Studi dan Analisis Mengenai Amplified Boomerang Attack

Timotius Grady Limandra (13504082)
Jurusan Teknik Informatika ITB, email: if14082@students.if.itb.ac.id

Abstraksi – Makalah ini akan membahas mengenai perkembangan salah satu jenis serangan pada kriptografi, yaitu *Amplified boomerang attack*. Serangan ini dapat dikategorikan sebagai *Chosen-plaintext attack* Pada awalnya penulis akan membahas mengenai perkembangan *Amplified boomerang attack* yang merupakan pengembangan dari *Boomerang attack*. Penulis juga akan melakukan perbandingan terhadap dua jenis serangan ini. Kemudian penulis akan memberikan contoh serangan terhadap suatu algoritma tertentu.

Kata Kunci: *Amplified boomerang attack*, *Chosen-plaintext attack*, *Boomerang attack*, *MARS Core*, *Serpent*

1. PENDAHULUAN

Amplified boomerang attack, merupakan salah satu jenis serangan dalam kriptografi berbasis *differential cryptanalysis* yang biasanya merupakan *Chosen-plaintext attack*. Dalam penerapannya, *Chosen-plaintext attack*, kriptanalis dapat memilih plainteks tertentu untuk dienkripsikan yang biasanya plainteks – plainteks yang lebih mengarahkan penemuan kunci.

Dilihat dari sejarah perkembangannya, *amplified boomerang attack* merupakan pengembangan dari *boomerang attack* (diperkenalkan oleh David Wagner pada tahun 1999) dan *inside-out attack*. *Amplified boomerang attack* terbukti cukup efektif dalam melakukan serangan terhadap beberapa *cipher* blok, seperti MARS dan Serpent.

2. HASIL DAN PEMBAHASAN

Pada tahun 1999, David Wagner memperkenalkan 2 jenis serangan baru dalam kriptografi, yaitu: *boomerang attack* dan *inside-out attack*. Kedua jenis serangan ini merupakan konsep dasar dari *amplified boomerang attack*. Serangan dilakukan pada *cipher* blok E yang terdiri dari e_0 dan e_1. $E(X) = e_0(e_1(X))$

Untuk keteranggan lebih lanjut:
X_i = plainteks
$Y_i = e_0(X_i)$
$Z_i = e_1(Y_i)$ (cipherteks)

Inside-out Attack

Pertama - tama kita akan membahas terlebih dahulu cara kerja dari *inside-out attack*. Misalnya kita memiliki kondisi sebagai berikut:

$\Delta_0 \rightarrow \Delta_1$ melalui e_1
$\Delta_0 \rightarrow \Delta_2$ melalui e_0^{-1}

Dalam kasus ini, kita dapat melakukan serangan untuk memisahkan E dari permutasi acak dengan cara sebagai berikut:

1. Cari pasangan plainteks dan cipherteks di mana R pasang teks memiliki perbedaan di tengah – tengahnya (misal: pasangan $R_{(i,j)}$ di mana $Y_i \oplus Y_j = \Delta_0$)
2. Identifikasi pasangan input di mana $X_i \oplus X_j = \Delta_2$
3. Identifikasi pasangan output di mana $Y_i \oplus Y_j = \Delta_1$
4. Hitung jumlah pasangan yang *overlap* (pasangan yang tepat ada pada input maupun output). Jika perhitungan ini menghasilkan hasil yang lebih besar dari perkiraan yang kita dapatkan dari permutasi acak, kita dapat memisahkan E dari permutasi acak.

Anggap kita memiliki kemungkinan seperti berikut untuk pasangan i,j:
$Pr[X_i \oplus X_j = \Delta_2] = p_0$
$Pr[Z_i \oplus Z_j = \Delta_1] = p_1$

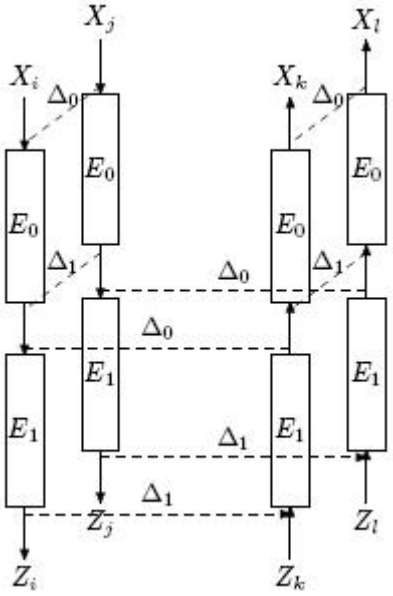
Kita dapat menarik kesimpulan bahwa kemungkinan untuk Δ_2 adalah p_0 dan kemungkinan untuk Δ_2 adalah p_0. Pada kasus ini kita dapat menarik kesimpulan:
N = Jumlah pasangan plaintks dan cipherteks
$N_0 = N * p_0$ = Jumlah pasangan input untuk permutasi acak
$N_0 = N * p_0 * p_1$ = Jumlah pasangan input yang juga merupakan pasangan output

Jumlah pasangan yang merupakan input yang juga merupakan pasangan output kemudian didistribusikan secara binomial dan dapat diperkirakan dengan distribus normal($\mu = N_1$ dan $\sigma \approx \sqrt{N_1}$). Ketika kita memiliki pasangan yang memenuhi $Y_i \oplus Y_j = \Delta_o$, maka i, j pasti merupakan pasangan baik untuk input maupun output. Misal R pasangan di dalam *cipher*, berarti E, di mana N_1 + R pasangan untuk input dan output, ketika dilakukan permutasi acak, akan sebesar N_1. Ketika R jauh lebih besar dari $\sqrt{N_1}$ akan menghasilkan deteksi dengan kemungkinan yang cukup tinggi. Hal ini memberikan celah untuk melakukan serangan ketika tidak ada perbedaan mencolok pada seluruh *cipher* dengan menunggu kemunculan perbedaan yang diperlukan yang muncul secara acak di dalam *cipher* tersebut.

Boomerang Attack

Selanjutnya kita akan membahas mengenai *boomerang attack* yang merupakan pendahulu dari

amplified boomerang attack. Misalkan, suatu cipher yang sama $E(X) = e_0(e_1(X))$, tetapi terdapat diferensial di antara e_0, e_0^{-1}, e_1^{-1}. Lalu, asumsikan bahwa ini adalah diferensial yang normal dan kemungkinan hanya memiliki satu. Serangan ini akan bekerja pada diferensial dengan probabilitas rendah.



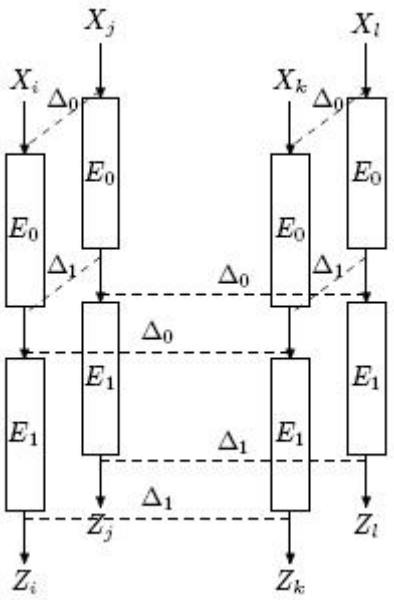
Gambar 1 Boomerang Attack

Dari gambar di atas dapat kita dapat memisahkan E dari permutasi acak dengan cara sebagai berikut:

1. Ambil pasangan untuk e_0 sebagai input, $X_0 \oplus X_1 = \Delta_0$
2. Kemudian e_0, terenkripsi menjadi $Y_0 \oplus Y_1$ dan memiliki relasi $Y_0 \oplus Y_1 = \Delta_1$
3. Kemudian e_1, terenkripsi menjadi $Z_0 \oplus Z_1$
4. Lalu kita membuat dua pasangan untuk e_0^{-1} dari pasangan ini dengan cara melakukan dekripsi $Z_2 = Z_0 \oplus \Delta_1$ dan $Z_3 = Z_1 \oplus \Delta_1$
5. Z_2,Z_3 dideskripsi menjadi Y_2,Y_3, dengan relasi $Y_2 \oplus Y_0 = \Delta_0$ dan $Y_3 \oplus Y_1 = \Delta_0$
6. Kemudian kita dapat menyimpulkan bahwa relasi antara Y_2 dan Y_3:
   $Y_0 \oplus Y_1 = \Delta_1$; $Y_0 \oplus Y_2 = \Delta_0$; $Y_1 \oplus Y_3 = \Delta_0$;
   Maka, $Y_2 \oplus Y_3 = \Delta_1$
7. Karena $Y_2 \oplus Y_3 = \Delta_1$ maka kita mendapatkan pasangan untuk e_0. Karena kita membahas mengenai diferensial normal, maka diferensial akan berjalan sebaliknya di sisi yang satunya lagi; $X_2 \oplus X_3 = \Delta_0$
8. Cara kerja serangan ini memilki probabilitas 2^{-128} untuk 128-bit *cipher* blok, jadi cara ini dapat digunakan secara efektif untuk memisahkan E dari permutasi acak

Amplified Boomerang Attack

Amplified boomerang attack sebenarnya merupakan gabungan dari *inside-out attack* dengan *boomerang attack* yang kemudian dikembangkan sehingga menjadi *Chosen-plaintext attack*.



Gambar 2 Amplified Boomerang Attack

Misalnya, kita berhadapan dengan 128-bit *cipher* blok, kemudian kita melakukan *request* terhadap 2^{65} pasangan *chosen* plainteks X_{2i}, X_{2i+1} yang berarti $X_{2i} \oplus X_{2i+1} = \Delta_0$. Karena kita membahas mengenai diferensial normal, maka diferensial akan berjalan sebaliknya di sisi yang satunya lagi; $Y_{2i} \oplus Y_{2i+1} = \Delta_1$. Karena terdapat 2 pasang, yaitu i dan j, maka $Y_{2i} \oplus Y_{2j} = \Delta_0$.

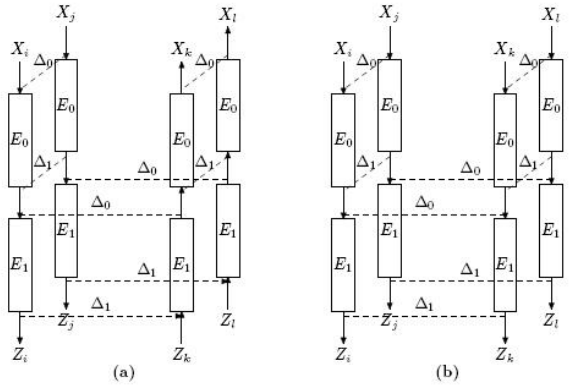
Dari gambar di atas dapat kita simpulkan bahwa:
$Y_{2i} \oplus Y_{2i+1} = \Delta_1$; $Y_{2j} \oplus Y_{2j+1} = \Delta_1$; $Y_{2i} \oplus Y_{2j} = \Delta_0$
Maka, $Y_{2i+1} \oplus Y_{2j+1} = \Delta_0$
Oleh karena itu, $Z_{2i} \oplus Z_{2j} = \Delta_1$; $Z_{2j+1} \oplus Z_{2j+1} = \Delta_1$;

Ada sekitar 2^{129} kemungkinan pasangan i,j. Probabilitas untuk setiap pasangan yang diberikan untuk 2 persamaan terakhir adalah 2^{-256}. Kita dapat menggunakan tekhnik di atas untuk memisahkan E dari permutasi acak.

Serangan ini disebut *amplified boomerang attack* karena struktur *boomerang* yang terbentuk memperkuat efek yang muncul karena kejadian dengan probabilitas rendah ($Y_{2i} \oplus Y_{2j} = \Delta_0$) sehingga mudah untuk terdeteksi.

Perbandingan Boomerang Attack dan Amplified Boomerang Attack

Perhatikan kedua gambar berikut ini;



Gambar 3 (a) Boomerang (b) Amplified Boomerang

Perbedaan yang langsung terlihat adalah arah serangan yang dilakukan kedua jenis serangan ini sangatlah berbeda. Di samping itu masih ada beberapa perbedaan lain pada kedua jenis serangan ini. Pada *boomerang attack* dibutuhkan *query* yang jauh lebih sedikit dibandingkan dengan *amplified boomerang attack*. Hal ini dikarenakan pada *amplified boomerang attack* kita membutuhkan lebih banyak pasangan input untuk mendapatkan relasi yang tepat di antara pasangan – pasangan tersebut.

Berikut ini adalah beberapa kelebihan *boomerang amplified attack* bila dibandingkan dengan *boomerang attack*:

1. Ketika akan melakukan sebuah serangan, sering kali kita harus menebak kunci pada akhir *cipher*. Dengan menggunakan serangan dengan jenis *choosen plaintexts / adaptive chosen ciphertext* kita harus menambah jumlah *request plaintext* ataupun *ciphertext* ketika harus menebak kunci pada sisi satunya lagi. Dengan menggunakan serangan dengan jenis *choosen plaintext only*, kita dapat menebak kunci pada akhir *cipher* tanpa harus menambah jumlah *chosen plaintexts* yang diminta.
2. Serangan ini tidak hanya dapat digunakan pada pasangan *tuple*, tapi juga dapat digunakan pada *k-tuples* (k = jumlah *tuple*).
3. Kita dapat menggunakan *boomerang amplifier attack* di sebagian kecil dari sebuah blok. Hal ini tidak dapat dilakukan jika menggunakan *boomerang attack* yang biasa.

MARS Core

Dalam kriptografi, MARS adalah salah satu *cipher* blok yang dirancang oleh IBM pada tahun 1998 (Don Coppersmith dan tim) MARS memiliki ukuran blok 128-bit dan variabel kunci di antara 128 sampai 448 bit. Tidak seperti *cipher* blok yang lain, MARS memliki struktur yang heterogen dan memiliki kunci yang yang memiliki relasi dengan data yang ada.

Serangan Amplified Boomerang Attack pada MARS Core

MARS memiliki *three round differential* dengan probabilitas: $(0,0,0,2^{31}) \rightarrow (2^{31},0,0,0)$. Selain itu, MARS juga memiliki *three round truncated differential* dengan probabilitas: $(0,0,0,\alpha) \rightarrow (\beta,0,0,0)$. Kita dapat menggunakan kedua karakteristik ini untuk melakukan *amplified boomerang attack* pada *six rounds cipher*.

Serangan dilakukan dengan cara meminta 2^{48} pasang input X_{2i}, X_{2i+1} yang memenuhi syarat $X_{2i} \oplus X_{2i+1} = (0,0,0,2^{31})$. Seperti yang sudah dijelaskan sebelumnya, pasangan tersebut akan dienkripsi menjadi Y_{2i}, Y_{2i+1} yang memenuhi syarat $Y_{2i} \oplus Y_{2i+1} = (2^{31},0,0,0)$. Setelah memiliki 2^{48} pasang tersebut, kita akan mencari satu pasang (i,j) di mana $Y_{2i} \oplus Y_{2j} = (0,0,0,\alpha)$ dengan syarat $\alpha \neq 0$. Untuk pasangan ini, kita dapat memecahkan $Y_{2i+1} \oplus Y_{2j+1}$ yang juga menghasilkan $(0,0,0,\alpha)$. Kita telah mendapatkan dua pasang input yang tepat dan dua pasang output yang tepat. Di antara 2^{95} pasangan input terdapat 2^{48} pasangan input yang tepat. Probabilitas untuk medapat output secara random $(\beta,0,0,0)$ dengan syarat $\beta \neq 0$ adalah 2^{-96} dan dengan menggunaka *4-tupple* dengan dua pasangan output maka kita dapat dengan mudah memisahkan *six rounds cipher* dari permutasi acak.

Serpent

Serpent adalah *symmetric key cypher* blok yang didesign oleh Ross Anderson, Eli Biham, dan Lars Knudsen. Serpent mempunyai ukuran blok sebesar 128 bit dan melakukan 32 kali operasi.

$X_i \leftarrow B_i \oplus K_i$

$Y_i \leftarrow S_i(X_i)$

$B_{i+a} \leftarrow L(Y_i)$ $\quad i = 0,...,30$

$B_{i+a} \leftarrow Y_i \oplus K_{i+1}$ $\quad i = 31$

Keterangan:

B_i = enkripsi ke – i

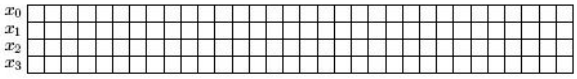
B_0 = plainteks

B_{32} = chiperteks

K_i = subkey

S_i = box ke – i

Serpent menggunakan diagram sebagai berikut sebagai representasinya:

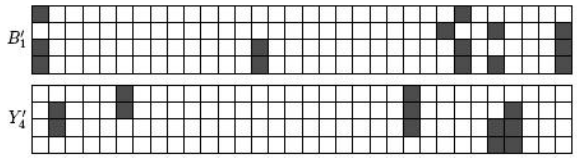


Gambar 4 Diagram Serpent

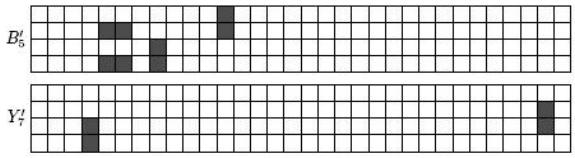
Serpent menggunakan delapan *S-boxes S* di mana i merupakan kelipatan 8 (0,8,16,24,...). Setiap boks menerima empat input bit dan mengeluarkan empat input bit. Input dan output ini kemudian akan berkorespondensi dengan kolom sesudahnya.

Serangan Amplified Boomerang Attack pada Serpent

Kita akan mengambil contoh pada serpent dengan *seven-round variant*. Dengan ketentuan E_0 berkorespondensi dari *round*-1 sampai *round*-4 dan E_1 berkorespondensi dari *round*-5 sampai *round*-7.



Gambar 5 $B'_1 \rightarrow Y'_4$



Gambar 6 $B'_5 \rightarrow Y'_7$

$B'_1 \rightarrow Y'_4$ adalah 4-*round characteristic* melalui E_0 dengan probabilitas 2^{-31} dan $B'_5 \rightarrow Y'_7$ adalah 3-*round characteristic* melalui E_1 dengan probabilitas 2^{-16}. Kita dapat menggabungkan dua karakteristik ini dengan membentuk 7-*round boomerang attack* yang membutuhkan 2^{95} *chosen plaintext queries* dan 2^{95} *adaptive chosen ciphertext queries*. Dengan menggunakan *amplified boomerang attack* kita dapat menyusun *chosen-plaintext only* tetapi membutuhkan 2^{113} *chosen plaintest queries*.

Detail serangan yangg dilakukan adalah sebagai berikut. Pertama, kita meminta 2^{112} pasang plainteks dengan input B'_1. Setelah mengenkripsi dengan setengah bagian pertama dengan E_0, kita akan mendapatkan sekitar 2^{84} pasang yang dapat memenuhi karakteristik $B'_1 \rightarrow Y'_4$. Ada sekitar 2^{161} cara untuk membentuk quartets dengan 2^{81} pasang ini. Dapat diprediksi sekitar 2^{33} quartets (Y_4^0, Y_4^1) dan (Y_4^2, Y_4^3) adalah $Y_4^0 \oplus Y_4^2 = L^{-1}(B'_5)$. Meskipun demikian, karena (Y_4^0, Y_4^1) dan (Y_4^2, Y_4^3) adalah pasangan yang tepat untuk setengah bagian pertama dari *cipher* dan $Y_4^0 \oplus Y_4^1 = Y_4^2 \oplus Y_4^3 = Y'_4$, kita mengetahui bahwa $Y_4^1 \oplus Y_4^3$ harus sama dengan $L^{-1}(B'_5)$. Karena itu, kemunculan acak di antara Y_4^0 dan Y_4^2 telah diperkuat untuk mengikutsertakan Y_4^1 dan Y_4^3. Kemudian, pada input ke E_1, kita dapat memprediksi sekitar 2^{33} *quartets* (B'_5, B'_5) di antara pasangan tersebut. hal ini membuat kita mengetahui bahwa sekitar 2 *quartets* setelah 7-*round* (Y'_7, Y'_7). Kita dapat mengidentifikasi dua *quartets* ini dengan melakukan *hash* terhadap pasangan teks original dengan pasangan *cipertext* yang didapat sebelumnya kemudian diXOR dengan (Y'_7, Y'_7). Pada distribusi acak, probabilitas untuk menemukan kemunculan tunggal pasangan yang berbeda dari perbandingan sebelumnya (Y'_7, Y'_7) adalah sekitar 2^{-33}.

3. KESIMPULAN

Amplified boomerang attack merupakan salah satu tekhnik serangan dalam kriptografi yang merupakan jenis *choosen plainrext attack*. Algoritma serangan ini merupakan pengembangan lebih lanjut dari dua algoritma sebelumnya, yaitu *inside-out attack* dan *boomerang attack*. *Amplified boomerang attack* memang membutuhkan *query* yang lebih banyak dibandingkan pendahulunya, tetapi mempunyi beberapa kelebihan bila dibandingkan dengan *boomerang attack*, misalnya bisa menebak kunci pada akhir *cipher* tanpa harus menambah jumlah *chosen plaintexts* yang diminta, dapat digunakan pada pasangan *k-tuple*, dan dapat melakukan serangan pada bagian spesifik saja tanpa mempengaruhi keseluruhan cipherteks.

Pada kesempatan ini, penulis mencoba menggunakan *amplified boomerang attack* pada dua algoritma yang dianggap cukup baik, yaitu MARS Core dan Serpent. Jika membaca data – data hasil pengetesan, maka dapat diambil kesimpulan bahwa *amplified boomerang attack* ini bisa dibilang efektif dalam membongkar kedua algoritma tersebut.

DAFTAR REFERENSI

http://www.research.ibm.com/security/mars.html
http://www.cl.cam.ac.uk/ftp/users/rja14/serpent.pdf
http://www.users.zetnet.co.uk/hopwood/crypto/scan/cs.html