Analisa Kriptanalisis Deferensial Pada Twofish

Arif Suprabowo
13503122

Teknik Informatika
Sekolah Teknik Elektro dan Informatika
Institut Teknologi bandung
Email : *if13122@students.if.itb.ac.id*

Abstraksi

Kriptanalisis differensial merupakan teknik kriptanalisis yang banyak digunakan untuk melakukan pengujian terhadap kekuatan algoritma-algoritma penyandian modern yang memiliki struktur yang sangat komplek. Kriptanalisis deferensial pertama kali diperkenalkan oleh Biham – Shamir untuk melakukan kriptanalisis terhadap algoritma DES. Seiring berkembangnya waktu, metode kriptanalisis deferensial mengalami banyak kemajuan terutama dalam melakukan efisiensi operasi penambahan modulo 2^n.

Paper ini akan membahas mengenai kriptanalisis deferensial serta studi kasus terhadap pengaplikasiannya pada algoritma Twofish. Informasi pada paper diharapkan dapat memberikan gambaran mengenai salah satu pemanfaatan teknik kriptanalisis differensial.

Kata kunci : Kriptanalisis deferensial, DES, Twofish, modulo 2^n

1. Pendahuluan

Keamanan data merupakan isu yang paling berkembang pada abad 21. Komunikasi data yang mulai beralih dari system konvensional menuju system digital menyebabkan banyak kekhawatiran akan keamanan data yang ditransportasikan.

Pemanfaatan jaringan public dalam bentuk internet menyebabkan perlunya suatu perlakkuan khusus terhadap data yang dikirimkan. Jaringan public merupakan jaringan yang terbuka sehingga ada beberapa pihak yang memanfaatkan jaringan ini untuk mendapatkan informasi-informasi berharga dengan cara membaca paket-paket data yang melintasi jaringan ini.

Penyandian data (enkripsi-dekripsi) merupakan metode yang banyak digunakan untuk melakukan pengamanan data sebelum dikirimkan. Penyandian data dengan metode blok cipher adalah salah satu metode yang paling banyak digunakan. Metode ini menyebabkan data yang disandikan memiliki panjang yang hamper sama dengan data sebelum di sandikan.

Twofish merupakan salah satu algoritma penyandian yang memanfaatkan blok cipher dan merupakan salah satu algoritma kunci simetri modern. Twofish merupakan salah satu dari finalis kompetisi AES (Advance Encryption Standard). Pendekatan modern dimana sifat algoritma enkripsi yang tidak dirahasiakan lagi membuka kesempatan para kriptanalisis untuk melakukan pengujian kekuatan algoritma tersebut.

Salah satu metode kriptanalisis yakni Kriptanalisis deferensial pada tahun 2000 memberikan pengetahuan akan kemungkinan algoritma Twofish dapat dipecahkan. Paper yang di presentasikan oleh Shiho dan Lisa memberikan kesempatan pada kriptanalisis untuk menguji kekuatan Twofish.

Makalah ini akan memberikan pengetahuan mengenai kriptanalisis deferensial , algoritma twofish dan metode kriptanalisis deferensial yang

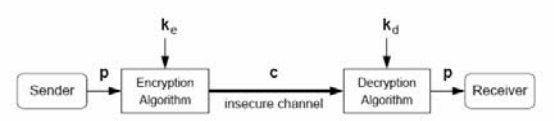
digunakan oleh shiho untuk menguji kekuatan Twofish.

2. *Ciphertext*

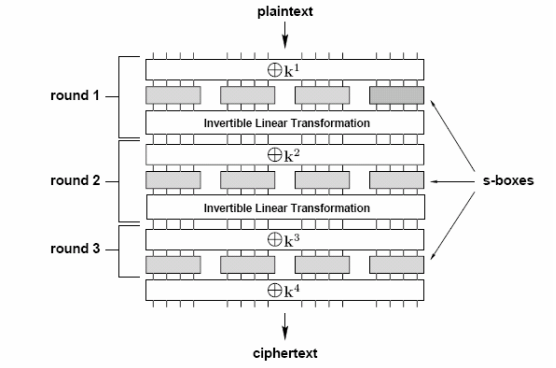
Chipertext merupakan rangkaian dari blok-blok chiper yang telah mengalami beberapa kali pengulangan dari operasi-operasi seperti substitusi,transposisi penambahan maupun perkalian modular, serta transformasi linier. Banyak metode yang dilakukan untuk melakukan beberapa kali putaran penyandian data, salah satu diantaranya menggunakan jaringan Feistel.

Feistel akan membagi blok menjadi 2 yaitu blok kanan dan blok kiri kemudian salah satu blok akan dilakukan penyandian dan kemudian di lakukan pertukaran posisi antara blok kanan dan blok kiri pada akhir masing-masing tahap putaran Feistel.

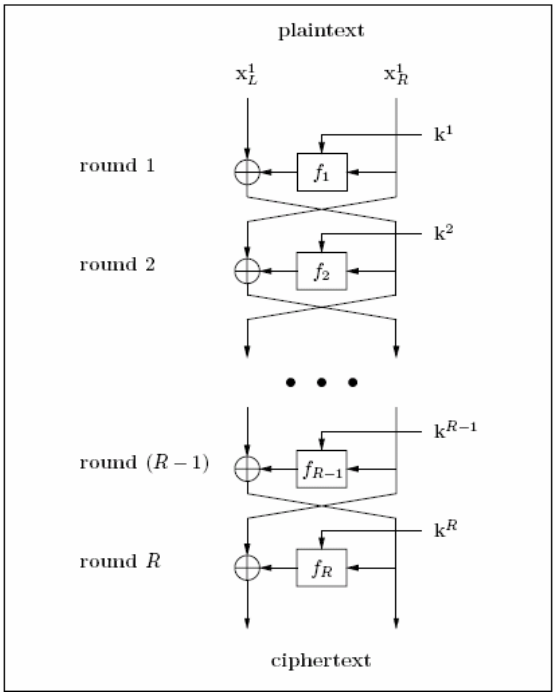
Berikut ini adalah beberapa skema yang banyak digunakan untuk sebagai dasar perancangan algoritma penyandian (enkripsi).



Skema enkripsi dan dekripsi



Skema enkripsi
Substitution-Permutation Network



Skema enkripsi Feistel Network

3. Keamanan *Chipertext[2]*

Sangat sulit menentukan secara matematis apakah suatu chipertext tersebut aman atau tidak. Namun terdapat suatu panduan bahwa apabila chipertext tersebut memiliki tingkat *random* data yang tinggi maka chipertext tersebut dapat dikategorikan memiliki tingkat keamanan yang tinggi.

Secara ilmiah, suatu chipertext akan diuji tingkat keamanannya berdasarkan tingkat algoritma yang digunakan untuk menyusunnya. Hal ini didasarkan bahwa banyak algoritma penyandian bersifat terbuka, sehingga berbagai pihak dapat memempeljari algoritma tersebut. Apabila hasil publikasi dari algoritma yang digunakan untuk melakukan penyandian data sampai dalam jangka waktu yang lama belum didapatkan publikasi mengenai metode untuk melakukan kriptanalisis, dapat disimpulkan sementara bahwa *chipertext* yang diperoleh dari algoritma tersebut memiliki tingkat keamanan yang tinggi.

4. Serangan terhadap blok cipher

Pada sat ini banya berbagai metode untuk melakukan serangan kepada blok cipher. Secara khusus serangan-serangan ini melibatkan penurunan kunci (*a total break*), walaupun hal ini

mungkin dilakukan dengan melakukan perancangan sebuah algoritma deskripsi ciphertext tanpa harus mengetahui kunci yang digunakan (*global deduction*)[7].

Serangan – serangan pada blok cipher dapat di kategorikan sebagai berikut, tergantung dengan informasi yang tersedia untuk melakukan serangan[8].

a) **Hanya ciphertext**

Penyerang memiliki satu atau lebih cipherteks.

b) **Known-plaintext**

Penyerang memiliki satu atau lebih plainteks dan cipherteks yang berkoresponden dengannya.

c) **Chosen-plaintext**

Penyerang dapat memilih sekumpulan plainteks (cipherteks) untuk dilakukan enkripsi (dekripsi) untuk mendapatkan koterkaitan chiperteks (plainteks).

d) **Adaptive chosen-plaintext (Adaptive chosen-ciphertext)**

Penyerang dapat mengirimkan plainteks (cipherteks) untuk enkripsi (dekripsi) untuk dibandingkan dengan pengiriman data yang sebelumnya.

Kompleksitas serangan adalah perlunya sejumlah data yang akan digunakan untuk melakukan serangan.

Kompleksitas waktu adalah kebutuhan maksimum dari kompleksitas data dan kebutuhan langkah-langkah untuk proses tambahan[9].

Sedangkan jenis-jenis serangan untuk mendapatkan informasi dari kunci yang digunakan :

1. **Exhaustive Key Search**

   Diberikan pasangan <plainteks, cipherteks > yang telah diketahui <p,c>. Exhaustive key search melibatkan enkripsi p dengan masing-masing 2^K kunci, yakni dengan membuang semua kunci yang tidak menghasilkan ciperteks yang sesuai, c.

2. L**inier Kriptanalisis**

   Metode ini diperkenalkan oleh Matsui pada tahun 1993[10], adalah serangan berjenis known-plaintext yang diyakini sebagai salah satu jenis serangan yang cukup kuat terhadap blok cipher.

   Metode ini pertama kali digunakan sebagai serangan kepada DES[11]— Matsui memerlukan 2^{43} pasangan <plainteks,cipherteks> dan kompleksitasnya mencapai 2^{30}.

   Linier Kriptanalisis memerlukan adanya sejumlah besar perkiraan nilai probabilistic linier untuk seluruh cipher dikurangi satu atau lebih *outer round*.

3. **Diferensial Kriptanalisis**

   Kriptanalisis deferensial merupakan chosen-plaintext attack yang di perkenalkan oleh Biham dan Shamir pada tahun 1990. Metode ini akan dijelaskan lebih lanjut pada bagian Kriptanalisis deferensial di makalah ini.

4. **Higher-Order Defferential Cryptanalysis**

   Higher-order differential cryptanalysis menggunakan konsep *derivative* dari pemetaan Boolean[12]. Serangan melibatkan pengaturan sekelompok persamaan seperti derivative dan *incorporating certain* bit-bit dari subkey, serta kemudian menggunakan brute force untuk menentukan nilai yang benar dari subkey. Jakobsen dan Knudsen[14,15] mendemonstrasikan bahwa cipher yang aman terhadap serangan kriptanalisis deferensial sederhana mungkin akan mudah diserang dengan *Higher-order differential cryptanalysis.*

5. **Algebraic Attacks**

   Algebraic attack akan mengekploitasi komponen-komponen dari cipher yang dapat direpresentasikan oleh operasi – operasi dalam *certain algebraic structure* (group,ring, dan field[16])

5. Kriptanalisis Deferential

Kriptanalisis differensial merupakan kriptanalisis yang memanfaatkan serangat secara statistik yang dapat diaplikasikan untuk berbagai pemetaan berulang. Metode ini di populerkan oleh Biham dan Shamir[1].

Differential Kriptanalisis didasarkan pada observasi dalam jumlah besar dari chipertext X yang mengenkripsi plaintext P, dan keterkaitan nilai X* selama pengenkripsian P*, maka *difference* X` = X + X*, dimana + merupakan komponen dari wise XOR. Untuk memanfaatkan teknik serangan Biham-Shamir ini, 2^{47} pasangan plaintext diperlukan untuk menentukan kunci dari DES. Secara substansial, akan diperlukan pasangan plaintext yang lebih sedikit apabila serangan dilakukan pada putaran Feistel yang lebih sedikit daripada yang seharusnya.

Keuntungan dari Kriptanalisis deferensial kemudahan untuk memprediksi *output difference* dari operasi linier yang diberikan *input difference* :

1. ***Operasi Unary(E,P,IP)***
   $(P(X))` = P(X) + P(X^*) = P(X`)$

2. ***Operasi Binary***
   $(X + Y)` = (X + Y) + (X^* + Y^*) = X`+Y`$

3. ***Mixing the key***
   $(X + K)` = (X + K) + (X^* + K) = X`$

Tanda + menerangkan operasi bitwise XOR.
Dari hal diatas menunnjukkan bahwa perbedaan (*differences*) adalah linier pada operasi linier dan pada faktanya, hasilnya adalah *key independent*.

6. Efisiensi Algoritma untuk properti komputasi deferensial

Sampai saat ini permasalahan melakukan pengevaluasian properti diferensial dari penjumlahan yang berkenaan dengan XOR adalah sangat sulit. Oleh sebab itu, efisiensi algoritma mengabaikan XOR dan akan menggunakan penjumlahan yang selalu modulo 2^n.

Algoritma untuk menghitung propabilistik deferensial (DP) untuk penambahan $DP^+(\alpha,\beta \dashrightarrow \gamma) := P_{x,y}[(x+y)\text{xor}((x \text{ xor } \alpha)+(y \text{ xor } \beta))= \gamma]$ adalah eksponensial pada *n*. Kompleksitas dari algoritma tersebut untuk memaksimmkan probabilitas differensial $DP^+_{max}(\alpha, \beta) := \max_{\gamma} DP^+(\alpha, \beta \dashrightarrow \gamma)$, sedangkan untuk maksimum ganda probabilitas differensial $DP^+_{2max}(\alpha) := \max_{\gamma, \beta} DP^+(\alpha, \beta \dashrightarrow \gamma)$.Untuk berbagai properti diferensial yang lain dari operasi penjumlahan juga eksponensial pada *n*.

Efisiensi algoritma yang diajukan oleh [3] adalah sebagai berikut :

Log-time algorithm untuk (a,b)-optimal g

```
INPUT :(a,b)
OUTPUT:(a,b)-optimal g
    1.  r <-- a ∘ 1
    7.  e <-- not(a xor b)∘ not(r)
    8.  a <-- e ∘ (e <<1) ∘ (a xor (a << 1))
    9.  p <-- aop^r (a)
    10. a <-- (a V (a >> 1)) ∘ not(r)
    11. b <-- (a V e) << 1
    12. g <-- ((a xor p) ∘ a) V ((a xor b xor (a <<
        1)) ∘ not(a) ∘ b)  V (a ∘ not(a)  ∘not(b))
    13. g <-- (g ∘ not(1)) V ((a xor b) ∘ 1)
    14. return g
```

Log-time algorithm untuk $DP^+_{2max}(a)$

```
INPUT : a
OUTPUT:DP+ 2max(a)
    1.  Return 2^-whi(Cr(a,a),mask(n-1))
```

Sedangkan algoritma Hamming dan aop adalah sebagai berikut :

Hamming Weight (Wh)

```
INPUT : x
OUTPUT:Wh(x)
    1.  x <-- x −((x >> 1) ∘ 0x55555555L )
    2.  x <-- (x ∘ 0x33333333L) + ((x >> 2) ∘
        0x33333333L)
    3.  x <-- (x +(x >> 4)) ∘ 0x0F0F0F0FL
    4.  x <-- x + (x >> 8)
    5.  x <-- (x +(x >> 16)) ∘ 0x0000003FL
    6.  Return x
```

Log-time algorithm untuk all-one-parity [aop(x)]

```
INPUT : x E ∑^n , n adalah perpangkatan 2 (power
of 2)
OUTPUT:aop(x)
1.  x[1] = x ∘ (x >> 1)
2.  For i <- 2 to log 2n-1 do x[i] <-- x[i-1] ∘(x[i]
    >> 2^i-1)
3.  y[i] <-- x ∘ not(x[1])
4.  For i <- 2 to  log 2n do y[i] <-- y[i-1] V ((y[i-
    1] >> 2^i-1 ) ∘ x[i-1])
5.  return y[log 2n]
```

Efisiensi algoritma ini ditujukan untuk mengurangi waktu komputasi dari properti differensial yang menggunakan operasi penambahan modulo 2^n.Algoritma ini mampu mencapai kompleksitas Θ(log n) untuk hampir semua properti termasuk propabilitas differensial dengan penambahan modulo. Walaupun begitu, komputasi menggunakan brute force tetap digunakan untuk melakukan serangan. Efisiensi algoritma ini mampu mengurangi kompleksitas dari Omega(2^{4n}) menjadi Θ(log n).

Efisiensi algoritma ini untuk menjawab permasalahan yang terjadi dimana hanya sedikit strategi yang berhasil dari kriptanalisis differensial. Hal ini dikarenakan karena algoritma – algoritma penyandian telah mengalami kemajuan dimana chipertext memiliki struktur yang sangat komplek. Struktur yang komplek ini membuat evaluasi secara tepat dari komponen - komponen differensialnya menjadi tidak mungkin dikerjakan dengan mudah.

7. Twofish

Twofish menggunakan rangkaian jaringan Feistel dengan 16 putaran dengan penambahan *whitening* pada masukan dan keluaran.

Plaintext dibagi menjadi 32 bit kata. Pada tahap masukan *whitening* , data ini akan di XOR kan dengan 4 buah kunci. Kemuadian, dilanjutkan dengan memasuki jaringan Feistel sebanyak 16 kali.

Pada tiap-tiap putaran, 2 kata pada bagian kiri digunakan sebagai input fungsi g. (Salah satu dari 2 kunci tersebut dirotasi sebesar 8 bit terlebih dahulu). Fungsi g terdiri dari 4 *byte-wide key-dependent* S-box, diikuti oleh tahap *linear mixing* berdasarkan MDS matrix. Hasil dari 2 fungsi g kemudian digabungkan menggunakan *Pseudo Hadamard Transform* (PHT) , dan 2 kata kunci ditambahkan. Dua buah hasil dari proses ini kemudian di XOR kan kedalam kata pada bagian kanan (salah satu dari 2 hasil tersebut di rotasi ke kiri dengan 1 bit, yang lain di rotasi ke sebelah kanan). Hasil dari operasi dari blok kanan dan kiri kemudian dipertukarkan sebagai masukan untuk tahap selanjutnya.

Secara formal, 16 bytes plainteks p0,…..p15 dibagi menjadi 4 kata P0,….,P3 yang berisi 32 bit mnggunakan konvensi little-endian.

$$P_i = \sum_{j=0}^{3} p_{(4i+j)} \cdot 2^{8j} \qquad i = 0, \ldots, 3$$

Pada tahap masuan *whitening* , kata – kata ini di XOR kan dengan 4 kata dari kunci internal.

$$R_{0,i} = P_i \oplus K_i \qquad i = 0, \ldots, 3$$

Pada tiap – tiap 16 putaran, 2 kata pertama digunakan sebagai input untuk fungsi F. Kata ketiga di XOR kan dengan keluaran pertama dari fungsi F serta kemudian di rotasi kekiri sebesar 1 bit. Sedangkan kata keempat di rotasi kekiri 1 bot serta kemudian di XORkan dengan keluaran kedua dari fungsi F.

$$\begin{aligned}
(F_{r,0}, F_{r,1}) &= F(R_{r,0}, R_{r,1}, r) \\
R_{r+1,0} &= \mathrm{ROR}(R_{r,2} \oplus F_{r,0}, 1) \\
R_{r+1,1} &= \mathrm{ROL}(R_{r,3}, 1) \oplus F_{r,1} \\
R_{r+1,2} &= R_{r,0} \\
R_{r+1,3} &= R_{r,1}
\end{aligned}$$

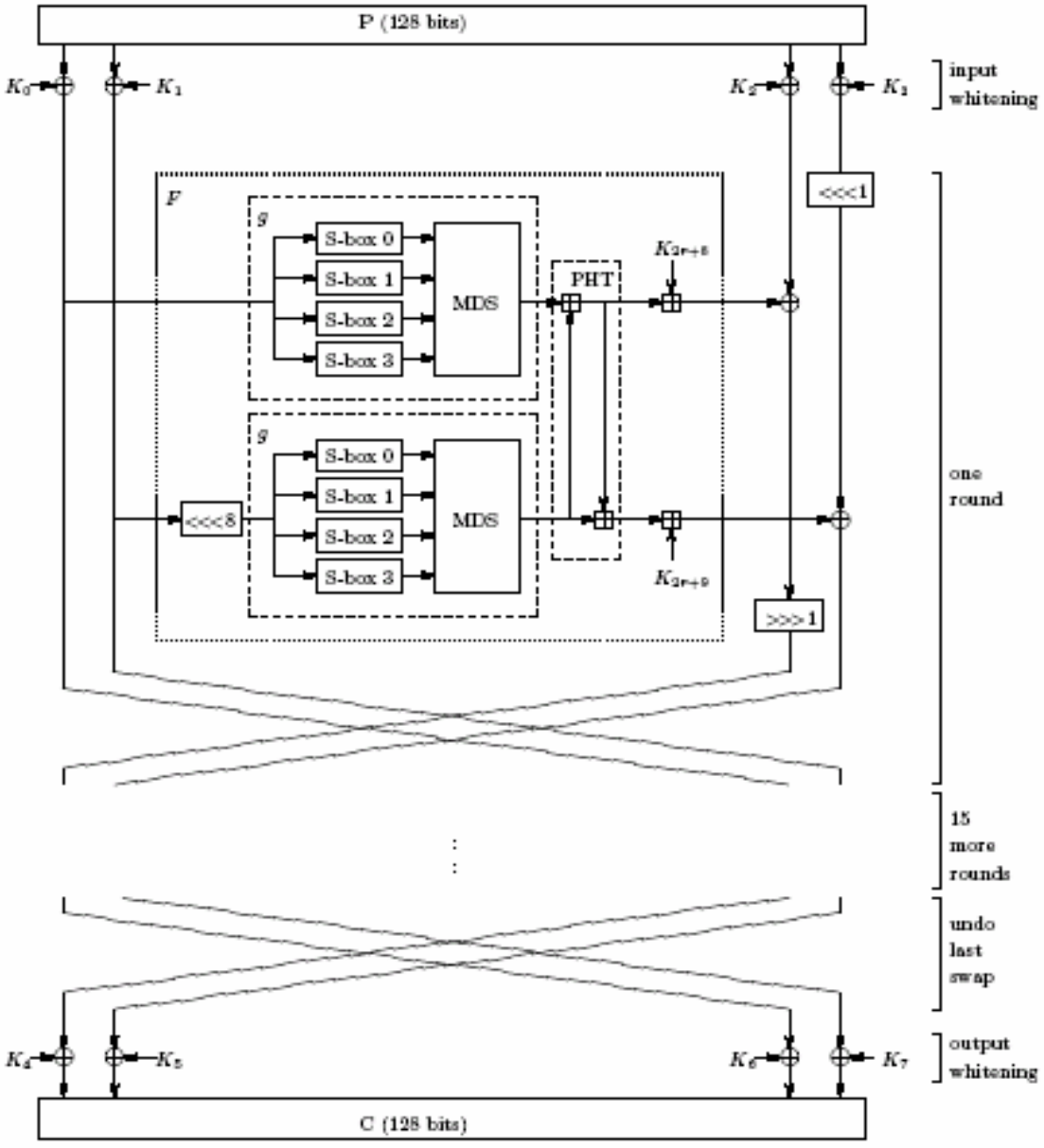
Untuk r=0,…,15 dan dimana ROR dan ROL adalah fungsi yang merotasi argumentasi pertama (32-bit kata) ke kiri atau kekanan oleh sejumlah bit yang diindikasikan oleh argument kedua.

Pada tahap keluaran whitening tidak dilakukan XOR pada putaran terakhir, dan XOR dikenakan dengan 4 kata dari kunci internal.

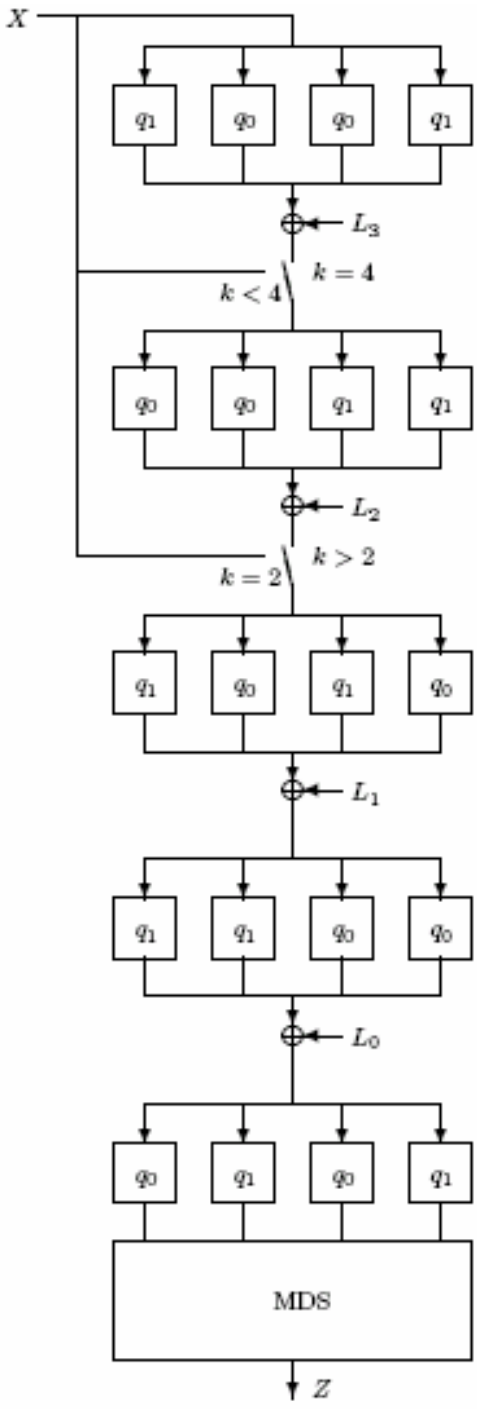
$$C_i = R_{16,(i+2) \bmod 4} \oplus K_{i+4} \qquad i = 0, \ldots, 3$$

Empat kata dari ciphertext dituliskan sebagai 16 byte $_{c0,\ldots,c15}$ menggunakan konvensi little-endian yang digunakan untuk plainteks.

$$c_i = \left\lfloor \frac{C_{\lfloor i/4 \rfloor}}{2^{8(i \bmod 4)}} \right\rfloor \bmod 2^8 \qquad i = 0, \ldots, 15$$



Skema Algoritma Twofish



Skema fungsi h dari Twofish

7.1. Fungsi F

Fungsi F adalah sebuah permutasi *key-dependent* pada nilai 64 bit. Fungsi ini memerlukan 3 argument yakni dua masukan kata R0 dan R1 serta nomor putaran r yang digunakan untuk memilih subkey yang sesuai. R0 di alirkan ke fungsi g, yakni T0 sebagai hasilnya. R1 di rotasikan kekiri sebesar 8 bit dan kemudian dialirkan ke fungsi g, yakni T1 sebagai hasilnya. Hasil T0 dan t1 kemudian di kombinasikan dalam PHT dan 2 kata dari pengembangan kunci ditambahkan.

$$\begin{aligned} T_0 &= g(R_0) \\ T_1 &= g(\mathrm{ROL}(R_1, 8)) \\ F_0 &= (T_0 + T_1 + K_{2r+8}) \bmod 2^{32} \\ F_1 &= (T_0 + 2T_1 + K_{2r+9}) \bmod 2^{32} \end{aligned}$$

Dimana (F0,F1) adalah hasil dari F.

7.2. Fungsi g

Fungsi g merupakan kunci dari algoritma Twofish. Masukan kata X di bagi menjadi 4 byte. Masing-masing byte di jalankan melewati masing-masing *key-dependent* S-box.S-box menerima masukan 8 bit dan menghasilkan 8 bit keluaran. Keempat hasil di interpretasikan sebagai vector dengan panjang 4 dari GF(2^8), dan dilakukan perkalian matriks MDS 4x4.

$$\begin{aligned} x_i &= \lfloor X/2^{8i} \rfloor \bmod 2^8 \qquad i = 0, \ldots, 3 \\ y_i &= s_i[x_i] \qquad i = 0, \ldots, 3 \\ \begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} &= \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \mathrm{MDS} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix} \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix} \\ Z &= \sum_{i=0}^{3} z_i \cdot 2^{8i} \end{aligned}$$

Dimana s_i adalah *key-dependent* S-box dan Z adalah hasil dari fungsi g. GF(2^8) direpresentasikan dengan GF(2)[x]/v(x) dimana $v(x) = x^8 + x^6 + x^5 + x^3 + 1$ adalah primitive polynomial dari tingkat delapan GF(2). Field elemen $a = \sum^7_{i=0}$ dengan ai merupakan elemen bagian GF(2) di identifikasikan dengan nilai byte $\sum^7_{i=0} a_i 2^i$. MDS matrik yang digunakan adalah :

$$\mathrm{MDS} = \begin{pmatrix} 01 & \mathrm{EF} & 5\mathrm{B} & 5\mathrm{B} \\ 5\mathrm{B} & \mathrm{EF} & \mathrm{EF} & 01 \\ \mathrm{EF} & 5\mathrm{B} & 01 & \mathrm{EF} \\ \mathrm{EF} & 01 & \mathrm{EF} & 5\mathrm{B} \end{pmatrix}$$

7.3. Penjadwalan kunci

Penjadwalan kunci harus mampu menyediakan 40 kata kunci internal K0,…,K39 dan 4 *key-dependent* S-box yang digunakan pada fungsi g.Penanmabahn padding akan dilakukan apabila panjang kunci kurang dari yang ditentukan (128,192,256) akan ditambahkan padding di akhir dengan nilai *zero*.

Nilai k = N/64. Kunci M terdiri dari 8k byte $m_0,\ldots,m_{8k-1}$.Byte-byte tersebut di konversikan kedalah 2k kata berisi 32 bit dengan masing masing

$$M_i = \sum_{j=0}^{3} m_{(4i+j)} \cdot 2^{8j} \qquad i = 0, \ldots, 2k-1$$

Dan kemudian dimasukkan kedalam vector dengan panjang k

$$\begin{aligned} M_e &= (M_0, M_2, \ldots, M_{2k-2}) \\ M_o &= (M_1, M_3, \ldots, M_{2k-1}) \end{aligned}$$

Tiga vector kata dengan panjang k juga diturunkan dari kunci. Hal ini dilakukan dengan menempatkan byte-byte kunci dalam pengelompokan 8 yang merepresentasikan sebagai vector GF(2^8), dan di lakukan operasiperkalian dengan matrik 4x8 yang diturunkan dari RS. Masing-masing hasil yang berisi 4 byte kemudian diinterpretasikan sebagai 32-bit kata.

$$\begin{pmatrix} s_{i,0} \\ s_{i,1} \\ s_{i,2} \\ s_{i,3} \end{pmatrix} = \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \mathrm{RS} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix} \cdot \begin{pmatrix} m_{8i} \\ m_{8i+1} \\ m_{8i+2} \\ m_{8i+3} \\ m_{8i+4} \\ m_{8i+5} \\ m_{8i+6} \\ m_{8i+7} \end{pmatrix}$$

$$S_i = \sum_{j=0}^{3} s_{i,j} \cdot 2^{8j}$$

Untuk i=0,…,k-1 , dan

$$S = (S_{k-1}, S_{k-2}, \ldots, S_0)$$

RS matrik yang diberikan adalah :

$$\mathrm{RS} = \begin{pmatrix} 01 & \mathrm{A4} & 55 & 87 & 5\mathrm{A} & 58 & \mathrm{DB} & 9\mathrm{E} \\ \mathrm{A4} & 56 & 82 & \mathrm{F3} & 1\mathrm{E} & \mathrm{C6} & 68 & \mathrm{E5} \\ 02 & \mathrm{A1} & \mathrm{FC} & \mathrm{C1} & 47 & \mathrm{AE} & 3\mathrm{D} & 19 \\ \mathrm{A4} & 55 & 87 & 5\mathrm{A} & 58 & \mathrm{DB} & 9\mathrm{E} & 03 \end{pmatrix}$$

7.3.1. Penambahan panjang kunci

Panjang kunci yang didefinisikan untuk twofish memiliki panjang 128 bit, 192 bit , serta 256 bit. Untuk masukan dengan panjang kunci yang tidak memenuhi criteria diatas maka akan ditambahkan *zero byte* pada akhir kunci sampai panjang kunci memenuhi salah satu dari criteria diatas

7.3.2. Fungsi h

Kata (word) di bagi kedalam byte.

$$\begin{aligned} l_{i,j} &= \lfloor L_i/2^{8j} \rfloor \bmod 2^8 \\ x_j &= \lfloor X/2^{8j} \rfloor \bmod 2^8 \end{aligned}$$

Untuk i=0,…,k-1 dan j=0,…3. Kemudian substitusi sekuensial dan operasi XOR diaplikasikan.

$$y_{k,j} = x_j \qquad j = 0, \ldots, 3$$

Apabila k= 4 , maka

$$\begin{aligned} y_{3,0} &= q_1[y_{4,0}] \oplus l_{3,0} \\ y_{3,1} &= q_0[y_{4,1}] \oplus l_{3,1} \\ y_{3,2} &= q_0[y_{4,2}] \oplus l_{3,2} \\ y_{3,3} &= q_1[y_{4,3}] \oplus l_{3,3} \end{aligned}$$

Apabila k >= 3 , maka

$$\begin{aligned} y_{2,0} &= q_1[y_{3,0}] \oplus l_{2,0} \\ y_{2,1} &= q_1[y_{3,1}] \oplus l_{2,1} \\ y_{2,2} &= q_0[y_{3,2}] \oplus l_{2,2} \\ y_{2,3} &= q_0[y_{3,3}] \oplus l_{2,3} \end{aligned}$$

Untuk semua kasus, maka

$$\begin{aligned} y_0 &= q_1[q_0[q_0[y_{2,0}] \oplus l_{1,0}] \oplus l_{0,0}] \\ y_1 &= q_0[q_0[q_1[y_{2,1}] \oplus l_{1,1}] \oplus l_{0,1}] \\ y_2 &= q_1[q_1[q_0[y_{2,2}] \oplus l_{1,2}] \oplus l_{0,2}] \\ y_3 &= q_0[q_1[q_1[y_{2,3}] \oplus l_{1,3}] \oplus l_{0,3}] \end{aligned}$$

Disini, q0 dan q1 adalah permutasi yang telah *fix* dengan nilai 8-bit.Vektor keluaran y_i kemudian di lakukan operasi perkalian matrik MDS.

$$\begin{pmatrix} z_0 \\ z_1 \\ z_2 \\ z_3 \end{pmatrix} = \begin{pmatrix} \cdot & \cdots & \cdot \\ \vdots & \text{MDS} & \vdots \\ \cdot & \cdots & \cdot \end{pmatrix} \cdot \begin{pmatrix} y_0 \\ y_1 \\ y_2 \\ y_3 \end{pmatrix}$$

$$Z = \sum_{i=0}^{3} z_i \cdot 2^{8i}$$

7.3.3. Key-dependent S-box

S-box pada fungsi g dapat didefinisikan sebagai

$$g(X) = h(X, S)$$

Untuk i=0,…,3 *key-dependent* S-box s_i di bentuk dengan melakukan pemetaan dari x_i ke y_i dalam fungsi h, dimana list L sama dengan vector S yang diturunkan dari kunci.

7.3.4. Pengembangan kunci K_j

Kata yang dihasilkan dari pengembangan kunci didefinisikan menggunakan fungsi h.

$$\begin{aligned} \rho &= 2^{24} + 2^{16} + 2^8 + 2^0 \\ A_i &= h(2i\rho, M_e) \\ B_i &= \mathrm{ROL}(h((2i+1)\rho, M_o), 8) \\ K_{2i} &= (A_i + B_i) \bmod 2^{32} \\ K_{2i+1} &= \mathrm{ROL}((A_i + 2B_i) \bmod 2^{32}, 9) \end{aligned}$$

Konstanta ρ digunakan untuk melakukan duplikasi byte; Ini memiliki property bahwa untuk i=0,…,255, kata i ρ terdiri dari 4 byte yang sama, masing-masing dengan dengan nilai i. Fungsi h diaplikasikan untuk kata dari tipe ini. Untuk A_i , nilai byte nya adalah 2i, dan argument kedua h adalah M_e . Ai dan Bi dikombinasikan dalam PHT. Salah satu dari hasil kemudian di lakukan rotasi 9 bit. Dua buah hasil operasi akan membentuk 2 kata sebagai pengembangan dari kunci

7.3.5. Permutasi q_0 dan q_1

Permutasi q_0 dan q_1 merupakan permutasi dengan nilai 8 bit. Masing – masing dibangun dari 4-bit permutasi yang berbeda. Untuk setiap nilai x, korespondensi nilai keluaran y didefinisikan sebagai berikut :

$$
\begin{aligned}
a_0, b_0 &= \lfloor x/16 \rfloor, x \bmod 16 \\
a_1 &= a_0 \oplus b_0 \\
b_1 &= a_0 \oplus \mathrm{ROR}_4(b_0, 1) \oplus 8a_0 \bmod 16 \\
a_2, b_2 &= t_0[a_1], t_1[b_1] \\
a_3 &= a_2 \oplus b_2 \\
b_3 &= a_2 \oplus \mathrm{ROR}_4(b_2, 1) \oplus 8a_2 \bmod 16 \\
a_4, b_4 &= t_2[a_3], t_3[b_3] \\
y &= 16\, b_4 + a_4
\end{aligned}
$$

Dimana ROR_4 adalah fungsi yang mirip dengan ROR yang dirotasi sebesar 4-bit. Untuk permutasi q_0 , 4 bit S-box yang digunakan

$$
\begin{aligned}
t_0 &= [\ 8\ 1\ 7\ D\ 6\ F\ 3\ 2\ 0\ B\ 5\ 9\ E\ C\ A\ 4\ ] \\
t_1 &= [\ E\ C\ B\ 8\ 1\ 2\ 3\ 5\ F\ 4\ A\ 6\ 7\ 0\ 9\ D\ ] \\
t_2 &= [\ B\ A\ 5\ E\ 6\ D\ 9\ 0\ C\ 8\ F\ 3\ 2\ 4\ 7\ 1\ ] \\
t_3 &= [\ D\ 7\ F\ 4\ 1\ 2\ 6\ E\ 9\ B\ 3\ 0\ 8\ 5\ C\ A\ ]
\end{aligned}
$$

Dimana masing-masing 4-bit S-box direpresentasikan dengan sebuah list masukan menggunakan notasi hexadecimal. Untuk permutasi q_1 , 4-bit S-box yang digunakan

$$
\begin{aligned}
t_0 &= [\ 2\ 8\ B\ D\ F\ 7\ 6\ E\ 3\ 1\ 9\ 4\ 0\ A\ C\ 5\ ] \\
t_1 &= [\ 1\ E\ 2\ B\ 4\ C\ 3\ 7\ 6\ D\ A\ 5\ F\ 9\ 0\ 8\ ] \\
t_2 &= [\ 4\ C\ 7\ 5\ 1\ 6\ 9\ A\ 0\ E\ D\ 8\ 2\ B\ 3\ F\ ] \\
t_3 &= [\ B\ 9\ 5\ 1\ C\ 3\ D\ E\ 6\ 4\ 7\ F\ 2\ 0\ 8\ A\ ]
\end{aligned}
$$

8. Penerapan Kriptanalisis Deferensial pada Twofish

Usaha Kriptanalisis untuk melakukan pengujian keamanan Twofish menggunakan *truncated differential Cryptanalysis*. Tipe dari truncated differential Cryptanalysis adalah menggunakan karakteristik *byte*, karena nilai dari *difference* pada *byte* dapat dibedakan antara 1 dan 0, serta pengukuran dari *difference* adalah XOR.Penggunaan karakteristik *byte wise* akan memudahkan investigasi dengan seksama pada *non-uniformity* distribusi *difference*.

Usaha untuk melakukan kriptanalisis differensial digunakan komputasi yang telah diefisienkan[5]. Perbandingan kompleksitas yang dihasilkan dapat dilihat pada tabel I. Algoritma yang telah diefisienkan tersebut dapat diperluas untuk melakukan komputasi probabilitas *truncated differential* pada operasi penjumlahan 2^n.

Putaran	whitening	kunci	kriptanalisis	kompleksitas	kondisi	referensi
6	w/o	128	imposible differential	2^{128}		[6]
6	w/o	192	imposible differential	2^{160}		[6]
6	w/o	256	imposible differential	2^{192}		[6]
6	w/	256	imposible differential	2^{256}		[6]
8	w/	*any*	differential attack	-	$> 2^{-20}$ fraksi dari S-box	[5]

Tabel I. Perbandingan kompleksitas *efficient algorithm*

Untuk x,y,z E $GF(2)^n$, fungsi dari mod 2^n didefinisikan sebagai berikut :

$$f(x, y) = x + y = z \pmod{2^n}.$$

Delta(x) E $GF(2)^n$ dibagi menjadi t-bit sub blok yang dapat dinyatakan sebagai :

$$\Delta x = (\Delta x_{m-1}^{[t]}, \ldots, \Delta x_1^{[t]}, \Delta x_0^{[t]}),$$

dimana m = n/t adalah jumlah dari sub blok.

Truncated Differential Probabilities (TDP) untuk suatu fungsi f didefinisikan sebagai berikut :

$$\mathrm{TDP}_f(\delta x, \delta y, \delta z) = \frac{1}{c} \sum_{\chi(\Delta x, \Delta y, \Delta z) = (\delta x, \delta y, \delta z)} \mathrm{DP}_f(\Delta x, \Delta y, \Delta z), \quad (1)$$

dimana c adalah sejumlah pasangan (D(x),D(y)) yang memenuhi kondisi X(D(x),D(y)) = (delta(x),delta(y)). Dengan memasukkan persamaan Hamming weight, maka nilai c dapat diperoleh dari :

$$c = (2^t - 1)^{\mathrm{Wh}(D(x)) + \mathrm{Wh}(D(y))}$$

Apabila kita memiliki n=32 dan t = 8 (konfigurasi karakteristik *byte*) maka jumlah yang mungkin dari *truncated differential* adalah $(2^{n/t})^3 = 2^{12.}$.

Jika Wh(D(x)) + Wh(D(y)) >= 6, maka c >= 2^{48} . Nilai c ini sangatlah besar sehingga akan menyebabkan biaya untuk melakukan komputasi TDP akan sangat mahal, walaupun perhitungan DP nya sendiri dapat dilakukan efisiensi.

Untuk mengatasi permasalahan tersebut, [4] melakukan masing-masing sub blok secara independent. Secara spesifiknya, akan dilakukan komputasi PS (*partial sum of differential probabilities*) untuk masing-masing sub-blok dengan mengabaikan *carry* dari satu sub-blok ke sub-blok selanjutnya. Kemudian hasil dari masing-masing PS akan digabungkan untuk mendapatkan nilai TDP untuk suatu fungsi f.

Untuk tiap-tiap sub-blok perlu di dipertimbangkan *difference* pada *carryin* (Dcin) dari sub-blok sebelumnya dan *difference* dari carryout(Dcout) untuk sub-blok selanjutnya.

- Nilai Dcin adalah 0 atau 1
- Peluang dari Dcout (P(Dcout))adalah 0,0.5,1

$$\text{TDP} = \sum_{\text{Condition PS}}^{\text{PS}(\delta x_i, \delta y_i, \delta z_i, d, p)} \text{DP}(\Delta x_i^{[t]}, \Delta y_i^{[t]}, \Delta z_i^{[t]}),$$

dimana *Condition PS* adalah

$$\begin{aligned} \chi(\Delta x_i^{[t]}, \Delta y_i^{[t]}, \Delta z_i^{[t]}) &= (\delta x_i, \delta y_i, \delta z_i), \\ \Delta cin &= d, \\ P_{\Delta cout} &= p. \end{aligned}$$

TDP_{MDS} atau TDP (Maximum Distance Separable) didefinisikan sebagai berikut :

$$\text{TDP}_{\text{MDS}}(\delta x, \delta y) = \frac{1}{c} \sum_{\chi(\Delta x, \Delta y) = (\delta x, \delta y)} \Pr[\text{MDS}(x) \oplus \text{MDS}(x \oplus \Delta x) = \Delta y],$$

dimana c adalah nilai delta(x) yang memenuhi kondisi X(delta(x)) = Dx

TDP_{MDS} yang ditentukan oleh *Hamming weight* dari Dx dan Dy adalah :

$w_H(\delta x)$	$w_H(\delta y)$ 0	1	2	3	4
0	1	0	0	0	0
1	0	0	0	0	1
2	0	0	0	$2^{-7.994}$	$2^{-0.023}$
3	0	0	$2^{-15.989}$	$2^{-8.017}$	$2^{-0.023}$
4	0	$2^{-23.983}$	$2^{-16.012}$	$2^{-8.017}$	$2^{-0.023}$

Probabilitas *Truncated Differential* MDS

Hasil dari truncated defferential dengan banyaknya putaran :

(1) 16 Putaran

round		probability
1	0 0 0 1	$2^{0.000000}$
2	0 1 f f	$2^{-0.028330}$
3	f f f e	$2^{-8.118785}$
4	f e f f	$2^{-8.209239}$
5	f f 7 f	$2^{-16.299694}$
6	7 f f f	$2^{-16.390147}$
7	f f b f	$2^{-24.480603}$
8	b f f f	$2^{-24.571056}$
9	f f 7 f	$2^{-32.661511}$
10	7 f f f	$2^{-32.751965}$
11	f f b f	$2^{-40.842420}$
12	b f f f	$2^{-40.932874}$
13	f f 7 f	$2^{-49.023329}$
14	7 f f f	$2^{-49.113783}$
15	f f b f	$2^{-57.204238}$
16	b f f f	$2^{-57.294692}$

Dengan probabilitas sebesar $2^{-57..3}$, maka dengan perkiraan akan didapatkan sebuah sebuah pasangan <plaintext,ciphertext> dari 2^{100} chosen-plaintext, dan ada sekitar 2^{28} pasangan yang bagus. Nilai probabilitas yang dihasilkan lebih tinggi dari pada yang pernah dihasilkan oleh Knudsen yakni 2^{256} , sehingga kemungkinan jumlah pasangan yang bagus untuk metoda differensial yang dilakukan juga akan semakin besar.

(2) 12 Putaran

round		probability
1	0 0 0 1	$2^{0.000000}$
2	0 1 f f	$2^{-0.028330}$
3	f f f e	$2^{-8.118785}$
4	f e f f	$2^{-8.209239}$
5	f f 7 f	$2^{-16.299694}$
6	7 f f f	$2^{-16.390147}$
7	f f b f	$2^{-24.480603}$
8	b f f f	$2^{-24.571056}$
9	f f 7 f	$2^{-32.661511}$
10	7 f f f	$2^{-32.751965}$
11	f f b f	$2^{-40.842420}$
12	b f f f	$2^{-40.932874}$

Dari data diatas, diperkirakan akan mendapatkan satu pasangan <plainteks,cipherteks> yang bagus dari 2^{34} *chosen-plaintext* dengan menggunakan struktur dari byte terakhir pada plaintext. Terdapat 2^{94} kira-kira pasangan yang bagus.

9. Kajian hasil kriptanalisis differensial pada Twofish

Shenier sebagai salah satu dari perancang algoritma Twofish memberikan beberapa tanggapan dari usaha kriptanalisis deferensial pada Twofish[18]

(a) Shiho dan Lisa mendapatkan data untuk 12-putaran diperoleh *truncated differential* sebesar $2^{-40.9}$ sehingga diperkirakan bahwa byte kedua dari *chipertext difference* akan bernilai 0 ketika plaintext difference adalah *all-zeros* kecuali pada byte terakhirnya.

Shenier memperkirakan bahwa kemungkinan byte kedua dari ciphertext akan bernilai 0 memiliki peluang 2^{-8} , nilai ini lebih besar daripada probabilitas yang di hasilkan oleh Shiho

(b) Seperti halnya pada point (a), nilai probabilitas untuk menyatakan bahwa byte kedua dari *chipertext difference* akan bernilai 0 ketika *plaintext difference* adalah *all-zeros* kecuali pada byte terakhirnya juga sangat kecil dari 2^{-8}.

Namun, apabila byte kedua dari chipertext difference akan 0 dengan probabilitas $2^{-8}+2^{-57.3}$ untuk Twofish adalah benar, maka terdapat kemungkinan untuk melakukan serangan secara khusus dengan 2^{100} *chosen-plaintext* . Akan tetapi praduga ini masih belum dapat dipastikan kebenarannya secara pasti.

10. Kesimpulan

.

- Kriptanalisis diferensial sering digunakan untuk menguji ketangguhan algoritma penyandian yang memanfaatkan jaringan Feistel.
- Twofish yang memanfaatkan jaringan Feistel merupakan algoritma yang cocok untuk dilakukan kriptanalisis diferensial
- Kriptanalisis differensial memberikan pengetahuan tambahan mengenai teknik untuk melakukan kriptanalisis terhadap cipherteks.
- Efisiensi algoritma yang dikembangkan oleh Shiho mampu menurunkan kompleksitas penambahan modulo 2^n menjadi O(log n)
- Kriptanalisis differential yang dikemukakan oleh Shiho masih belum dapat dijadikan bahwa algoritma Twofish telah terpecahkan.

11. Daftar Pustaka

[1] Biham,Shamir,"Differential Cryptanalysis of the Data Encryption Standard",Springer Verlag,1993

[2] http://www.faqs.org/faqs/cryptography-faq/part05

[3] Helger Lipmaa dan Shiho Moriai.Efficient Algorithms for Computing Differential Properties of Addition

[4] Moriai,Shiho, Yin,Yiqun Lisa, "Cryptanalysis of Twofish (II)",2000

[5] Lipmaa,Helger , Moriai, Shiho, "Efficient Algorithms for Computing Differential Properties of Addition",2000

[6] N. Ferguson,"Impossible differential in Twofish", Twofish Technical Report #5,Oktober 5, 1999.

[7] L. Knudsen, *Block Ciphers—Analysis, Design and Applications,* Ph.D. Thesis,University of Aarhus, Aarhus, Denmark, 1994

[8] A.J. Menezes, P.C. van Oorschot, and

S.A. Vanstone, *Handbook of Applied Cryptography*,CRC Press, 1997

[9] Keliher, Liam, *Linear Cryptanalysis Od Substitutuion-Permutation Networks*,Ph.D Thesis, Queen's University, Ontarion, Canada,2003

[10] M. Matsui, *Linear cryptanalysis method for DES cipher,* Advances in Cryptology—EUROCRYPT'93, LNCS 765, pp. 386–397, Springer-Verlag,1994.

[11] M. Matsui, *The first experimental cryptanalysis of the Data Encryption Standard,*Advances in Cryptology—CRYPTO'94, LNCS 839, pp. 1–11, Springer-Verlag,1994.

[12] M. Matsui and A. Yamagishi, *A new method for known plaintext attack of FEAL cipher,* Advances in Cryptology—EUROCRYPT'92, LNCS 658, pp. 81–91, Springer-Verlag, 1993.

[13] L.R. Knudsen, *Truncated and higher order differentials,* Fast Software Encryption:Second International Workshop, LNCS 1008, pp. 196–211, Springer-Verlag,1995

[14] T. Jakobsen and L. Knudsen, *The interpolation attack on block ciphers,* FastSoftware Encryption (FSE'97), LNCS 1267, pp. 28–40, Springer-Verlag, 1997

[15] T. Jakobsen and L. Knudsen, *Attacks on block ciphers of low algebraic degree,*Journal of Cryptology, Vol. 14, No. 3, pp. 197–210, 2001.

[16] T. Hungerford, *Algebra,* Graduate Texts in Mathematics 73, Springer-Verlag,1974.

[17] Schneier, Bruce, et al, *Twofish : A 128-Bit Block Cipher,*June 15, 1998.

[18] Schneir,Bruce, *http://www.schneier.com/blog/archives/2005/11/twofish_cryptan.html,* November 23,2005