Generic Constructions for Iterated Hash Functions

Bart Preneel

COSIC – Kath. Univ. Leuven, Belgium & ABT Crypto

bart.preneel(AT)esat.kuleuven.be

April 2007

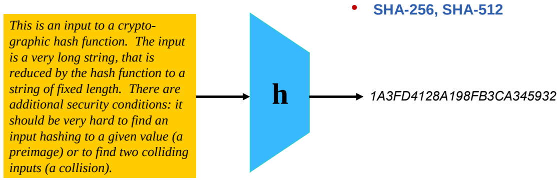
Outline

- definitions
- applications
- generic attacks
- attacks on iterated constructions
- attacks on custom designed hash functions: MD5, SHA, SHA-1
- alternative constructions
- pseudo-randomness
- conclusions

Hash functions

- MDC (manipulation detection code)
- Protect short hash value rather than long text

- (MDC-2)
- (MD5)
- (SHA-1)
- RIPEMD-160
- SHA-256, SHA-512



Hash function flavours

cryptographic hash function

MAC — MDC ← this talk

MDC: OWHF, UOWHF (TCR), CRHF

Informal definitions (1)

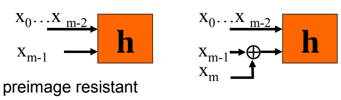
- no secret parameters
- input string *x* of arbitrary length ⇒ output h(x) of fixed bitlength *n*
- computation "easy"
- One Way Hash Function (OWHF)
  - preimage resistance
  - 2nd preimage resistance
- Collision Resistant Hash Function (CRHF): OWHF +
  - collision resistant

Security requirements (n-bit result)

preimage	2nd preimage	collision
? → h → h(x)	x ≠ ? → h, h → h(x) = h(x')	? ≠ ? → h, h → h(x') = h(x')
2^n	2^n	$2^{n/2}$

Informal definitions (2)

- preimage resistant ⇏ 2nd preimage resistant
  - take a preimage resistant hash function; add an input bit b and replace one input bit by the sum modulo 2 of this input bit and b



- 2nd preimage resistant ⇏ preimage resistant
  - if h is OWHF, $\underline{h}$ is 2nd preimage resistant but not preimage resistant:
    $\underline{h}(x) = 0 \,||\, x$ if $|x| \le n$
    $1 \,||\, h(X)$ otherwise
- collision resistant ⇒ 2nd preimage resistant
- [Simon 98] one cannot derive collision resistance from "general" preimage resistance (there exists no black box reduction)

Formal definition: (2nd) preimage resistance

Notation: $\Sigma = \{0,1\}$, $l(n)>n$

A **one-way hash function (OWFH)** H is a function with domain $D=\Sigma^{l(n)}$ and range $R=\Sigma^n$ that satisfies the following conditions:

- **preimage resistance**: let x be selected uniformly in D and let M be an adversary that on input $h(x)$ uses time $\le t$ and outputs $M(h(x)) \in D$. For each adversary M,
  $$Pr_{x \in D} \{ h(M(h(x)))=h(x) \} < \varepsilon$$
  Here the probability is also taken over the random choices of M.
- **2nd preimage resistance**: let x be selected uniformly in $D=\Sigma^{l(n)}$ and let M' be an adversary who on input x uses time $\le t$ and outputs $x' \in D$ with $x' \ne x$. For each adversary M',
  $$Pr_{x \in D} \{ h(M'(x))=h(x) \} < \varepsilon$$
  Here the probability is taken over the random choices of M'.

Formal definitions: collision resistance

A **collision-resistant hash function (CRHF)** $\boldsymbol{H}$ is a function family $\{h_S\}$ with domain $D=\Sigma^{l(n)}$ and range $R=\Sigma^n$ that that satisfies the following conditions:

- the functions h_S are preimage resistant and second preimage resistant)
- collision resistance: let F be a collision string finder that on input $S \in \Sigma^s$ uses time $\le t$ and outputs either "?" or a pair $x, x' \in \Sigma^{l(n)}$ with $x' \ne x$ such that $h_S(x')=h_S(x)$. For each F,
  $$Pr_S \{ F(\boldsymbol{H}) \ne \text{"?"} \} < \varepsilon$$
  Here the probability is also taken over the random choices of F.

Formal definitions - continued

- For collision resistance: considering a **family** of hash functions indexed by a parameter ("key") is essential for formalization (but see Rogaway '06: "formalizing human ignorance")
- For (2nd) preimage resistance, one can choose the challenge (x) and/or the key that selects the function.
- This gives three flavours [Rogaway-Shrimpton'04]
  - random challenge, random key (Pre and Sec)
  - random key, fixed challenge (ePre and eSec everywhere) (eSec=UOWHF)
  - fixed key, random challenge (aPre and aSec - always)
- Complex relationship (see figure on next slide).

Relation between formal definitions
[Rogaway-Shrimpton'04]

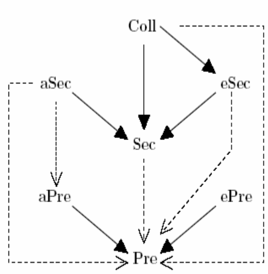


Figure 1: Summary of the relationships among seven notions of hash-function security. Solid arrows represent conventional implications, dotted arrows represent provisional implications (their strength depends on the relative size of the domain and range), and the lack of an arrow represents a separation.

Applications

- digital signatures: OWHF/CRHF, `destroy algebraic structure'
- information authentication: protect authenticity of hash result
- protection of passwords: preimage resistant
- confirmation of knowledge/commitment: OWHF/CRHF
- pseudo-random string generation/key derivation
- micropayments (e.g., micromint)
- construction of MACs, stream ciphers, block ciphers
- (redundancy: hash result appended to data before encryption)

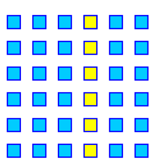
Applications (2)

- Collision resistance is not always necessary
- Other properties are needed:
  - pseudo-randomness if keyed (with secret key)
  - near-collision resistance
  - partial preimage resistance
  - multiplication freeness
  - random oracle property
- how to formalize these requirements and the relation between them?

Brute force (2^{nd}) preimage

- If one can attack 2^t simultaneous targets, the effort to find a single preimage is 2^{n-t}
  - note for t = n/2 this is $2^{n/2}$
- [Hellman80] if one has to find (second) preimages for many targets, one can use a time-memory trade-off with $\Theta(2^n)$ precomputation and storage $\Theta(2^{2n/3})$
  - inversion of one message in time $\Theta(2^{2n/3})$
- [Wiener02] if $\Theta(2^{3n/5})$ targets are attacked, the full cost per (2^{nd}) preimage decreases from $\Theta(2^n)$ to $\Theta(2^{2n/5})$
- answer: randomize hash function
  - salt, spice, “key”: parameter to index family of functions

Birthday paradox for collisions



- How hard is it to find a collision for a hash function with an n-bit result?
- $2^{n/2}$ evaluations of the hash function
- Indeed, the number of pairs of outputs = (1/2) $2^{n/2} \cdot 2^{n/2}$
- conclusion: $n \geq 256$ or more for long-term security

The birthday paradox (2)

- Given a set with S elements
- Choose r elements at random (with replacements) with r « S
- The probability p that there are at least 2 equal elements (a collision) is 1 - exp (- r(r-1)/2S)
- The number of collisions follows a Poisson distribution with $\lambda = r(r-1)/2S$
  - The expected number of collisions is equal to λ
  - The probability to have c collision is $e^{-\lambda} \lambda^c / c!$
- S large, $r = \sqrt{S}$, p = 0.39
- S = 365, r = 23, p = 0.50

The birthday paradox (3) - proof

$$q = 1-p = 1 \cdot \overbrace{((S-1)/S) \cdot ((S-2)/S) \ldots ((S-(r-1))/S)}^{r \text{ terms}}$$

or $q = \Pi_{k=1}^{r-1} (S-k/S)$

$\ln q = \Sigma_{k=1}^{r-1} \ln (1-k/S) \cong \Sigma_{k=1}^{r-1} -k/S = -r(r-1)/2S$

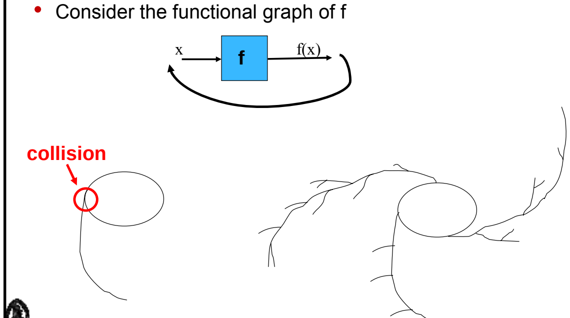
$\boxed{\text{Taylor: if } x \ll 1\text{: } \ln(1-x) \cong x}$

$\boxed{\text{summation: } \Sigma_{k=1}^{r-1} k = r(r-1)/2}$

- hence p = 1 – q = 1 - exp (- r(r-1)/2S)

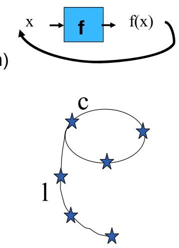
Brute force collision search

- Consider the functional graph of f



Brute force collision search (2)

- Efficient implementation of the birthday attack [Pollard78][Quisquater89]
  - very little memory (cycle finding algorithm)
  - full parallelism [Wiener94]
- Distinguished point (d bits)
  - $\Theta(e2^{n/2} + e\,2^{d+1})$ steps
  - $\Theta(n2^{n/2-d})$ memory
  - with e the cost of one function evaluation
- [Wiener02] full cost: $\Theta(e\,n2^{n/2})$



$l = c = (\pi/8)\, 2^{n/2}$

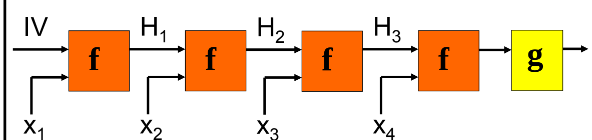
Brute force attacks in practice

- (2^{nd}) preimage search
  - n = 128: 500 M$ for 1 year if one can attack 2^{48} targets in parallel
  - n = 128: 500 B$ for 1 year if one can attack 2^{38} targets in parallel
- parallel collision search
  - n = 128: 100 K$ for 1 month
  - n = 160: 500 M$ for 1 year
  - need 256-bit result for long term security (25 years or more)

Can we get rid of collision resistance?

- collision resistance
  - requires double output lengths
  - requires family of functions for formalization
  - is hard to achieve (e.g., not by black box reduction from one-wayness)
- UOWHF (TCR, eSec) randomize hash function after choosing the message
- [Halevi-Krawczyk05] randomized hashing = RMX mode:
  $H(\, r \,||\, x_1 \oplus r \,||\, x_2 \oplus r \,||\, \dots \,||\, x_t \oplus r \,)$
- needs e-SPR (not met by MD5 and SHA-1 reduced to 53 rounds)
- issues with insider attacks (i.e. attacks by the signer)

Hash function: iterated structure



Split messages into blocks of fixed length and hash them block by block with a compression function f

Efficient and elegant
But many problems…

Security relation between f and h

- Iterating f can degrade its security
  - example: removing x_1 and replacing IV by H_1 leads to a trivial collision or 2^{nd} preimage
- Solution: Merkle-Damgard (MD) strengthening (popular!)
  fix IV, use unambiguous padding and insert length at the end
- [MD89] f is collision resistant $\Rightarrow$ h is collision resistant
- [Lai-Massey92] f is 2^{nd} preimage resistant $\Leftrightarrow$ h is 2^{nd} preimage resistant

?

Construction: relation between f and h (2)

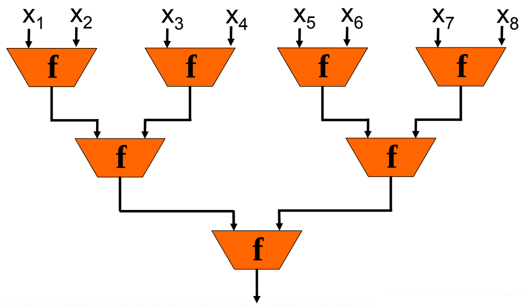
[Damgård-Merkle 89]

Let *f* be a collision resistant function mapping *l* to *n* bits (with *l* > *n*).

- If the padding contains the length of the input string, and if *f* is preimage resistant, the iterated hash function *h* based on *f* will be a CRHF.
- If an unambiguous padding rule is used, the following construction will yield a CRHF (l-n>1):
  $H_1 = f(H_0 \,||\, 0 \,||\, x_1)$
  $H_i = f(H_{i-1} \,||\, 1 \,||\, x_i)\quad i=2,3,\dots t.$

Comment: tree structure

already suggested by Damgård in 1989; further work by Sarkar et al.



Construction: relation between f and h (3)

[Lai-Massey'92]

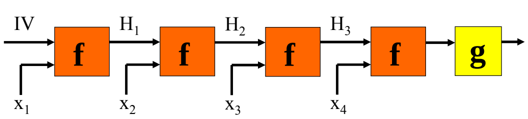
Assume that the padding contains the length of the input string, and that the message x (without padding) contains at least two blocks.

Then finding a second preimage for h with a fixed IV requires 2^n operations **iff** finding a second preimage for f with arbitrarily chosen H_{i-1} requires 2^n operations.

- this theorem is not quite right (see below)
- very few hash functions have a strong compression function
- very few hash functions are designed based on a strong compression function in the sense that they treat x_i and H_{i-1} in the *same way*.

Security relation between f and h (4)

- MD does **not** work for UOWHF [BR97]
- MD with envelope method (prepend and append secret key) works for pseudo-randomness/MAC [BCK96]
  - but there are some problems and HMAC is a better construction
- MD needs output transformation for random oracle properties [Coron+05]
  - if one knows $h(x)$, easy to compute $h(x \,||\, y)$ without knowing x

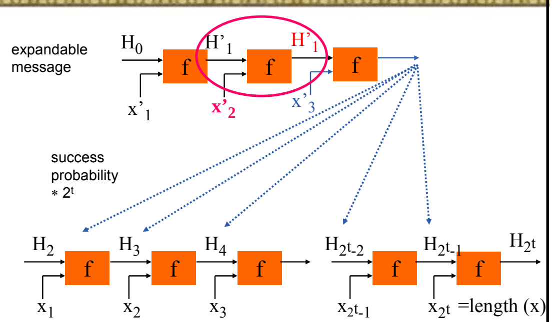


Defeating MD for 2nd preimages

[Dean-Felten-Hu'99] and [Kelsey-Schneier'05]

- Known since Merkle: if one hashes **2^t messages**, the average effort to find a second preimage for one of them is 2^{n-t}.
- **New**: if one hashes **2^t message blocks** with an iterated hash function, the effort to find a second preimage is only $t\,2^{n/2+1} + 2^{n-t+1}$.
- Idea: create expandable message using fixed points
  - Finding fixed points can be easy (e.g., Davies-Meyer).
- find 2nd preimage that hits any of the 2^t chaining values in the calculation
- stretch the expandable message to match the length (and thus the length field)
- But still very long messages for attack to be meaningful
  - n=128, t=32, complexity reduced from 2^{128} to 2^{97}, length is 256 Gigabyte

Defeating MD for 2nd preimages (2)

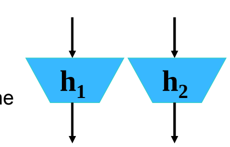


$h(x'_1 \,||\, x'_2 \,||\, x'_2 \,||\, x'_2 \,||\, x'_2 \,||\, x'_3 \,||\ldots||\, x_{2t-1} \,||\, x_{2t}) = h(x_1 \,||\, x_2 \,||\, x_3 \,||\ldots||\, x_{2t-1} \,||\, x_{2t})$

How (NOT) to strengthen a hash function?

[Joux '04]

- Answer: concatenation
- h_1 (n1-bit result) and h_2 (n2-bit result)
- Intuition: the strength of g against collision/(2nd) preimage attacks is the product of the strength of h_1 and h_2
  - if both are "independent"
- But….

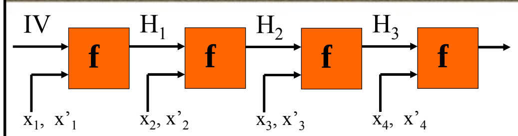


$g(x) = h_1(x) \,||\, h_2(x)$

Multi-collisions [Joux '04]

- Consider h_1 (n1-bit result) and h_2 (n2-bit result), with $n1 \geq n2$.
- The concatenation of two iterated hash functions ($g(x)= h_1(x) \| h_2(x)$) is as most as strong as the strongest of the two (even if both are independent).
- Cost of collision attack against g at most
  $n1 \, . \, 2^{n2/2} + 2^{n1/2} \ll 2^{(n1 + n2)/2}$
- Cost of (2nd) preimage attack against g at most
  $n1 \, . \, 2^{n2/2} + 2^{n1} + 2^{n2} \ll 2^{n1 + n2}$
- If either of the functions is weak, the attacks may work better.
- Main observation: finding multiple collisions for an iterated hash function is not much harder than finding a single collision.

Multi-collisions (2) [Joux '04]



- For IV: collision for block 1: x_1, x'_1
- For H_1: collision for block 2: x_2, x'_2
- For H_2: collision for block 3: x_3, x'_3
- For H_3: collision for block 4: x_4, x'_4
- Now $h(x_1||x_2||x_3||x_4) = h(x'_1||x_2||x_3||x_4) = h(x'_1||x'_2||x_3||x_4) = \ldots = h(x'_1||x'_2||x'_3||x'_4)$ **a 16-fold collision**

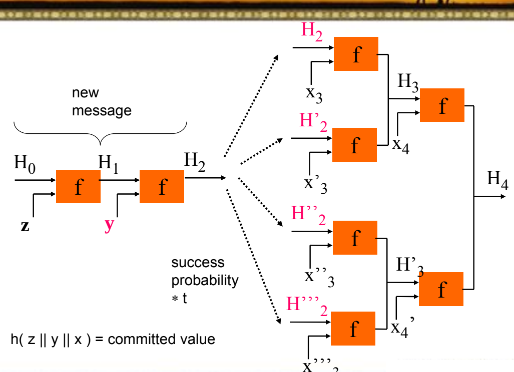
Other issues with iteration: herding

- Herding attack [Kelsey,Kohno'06]
  - reduces security of commitment using a hash function
  - example (n=128, t=42): with a storage of 100 Terabyte and a precomputation of 2^{86} steps, a 128-bit commitment computed using an iterated hash function can be spoofed with effort 2^{86} steps

Herding attack (2)

- protocol: publish *h(x)*, reveal *x* at later date
- find second preimage $x' = z \| y \| x$ with *z* and *y* selected in 2020
- approach: generate collision tree (diamond structure) of 2^t values H_{i-1} and x_i hashing to the same value (cost $2 \, . \, 2^{t/2} \, . \, 2^{n/2}$)
- z = result of all Australia cricket games between 2010 and 2020
- try in 2020 random strings y until $h(z \| y) = H_{j-1}$ for some *j* (cost 2^{n-t})
- then $h(z \| y \| x_j) = h(x)$, so you can claim that you "knew" z in 2006

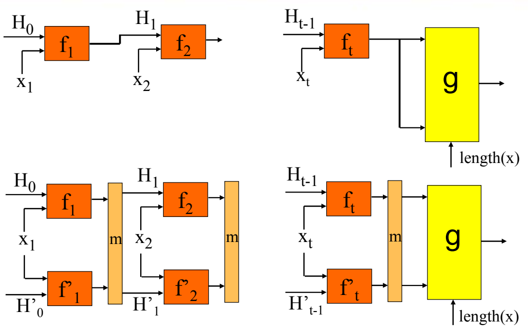
Herding attack (3)



Improving MD iteration

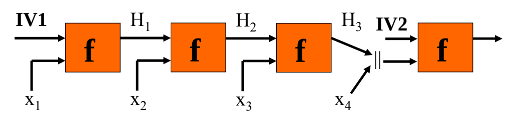
- add salting (family of functions, randomization)
- add a strong output transformation g (which includes total length and salt)
- preclude fix points: counter $f \rightarrow f_i$ (Biham) or dithering (Rivest)
- multi-collisions, herding: avoid breakdown at $2^{n/2}$ with larger internal memory (e.g., RIPEMD, [Lucks05])
- rely on principles of block cipher design, but with larger security margins
- be careful when combining smaller building blocks (à la MDC-2/MDC-4)
- can we build in parallelism and incrementality in an elegant way?

Some ideas



Many more ideas….

- [Biham-Dunkelman 06] Haifa: bit counter and salt input to f
- [Bellare-Ristenpart 06] EMD transform (envelope MD): preserves CR, PRF, RO



- [Andreeva+06] analysis of preservation of CR, (e/a/-)PR, (e/a/-) SPR, (RO, PRF)

Hash function constructions

- many (50+) broken schemes:
  - Rabin, Jueneman,X.509 Annex D, FFT-hash I and II, N-hash, Snefru, MD2, …
- fast schemes for 32-bit machines:
  - most popular designs: MD4 and MD5
  - US government (NIST): SHA (aka SHA-0) and SHA-1
  - Europe: RIPEMD-160
- the next generation: SHA-256, SHA-512, Whirlpool,…
- block cipher based
- based on algebraic constructions

MD4

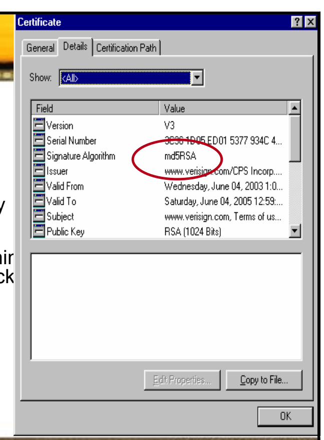
- designed by Rivest in 1990
- 3 rounds
- collisions for 2 rounds [Merkle'90, denBoerBosselaers'91]
- collisions for full MD4 in 2^{20} steps [Dobbertin'96]
- (second) preimage for 2 rounds [Dobbertin'97]
- collisions for full MD4 **by hand** [Wang+'04]
- practical preimage attack for 1 in 2^{56} messages [Wang+'05]
- abandoned since 1993

MD5

- designed by Rivest in 1991
- 4 rounds
- "collisions" for compression function f [denBoerBosselaers93] - ΔIV
- real collisions for compression function f [Dobbertin96] – wrong IV
- **real collisions in 2^{39} steps [Wang+'04]**
  … now in minutes (2^{30})!!

MD5

- Advice (RIPE since 1992, RSA since 1996): stop using MD5
- Largely ignored by industry (click on a cert...)
- Collisions for MD5 are within range of a brute force attack anyway (2^{64})
- But now in minutes…



SHA-1

- SHA designed by NIST (NSA) in '93
- 5 rounds
- redesign after 2 years ('95) to SHA-1

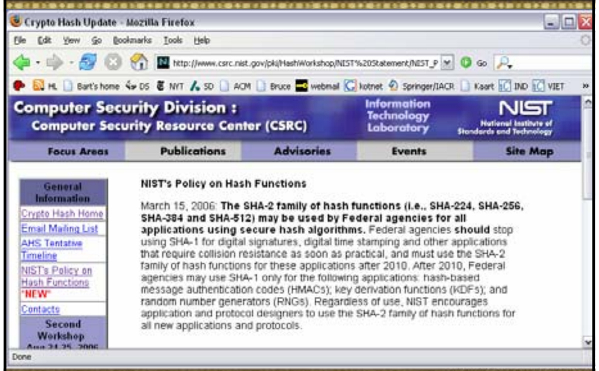
- Collisions for SHA-0 in 2^{51} [Joux+'04]
- Collisions for SHA-0 in 2^{39} [Wang+'05]

- **Collisions for SHA-1 in 2^{63} [Wang+'05]**
  (full details have not been independently verified)

SHA-1 (continued)

- De Cannière-Rechberger 06:
  - automated search for characteristics
  - collision for 64 out of 80 rounds in 2^{35} – highly structured

- Jun Yajima, Yu Sasaki, Teruyoshi Iwasaki, Yusuke Naito, Takeshi Shimoyama, Noboru Kunihiro, Kazuo Ohta (rump Crypto '06)
- Hawkes-Paddon-Rose
- Sugita-Kawazoe-Imai – Gröbner basis (no improvement)


Crypto Hash Update - Mozilla Firefox

Computer Security Division : Computer Security Resource Center (CSRC)

Focus Areas | Publications | Advisories | Events | Site Map

General Information: Crypto Hash Home, Email Mailing List, AHS Tentative Timeline, NIST's Policy on Hash Functions "NEW", Contacts

NIST's Policy on Hash Functions

March 15, 2006: **The SHA-2 family of hash functions (i.e., SHA-224, SHA-256, SHA-384 and SHA-512) may be used by Federal agencies for all applications using secure hash algorithms.** Federal agencies **should** stop using SHA-1 for digital signatures, digital time stamping and other applications that require collision resistance as soon as practical, and must use the SHA-2 family of hash functions for these applications after 2010. After 2010, Federal agencies may use SHA-1 only for the following applications: hash-based message authentication codes (HMACs); key derivation functions (KDFs); and random number generators (RNGs). Regardless of use, NIST encourages application and protocol designers to use the SHA-2 family of hash functions for all new applications and protocols.

From: "Cryptography Simplified in Microsoft .NET" Paul D. Sheriff (PDSA.com) [Nov. 2003]

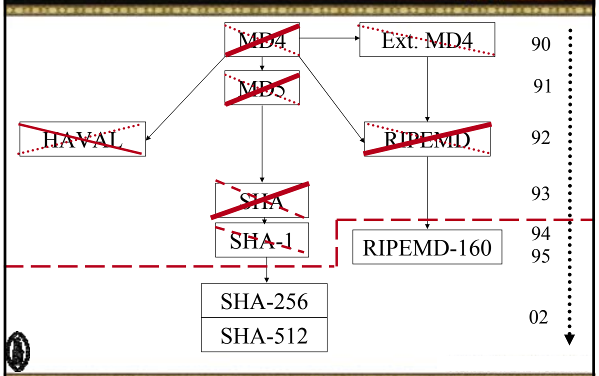
How to Choose an Algorithm

msdn

- For example, SHA1 uses a 160-bit encryption key, whereas MD5 uses a 128-bit encryption key; thus, SHA1 is more secure than MD5 and thus is a much harder hash to break.
- Another point to consider about hashing algorithms is whether or not there are practical or theoretical possibilities of collisions. Collisions are bad since two different words could produce the same hash. SHA1, for example, has no practical or theoretical possibilities of collision. **MD5 has the possibility of theoretical collisions, but no practical possibilities.** So choosing an algorithm comes down to the level of security you need.

In April 2007 this information is still available on MSDN

MDx-type hash function history



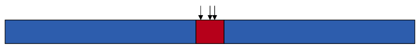
MDx-type cryptanalysis

- **Serious flaws in MD4 and MD5** [RIPE '91-'92]
- **SHA replaced by SHA-1** [NSA '94]
- **Collisions for MD4, problem in ext.-MD4** [Dobbertin '96]
- **More problems of MD5 and RIPEMD** [Dobbertin '96]

- **Collisions for Haval** [Biryukov, Van Rompay, Preneel '02]
- **Collisions for SHA-0** [Joux '04]
- **Collisions for MD4 (by hand), MD5, and RIPEMD** [Wang, Feng, Lai, Yu '04]
- **Attack on 53 out of 80 rounds of SHA-1** [Oswald-Rijmen'04 and Biham-Chen] '04]
- **2^{39} attack on SHA-0** [Wang,Yu,Yin '05]
- **2^{69} attack on SHA-1** [Wang, Yin, Yu '05]
- **2^{63} attack on SHA-1** [Wang, Yao, Yao '05]

Impact of collisions (1)

- collisions for MD5, SHA-0, SHA-1
  - two messages differ in a few bits in 1 to 3 512-bit input blocks
  - limited control over message bits in these blocks
  - but arbitrary choice of bits before and after them



- what is achievable?
  - 2 colliding executables
  - 2 colliding postscript/gif/… documents [Lucks, Daum '05]
  - 2 colliding RSA public keys – thus with colliding X.509 certificates [Lenstra, Wang, de Weger '04]
  - **2 arbitrary colliding files (no constraints) for 100K$**

Impact of collisions (2)

- digital signatures: only an issue if for non-repudiation
- none for signatures computed before attacks were public (1 August 2004)
- none for certificates if public keys are generated at random in a controlled environment
- substantial for signatures after 1 August 2005 (cf. traffic tickets in Australia)

And (2nd) preimages?

- security degrades with number of applications
- for large messages even with the number of blocks (cf. supra)
- specific attacks exist for MD2/MD4
- For MD5/SHA-1: not a threat for current applications

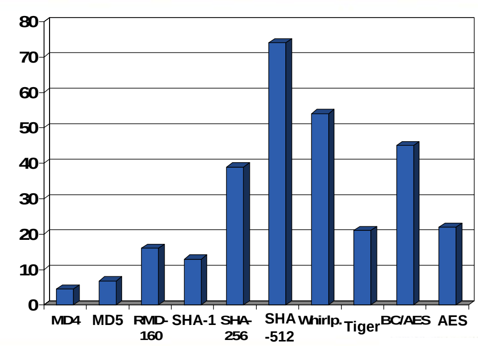
Fixes/Alternatives (1)

- RIPEMD-160 seems more secure than SHA-1 ☺
- message precoding
- small patches to SHA-1
- use more recent standards (slower on 32-bit machines)
  - SHA-256, SHA-512
  - Whirlpool
- block cipher based schemes:
  - well studied
  - due to key schedule for every encryption at least 3-4 times slower than AES encryption

Fixes/Alternatives (2)

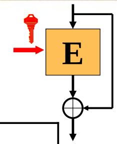
- number theoretic schemes
  - secure but very slow (1 multiplication per bit)
  - speedup by [Contini,Lenstra,Steinfeld 05] VSH
    - still 20 times slower than SHA-1
    - only collision resistance; some other weaknesses
  - topic for further research (lattices, matrices)
- use older schemes: Tiger, Snefru with more rounds, block cipher based schemes (slow)
- start from scratch?

Performance of hash functions
(cycles/byte) Pentium III



Hash function: pseudorandom function (1)

- MDx are based on a block cipher with a feedforward: where to put the key?
- if keyed to the message input: related key boomerang distinguisher attacks apply [Kim+06]



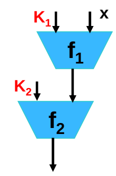
	Rounds of attack	Data complexity
Haval-4	96	$2^{11.6}$ RK-CP + 2^{6} RK-ACC
MD4	48	2^{6} RK-CP + 2^{6} RK-ACC
MD5	64	$2^{13.6}$ RK-CP + $2^{11.6}$ RK-ACC
SHA-1	59 of 80	$2^{70.3}$ RK-CP + $2^{68.3}$ RK-ACC

many hash functions are based on pretty weak block ciphers

Hash function: pseudorandom function (2)

- HMAC keys through the IV (plaintext) [Kim+06]
  - collisions for MD5 invalidate current security proof of HMAC-MD5
  - new attacks on reduced version of HMAC-MD5 and HMAC-SHA-1

	Rounds in f2	Rounds in f1	Data complexity
Haval-4	128	102 of 128	2^{254} CP
MD4	48	48	2^{74} CP
MD5	64	33 of 64	$2^{126.1}$ CP
SHA	80	80	2^{109} CP
SHA-1	80	43 of 80	$2^{154.9}$ CP



no problem yet for most widely used schemes

Hash function: pseudorandom function (3)

- Recent improvements: HMAC/NMAC attacks
  - [Yin-Contini06] better attack on HMAC-MD4 and key recovery attack on NMAC-MD5 (Asiacrypt 2006)
  - [Rechberger-Rijmen06] eprint.iacr.org and Financial Crypto

	Rounds in f2	Rounds in f1	Data complexity
SHA-1	80	53 of 80	$2^{98.5}$ CP

  - Further improvements announced [Biham-Yin]

Hash functions: conclusions

- hash functions such as SHA-1 would have needed 128-160 rounds instead of 80
- recent attacks are not dramatic for all applications, but they form a clear warning: upgrade asap
- limited understanding (theory and practice)
- use weaker security assumptions if possible (UOWHF??)
- research on new and more robust designs with extra features

Hash functions: further reading

- ECRYPT workshops in May 2007 and June 2005 + statement on hash functions at http://www.ecrypt.eu.org
- NIST workshop October 31-November 1, 2005 and August 24-25, 2006 http://www.csrc.nist.gov/pki/HashWorkshop/index.html
- The IACR eprint server http://eprint.iacr.org
- My 1993 PhD thesis http://homes.esat.kuleuven.be/~preneel
- Overview paper from 1998 (LNCS 1528) http://www.cosic.esat.kuleuven.be/publications/article-246.pdf

Thank you for your attention