Metoda Enkripsi *Blowfish*

Dendy Narendra
Program Studi Teknik Informatika STEI ITB, Bandung, NIM: 13504033

Abstraksi – *Makalah ini menjelaskan Metoda Enkripsi Blowfish.*

Kata Kunci*: Blowfish*

1. PENDAHULUAN

Dalam kriptografi, *Blowfish* adalah *cipher block* berkunci, yang didesain oleh *Bruce Schneier* pada tahun 1993 yang mencakup jumlah besar *cipher* dan enkripsi. *Blowfish* memberikan hasil enkripsi yang baik (sulit untuk dipecahkan) dan sampai saat ini belum ada kriptanalisis yang mengklaim telah dapat memecahkannya. Hal tersebut mungkin juga dikarenakan sistem enkfirpsi lain yang lebih banyak mendapat perhatian seperti *Advanced Encryption Standard* (*AES*).

Schneier mendesain *Blowfish* sebagai algoritma untuk tujuan umum, mulanya ditujukan untuk menggantikan DES yang semakin tua dan bebas dari bermacam masalah layaknya algoritma lainnya. Saat itu, meski terdapat banyak algoritma, kebanyakan telah dipatenkan, ataupun dirahasiakan. Akan tetapi Scheneier telah menyatakan bahwa *Blowfish* tidak dipatenkan di negara manapun. Karena itu, algoritma *Blowfish* sampai saat ini bisa digunakan secara bebas oleh siapapun.

2. DAFTAR ISTILAH

- *Jaringan Feistel:*
  Proses yang sering digunakan dalam hampir semua algoritma *cipher* block yang cara kerjanya sebagai berikut:
  1. Bagi blok yang panjangnya n bit menjadi dua bagian, kiri (L) dan kanan (R), yang masing-masing panjangnya n/2
  2. Definisikan *cipher* blok berulang dimana hasil putaran ke-i ditentukan dari hasil putaran sebelumnya.

- *S-Box:*
  Matriks yang berisi substitusi sederhana yang memetakan satu atau lebih bit dengan satu atau lebih bit yang lain

- *Cipher block:*
  Algoritma kriptografi beroperasi pada plainteks/*cipher*teks dalam bentuk blok bit, yang dalam hal ini rangkaian bit dibagi menjadi blok-blok bit yang panjangnya sudah ditentukan sebelumnya.

- *DES:*
  Algoritma *cipher* blok yang populer karena telah dijadikan standar algoritma enkripsi kunci-simetri, meskipun kini standar tersebut telah diganti dengan algoritma yang baru.

3. DASAR TEORI

Blowfish merupakan metoda enkripsi yang mirip dengan *DES* (*DES-like cipher*) dan diciptakan oleh *Bruce Schneier* yang ditujukan untuk mikroposesor besar (32 bit ke atas dengan *cache* data yang besar). *Blowfish* dikembangkan untuk memenuhi kriteria disain sebagai berikut:

- Cepat, pada implementasi yang optimal *Blowfish* dapat mencapai kecepatan 26 *clock cycle* per *byte*.

- Kompak, *Blowfish* dapat berjalan pada memori kurang dari 5 KB.

- Sederhana, *Blowfish* hanya menggunakan operasi yang simpel: penambahan (*addition*), *XOR*, dan penelusuran tabel (*table lookup*) pada *operand* 32 bit. Desainnya mudah untuk dianalisa yang membuatnya resisten terhadap kesalahan implementasi.

- Keamanan yang variabel, panjang kunci *Blowfish* dapat bervariasi dan dapat mencapai 448 bit (56 *byte*).

Blowfish dioptimasikan untuk aplikasi dimana kunci tidak sering berubah, seperti jalur komunikasi atau enkripsi file otomatis. *Blowfish* jauh lebih cepat dari *DES* bila diimplementasikan pada 32 bit mikroprosesor dengan *cache* data yang besar, seperti *Pentium* dan *Power PC*, *Blowfish* tidak cocok untuk aplikasi seperti packet switching, dengan perubahan kunci yang sering, atau sebagai fungsi hash satu arah. Kebutuhan memorinya yang besar tidak memungkinkan untuk aplikasi kartu pintar (*smart card*).

4. PEMBAHASAN

Deskripsi dari *Blowfish*

Blowfish merupakan blok *cipher* 64-bit dengan panjang kunci variabel. Algoritma ini terdiri dari dua

bagian: key expansion dan enkripsi data. Key expansion merubah kunci yang dapat mencapai 448 bit menjadi beberapa array subkunci (*subkey*) dengan total 4168 byte.

Enkripsi data terdiri dari iterasi fungsi sederhana sebanyak 16 kali. Setiap putaran terdiri dari permutasi kunci-dependent dan substitusi kunci- dan data-dependent. Semua operasi adalah penambahan dan XOR pada variable 32-bit. Tambahan operasi lainnya hanyalah empat penelusuran tabel (table lookup) array berindeks untuk setiap putaran.

Blowfish menggunakan subkunci yang besar. Kunci ini harus dihitung sebelum enkripsi atau dekripsi data.

Array P terdiri dari delapan belas 32-bit subkunci:

P1,P2, . . . ,P18

Empat 32-bit S-box masing-masing mempunyai 256 entri:

S1,0, S1,1, . . . , S1,255

S2,0, S2,1, . . . , S2,255

S3,0, S3,1, . . . , S3,255

S4,0, S4,1, . . . , S4,255

Metoda selengkapnya untuk menghitung subkunci ini akan dijelaskan pada bagian bawah.

Blowfish merupakan algoritma yang menerapkan jaringan Feistel (Feistel network) yang terdiri dari 16 putaran. Input merupakan elemen 64 bit, X. Untuk mengenkrip:

Bagi X menjadi dua 32-bit: XL, XR

untuk i = 1 sampai 16

XL = XL xor Pi

XR = F(XL) xor XR

Tukar XL dan XR

Tukar XL dan XR (batalkan penukaran terakhir)

XR = XR xor P17

XL = XL xor P18

Kombinasikan kembali XL dan XR

Fungsi F adalah sebagai berikut:

Bagi XL, menjadi empat bagian 8-bit: a, b, c dan d

F(XL) = ((S1,a + S2,b mod 232) xor S3,c) + S4,c mod 232

Dekripsi sama persis dengan enkripsi, kecuali P1, P2, ..., P18 digunakan pada urutan yang terbalik.

Subkunci dihitung menggunakan algoritma *Blowfish*, metodanya adalah sebagai berikut:

1. Pertama-tama inisialisasi P-array dan kemudian empat S-box secara berurutan dengan string yang tetap. String ini terdiri digit hexadesimal dari pi.

2. XOR P1 dengan 32 bit pertama kunci, XOR P2 dengan 32 bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P18). Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.

3. Enkrip semua string nol dengan algoritma *Blowfish* dengan menggunakan subkunci seperti dijelaskan pada langkah (1) dan (2).

4. Ganti P1 dan P2 dengan keluaran dari langkah (3)

5. Enkrip keluaran dari langkah (3) dengan algoritma *Blowfish* dengan subkunci yang sudah dimodifikasi.

6. Ganti P3 dan P4 dengan keluaran dari langkah (5).

7. Lanjutkan proses tersebut, ganti seluruh elemen dari P-array, dan kemudian seluruh keempat S-box berurutan, dengan keluaran yang berubah secara kontinyu dari algoritma *Blowfish*.

Total diperlukan 521 iterasi untuk menghasilkan semua subkunci yang dibutuhkan. Aplikasi kemudian dapat menyimpan subkunci ini dan tidak dibutuhkan langkah-langkah proses penurunan ini berulang kali, kecuali kunci yang digunakan berubah.

Keamanan dari *Blowfish*

Tidak ada kelemahan yang berarti dari algoritma *Blowfish* yang dapat ditemukan sampai saat ini, kecuali adanya weak key, dimana dua entri dari S-box mempunyai nilai yang sama. Tidak ada cara untuk mencek weak key sebelum melakukan key expansion. Bila dikuatirkan hal ini dapat mengurangi keamanannya maka dapat dibuat rutin untuk mengecek entri S-box, walaupun hal ini tidak perlu.

Sampai saat ini tidak ada cryptanalysis yang berhasil

tehadap *Blowfish*, untuk amannya jangan menggunakan *Blowfish* dengan kurang dari 16 putaran (round).

5. KESIMPULAN

Blowfish merupakan salah satu *cipher block* tercepat yang digunakan secara luas, kecuali ketika pergantian kunci. Setiap kunci baru membutuhkan praproses yang setara dengan mengenkripsi sekitar 4 *kilobyte* teks, yang bila dibandingkan dengan *cipher block* yang lain, sangat lambat. Hal ini menyebabkan penggunaannya terbatas untuk beberapa aplikasi.

Dalam suatu aplikasi, sesungguhnya hal ini merupakan suatu kelebihan: *hashing password* yang digunakan pada *Open BSD* menggunakan algoritma yang diturunkan dari *Blowfish* yang justru memanfaatkan kelemahan tadi. Idenya adalah usaha komputasi ekstra yang dibutuhkan memberikan perlindungan terhadap serangan *'dictionary'*.

Dalam beberapa implementasi, *Blowfish* membutuhkan ukuran memori yang cukup besar, sekitar 4 *kilobytes* dari RAM. Hal ini tidak menjadi masalah bagi PC ataupun laptop, akan tetapi hal ini membatasi penggunaan algoritma ini untuk sistem yang lebih kecil dan rumit seperti halnya *SmartCard*.

Blowfish tidak dilindungi paten apapun dan karena itu tersedia dengan bebas bagi siapapun untuk menggunakannya. Hal ini membuat algoritma ini cukup populer penggunaannya dalam dunia kriptografi.

6. DAFTAR REFERENSI

[1] Budi Setiawan, Metode Enkripsi *Blowfish*, http://www.bimacipta.com/*Blowfish*.htm

[2] Menezes, P. van Oorschot and S. Vanstone, Handbook of Applied Cryptography, CRC Press, 1996.

[3] Schneier, Applied Cryptography - Protocol, Algorithm, and Source Code in C, second edition, John Willey & Sons, 1996.

[4] Wikipedia, *Blowfish* (*cipher*), http://en.wikipedia.org/wiki/*Blowfish*_%28*cipher*%29.