Demise of MD5 and SHA-1

Designing the New Hash

Stanisław Paweł Radziszowski
Department of Computer Science
Rochester Institute of Technology

August 2008

Abstract

A hash function $H : \{0,1\}^* \rightarrow \{0,1\}^m$ produces an m-bit digest of an arbitrary message, file, or even an entire file system. Typically, one wants hash functions to be easy to compute, but also infeasible to invert or to find collisions (pairs of inputs which hash to the same value). Hash functions are fundamental cryptographic primitives, and they are used extensively in authentication, preserving data integrity, digital signatures, and many other security applications. The two most widely used hash functions are MD5 (Message Digest, $m = 128$) and SHA-1 (Secure Hash Algorithm, $m = 160$), the latter supported by the US government as a standard FIPS-180-2. The collisions for MD5 were found four years ago, and by now they can be produced quickly by software available on the Net. The SHA-1 algorithm seems also to be in trouble (and other algorithms in the SHA family, with $m = 256, 384, 512$, might follow). No collisions for SHA-1 have been found so far, but attacks much better than the simple birthday attack approach have been designed. Breaking SHA-1 soon is a likely possibility.

On January 23, 2007, NIST (National Institute of Standards and Technology) announced an initiative to design a new hash for this century, the Advanced Hash Standard (AHS), likely to be dubbed SHA-3. The competition is open, submissions are due October 31, 2008, and it is planned to conclude in 2012.

`http://www.csrc.nist.gov/pki/HashWorkshop/timeline.html`

These developments are quite similar to the recent history of symmetric block ciphers - breaking of the DES (Data Encryption Standard) and an emergence of the AES (Advanced Encryption Standard) in 2001 as the winner of a multiyear NIST competition.

This talk gives the background on hash function design, outlines the attacks on MD5 and SHA-1, and overviews the scenario of what the teams submitting new designs for the AHS will consider.

Hash - simple, powerful idea

anything
(email, program, document, movie, file system ...)

$$x = y$$

$$\Updownarrow^*$$

$$H(x) = H(y)$$

256 bits
(32 bytes, like this "napisze do ciebie z dalekiej pod" ... no more)

Hashes in Practice

Applications of (cryptographic) hashes

- hash then sign
- time-stamping
- data authentication
- checksumming
- PGP email
- shadow passwords
- networking: SSL, SSH, VPN
- signatures: DSA, DSS (FIPS 186)
- MACs, HMAC (FIPS 198)
- PRNG's, diffusers
- stream ciphers

The Problem

Design a (cryptographic) *hash* function $H : \{0,1\}^* \to \{0,1\}^m$ such that:

- H is *preimage resistant*, i.e. given z, it is infeasible to find any x such that $H(x) = z$
- H is *collision resistant*, i.e. it is infeasible to find any pair x and y such that $H(x) = H(y)$
- H is *resistant* to *second preimage-*, *zero preimage-* ($H^{-1}(0^m)$), *length extension-*, and other attacks.
- H is fast to compute, uses small memory
- H can operate in the streaming mode

Very LARGE bound on input length can be given, pick m as small as possible but still guaranteeing resistance properties

Merkle-Damgård iterated hash

Notation

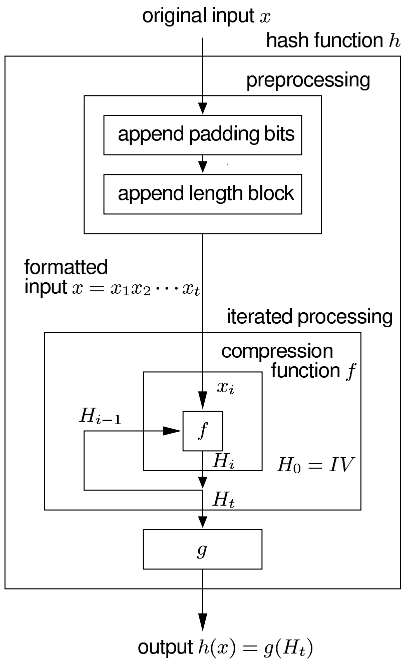
$x \in \{0,1\}^*$ - input message
blocks $m_i (= x_i)$ all of length $|m_i| = b$
$M(x) = m_1 m_2 \cdots m_t$ - formatted input
m_t - padded, includes as tail $|x|$ in binary

IV - initialization vector
H_i - chaining variables
g - postprocessing function
$compress$ - a "kind" of OWF

$H(x)$

$H_0 = IV$;
for $i = 1, 2, \ldots, t$ do
  $H_i = compress(H_{i-1} \| m_i)$;
return $H(x) = g(H_t)$;

Merkle-Damgård iterated hash



(CRC Handbook of Applied Cryptography, [9])

Merkle-Damgård iterated hash

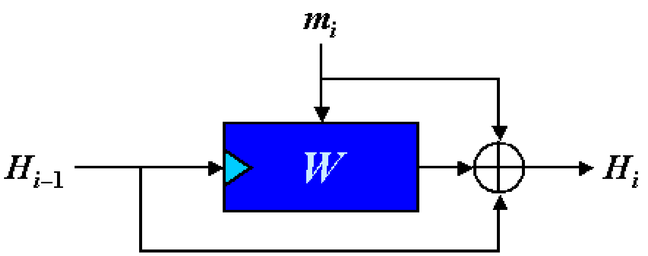
Compression in r simpler rounds

Each m_i is "unfolded" into message schedule m_{ij}, $1 \leq j \leq r$

Each round "absorbs" one m_{ij}

Compression from cipher

Using block cipher W to obtain compression function of the hash



Miyaguchi-Preneel compression, Whirlpool [10]

Compression in iterated MD hash

designed specially for hash (Klima 2007)

- all open vs. secret key in ciphers
- fixing key makes a permutation from ciphers, no need for this in hashes
- can get better performance

from block cipher (Biham 2005)

- known much better than hashes
- there is no evidence that cipher design must lead to worse performance
- easily foil differential attacks
- no more multi-block attacks
- many rounds in hashes hide weaknesses, better use less but stronger rounds
- SHA-2 is just more of the same

Theory needs your help

Theorem
(most of the time - in various scenarios)
Resistant compression implies resistant hash.
Resistant hash implies resistant compression.

Problem
Find a way to study collision resistant compression using complexity theory.
(more than in CRC Handbook 9.8.2)

Characterize more formally:
"This n-to-m-bit compression needs essentially $2^{m/2}$ tests to find a collision and essentially 2^m effort to find any preimage."

People do it normally in random oracle model in probabilistic combinatorics language.

Hashes in Practice

The two most used hash functions
both of Merkle-Damgård type

- **MD5**, Rivest 1992

  128 bit hash, 512 bit blocks
  iterating 64-round compression c_{MD5}

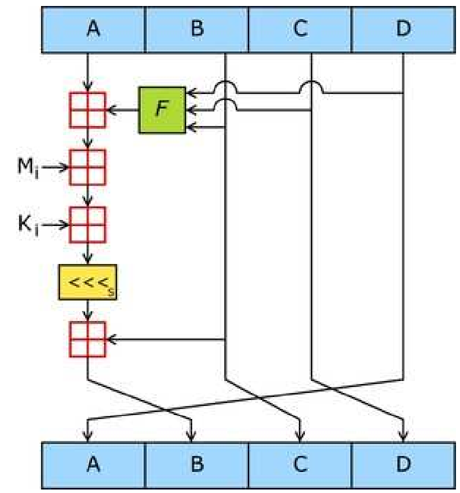
  $$c_{MD5} : \{0,1\}^{640} \rightarrow \{0,1\}^{128}$$

- **SHA-1**, NSA/NIST 1995,
  created mainly for use in DSA

  160 bit hash, 512 bit blocks
  iterating 80-round compression c_{SHA-1}

  $$c_{SHA-1} : \{0,1\}^{672} \rightarrow \{0,1\}^{160}$$

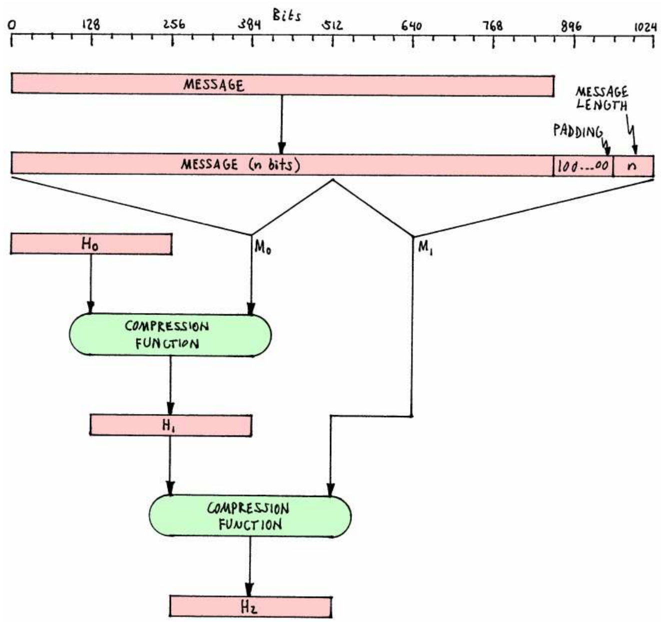
MD5



MD5 round structure
(Wikipedia)

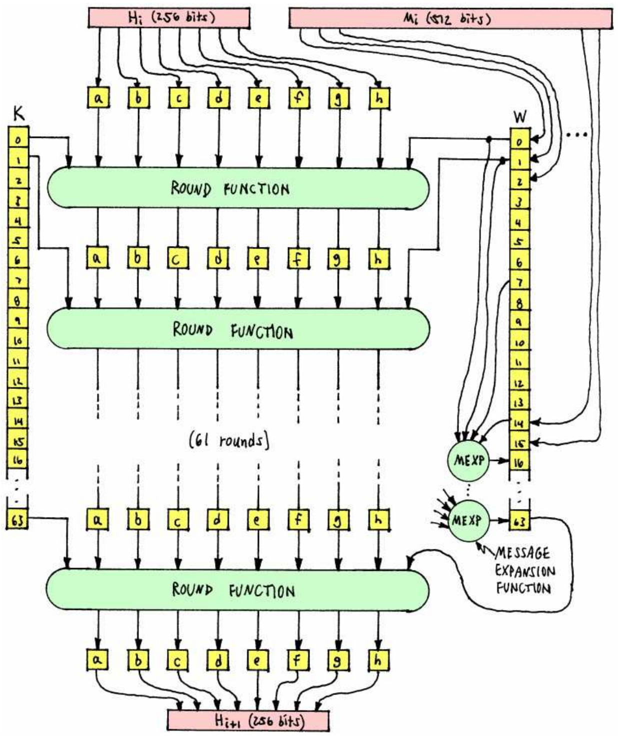
each unit is a 32-bit word

SHA-1



basic SHA-x structure
(fig. Alan Kaminsky, RIT, 2004)

SHA-2

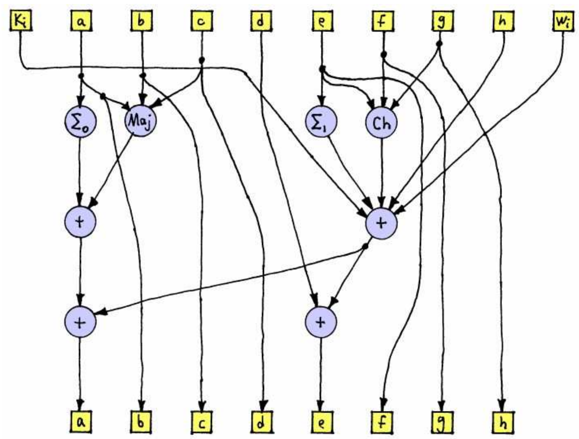


structure of SHA-2 compression (fig. Alan Kaminsky)

SHA-2

FIPS 180-2, 2002
Modes for 224, 256, 384 and 512 bits

each unit is a 32-bit word



one round of SHA-256 compression
(fig. Alan Kaminsky)

Brief History (SHA-family biased)

- 1990 - **MD4**, Rivest, $m = 128$
- 1992 - **MD5**, Rivest, modified MD4
- 1993 - **SHA-0**, NIST, MD-like design
- 1995 - **SHA-1**, FIPS-180-1 $m = 160$
- 2002 - **SHA-2** family, NIST, FIPS-180-2 for $m = 256$, 384 and 512 bit digests

Brief History, contd.

- 2003 - **Whirlpool**, Rijmen-Barreto, AES-like cipher W inside, $m = 512$
- 2004-2006 Wang, Yu, Yin, et al. collision attacks on MD5 and SHA-1
- 2007 - NIST calls for new designs
- 2012 - **AHS/SHA-3** recommended to users

All above hashes (so far) follow Merkle-Damgård template.

Birthday Attack

Counting fishes in a lake, Schnabel (1938)

Theorem (Birthday Paradox)
Random sampling of q elements of the domain of size M will produce at least one collision with probability ϵ if

$$q \approx \sqrt{2M \ln \frac{1}{1-\epsilon}}$$

$q \approx 1.17\sqrt{M}$ for $\epsilon = 1/2$

(M=365, q=23)
Among 23 random people at least two of them have the same birthday with probability at least 1/2.

Generic Attack

Sheer power of computing

- 1998, effort 2^{56}, **DES**↓
- 2007, effort 2^{64} is possible
  $M = 2^{128}$, **MD5**↓
- 2020, effort 2^{80} may be feasible
  $M = 2^{160}$, **SHA-1**↓
- effort 2^{112}, won't be feasible for long

Conclusion. Requiring $m \geq 224$ for **AHS** seems reasonable (224 is the smallest multiple of 32 which prevents birthday attack well).

Preimage attacks are much more difficult, MD5 and SHA-1 are still strong.

Chinese attacks on MD5/SHA-1

Wang, Yu, Yin (1995 - 2004 - 2006)
Probabilistic differential cryptanalysis found collisions in **MD5** and other hashes.

- track simultaneosuly bitwise
  XOR and (mod 2^{32}) differences
- special difference bits in special rounds propagate with probability 1
- good differential paths
- multi-block manipulation
- heuristic approximation

Collisions for full 80 rounds CAN be found (still not done) with

$$2^{80} \to 2^{69} \to 2^{63}$$

SHA-1 computations.

Vast experience and intuition were needed to develop this approach by hand.

Attacks on MD5/SHA-x (1996-2007)

- Collisions for **MD5** in seconds
- Collisions for **SHA-1** likely soon
- **SHA-2** not (yet) threatened
- Preimages almost hopeless

- Several authors (e.g. Black+ 2006, Klima 2007) correct, experiment with and improve on hard to read Wang+ papers.
- No differential paths (much) better than those found in original attacks. Various attempts made to automate the search.
- Satoh, IBM Japan (2005) - collisions for **SHA-1** COULD be found on a $10M special system in 127 days.

Faking documents

Lenstra, Wang, Yin (2005)

Given $x_1 \neq x_2 \wedge MD5(x_1) = MD5(x_2)$ construct two well-formed distinct RSA moduli n_1, n_2 with prefixes x_1, x_2.

This leads to two X.509 PKI certificates, differing only on the public key, but with the same MD5 hash.

Old trick on ASCII texts, philosophical

Any text has 2^k equivalent versions for any k. Thus, for any two texts there exist their equivalent versions colliding for SHA-x.

New trick on the Net, really scary

For any two texts one can effectively produce their postscript equivalents colliding for MD5 (same for pdf, WORD, tiff, ...).

Recommendations

- no more MD5
- NIST: SHA-1 out by 2010
- use SHA-x, $x \geq 224$
- in each case analyze which type of resistance is really needed, if only preimage then SHA-1 may stay around little longer
- design the new hash **AHS**, long time (30+ years) solution, should be parametrizable

SHA-3 acceptability requirements

- **A.1**
  Free worldwide.
- **A.2**
  Implementable on varied hardware and software platforms.
- **A.3**
  Must support 224, 256, 384 and 512 bit digests, and messages of at least up to 2^{64} bits.

SHA-3 submission requirements

August 31, 2008 - optional presubmission
October 31, 2008 - full submission

- **B.1** Completely specified, rationale given for choices made, attack scenarios and resistance analysis, parameterizable
- **B.2** Source in ANSI C
- **B.3** Time and space requirements for hardware and software for 8-, 32- and 64-bit platforms
- **B.4** Documentation in English
- **B.5** Issued or pending patents
- **B.6** Self-evaluation

SHA-3 evaluation criteria

- **C.1**
  Security
- **C.2**
  Cost (time and space complexity)
- **C.3**
  Algorithm and implementation characteristics (flexibility, parameterizable, easy to parallelize, and ... simplicity)

HFC/SHA-3 competition calendar

Hash Function Candidate timeline

- 2007, 1-3Q - minimum requirements
- 2008, October 31 - submissions deadline
- 2009, public comments period
  2Q - First **HFC** Conference
- 2010, public comments period
  2Q - Second **HFC** Conference
  4Q - final round begins
- 2011, 4Q - end of public comments
- 2012, 1Q - Final **HFC** Conference
  2Q - select the winner
  3Q - draft documents
  public comments, tuning up
  4Q - **SHA-3** proposed to
  the Secretary of Commerce

General AHS design

- Must be resistant to all known attacks
- Small memory and long inputs seem to imply an iterated hash
- A compression doing less job than a block cipher was and can be risky
- Secure hashes from modular number theory are possible, but painfully slow
- Rather one parameterized hash than several special purpose hashes
- Take constants from math (like fractional part of $\sqrt[3]{p_i}$ in SHA-2, p_i the i-th prime). Constants in DES and SHA-1 are a mystery.

Block cipher based AHS

Nice properties

- Can be massively parallelized,
  NONE of the standard hashes can.
- Resistant to the length extension attack
- Immune to linear cryptanalysis
- Immune to differential cryptanalysis
- Good confusion
- Good diffusion
- Uses better understood components
- Incrementability. Small length-preserving message changes permit fast hash update.

Block cipher based AHS

Things to look at

- Get large blocks from large ciphers like **W** of AES-type Whirlpool or Maelstrom.
- Compression must be fast, and so better be byte and word oriented, and easy to parallelize and pipeline in hardware.
- For software parallelizability use some tree-structured result collection.

The dilemma of
SEQUENTIAL vs. PARALLELIZABLE

References

1. Mihir Bellare, Roch Guérin and Philip Rogaway, XOR MACs: New Methods for Message Authentication Using Finite Pseudorandom Functions, *LNCS* **963**, (1995) 15–28.
2. Mihir Bellare and Daniele Micciancio, A new paradigm for collision-free hashing: incrementality at reduced cost, *LNCS* **1233**, (1997) 163–192.
3. Eli Biham, Recent Advances in Hash Functions: The Way to Go, slides from various conference presentations, 2005.
4. John Black, Martin Cochran and Trevor Highland, A Study of the MD5 Attacks: Insights and Improvements, *LNCS* **4047**, (2006) 262–277.
5. Vlastimil Klima, posts at the NIST Cryptographic Hash Project forum, 2007, `hash-forum@nist.gov`.
6. National Institute of Standards and Technology, Cryptographic Toolkit, Secure Hashing, specification of SHA-2 standards, 2002, `http://csrc.nist.gov/CryptoToolkit/tkhash.html`
7. National Institute of Standards and Technology, Tentative Timeline of the Development of New Hash Functions, 2007, `http://www.csrc.nist.gov/pki/HashWorkshop/timeline.html`
8. MD5, `http://en.wikipedia.org/wiki/MD5`
9. Alfred J. Menezes, Paul C. van Oorschot and Scott A. Vanstone, *CRC Handbook of Applied Cryptography*, CRC Press 1996. `http://www.cacr.math.uwaterloo.ca/hac`
10. Vincent Rijmen and Paulo S. L. M. Barreto, The WHIRLPOOL Hash Function (2003), `http://paginas.terra.com.br/informatica/paulobarreto/WhirlpoolPage.html`
11. Akashi Satoh, Hardware Architecture and Cost Estimates for Breaking SHA-1, *LNCS* **3650**, (2005) 259–273.

References

12. Zoe Emily Schnabel, The Estimation of the Total Fish Population of a Lake, *American Mathematical Monthly*, **45** (6), (1938) 348–352.
13. Bruce Schneier, *Applied Cryptography*, second edition, John Wiley & Sons, 1996.
14. William Stallings, The Whirlpool Secure Hash Function, *Cryptologia*, **30**, (2006) 55–67.
15. Douglas R. Stinson, *Cryptography: Theory and Practice*, third edition, CRC Press 2006.
16. Douglas R. Stinson, Some observations on the theory of cryptographic hash functions, *Designs, Codes and Cryptography* **38**, (2006) 259–277.
17. Xiaoyun Wang and Hongbo Yu, How to Break MD5 and Other Hash Functions, *LNCS* **3494**, (2005) 19–35.
18. Xiaoyun Wang, Hongbo Yu and Yiqun Lisa Yin, Efficient Collision Search Attacks on SHA-0, *LNCS* **3621**, (2005) 1–16.
19. Xiaoyun Wang, Yiqun Lisa Yin and Hongbo Yu, Finding Collisions in the Full SHA-1, *LNCS* **3621**, (2005) 17–36.

Revisions

Revision #1, March 2007
Revision #2, August 2008