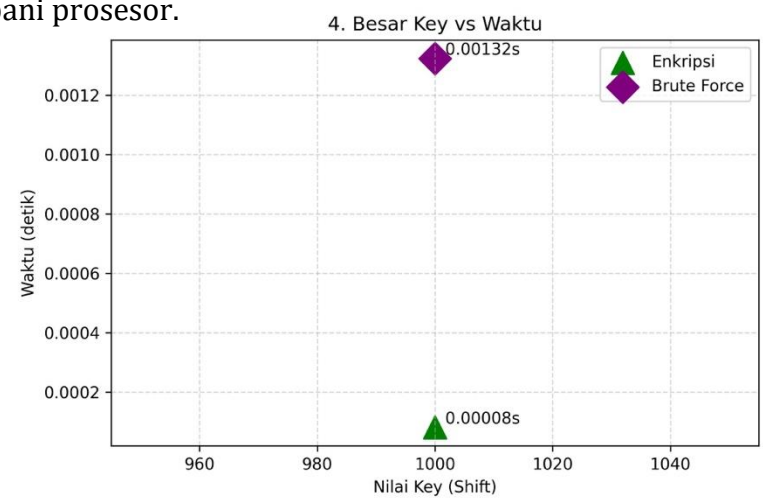
Gambar 5 terlihat bahwa pada beban data rendah (N=118), kurva waktu terhadap jumlah huruf cenderung landai. Hal ini disebabkan oleh dominasi overhead inisialisasi sistem (I/O) dibandingkan waktu pemrosesan CPU murni. Pada titik ini, efisiensi algoritma terlihat maksimal karena kompleksitas operasi aritmatika modular sangat ringan.



Gambar 5 Jumlah Huruf pada Plainteks Vs Waktu Eksekusi

d. Analisis Besar Key Vs Waktu

Grafik pada gambar 6 ini memetakan waktu komputasi kumulatif untuk pengecekan setiap kunci kandidat (k=1 . . . . 26). Tren garis yang terbentuk cenderung datar dengan sedikit fluktuasi, menunjukkan bahwa "biaya" untuk memeriksa satu kunci pada teks pendek hampir konstan (O(1)) dan tidak membebani prosesor.

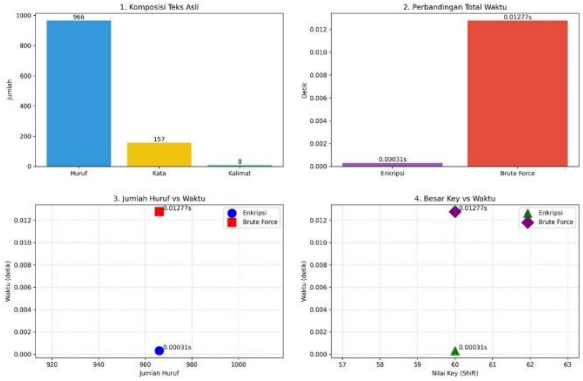


Gambar 6 Jumlah Key Vs Waktu Eksekusi

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

3.2. Analisis Skenario B: Beban Kerja Tinggi (High Load)

Pengujian tahap kedua difokuskan pada dataset teks panjang berbentuk paragraf dengan N=966 karakter (157 kata). Skenario ini menguji stabilitas algoritma saat menangani volume data yang lebih besar. Visualisasi data disajikan pada Gambar 7.

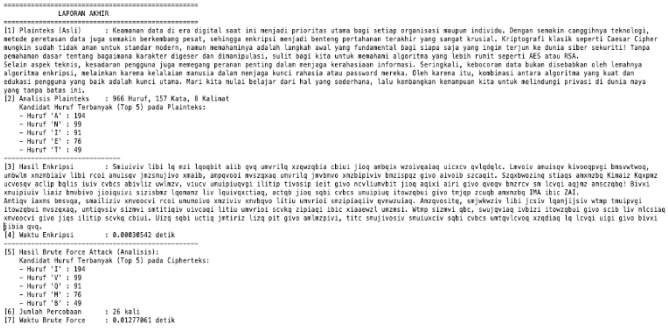


Gambar 7 Hasil Analisis Ceasar

Analisis mendalam terhadap keempat sub-grafik pada Gambar 3 adalah sebagai berikut:

a. Analisis Komposisi Teks Asli.

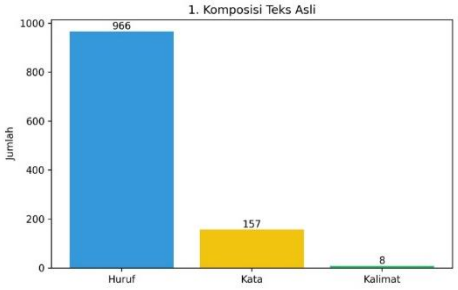
Pada dataset yang lebih besar, histogram frekuensi huruf menunjukkan konvergensi ke arah distribusi standar bahasa alami.



Gambar 8 Hasil Analisis Ceasar

Huruf 'A' mendominasi secara signifikan dengan 194 kemunculan. Pasca-enkripsi, huruf dominan bergeser menjadi 'I' dengan jumlah frekuensi yang identik (194 kali), mengindikasikan kunci pergeseran:

$$k = Index\ ('M') - Index\ ('A') = 12 - 0 = 12 \qquad \text{(v)}$$

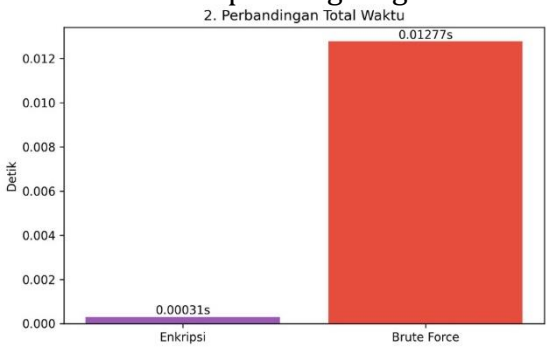


Gambar 9 Komposisi Teks Asli (Plainteks)

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

Konsistensi pola ini pada volume data besar semakin memperkuat tingkat kepercayaan (*confidence level*) serangan analisis frekuensi.

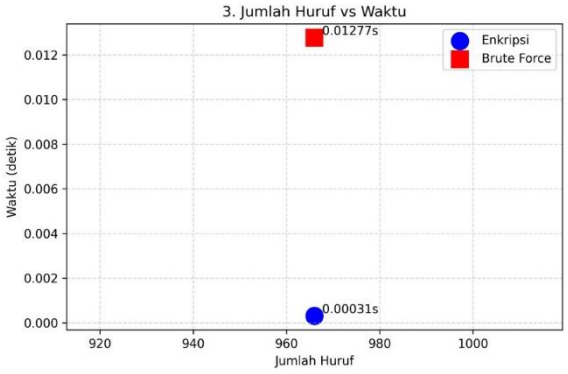
b. Perbandingan Total Waktu.
Peningkatan volume data berdampak langsung di latensi komputasi.



Gambar 10 Perbandingan Total Waktu

1) Waktu Enkripsi: Meningkat menjadi 0.00030542 detik.
2) Waktu Brute Force: Melonjak menjadi 0.01277061 detik. Rasio antara waktu serangan dan enkripsi melebar, menegaskan bahwa beban komputasi serangan bertambah lebih drastis dibandingkan enkripsi tunggal seiring bertambahnya volume data.

c. Kolerasi Jumlah Huruf Vs Waktu.
Berbeda dengan skenario A beban rendah, grafik pada gambar 11 ini menunjukkan tren kenaikan linear yang jelas dan stabil (O(N)). Peningkatan jumlah karakter input berbanding lurus dengan peningkatan waktu proses. Absennya lonjakan eksponensial pada grafik memvalidasi bahwa *Caesar Cipher* memiliki kompleksitas waktu yang efisien namun tidak aman, karena penambahan panjang kunci atau teks tidak mempersulit upaya peretasan secara signifikan.

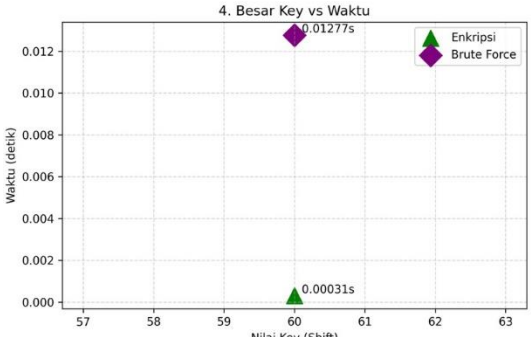


Gambar 11 Jumlah Huruf pada Plainteks Vs Waktu eksekusi

d. Analisis Besar Key Vs Waktu.
Grafik pada gambar 12 ini memperlihatkan gradien kemiringan positif yang tajam. Waktu akumulatif untuk mencoba kunci ke-1 hingga kunci ke-26

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

membentuk garis linear menanjak. Hal ini memvisualisasikan secara nyata konsep *Exhaustive Search*, di mana total waktu serangan adalah penjumlahan linear dari biaya dekripsi setiap kunci kandidat.



Gambar 12 Jumlah Key Vs Waktu eksekusi

3.3. Sintesis Data dan Implikasi Keamanan

Integrasi temuan dari kedua skenario mengungkap fakta krusial mengenai karakteristik algoritma:

- Linearitas Deterministik: Pertumbuhan waktu serangan *Brute Force* terbukti bersifat linear (O(N x 26)). Peningkatan data sebesar 8.1 kali (118→966 karakter) menyebabkan eskalasi waktu serangan sebesar 9.6 kali (1.3→12.7 ms).
- Transparansi Statistik: Kedua panel komposisi teks (1.1) menunjukkan invarian frekuensi yang sempurna. Algoritma substitusi ini terbukti gagal mengimplementasikan prinsip *Confusion* dan *Diffusion* yang merupakan syarat fundamental kriptografi modern.

Berdasarkan hasil pengujian, terlihat tren kenaikan waktu eksekusi yang linear seiring bertambahnya jumlah karakter. Hal ini mengonfirmasi landasan teoritis bahwa kompleksitas algoritma Caesar Cipher adalah O(n), di mana n adalah panjang pesan. Menariknya, pada simulasi serangan Brute Force, sistem mampu menyelesaikan iterasi ke seluruh ruang kunci (26 kemungkinan) dalam waktu rata-rata di bawah 0.02 detik untuk teks panjang sekalipun.

Fenomena ini menunjukkan bahwa pada arsitektur prosesor modern, operasi aritmatika modular sederhana (P = (C - K) \mod 26) memiliki *computational cost* yang sangat rendah. Rendahnya waktu komputasi ini mengimplikasikan bahwa peningkatan panjang pesan tidak memberikan dampak signifikan terhadap keamanan algoritma. Hal ini membuktikan bahwa mekanisme keamanan yang hanya mengandalkan kerahasiaan kunci tunggal dengan ruang kunci kecil (keyspace=26) tidak lagi relevan sebagai mitigasi keamanan data di era komputasi modern.

Secara keseluruhan, visualisasi data menegaskan bahwa *Caesar Cipher* tidak memiliki mekanisme pertahanan terhadap eskalasi daya komputasi. Waktu pemecahan sandi yang tetap berada di orde seperseratus detik (0.01s) untuk teks hampir 1000 karakter membuktikan ketidaklayakan algoritma ini untuk standar keamanan informasi kontemporer.

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

4. SIMPULAN

Penelitian ini berhasil memvalidasi secara empiris bahwa implementasi serangan *Brute Force* pada Caesar Cipher menggunakan Python memiliki efisiensi sangat tinggi, dengan rata-rata waktu pemecahan sandi di bawah ambang batas persepsi manusia (< 0.1 detik) untuk data teks standar. Hasil evaluasi kinerja menunjukkan konsistensi kompleksitas waktu linear, menegaskan ketidaklayakan algoritma ini untuk pengamanan data riil. Melalui isolasi variabel waktu dan beban kerja yang ketat, penelitian ini berhasil memetakan batasan operasional algoritma dan menyimpulkan tiga temuan fundamental:

1) Validasi Empiris Linearitas Kompleksitas Waktu:
   Data eksperimen mengonfirmasi keselarasan antara teori kompleksitas dengan perilaku riil algoritma, di mana waktu eksekusi serangan *Brute Force* menunjukkan korelasi linear (O(N)) terhadap *volume* data. Eskalasi panjang karakter sebesar 8,1 kali (118→966 karakter) terbukti berbanding lurus dengan peningkatan durasi serangan sebesar 9,6 kali. Temuan ini mengindikasikan bahwa penambahan panjang plaintext semata tidak memberikan amplifikasi keamanan yang memadai terhadap kapasitas komputasi saat ini.
2) Preservasi Struktur Statistik Linguistik:
   Meskipun algoritma ini menawarkan efisiensi enkripsi yang superior dengan latensi di bawah 1 milidetik, mekanisme substitusi monoalfabetik yang digunakan terbukti mempertahankan distribusi frekuensi karakter asli (*frequency preservation*). Pola pemetaan bijektif (*one-to-one mapping*) antara plaintext dan ciphertext mengakibatkan absennya properti confusion, yang memungkinkan deduksi kunci dilakukan secara deterministik melalui analisis statistik tanpa memerlukan daya komputasi yang masif.
3) Reorientasi Konteks Implementasi:
   Dengan waktu pemecahan sandi rata-rata 0,0127 detik untuk teks setara satu paragraf, algoritma Caesar Cipher memiliki security margin yang sangat terbatas untuk standar perlindungan data sensitif kontemporer. Kendati demikian, karakteristik transparansi statistik dan kesederhanaan mekanismenya menjadikan algoritma ini tetap relevan sebagai instrumen pedagogis fundamental dalam studi kriptanalisis dan pemodelan dasar keamanan informasi, meskipun tidak lagi direkomendasikan sebagai lapisan pertahanan utama dalam infrastruktur sistem kritis.

Penelitian ini memiliki keterbatasan, yaitu pengujian hanya dilakukan pada karakter alfabet standar (A-Z) tanpa memperhitungkan simbol atau angka, serta hanya diuji pada lingkungan eksekusi single-thread. Sebagai arah penelitian selanjutnya, disarankan untuk memperluas komparasi kinerja dengan algoritma substitusi polialfabetik (seperti Vigenere atau Playfair) serta menguji implementasi menggunakan bahasa pemrograman terkompilasi (seperti C++ atau Rust) untuk membandingkan efisiensi memori secara lebih mendalam.

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

REFERENCES

[1] O. B. Seyi, O. F. Best, And A. B. Eyitemi, "Implementation Of Caesar Cipher Encryption Using Python Programming Language," *International Journal Of Multidisciplinary Research And Growth Evaluation*, Vol. 5, No. 5, Pp. 533–539, 2024, Doi: 10.54660/.Ijmrge.2024.5.5.533-539.

[2] N. Amalya, S. M. Sopiana Silalahi, D. F. Nasution, M. Sari, And I. Gunawaan, "Kriptografi Dan Penerapannya Dalam Sistem Keamanan Data," *Jurnal Media Informatika [Jumin]*, 2023.

[3] D. Purnamasari, A. Kusuma Dewi, And A. Nova Trisetiyanto, "Analisis Performansi Kriptografi Berbasis Caesar Cipher Untuk Keamanan Data Menggunakan Python Pada Tembang Macapat," *Journal Of Systems, Information Technology, And Electronics Engineering*, Vol. 50, No. 2, Pp. 50–54, 2021, [Online]. Available: Http://E-Journal.Ivet.Ac.Id/Index.Php/Jsitee

[4] M. Ferdiansyah, M. Ferdiansyah Hidayat, M. Ridho, And Z. Indra, "Implementasi Metode Caesar Chiper Dalam Kriptografi Untuk Keamanan Data Pesan," *Qistina Jurnal Multidisiplin Indonesia*, Vol. 3, No. 2, 2024.

[5] R. Gusmana, H. Haryansyah, And F. Fitria, "Implementasi Algoritma Affine Cipher Dan Caesar Cipher Dalam Mengamankan Data Teks," *Sebatik*, Vol. 26, No. 2, Pp. 517–524, Dec. 2022, Doi: 10.46984/Sebatik.V26i2.2084.

[6] R. M. T. Pasaribu, R. Dewi, And I. Parlina, "Implementasi Kombinasi Algoritma Rinjdael Dengan Caesar Cipher Pada Pengamanan Dokumen Digital," *Hello World Jurnal Ilmu Komputer*, Vol. 1, No. 1, Pp. 36–52, May 2022, Doi: 10.56211/Helloworld.V1i1.11.

[7] M. Tahboush, A. Hamdan, M. Klaib, M. Adawy, And F. Alzobi, "Detection Optimization Of Brute-Force Cyberattack Using Modified Caesar Cipher Algorithm Based On Binary Codes (Mcbc)," 2025. [Online]. Available: Www.Ijacsa.Thesai.Org

[8] B. S. Olanrewaju, V. O. Eguavoen, And O. A. Okunade, "A Modified Caesar Cipher Encryption Technique For Secured Information Sharing," *Fudma Journal Of Sciences*, Vol. 9, No. 5, Pp. 180–186, May 2025, Doi: 10.33003/Fjs-2025-0905-3494.

[9] R. A. Manurung, Sutarman, And S. Efendi, "Comparative Analysis Of The Performance Of Four Symmetric Algorithms On Digital File Security," *Jite (Journal Of Informatics And Telecommunication Engineering)*, Vol. 8, No. 2, Jan. 2025, Doi: 10.31289/Jite.V8i2.13978.

[10] M. Zulfikar, T. Imanuddin, And N. E. Prastyo, "Analisis Perbandingan Tingkat Kompleksitas Waktu Enkripsi Dan Tingkat Keamanan Enkripsi Pada Algoritalgoritma Kriptografi Rsa, Des, Aes," *Jurnal Siteba*, No. 2, 2023.

[11] M. K. Fauzi And A. Setiawan, "Implementasi Algoritma Vigenere Chiper Dan Caesar Chiper Untuk Pengamanan Password Dalam Penerimaan Siswa Baru," *Jid*, Vol. 2, No. 3, P. 1083, Sep. 2024, [Online]. Available: Http://Kti.Potensi-Utama.Ac.Id/Index.Php/Jid

[12] H. Bancin, M. A. Panjaitan, S. Putri, And A. B. Nasution, "Implementation Of Cryptography With The Caesar Cipher Method To Secure Data Files In Java Netbeans," *Jtecs : Jurnal Sistem Telekomunikasi Elektronika Sistem Kontrol Power Sistem Dan Komputer*, Vol. 3, No. 1, P. 79, Feb. 2023, Doi: 10.32503/Jtecs.V3i1.3210.

[13] M. B. Fajar, A. Wahid, G. Darma Dirawan, M. Syahid Nur Wahid, And A. Akram Nur Risal, "Python Dan Kriptografi: Edukasi Dan Pengabdian Untuk Masa Depan Yang Aman," *Jurnal Kreativa: Kemitraan Responsif Untuk Aksi Inovatif Dan Pengabdian Masyarakat*, Vol. 1, No. 2, 2024.

[14] Priyono, "Penerapan Algoritma Caesar Cipher Dan Algoritma Vigenere Cipher Dalam Pengamanan Pesan Teks," *Jurnal Riset Komputer (Jurikom)*, Vol. 3, No. 5, Pp. 351–356, Oct. 2020.

[15] V. Vebby, S. Ahmad, And L. Lhaura Van Fc, "Penerapan Algoritma Caesar Cipher Dalam Metode Kriptografi Klasik Pada Panic Button," *Zonasi: Jurnal Sistem Informasi*, Vol. 5, No. 1, Pp. 126–136, Feb. 2023, Doi: 10.31849/Zn.V5i1.13075.

[16] I. M. Yusup, C. Carudin, And I. Purnamasari, "Implementasi Algoritma Caesar Cipher Dan Steganografi Least Significant Bit Untuk File Dokumen," *Jurnal Teknik Informatika Dan Sistem Informasi*, Vol. 6, No. 3, Dec. 2020, Doi: 10.28932/Jutisi.V6i3.2817.

[17] S. Ahmad, "Penerapan Algoritma Caesar Cipher Dalam Metode Kriptografi Klasik Pada Panic Button (Studi Kasus Fasilkom Unilak)," Pekanbaru, 2022.

[18] D. S. Nugroho, M. R. Saputra, And A. Ramadhan, "Implementasi Algoritma Caesar Cipher Untuk Membuka Pesan (Decrypt) Menggunakan Bahasa Pemrograman Java," *Kohesi: Jurnal Multidisiplin Saintek*, Vol. 5, No. 9, 2024.

[19] Uci Julya Ningsih, Sophia Salsabila, Isniar Hutapea, Dewi Santika, And Indra Gunawan, "Pendekripsian Caesar Chiper Dengan Menggunakan Teknik-Teknik Kriptanalisis," *Jurnal Ilmu Komputer Dan Multimedia*, Vol. 1, No. 1, Pp. 11–15, Jun. 2024, Doi: 10.46510/Ilkomedia.V1i1.10.

Djtechno : Jurnal Teknologi Informasi
Vol. 6, No. 3, Desember 2025
E-ISSN: 2745-3758, P-ISSN : 2776-8546 DOI: 10.46576/djtechno

[20] Ratih Adinda Destari, "Implementasi Kriptografi Dengan Caesar Cipher Pada Fitur Pesan Di Whatsapp Untuk Keamanan," *Mars : Jurnal Teknik Mesin, Industri, Elektro Dan Ilmu Komputer*, Vol. 3, No. 4, Pp. 169–176, Jul. 2025, Doi: 10.61132/Mars.V3i4.978.

[21] M. R. Abu Jihad Plaza, R. Hartono, And S. Surya Intan, "Penerapan Kriptografi Caesar Chiper Pada Aplikasi Chatting Berbasis Local Area Network," 2021.

[22] N. Chafid And H. Soffiana, "Impelementasi Algoritma Kriptografi Klasik Caesar Untuk Rancang Bangun Aplikasi E-Voting Berbasis Web (Studi Kasus : Sman 10 Tangerang)."

[23] Y. Raka, A. Pratama, A. Faizin, A. Zulham, And F. Havy, "Implementasi Kombinasi Caesar Cipher Dan Polyalphabetic Cipher Untuk Keamanan Data Pribadi Pada E-Voting Dalam Pemilihan Ketua Humanika," *Jurnal Teknik Elektro Dan Informatika*, Vol. 20, No. 2, Pp. 1–11, Aug. 2025.

[24] A. W. Aranski, R. Wahdini, F. Anggraini, And F. Khairani, "Implementasi Algoritma Caesar Cipher Untuk Keamanan Data Anggota Perpustakaan Di Universitas Xyz," *Jurnal Siteba*, Vol. 2, No. 1, Pp. 34–41, 2023.