Implementasi Logistic Map Berbasis Kompleks: Aplikasi Bilangan Kompleks dalam Sistem Kaotis Chaos-based Cryptography

Rafen Max Alessandro - 13523031[1,2]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]13523031@std.stei.itb.ac.id, [2]rafen.max@gmail.com

***Abstract*—*Chaos-based cryptography* memanfaatkan sifat sistem kaotis yang deterministik tetapi sulit diprediksi untuk menghasilkan algoritma enkripsi yang aman dan efisien. Bilangan kompleks dapat digunakan untuk mengembangkan sistem kaotis dengan memberikan dimensi tambahan dan meningkatkan kompleksitas model sistem nonlinear, menciptakan pola kunci enkripsi yang lebih kaotis dan sulit untuk diprediksi. Makalah ini mengkaji peran bilangan kompleks dalam pembentukan sistem kaotis dan implementasi praktis dari Logistic Map. Studi kasus Logistic Map berbasis kompleks diimplementasikan untuk memodelkan langkah-langkah enkripsi, disertai analisis ketahanan sistem terhadap serangan *brute force* dan analisis statistik. Didapatkan kesimpulan bahwa penggunaan bilangan kompleks secara signifikan meningkatkan kredibilitas sistem kaotis dalam *chaos-based cryptography*, tetapi terdapat keterbatasan yang memerlukan optimisasi lebih lanjut untuk implementasi sistem kaotis dalam skala besar.**

***Keywords*—bilangan kompleks, *chaos-based cryptography*, Logistic Map, sistem kaotis.**

I. PENDAHULUAN

Kemajuan teknologi mendorong masyarakat untuk memasuki era digital yang semakin terhubung, era dengan perkembangan pesat dan signifikan terhadap distribusi data dan informasi. Perkembangan tersebut mentransformasi data untuk dapat dengan mudah dan cepat melintasi batasan geografis dan sosial. Namun, kemudahan distribusi data turut menghadirkan risiko besar, seperti ancaman terhadap privasi, pencurian data, serta manipulasi informasi secara tidak bertanggung jawab. Keamanan data menjadi salah satu pilar utama yang harus diusahakan dalam upaya menjaga stabilitas dan kepercayaan masyarakat di era digital, sebagai fondasi utama dalam melindungi hak-hak digital masyarakat dan menjamin keberlangsungan hubungan digital yang konstruktif.

Untuk menjawab tantangan tersebut, kriptografi berperan sebagai salah satu solusi utama dalam menjaga integritas dan privasi data. Kriptografi adalah disiplin ilmu yang memiliki fokus utama terhadap pengamanan komunikasi dan informasi. Keamanan data bergantung terhadap algoritma enkripsi yang mampu mengamankan data terhadap berbagai jenis serangan, seperti *brute force* maupun analisis statistik. Oleh karena itu, inovasi *chaos-based cryptography* muncul sebagai pendekatan yang memberikan potensial solusi melalui pemanfaatan karakteristik unik fungsi sistem kaotis.

Sistem kaotis dapat digunakan dengan baik memanfaatkan karakteristik yang dimiliki, yaitu berdasar terhadap aturan tertentu tetapi sulit untuk diprediksi. Hal ini memungkinkan fungsi sistem kaotis untuk menghasilkan kunci enkripsi dengan sensitivitas tinggi terhadap kondisi *set-up*, memiliki dinamika nonlinear yang rumit, dan ideal untuk diaplikasikan dalam lingkup kriptografi. Pola dinamika yang dihasilkan oleh suatu fungsi sistem kaotis dapat disesuaikan untuk meningkatkan kerumitan kunci, sehingga keamanan data lebih dapat dijamin terhadap serangan yang memanfaatkan pola-pola sistemik algoritma tradisional.

Penggunaan bilangan kompleks dalam sistem kaotis memberikan dimensi tambahan yang meningkatkan tingkatan kerumitan pola kunci, menciptakan pola yang lebih acak dibandingkan pendekatan berbasis bilangan real. Selain itu, penggunaan bilangan kompleks sejalan dengan karakteristik esensial *chaos-based crypthography* yang sangat bergantung terhadap kondisi *set-up*. Dengan kerangka tersebut, bilangan kompleks tidak hanya menguatkan fungsi sistem kaotis, tetapi juga menjadi solusi inovatif yang dapat digunakan untuk meningkatkan keamanan data di era digital.

Makalah ini membahas secara sistematis peran

bilangan kompleks dalam pengembangan fungsi sistem kaotis dalam *chaos-based crypthography* sebagai sebuah pendekatan dalam menciptakan algoritma enkripsi yang kredibel. Dilakukan pula eksplorasi dalam implementasi melalui pembentukan Logistic Map berbasis kompleks sebagai studi kasus yang memodelkan langkah-langkah enkripsi dan dekripsi, untuk kemudian dianalisis sebagai bahan evaluasi dalam pengaplikasian sistem kaotis tingkat lanjut dalam skala yang lebih besar.

II. LANDASAN TEORI

A. Bilangan Kompleks

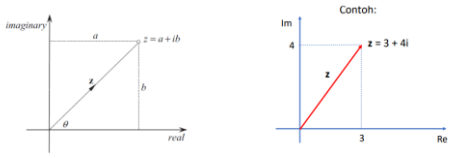
Bilangan kompleks dinyatakan sebagai bilangan yang disusun atas dua komponen utama, yaitu real dan imajiner, dengan representasi umum diberikan oleh formula

$$z = a + bi \tag{1}$$

dengan komponen a menyatakan bilangan real dan komponen b menyatakan bilangan imajiner. i adalah unit imajiner yang didefinisikan sebagai

$$i = \sqrt{-1} \ / \ i^2 = -1 \tag{2}$$

Bilangan kompleks dapat disajikan pada bidang kompleks melalui Diagram Argand sebagai sebuah vektor, dengan sumbu-x memetakan komponen real dan sumbu-y memetakan komponen imajiner [1].



Gambar 1. Diagram Argand Menyajikan Bilangan Kompleks Sebagai Vektor
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

Penyajian sebagai vektor memungkinkan bilangan kompleks untuk digambarkan dalam lingkup dua dimensi, memberikan fleksibilitas yang lebih luas untuk dilakukan analisis matematis dibandingkan bilangan real yang terbatas dalam lingkup satu dimensi. Analisis yang dilakukan meliputi manipulasi bilangan kompleks memanfaatkan operasi-operasi dasar pada bilangan kompleks. Operasi-operasi tersebut adalah sebagai berikut:

1. Penjumlahan dan pengurangan

Operasi penjumlahan pada bilangan kompleks dilakukan untuk masing-masing komponen. Komponen real dijumlahkan dengan komponen real dan komponen imajiner dijumlahkan dengan komponen imajiner. Prinsip yang sama juga berlaku untuk operasi pengurangan [1].

$$\begin{aligned} z_1 &= a_1 + b_1 i \\ z_2 &= a_2 + b_2 i \quad + \\ \hline z_1 + z_2 &= (a_1 + a_2) + (b_1 + b_2)i \end{aligned}$$

Gambar 2. Penjumlahan Dua Bilangan Kompleks
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

2. Perkalian

Perkalian bilangan kompleks didasarkan terhadap metode perkalian aljabar untuk polinomial dengan dua suku. Karena i^2 sama dengan -1, maka hasil dari perkalian kedua komponen imajiner akan menghasilkan bilangan real dengan tanda yang berlawanan [1].

$$\begin{aligned} z_1 z_2 = (a_1 + b_1 i)(a_2 + b_2 i) &= a_1 a_2 + a_1 b_2 i + b_1 a_2 i + b_1 b_2 i^2 \\ &= a_1 a_2 + a_1 b_2 i + b_1 a_2 i - b_1 b_2 \quad (\text{karena } i^2 = -1) \\ z_1 z_2 &= (a_1 a_2 - b_1 b_2) + (a_1 b_2 + b_1 a_2) i \end{aligned}$$

Gambar 3. Perkalian Dua Bilangan Kompleks
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

3. Konjugasi

Bilangan kompleks memiliki karakteristik konjugasi yang menyatakan bilangan kompleks serupa tetapi memiliki tanda yang berlawanan untuk komponen imajiner. Konjugasi dari suatu bilangan kompleks z dinyatakan dengan z^*. Sebagai contoh, untuk bilangan kompleks $z = 3 + 2i$, maka $z^* = 3 - 2i$ [1].

4. Pembagian

Pada pembagian bilangan kompleks, dilakukan perkalian sekawan dengan konjugasi bilangan kompleks pembagi. Operasi ini bertujuan untuk memastikan tidak terjadi rekursif pembagian dengan bilangan kompleks hingga tidak berujung [1].

Misalkan $z_1 = a_1 + ib_1$ dan $z_2 = a_2 + ib_2$

maka

$$\frac{z_1}{z_2} = \frac{z_1}{z_2} \cdot \frac{z_2^*}{z_2^*} = \frac{a_1 + ib_1}{a_2 + ib_2} \cdot \frac{a_2 - ib_2}{a_2 - ib_2}$$

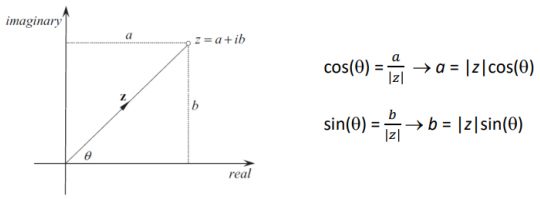
Jadi,

$$\begin{aligned} \frac{z_1}{z_2} &= \frac{a_1 + ib_1}{a_2 + ib_2} \cdot \frac{a_2 - ib_2}{a_2 - ib_2} \\ &= \frac{(a_1 a_2 + b_1 b_2) + i(a_2 b_1 - a_1 b_2)}{a_2^2 + b_2^2} \\ &= \left(\frac{a_1 a_2 + b_1 b_2}{a_2^2 + b_2^2} \right) + i \left(\frac{a_2 b_1 - a_1 b_2}{a_2^2 + b_2^2} \right) \end{aligned}$$

Gambar 4. Pembagian Dua Bilangan Kompleks
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

5. Representasi Bentuk Polar

Menggunakan Diagram Argand, bilangan kompleks dapat dinyatakan dalam bentuk polar, yaitu representasi bilangan kompleks yang didasarkan terhadap modulus (|z|) dan sudut vektor terhadap sumbu-x (θ), dengan modulus mengacu terhadap panjang vektor yang terbentuk pada Diagram Argand, yaitu nilai $\sqrt{a^2 + b^2}$ [1].



- Jadi, $z = a + bi = |z|\cos(\theta) + i|z|\sin(\theta) = |z|(\cos(\theta) + i\sin(\theta))$

Gambar 5. Representasi Bilangan Kompleks dalam Bentuk Polar
Sumber: https://informatika.stei.itb.ac.id/~rinaldi.munir/

B. Sistem Kaotis

Sistem kaotis mengacu kepada sistem yang menunjukkan sensitivitas tinggi terhadap kondisi *set-up*, sehingga perubahan kecil yang bersifat insignifikan pada penetapan kondisi set-up berujung kepada hasil akhir yang jauh berbeda. Sistem kaotis dibentuk menggunakan sistem yang bersifat deterministik, yaitu sistem yang sepenuhnya ditentukan oleh kondisi awal dan aturan yang dijalani. Namun, meskipun sistem ini tidak memiliki elemen acak dalam penerapannya, tingkat sensitivitas tinggi yang dimilikinya menyebabkan hasil akhir dari sistem tidak dapat diprediksi dalam jangka panjang. Perbedaan nilai menyebabkan pola iterasi yang jauh berbeda sehingga rekursivitas fungsi membentuk pola yang tampak acak, walaupun sebenarnya pola yang dihasilkan mengikuti aturan deterministik yang mendasarinya. Sistem kaotis menjadi sebuah sistem yang deterministik tetapi sulit untuk diprediksi.

Salah satu contoh fungsi sistem kaotis yang sering digunakan adalah Logistic Map, sebuah fungsi iteratif sederhana yang memanfaatkan model sistem nonlinear untuk menghasilkan dinamika kaotis. Model sistem linear mengacu kepada representasi matematis sistem dinamis dengan hubungan *input* dan *output* yang tidak linear, sehingga perubahan pada *input* tidak diikuti dengan perubahan yang proporsional pada *output* secara kontinu [2]. Logistic Map dirumuskan dengan persamaan berikut.

$$x_{n+1} = rx_n(1 - x_n) \quad (3)$$

Dengan x_n adalah nilai pada iterasi ke-*n*, *r* adalah parameter kontrol, dan nilai *x* diatur untuk berada dalam interval [0, 1]. Fungsi ini awalnya dirancang sebagai model pertumbuhan populasi dengan keterbatasan sumber daya, sebelum didapati bahwa Logistic Map menunjukkan dinamika kaotis untuk suatu nilai *r* [3].

Variabel *r* sebagai parameter kontrol memegang peranan penting dalam menentukan tingkat kaotis sistem. Ketika parameter *r* bernilai lebih kecil dari 3, hasil iterasi akan menuju nilai tetap. Namun, ketika *r* dibesarkan hingga mendekati 4, hasil iterasi menunjukkan perubahan bertahap dari nilai tetap hingga osilasi secara periodik sebelum akhirnya menjadi kaotis, dimana pola iterasi yang terbentuk menjadi sangat kompleks dan sulit untuk diprediksi.

C. Chaos-based Cryptography

Chaos-based cryptography adalah pendekatan kriptografi yang mengaplikasikan karakteristik sistem kaotis untuk menciptakan algoritma enkripsi yang aman, efisien, dan sulit untuk diprediksi. Pendekatan ini mengoptimalkan penggunaan sifat deterministik sistem kaotis untuk menghasilkan kunci enkripsi yang kompleks dan tahan terhadap serangan seperti *brute force* dan analisis statistik. Sifat deterministik sistem kaotis juga dimanfaatkan untuk memastikan pola enkripsi dapat direproduksi dengan baik untuk digunakan dalam proses dekripsi [4].

III. ANALISIS

Karakteristik bilangan kompleks yang secara alami merepresentasikan lingkup dua dimensi dimanfaatkan secara ideal sebagai model sistem nonlinear. Sistem yang dibentuk menggunakan bilangan kompleks dimodelkan berdasarkan dimensi bilangan real dan bilangan imajiner, sehingga melalui iterasi operasi perkalian dan eksponensial bilangan kompleks, terbentuk interaksi dinamis yang terjadi secara bersamaan antara dua dimensi tersebut. Perilaku dinamis yang terbentuk menciptakan pola yang menunjukkan sifat-sifat kaotis, yaitu sensitivitas tinggi terhadap kondisi awal, teratur, tetapi sulit untuk diprediksi. Pemanfaatan karakteristik bilangan kompleks sebagai model sistem nonlinear secara optimal memungkinkan fungsi sistem kaotis untuk menghasilkan pola yang lebih kaya dan beragam, memberikan lebih banyak variabel untuk diolah dan dianalisis. Kerumitan pola kunci dan sensitivitas terhadap perubahan kecil pada kondisi *set-up* menjadi keunggulan dalam *chaos-based cryptography* untuk menciptakan sistem enkripsi yang aman dan kredibel. Oleh sebab itu, dapat dilakukan implementasi bilangan kompleks terhadap sistem kaotis untuk membentuk *chaos-based cryptography* yang lebih aman dan kredibel, misalnya terhadap sistem Logistic Map.

Dengan mengintegrasikan bilangan kompleks, dapat dibentuk Logistic Map berbasis kompleks yang menghasilkan pola yang jauh lebih sulit diprediksi

dibandingkan dengan Logistic Map berbasis real. Perubahan tersebut mengakibatkan fungsi Logistic Map untuk beralih menjadi sebagai berikut.

$$z_{n+1} = rz_n(1 - z_n), \quad z \in \mathbb{C} \tag{4}$$

Penggunaan bilangan kompleks menyebabkan pola yang dihasilkan oleh Logistic Map tidak hanya dipicu oleh parameter kontrol, tetapi juga oleh interaksi antara komponen real dan komponen imajiner yang dimiliki oleh bilangan kompleks. Interaksi antara kedua komponen menciptakan dinamika dua dimensi yang tidak hanya memberikan pengaruh terhadap pola yang terbentuk, tetapi juga dipengaruhi oleh distribusi nilai z_n di dalam ruang kompleks. Hal ini mengakibatkan terjadi lapisan faktor tambahan yang turut memengaruhi hasil pola yang terbentuk. Sensitivitas tinggi terhadap kondisi awal diperkuat oleh interaksi yang terjadi antar dimensi, sehingga pola yang dihasilkan mengalami kenaikan tingkatan kompleksitas dalam taraf yang signifikan.

Logistic Map berbasis real menggunakan nilai di antara 0 hingga 1 sebagai hasil dari iterasi pola untuk memastikan stabilitas iterasi dan pola yang dihasilkan lebih terkendali. Maka, konsep yang sama juga berlaku pada Logistic Map berbasis kompleks. Untuk membentuk iterasi yang stabil dan pola yang lebih terkendali, modulus dari bilangan kompleks yang digunakan sebagai parameter kondisi *set-up* dibatasi pada interval yang sama, yaitu [0, 1], atau dinyatakan dalam hubungan $|z| = \sqrt{a^2 + b^2}$ untuk $z = a + bi$. Hubungan tersebut menyebabkan nilai a dan b berada dalam batasan $a^2 + b^2 \leq 1$.

Berdasarkan prakondisi tersebut, modulus bilangan kompleks dapat dibatasi untuk berada di dalam rentang 0 hingga 1, sehingga untuk hasil iterasi z_n dimana $|z_n| > 1$, dapat dilakukan normalisasi melalui persamaan matematika berikut

$$z_n = \frac{z_n}{|z_n|} \tag{5}$$

untuk menjaga stabilitas memastikan sistem menghasilkan pola kaotis yang terkendali, memenuhi kebutuhan *chaos-based cryptography* yang didasarkan terhadap pola kunci yang aman dan stabil.

IV. Implementasi Logistic Map Berbasis Kompleks

Berdasarkan landasan teori dan analisis mengenai bilangan kompleks, sistem kaotis, dan *chaos-based cryptography*, ketiga cakupan tersebut dapat digunakan untuk membentuk Logistic Map berbasis kompleks menggunakan bahasa pemrograman Python. Tahapan dalam pembentukan dan pengujian Logistic Map berbasis kompleks adalah sebagai berikut.

1. Pembentukan kunci enkripsi

Sebagai nilai yang mengendalikan iterasi dalam pembentukan kunci, perlu ditentukan suatu nilai bilangan kompleks dan parameter kontrol sistem sebagai parameter dalam pembentukan fungsi. Sesuai dengan karakteristik Logistic Map berbasis kompleks, dibutuhkan komponen real dan imajiner dari bilangan kompleks yang memenuhi prakondisi $a^2 + b^2 \leq 1$, serta parameter kontrol sistem dengan nilai mendekati 4 agar sistem menghasilkan pola kaotis yang terkendali sebagai kunci. Sebagai contoh dalam studi kasus penelitian ini, dapat digunakan bilangan kompleks z = 0.1352 + 0.3031i dan parameter kontrol sistem r = 3.8 sebagai parameter kondisi *set-up* dalam implementasi fungsi.

```
# Parameter kondisi set-up Logistic Map
z0 = 1.352 + 3.031j     # bilangan kompleks
r = 3.8                 # parameter kontrol
```

Gambar 6. Penetapan Parameter Kondisi *Set-up*
Sumber: dokumentasi pribadi

2. Penyusunan fungsi kaotis

Menggunakan parameter yang telah ditentukan, dapat dibentuk program pada bahasa pemrograman Python sebagai implementasi nyata Logistic Map berbasis kompleks. Fungsi logistic_map_complex menerima parameter z dan r sebagai bilangan kompleks dan parameter kontrol yang digunakan sebagai kondisi *set-up*, dan *iterations* yang menyatakan jumlah dilakukannya iterasi. Pada setiap iterasi, dilakukan perhitungan nilai z_n menggunakan model sistem nonlinear, normalisasi nilai z_n untuk nilai $|z_n| > 1$, dan pembentukan kunci berdasarkan interaksi antara dua dimensi yang dimiliki oleh bilangan kompleks.

```
def logistic_map_complex(z, r, iterations):

    results = []
    for _ in range(iterations):

        # Pemodelan sistem nonlinear Logistic Map
        z = r * z * (1 - z)

        # Normalisasi jika modulus lebih besar dari 1
        if abs(z) > 1:
            z = z / abs(z)

        # membentuk pola enkripsi berdasarkan interaksi antara dua dimensi
        z_combined = abs(z) * np.angle(z)

        # gabungkan setiap pola enkripsi ke dalam satu array
        results.append(z_combined)

    return results
```

Gambar 7. Fungsi logistic_map_complex
Sumber: dokumentasi pribadi

3. Tahapan enkripsi

Setelah kunci enkripsi terbentuk, *chaos-based*

cryptography dapat diimplementasikan terhadap suatu string untuk mengubah *input* menjadi *array of integer*. Transformasi ini dilakukan melalui operasi xor antara nilai ASCII setiap *character* pada *input* dengan kunci enkripsi pada indeks yang sesuai. Keseluruhan tahapan enkripsi *input* dilakukan dalam fungsi encrypt_with_logistic_map sebagai berikut.

```python
def encrypt_with_logistic_map(input_string, z0, r):

    # Konversi string menjadi angka ASCII
    ascii_values = string_to_ascii(input_string)

    # Tentukan jumlah iterasi berdasarkan panjang string
    iterations = len(ascii_values)

    # Hasilkan pola iterasi Logistic Map
    logistic_pattern = logistic_map_complex(z0, r, iterations)

    # Skala pola iterasi ke dalam bentuk integer
    scaled_pattern = [int(abs(x) * 1000) for x in logistic_pattern]

    # Lakukan operasi XOR antara ASCII dan pola iterasi
    encrypted_values = [ascii_values[i] ^ scaled_pattern[i] for i in range(iterations)]

    return encrypted_values
```

Gambar 8. Fungsi encrypt_with_logistic_map
Sumber: dokumentasi pribadi

```
Masukkan string untuk dienkripsi: makalah algeo
Hasil enkripsi: [634, 865, 457, 975, 240, 890, 655, 487, 790, 130, 1069, 347, 1067]
```

Gambar 9. Hasil Enkripsi String "makalah algeo" Menggunakan Fungsi encrypt_with_logistic_map
Sumber: dokumentasi pribadi

Gambar 8 menunjukkan algoritma program yang melakukan perkalian setiap pola kunci enkripsi (tersimpan di dalam variabel logistic_pattern) dengan 1000 lalu dibulatkan ke bawah (*round down*). Langkah ini bertujuan untuk membentuk kunci enkripsi yang berupa bilangan bulat (*integer*) agar operasi xor dapat dilakukan, tetapi tetap mempertahankan kompleksitas dan dinamika kaotis dari kunci enkripsi.

4. Tahapan dekripsi

Setelah dilakukan enkripsi, dekripsi dapat dilakukan dengan tahapan yang sama dengan memanfaatkan sifat inversi diri (*self-inverse property*) dari operasi xor. Jika operasi xor dilakukan terhadap hasil dari operasi xor sebelumnya, efek dari operasi xor akan dibatalkan. Dengan kata lain, berlaku persamaan (*Plain text* ⊕ *Key*) ⊕ *Key* = *Plain text* yang digunakan dalam proses dekripsi data hasil enkripsi.

```python
def decrypt_with_logistic_map(encrypted_values, z0, r):

    # Tentukan jumlah iterasi berdasarkan panjang ciphertext
    iterations = len(encrypted_values)

    # Hasilkan pola iterasi Logistic Map
    logistic_pattern = logistic_map_complex(z0, r, iterations)

    # Skala pola iterasi ke dalam bentuk integer
    scaled_pattern = [int(abs(x) * 1000) for x in logistic_pattern]

    # Lakukan operasi XOR untuk mendapatkan kembali nilai ASCII
    ascii_values = [encrypted_values[i] ^ scaled_pattern[i] for i in range(iterations)]

    # Konversi kembali ke string
    return ''.join(chr(value) for value in ascii_values)
```

Gambar 10. Fungsi decrypt_with_logistic_map
Sumber: dokumentasi pribadi

```
Masukkan komponen real: 1.352
Masukkan komponen imajiner: 3.031
Masukkan parameter kontrol: 3.8
Hasil dekripsi: makalah algeo
```

Gambar 11. Hasil Dekripsi String Secara Tepat Menggunakan Fungsi encrypt_with_logistic_map untuk String "makalah algeo"
Sumber: dokumentasi pribadi

Untuk melakukan dekripsi, program meminta *input* parameter-parameter yang digunakan dalam kondisi *set-up* Logistic Map. Ketika dimasukkan *input* yang sesuai, yaitu parameter-parameter yang digunakan dalam kondisi *set-up*, maka didapatkan hasil dekripsi yang tepat sesuai dengan string sebelum dilakukan enkripsi, dan *chaos-based cryptography* dapat dinyatakan berlaku sesuai dengan prosedur kriptografi yang diharapkan beroperasi.

Sesuai dengan karakteristik sensitivitas tinggi yang dimiliki oleh sistem kaotis, perubahan minimal pada parameter yang digunakan dalam kondisi *set-up* Logistic Map akan menghasilkan kunci enkripsi yang jauh berbeda. Maka, ketika dimasukkan *input* yang tidak sesuai, bahkan dengan perbedaan sebesar 0.001, hasil dekripsi menjadi berantakan dan tidak menggambarkan string yang dilakukan enkripsi, seperti yang ditunjukkan pada gambar-gambar berikut.

```
Masukkan komponen real: 1.353
Masukkan komponen imajiner: 3.031
Masukkan parameter kontrol: 3.8
Hasil dekripsi: mɳmco]X8BRAĤ
```

Gambar 12. Hasil Dekripsi String Secara Tidak Tepat Menggunakan Fungsi encrypt_with_logistic_map untuk String "makalah algeo"
Sumber: dokumentasi pribadi

```
Masukkan komponen real: 1.352
Masukkan komponen imajiner: 3.0311
Masukkan parameter kontrol: 3.8
Hasil dekripsi: makal`g!noh↔↕
```

Gambar 13. Hasil Dekripsi String Secara Tidak Tepat Menggunakan Fungsi encrypt_with_logistic_map untuk String "makalah algeo"
Sumber: dokumentasi pribadi

```
Masukkan komponen real: 1.352
Masukkan komponen imajiner: 3.031
Masukkan parameter kontrol: 3.81
Hasil dekripsi: makalgi.bqq@
```

Gambar 14. Hasil Dekripsi String Secara Tidak Tepat Menggunakan Fungsi encrypt_with_logistic_map untuk String "makalah algeo"
Sumber: dokumentasi pribadi

V. Analisis Sistem

Setelah melakukan studi kasus implementasi langsung Logistic Map berbasis kompleks, dapat dilakukan pengkajian terhadap kelebihan dan kekurangan dari integrasi bilangan kompleks dalam sistem kaotis sebagai *chaos-based cryptography*.

A. Keunggulan

Sistem kaotis berbasis kompleks menawarkan tingkat keamanan yang tinggi. Sensitivitas ekstrem terhadap parameter kondisi *set-up* memberikan perubahan pola iterasi yang sama sekali berbeda untuk perubahan sekecil apa pun terhadap bilangan kompleks dan parameter kontrol yang digunakan. Sifat bilangan kompleks yang disusun atas dua komponen, real dan imajiner, memberikan dimensi tambahan dalam pembentukan pola kunci. Interaksi antar komponen meningkatkan dinamika kaotis sebagai tambahan dalam lapisan keamanan yang tidak dimiliki oleh sistem berbasis real. Tambahan dari lapisan keamanan yang terbentuk meningkatkan ketahanan terhadap serangan khususnya serangan *brute force* dan analisis statistik.

Dalam serangan *brute force*, dibutuhkan tebakan dengan tingkat akurasi terlalu tinggi untuk menduga parameter kondisi *set-up*, bahkan hampir mustahil untuk dilakukan dalam waktu yang wajar. Lalu, pola iterasi dengan dinamika kaotis yang dihasilkan sistem menyebabkan serangan analisis statistik kurang efektif karena tidak adanya pola yang dapat digunakan untuk melakukan prediksi kunci enkripsi yang terbentuk.

B. Keterbatasan

Walaupun demikian, implementasi sistem kaotis berbasis kompleks memiliki tingkatan kompleksitas komputasi yang lebih rumit dibandingkan dengan basis real. Operasi bilangan kompleks yang digunakan dalam sistem, meliputi perkalian dan eksponensial, membutuhkan daya komputasi yang lebih besar karena melibatkan perhitungan pada dua komponen secara simultan. Dalam implementasi skala yang jauh lebih besar, kompleksitas komputasi dapat menjadi hambatan, khususnya pada perangkat dengan kapasitas komputasi yang terbatas.

Keterbatasan lain yang dimiliki oleh sistem adalah ketergantungannya terhadap ketepatan numerik. Bilangan kompleks diwakili oleh *floating-point* yang memiliki keterbatasan presisi. Kesalahan pembulatan yang sangat mungkin untuk terjadi dalam pembentukan kunci enkripsi dapat menyebabkan penyimpangan pola iterasi secara substansial sehingga proses dekripsi tidak dapat berlangsung. Oleh karena itu, implementasi sistem kaotis berbasis kompleks memerlukan pengelolaan presisi yang tepat untuk memastikan kesalahan numerik tidak memengaruhi kredibilitas sistem dalam membentuk kunci enkripsi.

Berdasarkan pertimbangan tersebut, sistem kaotis berbasis kompleks memberikan potensi yang tidak dapat diabaikan dalam pengembangan algoritma enkripsi modern. Namun, diperlukan optimisasi yang ekstensif dan komprehensif untuk mengatasi keterbatasan yang dimiliki dan menjaga integritas utilitas sistem sebagai *chaos-based cryptography*.

VI. Kesimpulan

Bilangan kompleks memberikan kontribusi signifikan dalam meningkatkan dinamika kaotis dan kompleksitas sistem kaotis, menjadikannya sebagai implementasi yang ideal untuk digunakan dalam *chaos-based cryptography*. Integrasi bilangan kompleks pada sistem kaotis, contohnya pada Logistic Map yang memanfaatkan model sistem nonlinear, membentuk pola kunci enkripsi yang deterministik, tetapi memiliki sensitivitas ekstrem terhadap parameter kondisi set-up dan sulit untuk diprediksi. Karakteristik tersebut menjadi keunggulan bagi sistem kaotis berbasis kompleks untuk diterapkan dalam lingkup kriptografi untuk mengamankan data di era digital yang semakin terhubung.

Melalui analisis studi kasus Logistic Map berbasis kompleks, didapati bahwa penggunaan sistem kaotis berbasis kompleks memberikan hasil afirmatif dalam melakukan enkripsi dan dekripsi data. Namun, terdapat keterbatasan dalam konteks efisiensi komputasi skala besar dan ketergantungan terhadap ketepatan numerik dalam pembentukan kunci, sehingga diperlukan optimasi lebih lanjut untuk meningkatkan kredibilitas sistem tanpa mempertaruhkan tingkat keamanan dan kompleksitas sistem.

Seturut dengan berkembangnya kebutuhan akan keamanan data yang lebih kredibel dan tahan terhadap berbagai jenis serangan, *chaos-based cryptography* berbasis kompleks memberikan potensi besar sebagai solusi inovatif dalam melindungi data digital. Melalui penelitian dan optimisasi lebih lanjut, bilangan kompleks dapat menjadi elemen krusial yang berpengaruh dalam pengembangan kriptografi modern, memberikan peran relevan untuk melindungi integritas data di era digital.

VII. Ucapan Terima Kasih

Penulis memanjatkan puji syukur kepada Tuhan yang Maha Esa atas berkat dan rahmat-Nya yang melimpah dalam proses penulisan makalah sehingga makalah dapat diselesaikan dengan baik dan tepat waktu. Penulis juga mengucapkan terima kasih yang sebesar-besarnya kepada Bapak Ir. Rila Mandala, M.Eng., Ph.D. selaku dosen

pengampu mata kuliah IF2123 Aljabar Linear dan Geometri kelas K01 atas waktu, bimbingan, dan ilmu pengetahuan yang telah diberikan sebagai bekal dalam pembentukan makalah. Akhir kata, penulis menyampaikan dukungan semangat kepada teman-teman penulis yang sedang berjuang bersama-sama untuk menyelesaikan masa studi semester ganjil tahun ajaran 2024/2025.

VIII. LAMPIRAN

Source code program yang dibentuk sebagai studi kasus implementasi Logistic Map berbasis kompleks dapat diakses melalui tautan berikut: https://github.com/rafenmaxxx/LogisticMapComplex.git

REFERENCES

[1] Munir, Rinaldi. "Homepage Rinaldi Munir". https://informatika.stei.itb.ac.id/~rinaldi.munir/. (diakses pada 28 Desember 2024).

[2] Kocarev, L., & Jakimoski, G. (2001). Logistic map as a block encryption algorithm. *Physics Letters A*, *289*(4-5), 199-206.

[3] Pecora, L. M., & Carroll, T. L. (2015). Synchronization of chaotic systems. *Chaos: An Interdisciplinary Journal of Nonlinear Science*, *25*(9).

[4] El-Latif, A. A. A., Ramadoss, J., Abd-El-Atty, B., Khalifa, H. S., & Nazarimehr, F. (2022). A novel chaos-based cryptography algorithm and its performance analysis. *Mathematics*, *10*(14), 2434.

PERNYATAAN

Dengan ini saya menyatakan bahwa makalah yang saya tulis ini adalah tulisan saya sendiri, bukan saduran, atau terjemahan dari makalah orang lain, dan bukan plagiasi.

Bandung, 27 Desember 2024

Rafen Max Alessandro
13523031