Studi dan Implementasi dari Teori Chaos dengan Logistic Map sebagai Pembangkit Bilangan Acak Semu dalam Kriptografi

Arnold Nugroho Sutanto - 135007102
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]hey.nold89@gmail.com

***Abstrak*— Makalah ini akan membahas tentang penggunaan Teori Chaos sebagai pembangkit bilangan acak semu (PRNG) dalam Kriptografi. Oleh karena tingkat keamanan kriptografi juga terletak pada bilangan random yang digunakan, algoritma fungsi pembangkitan bilangan acak bukanlah suatu hal yang bisa diaanggap remeh. Untuk itu, diperlukan Teori Chaos yang dianggap mangkus untuk dipakai dalam pembangkitan bilangan acak. Pada makalah kali ini, Persamaan Logistik (Logistic Map) yang didasarkan pada Teori Chaos akan dibahas secara mendetail dan ditunjukkan hasil implementasinya. Selain itu, makalah ini juga akan memperlihatkan algoritma dan pengimplementasian dari Persamaan Logistik sebagai pembangkit urutan bit-bit acak semu yang merupakan dasar dari PRNG. Sebagai kesimpulan, kita akan membahas mengenai kemangkusan dari penggunaan Logistic Map sebagai pembangkit bilangan acak semu dibandingkan metode PRNGs lainnya.**

***Kata kunci*— Teori Chaos, PRNG, Kriptografi, Logistic Map.**

I. PENDAHULUAN

Bilangan acak telah dipakai di berbagai simulasi, eksperiment statistik, dalam metode Monte Carlo untuk analisis numerik, dan tentu saja dalam kriptografi dan steganografi [1]. Hubungan antar pembangkitan bilangan acak dan kriptografi sangatlah dekat karena metode kriptografi modern yang baik selalu menggunakan bilangan acak[5]. Session keys, vektor inisialisasi, salts untuk hash dengan passwords, parameter unik dalam digital signature operations diasumsikan acak oleh pendesain sistem. Selain itu, kriptografi sendiri yang seharusnya dapat menghasilkan chipertext yang terlihat sebagai bit-bit yang benar-benar acak.

Pembangkit bilangan acak yang akan dibahas dalam makalah ini didasarkan pada Teori Chaos. Teori Chaos telah mendapatkan perhatian khusus dari komunitas sains selama 2 dekade terakhir ini [9]. Begitu banyak usaha telah diinvestasikan untuk mencoba menerapkan konsep dari fisika dan matematika ke dalam aplikasi *engineering* dunia nyata. Salah satu aplikasi dari teori Chaos yang terlihat menjanjikan adalah dalam dunia kriptografi. Sifat keacakan dan kedinamikan sistem merupakan kebutuhan utama dalam metode kriptografi. Oleh karena itu, akhir-akhir ini telah banyak diterapkan teori Chaos untuk diimplementasikan sebagai metode kriptografi.

Selain itu, seperti yang telah dijelaskan sebelumnya, Salah satu kegunaan lain dari teori Chaos dalam kriptografi adalah sebagai pembangkit bilangan acak yang digunakan dalam proses pembangkitan kunci. Misal, untuk kunci simetri, metode one time pad dapat menggunakan teori chaos untuk menghasilkan kunci random yang diinginkan. Atau pada metode RSA yang menggunakan bilangan acak dalam pembuatan kunci dekripsi maupun enkripsi. Serangan terhadap PRNGs (Pseudo Random Number Generators) yang digunakan dalam kriptografi memberikan peluang yang cukup besar bagi kriptanalis untuk mendapatkan kunci rahasia [5]. Oleh karena itu, sistem pembangkit bilangan acak dapat menjadi titik lemah yang menentukan tingkat keamanan kriptografi.

II PRNGS DALAM KRIPTOGRAFI

A. Pentingnya Pemilihan Desain PRNGs

Kriptografi yang dirancang dengan baik hampir selalu menggunakan bilangan acak. Sayangnya, banyak aplikasi kriptografi tidak memiliki sumber bit acak yang terpercaya [5]. Sebaliknya, mereka menggunakan mekanisme kriptografi yang disebut Pseudo-Random Number Generator (PRNG) untuk menghasilkan nilai acak ini.

PRNG telah menjadi salah satu jenis primitif dalam kriptografi. Dengan begitu, PRNG dapat menjadi satu titik kegagalan bagi banyak sistem kriptografi. Dalam artian, tidak peduli pemilihan desain algoritma kriptograf, penyerangan terhadap PRNG akan merusak sekuritas dari algoritma tersebut. Apalagi, saat ini banyak sistem yang menggunakan PRNGs dengan rancangan yang buruk atau menggunakannya sehingga membuat berbagai serangan lebih mudah daripada seharusnya.

B. Desain PRNGs yang baik

Menurut [5], sebuah desain PRNG yang baik adalah :

1. PRNG berbasis pada sesuatu yang kuat. Penggunaan primitif kriptografi yang dipercaya kuat mengimplikasikan bahwa sebuah serangan yang sukses juga berarti adalah serangan terhadap primitif tersebut.

2. State dari PRNG berubah setiap waktu. Internal state yang rahasia harus dapat berubah setiap waktu. Hal ini dapat mencegah bahwa sebuah state tunggal dapat tidak dapat direcover kembali.
3. "Catastrophic resseding" dari PNG. Reseeding diperlukan untuk mencegah penyerangan dengan menebak secara iteratif.
4. Menghindari backtraking. PRNG harus didesain untuk menghindari backtracking. Idealnya, hal ini berarti keluaran t tidak bisa ditebak dalam prakteknya untuk penyerang yang mengetahui state PRNG saat waktu time t+1.
5. Dapat menghindari penyerangan dengan menentukan input yang mungkin. Input dari PRNG seharusnya digabungkan dengan state dari PRNG sehingga seorang penyerang yang mengetahui input PRNG, harus mengetahui juga state dari PRNG tersebut. Begitu pula dengan sebaliknya.
6. Pulih dari kompromisasi dengan cepat. PRNG harus dapat memanfaatkan setiap bit entropy yang menjadi masukkannya. Seorang penyerang yang ingin mempelajari effect dari state PRNG dari urutan input harus dapat menebak input keseluruhan.

III. TEORI CHAOS

A. Definisi Teori Chaos

Teori Chaos adalah suatu sub-disiplin matematika yang mempelajari sistem kompleks. Sistem kompleks yang dimaksud adalah sistem yang berisi gerakan begitu banyak (sehingga banyak elemen yang bergerak) yang memerlukan komputer untuk menghitung semua berbagai kemungkinan [2]. Contoh dari sistem yang kompleks yang membantu untuk memahami Teori Chaos adalah sistem cuaca bumi atau perilaku air mendidih di atas kompor. Sistem Chaos terdapat di mana-mana, dari pertimbangan alam yang paling intim untuk seni apapun.

Saat ini, masih belum terdapat definisi yang pasti mengenai Chaos itu sendiri. Meskipun begitu, kebanyakan orang akan setuju mengenai definisi berikut ini. Chaos adalah perilaku waktu asimtotik dan aperiodik dalam sistem deterministik yang menunjukkan ketergantungan sensitif terhadap kondisi awal [2].

Hal yang mendasar tentang teori Chaos adalah perubahan terkecil dalam sistem dapat menyebabkan perbedaan yang sangat besar dalam perilaku yang sistem. Efek kupu-kupu adalah salah satu gambar yang paling populer untuk menggambarkan Chaos. Idenya adalah bahwa seekor kupu-kupu yang mengepakkan sayap di Argentina dapat menyebabkan tornado di Texas tiga minggu kemudian. Sebaliknya, dalam sebuah salinan identik dunia tanpa kupu-kupu Argentina, tidak ada badai seperti itu telah muncul di Texas. Versi matematika dari properti ini dikenal sebagai ketergantungan yang sensitif.

B. Sifat Chaotic

Selain menunjukkan ketergantungan yang sensitif, sistem chaotic memiliki dua sifat lainnya: mereka deterministik dan nonlinier [6].

- ✓ Perilaku aperiodik waktu asimtotik: Perilaku ini berarti keberadaan lintasan fase-ruang yang tidak menetap untuk poin tertentu atau orbit periodik.
- ✓ Deterministik: Hal ini berarti bahwa persamaan gerak
  sistem tidak memiliki masukkan acak. Dengan kata lain, perilaku tidak beraturan sistem muncul dari dinamika non linear.
- ✓ Ketergantungan sensitif pada kondisi awal: Seperti yang telah dijelaskan sembelumnya, sifat ini berarti bahwa lintasan terdekat di fase-ruang terpisah secara kecepatan eksponensial dalam waktu: yaitu, sistem memiliki eksponen Liapunov positif [2].

C. Peta-peta Chaos

Dalam matematika, sebuah peta chaotic berarti peta yang memiliki beberapa jenis perilaku chaotic. Peta tersebut diparameterisasi dengan waktu-diskrit atau parameter kontinu-waktu[2]. Peta diskrit biasanya mengambil bentuk fungsi iterasi. Peta Chaotic sering ditemukan dalam penelitian sistem dinamik. Contoh dari peta chaotic antara lain Peta Henon, Tent Map, Lorenz attractor, dan Peta Logistic. Pada bagian ini, akan dijelaskan mengenai peta-peta ini secara singkat kecuali persamaan Logistik yang akan dijabarkan lebih lanjut lagi dalam bagian lain.

Persamaan Tent -- Dalam matematika, tent map adalah fungsi iterasi, dalam bentuk tenda, membentuk suatu sistem dinamik berbasis waktu diskrit. Dibutuhkan titik Xn pada garis real dan kemudian memetakan titik tersebut ke titik lain:

$$x_{n+1} = \begin{cases} \mu x_n, & x < 1/2 \\ \mu(1 - x_n), & x \geq 1/2 \end{cases} \quad (1)$$

Di mana μ bernilai positif konstan.

Lorenz Attractor, yang diberi nama sesuai dengan penemunya Edward N. Lorenz, adalah struktur 3-dimensi yang bersesuaian dengan perilaku jangka panjang dari aliran chaotic, terkenal oleh karena bentuk kupu-kupunya.

Persamaan yang terdapat dalam Lorenz attractor :

$$\frac{dx}{dy} = \sigma(y - x)$$
$$\frac{dy}{dt} = x(\rho - z) - y$$
$$\frac{dz}{dt} = xy - \beta z \quad (2)$$

Di mana σ disebut sebagai nomor *Prandtl* dan ρ disebut nomor *Rayleigh*. Semua $\sigma, \rho, \beta > 0$, tetapi biasanya $\sigma = 10$, $\beta = 8/3$, dan ρ bervariasi. Sistem ini memiliki perilaku chaotic saat $\rho = 28$.

Persamaan Henon -- Peta Henon adalah peta prototipikal 2-D teriterasi dan terinvertible dengan solusi chaotic yang diusulkan oleh astronom Perancis Michel

Henon sebagai model sederhana dari peta Poincare untuk model Lorenz [3]:

$$x_{n+1} = 1 + ax_n^2 + bx_{n-1} \quad (3)$$

Parameter b adalah ukuran dari tingkat kontraksi daerah (disipasi), dan peta Henon adalah peta kuadratik 2-D yang paling umum dengan properti yang kontraksinya independen dari x dan y. Untuk b = 0, peta Henon berkurang menjadi peta kuadrat, yang merupakan konjugasi ke peta logistik. Terdapat solusi terbatas untuk persamaan Henon selama rentang nilai a dan b, dan sebagian dari rentang ini (sekitar 6%) menghasilkan solusi *chaotic*.

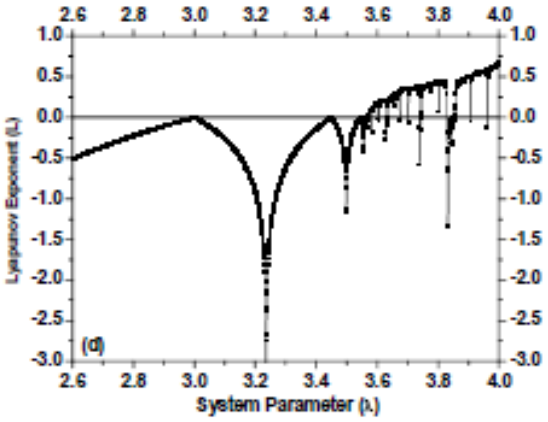
IV. PERSAMAAN LOGISTIK

Persamaan Logistik (Logistic Map) adalah salah satu bentuk yang paling sederhana dari proses chaotic. Pada tahun 1976 Mei [11] menunjukkan bahwa model sederhana ini menunjukkan perilaku yang kompleks. Kemudian Fiegenbaum melaporkan beberapa fitur universal kuantitatif, yang menjadi ciri dari studi kontemporer dari Chaos. Karena kesederhanaan matematisnya, model ini terus memuncukan ide-ide baru dalam teori chaos serta aplikasi kekacauan dalam kriptografi [11]. Berikut adalah persamaan dari peta logistik.

$$X_{n+1} = \lambda X_n (1 - X_n) \quad (4)$$

Di mana Xn adalah variabel state, yang terletak di interval [0,1] dan α disebut sebagai parameter sistem yang dapat memiliki nilai antara 1 dan 4.

Pada dasarnya, peta ini, seperti peta satu-dimensi lainnya, adalah aturan untuk mendapatkan sebuah bilangan dari bilangan lain. Parameter adalah tetap, tetapi jika orang mempelajari peta untuk nilai yang berbeda dari λ (hingga 4, interval unit lain tidak lagi invarian), akan ditemukan bahwa adalah katalis untuk sifat *chaotic*.



Gambar 1 Perilaku dari Logistic Map : Lyapunov exponent (pengukuran kuantiatif dari sifat chaos) sebagai fungsi dari parameter α

Pada gambar di atas, dibuat sebuah grafik dari persamaan Lyapunof yang mampu mengukur sifat ke-chaos-an secara kuantitatif. Nilai Lyapunof yang bernilai positif menunjukkan bahwa persamaan tersebut memiliki sensitivitas yang tinggi terhadap nilai awal. Dari gambar 1, dapat dilihat bahwa persamaan yang menghasilkan sifat chaos terbesar adalah saat λ bernilai sekitar 4. Logistic Map tidak akan bersifat chaos saat λ bernilai < 3.559 yang menghasilkan nilai Lypunof negatif. Menurut perhitungan yang telah dilakukan, dapat disimpulkan bahwa logistic map memberikan karakter chaos terbaik saat α bernilai sangat dekat dengan 4.

V. PRBG MENGGUNAKAN PERSAMAAN LOGISTIK

A. Pseudo Random Bit Generator (PRBG)

A bit generator acak (RBG) adalah sebuah perangkat atau algoritma, yang memiliki output urutan yang secara statistik independen dan digit biner unbias [11]. Generator tersebut memerlukan sumber keacakan alami (non-deterministic). Dalam lingkungan yang paling praktis merancang perangkat keras atau perangkat lunak untuk mengeksploitasi sumber keacakan alami serta menghasilkan urutan bit bebas dari bias dan korelasi adalah hal yang sulit. Dalam situasi demikian, masalah dapat diperbaiki dengan mengganti generator bit acak dengan pseudo random bit generator (PRBG).

A pseudo random bit generator (PRBG) adalah algoritma deterministik, yang menggunakan urutan biner benar-benar acak dengan panjang k yang disebut sebagai *seed* masukan dan menghasilkan urutan biner dengna panjang l >> k, yang disebut urutan bit acak semu (tampaknya acak). Output dari PRBG tidaklah benar-benar acak, bahkan jumlah urutan output yang mungkin adalah paling hanya sebagian kecil fraksi dari semua kemungkinan urutan biner dengan panjang l. Maksud dasarnya adalah untuk mengambil sebagian kecil urutan bit yang benar-benar acak dengan panjang k dan memperluas ke sebuah urutan bit yang jauh lebih besar dengan panjang l sedemikian rupa sehingga musuh tidak dapat dengan efisien membedakan antara urutan output dari PRBG dan urutan yang benar-benar acak dengan panjang l. Urutan bit yang dihasilkan oleh PRBG ini akan dijadikan angka-angka sehingga dengan mudahnya kita mendapatkan PNBG dari PRBG.

B. Algoritma PRBG dengan Persamaan Logistik

Algoritma ini dikemukakan oleh Vinod Patidar and K. K. Sud pada tahun 2008 [11]. PRBG yang diusulkan didasarkan pada dua peta logistik, didasarkan pada dua peta logistik, mulai kondisi awal independen yang acak $(X_0, Y_0 \in [0,1] \; dan \; X_0 \neq Y_0)$.

$$X_{n+1} = \lambda_1 X_n (1 - X_n), \quad (5)$$

$$Y_{n+1} = \lambda_2 Y_n (1 - Y_n) \quad (6)$$

Urutan bit dihasilkan dengan membandingkan output dari kedua peta logistik dengan cara berikut :

$$g(X_{n+1}, Y_{n+1}) = \begin{cases} 0, & X_{n+1} > Y_{n+1} \\ 1, & X_{n+1} \leq Y_{n+1} \end{cases}$$

Kondisi awal (X_0, Y_0) merupakan seed dari PRBG, jika seed tersebut sana dari seed sebelumnya, urutan bit yang

dihasilkan akan 100% sama oleh karena prosedur deterministik di atas. Blok diagram dari algoritma ini adalah :

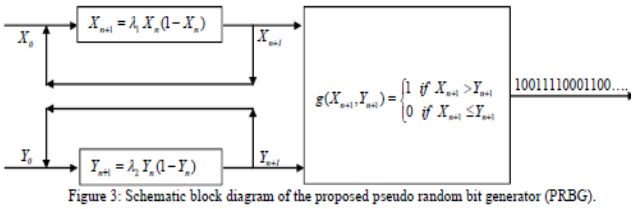


Figure 3: Schematic block diagram of the proposed pseudo random bit generator (PRBG).

Gambar 2 Blok Diagram dari PRBG yang diusulkan [11]

Dalam studi analisis baru-baru ini Li et [4] menunjukkan bahwa urutan biner yang dihasilkan dengan cara membandingkan keluaran dari dua Chaos Map akan memiliki sifat kriptografi sempurna jika persyaratan berikut dipenuhi:

i. Kedua peta harus menghasilkan lintasan asimtotik independen saat n → ∞,
ii. kedua peta surjective pada interval yang sama,
iii. kedua peta memiliki distribusi $P_1(x)$ dan $P_2(x)$ y dengan densitas invarian dan ergodic pada interval yang ditetapkan,
iv. baik $P_1(x) = P_2(x)$ atau $P_1(x)$ dan $P_2(x)$ simetri pada poin tengah dari interval.

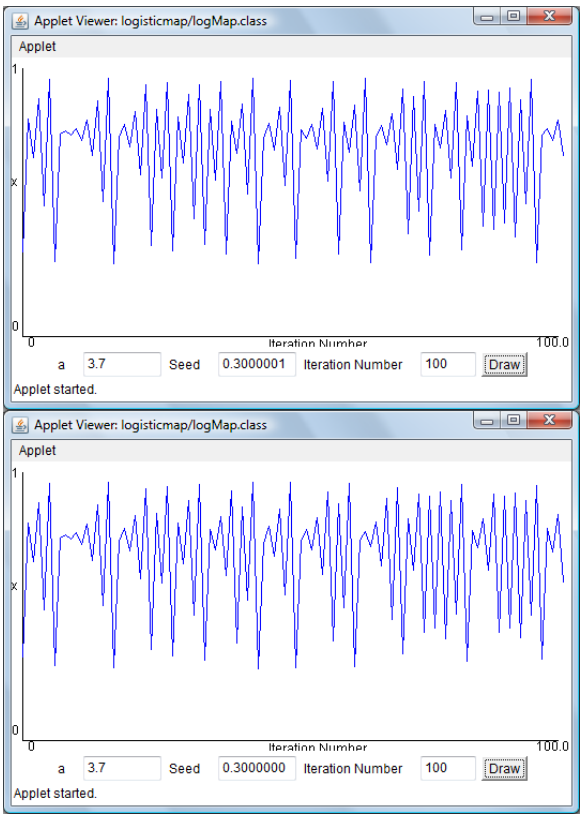
Peta logistik memiliki semua properti di atas ketika peta logistik menunjukkan sifat chaotic[11]. Untuk memenuhi kondisi ke (ii), kita harus memilih nilai λ untuk kedua peta logistik yang dipakai. Berdasarkan penjelasan di bagian IV, nilai λ yang sebaiknya dipilih sangat mendekati nilai 4.0 agar seed mendapat interval yang besar. Jika ingin memilih λ selain 4.0, diperlukan analisis yang cermat dari eksponen Lyapunov. Semakin besar nilai eksponen Lyapunov, semakin kecil kolerasi antar peta yang digunakan.

VI IMPLEMENTASI LOGISTIC MAP

A. Hasil Implementasi Persamaan Logistik

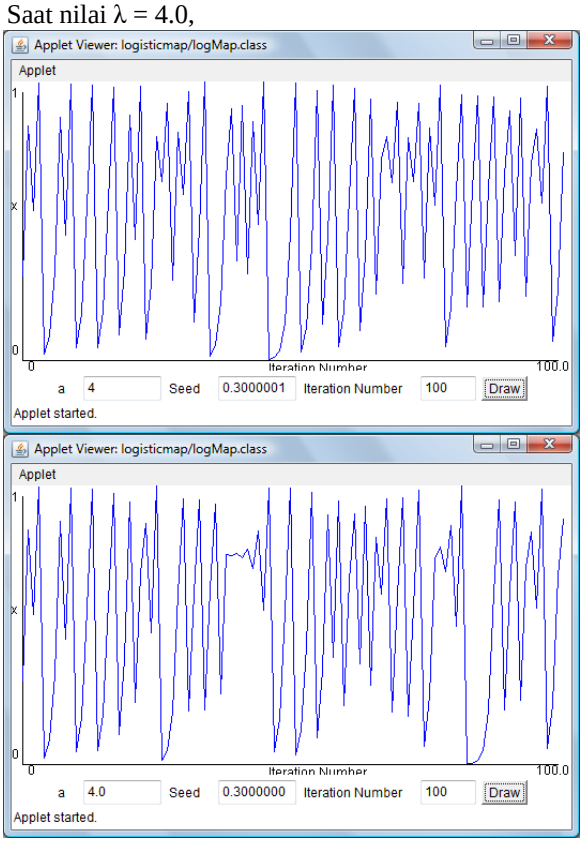
Implementasi yang dilakukan menggunakan bahasa Java dan dibuat dalam bentuk Java Applet. Berikut akan ditunjukkan hasil implementasi dari Logistic Map berupa grafik nilai x dan perubahan grafik yang terjadi menggunakan nilai λ yang berbeda-beda. Nilai X_0 (seed) yang digunakan adalah 0.3000001 dan 0.3 Sebagai catatan, source code menggunakan acuan dari [8]

Saat nilai λ = 3.7



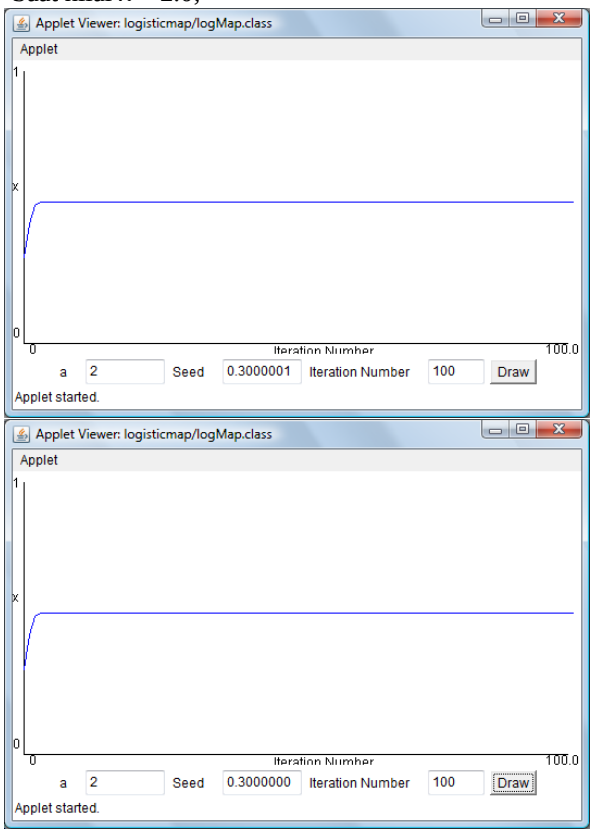
Gambar 3 Map yang dihasilkan dengan nilai λ =3.6 : seed = 0.300001 (atas), seed = 0.3 (bawah)

Saat nilai λ = 4.0,



Gambar 4 Map yang dihasilkan dengan nilai λ =4.0 : seed = 0.300001 (atas), seed = 0.3 (bawah)

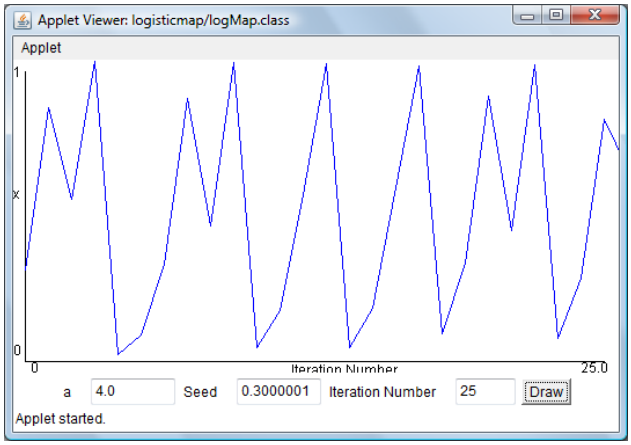
Saat nilai λ = 2.0,

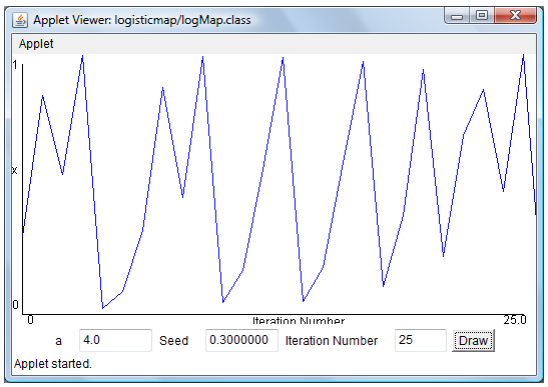


Gambar 5 Map yang dihasilkan dengan nilai λ =2.0 : seed = 0.300001 (atas), seed = 0.3 (bawah)

B. Analisis Hasil Implementasi

Sesuai dengan teori yang telah dijelaskan sebelumya, tampak bahwa pemilihan nilai λ sangat mempengaruhi sifat chaotic dari peta yang dihasilkan. Pemilihan λ harus memperhatikan grafik exponential Lipunov yang mengukur keacakan peta secara kuantitaif. Akan tetapi, dapat kita lihat bahwa pemilihan λ = 4.0 (yang memiliki nilai Lipunov terbesar) menghasilkan peta yang tidak seberapa berubah pada awalnya. Hal ini dapat kita lihat dengan lebih jelas melalui gambar berikut.





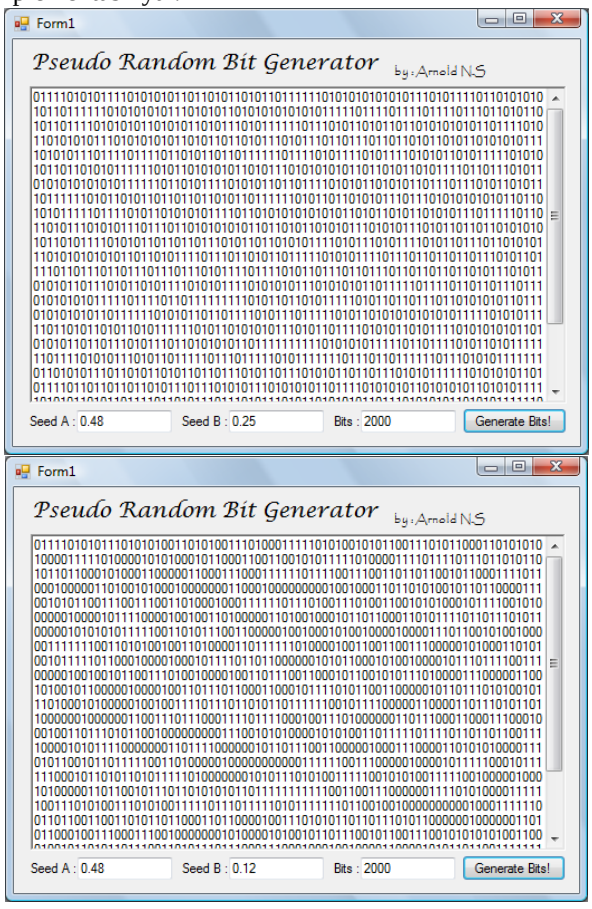
Gambar 6 Map yang dihasilkan dengan nilai λ =4.0 dengan iterasi 25 kali : seed = 0.300001 (atas), seed = 0.3 (bawah)

Sebagaimana diperlihatkan dengan gambar di atas, pada kira-kira 20 iterasi pertama tidak terjadi perubahan yang signifikan. Hal ini akan kita perjelas saat implementasi dari PRBG.

VII IMPLEMENTASI PRBG DENGAN LOGISTIC MAP

A. Hasil Implementasi PRBG

PRBG diimplemntasikan menggunakan bahasa C# berbasis .NET Framework. Hasil implementasi berupa Windows Desktop Application. Berikut adalah hasil implementasinya :



Gambar 7 Urutan Bit yang dihasilkan : Seed pertama 0,48 dan Seed kedua 0,25(atas); Seed Pertama 0,48 dan Seed kedua 0.13(bawah)

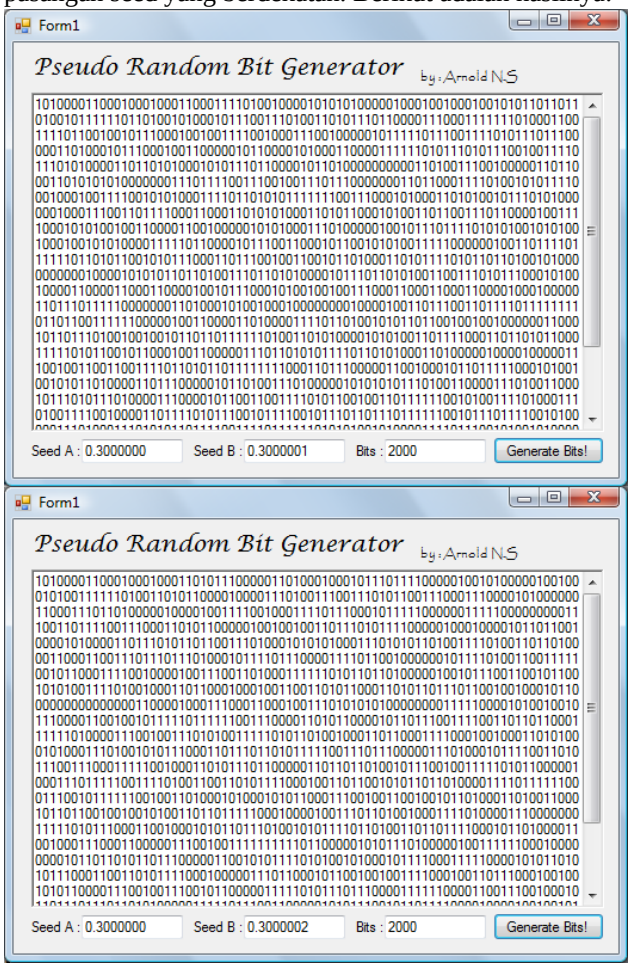
Untuk lebih jelasnya, berikut adalah beberapa bit pertama

yang dihasilkan pada kedua pasangan seed di atas :

A : 01111010101111010101011011010110101101111111010

B : 01111010101110101010011010100111010001111110101

Pada percobaan kedua kita mencoba menggunakan pasangan seed yang berdekatan. Berikut adalah hasilnya:



Gambar 8 Urutan Bit yang dihasilkan : Seed pertama 0,3 dan Seed kedua 0,3000001(atas); Seed Pertama 0,3 dan Seed kedua 0.3000002(bawah)

Beberapa bit pertama yang dihasilkan pada kedua pasangan seed di atas :

A : 10100001100010001000110001111010010000101010

B : 10100001100010001000110101110000011010001000 10

B. Analisis Hasil Implementasi

Dapat kita lihat bahwa implementasi dari PRGB dengan menggunakan Logistic Map menghasilkan uratan bit yang memang terlihat acak. Akan tetapi, karena sifat dari Logistic Map, bit-bit awal hasil implementasi ini terlihat sama dengan pasangan seed yang berbeda namun berdekatan. Seperti bit-bit awal yang dihasilkan oleh pasangan seed 0.3 dan 0.3000001 (Pasangan A) dengan pasangan seed 0.3 dan 0.3000002 (Pasangan B):

A : **1010000110001000100011**0001111010010000101010

B : **1010000110001000100011**0101110000011010001000

Jumlah bit yang sama ini akan berbanding lurus dengan seberapa dekat pasangan seednya. Misal pasangan seed 0.3000000000001 dan 0.3 dengan pasangan seed 0.3000000000002 dan 0.3 akan memberikan jumlah bit-bit awal yang sama lebih dari pasangan A dan B.

Bit-bit awal yang dihasilkan oleh PRBG ini tidak terlalu acak sehingga jika bit-bit ini digunakan dapat diserang dengan mudah yaitu dengan menebak-nebak pasangan input seed dengan tingkat ketelitian yang kecil. Oleh karena itu, sebaiknya pseudo random bit yang digunakan dimulai dari bit yang dihasilkan pada iterasi 100 ke atas (lebih besar maka lebih baik). Semakin besar urutan iterasi yang digunakan, semakin tinggi tingkat ketelitian seed yang perlu dipakai. Hal ini diperlukan agar jika dilakukan brute force attack terhadap PRNG ini diperlukan tingkat ketelitian sangat besar yang menghasilkan kemungkinan pasangan seed yang sangat luas pula.

VIII. KESIMPULAN

Peta Logistik dan PRBG yang dikemukakan oleh Vinod Patidar and K. K. Sud telah berhasil diimplementasikan. PRBG ini dapat dijadikan sebagai desain PRNG yang baik. Hal ini dapat dipastikan karena selain memenuhi syarat kriptografi sempurna (untuk menghasilkan bilangan pseudo random yang baik), desain PRNG ini juga memenuhi keenam syarat desain PRNGs yang dikemukakan dalam [5]. PRNG ini memiliki fitur-fitur seperti

- ✓ Reseeding secara terus menerus untuk menghindari penebakan secara iteratif
- ✓ Tidak bisa dilakukan backtracking
- ✓ State yang berubah setiap waktu
- ✓ Sifat ketergantungan terhadap input masukkan (seed) yang sangat tinggi (sifat dari teori chaos)

Adanya fitur-fitur ini menghasilkan desain PRNG yang mangkus. Akan tetapi, bit-bit yang digunakan sebaiknya dimulai setelah iterasi ke 100. Hal ini diperlukan karena sifat chaos dari persamaan logistic yang baru muncul setelah beberapa iterasi.

Keunggulan maupun kekurangan dari PRNG dengan desain Logistic Map ini masih belum bisa dibandingkan dengan PRNG yang lain. Akan tetapi, perlu diingat bahwa menurut [5], sudah terdapat kriptanalisis yang dapat dilakukan untuk PRNG yang ada sekarang seperti ANSI X9.17 PRNG, DSA PRNG, dan RSAREF PRNG. Oleh karena itu, PRNG dengan Logistic Map ini perlu dieksplorasi lebih lanjut untuk disempurnakan sehingga dapat digunakan sebagai pembangkit bilangan acak semu yang mangkus dalam Kriptografi.

REFERENCES

[1] R. Wagner. Near. *The Laws of Cryptography with Java Code*, , University of Texas, 2003.

[2] Ipek Zeynep, *Chaos and Chaotic Maps.* Belmont, Cankaya University, 2009

[3] J.C.Sprott, *Henon Map Correlation Dimension*. Department of Physics, University of Wisconsin, Madison, 1997.

[4] Li, S., Mou, X. & Cai, Y. Pseudo-random bit generator based on couple chaotic systems and its application in stream-ciphers cryptography. In Progress in Cryptology – INDOCRYPT 2001, Lecture Notes in Computer Science, vol. 2247, 316–329, 2001.E. H. Miller, “A note on reflector arrays (Periodical style—Accepted

for publication)," *IEEE Trans. Antennas Propagat.*, to be published.
[5] Kesley, John, *Cryptanalytic* Attacks *on Pseudorandom Number Generators*.
[6] http://plato.stanford.edu/entries/chaos/#WhaChaThe diakses pada tanggal 14 Mei 2010
[7] http://www.abarimpublications.com/ChaosTheoryIntroduction.html diakses pada tanggal 14 Mei 2010
[8] http://math.la.asu.edu/~chaos/logistic.html diakses pada tanggal 16 Mei 2010
[9] Cristea, Bogdan., Cehan, Constantin. *Applications of Chaos Theory in Cryptography*.
[10] Rinaldi Munir, Bambang Riyanto, dan Sarwono Sutikno. *Perancangan Algoritma Kriptografi Stream Cipher dengan Chaos*, ITB, Bandung, 2006.
[11] V. Patidar., K.K.Sud. *A Pseudo Random Bit Generator Based on Chaotic Logistic Map and its Statistical Testing*, Sir Padmapat Singhania University, Bhatewar, Udaipur – 313 601, India, 2008.

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, 29 April 2010

Arnold Nugroho S
13507102