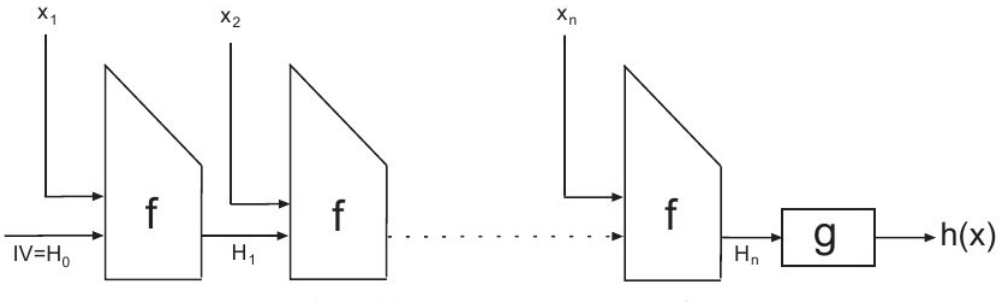
Vybrané útoky proti hašovací funkci MD5

1 Úvod, vymezení

V práci popisuji vybrané útoky proti bezpečnosti hašovací funkce MD5. Nejdříve uvádím zjednodušený algoritmus MD5 a následně rozebírám dva praktické útoky. Nerozebírám teoretické předpoklady hašovacích funkcí.

2 Popis MD5

MD5[1] je algoritmus optimalizovaný pro 32 bitové procesory, vychází z MD4 (oproti MD4 je mírně pomalejší). Převádí řetězec libovolné délky na řetězec o délce 128 bitů. V algoritmu proběhne jako první doplnění zprávy a příprava pomocných registrů (čtyři 32bitová slova A, B, C, D) a následně je zpráva podle Merkle-Damgårdova schématu zpracována po blocích o velikosti 512 bitů (pro každý blok 4 kola o 16 krocích, tj. celkem 64 rund).



Obrázek 1: Merkle-Damgårdovo schéma (struktura) (zdroj: [3, str. 200]); x_i jsou pro MD5 bloky o 512 bitech, H_i řetězce o délce 128 bitů a $h(x)$ výsledný hash.

Doplnění zprávy probíhá probíhá následovně: nejdříve je připojen jeden bit 1 a poté množství 0 tak, aby délka zprávy byla kongruentní s 448 modulo 512 (k doplnění dochází i v případě, že zpráva již podmínku splňuje). Následně je připojena do posledních 64 bitů délka zprávy (v případě délky větší než 2^{64} je použito pouze spodních 64 bitů, neboli délka zprávy $mod 2^{64}$).

Počáteční inicializační vektor – registry (pevně stanoveny ve standardu):

a_0 :	01	23	45	67
b_0 :	89	ab	cd	ef
c_0 :	fe	dc	ba	98
d_0 :	76	54	32	10

Po doplnění proběhne rozdělení zprávy na N po sobě následujících bloků $M_1, M_2, ..., M_N$ délky 512 bitů. Při samotném zpracování projde v případě zprávy o N blocích algoritmus MD5 $N+1$ stavy (mezivýsledky) IHV_i, pro $0 \leq i \leq N$. Každý mezivýsledek IHV_i se skládá ze čtyř 32 bitových slov a_i, b_i, c_i, d_i (pro $i = 0$ viz výše). Mezivýsledky IHV_i jsou pro $i = 1, 2, ...N$ vypočítány kompresní funkcí MD5 ($MD5C$) takto:

$$IHV_i = MD5C(IHV_{i-1}, M_i).$$

[1] Message-Digest algorithm 5

Výsledný hash je poslední mezivýsledek IHV_N, určený jako spojení a_N, b_N, c_N, d_N, převedený zpět z malého endianu.

2.1 Kompresní funkce MD5

Vstup do kompresní funkce $MD5C(IHV, B)$ je složený z mezivýsledku $IHV = (a, b, c, d)$ a 512 bitového bloku B. Kompresní funkce obsahuje 64 rund (0..63, rozdělených do 4 kol po 16 krocích. Každá runda t obsahuje modulární operace, rotaci (směrem vlevo), nelineární funkci f_t a konstanty AC_t, RC_t.

$$AC_t = \lfloor 2^{32} |sin(t+1)| \rfloor, 0 \leq t \leq 64$$

$$(RC_t, RC_{t+1}, RC_{t+2}, RC_{t+3}) = \begin{cases} (7, 12, 17, 22) & \text{pro } t = 0, 4, 8, 12, \\ (5, 9, 14, 20) & \text{pro } t = 16, 20, 24, 28, \\ (4, 11, 16, 23) & \text{pro } t = 32, 36, 40, 44, \\ (6, 10, 15, 21) & \text{pro } t = 48, 52, 56, 60. \end{cases}$$

Nelineární funkce f_t závisí na kole:

$$f_t(X, Y, Z) = \begin{cases} F(X, Y, Z) = (X \wedge Y) \oplus (\overline{X} \wedge Z) & \text{pro } 0 \leq t < 16, \\ G(X, Y, Z) = (Z \wedge X) \oplus (\overline{Z} \wedge Y) & \text{pro } 16 \leq t < 32, \\ H(X, Y, Z) = X \oplus Y \oplus Z & \text{pro } 32 \leq t < 48, \\ I(X, Y, Z) = Y \oplus (X \vee \overline{Z}) & \text{pro } 48 \leq t < 64. \end{cases}$$

Blok B je rozdělen do 16 po sobě následujících 32 bitových slov $m_0, ..., m_{15}$ a rozšířený na 64 slov W_t, pro $0 \leq t \leq 64$, každé o 32 bitech:

$$W_t = \begin{cases} m_t & \text{pro } 0 \leq t < 16, \\ m_{(1+5t) mod 16} & \text{pro } 16 \leq t < 32, \\ m_{(5+3t) mod 16} & \text{pro } 32 \leq t < 48, \\ m_{(7t) mod 16} & \text{pro } 48 \leq t < 64. \end{cases}$$

Pro $t = 0, 1, ..., 63$ uchovává algoritmus kompresní funkce 4 stavová slova $Q_t, Q_{t-1}, Q_{t-2}, Q_{t-3}$. Ta jsou na začátku inicializována jako $Q_0, Q_{-1}, Q_{-2}, Q_{-3} = (b, d, d, a)$ a pro $t = 0, 1, ..., 63$ jsou postupně upravena následujícím způsobem:

$$\begin{aligned} F_t &= f_f(Q_t, Q_{t-1}, Q_{t-2}), \\ T_t &= F_t + Q_{t-3} + AC_t + W_t, \\ R_t &= RL(T_t, RC_t), \\ Q_{t+1} &= Q_t + R_t. \end{aligned}$$

Potom po provedení všech výpočtů jsou výsledná „stavová“ slova přidává k mezivýsledku hašovací funkce a vrácena jako výsledek:

$$MD5C(IHV, B) = (a + Q_{61}, b + Q_{64}, c + Q_{63}, d + Q_{62})$$

Alternativní zápis kompresní funkce (RFC):

```
/* Process each 16-word block. */
For i = 0 to N/16-1 do

  /* Copy block i into X. */
  For j = 0 to 15 do
    Set X[j] to M[i*16+j].
  end /* of loop on j */

  /* Save A as AA, B as BB, C as CC, and D as DD. */
  AA = A
  BB = B
  CC = C
  DD = D

  /* Round 1. */
  /* Let [abcd k s i] denote the operation
       a = b + ((a + F(b,c,d) + X[k] + T[i]) <<< s). */
  /* Do the following 16 operations. */
  [ABCD  0  7  1]  [DABC  1 12  2]  [CDAB  2 17  3]  [BCDA  3 22  4]
  [ABCD  4  7  5]  [DABC  5 12  6]  [CDAB  6 17  7]  [BCDA  7 22  8]
  [ABCD  8  7  9]  [DABC  9 12 10]  [CDAB 10 17 11]  [BCDA 11 22 12]
  [ABCD 12  7 13]  [DABC 13 12 14]  [CDAB 14 17 15]  [BCDA 15 22 16]

  /* Round 2. */
  /* Let [abcd k s i] denote the operation
       a = b + ((a + G(b,c,d) + X[k] + T[i]) <<< s). */
  /* Do the following 16 operations. */
  [ABCD  1  5 17]  [DABC  6  9 18]  [CDAB 11 14 19]  [BCDA  0 20 20]
  [ABCD  5  5 21]  [DABC 10  9 22]  [CDAB 15 14 23]  [BCDA  4 20 24]
  [ABCD  9  5 25]  [DABC 14  9 26]  [CDAB  3 14 27]  [BCDA  8 20 28]
  [ABCD 13  5 29]  [DABC  2  9 30]  [CDAB  7 14 31]  [BCDA 12 20 32]

  /* Round 3. */
  /* Let [abcd k s t] denote the operation
       a = b + ((a + H(b,c,d) + X[k] + T[i]) <<< s). */
  /* Do the following 16 operations. */
  [ABCD  5  4 33]  [DABC  8 11 34]  [CDAB 11 16 35]  [BCDA 14 23 36]
  [ABCD  1  4 37]  [DABC  4 11 38]  [CDAB  7 16 39]  [BCDA 10 23 40]
  [ABCD 13  4 41]  [DABC  0 11 42]  [CDAB  3 16 43]  [BCDA  6 23 44]
  [ABCD  9  4 45]  [DABC 12 11 46]  [CDAB 15 16 47]  [BCDA  2 23 48]

  /* Round 4. */
  /* Let [abcd k s t] denote the operation
       a = b + ((a + I(b,c,d) + X[k] + T[i]) <<< s). */
  /* Do the following 16 operations. */
  [ABCD  0  6 49]  [DABC  7 10 50]  [CDAB 14 15 51]  [BCDA  5 21 52]
  [ABCD 12  6 53]  [DABC  3 10 54]  [CDAB 10 15 55]  [BCDA  1 21 56]
  [ABCD  8  6 57]  [DABC 15 10 58]  [CDAB  6 15 59]  [BCDA 13 21 60]
  [ABCD  4  6 61]  [DABC 11 10 62]  [CDAB  2 15 63]  [BCDA  9 21 64]

  /* Then perform the following additions. (That is increment each
     of the four registers by the value it had before this block
     was started.) */
  A = A + AA
  B = B + BB
  C = C + CC
  D = D + DD

end /* of loop on i */
```

(Zdroj: [9])

3 Útoky (nejen) proti MD5

3.1 Slovníkový útok

Jde o útok, kdy na základě znalosti výsledku hašovací funkce MD5 hledáme jeden z možných původních řetězců; při očekávané malé délce původního řetězce (například méně než 448 bitů) je možné se značnou pravděpodobností předpokládat, že náš nalezený řetězec odpovídá původnímu

a nejde o kolizi. Teoretická složitost útoku je 2^{64}. Typickou úlohou je nalezení původních hesel po získání databáze, ve které je vždy pouze uživatelské jméno a hash.

Pro množiny s malým počtem prvků (například sto nejčastěji používaných hesel, řádově do desítek tisíc záznamů) je nejrychlejší slovníkový útok (útok se slovníkem), tj. prohledávání dvojic předem vypočítaných prvků *hash – původní řetězec*. Takové hledání poskytuje výsledky do 2 vteřin.

Znaky (slova)	Maximální délka	Objem dat (GB), odhad
100 000 slov $\sim$	10 znaků	0.0026
a-z	6	7
a-z, A-Z	6	454
a-z, A-Z, 0-9	6	1281
a-z, A-Z, 0-9	8	5546713
a-z, A-Z, 0-9, 20 speciálních znaků	10	375 711 425 084

Při velkém objemu dat jsou nároky na úložný prostor prohibitivní.

3.2 Rainbow tables

Snižuje nároky slovníkového útoku na prostor pomocí ukládání pouze částí výpočtu ze všech možných výsledků hašovací funkce („time-memory tradeoff“). Původní výpočet:

$$C_0 = S_k(P_0)$$

kde P je otevřený text, C výsledek hašovací (šifrovací) funkce je možné nahradit po zavedení redukční funkce R, která z haše vytváří nový klíč (mezivýsledek) a při procházení všech možných klíčů: $k_i \xrightarrow{S_{k_i}(P_0)} C_i \xrightarrow{R(C_i)} k_{i+1}$ Označením $R(S_k(P_0))$ za $f(k)$:

$$k_i \xrightarrow{f} k_{i+1} \xrightarrow{f} k_{i+1}$$

Problémy postupu: kolize řetězů (začátek jsou různé klíče, klíč nemusí být v tabulce). Některá řešení: více tabulek, používání „distinguished points“ (tj. ukončení řetězu na výsledku, který je vhodným způsobem rozpoznatelný, ne jen po určitém počtu iterací).

	Slovníkový útok	Útok hrubou silou	Rainbow table
Prostor klíčů	23 109	$\sim$ 8 miliard	$\sim$ 8 miliard
Příprava	1.05 vteřiny	96 hodin (odhad)	20 hodin
Rychlost vyhledávání	$<$ 1 vteřina	Dle algoritmu	2.6 vteřiny (maximum)
Objem dat	$\sim$ 947 KB	300GB	$\sim$ 611 MB

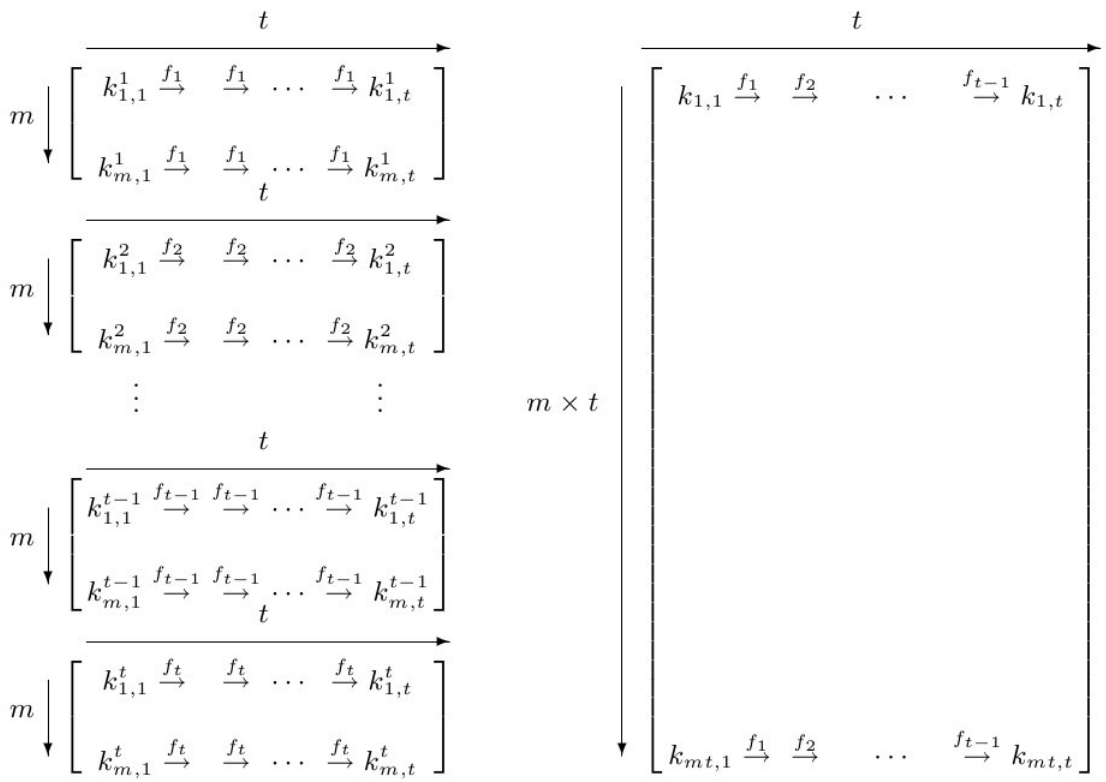
3.3 Hledání kolize

V roce byl představen teoretický útok hledání kolize (Wang et al., „čínský útok“) přes sledování mezivýsledků IHV_i, případně s nalezením dvou zpráv se stejným hašem a začátkem.

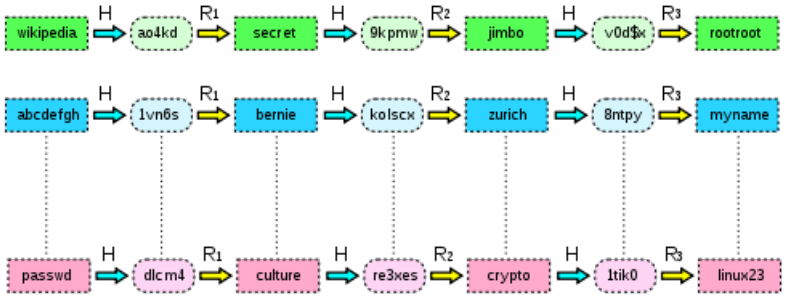
Nejzajímavější současný úspěch je nalezení kolize MD5 pro dva různé X-509 certifikáty. Útok probíhá následujícím způsobem: jsou vytvořeny dvě zprávy (certifikáty), doplněny tak, aby měly stejný počet 512 bitových bloků (viz Merkle-Damgård); v rámci slabin MD5[2]) jsou pak doplněny v některých polích (u [6] v RSA modulu) tak, aby měly stejný hash (útok předpokládá, že certifikáty jsou generované v prakticky stejný čas kvůli náhodným polím v certifikátu). Obdobný útok je možné realizovat se dvěma upravitelnými dokumenty (Postscript nebo „Word“ s obrázky),

[2]Mimo rozsah této práce

Karel Kohout
(karel.kohout@centrum.cz)



Obrázek 2: Rainbow table (vpravo), vlevo klasické tabulky pro t-m (zdroj: [1]).



Obrázek 3: Rainbow table (zdroj: [4]).

kde je na konci dokumentu v rámci zdánlivě nenápadného prvku (čárový kód, firemní logo) upraveno několik pixelů tak, aby zfalšovaný dokument měl stejný haš jako původní (též útok „Nostradamus“ s předpovědí výsledku voleb).

4 Závěr

Hašovací funkce MD5 je prakticky (nejen teoreticky) zranitelná pro většinu svých použití a měla by být nahrazena jinou, standardní a ověřenou funkcí (rodina SHA-2, budoucí SHA-3).

Zdroje]plain Poznámka: všechny odkazy byly mezi 16.5.2010 a 18.5.2010 funkční (vzhledem k délce psaní práce neuvádím data u každého odkazu zvlášť). Platí i pro odkazy v textu.

Reference

[1] Black, J., Cochran, M, A study of the md5 attacks: Insights and improvements
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.88.8007&rep=rep1&type=pdf`

[2] Menezes, Alfred J., Oorschot, Paul C. van, Vanstone, Scott A.: Handbook of applied cryptography, CRC Press, 1996, ISBN 9780849385230

[3] Stinson, Douglas: Cryptography - Theory and practice, CRC Press, 1995, ISBN: 0849385210

[4] Wikipedia, Rainbow table diagram
`http://en.wikipedia.org/wiki/File:Rainbow_table1.svg`

[5] Stevens, M., Lenstra, A., Weger, B. de, Chosen-pre[U+FB01]x Collisions for MD5 and Applications
`homepages.cwi.nl/~stevens/papers/stJOC-SLdW.pdf`

[6] Stevens, M., Lenstra, A., Weger, B. de, Chosen-prefix Collisions for MD5 andColliding X.509 Certi[U+FB01]cates for Di[U+FB00]erent Identities
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.140.3579&rep=rep1&type=pdf`

[7] Stevens, M. On Collisions for MD5 (Master's thesis)
`http://www.win.tue.nl/hashclash/On%20Collisions%20for%20MD5%20-%20M.M.J.%20Stevens.pdf`

[8] Wang, X., Yu, H. How to Break MD5 and Other Hash Functions
`http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.102.6718&rep=rep1&type=pdf`

[9] The MD5 Message-Digest Algorithm, RFC 1321
`http://www.ietf.org/rfc/rfc1321.txt`