Proposal Makalah IF3058 Kriptografi

Studi dan Perbandingan Algoritma RC6 dan Blowfish

Reza Brianca Widodo / 13507013[1]
Program Studi Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi Bandung, Jl. Ganesha 10 Bandung 40132, Indonesia
[1]if17013@students.if.itb.ac.id

***Abstrak*—Informasi saat ini telah menjadi kebutuhan utama yang menentukan daya saing sebuah pihak. Semakin berharganya informasi membuat semakin ketatnya penjagaan terhadap informasi tersebut. Di sisi lain, semakin banyak pihak yang menginginkan informasi rahasia tersebut untuk dimiliki. Kriptografi merupakan ilmu untuk menyembunyikan pesan. Makalah ini membahas dua buah kriptografi modern yang termasuk dalam kategori blok cipher yaitu Blowfish dan RC6. Pembahasan mencakup mekanisme setiap algoritma dan perbandingan performansi dari kedua jenis algoritma terhadap file berupa dokumen teks. Hasilnya algoritma Blowfish dapat melakukan proses enkripsi dan dekripsi yang lebih cepat daripada RC6. Hal tersebut juga berlaku pada troughput yang dihasilkan oleh Blowfish. Dari sisi keamanan, sampai saat ini belum ada kriptanalis yang dilaporkan mampu menembus algoritma Blowfish, sedangkan untuk algoritma RC6 masih diperlukan waktu yang sangat lama untuk dapat menembusnya sehingga kedua jenis algoritma tersebut masih termasuk aman hingga saat ini.**

***Kata kunci*— Blowfish, informasi, keamanan, kriptografi, performansi, RC6**

I. PENDAHULUAN

Pada jaman sekarang, kebutuhan akan informasi menjadi suatu hal yang mutlak diperlukan. Informasi saat ini telah menjadi sumber daya yang sangat esensial bagi banyak elemen. Pihak yang memiliki informasi lebih banyak akan memiliki keuntungan kompetitif dari pesaingnya. Begitu juga sebaliknya, pihak yang tidak mendapatkan informasi tertentu akan sulit bersaing dengan pihak pesaingnya.

Pentingnya informasi membuat banyak pihak yang menjaga informasi tersebut dengan ketat. Di sisi lain, informasi rahasia akan sangat menguntungkan bagi pihak pesaing jika dapat diketahui. Hal inilah yang menyebabkan lahirnya mekanisme pengamanan informasi. Salah satu mekanismenya adalah dengan kriptografi.

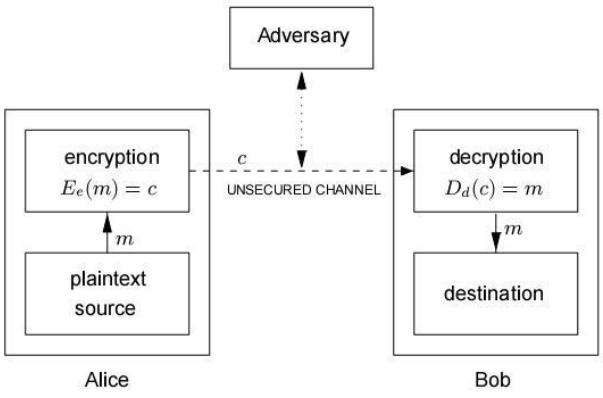
II. KRIPTOGRAFI

Kriptografi adalah ilmu untuk menyembunyikan pesan. Selain itu juga dapat berarti ilmu yang mempelajari teknik matematis terkait aspek-aspek keamanan informasi[3]. Aspek yang dimaksud adalah sebagai berikut :

1. *Confidentiality*
   Aspek ini bertujuan untuk menjaga isi dari informasi dari pihak yang tidak diijinkan untuk memilikinya.
2. *Data Integrity*
   Aspek ini bertujuan untuk mendeteksi perubahan data. Perubahan data ini antara lain penambahan, pengurangan ataupun penggantian isi data.
3. *Authentication*
   Aspek ini bertujuan untuk melakukan identifikasi. Aspek ini berlaku untuk pihak yang melakukan akses informasi dan juga terhadap informasi itu sendiri.
4. *Non-repudiation*
   Aspek ini bertujuan agar pihak yang melakukan akses informasi tidak dapat menyangkal bahwa dia telah melakukan akses[3].

Pada awal kemunculannya, kriptografi klasik menerapkan prinsip substitusi dan transposisi dalam proses penyembunyian pesan. Pada masa tersebut, operasi yang dilakukan masih belum berbasis komputer sehingga kriptografi pada masa itu masih berbasis alfabet saja. Seiring dengan kemunculan komputer, maka muncullah kriptografi modern. Kriptografi pada masa ini memiliki prinsip dasar yang mirip dengan kriptografi klasik, hanya saja operasi yang dilakukan tidak lagi berbasis alfabet namun telah berbasis pada bilangan biner.

Secara umum proses kriptografi dapat digambarkan sebagai berikut.



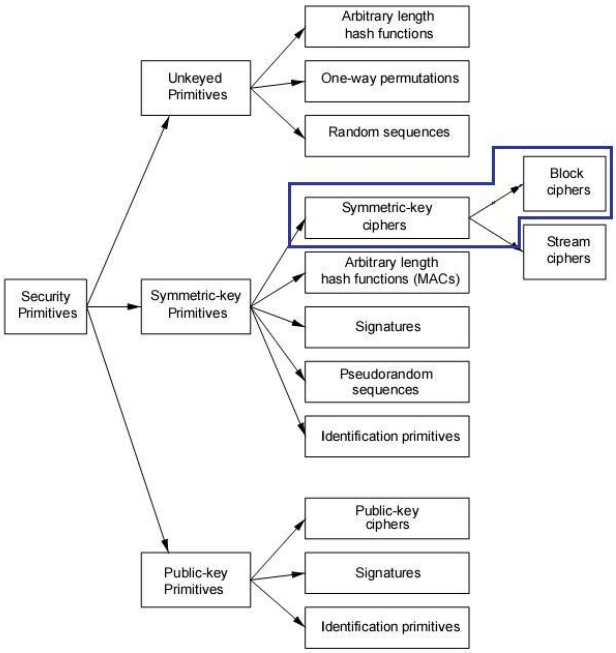
Gambar 1. Alur kriptografi[3]

Pada gambar 1 terdapat diagram yang menjelaskan secara sederhana tentang penggunaan kriptografi. Pihak yang terlibat antara lain adalah Alice sebagai pengirim pesan, Bob sebagai penerima pesan, pesan asli atau plainteks, pesan terenkripsi atau cipherteks, kunci, proses enkripsi dan dekripsi.

Proses yang terjadi secara singkat adalah sebagai berikut :

1. Alice ingin mengirim pesan kepada Bob
2. Pesan asli (plainteks) mengalami proses enkripsi menjadi pesan yang akan dikirim (cipherteks)
3. Cipherteks dikirim kepada Bob
4. Cipherteks mengalami proses dekripsi untuk menghasilkan plainteks semula.

Dalam perkembangannya, ilmu kriptografi memiliki perkembangan sesuai dengan kebutuhan dan kompleksitasnya. Gambar 2 menunjukkan klasifikasi kriptografi modern.



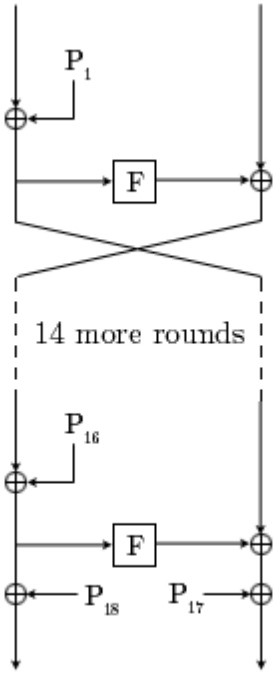
Gambar 2. Klasifikasi kriptografi modern[3]

Dari gambar 2 terlihat bahwa kriptografi modern terbagi menjadi 3 cabang utama. Pada makalah ini akan dibahas secara spesifik mengenai algoritma kunci simetri. Algoritma kunci simetri adalah algoritma yang menggunakan kunci yang sama dalam proses enkripsi dan dekripsinya[2]. Algoritma jenis ini dapat dibagi menjadi 2 jenis yaitu cipher blok dan cipher aliran. Cipher blok melakukan enkripsi setiap blok yang terdiri dari n-bit plainteks, sedangkan cipher aliran melakukan enkripsi setiap bit dari plainteks tersebut.

Algoritma kunci simetri memiliki kelebihan dalam kecepatan proses enkripsi maupun dekripsi. Hal ini disebabkan oleh penggunaan kunci yang sama baik pada waktu enkripsi maupun dekripsi. Selain itu, fungsi enkripsi dan dekripsi yang digunakan juga relatif sama hanya saja proses penjadwalan kuncinya menjadi terbalik urutannya.

Pada makalah ini akan dibahas dua buah jenis algoritma yang termasuk kategori kunci simetri yaitu RC6 dan Blowfish. Pembahasan mencakup cara kerja dari setiap algoritma tersebut, perbandingan cara kerja keduanya, dan performansi dari kedua algoritma tersebut.

III. BLOWFISH



Gambar 3. Diagram Struktur Blowfish[4]

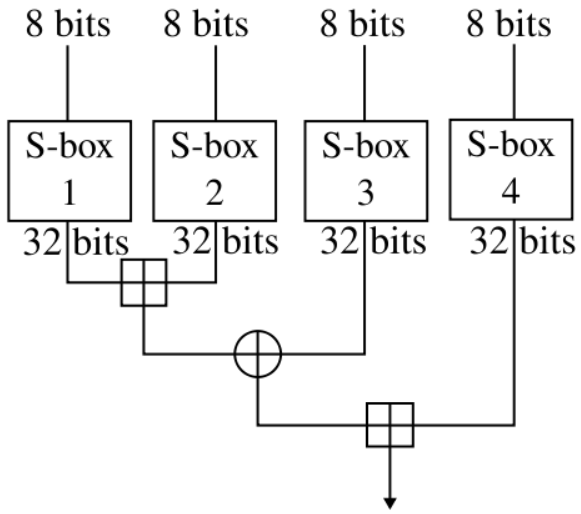
Blowfish merupakan blok cipher 64-bit dengan panjang kunci variabel. Algoritma ini terdiri dari dua bagian: *key expansion* dan enkripsi data. *Key expansion* merubah kunci yang panjangnya antara 1 sampai 448 bit menjadi beberapa array subkunci (*subkey*) dengan total 4168 byte. Semakin panjang kunci maka semakin tinggi pula tignkat keamanannya.

Enkripsi data terdiri dari iterasi fungsi enkripsi 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[4].

Blowfish merupakan algoritma yang menerapkan jaringan Feistel (*Feistel network*) yang terdiri dari 16 putaran. Masukan dari jaringan ini berupa 64 bit plainteks. Secara sederhana gambar 3 dapat dijelaskan sebagai berikut[4]:

```
i = 1
loop from j to 16
Rj = Lj-1 XoR Pj
Lj = F(Rj) XoR Rj-1
end loop
L17 = R16 XoR P18
R17 = L16 XoR P17
```

Blok cipherteks adalah hasil kombinasi kembali L17 dan R17.



Gambar 5. Proses enkripsi pada Blowfish[4]

Fungsi enkripsi F pada algoritma Blowfish adalah sebagai berikut [4]:

1. Bagi X_L, menjadi empat bagian 8-bit: *a*, *b*, *c* dan *d*
2. $F(X_L) = ((S_{1,a} + S_{2,b} \bmod 2^{32}) \text{ xor } S_{3,c}) + S_{4,c} \bmod 2^{32}$

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

Array P terdiri dari delapan belas 32-bit subkunci:
$P_1, P_2, \ldots, P_{18}$

Empat 32-bit S-box masing-masing mempunyai 256 entri:
$S_{1,0}, S_{1,1}, \ldots, S_{1,255}$
$S_{2,0}, S_{2,1}, \ldots, S_{2,255}$
$S_{3,0}, S_{3,1}, \ldots, S_{3,255}$
$S_{4,0}, S_{4,1}, \ldots, S_{4,255}$

Subkunci dihitung menggunakan algoritma Blowfish, metodanya adalah sebagai berikut[4]:

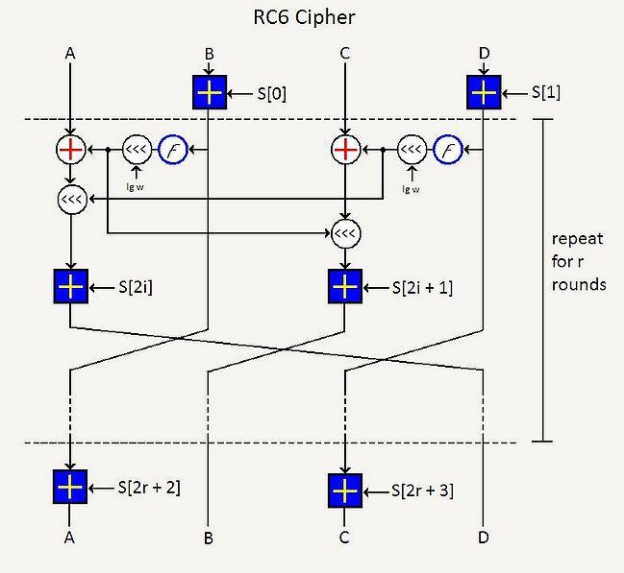
1. Pertama-tama inisialisasi P-array dan kemudian empat S-box secara berurutan dengan string yang tetap. String ini terdiri dari digit hexadesimal dari pi.
2. XOR P_1 dengan 32 bit pertama kunci, XOR P_2 dengan 32 bit kedua dari kunci dan seterusnya untuk setiap bit dari kunci (sampai P_{18}). Ulangi terhadap bit kunci sampai seluruh P-array di XOR dengan bit kunci.
3. Enkrip 0 sebanyak 64 bit 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 P_3 dan P_4 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.

Dekripsi untuk Blowfish bersifat maju kedepan sehingga dapat tetap menggunakan fungsi enkripsi yang sama. Perbedaan dengan proses enkripsi adalah masukannya berupa cipherteks dan pemanggilan jadwal kunci yang terbalik dari proses enkripsi[4].

```
j = 1
loop from j to 16
Rj = Lj-1 XoR P19-j
Lj = F(Rj)XoR Rj-1
end loop
L17 = R16 XoR P1
R17 = L16 XoR P2
```

IV. RC6



Gambar 6. Diagram struktur pada RC6[5]

Algoritma RC6 merupakan pengembangan dari algoritma RC5. Algoritma ini juga termasuk salah satu finalis dalam kompetisi AES untuk menggantikan DES yang sudah dianggap tidak aman. RC6 dapat dilihat sebagai dua buah algoritma RC5 yang berjalan secara paralel seperti pada gambar 6. Jumlah putaran yang digunakan adalah 20 sebagai salah satu syarat kompetisi AES[5].

RC6 merupakan cipher blok 128 bit dan menggunakan panjang kunci 128, 192 atau 256 bit. Walaupun terdapat 3 jenis panjang kunci, pada prakteknya panjang kunci yang digunakan adalah 128 atau 256 bit. Plainteks disimpan dalam empat w-bit register masukan A, B,C, dan D. kemudian diperlukan w-bit kunci putaran yang disimpan dalam array S[0,1, ..., 2r+3]

Proses enkripsi yang dilakukan adalah sebagai berikut[5] :

```
B = B + S[0]
D = D + S[1]
for i = 1 to r do
{
t = (B*(2B + 1)) <<< lg w
u = (D*(2D + 1)) <<< lg w
```

```
A = ((A ⊕ t) <<< u) + S[2i]
C = ((C ⊕ u) <<< t) + S[2i + 1]
(A, B, C, D)  =  (B, C, D, A)

}
A = A + S[2r + 2]
C = C + S[2r + 3]
```

Perlu diketahui bahwa operasi penambahan dilakukan dengan modulus 2^w dan operasi lg adalah nilai logaritma basis 2. Untuk proses dekripsi dilakukan proses sebagai berikut [5]:

```
C = C - S[2r + 3]
A = A - S[2r + 2]
for i = r downto 1 do
{
(A, B, C, D) = (D, A, B, C)
u = (D*(2D + 1)) <<< lg w
t = (B*(2B + 1)) <<< lg w
C = ((C - S[2i + 1]) >>> t) ⊕ u
A = ((A - S[2i]) >>> u) ⊕ t
}
D = D - S[1]
B = B - S[0]
```

Pengaturan kunci dilakukan berdasarkan masukan kunci dari pengguna. proses ini sama dengan apa yang diterapkan pada algoritma RC5[6]. Masukan tersebut akan digunakan untuk mengisi array kunci S yang beranggotakan 2(r+1) elemen. Proses pengaturan kunci terdiri atas tiga tahap. Sebelum masuk ke dalam tahapan perlu diketahui definisi dari *Magic Constant*.

Algoritma ekspansi kunci menggunakan dua buah konstanta berukuran *word* yaitu P_w dan Q_w. Konstanta tersebut berisi nilai berikut [6]:

$$P_w = \text{Odd}((e - 2)2^w)$$
$$Q_w = \text{Odd}((\Phi - 1)2^w)$$

dengan e = 2,718281828459... (basis logaritma natural) dan Φ = 1,618033988749... (rasio emas)

Fungsi Odd(x) adalah mencari nilai ganjil terdekat dengan x dengan pembulatan ke atas jika nilainya genap.

Untuk nilai w = 16, 32, dan 64 nilai P_w dan Q_w adalah sebagai berikut[6] :

P16 = b7e1
Q16 = 9e37

P32 = b7e15163
Q32 = 9e3779b9

P64 = b7e151628aed2a6b
Q64 = 9e3779b97f4a7c15

Tahap pertama dari ekspansi kunci adalah melakukan konversi dari byte (array K) menjadi word (array L). prosesnya adalah sebagai berikut[6] :

```
for i = b - 1 downto 0 do
L [i/u] = (L[i/u] <<< 8 ) + K[i]
```

Tahap kedua dari ekspansi kunci adalah mengisi nilai dari array S menggunakan progresi aritmatik modulo 2^w yang ditentukan dari nilai P_w dan Q_w. Karena Q_w bernilai ganjil, periodenya akan menjadi 2^w.

```
S[0] = Pw
for i = 1 to t - 1 do
S[i] = S[i - 1] + Qw
```

Tahap terakhir dari ekspansi kunci adalah mencampurkan kunci masukan dari pengguna dengan array S dan L. karena adanya kemungkinan perbedaan ukuran dari S dan L, array yang lebih besar akan diproses sebanyak tiga kali, dan yang lainnya juga dapat menjadi diproses lebih banyak.

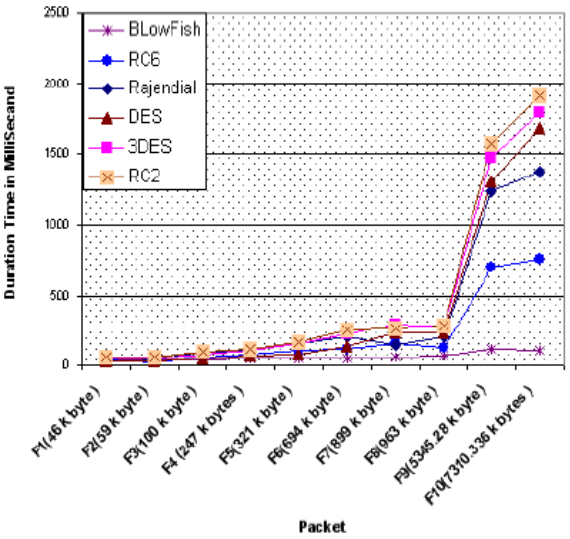
```
i = j = 0
A = B = 0
do 3 * max (t,c) times :
A = S[i] = (S[i] + A + B) <<< 3
B = L[j] = (L[j] + A + B) <<< (A + B)
i = (i + 1) mod (t)
j = (j + 1) mod (c)
```

V. PERBANDINGAN BLOWFISH DAN RC6

Untuk algoritma Blowfish, sampai saat ini belum ditemukan titik kelemahan yang berarti kecuali adanya *weak key*. *Weak key* adalah nilai dari sebuah kunci dapat menghasilkan dua buah entri pada S-Box bernilai sama. tidak ada cara untuk mengecek weak key sebelum melakukan key expansion. Untuk meningkatkan keamanan sebaiknya tidak menggunakan algoritma Blowfish dengan jumlah putaran kurang dari 16 putaran. Hal ini disebabkan telah ditemukan teknik *second-order differential attack* yang mampu memecahkan Blowfish dengan 4 putaran namun tidak untuk putaran yang lebih banyak lagi[4].

Pada algoritma RC6, cara yang harus dicoba oleh kriptanalis untuk membongkar dengan menggunakan cara exhaustive search adalah $\{2^{8b}, 2^{1408}\}$. Terdapat alternatif penyerangan lain namun masih memerlukan jumlah data yang banyak dan mendapatkan 2^a blok dari pasangan plainteks-cipherteks. walaupun dengan kecepatan proses 1 Terabit per detik atau setara dengan 10^{12} bit tiap detiknya, waktu yang diperlukan oleh 50 komputer dengan kecepatan tersebut untuk mendapatkan 2^{64} blok data tetap lebih dari setahun, untuk 2^{80} blok data diperlukan lebih dari 98000 tahun dan untuk 2^{128} blok data diperlukan lebih dari 10^{19} tahun[5].

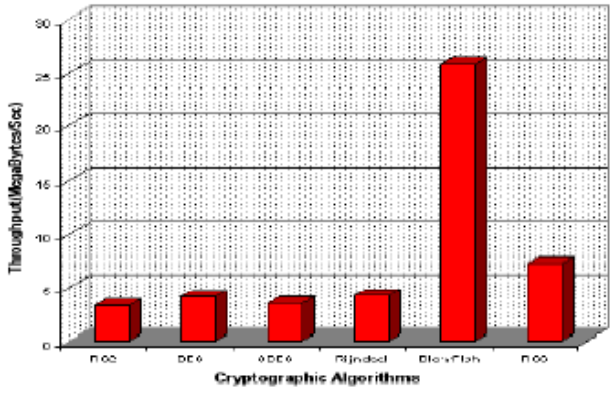
Berdasarkan penelitian yang dilakukan oleh Elminaam dkk, didapatkan perbandingan penggunaan waktu enkripsi yang menunjukkan performa dari beberapa jenis algoritma simetri. File yang digunakan untuk enkripsi adalah data teks dengan besar file antara 46 KB hingga 7,3 MB. Hasil dari percobaan tersebut terdapat pada gambar 7.



Gambar 7. Grafik perbandingan waktu yang diperlukan untuk melakukan enkripsi[1]

Pada gambar 7 terlihat bahwa algoritma Blowfish berada pada peringkat pertama kemudian diikuti oleh algoritma RC6. Hal ini menunjukkan Blowfish mampu melakukan enkripsi yang lebih cepat (kurang dari 500 ms untuk ukuran file 7,3 MB) daripada RC6 (antara 500 ms sampai 1000 ms untuk ukuran file 7,3 MB).

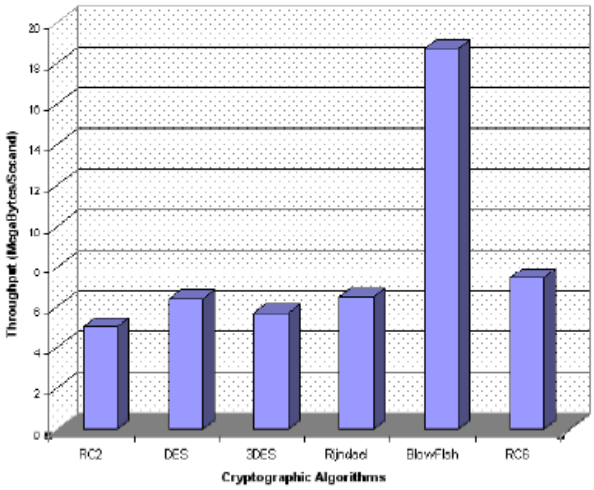
Hasil penghitungan *troughput* enkripsi untuk setiap algoritma tersebut terdapat pada gambar 8.



Gambar 8. Grafik perbandingan *troughput* enkripsi[1]

Pada gambar 8 terlihat bahwa troughput Blowfish juga lebih tinggi (lebih besar dari 25 MB per detik) daripada RC6 (antara 5 sampai 10 MB per detik).

Hasil penghitungan *troughput* dekripsi untuk setiap algoritma tersebut terdapat pada gambar 9.



Gambar 9. Grafik perbandingan *troughput* dekripsi[1]

Pada gambar 9 terlihat lagi dominasi Blowfish daripada algoritma RC6. blowfish mampu menghasilkan troughput pada saat dekripsi antara 19 sampai 20 MB per detik. RC6 di sisi lain mampu menghasilkan troughput antara 6 sampai 8 MB per detik.

V. KESIMPULAN

Kedua jenis algoritma blok cipher memiliki tingkat keamanan yang tinggi jika ketentuan yang berlaku dalam mekanisme enkripsi di dalamnya terpenuhi, seperti jumlah putaran minimal yang harus dilakukan. Hal ini dapat dilihat dari belum adanya laporan yang menunjukkan bahwa kedua jenis algoritma ini mudah ditembus.

Blowfish memiliki keunggulan dalam hal efisiensi waktu enkripsi dan dekripsi. Hal ini diakibatkan oleh penggunaan jaringan feistel yang membagi blok masukan menjadi 2, sehingga proses di dalam fungsi enkripsi dapat berjalan dengan lebih cepat. Algoritma RC6 juga memiliki keamanan yang baik, namun proses di dalam jaringan feistelnya cukup rumit karena membagi blok masukan menjadi 4. Akibatnya, waktu yang diperlukan untuk melakukan proses enkripsi maupun dekripsi menjadi lebih lama karena proses didalamnya dapat dianalogikan sebagai 2 buah feistel yang berjalan secara paralel.

Secara umum, baik Blowfish maupun RC6 dapat digunakan dalam mengamankan informasi. Pemilihan algoritma mana yang digunakan dapat disesuaikan dengan kebutuhan pengamanan informasi itu sendiri. Blowfish menjanjikan tingkat efisiensi yang lebih baik dari RC6, namun dengan tingkat keamanan yang relatif sama.

REFERENCES

[1] Elminaam, D.S, Kader, H.M, Hadhoud, M.M. 2009. Evaluating the Performance of Symmetric Encryption Algorithms. HTI 10th Ramadan City and Faculty of

Computers and Information Minufiya University, Egypt.
[2] http://www.encryptionanddecryption.com/algorithms/symmetric_algorithms.html
[3] Menezes, A.J, Oorschot, P.C, Vanstone, S.A. 1996. Handbook of Applied Cryptography. CRC Press.
[4] Putra, M.R. 2008. Studi dan Implementasi the Blowfish Encryption Algorithm dalam Bahasa Pemrograman C++. Institut Teknologi Bandung.
[5] Rivest, R.L, Robshaw, M.J.B, Sidney. R, Lin, Y.L. 1998. The RC6 Block Cipher. MIT Laboratory for Computer Science and RSA Laboratories.
[6] Rivest, R.L. 1994. The RC5 Enryption Algorithm. MIT Laboratory for Computer Science.

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, 22 Maret 2011

Reza Brianca Widodo / 13507013