Cryptography and Network Security Chapter 11

Fifth Edition

by William Stallings

Lecture slides by Lawrie Brown

(with edits by RHB)

Chapter 11 – Cryptographic Hash Functions

Each of the messages, like each one he had ever read of Stern's commands, began with a number and ended with a number or row of numbers. No efforts on the part of Mungo or any of his experts had been able to break Stern's code, nor was there any clue as to what the preliminary number and those ultimate numbers signified.

—*Talking to Strange Men,* Ruth Rendell

Outline

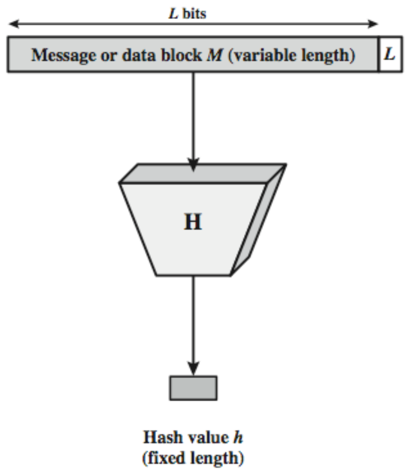
- we consider:
  - hash functions
    - — uses, requirements, security
  - hash functions based on block ciphers
  - SHA-1, SHA-2, SHA-3

Hash Functions

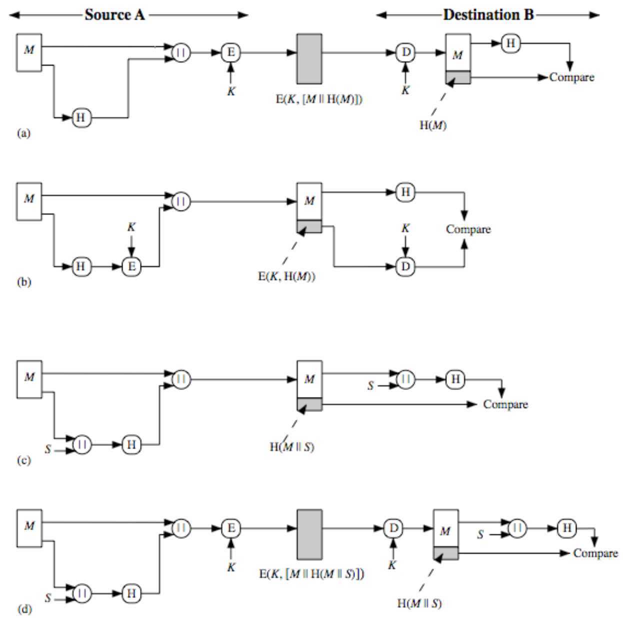
- condenses arbitrary message to a fixed size

  `h = H(M)`
- usually assume hash function is public
- hash used to detect changes to message
- want a cryptographic hash function such that
  - computationally infeasible to find data that maps to a specific hash (one-way property)
  - computationally infeasible to find two different data with same hash (collision-free property)

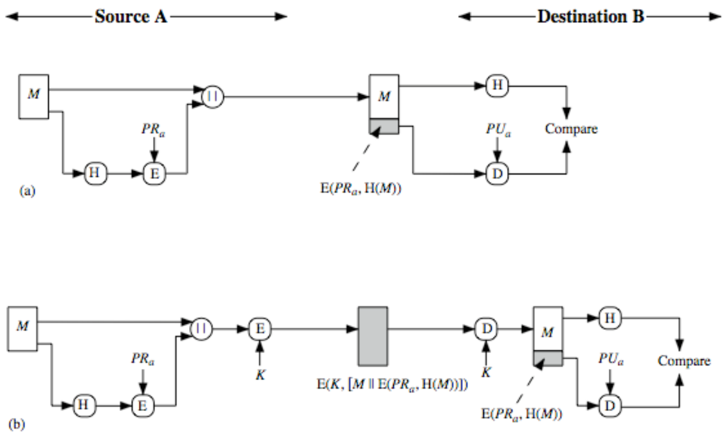
Cryptographic Hash Function



Hash Functions & Message Authent-ication



Hash Functions & Digital Signatures



Other Hash Function Uses

- to create a one-way password file
  - store hash of password, not actual password
- for intrusion detection and virus detection
  - keep & check hashes of files on system
- pseudorandom function (PRF) or pseudorandom number generator (PRNG)

Two Simple Insecure Hash Functions

- consider two simple insecure hash functions
- bit-by-bit exclusive-OR (XOR) of every block
  - $C_i = b_{i1}$ XOR b_{i2} XOR ... XOR b_{im}
  - a longitudinal redundancy check
  - reasonably effective as data integrity check
- one-bit circular shift on hash value
  - for each successive n-bit block
    - rotate current hash value left by 1 bit and XOR block
  - good for data integrity but useless for security

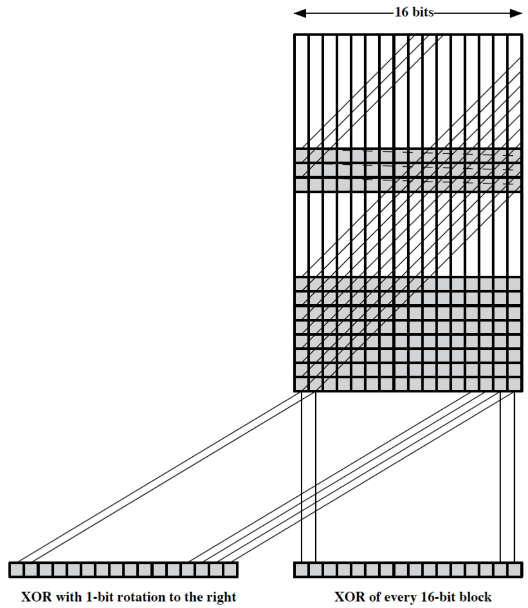


Figure 11.4 Two Simple Hash Functions

Hash Function Requirements

Requirement	Description
Variable input size	H can be applied to a block of data of any size.
Fixed output size	H produces a fixed-length output.
Efficiency	H(*x*) is relatively easy to compute for any given *x*, making both hardware and software implementations practical.
Preimage resistant (one-way property)	For any given hash value *h*, it is computationally infeasible to find *y* such that H(*y*) = *h*.
Second preimage resistant (weak collision resistant)	For any given block *x*, it is computationally infeasible to find *y* ! *x* with H(*y*) = H(*x*).
Collision resistant (strong collision resistant)	It is computationally infeasible to find any pair (*x*, *y*) such that H(*x*) = H(*y*).
Pseudorandomness	Output of H meets standard tests for pseudorandomness

Attacks on Hash Functions

- have brute-force attacks and cryptanalysis
- a preimage or second preimage attack
  - find y s.t. H(y) equals a given hash value
- collision resistance
  - find two messages x and y with same hash H(x) = H(y)
- hence value $2^{m/2}$ determines strength of hash code against brute-force attacks
  - 128-bits inadequate, 160-bits suspect

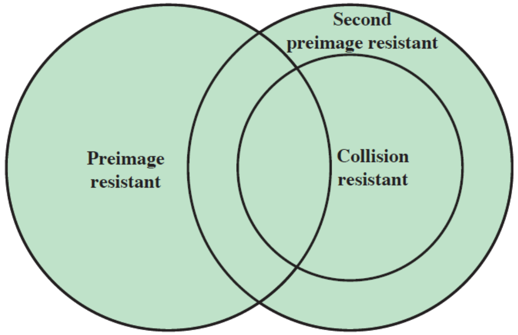
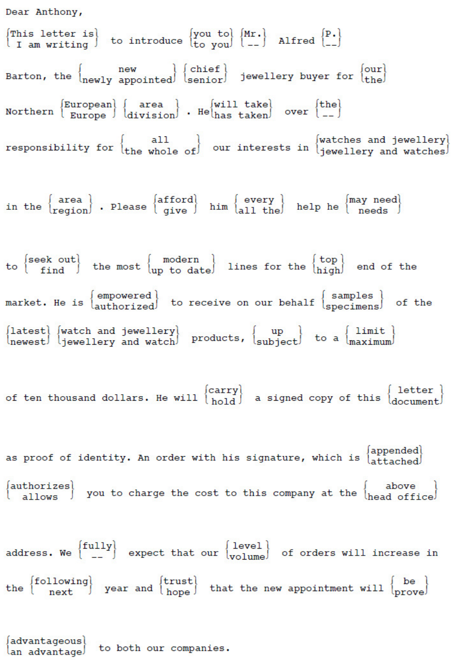


Figure 11.5 Relationship Among Hash Function Properties

Birthday Attacks

- might think a 64-bit hash is secure
- but by **Birthday Paradox** is not
- **birthday attack** works thus:
  - given user prepared to sign a valid message x
  - opponent generates $2^{m/2}$ variations x' of x, all with essentially the same meaning, and saves them
  - opponent generates $2^{m/2}$ variations y' of a desired fraudulent message y
  - two sets of messages are compared to find pair with same hash (probability > 0.5 by birthday paradox)
  - have user sign the valid message, then substitute the forgery which will have a valid signature
- conclusion is that need to use larger MAC/hash



Dear Anthony,

{This letter is / I am writing} to introduce {you to / to you} {Mr. / --} Alfred {P. / --} Barton, the {new / newly appointed} {chief / senior} jewellery buyer for {our / the} Northern {European / Europe} {area / division}. He {will take / has taken} over {the / --} responsibility for {all / the whole of} our interests in {watches and jewellery / jewellery and watches} in the {area / region}. Please {afford / give} him {every / all the} help he {may need / needs} to {seek out / find} the most {modern / up to date} lines for the {top / high} end of the market. He is {empowered / authorized} to receive on our behalf {samples / specimens} of the {latest / newest} {watch and jewellery / jewellery and watch} products, {up / subject} to a {limit / maximum} of ten thousand dollars. He will {carry / hold} a signed copy of this {letter / document} as proof of identity. An order with his signature, which is {appended / attached} {authorizes / allows} you to charge the cost to this company at the {above / head office} address. We {fully / --} expect that our {level / volume} of orders will increase in the {following / next} year and {trust / hope} that the new appointment will {be / prove} {advantageous / an advantage} to both our companies.

Figure 11.6 A Letter in 2^{37} Variations

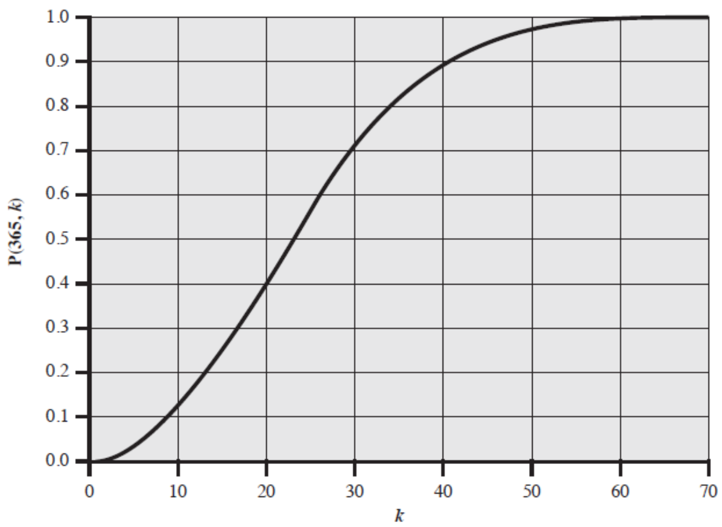


Figure 11.13 The Birthday Paradox

The Birthday Paradox.

Let there be N possible hash values. How many messages are needed before probability of a clash is about $\frac{1}{2}$?

Prob. of NO clash with 2 msgs. $= 1-1/N$

Prob. of NO clash with 3 msgs. $= [1-1/N][1-2/N]$

$\ldots \qquad \ldots$

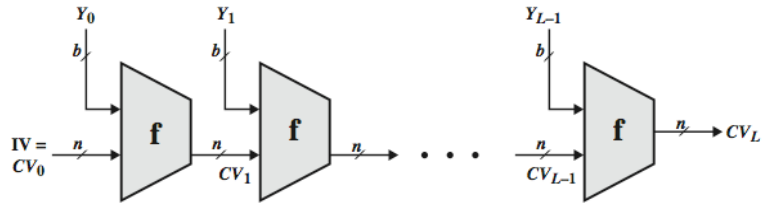
Prob. of NO clash with $m+1$ msgs. $= [1-1/N][1-2/N]\ldots[1-m/N]$

Now $[1-1/x] \approx e^{-\frac{1}{x}}$, so $\prod_{k=1}^{m} [1-k/N] \approx e^{-(\Sigma_{k=1}^{m} k)/N} \approx e^{-m^2/2N} \approx \frac{1}{2}$

if $m \approx 1.2\sqrt{N}$ ..... So if $m \approx \sqrt{N}$, probability of a clash is about $\frac{1}{2}$.

Hash Function Iterative Structure

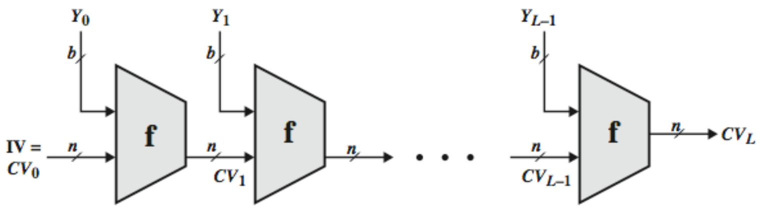
- hash functions use an iterative structure
- they process the message in blocks, with suitable care about padding and length



- attacks focus on collisions in function f

Hash Function Cryptanalysis

- cryptanalytic attacks try to exploit some property of algorithm f
- attempt to do it faster than exhaustive search



Block Ciphers as Hash Functions

- a large number of hash functions exist
- 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 small (64-bit) if use DES
  - both due to direct birthday attack
  - and to "meet-in-the-middle" attack
- other variants also susceptible to attack

Secure Hash Algorithm

- SHA originally designed by NIST & NSA in 1993
- was revised in 1995 as SHA-1
- US standard for use with DSA signature scheme
  - standard is FIPS 180-1 1995, also Internet RFC3174
  - algorithm is SHA, the standard is SHS
- based on design of MD4 with key differences
- produces 160-bit hash values
- 2005 results on security of SHA-1 have raised concerns on its use in future applications
- these days use of SHA-1 is discouraged

Revised Secure Hash Standard

- NIST issued revision FIPS 180-2 in 2002
- adds 3 additional versions of SHA
  - SHA-256, SHA-384, SHA-512
- designed for compatibility with increased security provided by the AES cipher
- structure and detail is similar to SHA-1
- hence analysis should be similar
- but security levels are rather higher

SHA Versions

Table 11.3 Comparison of SHA Parameters

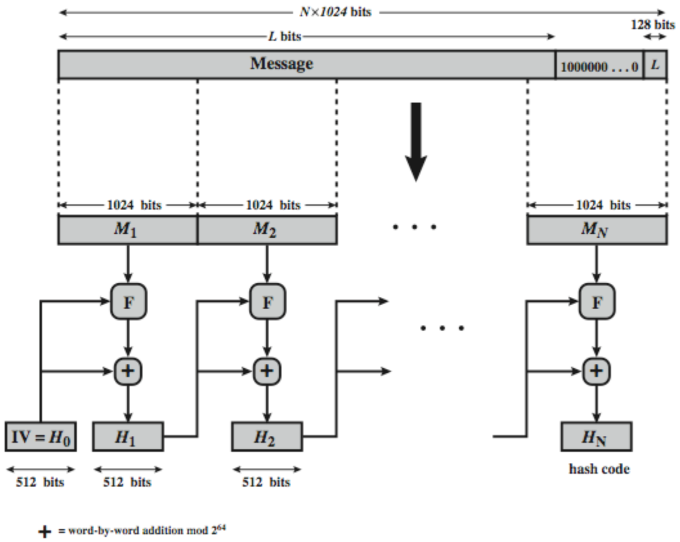
	SHA-1	SHA-224	SHA-256	SHA-384	SHA-512
Message Digest Size	160	224	256	384	512
Message Size	$< 2^{64}$	$< 2^{64}$	$< 2^{64}$	$< 2^{128}$	$< 2^{128}$
Block Size	512	512	512	1024	1024
Word Size	32	32	32	64	64
Number of Steps	80	64	64	80	80

Note: All sizes are measured in bits.

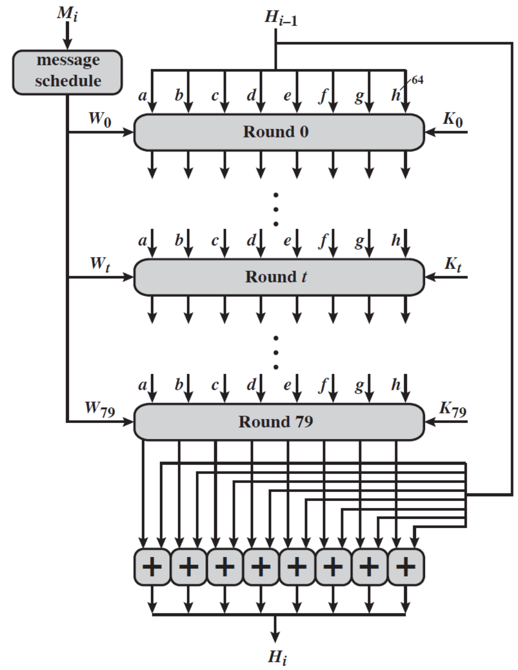
SHA-512 Compression Function

- the heart of the algorithm
- it processes message in 1024-bit blocks
- it consists of 80 rounds per block
  - updating a 512-bit buffer
  - using a 64-bit value Wt derived from the current message block
  - and a round constant based on cube root of first 80 prime numbers

SHA-512 Overview



Processing one 1024 bit block



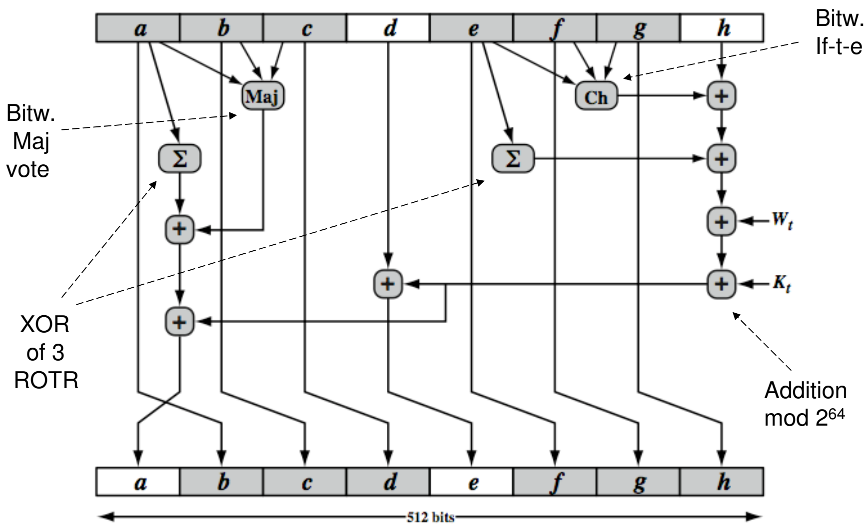
H_i Initial Values

$H_{0,0}$ = 6A09E667F3BCC908	$H_{0,4}$ = 510E527FADE682D1
$H_{0,1}$ = BB67AE8584CAA73B	$H_{0,5}$ = 9B05688C2B3E6C1F
$H_{0,2}$ = 3C6EF372FE94F82B	$H_{0,6}$ = 1F83D9ABFB41BD6B
$H_{0,3}$ = A54FF53A5F1D36F1	$H_{0,7}$ = 5BE0CDI9137E2179

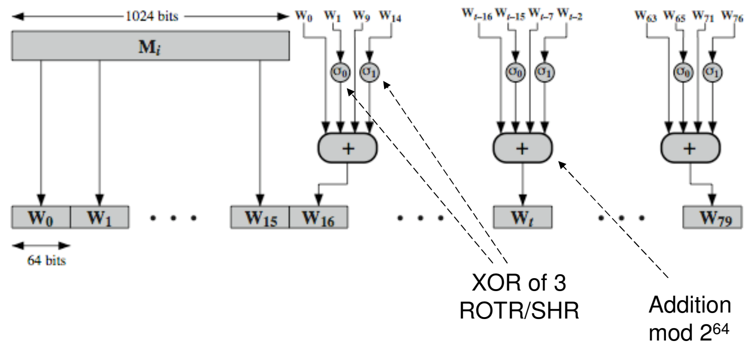
Table 11.4 SHA-512 Constants K_i

428a2f98d728ae22	7137449123ef65cd	b5c0fbcfec4d3b2f	e9b5dba58189dbbc
3956c25bf348b538	59f111f1b605d019	923f82a4af194f9b	ab1c5ed5da6d8118
d807aa98a3030242	12835b0145706fbe	243185be4ee4b28c	550c7dc3d5ffb4e2
72be5d74f27b896f	80deb1fe3b1696b1	9bdc06a725c71235	c19bf174cf692694
e49b69c19ef14ad2	efbe4786384f25e3	0fc19dc68b8cd5b5	240ca1cc77ac9c65
2de92c6f592b0275	4a7484aa6ea6e483	5cb0a9dcbd41fbd4	76f988da831153b5
983e5152ee66dfab	a831c66d2db43210	b00327c898fb213f	bf597fc7beef0ee4
c6e00bf33da88fc2	d5a79147930aa725	06ca6351e003826f	142929670a0e6e70
27b70a8546d22ffc	2e1b21385c26c926	4d2c6dfc5ac42aed	53380d139d95b3df
650a73548baf63de	766a0abb3c77b2a8	81c2c92e47edaee6	92722c851482353b
a2bfe8a14cf10364	a81a664bbc423001	c24b8b70d0f89791	c76c51a30654be30
d192e819d6ef5218	d69906245565a910	f40e35855771202a	106aa07032bbd1b8
19a4c116b8d2d0c8	1e376c085141ab53	2748774cdf8eeb99	34b0bcb5e19b48a8
391c0cb3c5c95a63	4ed8aa4ae3418acb	5b9cca4f7763e373	682e6ff3d6b2b8a3
748f82ee5defb2fc	78a5636f43172f60	84c87814a1f0ab72	8cc702081a6439ec
90befffa23631e28	a4506cebde82bde9	bef9a3f7b2c67915	c67178f2e372532b
ca273eceea26619c	d186b8c721c0c207	eada7dd6cde0eb1e	f57d4f7fee6ed178
06f067aa72176fba	0a637dc5a2c898a6	113f9804bef90dae	1b710b35131c471b
28db77f523047d84	32caab7b40c72493	3c9ebe0a15c9bebc	431d67c49c100d4c
4cc5d4becb3e42b6	597f299cfc657e2a	5fcb6fab3ad6faec	6c44198c4a475817

SHA-512 Round Function



SHA-512 Round Function



SHA-3

- In hashes, nothing secret, easier to attack
- SHA-1 considered insecure these days
- 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 next gen NIST hash function
  - goal was to have it in place by 2012

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 and memory
  - characteristics: such as flexibility and simplicity