SHA-3

L'ubomíra Balková

Úvod do kryptologie

29. dubna 2010

Prolomení hašovacích funkcí

- "masová" kryptografie na nedokazatelných (nedokázaných) principech
- prolomení je přirozená věc
- 2004 - prolomena MD5
- SHA-1 platí do konce 2010
- současný platný standard SHA-2 je 3krát pomalejší

Soutěž o SHA-3

- 2007 - NIST (americký úřad pro standardizaci) soutěž na nový hašovací standard SHA-3
  - požadavky: rychlost (SHA-1 7,5 cyklu procesoru/byte, SHA-2 20 cyklů procesoru/byte), nároky na paměť (stovky bytů)
  - 64 algoritmů od 191 kryptologů (univerzity a elektronické giganty, ale i největší světoví výrobci z oblasti čipů) např. firmy: Microsoft, Sony, RSA, Intel, IBM, MIT, PGP, Hitachi, známá jména: Rivest, Schneier
  - Češi spojeni s 2 kandidáty: EDON-R (Aleš Drápal, Vlastimil Klíma, vlastník a vynálezce Danilo Gligoroski), BMW (vlastník-vynálezce Gligoroski-Klíma)
- červenec 2009 - vybráno 14 kandidátů: BLAKE, BLUE MIDNIGHT WISH, CubeHash, ECHO, Fugue, Grøstl, Hamsi, JH, Keccak, Luffa, Shabal, SHAvite-3, SIMD, Skein
- podzim 2010 - bude vybráno 5 finalistů
- 31.12.2012 - vyhlášení vítěze

14 kandidátů

64 bitový procesor, 256 bitový hašový kód, rychlost v cyklech/byte		
1	Blue Midnight Wish	7.55
2	Skein	7.6
3	Shabal	8.03
4	BLAKE	8.19
5	Keccak	10
6	CubeHash	11
7	SIMD	11
8	Luffa	13.4
9	SHA-256	15.34
10	JH	16.8
11	Grøstl	22.2
12	Hamsi	25
13	SHAvite-3	26.7
14	Fugue	28
15	ECHO	28.5

64 bitový procesor, 512 bitový hašový kód, rychlost v cyklech/byte		
1	Blue Midnight Wish	3.88
2	Skein	6.1
3	Shabal	8.03
4	BLAKE	9.29
5	CubeHash	11
6	SIMD	12
7	SHA-512	12.59
8	JH	16.8
9	Keccak	20
10	Luffa	23.2
11	Hamsi	25
12	Grøstl	30.5
13	SHAvite-3	38.2
14	ECHO	53.5
15	Fugue	56

BMW = Blue Midnight Wish

- vznikla spoluprací Vlastimila Klímy na vylepšení Turbo-SHA Danila Gligoroského a Sveina Knapskoga
- po roce práce (konec 2008) odeslána do soutěže NISTu
- pracovní název nejprve Blue Wish, pak ale autoři zjistili, že jde o registrovanou značku, když se po x-té o půlnoci blížili ke konečné variantě algoritmu, napadlo je Blue Midnight Wish

Nové nápady

- požadavek: větší bezpečnost i rychlost $\Rightarrow$ nutnost nových nápadů
- soustavy rovnic tvořené polynomy o mnoha neznámých (booleovské proměnné) nedovedeme řešit v polynomiálním čase
- ALE! spočíst hodnotu náhodného polynomu 32. stupně s proměnnými $a_0, a_1, \ldots, a_{31}$ a $b_0, b_1, \ldots, b_{31}$, např.

$$a_0 * a_1 * \cdots * a_{31} \oplus a_0 * b_0 * b_1 * b_2 * a_{12} \oplus \cdots \oplus$$

  je náročné na paměť i čas ($\sim$ počtu $*$ a $\oplus$)
- existuje operace, kterou moderní procesory zvládají v 1 taktu (nejrychleji, jak je to možné), a přesto poskytuje 32 polynomů vysokých řádů najednou!

Operace ADD

- jedná se o operaci ADD!
- označme $a_{31} \dots a_1 a_0$ binární zápis a a $b_{31} \dots b_1 b_0$ binární zápis b, $s_{32} \dots s_1 s_0$ binární zápis $s = a + b$ a $c_{32} \dots c_2 c_1$ bity přenosu (carry), pak

$$s = (c_{32}, a_{31} \oplus b_{31} \oplus c_{31}, \dots, a_1 \oplus b_1 \oplus c_1, a_0 \oplus b_0)$$

$$c_1 = a_0 * b_0$$

$$c_2 = a_1 * b_1 \oplus a_1 * c_1 \oplus b_1 * c_1$$

$$c_3 = a_2 * b_2 \oplus a_2 * c_2 \oplus b_2 * c_2$$

$$\dots$$

$$c_{32} = a_{31} * b_{31} \oplus a_{31} * c_{31} \oplus b_{31} * c_{31}$$

Kandidáti na SHA-3

- dosadíme jednotlivé výrazy pro bity přenosu do vyšších bitů:

$$c_1 = a_0 * b_0 \quad \text{1 term řádu 2}$$

$$c_2 = a_1 * b_1 \oplus a_1 * a_0 * b_0 \oplus b_1 * a_0 * b_0 \quad \text{1 term řádu 2 a 2 termy řádu 3}$$

- jedinou operací $a + b$ tak vznikne 32 polynomů, které dohromady obsahují přes 2 miliardy termů řádu 2 - 31

Termy v součtu 32-bitových čísel

term řádu	0	1	2	3	4	5	6	7	8	9	10	11	12	31	Součet
bit carry															
c1			1												1
c2			1	2											3
c3			1	2	4	0									7
c4			1	2	4	8	0	0	0	0	0	0	0	0	15
c5			1	2	4	8	16	0	0	0	0	0	0	0	31
c6			1	2	4	8	16	32	0	0	0	0	0	0	63
c7			1	2	4	8	16	32	64	0	0	0	0	0	127
c8			1	2	4	8	16	32	64	128	0	0	0	0	255
c9			1	2	4	8	16	32	64	128	256	0	0	0	511
c10			1	2	4	8	16	32	64	128	256	512	0	0	1023
c11			1	2	4	8	16	32	64	128	256	512	1024	0	2047
c12			1	2	4	8	16	32	64	128	256	512	1024	0	4095
c13			1	2	4	8	16	32	64	128	256	512	1024	0	8191
c14			1	2	4	8	16	32	64	128	256	512	1024	0	16383
c15			1	2	4	8	16	32	64	128	256	512	1024	0	32767
c16			1	2	4	8	16	32	64	128	256	512	1024	0	65535
c17			1	2	4	8	16	32	64	128	256	512	1024	0	131071
c18			1	2	4	8	16	32	64	128	256	512	1024	0	262143
c19			1	2	4	8	16	32	64	128	256	512	1024	0	524287
c20			1	2	4	8	16	32	64	128	256	512	1024	0	1048575
c21			1	2	4	8	16	32	64	128	256	512	1024	0	2097151
c22			1	2	4	8	16	32	64	128	256	512	1024	0	4194303
c23			1	2	4	8	16	32	64	128	256	512	1024	0	8388607
c24			1	2	4	8	16	32	64	128	256	512	1024	0	16777215
c25			1	2	4	8	16	32	64	128	256	512	1024	0	33554431
c26			1	2	4	8	16	32	64	128	256	512	1024	0	67108863
c27			1	2	4	8	16	32	64	128	256	512	1024	0	134217727
c28			1	2	4	8	16	32	64	128	256	512	1024	0	268435455
c29			1	2	4	8	16	32	64	128	256	512	1024	0	536870911
c30			1	2	4	8	16	32	64	128	256	512	1024	536870912	1073741823
														součet	2147483616

Vlastnosti BMW

- používá jen operace ADD a XOR, operace bitových posunů a cyklických bitových posunů
- podobně vypadají všichni nejrychlejší kandidáti (požadavek na rychlost vedl logicky k využití operace ADD)

Společné prvky SHA-2 a BMW

- iterativní princip a kompresní funkce
- padding = doplnění hašované zprávy na potřebný počet bitů, zarovnání na nejbližší násobek 512 nebo 1024 bitů (podle toho, zda jde o BMW256/BMW512, resp. SHA256/SHA512)

Hašování

1. předzpracování
   - doplň zprávu M jednoznačně definovaným způsobem o délku zprávy v bitech a doplněk
   - rozděl zprávu na celistvý násobek N m-bitových bloků $M_1, M_2, \ldots, M_N$
   - nastav počáteční hodnotu průběžné haše $H_0 := IV$
2. výpočet haše
   - for $i = 1$ to N

$$H_i := f(M_i, H_{i-1})$$

3. závěr

$$H(M) := \text{definovaných } n \text{ bitů z hodnoty } H_N$$

Mouchy SHA-2 podle NISTu

- možnost nalezení multikolizí rychlejší než u náhodného orákula
  - u náhodného orákula složitost nalezení $r = 2^k$ multikolizí $2^{n(r-1)/r}$ operací, u SHA-2 pouze $k2^{n/2}$ (Joux)
- možnost nalezení kolize stejná jako u náhodného orákula ($2^{n/2}$ podle narozeninového paradoxu)
- náchylnost na útok prodloužením zprávy

Dvojnásobná pumpa

- využita částí kandidátů na SHA-3
- navržena k řešení výše uvedených much Lucksem
- jde o zdvojnásobení šířky průběžné haše a výsledná haš je pak polovina průběžné haše
- k nalezení kolizí Jouxovým útokem je třeba $k2^n$ operací
- výpočet ale pak trvá cca. 4krát déle

Pumpa BMW

Algorithm: BLUE MIDNIGHT WISH

Input: Message M of length l bits, and the message digest size n.

Output: A message digest *Hash*, that is n bits long.

1. Preprocessing
   (a) Pad the message M.
   (b) Parse the padded message into N, m-bit message blocks, $M^{(1)}, M^{(2)}, \ldots, M^{(N)}$.
   (c) Set the initial value of the double pipe $H^{(0)}$.
2. Hash computation

   For $i = 1$ to N

   {

   $Q_a^{(i)} = f_0(M^{(i)}, H^{(i-1)});$

   $Q_b^{(i)} = f_1(M^{(i)}, Q_a^{(i)});$

   $H^{(i)} = f_2(M^{(i)}, Q_a^{(i)}, Q_b^{(i)});$

   }
3. *Hash* =Take_n_Least_Significant_Bits$(H^{(N)})$.

Obr.: Dvojnásobná (a lokálně dočasně čtyřnásobná) pumpa u BMW

Kryptografické využití hašovacích funkcí

- jednoznačná identifikace dat (jednoznačná reprezentace vzoru, digitální otisk dat, jednoznačný identifikátor dat, to vše zejména pro digitální podpisy)
- kontrola integrity (kontrola shodnosti velkých souborů dat)
- ukládání a kontrola přihlašovacích hesel
- prokazování autorství
- prokazování znalosti
- autentizace původu dat
- nepadělatelná kontrola integrity
- pseudonáhodné generátory, derivace klíčů

Kryptografické využití hašovacích funkcí

Digitální otisk dat (digital fingerprint) nebo výtah zprávy (message digest)

- z velkých dat vytváří hašovací funkce jejich identifikátor (díky bezkoliznosti nejsme schopni nalézt jinou zprávu se stejnou haší)
- v řadě zemí stojí digitální otisky na úrovni otisků prstů pro identifikaci lidí, proto při digitálním podpisu zprávy stačí podepsat její haš

Kryptografické využití hašovacích funkcí

Ukládání přihlašovacích hesel

- místo hesel se ukládají jejich haše
- přihlašování do systému = výpočet haše a jeho porovnání s uloženou hodnotou
- haše neodhalují hesla, z nichž jsou vypočteny

Kryptografické využití hašovacích funkcí

Klíčovaný hašový autentizační kód HMAC

- hašováním zpracovává nejen zprávu M, ale i tajný klíč K
- použití:
  - nepadělatelné zabezpečení zpráv (útočník nemůže data měnit bez změny HMACu a ten bez tajného klíče nevypočte správně)
  - průkaz znalosti při autentizaci entit (dotazovatel odešle challenge, od prokazovatele obdrží HMAC(challenge, K), útočník z odpovědi nezíská klíč K)
- značení HMAC-SHA-1(M,K)

Kryptografické využití hašovacích funkcí

Pseudonáhodné generátory

- chování jako náhodná orákula, změna 1 bitu na vstupu $\Rightarrow$ změna každého výstupního bitu s pravděpodobností 50%
- PKCS#1 v.2.1(RSA Cryptography Standard) definuje MGF1 (Mask Generation Function)

  $h(seed||0x00000001), h(seed||0x00000002), h(seed||0x00000003), \ldots$

  kde $0x$ hexadecimální vyjádření daného čísla
- standard pro podpisové schéma DSA definuje náhodný generátor

  $$x_0 := K, \; x_{j+1} := h((K + 1 + x_j) \mod 2^m),$$

  kde K je počáteční vstup (klíč) a m délka bloku hašovací funkce h
- PRNG pro nenáhodný seed (např. password)

Důsledky prolomení

Změna v kryptografických algoritmech, které jsou používány pro vytváření elektronického podpisu
Na základě aktuálních poznatků v oblasti kryptografie a dokumentu ETSI TS 102 176-1 V2.0.0 (ALGO Paper) Ministerstvo vnitra stanoví:

- Kvalifikovaní poskytovatelé certifikačních služeb ukončí vydávání kvalifikovaných certifikátů s algoritmem SHA-1 do 31. 12. 2009. Od 1. 1. 2010 budou tito poskytovatelé vydávat kvalifikované certifikáty podporující některý z algoritmů SHA-2.
- Zároveň je od uvedeného data stanovena minimální přípustná délka kryptografického klíče pro algoritmus RSA na 2048 bitů.