Teknik Serangan Terhadap Algoritma Hasing MD5

Salman M Ibadurrahman – NIM: 13506106

Program Studi Teknik Informatika, Institut Teknologi Bandung
Jl. Ganesha 10, Bandung
E-Mail: if16106@students.if.itb.ac.id

Abstrak

Makalah ini membahas mengenai teknik serangan yang digunakan untuk melakukan kripanalis terhadap fungsi hash *Message Digest 5* (MD5). Walaupun algortima hash pada MD5 ini merupakan fungsi hash yang irreversible (tidak dapat dikemablikan ke awal), menunjukan kekuatan fungsi ini, tetapi algoritma ini masih terdapat cacatnya. Ditemukan kolisi antara hasil MD5 dengan hasil serangan terhadap fungsi ini.

Kata kunci: fungsi hash, kolisi, diferensial.

1. Pendahuluan

Message digest 5 atau lebih dikenal dengan MD5 merupakan sebuah algortitma hashing yang didesain oleh Ronald Rivest pada tahun 1991. Algoritma ini merupakan perbaikan dari algoritma MD4. Fungsi hasing yang dilakukan menghasilkan 128 bit. Jadi, fungsi MD5 ini melakukan hashing terhadap plain teks seberapapun panjangnya, akan dilakukan fungsi hasing terhadap plain teks yang diberikan, lalu menghasilkan teks hasil hasing yang berukuran fix 128 bit dan hasil hasing ini lebih dikenal dalam bentuk hexadecimal-nya.

2. Algoritma MD5

Secara umum, langkah – langkah yang dilakukan oleh MD5 dalam melakukan fungsi hashing antara lain:

- Penambahan padding bits.
- Penambahan nilai panjang pesan semula.
- Melakukaninisiasi buffer MD.
- Pengolahan pesan pada bentuk blok – blok berukuran 512 bit.

3. Kripanalis Terhadap MD5

Pada tahun 2004, Wang et al melakukan studi tentang kolisi pada MD5. Dan pada tahun 2005 tim mereka mepresentasikan serangan terhadap MD5. Serangan yang dilakukan adalah dengan melakukan 2 blok serangan kolisi dengan kompleksitas sebesar 2^{39}. Hal ini terlihat jauh lebih kecil bila dibandingkan dengan fungsi hasing itu sendiri, yaitu 2^{64}.

Dalam metoda serangan Wang tersebut dijelaskan cara yang efisien dalam mencari 1024 bit string sehingga MD5(M) = MD5(M'). Mereka melakukan traking dalam mencari perbedaan pada komputasi MD5(M) dan MD5(M'). Jika dinotasikan, Qi sebagai keluaran dari putaran ke-i pada MD5 dengan input M, dan Q'_i sebagai keluaran dari putaran ke i pada MD5 dengan input M'. Lalu menghasilkan 128 bit a_i (64 bit pada blok pertama dan 64 bit pada blok yang kedua), dan $0 \leq i \leq 128$ dan jika ditemukan M dimana MD5(M) = MD5(M'), maka $Q_i - Q'_i = a_i$ dan ini berlaku untuk semua putaran, misal pada komputasi $MD5_c(M)$ dan $MD5_c(M')$, maka $Q_i - Q'_i = a_{i+64}$.

Dan Wang menemukan metoda dalam menemukan M. Metoda ini dijelaskan dengan pseudo code di bawah ini.

Algortima find_collision

While collision = false **do**

1. Gunakan bilangan random dan metoda yang deterministic hingga menemukan M dengan memenuhin kondisi Q_i.

2. Lakukan komputasi pada Q_i dan Q'_i apakah hasil diferensial tepat.

3. **if** (diferensial_akhir tepat) **then** collision := true

Else collision ← false

End do

Return M

Sebagai catatan, bahwa pseudo code di atas adalah dilakukan sekali pada tiap blok M. 512 bit pertama, M_0 ditemukan semua diferensial blok pertama, maka M_1 ditemukan.

4. Mencari Diferensial dan Kondisi yang Memenuhi

Derivation (turunan) dan langkah – langkah diferensial yang dilakukan oleh tim Wang tidak dijelaskan. Sehingga saat ini hanya ada asumsi – asumsi dan spekulasi. Antara lain:

- Fungsi Φ pada putaran ketiga adalah bitwise XOR dari input oleh karena itu linear. Sedikit saja perubahan pada input dapat merubah output pada bits yang sama.

- Diferensial adalah 0 pada sedikit langkah terakhir pada putaran kedua dan sedikit langkah pertama pada putaran ketiga.

- Diferensial adalah 2^{31} pada hampir setiap nilai langkah pada putaran ketiga dan keempat.

Dari penjelasan di atas mengatakan bahwa pada putaran ketiga terdapat perbedaan dengan putaran lainnya. Penyebabnya adalah pada sedikit langkah pada putaran kedua memiliki diferensial yang 0 dan bit pertama berbeda pada putaran ketiga pada langkah ke-34.

$$Q'_{34} = Q_{34} + ((H(Q_{33}, Q_{32}, Q_{31}) + Q_{30} + y_{34} + m_{11} + 2_{15}) <<< 16)$$

Yang menghasilkan

$$Q'_{34} = Q_{34} + (2_{15} << 16) = Q_{34} + 2_{31} = Q_{34}[31]$$

Pada langkah ke 35, perbedaan bit yang lain diperlihatkan karena m'14 = m14 + 2^{31}:

$$Q'_{35} \leftarrow Q'_{34} + ((H(Q'_{34}, Q_{33}, Q_{32}) + Q_{31} + y_{35} + m_{14} + 2^{31}) <<< 23)$$

Lalu mengisi Q'_{34} dengan $Q_{34} + 2^{31}$ didapat:

$$Q'_{35} \leftarrow 2^{31} + Q34 + ((H(Q_{34} + 2^{31}, Q_{33}, Q_{32}) + Q_{31} + y_{35} + m_{14} + 2^{31}) <<< 23)$$

$$= 231 + Q_{34} + ((H(Q_{34}, Q_{33}, Q_{32}) + 2^{31} + Q_{31} + y_{35} + m_{14} + 2^{31}) <<< 23)$$

$$= 2^{31} + Q_{34} + ((H(Q_{34}, Q_{33}, Q_{32}) + Q_{31} + y_{35} + m_{14}) <<< 23) = Q_{35} + 2^{31} = Q_{35}[31]$$

Penjelasan di atas membawa kita menyimpulkan bahwa hal – hal di bawah ini digunakan untuk membedakan message dan nilai diferensial tiap langkah.

- Anggap bahwa perbedaan message diperlihatkan pada putaran pertama dan putaran kedua dapat digunakan oleh f ungsi Φ sehingga tidak ada perbedaan pada langkah pertama di putaran ketiga.

- Ambil perbedaan message sehingga perbedaan pada bit ke 31 dapat diambil nilainya.

- Untuk langkah – langkah di atas mencari untuk melakukan minimalisasi kondis pada putara kedua, hindari teknik multi message modifikasi.

Dengan atribut dari fungsi Φ_i ,untuk putaran kesatu dan kedua, langkah – langkah tersebut sangat mungkin.

x	y	z	$\Delta x \Rightarrow \Delta H$	$\Delta y \Rightarrow \Delta H$	$\Delta z \Rightarrow \Delta H$
0	0	0	√	√	√
0	0	1	√	√	√
0	1	0	√	√	√
0	1	1	√	√	√
1	0	0	√	√	√
1	0	1	√	√	√
1	1	0	√	√	√
1	1	1	√	√	√

x	y	z	$\Delta x \Rightarrow \Delta I$	$\Delta y \Rightarrow \Delta I$	$\Delta z \Rightarrow \Delta I$
0	0	0		√	√
0	0	1	√	√	√
0	1	0		√	√
0	1	1	√	√	√
1	0	0		√	
1	0	1	√	√	
1	1	0		√	
1	1	1	√	√	

Gambar 1 Output dari perbedaan H = Φ_i, 32 ≤ i < 48 dan I = Φ_i, 48 ≤ i < 64.

Kondisi pada Qi merupakan kondisi pada bit Qi. Sebagai contoh, untuk blok pertama dekat dengan kolisi MD5 untuk menjamin perbedaan yang diperlukan yaitu 8 signifikan bit terakhir dari Q4 adalah nol. Terdapat 290 kondisi pada tiap putaran untuk blok pertama dan terdapat 310 kondisi pada blok kedua. Bagaimanapun, kondisi terbanyak didapat pada putaran pertama dan putaran kedua. Hal ini sangat penting, karena selama putaran pertama, sebuah kondisi dapat mengubah M sehingga semua kondisi terpenuhi karena pada saat tersebut sebuah kondisi memiliki control pada M dan pada tiap perubahan tidak berpengaruh terhadap komputasi utama.

5. Merancang Algoritma Serangan Cepat

Ide dasar dari serangan ini adalah untuk N = U +[L + F(X, Y, Z) + M+ konstanta]<<<k, kita dapat mengubah nilai dari bit n pada N dengan mencari bit n pada U dan bit n-k pada L, X, Y, Z, M; Kita dapat juga mencari bit – bit kurang dari n pada U dan bit – bit kurang dari n-k pada L, X, Y, Z, M untuk mengubah nilai dari bit n pada n dengan carry.

Dengan melakukan percobaan, kita dapat menemukan bahwa teknik message modifation dasar menyebutkan bahwa kondisi yang memenuhi tidak selalu berhasil karena Carry dan Borrow pada bit 32.

5.1 Algoritma Serangan Untuk blok pertama

(a) pilih bilangan random 32 bit untuk m_0, m_1, m_2, …, m_{15}.

(b) Lakukan perhitungan pada langkah satu dan langkah dua algoritma MD5, ubah m_2, …, m_{13}, m_{14}, m_{15} dengan teknik dasar message modification. Sebagai contoh, m_5 diubah untuk meyakinkan bahwa condisi d_2 pada tabel 1 terpenuhi (Lihat table):

$d_{2,1} = 1, d_{2,2} = a_{2,2}, d_{2,3} = 0, d_{2,4} = a_{2,4}, d_{2,5} = a_{2,5}, d_{2,6} = 0, d_{2,7} = 1, d_{2,8} = 0, d_{2,9} = 0,$
$d_{2,10} = 0, d_{2,11} = 1, d_{2,12} = 1, d_{2,13} = 1, d_{2,14} = 1, d_{2,15} = 0, d_{2,16} = 1, d_{2,17} = 1, d_{2,18} = 1$
$d_{2,19} = 1, d_{2,20} = 1, d_{2,21} = 1, d_{2,22} = 1, d_{2,23} = 1, d_{2,24} = 0, d_{2,25} = a_{2,25}, d_{2,26} = 1,$
$d_{2,27} = a_{2,27}, d_{2,28} = 0, d_{2,29} = a_{2,29}, d_{2,30} = a_{2,30}, d_{2,31} = a_{2,31}, d_{2,32} = 0$

Tabel 1 Kondisi pada d_2 pada diferensial iterasi pertama.

Modifikasi dasar:

$d_2 = \{(d_2)^[(d_2)\&(0xfd8043be)] \wedge [(a2)\&(0x7500001a)]\} \mid (0x027fbc41)$

$m_5 = [(d_2-a_2)>>>12] - d_1 - F(a_2, b_1, c_1) - 0x4787c62a.$

Karena penggunaan teknik pencarian dengan range yang kecil pada langkah selanjutnya, maka ditambahlah 3 kondisi tambahan $c_{4,9} = 0$, $c_{4,21}=0$ dan $c_{4,23}=0$ pada c_4 ketika memodifikasi m_{14}.

(c) Memilih pada 32 bit nilai secara random, tetapi menghasilkan kondisi $a_{5,4}=b_{4,4}$, $a_{5,16} = b_4$, 16, a_5, 18=0, $a_{5,31}=1$ dan $a_{5,32}=b_{4,32}$ terpenuhi, lalu lakukan komputasi pada langkah ke 18:

$\sum_{18} = d_4+G(a_5, b_4, c_4)+m_6+0xc040b340$
$d_5=a_5+\sum_{18}<<<9.$

Jika kondisi $d_{5,18} = 1$, $d_{5,30} = a_{5,30}$ dan $d_{5,32} = a_{5,32}$ tidak semua terpenuhi, maka gunakan teknik pencarian dengan range yang kecil untuk mendapatkan hasil yang tepat. Berdasarkan hasil dari komputasi langkah ke 18 dan kondisi tambahan $c_{4,9} =$

0, $c_{4,21}$=0 dan $c_{4,23}$=0, maka pencarian bit $b_{4,9}$, $b_{4,21}$, $b_{4,23}$ pada b_4, kita dapat mengubah nilai dari bit $d_{5,18}$ = 1, $d_{5,\ 30}$ = $a_{5,\ 30}$ dan $d_{5,\ 32}$ = $a_{5,\ 32}$ terpenuhi, kita perlu melakukan update terhadap nilai m_{15}:

M_{15} = [(b_4-c_4)>>>22]-b_3-F(c_4, d_4, a_4)-0x49b40921.

(d) Lakukan komputasi pada langkah ke 17:

$\sum_{17}$ = a_4+G(b_4, c_4, d_4)+m_1+0xf61e2562e a_5=b4+$\sum_{17}$<<<5.

Jika kondisi $\sum_{17,25}$ ~ $\sum_{17,\ 27}$ tidak semua bernilai 1 dan $a_{5,32}$ = $b_{4,32}$ tidak semua terselesaikan, maka cari bit - bit $d_{1,4}$, $d_{1,5}$ pada d1 dan bit – bit $b_{4,21}$, $b_{4,21}$, $b_{4,21}$, $b_{4,21}$ pada b_4 lalu lakukan update nilai m1, m_{15} (m_1=[(d_1-a_1)>>>12] –dd_0-F(a_1, bb_0, cc_0) -0xe8c7b756, m_{15} = [(b_4-c_4)>>>22]-b_3 –F(c_4, d_4, a_4) -0x49b40821) untuk mendapatkan nilai yang tepat. Lakukan komputasi kembali langkah 17, jika kondisi $a_{5,4}$=$b_{4,4}$, $a_{5,16}$=$b_{4,16}$ dan $a_{5,18}$=0 tidak semua didaptkan, maka cari bit – bit $d_{1,11}$, $d_{1,23}$, dan $d_{1,25}$ pada d_1 dan lakukan update nilai m_1 (m_1 = [(d_1-a_1)>>>12] –dd_0 –F(a_1, bb_0. c_0) -0xe8c7b756) utnuk mendapatkan nilai yang tepat.

(e) Lanjutkan lakukan perhitungan dengan langkah – langkah sisanya, jika kondisi pada tabel 4 tidak memenuhi, langusng saja melompat ke langkah (c). Jika semua kondisi memenuhi, lanjutkan kepada algoritma serangan pada blok yang kedua. Dan pada langkah ini kita mendapatkan output aa_0, bb_0, cc_0, dd_0 untuk dipassing ke algoritma serangan pada blok yang kedua.

Kesimpulan

MD5 walaupun merupakan algoritma yang irreversible, tetapi saat ini ditemukan metode yang dapat menyebabkan kolisi pada hasil hashing md5 tersebut, sehingga message asli dapat diketahui.

Dengan menggunakan range yang kecil dalam teknik melakukan pencarian dan mengabaikan komputasi dalam pengecekan karakteristik, kompleksitas dan probabilitas dalam pencarian message lebih optimum. Hal ini pula dapat diimplementasikan pada serangan terhadap MD4 dan RIPEMD.

Walalupun serangan dapat dilakukan dengan metoda-metoda di atas, pencarian atau kolisi masih memakan waktu yang cukup lama sehingga diperlukan perbaikan – perbaikan dalam metoda di atas. Yang diperlukan adalah pengembangannya, tidak perlu mencari metoda baru. `

Daftar Pustaka

1. Philip Hawkes, Michael Paddon, and Gregory G. Rose. Musings on the Wang et al. MD5 collision. Cryptology ePrint Archive, Report 2004/264, 2004. http://eprint.iacr.org/2004/

2. http://en.wikipedia.org/wiki/MD5

3. Rinaldi Munir, Diktat kuliah Kriptografi IF3058. Institut Teknologi Bandung.