Perbandingan Penggunaan Teknik Pembangkitan Bilangan Random *Keystream Generator* dengan Teknik *Chaos* pada *Stream cipher*

Ratna Ekasari Prihandini - 13504043
Jurusan Teknik Informatika, Sekolah Teknik Elektro dan Informatika, ITB, Bandung, email: prihandini.re@google.com

Abstract - *Kunci merupakan bagian terpenting dalam menjamin kekuatan kriptografi. Pada stream cipher kunci direpresentasikan sebagai kumpulan bilangan acak yang disebut dengan keystream. Cara untuk membangkitkan keystream adalah dengan menggunakan keystream generator. Keystream yang dibangkitkan oleh keystream generator ini memiliki kelemahan sehingga tingkat keamanannya kurang baik. Untuk itu, dibutuhkan suatu teknik lain yang berbeda sifat dengan keystream generator. Dalam makalah ini dipilih teknik chaos dan akan dibandingkan kekuatan penggunaanya dengan keystream generator pada stream cipher. Dari hasil perbandingan disimpulkan bahwa tingkat keacakan bilangan hasil dari teknik chaos lebih tinggi dibandingkan dengan hasil dari keystream generator.*

Kata Kunci : *stream cipher, chaos, kriptografi*, *keystream generator*

1. PENDAHULUAN

Kunci merupakan bagian terpenting dalam menjamin kekuatan kriptografi. Kunci berupa bilangan biasanya dibuat sangat *random* atau acak agar tidak mudah dipecahkan. Salah satu penggunaan bilangan acak ini adalah pada algoritma *stream cipher* dimana setiap karakter akan dienkripsi dengan menggunakan bit 1 atau 0. Teknik pembangkitan bilangan acak pada *stream cipher* yaitu dengan menggunakan *keystream generator*, termasuk *pseudo random number generator* dengan penggunaan *seed* agar hasilnya benar-benar acak. Namun, penggunaan teknik pembangkitan bilangan *random* ini dirasa masih kurang kuat karena masih terjadi perulangan pada suatu saat (periode terbatas) sehingga cipherteks dapat dipecahkan. Untuk itu digunakanlah teknik pembangkitan bilangan random lain yang dirasa lebih kuat karena dapat menghasilkan bilangan random yang benar-benar acak, yaitu dengan menggunakan teknik *chaos*.

Dalam makalah ini akan dibahas mengenai perbandingan penggunaan teknik pembangkitan bilangan random *keystream generator* dengan teknik *chaos* pada *stream cipher.* Diukur dari berbagai parameter untuk disimpulkan tingkat keacakan bilangannya.

2. STREAM CIPHER

Stream cipher merupakan teknik kriptografi yang melakukan enkripsi terhadap file dalam bentuk bit 1 dan 0 (manipulasi per bit) atau *byte* per *byte*. Setiap bit dari plainteks akan dilakukan manipulasi mengalir dari bit awal sampai dengan bit akhir. *Stream cipher* mempunyai dua bentuk yaitu bit cipher dan block cipher.

Bit cipher merupakan jenis *stream cipher* yang melakukan enkripsi per bit untuk plainteks sedangkan block cipher melakukan enkripsi per blok untuk plainteksnya. Setiap blok dienkripsi dengan menggunakan bit-bit kunci yang memiliki panjang yang sama dengan panjang blok. Secara matematis proses enkripsi dan dekripsi dapat digambarkan sebagai berikut :

$$E_K(P) = C \text{ dan } D_K(C) = P \qquad (1)$$

Fungsi E haruslah fungsi yang berkoresponden satu-ke-satu, sehingga

$$E^{-1} = D \qquad (2)$$

Kunci k yang digunakan untuk melakukan enkripsi dan dekripsi dibangkitkan secara acak menggunakan *keystream generator* dengan menggunakan umpan (*seed*) berupa masukan dari pengguna.

Algoritma *stream cipher* pada dasarnya mengadopsi *one time pad* (OTP) yaitu proses enkripsi dan dekripsi hanya dilakukan satu karakter setiap kali. Untuk itu keamanan sistem *cipher* aliran bergantung seluruhnya pada *keystream generator*. Semakin acak keluaran yang dihasilkan oleh pembangkit aliran-bit-kunci, semakin sulit kriptanalis memecahkan cipherteks. Terdapat tiga jenis keluaran dari *keystream generator* :

a. Jika pembangkit mengeluarkan aliran-bit-kunci yang seluruhnya nol, maka cipherteks sama dengan plainteks, sebab $ci = pi \oplus 0 = pi$ dan proses enkripsi menjadi tak berarti.
b. Jika pembangkit mengeluarkan *kesytream* yang berulang secara periodik, maka algoritma enkripsinya sama dengan algoritma enkripsi dengan XOR sederhana yang memiliki tingkat keamanan yang tidak berarti.

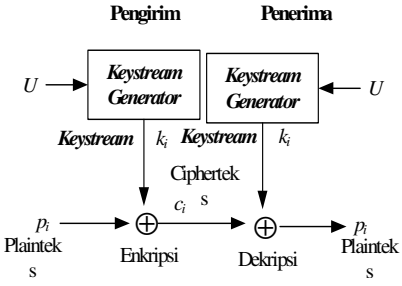
c. Jika pembangkit mengeluarkan *keystream* benar-benar acak (*truly random*), maka algoritma enkripsinya sama dengan *one-time pad* dengan tingkat keamanan yang sempurna. Pada kasus ini, panjang *keystream* sama dengan panjang plainteks, dan kita mendapatkan *cipher* aliran sebagai *unbreakable cipher.*

3. *KEYSTREAM GENERATOR* DAN TEKNIK *CHAOS*

3.1 *Keystream Generator*

Keystream Generator merupakan prosedur pembangkit bilangan-bilangan acak yang akan digunakan sebagai kunci untuk proses enkripsi dan dekripsi yang sama baik disisi pengirim maupun penerima.

Bentuk dari *keystream* atau kunci dapat berupa bit-bit atau blok-blok bit. Proses enkripsi dan dekripsi dengan menggunakan *keystream* dapat dilihat pada Gambar 1

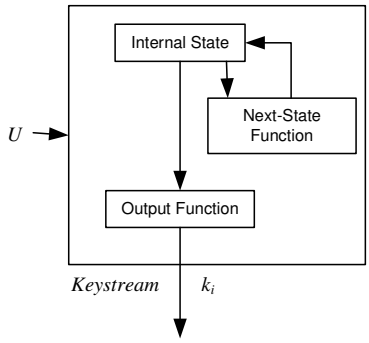


Gambar 1 : Proses Enkripsi Dekripsi dengan *Keystream Generator*

Keterangan :
U : Kunci masukan dari pengguna

Proses enkripsi dan dekripsi dilakukan dengan kunci k_i yang merupakan kunci internal hasil pembangkitan dari *keystream generator.*

Proses didalam *keystream generator* sendiri dapat dilihat pada Gambar 2



Gambar 2 : Proses dalam *Keystream Generator*

Fungsi yang didefinisikan dalam *keystream generator* dapat berupa fungsi apapun.
Contoh yang paling sederhana adalah fungsi XOR. Karena *U* konstan, maka *keystream* yang dihasilkan pada setiap baris tidak berubah jika bergantung hanya pada *U*.

Ini berarti pembangkit aliran-bit-kunci tidak boleh mulai dengan kondisi awal yang sama supaya tidak menghasilkan kembali *keystream* yang sama pada setiap baris. Untuk mengatasinya pembangkit menggunakan *seed* atau umpan sebagai masukan kedua terhadap *keystream generator.* Nilai umpan ini berbeda-beda untuk setiap barisnya. *Keystream generator* dengan penambahan umpan dan hasil dari fungsi matematis ini sering disebut dengan *Pseudo Random Number Generator* (PRNG).

3.2 Teknik *Chaos*

Teknik *chaos* merupakan teknik pembangkitan bilangan random dengan menggunakan rumus polinomial atau eksponensial seperti persamaan logistik di dalam ekologi yang digunakan untuk menghitung pertumbuhan populasi suatu spesies. Dasar dari teknik *chaos* ini adalah fungsi-fungsi yang menggambarkan sistem dinamis yang kompleks, *low-dimensional* dan *non-linier* serta kelakuannya sulit diprediksi. Secara matematis digambarkan sebagai :

$$X_{n+1} = F(X_n) \qquad (3)$$

Sifat dari teknik *chaos* yang sangat cocok terhadap kriptografi yaitu :

a. Sangat sensitif terhadap nilai awal, artinya perbedaan sangat kecil pada nilai awal dapat mengakibatkan perbedaan kunci yang dihasilkan sangat signifikan.
b. Deterministik, artinya dimungkinkan membangkitkan nilai-nilai *chaos* dengan kepastian.
c. Hubungan statistik dapat disembunyikan antara kunci yang satu dengan kunci yang lain dapat disembunyikan dengan baik.

4. IMPLEMENTASI *STREAM CIPHER* DENGAN TEKNIK *CHAOS*

Dalam *stream cipher* teknik *chaos* ini digunakan untuk menggantikan *keystream generator* yaitu untuk membangkitkan aliran kunci bilangan acak selanjutnya kunci tersebut digunakan untuk proses dekripsi dan enkripsi.

Hasil dari nilai *chaos* berupa nilai real (*continous*), disisi lain komputer memproses enkripsi dan dekripsi dalam bentuk bilangan diskrit 1 dan 0. Untuk itu diperlukan suatu fungsi lain yang akan mengkonversi nilai hasil *chaos* menjadi berbentuk integer.

Untuk menambah kompleksitas dari algoritma diperlukan suatu fungsi yang akan menentukan jumlah iterasi pada saat men-*generate* tiap kunci sehingga iterasinya berbeda-beda, ini akan mengakibatkan hubungan statistik antar kunci semakin dapat disembunyikan.

Pseudo code dari algoritma *stream cipher* dengan teknik *chaos* adalah sebagai berikut :

```
ArrayPlain ← ReadFile (input_file)
    Key ← input_key

    For i=0 to ArrayPlain.length do
       Key ← Generate_Key (Key)
       Keystream ← Konvers_Bit (Key)
       ArrCipher [i] ←
         Enkrip(ArrayPlain[i],Keystream)
    End

    WriteFile (ArrCipher)
```

Pseudo code dari fungsi *generate key* adalah sebagai berikut :

```
Function Generate_Key (input_key) → Integer
  Iterate ← Konvers_Int (input_key) +
                sizeOfInput_key
  Key ← input_key
  For i = 0 to Iterate do
        Key = Chaos_Func (Key)

  Output → Konvers_Int (Key)
```

Contoh fungsi *chaos* yang dapat digunakan diantaranya :

a. Persamaan logistik (*logistic map*) :
   $f(x) = r\,x(1 - x)$ (4)
b. *Henon map* :
   $xn = 1 + b(xn - 2 - xn - 3) + cx2n - 2$ (5)
c. *Arnold's cat map*:

$$f(x,y)=\begin{cases} (2x, y/2), & 0\le x\le 1/2, 0\le y\le 1 \\ (2x-1,(y+1)/2), & 1/2\le x\le 1, 0\le y\le 1 \end{cases} \quad (6)$$

Fungsi konversi nilai *chaos* menjadi integer dapat dilakukan dengan menggunakan fungsi pembulatan ke atas atau pembulatan ke bawah. Agar lebih rumit dapat digunakan fungsi matematis lain bergantung pada perancang. Contoh fungsi konversi integer yang diusulkan oleh [2]

$$T(x, size) = \left\| x * 10^{count} \right\|, x \neq 0 \quad (7)$$

Keterangan :
Count dimulai dari 1 dan bertambah 1 sampai $x * 10^{count} > 10^{size-1}$. Hasilnya hanya diambil nilai integernya saja.

Dari hasil implementasi didapatkan bahwa dengan menggunakan fungsi logistik dan fungsi pembulatan kebawah perbedaan nilai x yang sangat kecil (0,000001) antara x dekripsi dan enkripsi tidak akan menghasilkan plainteks semula.

5. PERBANDINGAN *KEYSTREAM GENERATOR* DAN TEKNIK *CHAOS*

Perbandingan antara *Keystream Generator* dan Teknik *Chaos* dalam pembangkitan bilangan acak untuk kunci dapat dilihat pada Tabel 1

Tabel 1. Perbandingan *Keystream Generator* dengan Teknik *Chaos*

Ukuran	***Keystream Generator***	**Teknik *Chaos***
Nilai Awal	Sama, dapat berbeda dengan penambahan umpan	Berbeda-beda
Ruang Fase	Integer	Real
Metode	Aljabar	Analitik
Putaran	Sama untuk setiap proses *generate key*	Berbeda-beda bergantung nilai *chaos* dari *key* sebelumnya
Parame-ter	Boolean, nilai diskrit	Real, nilai kontinu
Seba-ran	Difusi periodik	Sensitif tergantung nilai awal
Repre-sentasi digital	Aritmatik integer	Aritmatika non integer yang merepresen-tasikan nilai kontinu
Keaman-an	- Bergantung tingkat keacakan bilangan. Memungkin-kan terjadi *key* yang berisi nilai 0 seluruhnya sehingga cipherteks = plainteks. - *Key-stream* dapat berulang secara periodik - Hubungan statistik antar key mungkin diketahui	- Bergantung perbedaan nilai awal - *Keystream* tidak berulang secara periodik - Hubungan statistik antar key tidak diketahui karena perbedaan jumlah iterasi - Terdapat sejumlah tak hingga bilangan antara 0 dan 1, sehingga *exhaustive key search* tidak mungkin dilakukan

Pada *keystream generator* meskipun inputnya ditambahkan dengan umpan, jika perbedaan antara umpan yang satu dengan yang lain sangat kecil. Maka *keystream* yang akan dihasilkan pun besar nilainya akan relatif dekat dengan *keystream* sebelumnya.

Untuk itu, pada *keystream generator* umpan harus benar-benar acak, karena akan dioperasikan dengan nilai U (kunci masukan pengguna) yang sama dan dilakukan dengan jumlah putaran yang sama. Misal :
Operasi :

$U \oplus Z$

U : 1100101
Z_1 : 1000110 = 70
Key_1 : 0100011 = 35

U : 1100101
Z_2 : 1000111 = 71
Key_2 : 0100010 = 34

Untuk satu kali iterasi perbedaan nilai integer dari kunci yang dihasilkan 1 point.

Pada teknik chaos input pada setiap proses *generate* berbeda-beda. Input merupakan hasil dari fungsi sebelumnya, sehingga ketika ada perbedaan input awal yang diberikan maka hasilnya akan berbeda (adanya keterkaitan nilai x secara kontinu).

Perbedaan ini akan sangat signifikan karena jumlah iterasi untuk tiap *keystream* berbeda-beda tergantung pada nilai x sebelumnya. Cara ini juga berfungsi untuk menyembunyikan hubungan statistik antar *key*.
Misalnya :
Nilai masukan $x_1 = 32$
Nilai masukan $x_2 = 32{,}000001$
Nilai r = 4
dengan menggunakan persamaan (4)
$f(x_1) = | 4 \times 32 \times (1 - 32) | = 3960$
$f(x_2) = |4 \times 32{,}000001 \times (1 - 32{,}000001)$
$=|3968{,}000252000004$

Untuk satu kali iterasi perbedaan nilai integer setelah pembulatan ke bawah sebesar 8 poin.

Dapat diperkirakan bahwa untuk iterasi yang lebih banyak maka perbedaan nilai yang dihasilkan oleh teknik chaos akan sangat signifikan untuk input yang perbedaanya sangat kecil sekalipun.

6. KESIMPULAN

Teknik *chaos* dapat digunakan untuk membangkitkan bilangan random. Keacakan dari bilangan yang dihasilkan dengan menggunakan teknik *chaos* mempunyai tingkat keacakan yang lebih tinggi dari keacakan bilangan random hasil dari *keystream generator*. Hal ini dapat dilihat dari tingkat keamanan dan hasil bilangan acak yang dihasilkan oleh *chaos* tidak bersifat periodik, artinya bilangan yang dihasilkan benar-benar acak seperti pada *one time pad,* sehingga algoritma kriptografinya menjadi *unbreakable cipher.* Alasan lain ialah perbedaan input yang sangat kecil menghasilkan barisan kunci yang berbeda secara signifikan sehingga semakin sulit untuk dipecahkan.

DAFTAR REFERENSI

[1] Ninan Sajeeth Philip dan K. Babu Joseph, *Chaos for Stream Cipher,* Department of Physic, Cochin University of Science and Technology, 2001.

[2] James Lampton, *Chaos Cryptography: Protecting Data Using Chaos,* Mississippi School for Mathematics and Science.

[3] Stewart, John. *Cryptography with Chaotic Function.* Journal of Non-Linier Function. 2006.

[4] Munir, Rinaldi. *Diktat Kuliah IF5054 Kriptografi.*Departemen Informatika ITB. 2005