ANALISIS PERBANDINGAN UNJUK KERJA ALGORITMA LORENTZ, JULIA SET DAN TENT FUNCTION SEBAGAI ALGORITMA CHAOTIC

Rika Foelyati[1], M. Ary Murti[2], Asep Mulyana[3]
[1,2,3]Jurusan Teknik Elektro, Sekolah Tinggi Teknologi Telkom
Jln. Telekomunikasi, Dayeuh Kolot Bandung, 40257, Indonesia
E-mail: [2]mam@sttelkom.ac.id

ABSTRAKSI

Dalam jaringan komunikasi global seperti internet, keamanan data yang ditransmisikan harus dijaga dari pihak-pihak yang ingin mencuri atau merusak data tersebut. Untuk menjamin bahwa data aman dalam perjalanan dari pengirim sampai ke tujuannya, maka sebelumnya data dienkripsi menjadi bentuk yang tidak berarti. Terdapat banyak sekali algoritma enkripsi yang telah diterapkan dengan berbagai level/tingkat kehandalan. Seperti pada chaos misalnya, merupakan salah satu metoda enkripsi yang digunakan dalam sistem keamanan untuk suatu data yang sangat rahasia, yang hanya bisa dibuka dan dibaca oleh pihak-pihak yang tahu kunci-nya. Chaos merupakan suatu pengacakan data dimana mempunyai suatu sifat dasar yaitu sensitive pada kondisi awal. Chaos ini dipakai untuk mengenkripsi dan mendekripsi suatu file informasi, hal ini mengingat kemampuan chaos dalam pengacakan data informasi. Cryptography Symmetric merupakan suatu metode kriptografi yang menggunakan kunci yang sama antara si pengirim dan si penerima. Menentukan suatu sistem keamanan yang handal dan efektif sangat diperlukan dalam rangka pengamanan suatu file.

Untuk mengetahui sampai sejauh mana kehandalan algoritma Chaos tersebut dan bagaimana metoda analisis kehandalannya, dalam proyek akhir ini, dilakukan analisis dengan membandingkan tingkat kehandalan antara algoritma Lorentz, Julia Set, dan Tent Function dalam proses enkripsi dan dekripsi suatu file informasi. Metoda yang dilakukan adalah dengan melakukan simulasi menggunakan Matlab.

Analisa dilakukan berdasarkan beberapa parameter yaitu uji periodic, autocorrelation, power spectral density, avallanche effect dan panjang data output pada chaotic algorithm. Dari hasil analisa tersebut diperoleh bahwa algoritma Lorentz lebih handal daripada algoritma lainnya.

Kata kunci: *Algoritma Chaos, algoritma Lorentz, Julia Set, Tent Function, dan perbandingan*

1. PENDAHULUAN

Pada saat sekarang ini dimana jaringan sudah bersifat terbuka sehingga memudahkan para pemakai dalam mengakses informasi yang diinginkan. Oleh karena itu dibutuhkan suatu sistem keamanan yang dapat digunakan untuk mengamankan suatu *file* informasi. Sistem keamanan dengan cara pengacakan *file* dikenal sebagai kriptografi. Namun masih ada beberapa masalah yang dihadapi oleh teknologi kriptografi pada saat ini, di antaranya:

- Waktu proses yang lama, tidak sebanding dengan tingkat keamanan yang diinginkan.
- Panjang dari kunci kriptografi bersifat tetap, sehingga tingkat keamanan data tidak fleksibel.
- Tidak mudah diperbaharui.

Penggunaan sinyal yang bersifat chaos dalam sistem kriptografi merupakan salah satu metode baru yang banyak diharapkan dapat menjawab permasalahan di atas. Algoritma yang bersifat chaos merupakan salah satu jenis kriptografi yang handal yang dapat dipakai dalam proses kriptografi suatu *file* informasi. Hal ini mengingat kemampuannya dalam pengacakan data informasi. Chaos merupakan fenomena *dynamic* dan *unpredictable* yang mempunyai perilaku terbatas yang menghasilkan ciri-ciri dasar, yaitu: spectrum daya yang kontinu pada suatu pita frekuensi yang lebar, mempunyai kepekaan yang tinggi, bersifat ergodik pada suatu sample space terbatas.

2. KRIPTOGRAFI

Kriptografi adalah istilah Indonesia yang diterjemahkan dari kata *Crytography*, yang berasal dari istilah *Crypto* yang berarti rahasia dan *Graphia* yang berarti tulisan(pesan). Sehingga Kriptografi dapat kita artikan ilmu yang mempelajari tentang metode untuk membuat tulisan atau pesan rahasia

Pada transformasi ini terdapat dua macam masalah keamanan data, yaitu masalah privasi (*privacy*) dan keotentikan (*authentication*). Privasi mengandung arti bahwa data yang diinginkan hanya dapat dimengerti informasinya oleh penerima yang berhak. Sedangkan keotentikan mencegah pihak ketiga untuk mengirimkan data yang salah atau mengubah data yang dikirimkan. Adapun aspek-aspek keamanan pada Kriptografi adalah sebagai berikut:

- *Authentication*
- *Integrity*
- *Convidentiality*
- *Nonrepudiation*

Algoritma Kriptografi dipakai untuk memanggil suatu *ciphertext*, yaitu fungsi matematika yang digunakan untuk enkripsi dan

dekripsi. Berdasarkan kunci yang dipakai, algoritma Kriptografi dapat dibedakan atas dua golongan, yaitu:

a. *Symmetric Algorithms* disebut juga algoritma konvensional dimana menngunakan kunci yang sama untuk proses enkripsi dan dekripsinya

b. *Asymmetric Algorithms* menggunakan kunci enkripsi dan kunci dekripsi yang berbeda. Kunci enkripsi dapat disebarkan kepada umum dan dinamakan sebagai kunci publik (*public key*) sedangkan kunci dekripsi disimpan untuk digunakan sendiri dan dinamakan sebagai kunci pribadi (*private key*).

Keamanan suatu sistem kriptografi merupakan masalah yang paling fundamental. Dengan menggunakan sistem standar terbuka, maka keamanan suatu sistem kriptografi akan lebih mudah dan lebih cepat dianalisa. Mengingat kenyataan inilah maka sekarang tidak digunakan lagi algoritma rahasia yang tidak diketahui tingkat keamanannya.

Sebuah sistem kriptografi dirancang untuk menjaga *plaintext* dari kemungkinan dibaca oleh pihak-pihak yang tidak berwenang, yang secara umum dinamakan sebagai penyerang (*attacker*). Penyerang diasumsikan memiliki akses tak terbatas terhadap jalur tak aman yang digunakan untuk transaksi *ciphertext*. Oleh karena itu, penyerang dianggap memiliki akses langsung terhadap *ciphertext.*

2.1 Pengertian Teorema Chaos

Chos merupakan fenomena *dinamik* dan *unpredictable*. Salah satu teorema tentang chaos disampaikan oleh *Shilnikov* dan *C. Silva*. Menurut *Shilnikov*, chaos adalah perilaku dinamis yang mempunyai suatu himpunan yang unik dan invariant. Sedangkan menurut *C. Silva*, chaos adalah suatu system dinamis yang mempunyai perilaku terbatas yang menghasilkan tiga ciri dasar, yaitu:

- Mempunyai spektrum daya yang kontinyu pada suatu frekwensi tertentu. Ciri ini menunjukkan sinyal yang aperiodik dan sekaligus menjadikan chaos sering dianalogikan dengan sinyal *noise*.
- Mempunyai kepekaan yang tinggi terhadap kondisi awal.
- Bersifat ergodik pada suatu *sample space* terbatas.

Selain itu, salah satu definisi tentang chaos mengatakan bahwa suatu fungsi *f(.)* akan bersifat chaotic apabila memenuhi salah satu syarat dari dua buah kondisi berikut ini:

- *f(.) sensitive* terhadap kondisi awal dalam domainnya, hal ini mengandung pengertian bahwa, system harus memenuhi persamaan berikut ini:

$$|x-y| < \delta \text{ dan } |f(x) - f(y)| > \varepsilon$$

dimana:

$f^{[n]} = f(f(f...f(x)))$ adalah iterasi ke-n dari x dalam fungsi f,

n: bilangan integer positif, dan δ , $\varepsilon > 0$.

- *f(.)* mempunyai eksponen *Lyapunof* positif pada masing-masing titik dalam domainnya yang tidak *eventually periodic* (suatu sample yang membuat *f(.)* periodik).

Bertolak dari definisi tentang eksponen *Lyapunov*, bahwa jika diketahui suatu system memiliki λ(x) sebagai berikut:

$$\lambda(x) = \lim_{x\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln \left| \frac{df(x_i)}{dx} \right|$$

Maka untuk memenuhi persyaratan diatas, λ(x) harus selalu ada dalam domain-nya dan lebih besar dari nol.

2.2 Sistem Lorentz

Sistem *Lorentz* merupakan system fisik yang dapat mengilustrasikan *chaos motion* pada partikel fluida yang dipanaskan. Persamaan system *Lorentz* dinyatakan oleh *Edward Lorentz* seperti di bawah ini:

$$\frac{dx}{dt} = -\sigma x + \sigma y + F(x,y,z) \quad \frac{dx}{dt} = -\sigma x + \sigma y$$

$$\frac{dy}{dt} = -rx - y + G(x,y,z) \quad \frac{dy}{dt} = -rx - y - xz$$

$$\frac{dz}{dt} = -bz + H(x,y,z) \quad \frac{dz}{dt} = -bz + xy$$

Di mana F(x,y,z) = 0, G(x,y,z) = -xz, H(x,y,z) = xy, dan masing-masing memiliki kondisi awal sama dengan nol. Sehingga persamaan diatas dapat dinyatakan dalam bentuk:

$$\begin{bmatrix} \frac{dx}{dt} \\ \frac{dy}{dt} \\ \frac{dz}{dt} \end{bmatrix} = \begin{pmatrix} -\sigma & \sigma & 0 \\ r & -1 & 0 \\ 0 & 0 & -b \end{pmatrix} \begin{bmatrix} x \\ y \\ z \end{bmatrix}$$

$$A = \begin{pmatrix} -\sigma & \sigma & 0 \\ r & -1 & 0 \\ 0 & 0 & -b \end{pmatrix}$$

Perhitungan nilai *eigen* λ matrik A diperoleh melalui det (A –λ.I) = 0, dimana I adalah matrik identitas berukuran 3x3.

$$Det(A - \lambda.I) = \det \begin{pmatrix} -\sigma-\lambda & \sigma & 0 \\ r & -1-\lambda & 0 \\ 0 & 0 & -b-\lambda \end{pmatrix}$$

$$= -(b+\lambda)[\lambda^2 + (\sigma+1)\lambda + \sigma(1-r)] = 0$$

Sehingga λ1, λ2, λ3 masing-masing adalah:

$$\lambda_1 = \frac{-(\sigma+1)+\sqrt{(\sigma+1)+4r\sigma}}{2}$$

$$\lambda_2 = \frac{-(\sigma+1)-\sqrt{(\sigma+1)+4r\sigma}}{2}$$

$$\lambda_3 = -b$$

Salah satu system Lorentz yang digunakan sebagai dasar yaitu *Lorentz attractor 3 dimentional*. Sesuai dengan persamaan differensial biasa pada persamaan diatas, didapat suatu persamaan dengan menggunakan parameter σ= 10, r = 28, b = 8/3, seperti di bawah ini:

$$\frac{dx}{dt} = 10(y-x)$$

$$\frac{dy}{dt} = 28x - y - xz$$

$$\frac{dz}{dx} = xy - \frac{8}{3}z$$

Penggunaan parameter di atas dikarenakan sinyal yang diperoleh akan menghasilkan suatu lintasan dengan *attractor* yang berbeda.

2.3 Sistem Julia

Julia Set merupakan suatu fungsi kompleks yang menghasilkan suatu lukisan/ gambar dari banyak iterasi yang dilakukan sehingga menjadi suatu bentuk yang sangat indah.

Jika $z = x + yi$ dan $w = u + vi$, maka dengan menambahkan *z* atau *w* didapatkan:

$$z + w = (x+u) + (y+v)i$$

$$zw = (xu - yv) + (xv + yu)i$$

$$z^2 = (x^2 - y^2) + 2xyi$$

Sebuah bilangan kompleks $x + y_i$ dapat dipertim-bangkan dari sepasang bilangan real (x,y) maupun dari angka (x,y) dari suatu bidang.

Jika *f* suatu fungsi komplek. Bilangan kompleknya p merupakan suatu nilai tetap pada *f*, jika *f*(p) = p. Menurut '*fundamental theorem of algebra*' mengatakan jika g suatu fungsi komplek *polynomial*, maka disini z sebagai bilangan komplek. Dengan demikian g(z) = 0. Untuk ekivalen dikatakan bahwa setiap fungsi komplek *polynomial f.* Bahkan, jika *f* sebuah fungsi *polynomial,* dan g(z) = *f*(z) – z . Maka *f*(z) = z jika dan hanya jika g(z) = *f*(z) – z = 0. Karena itu 0 dari g merupakan nilai tetap dari *f*. Sehingga dengan mudah dapat dicari nilai tetap jika fungsi *f* didefinisikan $f(z) = z^2 + c$, karena z adalah nilai tetap dari *f* yang mana $z = z^2 + c$, atau ekivalennya $z^2 - z + c = 0$. Sehingga rumus nilai tetap dari *f*, yaitu:

$$z = \frac{1}{2} \pm \frac{1}{2}\sqrt{1-4c}$$

2.4 Tent Function

Salah satu contoh pembangkit sinyal chaos adalah persamaan fungsi *Tent* yang memiliki persamaan dasar:

$$x_{n-1} = \begin{cases} 2ax_n & 0 \le x_n \le 0{,}5 \\ 2a(1-x_n) & 0{,}5 \le x_n \le 1 \end{cases}$$

Besarnya a menentukan karakter chaos yang ditunjukkan. Untuk a > 0.5 akan diperoleh karakter sinyal chaos yang lebih dibandingkan untuk nilai a < 0.5. Hal ini dapat diperlihatkan ketika dihitung, eksponen ini menunjukkan sensitivitas suatu system terhadap perbedaan nilai awal yang terpaut sangat dekat yang diberikan setelah beberapa kali iterasi. Untuk bentuk eksponen *Lyapunov* dari fungsi *Tent* tersebut berdasarkan persamaan (2.2) akan didapat:

$$\lambda(x) = \lim_{x\to\infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln(2a)$$

Karena terdapat n kali hasil maka bentuk limit akan menjadi

$$\lambda = \ln(2a)$$

Sehingga untuk a > 0.5, perbedaan nilai awal sekecil apapun akan menghasilkan perbedaan yang tidak *periodic* untuk nilai iterasi pada kondisi berikutnya. Pada proyek akhir ini akan dipergunakan factor a = 1, sehingga nilai eksponen L*yapunov* akan menjadi ln(2.1) = ln(2) = 0.693 yang positif, dan terpenuhi syarat sinyal chaos.

Untuk faktor a = 1 persamaan dasar *Tent* akan menjadi:

$$x_{n-1} = \begin{cases} 2ax_n & 0 \le x_n \le 0{,}5 \\ 2(1-x_n) & 0{,}5 \le x_n \le 1 \end{cases}$$

3. PERANCANGAN

Pada bagian ini, sistem yang dirancang adalah suatu sistem untuk mengenkripsi dan mendekripsi suatu file informasi. Pada sistem dilakukan aplikasi dengan menggunakan Matlab.

Dalam sistem ini menggunakan suatu *symmetric cryptography algorithm* dengan menggunakan metode pengacakan yang bersifat *chaotic*. Pemilihan sistem simetris ini dikarenakan kunci yang dipakai di sisi pengirim dan penerima harus sama.

3.1 Kunci User

Kunci *user* disini adalah kunci yang diberikan oleh *user* dalam melakukan proses enkripsi maupun dekripsi. Kunci yang dimasukkan oleh *user* adalah

 ISSN: 1907-5022

berupa karakter-karakter. Dimana karakter-karakter tersebut direpesentasikan menjadi bentuk bilangan-bilangan menurut standar *ASCII*. Pada kunci *user* ini dibatasi oleh minimal jumlah karakter yang dimasukkan, sesuai dengan:

- *Chaos Lorentz system* sebagai *three dimensional chaos*
- *Chaos Julia Set* sebagai *two dimensional chaos*
- *Chaos Tent Function* sebagai *one dimensional chaos*

Untuk *chaos Lorentz*, kunci *user* yang harus dimasukkan minimal 3 karakter. Untuk *chaos Julia* minimal yang harus dimasukkan adalah dua karakter. Sedangkan untuk *chaos Tent* minimal satu karakter yang harus dimasukkan. Ini disesuaikan dengan *input* untuk masing-masing *chaos generator* sebagai kondisi awal *(initial condition)*. Dalam sistem ini maksimal jumlah karakter kunci *user* tidak dibatasi. Kunci *user* untuk tiap-tiap karakter diterjemahkan dulu kedalam bilangan *ASCII*, sesuai dengan komputisasi yang dilakukan.

Kunci *user* pada sistem ini digunakan sebagai *initial condition* pada *chaos generator*.

3.1.1 Inisialisasi kunci *user* untuk *chaos Lorentz*

Initial condition untuk *chaos Lorentz* sebagai *three dimensional chaos* adalah x, y, z. Inisialisasi kunci *user*-nya adalah sebagai berikut:

x = karakter 1 dari kunci * 0,01
y = karakter 2 dari kunci * 0,01
z = (∑(karakter 3 dari kunci sampai karakter ke-n))*(0,01 / (panjang kunci ke-2))

Inisialisasi x,y dan z

3.1.2 Inisialisasi kunci *user* untuk *chaos Julia*

Initial condition untuk *chaos Julia* sebagai *two dimensional chaos* adala x, y. dimana kondisi awal dari suatu *chaos Julia* ini dibentuk menjadi suatu bilangan kompleks sebagai syarat dalam proses pembangkitannya. Inisialisasi kunci *user*-nya adalah sebagai berikut:

a = karakter 1 dari kunci + karakter 2 dari kunci
b = ∑(karakter 3 dari kunci sampai karakter ke-n) inisialisasi a dan b

Dari inisialisasi diatas, maka didapatkan c = a + ib. Sehingga kondisi awal x dan y untuk *chaos Julia* adalah

$$x = \mathrm{Re}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1-4c}\right)$$

$$y = \mathrm{Im}\left(\frac{1}{2}+\frac{1}{2}\sqrt{1-4c}\right)$$

3.1.3 Inisialisasi kunci *user* untuk *chaos Tent*

Initial condition untuk *chaos Tent* sebagai *one dimensional chaos* yang digunakan dalam proses generator adalah x. Inisialisasi kunci *user*-nya adalah sebagai berikut:

x = (∑(karakter 1 sampai karakter ke-n))/(panjang kunci)

sesuai dengan syarat dari *chaos Tent Function*, bahwa untuk nilai dari suatu kondisi awal harus pada range 0 sampai dengan 1. Sehingga:

- Jika x < 100 , maka $x = \frac{x}{100}$
- Jika x > 100 , maka $x = \frac{x}{1000}$
- Jika x>1000, maka $x = \frac{x}{10000}$

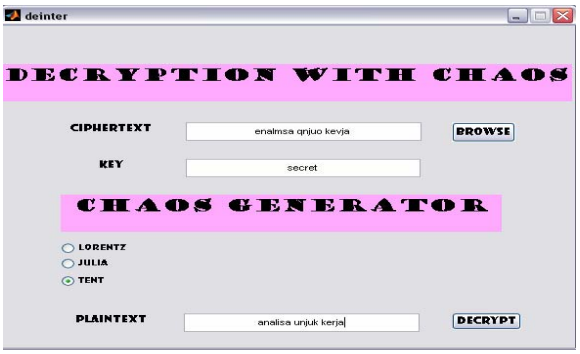
Dari nilai x diatas didapat *initial condition* yang sesuai dengan syarat dari persamaan.

4. IMPLEMENTASI

Tampilan pada sistem enkripsi ini menggunakan fasilitas GUI yang terdapat pada Matlab. Untuk lebih jelasnya dapat kita lihat pada gambar di bawah ini.

Gambar 1. Proses Sistem Enkripsi

Bagian enkripsi ini dimulai pada data *input* yang berupa *plaintext*. *Plaintext* disini dapat lansung dituliskan pada kolom atau dapat menggunakan *menu browse*. Selanjutnya dapat dituliskan kata yang akan dijadikan sebagai *key*. Jika *key* telah dituliskan, maka dapat menggunakan salah satu dari ketiga *Chaos Generator*. Setelah dipilih salah satu dari *Chaos Generator* tersebut, maka langkah terakhir dari proses enkripsi ini adalah data *output* yang menghasilkan *chipertext*.



Gambar 2. Proses Sistem Dekripsi

Sistem Dekripsi ini merupakan kebalikan dari proses enkripsi. Dimulai dengan memberi *input*

berupa *chipertext* yang merupakan hasil dari proses enkripsi yang telah dilakukan. *Chipertext* tersebut juga dapat diperoleh dengan menggunakan *menu browse*. *Key* yang digunakan harus sama dengan *key* yang digunakan pada proses enkripsi. Tahapan selanjutnya sama seperti pada proses enkripsi, yaitu memilih *Chaos Generator*. Kemudian langkah terakhir dari proses dekripsi ini adalah menekan menu *decrypt* agar dapat menghasilkan *plaintext* aslinya.

Output yang dihasilkan dari proses ini ditampilkan pada kolom *plaintext*. *Output* yang dihasilkan harus sama dengan *plaintext* yang digunakan pada proses enkripsi. Jadi, dapat dituliskan bahwa *output* dari proses dekripsi ini merupakan *plaintext* dari proses enkripsi.

5. PENGUJIAN DAN ANALISA

Dari hasil implementasi yang diperoleh, dapat dilakukan beberapa analisa yang berupa, uji *periodic*, *autocorrelation*, analisa sinyal dengan melihat *power spectral density*, serta analisa panjang data *output* dan *avalanche effect* dari ketiga sinyal chaos tersebut. Untuk analisa *autocorrelation*, *power spectral density*, analisa panjang data *output* serta *avallanche effect* digunakan sebagai penunjuk kualitas dari chaos dalam generator sinyal untuk suatu aplikasi kriptografi yang digunakan dalam pembuatan proyek akhir ini.

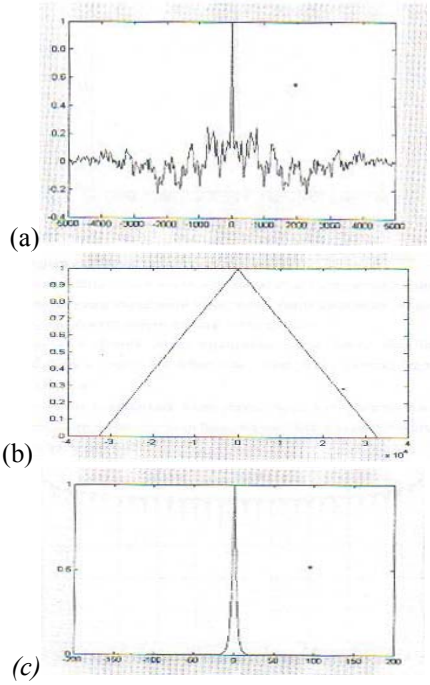
5.1 Uji *Periodic*

Uji *periodic* merupakan pengujian terhadap suatu sinyal untuk membuktikan bahwa sinyal tersebut mempunyai sifat *periodic* pada kondisi tertentu atau sebaliknya sesuai dengan sifat chaos yang *unperiodic*. Penganalisaan dilakukan pada masing-masing metode chaos, dimana menggunakan sinyal *output* yang terbentuk dari nilai-nilai pembangkitnya.

Dari nilai-nilai yang diperoleh menunjukkan sifat yang tidak *periodic*. Sehingga dapat diambil kesimpulan bahwa untuk *chaos Lorentz, Julia* dan *Tent* mempunyai sifat yang *unperiodic* dalam pembangkitannya.

5.2 *Autocorrelation*

Analisa *autocorrelation* melibatkan hasil dari tiap-tiap *chaos generator* yang dipakai, dimana masing-masing hasil/*output* dari masing-masing sinyal tersebut dianalisa terhadap sinyal itu sendiri. *Autocorrelation* merupakan suatu proses penyepadanan suat sinyal terhadap sinyal itu sendiri. *Autocorrelation* menyatakan keterkaitan antara sinyal ke-n dengan sinyal sebelum dan sesudahnya, sehingga dapat merepresentasikan sifat *unperiodic* dari deretan data-data chaos. Representasi dari masing-masing sinyal chaos dapat dilihat pada gambar-gambar di bawah ini:

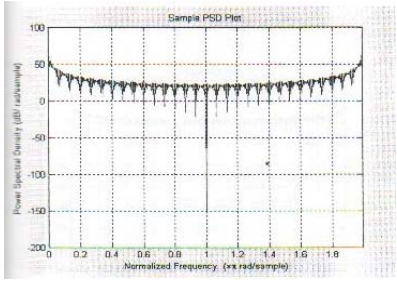


Gambar 3. Autocorrelation Chaos (a) Lorentz, (b) Julia, (c) Tent

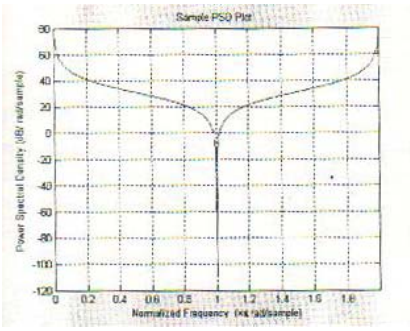
5.3 *Power Spectral Density*

Untuk menganalisa sejauh mana sifat sinyal chaos tersebut mempunyai spectral daya yang *continue* dalam suatu *range* yang lebar, maka dapat dilakukan dengan melihat sinyal *power spectral density* dari *output* masing-masing sinyal chaos tersebut.

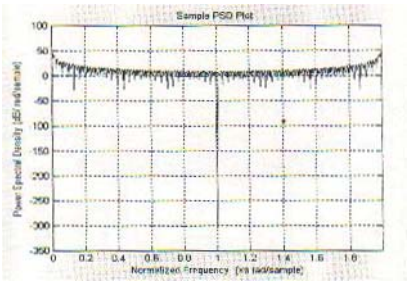
Power spectral density merupakan suatu fungsi dari frekuensi yang mempunyai sifat bahwa luas suatu lebar pita sama dengan daya sinyal X[n] dalam pita tersebut. Hal yang menjadi dasar karakteristik suatu sinyal chaos yang mempunyai *spectral daya* yang *continue* pada suatu lebar pita yang luas, dimana sering dianalogikan sebagai sinyal *noise* pada dunia telekomunikasi.



Gambar 4. Power Spectral Density Chaos Lorentz



Gambar 5. Power Spectral Density Chaos Julia



Gambar 6. Power Spectral Density Chaos Tent

5.4 Avallanche Effect

Salah satu karakteristik untuk menentukan baik atau tidaknya suatu algorithma kriptografi adalah dengan melihat *avalanche effect*-nya. *Avallance effect* merupakan suatu karakteristik dimana perubahan yang kecil terhadap *plaintext* maupun *key* akan menyebabkan perubahan yang signifikan terhadap *ciphertext* yang dihasilkan. Atau bisa juga diartikan bahwa perubahan satu bit pada *plaintext* maupun *key* akan menghasilkan perubahan banyak bit pada *ciphertext.* Menurut Bruce Schneir dalam "*Applied Cryptogrphy*", bahwa suatu *avallance effect* dikatakan baik jika perubahan bit yang dihasilkan berkisar antara 45–60% (50% adalah hasil yang sangat baik). Hal ini menyebabkan perbedaan yang cukup sulit bagi *cryptanalyst* untuk melakukan serangan.

Dari hasil pengujian, *avallanche effect* untuk perubahan kecil terhadap key adalah sebesar 48,66% untuk Lorentz, 45,19% untuk Julia dan 35,18% untuk Tent. Hasil dari analisa terhadap *avalanche effect* terhadap algoritma diatas mengindikasikan bahwa *key* memegang peran yang sangat penting. Dengan perubahan yang kecil terhadap *key*, akan terjadi perubahan yang sangat signifikan terhadap *output*.

5.5 Perbandingan Chaotic Algoritma

Dari pengujian pengujian diatas, dapat dilihat bahwa secara keseluruhan algoritma Lorentz memiliki perfomansi yang lebih baik dibandingkan algoritma Julia dan Tent. Seperti yang terlihat pada tabel 1.

Tabel 1. Perbandingan Chaotic Algoritma

Subjek Analisa	***Algoritma yang lebih baik***
Uji Periodic	Lorentz
Autocorrelation	Tent
Power Spektral Density	Lorentz
Panjang Data Output	Lorentz, Julia
Bit Error	

6. KESIMPULAN

a. Dari ketiga *chaotic algorithm* yang digunakan dan dengan *autocorrelation* yang dihasilkan, diperoleh bahwa algoritma lorentz lebih handal ditinjau dari *output generator* yang lebih *unpredictable* dibandingkan kedua algoritma yang lain.

b. Sinyal chaos mempunyai beberapa karakteristik, yaitu:
- *Unperiodic* dari sinyal yang dihasilkan
- *Sensitive dependence on initial condition* untuk setiap pembangkitannya
- *Autocorrelation* yang dihasilkan maksimal 1
- *Power spectral density* yang dihasilkan cenderung kontinu (sinyal *noise*)

c. Panjang data *output* dari ketiga algoritma tersebut tidak tergantung dari panjang *key*, tetapi bergantung dari panjang *plaintext*.

d. Pada *avallanche effect*, *key* memegan peran yang penting. Dengan perubahan yang kecil pada *key* menyebabkan perubahan yang signifikan terhadap *output*.

PUSTAKA

[1] Soplanit, Susany. 2004. *SCBIE paper*. Jakarta: Universitas Tarumanagara.

[2] Kurniawan, Yusuf. 2004. *Kriptografi: Keamanan Internet dan Jaringan Komunikasi*. Bandung: Penerbit Informatika.

[3] Anasti, Triwindu. 2004. *Implementasi Dekripsi dan Enkripsi File Teks dengan Metode Chaos*. Bandung: Proyek Akhir STTTelkom.

[4] Pamungkas, Angger Ardyanto. *Implementasi Algoritma MD5, SHA1 dan RC4 untuk Sistem Kriptografi pada Aplikasi Mobile Internet Berbasis Java*. Bandung: Tugas Akhir STTTelkom.

[5] Utami, Dewi. 2001. *Pengembangan Perangkat Keras Sistem Kriptografi dengan menggunakan Sinyal Chaos*. Bandung: Tesis Magister ITB.

[6] Away, Gunaidi Abdia. 2006. *The Shortcut of Matlab Programming*. Bandung: Penerbit Informatika.

[7] Lynch, Stephen. 2003. *Dynamical System with Application using Matlab.* Birkhauser.