E-ISSN : 3048-3220 (Media Online)

Implementasi Algoritma RSA dan Modifikasi *Caesar Time ASCII Cipher* untuk Keamanan Pesan Teks

Eka Yuslita Dewi[1*], Annisa Nur Azizah[1]

[1,2] Matematika, Universitas Nurul Huda
[*]Korespondensi: E-mail: dewiekayuslita@gmail.com

Abstrak

Perlindungan terhadap kerahasiaan data teks sensitif pada jaringan publik sangat krusial guna meminimalisir risiko insiden pencurian informasi oleh pihak yang tidak berwenang. Metode konvensional seperti *Caesar cipher* memiliki kelemahan besar pada penggunaan kunci statis yang polanya mudah diprediksi melalui analisis frekuensi. Penelitian ini bertujuan untuk menguji model modifikasi keamanan pesan teks melalui integrasi skema asimetris algoritma RSA dengan *Caesar time* ASCII *cipher* yang berbasis pergeseran kunci dinamis. Metode penelitian dilakukan secara sistematis melalui perancangan model matematika dan simulasi eksperimental berupa perhitungan numerik manual menggunakan ruang karakter operasi Modulo 127. Hasil pengujian simulasi manual pada pesan "UNUHA_JAYA" menunjukkan bahwa pemanfaatan nilai komponen waktu (*timestamp*) eksekusi berhasil mengubah parameter pergeseran karakter menjadi dinamis, sehingga menghasilkan nilai kunci akhir dinamis sebesar 61. Pengamanan parameter kunci dinamis tersebut menggunakan kunci publik RSA ($e = 7$ dan $n = 143$) sukses menghasilkan *ciphertext* kunci bernilai 125. Kesimpulannya, kombinasi kedua metode ini terbukti valid secara matematis dalam meningkatkan kompleksitas keamanan data teks dan mampu merekonstruksi kembali *plaintext* secara utuh melalui proses dekripsi tanpa ada kerusakan karakter.

Kata kunci*: Algoritma RSA, Caesar Cipher, Kunci Dinamis, Modulo 127, Parameter Waktu*

PENDAHULUAN

Dalam ekosistem digital yang terus berkembang, perlindungan terhadap integritas dan kerahasiaan kumpulan data sensitif menjadi prioritas yang tidak dapat ditawar. Tanpa adanya mekanisme proteksi tambahan, data tersebut sangat rentan terhadap berbagai ancaman siber, salah satunya adalah insiden pencurian data oleh pihak yang tidak berwenang (Anwar et al., 2019). Mengingat tingginya transmisi informasi sensitif, kebocoran data pada infrastruktur jaringan publik menuntut adanya sistem proteksi end-to-end yang lebih mutakhir (Fitri & Nasution, 2024). Kriptografi hadir sebagai disiplin ilmu yang berfokus pada transformasi pesan asli (*plaintext*) menjadi bentuk tersandi (*ciphertext*) guna menjamin aspek kerahasiaan, integritas, dan keaslian informasi. Melalui teknik enkripsi ini, data yang dikirimkan melalui jaringan akan disamarkan sedemikian rupa sehingga meskipun data tersebut berhasil disadap, informasinya tetap tidak akan bisa dimengerti oleh pihak yang tidak berhak (Amin, 2016). Penerapan teknologi ini menjadi solusi

E-ISSN : 3048-3220 (Media Online)

krusial dalam meminimalisir risiko penyalahgunaan data pada komunikasi digital, di mana penggunaan skema enkripsi yang tepat terbukti mampu memproteksi pesan dari akses tidak sah sekaligus membangun kepercayaan dalam pertukaran informasi di jaringan publik (Setiawan, 2025). Kriptografi merupakan metode penyandian informasi secara rahasia melalui pemanfaatan fitur enkripsi data untuk menyamarkan pesan serta proses dekripsi untuk mengembalikan pesan ke bentuk semula (Nanda et al., 2023).

Sebagai salah satu jenis kriptografi klasik *Caesar cipher* merupakan salah satu metode kriptografi klasik yang digunakan untuk mengamankan pesan dengan teknik substitusi huruf berdasarkan pergeseran tertentu dalam alfabet (Wasis, 2026). Dalam posisinya algoritma enkripsi yang umum digunakan, metode klasik ini dikenal sangat sederhana namun tetap dinilai cukup *powerful* dalam memberikan proteksi dasar pada data teks (Ningrum et al., 2023). Namun, *Caesar cipher* memiliki kelemahan besar pada kuncinya yang bersifat statis, sehingga pola pergeserannya sangat mudah diprediksi dan dipecahkan. Berdasarkan permasalahan yang diangkat dalam penelitian (Hasibuan, 2024), metode substitusi sederhana ini dinilai sangat rentan terhadap serangan analisis frekuensi maupun eksploitasi keamanan lainnya, sehingga *ciphertext* yang dihasilkan tetap memiliki kerentanan tinggi yang mudah ditembus menggunakan komputasi modern. Dalam penelitian (Sinaga & Frangk, 2023), upaya mengatasi kelemahan ini dilakukan dengan menambahkan *key* untuk menghilangkan pola linier pada metode klasik, namun sistem tersebut dilaporkan tetap rentan terhadap serangan analisis frekuensi karena kuncinya belum berubah secara dinamis di setiap waktu pengiriman. Oleh karena itu, penelitian ini menawarkan *state-of-the-art* berupa pengembangan kunci dinamis berbasis parameter waktu (*timestamp*) aktual untuk mengamankan nilai pergeseran. Nilai pergeseran dinamis tersebut dihasilkan secara otomatis setiap detiknya dan langsung diproteksi secara rahasia menggunakan algoritma asimetris RSA, sehingga tidak dapat diketahui oleh pihak ketiga.

Urgensi pengamanan dalam pertukaran informasi digital memicu kebutuhan akan sistem keamanan data yang lebih andal dan efisien, salah satunya melalui penerapan metode *hybrid encryption* (Maulana et al., 2025). Dalam upaya mengatasi berbagai keterbatasan pada skema tunggal, banyak penelitian merekomendasikan penggunaan *hybrid cryptosystem* yang mengombinasikan pendekatan simetris dan asimetris untuk mencapai keseimbangan optimal antara efisiensi proses dan keamanan data digital (Zuna et al., 2026). Salah satu algoritma asimetris yang sering diintegrasikan yaitu algoritma RSA, RSA merupakan metode kriptografi yang menggunakan kunci berbeda untuk proses enkripsi dan dekripsi (Rahayu et al., 2024). Tingkat keamanan dari algoritma ini bertumpu pada sulitnya memfaktorkan bilangan besar menjadi komponen bilangan prima, di mana dalam penerapannya melibatkan tiga tahapan utama, yaitu proses pembangkitan kunci publik dan privat, proses enkripsi, serta proses dekripsi (Dairi et al., 2023). Berdasarkan studi kasus pengamanan data teks, penerapan sistem hibrida terbukti sangat efektif dan mampu memberikan perlindungan optimal dalam praktik nyata (H. Putri et al.,

E-ISSN : 3048-3220 (Media Online)

2025). Penjelasan proses matematis dan implementasi yang diprogram dalam arsitektur ini menunjukkan bahwa pendekatan hibrida tersebut mampu mengamankan sekaligus memulihkan data teks asli secara utuh tanpa ada kerusakan (P. A. W. S. Dewi & Astawa, 2025). Menindaklanjuti kajian pada penelitian (N. A. Putri et al., 2024), integrasi antara *Caesar cipher* dan RSA telah dieksplorasi melalui metode enkripsi bertahap menggunakan kunci pergeseran. Meskipun skema dua lapis ini terbukti sukses menyandikan pesan asli menjadi *ciphertext* acak, metode konvensional tersebut dilaporkan masih memiliki keterbatasan dalam aspek efisiensi sistem.

Guna meningkatkan keamanan, penelitian ini mengembangkan modifikasi *Caesar time* ASCII *cipher*. Teknik ini menggantikan kunci statis dengan penggabungan nilai ASCII dan waktu (*timestamp*) sebagai variabel pergeseran yang dinamis (E. Y. Dewi et al., 2025). ASCII sendiri merupakan standar internasional dalam sistem pengkodean karakter yang berfungsi untuk mentransformasikan teks, angka, maupun simbol ke dalam bentuk nilai numerik desimal agar dapat dikenali dan diproses oleh sistem komputer. Penelitian ini memodifikasi *Caesar cipher* dengan mengintegrasikan nilai waktu (*timestamp*) dan sistem pengkodean ASCII sebagai variabel pergeseran dinamis. Penggunaan operasi modulo 127 memungkinkan sistem menangani cakupan karakter yang lebih luas, termasuk simbol dan angka. Untuk menjaga kerahasiaan parameter kunci dinamis tersebut, algoritma RSA diintegrasikan guna menciptakan sistem enkripsi hibrida yang tangguh namun tetap efisien.

METODE

Operasi Modulo

Penggunaan operasi modulo dalam algoritma kriptografi terbukti efektif untuk memperluas fleksibilitas dan jangkauan proteksi data teks. Jika pada skema klasiknya operasi modulo umumnya terbatas pada nilai 26 untuk mengakomodasi huruf alfabet saja, perluasan fungsi menggunakan operasi modulo 256 memungkinkan sistem untuk memproses seluruh cakupan karakter ASCII (Sutoyo et al., 2024). Penelitian ini menerapkan operasi modulo 127, alih-alih modulo 256, untuk memastikan transformasi data karakter tetap berada dalam rentang nilai desimal yang valid. Nilai 127 dipilih karena lebih optimal dalam mencakup seluruh karakter standar ASCII yang *printable* (huruf, angka, spasi, dan simbol standar) yang menjadi fokus utama pengamanan pesan teks.

Caesar Cipher

Sebagai salah satu jenis kriptografi klasik, *Caesar cipher* pernah digunakan secara luas dengan prinsip kerja yang mengandalkan teknik substitusi atau perpindahan (Febrianingsih & Hafiz, 2019). *Caesar cipher* merupakan suatu teknik konvensional serta menjadi landasan paling mendasar di dalam disiplin ilmu penyandian pesan (Limbong & Silitonga, 2017). Melalui transformasi teks asli menjadi pesan rahasia berbasis pergeseran kunci, mekanisme pada algoritma ini memungkinkan informasi yang dikirimkan dapat terjaga keamanannya (Pratiwi et al., 2022). Algoritma ini bekerja dengan

E-ISSN : 3048-3220 (Media Online)

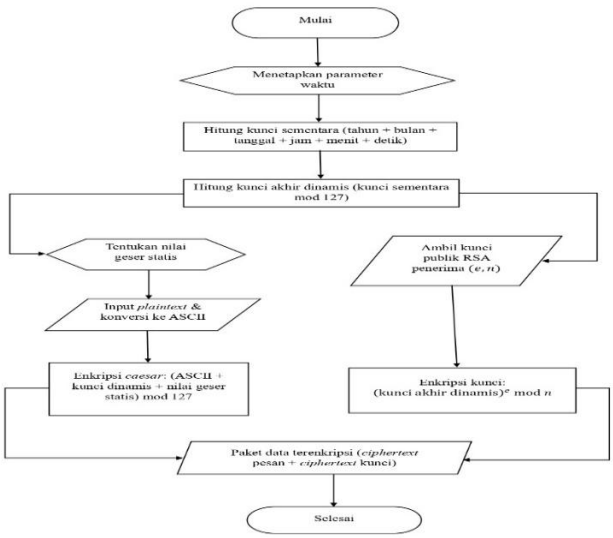
mensubstitusi setiap karakter *plaintext* menggunakan pergeseran posisi tertentu berdasarkan nilai kunci numerik. Proses enkripsi dan dekripsi dilakukan dengan menerapkan pergeseran alfabet yang tetap, sehingga pesan hanya dapat dipulihkan jika penerima menggunakan nilai kunci yang sama.

Algoritma RSA

RSA merupakan salah satu algoritma kriptografi asimetris paling populer yang dikembangkan pada tahun 1976 oleh tiga peneliti dari MIT, yaitu Ron Rivest, Adi Shamir, dan Leonard Adleman. Keunggulan utama dari teknik ini terletak pada penggunaan sepasang kunci yang berbeda untuk mengolah data, sehingga proses enkripsi dan dekripsi tidak menggunakan kunci yang sama (Rahajoeningroem & Aria, 2011). Dalam sistem ini, kunci publik digunakan secara terbuka untuk mengamankan atau menyandikan pesan teks, sedangkan kunci privat harus disimpan secara rahasia karena hanya kunci inilah yang dapat digunakan untuk membuka atau mendekripsi kembali pesan tersebut ke bentuk aslinya.

Tahapan Rangkain Penelitian

Penelitian ini dilakukan secara sistematis melalui beberapa tahapan yang berkesinambungan. Proses diawali dengan studi literatur dan identifikasi kelemahan metode *Caesar time ASCII cipher*, dilanjutkan dengan perancangan model integrasi antara metode tersebut dengan algoritma RSA. Tahap berikutnya adalah simulasi eksperimental menggunakan perhitungan numerik manual untuk menguji proses enkripsi dan dekripsi. Seluruh rangkaian diakhiri dengan analisis ketahanan hasil enkripsi untuk mengevaluasi efektivitas lapisan keamanan RSA dalam melindungi data. Alur kerja sistem dari *plaintext* hingga menjadi *ciphertext* disajikan pada Gambar 1.



Gambar 1. Diagram Alir Proses Enkripsi Menggunakan Algoritma Caesar Time ASCII dan RSA

 E-ISSN : 3048-3220 (Media Online)

HASIL PENELITIAN DAN PEMBAHASAN

Perumusan Algoritma *Caesar*

Secara matematis, proses transformasi data pada algoritma konvensional ini ditentukan oleh nilai pergeseran posisi karakter. Proses enkripsi untuk menghasilkan pesan tersandi menggunakan persamaan:

$$\mathrm{C} = (\mathrm{P} + \mathrm{K}) \bmod \mathrm{N} \qquad (1)$$

Sementara itu, proses pemulihan data atau dekripsi dilakukan dengan menerapkan prinsip kebalikan dari operasi enkripsi, yang dirumuskan melalui persamaan:

$$\mathrm{P} = (\mathrm{C} - \mathrm{K}) \bmod \mathrm{N} \qquad (2)$$

Di mana keterangan untuk masing-masing variabel adalah sebagai berikut:
C = karakter hasil enkripsi ($ciphertext$)
P = karakter pesan asli ($plaintext$)
K = nilai kunci pergeseran yang digunakan
N = jumlah keseluruhan karakter di dalam ruang alfabet yang diterapkan

Perumusan Algoritma RSA

Besaran-besaran yang digunakan pada algoritma RSA:
Algoritma RSA didasarkan pada teorema euler

$$a^{\phi(n)} \equiv 1(mod\ n) \qquad (3)$$

Yang dalam hal ini,

1. a harus relative prima terhadap n
2. $\phi(n) = n\left(1 - \frac{1}{P_1}\right)\left(1 - -\frac{1}{P_2}\right) \ldots \left(1 - -\frac{1}{P_r}\right)$, yang dalam hal ini $P_1, P_2, \ldots, P_r$ adalah factor prima dari n.

Algoritma Pembangkit Kunci

Tahap-tahap operasi matematisnya adalah sebagai berikut:

1. $\phi(n)$ adalah fungsi yang menentukan berapa banyak dari bilangan-bilangan 1, 2, 3, …, n yang relative prima terhadap n.
2. Berdasarkan sifat $a^k \equiv b^k \pmod{n}$ untuk k bilangan bulat ≥ 1, maka persamaan (3) dapat ditulis menjadi
   $$a^{k\phi(n)} \equiv 1^k(mod\ n)$$
   Atau
   $$a^{k\phi(n)} \equiv 1(mod\ n) \qquad (4)$$
3. Bila a diganti dengan m, maka persamaan (2) menjadi
   $$m^{k\phi(n)} \equiv 1(mod\ n) \qquad (5)$$
4. Berdasarkan sifat $ac \equiv bc\ (mod\ n)$, maka bila persamaan (5) dikali dengan m menjadi:
   $$m^{k\phi(n)+1} \equiv m(mod\ n) \qquad (6)$$
   Yang dalam hal ini m relative prima terhadap n.
5. Misalkan e dan d dipilih sedemikian sehingga
   $$e \cdot d \equiv 1(mod\phi(n)) \qquad (7)$$
   Atau

E-ISSN : 3048-3220 (Media Online)

$$e \cdot d = k\phi(n) + 1 \quad (8)$$

6. Sulihkan (8) ke dalam persamaan (6) menjadi:
$$m^{e \cdot d} \equiv m(mod\ n) \quad (9)$$
7. Persamaan (9) dapat ditulis Kembali menjadi
$$(m^e)^d \equiv m(mod\ n) \quad (10)$$
Yang artinya, perpangkatan m dengan e diikuti dengan perpangkatan dengan d menghasilkan Kembali m semula.
8. Berdasarkan persamaan (10), maka enkripsi dan dekripsi dirumuskan sebagai berikut:
$$E_e(m) = c \equiv m^e mod\ n \quad (11)$$
$$D_d(c) = c \equiv m^d mod\ n \quad (12)$$
9. Karena $e \cdot d = d \cdot e$, maka enkripsi diikuti dengan dekripsi ekivalen dengan dekripsi diikuti enkripsi:
$$D_d\big(E_e(m)\big) = E_e(D_d(m)) \equiv m^d mod\ n (13)$$
10. Oleh karena $m^d mod\ n \equiv (m + jn)^d mod\ n$ untuk sembarang bilangan bulat j, maka tiap *plaintext* $m, m+n, m+2n, \ldots,$ menghasilkan *ciphertext* yang sama. Dengan kata lain, transformasinya dari banyak ke satu. Agar transformasinya satu ke satu, maka m harus dibatasi dalam himpunan $\{0, 1, 2, \ldots, n-1\}$ sehingga enkripsi dan dekripsi tetap benar seperti pada persamaan (10) dan (11) Proses pembangkitan pasangan kunci RSA. Untuk menghasilkan sepasang kunci yang akan digunakan dalam proses kriptografi, langkah-langkah sistematis yang harus dilakukan adalah sebagai berikut:
11. Pilih dua buah bilangan prima sembarang, p dan q.
12. Hitung $n = p \cdot q$ (sebaiknya $p \neq q, sebab\ jika\ p = q\ maka\ n = p^2$ sehingga p dapat diperoleh dengan menarik akar pangkat dua dari n).
13. Hitung $\phi(n) = (p-1)(q-1)$.
14. Pilih kunci public, e, yang relative prima terhadap $\phi(n)$.
15. Bangkitkan kunci privat dengan menggunakan persamaan (7), yaitu $e \cdot d \equiv 1\big(mod\phi(n)\big)$. Perhatikan bahwa $e \cdot d \equiv 1(mod\phi(n))$ ekivalen dengan $e \cdot d = 1 + k\phi(n)$, sehingga d dapat dihitung dengan (14)
$$d = \frac{1 + k\phi(n)}{e}$$
Akan terdapat bilangan bulat k yang memberikan bilangan bulat d.
Hasil dari algoritma di atas
Kunci public: pasangan (e, n)
Kunci privat: pasangan (d, n)

Dengan catatan: n tidak bersifat rahasia, namum ia diperlukan pada perhitungan enkripsi/dekripsi.

Algoritma Enkripsi

1. Ambil kunci public penerima pesan e, dan modulus n.
2. Nyatakan *plaintext* m menjaadi blok-blok $m_1, m_2, \ldots,$ sedemikian sehingga setiap blok mempresentasikan nilai di dalam selang $[0, n-1]$.
3. Setiap blok m_i dienkripsi menjadi blok c_i dengan rumus
$$c_i = m_i^e\ mod\ n$$

E-ISSN : 3048-3220 (Media Online)

Algoritma Dekripsi

Setiap blok *ciphertext* c_i didekripsi Kembali menjadi blok m_i dengan rumus

$$m_i = c_i^d \ mod \ n.$$

Perumusan Algoritma *Caesar Time* ASCII dan RSA

Penelitian ini memodifikasi *Caesar cipher* menjadi *Caesar time ASCII cipher* dengan membatasi ruang karakter melalui operasi modulo 127, yang mencakup seluruh karakter standar ASCII yang *printable*. Berbeda dengan metode konvensional yang memiliki ruang kunci statis terbatas (0-25), sistem ini menggunakan parameter waktu (*timestamp*) sebagai variabel pergeseran dinamis. Kunci diperoleh dengan menjumlahkan komponen waktu (tahun, bulan, tanggal, jam, menit, detik) dan melakukan operasi modulo 127. Integrasi ini secara signifikan meningkatkan kompleksitas *key space* dan menciptakan sifat penyandian yang dinamis, sehingga distribusi karakter menjadi lebih acak dan lebih tahan terhadap serangan analisis frekuensi konvensional.

Analisis Batasan Karakter pada Operasi Modulo 127

Penggunaan operasi modulo 127 secara teori menghasilkan sisa bagi bernilai 0–31, yang dalam standar ASCII merupakan karakter yang tidak dapat dicetak (*non-printable characters*). Untuk mengatasi kendala tersebut, *ciphertext* dalam penelitian ini disajikan secara konsisten dalam bentuk deret angka desimal di dalam kurung siku. Melalui pendekatan ini, seluruh data hasil perhitungan tetap aman, terbaca dengan jelas, dan terhindar dari risiko eror pembacaan karakter pada aplikasi teks biasa.

Evaluasi Keterbatasan Keamanan Algoritma

Secara teoritis, modifikasi *Caesar time ASCII* ini memiliki keterbatasan ruang kunci (*key space*) yang dibatasi oleh fungsi modulo 127, sehingga hanya menghasilkan 127 kemungkinan pergeseran. Kondisi ini rentan terhadap serangan *brute force* skala mikro jika penyerang berhasil mengetahui waktu (*timestamp*) pengiriman data. Oleh karena itu, model ini difungsikan sebagai langkah awal (*proof of concept*). Untuk implementasi praktis yang lebih aman di masa depan, dibutuhkan pengembangan dengan ruang operasi yang jauh lebih besar seperti penggunaan Modulo 2^{128} guna menghindari serangan tebakan kunci (*key-guessing attack*).

Algoritma Enkripsi:

1. Proses enkripsi diawali dengan menetapkan parameter waktu (*timestamp*) yang ditentukan secara bebas sebagai basis data uji coba, yang meliputi komponen tahun, bulan, tanggal, jam, menit, dan detik. Seluruh komponen variabel waktu yang diperoleh kemudian diakumulasikan melalui operasi penjumlahan untuk menghasilkan nilai kunci sementara.
2. Nilai kunci akhir dinamis diperoleh dengan menghitung sisa hasil bagi (operasi modulo) dari nilai kunci sementara terhadap bilangan 127,

E-ISSN : 3048-3220 (Media Online)

yang selanjutnya berfungsi sebagai parameter pergeseran dinamis berbasis waktu.

3. Menentukan nilai geser konvensional (K) yang bersifat statis sebagai parameter ketetapan algoritma *Caesar cipher* di dalam sistem.
4. Karakter pada pesan asli (*plaintext*) dimasukkan sebagai input utama, lalu dikonversi ke dalam bentuk representasi numerik desimal berdasarkan standar kode ASCII.
5. Setiap nilai numerik ASCII dari karakter pesan tersebut kemudian dijumlahkandengan nilai kunci akhir dinamis dan nilai geser *Caesar* statis, lalu dihitung menggunakan operasi modulo 127 untuk menghasilkan nilai karakter tersandi (*ciphertext*).
6. (Proses RSA) Untuk mengamankan kunci akhir dinamis selama proses transmisi data, sistem mengambil parameter kunci publik milik penerima, yaitu eksponen enkripsi (e) dan nilai modulus (n), di mana nilai modulus (n) dihitung dari perkalian dua bilangan prima rahasia ($n = p.q$)
7. (Proses RSA) Nilai kunci akhir dinamis dinyatakan sebagai blok pesan (m_i) untuk kemudian dienkripsi menjadi blok *ciphertext* kunci (c_i) menggunakan fungsi eksponensial modular dengan rumus:
$$c_i = m_i^e \ mod \ n$$
8. Hasil akhir dari proses enkripsi menghasilkan sebuah paket data terintegrasi yang menggabungkan teks tersandi (*ciphertext*) pesan beserta *ciphertext* kunci (c_i) yang telah terproteksi oleh algoritma RSA, untuk kemudian dikirimkan kepada pihak penerima.

Algoritma Dekripsi

1. Proses dekripsi diinisialisasi ketika pihak penerima telah mendapatkan paket data terintegrasi yang memuat teks tersandi (*ciphertext*) pesan dan cipherteks kunci (c_i).
2. Proses RSA penerima mengekstrak *ciphertext* kunci (c_i) dari paket data, kemudian mendekripsinya kembali menjadi blok kunci asli (m_i) menggunakan parameter kunci privat miliknya, yaitu eksponen dekripsi (d) dan modulus (n), melalui rumus:
$$m_i = c_i^d \ mod \ n$$
3. Berdasarkan nilai blok kunci (m_i) yang telah yang telah dipulihkan, sistem memperoleh kembali nilai kunci akhir dinamis berbasis waktu yang digunakan pada saat enkripsi.
4. Sistem di sisi penerima secara otomatis memanggil nilai geser *Caesar* statis (K) yang telah tertanam secara konstan sebagai parameter internal pada sistem dekripsi.
5. Setiap karakter pada teks tersandi (*ciphertext*) pesan ditransformasikan ke dalam representasi numerik desimal standar kode ASCII.
6. Nilai desimal ASCII dari karakter tersandi tersebut kemudian dikurangi dengan nilai kunci akhir dinamis berbasis waktu dan nilai geser *Caesar* statis yang telah disepakati.

 E-ISSN : 3048-3220 (Media Online)

7. Hasil pengurangan nilai tersebut selanjutnya diproses menggunakan operasi matematis Modulo 127 untuk mengembalikan nilai desimal asli ke dalam rentang karakter standar ASCII yang valid, lalu dikonversi kembali ke bentuk tekstual untuk merekonstruksi pesan asli (*plaintext*) secara utuh.

Implementasi Metode Kriptografi Caesar Time ASCII dan RSA

Untuk mensimulasikan langkah-langkah dalam metode *Caesar time* ASCII *cipher* dan RSA ini, digunakan sampel data uji coba dengan parameter yang ditetapkan sebagai berikut:

Time: Tanggal 2026-06-07 09, jam 09:30:15
K (nilai geser statis) $= 3$
p (bilangan prima pertama) $= 11$
q (bilangan prima kedua) $= 13$
e(kunci enkripsi) $= 7$ (Kunci publik yang dipilih karena memenuhi syarat yang saling prima dengan nilai $\phi(n) = (p-1)(q-1) = 120$, 7 dan 120 terbukti saling prima)
$n = p \cdot q = 11 \cdot 13 = 143$
$d = \dfrac{1 + k\phi(n)}{e} = \dfrac{1 + k\ 120}{7} = 103$, dimana k harus bilangan bulat

Catatan Simulasi Kunci RSA

Perlu ditegaskan bahwa pemilihan nilai bilangan prima berskala kecil ($p = 11$ dan $q = 13$) dalam penelitian ini murni bertujuan sebagai *toy example* (contoh simulasi) untuk mempermudah visualisasi perhitungan matematis secara manual. Pada implementasi sistem keamanan digital yang sebenarnya, parameter tersebut sangat tidak aman karena mudah ditembus melalui faktorisasi prima secara cepat. Untuk menjamin keamanan data pada sistem nyata, ukuran panjang kunci RSA yang wajib diaplikasikan minimal adalah 2048-bit guna menangkal risiko serangan komputasi modern.

Proses Enkripsi

1. Pengambilan Parameter Waktu (*Timestamp*)
   Proses diawali dengan menentukan sampel data waktu (*timestamp*) secara bebas sebagai representasi saat simulasi enkripsi dilakukan. Dalam pengujian manual ini, waktu eksekusi diasumsikan ditetapkan pada: 2026-06-07 09:30:15. Akumulasi kunci sementara seluruh komponen waktu tersebut dijumlahkan.
   $$\text{Kunci Sementara} = 2026 + 6 + 7 + 9 + 30 + 15 = 2093$$
2. Penghitungan Kunci Akhir Dinamis
   Nilai kunci sementara kemudian di-modulo dengan bilangan 127 untuk membatasi rentang karakter sesuai standar ASCII yang digunakan:
   $$\text{Kunci Akhir Dinamis} = 2093 \pmod{127} = 61$$
   Maka, nilai pergeseran dinamis berbasis waktu yang diperoleh adalah 61, Nilai ini selanjutnya akan dinyatakan sebagai simbol m_i.
3. Penentuan Nilai Geser Statis *Caesar*

 E-ISSN : 3048-3220 (Media Online)

Sistem menentukan nilai pergeseran konvensional statis yang telah ditetapkan di dalam program. Diasumsikan nilai geser statis $(K) = 3$.

4. Konversi ASCII dan Proses Enkripsi Karakter
   Setiap karakter dari pesan **UNUHA_JAYA** dikonversi ke desimal ASCII, lalu dienkripsi dengan rumus:
   $Ciphertext =$ (Nilai ASCII $+ m_i + K$) (mod 127)$Ciphertext =$ (Nilai ASCII $+ 61 + 3$) (mod 127) = (Nilai ASCII $+ 64$) (mod 127)

Tabel 1. Simulasi Perhitungan Enkripsi Karakter pada Pesan Teks

Karakter (*plaintext*)	Nilai Desimal ASCII	Proses Matematika: (Nilai ASCII $+ 64$) (mod 127)
U	85	$(85 + 64) = 149 \pmod{127} = 22$
N	78	$(78 + 64) = 142 \pmod{127} = 15$
U	85	$(85 + 64) = 149 \pmod{127} = 22$
H	72	$(72 + 64) = 136 \pmod{127} = 9$
A	65	$(65 + 64) = 129 \pmod{127} = 2$
_	95	$(95 + 64) = 159 \pmod{127} = 32$
J	74	$(74 + 64) = 138 \pmod{127} = 11$
A	65	$(65 + 64) = 129 \pmod{127} = 2$
Y	89	$(89 + 64) = 153 \pmod{127} = 26$
A	65	$(65 + 64) = 129 \pmod{127} = 2$

Berdasarkan proses enkripsi, diperoleh nilai numerik ciphertext pesan sebagai berikut: [22, 15, 22, 9, 2, 32, 11, 2, 26, 2].

5. Pengamanan Kunci Akhir Dinamis dengan RSA
   Untuk melindungi nilai kunci dinamis $(m_i = 61)$ selama pengiriman, dilakukan enkripsi asimetris menggunakan kunci publik penerima. Diasumsikan parameter kunci publik penerima adalah eksponen enkripsi $e = 7$ dan nilai modulus $n = 143$. Dengan menggunakan rumus:

$$c_i = m_i^e \; mod \; n$$
$$c_i = 61^7 \; mod \; 143$$
$$c_i = 125$$

6. Hasil Akhir dari Proses Enkripsi
   Proses enkripsi selesai dan menghasilkan satu paket data terintegrasi yang siap ditransmisikan ke penerima pesan, yang terdiri dari:
   1. Teks Tersandi (*ciphertext* pesan): [22, 15, 22, 9, 2, 32, 11, 2, 26, 2].
   2. Teks Tersandi Kunci (*ciphertext* kunci (c_i)): 125

E-ISSN : 3048-3220 (Media Online)

Proses Dekripsi

1. Penerimaan Data Terintegrasi
   Proses dekripsi diawali ketika pihak penerima mendapatkan paket data terintegrasi hasil pengiriman. Paket data tersebut memuat teks tersandi (*ciphertext*) pesan yaitu: [22, 15, 22, 9, 2, 32, 11, 2, 26, 2] dan ciphertext kunci (c_i) bernilai 125.
2. Dekripsi *Ciphertext* Kunci Menggunakan RSA
   Sistem penerima mengekstrak nilai *ciphertext* kunci ($c_i = 125$) dari data tersebut, kemudian mendekripsinya kembali menjadi blok kunci asli (m_i) menggunakan parameter kunci privat milik penerima, yaitu eksponen dekripsi ($d = 103$) dan modulus ($n = 143$) melalui perhitungan eksponensial modular:
   $$m_i = c_i^d \ mod \ n$$
   $$m_i = 125^{103} \ mod \ 143$$
   $$m_i = 61$$
3. Pemulihan Nilai Kunci Akhir Dinamis
   Berdasarkan nilai blok kunci (m_i) yang telah dipulihkan pada langkah sebelumnya, sistem berhasil memperoleh kembali nilai kunci akhir dinamis berbasis waktu yang digunakan pada saat enkripsi, yaiu 61.
4. Nilai Geser *Caesar* Statis
   Sistem di sisi penerima secara otomatis memanggil nilai geser *Caesar* statis (K) yang telah tertanam secara konstan sebagai parameter internal pada sistem dekripsi, yaitu (K) = 3.
5. Transformasi Karakter *Ciphertext* ke Desimal ASCII
   Setiap karakter pada teks tersandi (*ciphertext*) pesan ditransformasikan ke dalam representasi numerik desimal standar kode ASCII. Teks Tersandi (*ciphertext* pesan): [22, 15, 22, 9, 2, 32, 11, 2, 26, 2].
6. Pengurangan Nilai Desimal Bersama Kunci Dinamis dan Statis
   Nilai desimal ASCII dari karakter tersandi tersebut kemudian dikurangi dengan nilai kunci akhir dinamis berbasis waktu $m_i = 61$, dan nilai geser *Caesar* statis (K) = 3 yang telah di sepakati, atau secara matematis dikurangkan dengan nilai 64 ($Ciphertext - 64$).
7. Operasi Modulo 127 dan Rekonstruksi Pesan Asli (*Plaintext*)
   Hasil pengurangan nilai tersebut selanjutnya diproses menggunakan operasi matematis Modulo 127 untuk mengembalikan nilai desimal asli ke dalam rentang karakter standar ASCII yang valid, lalu dikonversi kembali ke bentuk tekstual untuk merekonstruksi pesan asli (*plaintext*) secara utuh.

Tabel 2. Simulasi Perhitungan Dekripsi Karakter pada Pesan Teks

 E-ISSN : 3048-3220 (Media Online)

Karakter (Cipertext)	Proses Matematika: (Nilai ASCII − 64) (mod 127)	Karakter (Plaintext)
22	$(22-64) = -42 \pmod{127} = 85$	U
15	$(15-64) = -49 \pmod{127} = 78$	N
22	$(22-64) = -42 \pmod{127} = 85$	U
9	$(9-64) = -55 \pmod{127} = 72$	H
2	$(2-64) = -62 \pmod{127} = 65$	A
32	$(32-64) = -32 \pmod{127} = 95$	_
11	$(11-64) = -53 \pmod{127} = 74$	J
2	$(2-64) = -62 \pmod{127} = 65$	A
26	$(26-64) = -38 \pmod{127} = 89$	Y
2	$(2-64) = -62 \pmod{127} = 65$	A

KESIMPULAN DAN SARAN

Penelitian ini berhasil membuktikan bahwa integrasi *Caesar time ASCII cipher* dan algoritma RSA efektif meningkatkan keamanan pesan teks. Modifikasi pergeseran karakter berbasis *timestamp* dan modulo 127 menciptakan kunci dinamis yang mempersulit analisis frekuensi, sementara algoritma RSA mampu memproteksi parameter kunci dengan akurat. Pengujian menunjukkan bahwa sistem ini valid secara matematis dalam meningkatkan kompleksitas *key space* dan menjaga integritas pesan selama proses enkripsi maupun dekripsi.

Untuk pengembangan selanjutnya, disarankan melakukan pengujian dengan volume data yang lebih besar dan variasi karakter yang lebih kompleks untuk mengukur batas ketahanan metode ini. Selain itu, implementasi model perhitungan manual ini perlu ditransisikan ke dalam bentuk aplikasi digital atau bahasa pemrograman agar proses enkripsi dan dekripsi dapat dieksekusi secara otomatis, efisien, dan lebih cepat dalam performa komputasi.

REFERENSI

Amin, M. M. (2016). Implementasi kriptografi klasik pada komunikasi berbasis teks .*Jurnal Pseudocode*, *3*(2), 127196.

Anwar, B., Nugroho, N. B., Prayudha, J., & Azanuddin, A. (2019). Implementasi Algoritma RSA Terhadap Keamanan Data Simpan Pinjam. *Jurnal SAINTIKOM (Jurnal Sains Manajemen Informatika Dan Komputer)*, *18*(1), 30–34. https://doi.org/10.53513/jis.v18i1.100

Dairi, M. S., Asih, M. S., & Khairunnisa. (2023). Implementasi Algoritma Kriptografi RSA Dalam Aplikasi Sistem Informasi Perpustakaaan. *Jurnal Ilmu Komputer Dan Sistem Informasi*, *2*(1), 98–107. https://doi.org/10.70340/jirsi.v2i1.44

Dewi, E. Y., Sundawa, D. A., Hermansyah, B., & Azizah, A. N. (2025). Caesar time ascii cipher (caesar cipher versi waktu + ascii). *Seminar Nasional Pendidikan Matematika (SNPM)*, *1*, 576–582.

Dewi, P. A. W. S., & Astawa, I. G. S. (2025). Implementasi Sistem Kriptografi Hybrid RSA dan DES untuk Pengamanan Data Teks. *Jurnal Nasional Teknologi Informasi Dan Aplikasinya*, *4*(1), 47–52. https://doi.org/10.24843/JNATIA.2025.v04.i01.p06

Febrianingsih, R., & Hafiz, A. (2019). Implementasi kriptografi berbasis caesar chiper untuk keamanan data. *Jurnal Informasi dan Komputer*, *9*(2), 81–86. https://doi.org/10.35959/jik.v7i2.163

Fitri, M. A., & Nasution, M. I. P. (2024). Taktik Canggih untuk Memastikan Keamanan Data Perusahaan dan Mengatasi Ancaman Kebocoran Data di Masa Depan. *Journal of Sharia Economics Scholar (JoSES)*, *2*(2). https://doi.org/10.5281/zenodo.12608875

Hasibuan, A. Z. (2024). Implementasi Kriptografi Hybrid Menggunakan Caesar Cipher dan Affine Cipher Untuk Kemanan Teks. *Jurnal Rekayasa Sistem (JUREKSI)*, *2*(3 A), 2240–2250.

Limbong, T., & Silitonga, P. (2017). Testing the Classic Caesar Cipher Cryptography using of Matlab. *International Journal of Engineering Research & Technology (IJERT)*, *6*, 175–178. https://doi.org/10.17605/OSF.IO/PEMA5

Maulana, H., Tahir, M., Farrohah, N., Maulana, F., & Aprilianyani, R. R. (2025). PERANCANGAN SISTEM KEAMANAN FILE MENGGUNAKAN HYBRID ENCRYPTION UNTUK PERLINDUNGAN DATA. *Jurnal RESTIKOM : Riset Teknik Informatika Dan Komputer*, *7*(1), 87–96. https://doi.org/10.52005/restikom.v7i1.420

Nanda, N. A., Silalahi, S. M. S., Fatricia Nasution, D., Sari, M., & Gunawan, I. (2023). Kriptografi dan Penerapannya Dalam Sistem Keamanan Data. *Jurnal Media Informatika*, *4*(2), 90–93. https://doi.org/10.55338/jumin.v4i2.428

Ningrum, D. F., Tahir, M., Puspitasari, W. D. A., Permatasari, E., Rofiq, F. R., Hidayati, M., Sahroh, F., & Setiawan, A. (2023). Implementasi Keamanan Data menggunakan Kriptografi Caesar Chiper. *Jurnal Adijaya Multidisplin*, *1*(02), 388–396.

Pratiwi, R., Utami, L. C., Sakti, R. B., & Triase. (2022). Perancangan Keamanan Data Pesan Dengan Menggunakan Metode Kriptografi Caesar Cipher. *Bulletin of Information Technology (BIT)*, *3*(4), 367–373. https://doi.org/10.47065/bit.v3i4.420

Putri, H., Virna, L., Febrianti, T., & Sutabri, T. (2025). Pengamanan Data Transmisi Aplikasi Web Menggunakan Algoritma Kriptografi RSA: Studi Kasus dan Analisis. *Jurnal Manajemen Informatika & Teknologi*, *5*(1), 153–170. https://doi.org/10.51903/rdbsne23

Putri, N. A., Hesti, E., & Aryanti, A. (2024). Pengamanan Data Nilai Mahasiswa Menggunakan Algoritma Caesar Chiper dan RSA Berbasis Web. *Jurnal RESISTOR (Rekayasa Sistem Komputer)*, *7*(2), 61–70. https://doi.org/10.31598/jurnalresistor.v7i2.1645

Rahajoeningroem, T., & Aria, M. (2011). Studi dan implementasi algoritma rsa untuk pengamanan data transkrip akademik mahasiswa. *Majalah Ilmiah UNIKOM*, *Volume*. http://jurnal.unikom.ac.id/jurnal/studi-dan-implementasi.1f

Rahayu, A., Ardana, A. P., Pramudhita, C., Syafitri, D., & Sirega, R. Z. (2024). Perbandingan Algoritma RSA dengan Algoritma Blowfish Pada Perancangan Aplikasi Keamanan Data. *Jurnal Ilmu Komputer Dan Sistem Informasi (JIKOMSI)*, *7*(1), 203–207. https://doi.org/10.55338/jikomsi.v7i1.2875

Setiawan, A. B. (2025). Penggunaan Cryptography dalam Keamanan Pesan Digital. *Journal of Computer Science and Information Technology*, *1*(2), 60–66. https://doi.org/10.70716/jocsit.v1i2.261

Sinaga, R., & Frangk. (2023). Modifikasi Algoritma Caesar Chiper Dengan Menambahkan Key Untuk Peningkatan Keamanan. *CSRID (Computer Science Research and Its Development Journal)*, *15*(2), 156–166. https://doi.org/10.22303/csrid-.15.2.2023.156-166

Sutoyo, M. N., Qammaddin, Q., Rahayu, R., & Kariani, N. K. R. (2024). Pengamanan Data Berbasis Hill Cipher dengan Operasi Modulo pada Karakter ASCII. *Techno.Com*, *23*(4), 786–795. https://doi.org/10.62411/tc.v23i4.11523

Wasis. (2026). Caesar Chiper Algorithm In Message Security. *Jurnal Inotera*, *11*(1), 66–70. https://doi.org/10.31572/inotera.Vol11.Iss1.2026.ID578

Zuna, M., Wulansari, A., & Brastama Putra, A. (2026). PERBANDINGAN ALGORITMA KRIPTOGRAFI SIMETRIS DAN ASIMETRIS DALAM KEAMANAN DATA DIGITAL. *JATI (Jurnal Mahasiswa Teknik Informatika)*, *10*(1), 249–253. https://doi.org/10.36040/jati.v10i1.16642