Cryptographic Hash Functions

Raj Jain
Washington University in Saint Louis
Saint Louis, MO 63130
Jain@cse.wustl.edu

Audio/Video recordings of this lecture are available at:

http://www.cse.wustl.edu/~jain/cse571-11/

Overview

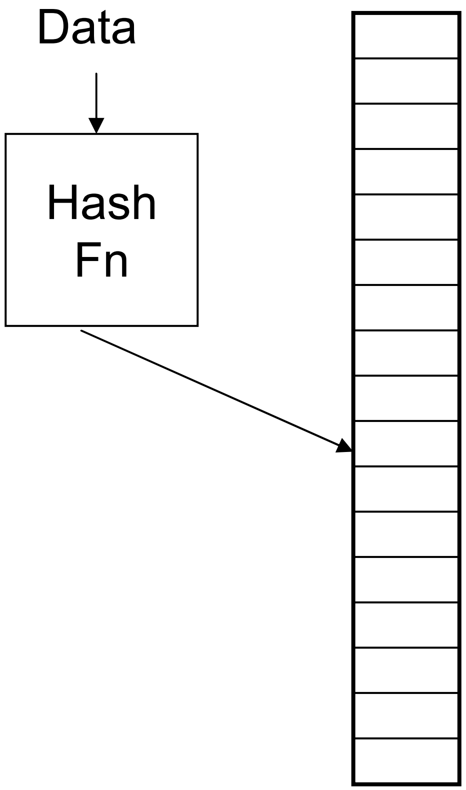
1. Cryptographic Hash Functions
2. Applications of Crypto Hash Functions
3. Birthday Problem
4. Secure Hash Algorithm (SHA)

These slides are based partly on Lawrie Brown's slides supplied with William Stallings's book "Cryptography and Network Security: Principles and Practice," 5th Ed, 2011.

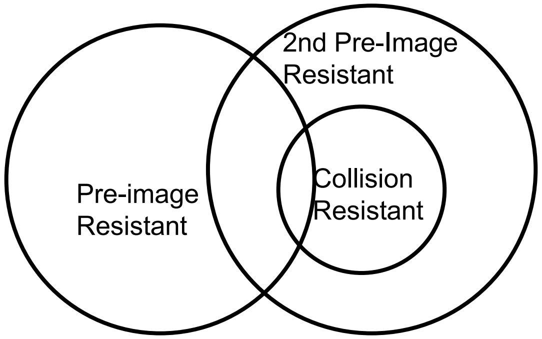
Hash Function

- Hash tables used in data searches
- The hash function should

1. Take variable size input
2. Produce fixed output size (Size of the table)
3. Be easy to compute
4. Be pseudorandom so that it distributes uniformly over the table
   ⇒ Minimizes collisions



Cryptographic Hash Functions



1. Variable Size Input
2. Fixed output size
3. Efficient computation
4. Pseudorandom
5. Pre-image Resistant = one-way
   It is not possible to find M, given h.
6. 2^{nd} Pre-image Resistant: = Weak Collision Resitant
   It is not possible to find y, such that h(y)=h(x)
7. Strong Collision Resistant: It is not possible to find any two x and y, such that h(y)=h(x)

Examples of Crypto Hash Functions

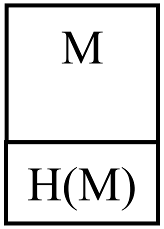
- MD4 = Message Digest 4 [RFC 1320] - 32b operations
- MD5 = Message Digest 5 [RFC 1321] - 32b operations
- SHA = Secure hash algorithm [NIST]
- SHA-1 = Updated SHA
- SHA-2 = SHA-224, SHA-256, SHA-384, SHA-512
  SHA-512 uses 64-bit operations

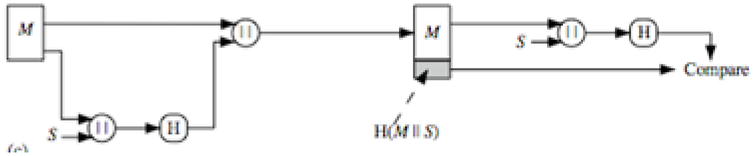
Applications of Crypto Hash Fn

1. **Message Authentication** = Integrity

   MD5 has is used to check if a file has been modified.

   Use a secret value before hashing so that no one else can modify M and hash

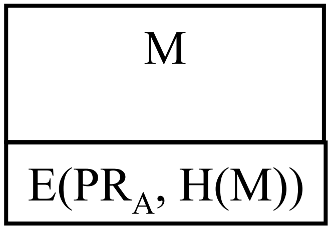




   Can encrypt Message, hash, or both for confidentiality

2. **Digital Signatures**: Encrypt hash with private key

3. **Password storage**: Hash of the user's password is compared with that in the storage. Hackers can not get password from storage.



4. **Pseudorandom number generation**: Hash an IV, Hash the hash, …, repeat

Birthday Problem

- What is the probability that two people have the same birthday (day and month)

K	Total	Different
2	365^2	365×364
3	365^3	$365 \times 364 \times 363$
$\cdots$		
k	365^k	$365 \times 364 \times 363 \times \cdots \times (365 - k + 1)$

$$\begin{aligned} P(\text{No common day}) &= \frac{365 \times 364 \times 363 \times ... \times (365 - k + 1)}{365^k} \\ &= \frac{365!}{365^k(365 - k)!} \end{aligned}$$

Birthday Problem (Cont)

- With 22 people in a room, there is better than 50% chance that two people have a common birthday
- With 40 people in a room there is almost 90% chance that two people have a common birthday
- If there k people, there are k(k-1)/2 pairs

$$\text{P(1 pair having common birthday)} = \frac{k(k-1)}{2\times 365}$$

$$k \geq \sqrt{365} \Rightarrow P > 0.5$$

- In general, n possibilities
  ⇒ √n trials to find a collision

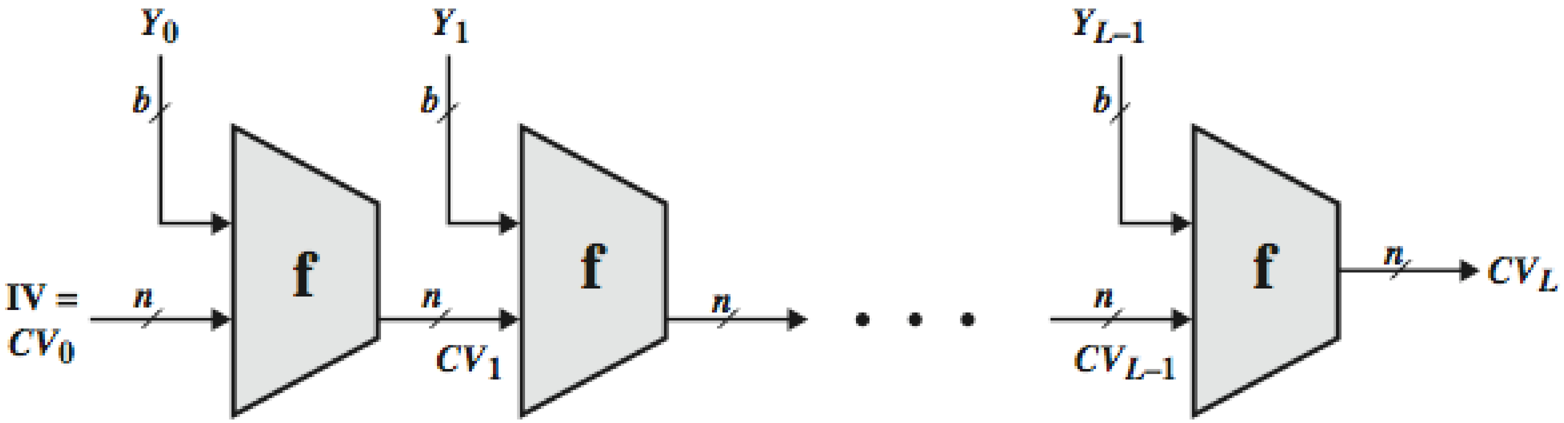
k	P
2	.01
3	.02
4	.03
...	...
19	.41
20	.44
21	.48
22	.51
23	.54
...	...
38	.88
39	.89
40	.90

Probability of Hash Collisions

- Arbitrary length message $\Rightarrow$ Fixed length hash
  $\Rightarrow$ Many messages will map to the same hash
- Given 1000 bit messages $\Rightarrow 2^{1000}$ messages
- 128 bit hash $\Rightarrow 2^{128}$ possible hashes
  $\Rightarrow 2^{1000}/2^{128} = 2^{872}$ messages/hash value
- n-bit hash $\Rightarrow$ Need avg $2^{n/2}$ tries to find two messages with same hash
- 64 bit hash $\Rightarrow 2^{32}$ tries (feasible)
- 128 bit hash $\Rightarrow 2^{64}$ tries (not feasible)

Hash Function Cryptanalysis

- Hash functions use iterative structure
  - Process message in blocks
- Compression function **f** takes previous output and next block to produce next output
- If compression function is collision resistant, the entire structure is collision resistant [Merkle 89]



Block Ciphers as Hash Functions

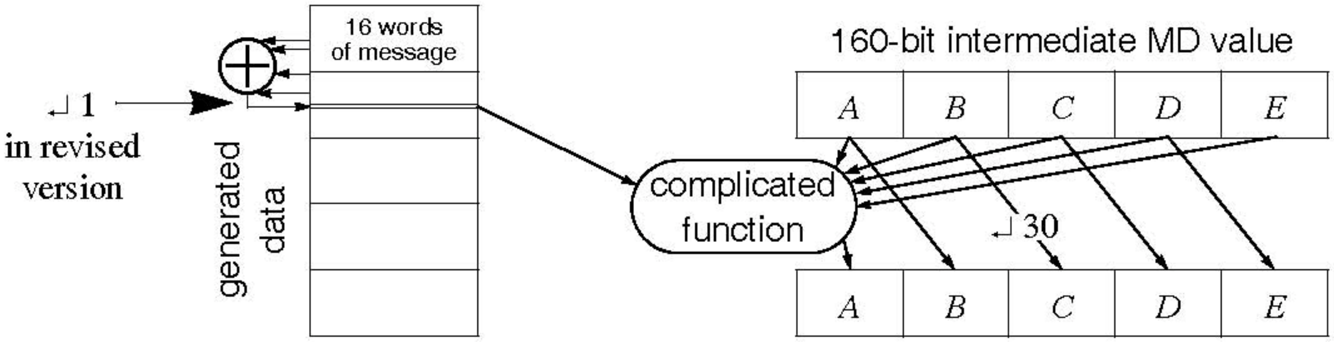
- Can use block ciphers as hash functions
  - Using H_0=0 and zero-pad of final block
  - Compute: $H_i = E_{M_i} [H_{i-1}]$
  - And use final block as the hash value
  - Similar to CBC but without a key
- Resulting hash is too small (64-bit)
  - Both due to direct birthday attack
  - And to "meet-in-the-middle" attack
- Other variants also susceptible to attack

Secure Hash Algorithm (SHA)

- Successor to and similar to MD5 (by Ron Rivest)
- SHA-0: FIPS PUB 180, 1993. Withdrawn shortly after publ.
- SHA-1: FIPS PUB 180-1, 1995. 160 bit hash
- SHA-2: FIPS PUB 180-2, 2002
  - SHA-224
  - SHA-256
  - SHA-384
  - SHA-512
- SHA-1 is used in TLS, SSL, PGP, SSH, S/MIME, and IPsec
  - Required by law in US Govt applications
  - Used in Digital Signature Standard
- Pseudo-codes for SHA algorithms are available.
- NIST certifies implementations.

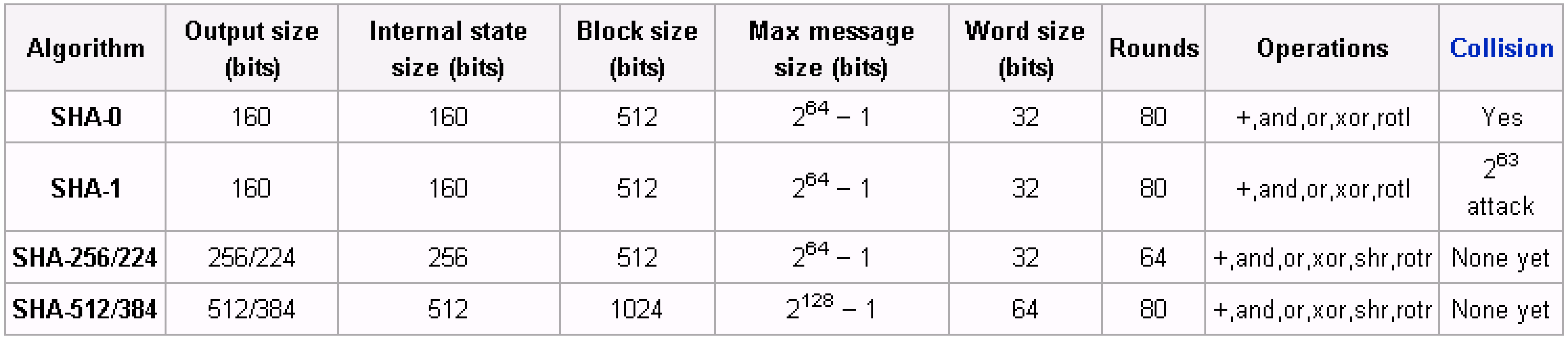
SHA-1 Algorithm

- 160 bit hash using 512 bit blocks and 32 bit operations
- Five passes (4 in MD5 and 3 in MD4) of 16 operations each
- Maximum message size is 2^{64} bit
- 512 bits are expanded to 5x512 bits:
  - n^{th} word = xor of n-3, n-8, n-14, and n-16
- In SHA-1 these words are rotated left by one bit before xor
- Total 80 words: $W_0, \ldots, W_{79}$



SHA-2

- SHA-256 uses 32-bit operations
- SHA-512 uses 64-bit operations
- Use different shift amounts and additive constants
- SHA-224 and SHA-384 are simply truncated versions of SHA-256 and SHA-512 using different initial values.
- SHA-224 matches the key length of two-key triple-DES

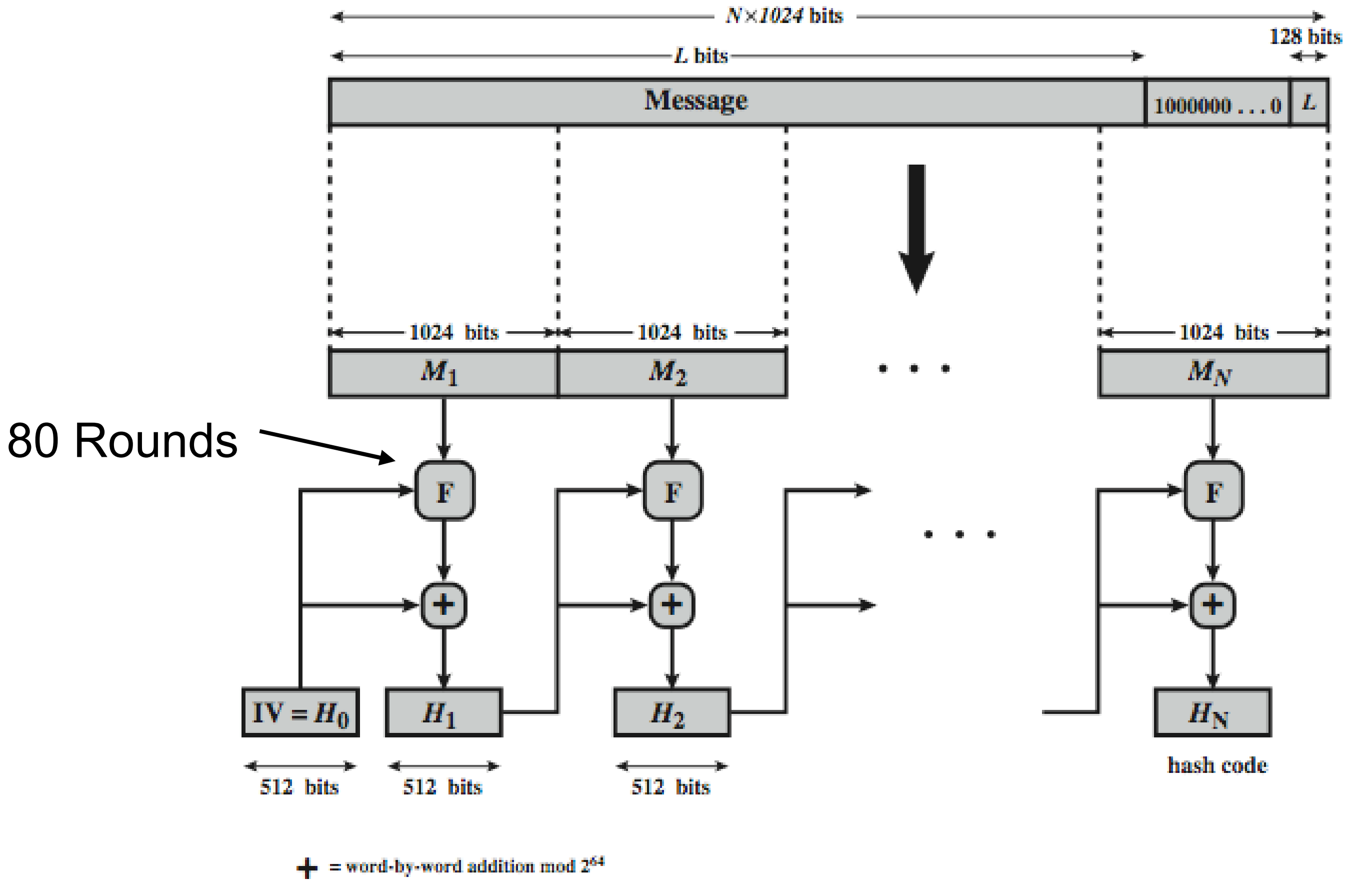


Algorithm	Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Word size (bits)	Rounds	Operations	Collision
SHA-0	160	160	512	$2^{64} - 1$	32	80	+,and,or,xor,rotl	Yes
SHA-1	160	160	512	$2^{64} - 1$	32	80	+,and,or,xor,rotl	2^{63} attack
SHA-256/224	256/224	256	512	$2^{64} - 1$	32	64	+,and,or,xor,shr,rotr	None yet
SHA-512/384	512/384	512	1024	$2^{128} - 1$	64	80	+,and,or,xor,shr,rotr	None yet

[Source: Wikipedia]

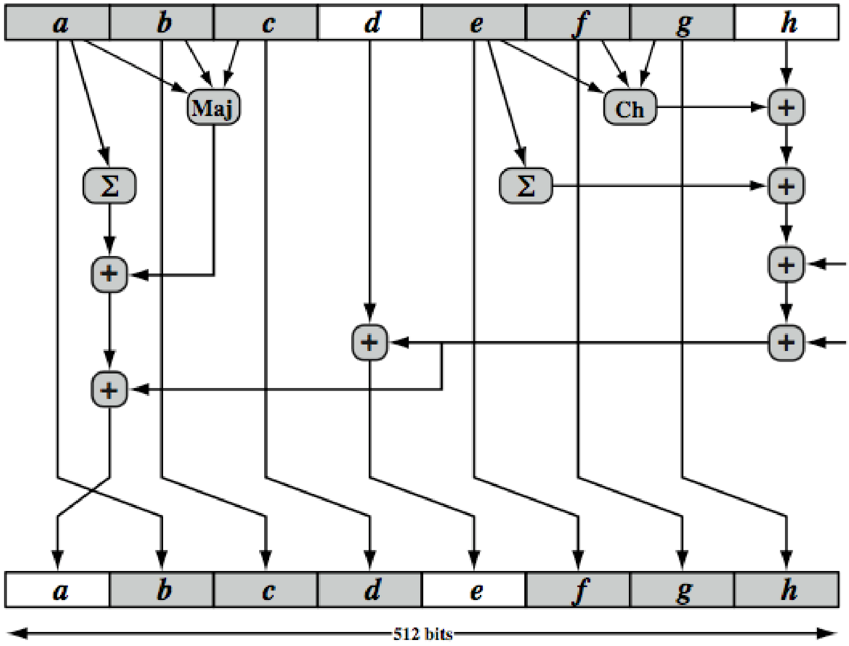
SHA-512 Overview

- 1. Append padding bits
- 2. Append length



80 Rounds

SHA-512 Round Function

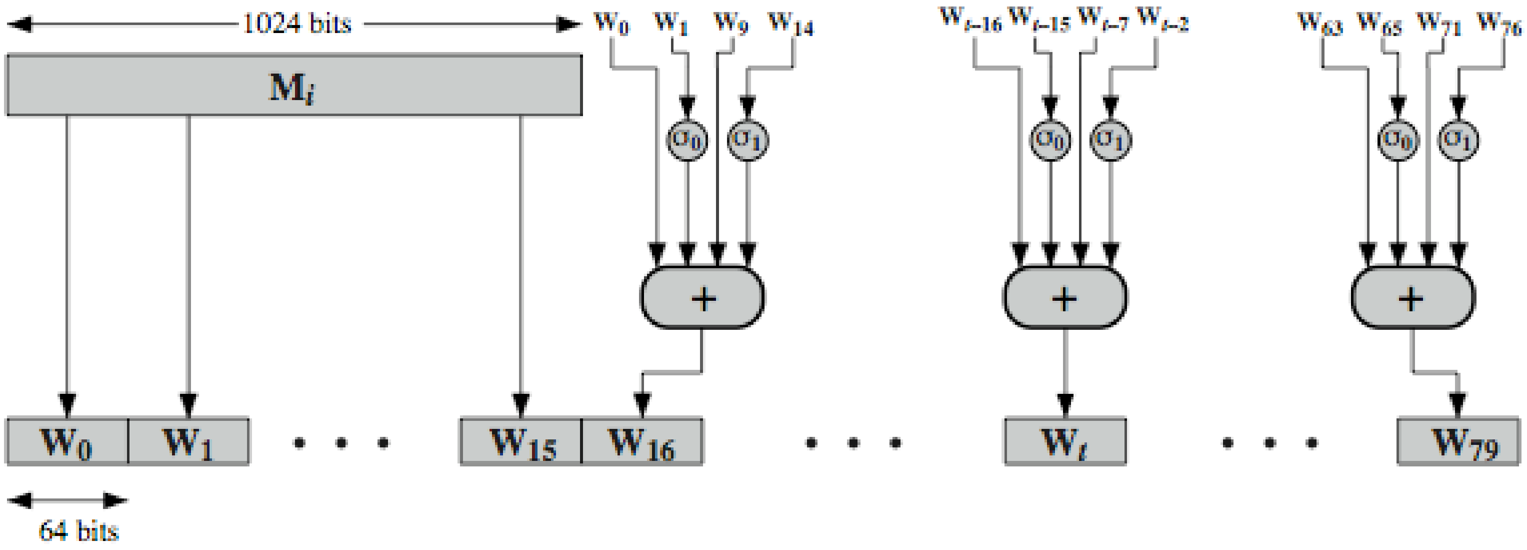


Input sequence word

Constant (see table)

- Conditional fn Ch(e,f,g): if e then f else g
  = (e AND f) ⊕(Not e and g)
- Majority Fn Maj(a, b, c): True if 2 of 3 args are true
  =(a AND b) ⊕(a AND c) ⊕(b AND c)

80-Word Input Sequence


1024 bits
M_i
W_0 W_1 W_9 W_{14}
W_{t-16} W_{t-15} W_{t-7} W_{t-2}
W_{63} W_{65} W_{71} W_{76}
σ_0
σ_1
+
W_0
W_1
W_{15}
W_{16}
W_t
W_{79}
64 bits

- $W_t = \sigma_1(W_{t-2}) + W_{t-7} + \sigma_0(W_{t-15}) + W_{t-16}$
- $\sigma_0(x) = \mathrm{ROTR}^1(x) + \mathrm{ROTR}^8(x) + \mathrm{SHR}^7(x)$
- $\sigma_1(x) = \mathrm{ROTR}^{19}(x) + \mathrm{ROTR}^{61}(x) + \mathrm{SHR}^6(x)$
- $\mathrm{ROTR}^n(x)$=rotate right by n bits
- $\mathrm{SHR}^n(x)$=Left shift n bits with padding by 0's on the right
- + = Addition modulo 2^{64}

SHA-3

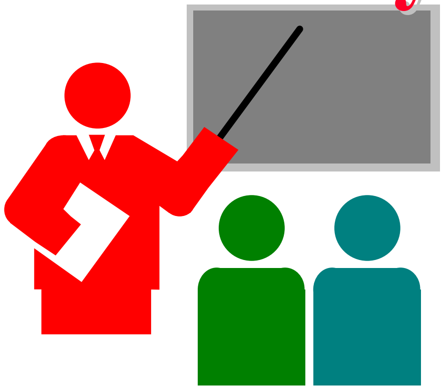
- SHA-2 (esp. SHA-512) seems secure
  - Shares same structure and mathematical operations as predecessors so have concern
- NIST announced in 2007 a competition for the SHA-3
  - Has had 3 rounds of narrowing down the selections
  - Five algorithms advanced to the third (and final) round in December 2010
  - Final selection to be announced by 2012

Ref: http://en.wikipedia.org/wiki/NIST_hash_function_competition

SHA-3 Requirements

- Replace SHA-2 with SHA-3 in any use
  - So use same hash sizes
- Preserve the online nature of SHA-2
  - So must process small blocks (512 / 1024 bits)
- Evaluation criteria
  - Security close to theoretical max for hash sizes
  - Cost in time & memory
  - Characteristics: such as flexibility & simplicity

Summary



1. Hash functions are used to get a digest of a message Must take variable size input, produce fixed size pseudorandom output, be efficient to compute
2. Cryptographic hash functions should be preimage resistant, 2nd preimage resistant, and collision resistant
3. Cryptographic hashes are used for message authentication, digital signatures, password storage
4. SHA-1 produces 160 bit output, SHA-224, SHA-256, SHA-384, and SHA-512 produce 224, 256, 384, and 512 bit outputs. All consist of 80 rounds.
5. SHA-3 competition is underway

Homework 11

- Compute the following hash function:

$$h = \left(7 + \sum_{i=1}^{k} (m_i)^2\right) \mod 251$$

  for a 4-byte message M={m_1, m_2, m_3, m_4}={128, 252, 33, 19} All are decimal numbers.

- Check if the hash function is:
  - A. Collision Resistant
  - B. Pre-image resistant
  - B. Second Pre-image Resistant
- Show counter examples for any property that is not satisfied.