Hash Functions: Practical Implications of Recent Analytic Results

Bill Burr
Manager, Security Technology Group
NIST
willliam.burr@nist.gov

Hash Functions

- Hash functions take a variable-length message and reduce it to a shorter fixed *message digest*
- Many applications: “Swiss army knives” of cryptography:
  - Digital signatures
  - Random number generation
  - Key update and derivation
  - One way function
  - Message authentication codes
  - Integrity protection
  - Malicious code recognition

Structure of Common Hash Functions

- Take a long message, break it into blocks
  - M_1, M_2, M_3…M_s (pad out last block)
- Let H be *n*-bits of state, and *f* be a "compression function" that operates on a block and the current state and "mixes" the block into the state
  - H_0 = some initialization vector
  - $H_i = f(H_{i-1}, M_i)$ for i=1,2,3…s
- Last output of compression function is the *n*-bit hash value or message digest.

Hash Function Properties

- Collision resistant
  - Can’t find any two different messages with the same hash value
- One Way
  - Given only a hash value, can’t construct a message (or “preimage”) that generates the hash. An attack that generates a second message with the same hash value as a given message is called a second pre-image attack.

Finding Hash Collisions

- Find two messages with the same digest
- Birthday “paradox”
  - Given a population of *n* equally probable values, we need roughly $\sqrt{n}$ random samples to expect to find a single collision
- Therefore any attack that finds a collision in much under $2^{n/2}$ operations is said to “break” the collision resistance property of the hash function

Finding Preimages

- Work backward from message digest to find a message that will produce it
- Expect to have to hash about 2^n messages to find an unknown pre-image for any particular selected message digest value
  - Any attack that finds a preimage in significantly under 2^n operations is a break of the one-way property or preimage resistance of a hash function.

Digital Signatures

- Many applications for hashes
- Digital signatures are perhaps the most demanding
  - Hash the message, then apply private key to the hash to generate the signature
- Potentially subject to collision attacks and second preimage attacks

Signature Collision Attack

- Find 2 messages with opposite meanings and the same digest value
  - I agree that…
  - I do not concur…
- Sign one, then repudiate the signature by claiming that you signed the other
- Collisions have to be found *before you sign*
- Doesn't help to forge a signature with an unknown private key
- $2^{n/2}$ work factor
  - SHA-1 gives about 80-bit security against collisions

Signature Second Preimage Attack

- Take a signed message and find a second message with the same message digest (the second preimage)
- You have just forged a signature for the second message
- Much harder than collision attack
  - 2^n versus $2^{n/2}$ operations
  - For SHA-1 about 160-bit security against a second preimage
- Can do any time after the first signature is created

Attack Summary

- Collision attack
  - Allows signer to repudiate signature
  - Must do before signing
  - $2^{n/2}$ operations – comparatively easy (but we make hashes big enough that it's still very hard)
- Second preimage attack
  - Allows anybody to forge a signature
  - Can do anytime after first signature
  - 2^{n} operations – comparatively hard
- We don't want to allow either one

Currently Used Hash Functions

- Only two in wide use in US today
  - MD5
    - Invented by Ron Rivest circa 1992
    - 128-bit hash
    - “Almost broken” by Hans Dobbertin circa 1995
    - Fully broken by collision attack Wang *et. al.* 2004
  - SHA-1
    - Developed by NSA circa 1995
    - “Apparently minor” revision of SHA-0
    - 160-bit hash
    - Not broken to date

MD5

- NIST never felt 128-bits was enough for a digital signature, so never adopted MD5
- “Nearly broken” in 1995 by Hans Dobbertin
  - Found collisions in the compression function itself
  - We were warned:
    - “The presented attack does not yet threaten practical applications of MD5, but it comes rather close… Therefore we suggest that in the future MD5 should no longer be implemented in applications like signature schemes where a collision-resident hash function is required.”
      - *Cryptobytes* Summer 1996

SHA-1

- FIPS 180-1, 160-bit message digest
- Compression function has an initial block expansion and 80 “rounds” of mixing
- SHA-1derived from SHA-0
  - Apparently minor revision: adds a rotate to the initial block expansion
    - This turns out to block recent differential hash collisions attacks
- NIST plans to end federal use of SHA-1 by 2010 in favor of SHA-256, to forestall future brute force collision attacks

Crypto 2004 Hash Breaks

- Between them Eli Biham, Fari Chen, Antoine Joux, Xiaoyung Wang, Xuejia Lai, Dengguo Feng, and Hongbo Yu presented successful full collision attacks on MD4, MD5, HAVAL-128, HAVAL-160, RIPEND and SHA-0 at Crypto 2004.
  - Of these only MD5 is widely used in the US today
  - ***All these algorithms are broken now*** and should not be used to generate signatures

Crypto 2004 SHA-1 Results

- Eli Biham found that you can break SHA-1 if you reduce the number of rounds from 80 to about 50
  - The rotate complicates using the differences
  - Can always break an algorithm if you simplify it enough
  - “Safety margin” still comparable to many other algorithms
- SHA-1 has not been broken and there isn’t a specific reason to suppose that it will be

Bottom line

- Collisions facilitate repudiation but not forgery
- Take this seriously:
  - Don't use MD5 or any of the broken hashes for signatures
- SHA-1 not broken
  - Not much reason to expect it will be any time soon
- NIST plans to phase out all 80-bit crypto by 2010
  - Moore's Law & brute force threatens all 80-bit crypto after 2010
    - SHA-1 for signatures, 1024 RSA/DSA, 160-bit EC-DSA, Skipjack
  - HMAC SHA-1 is OK after 2010 because it depends on the one way property of SHA-1, not its collision resistance
- FIPS 180-2 already in place with SHA-224, SHA-256, SHA-384 and SHA-512
  - Not much public analysis of these algorithms yet

Comparable Strengths

Size in bits

Sym. Key	56	80	112	128	192	256
Hash (for signatures)	128	160	224	256	384	512
Pub. Key	512	1k	2k	3k	7.5k	15k
EC	160		224	256	384	512

Sym. Key: Symmetric key encryption algorithms
MAC: Message Authentication Code
Pub. Key: Factoring or discrete log based public key algorithms
EC: Elliptic Curve based public key algorithms
White background: expected to be secure until at least 2030
Yellow background: Phase out use by 2010
Black background: not secure now

NIST
National Institute of Standards and Technology