Plan of Talk

- **Hash functions.**
- **Attacks on Hash functions :Birthday attack**
- **Secure Hash Algorithm**

Hash Functions

- We looked at the need for redundancy functions that break multiplicative property of RSA.
- What are the choices for this function?

Hash Functions

- A hash function H accepts a variable-length block of data M as input and produces a fixed-size hash value h = H(M).
  - `h = H(M)`
- We usually assume hash function is public.
- In practice hash used to detect changes to message.
- We want a cryptographic hash function
  - One-way property: computationally infeasible to find data mapping to specific hash.
  - Collision-free property: computationally infeasible to find two data to same hash.

Cryptographic Hash Function

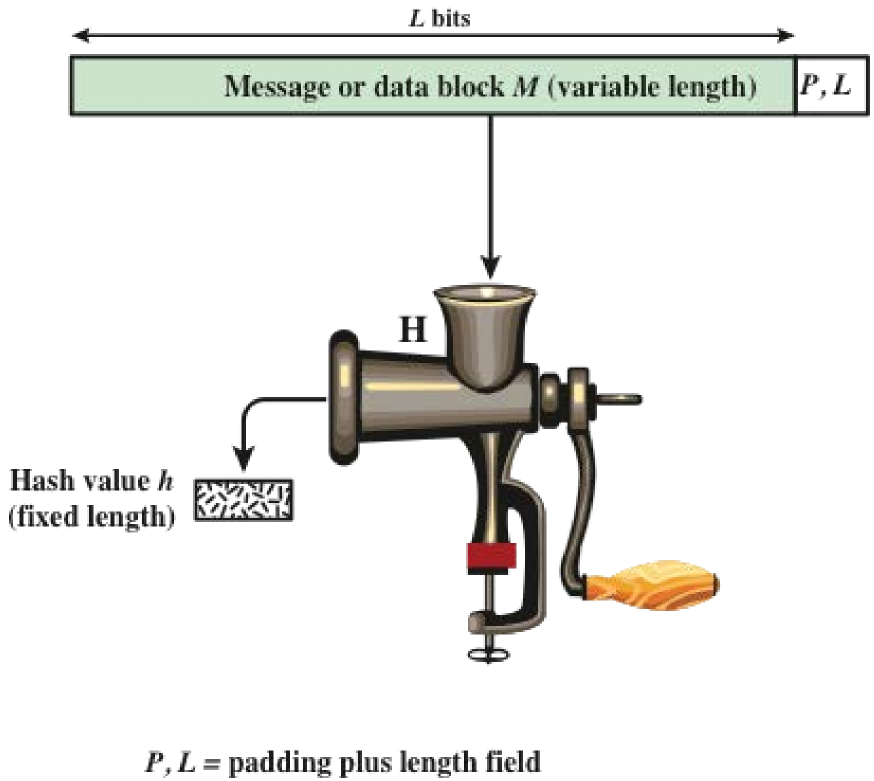


Figure 11.1 Cryptographic Hash Function; $h = H(M)$

Other Hash Function Uses

- One-way password file
  - store hash of password not actual password
- Intrusion detection and virus detection
  - keep & check hash of files on system
- Pseudorandom function (PRF) or Pseudorandom number generator (PRNG)

Two Simple Insecure Hash Functions

- Simple XOR Hash: 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 to left by1bit and XOR block
  - good for data integrity but useless for security

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 $y \neq x$ 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

Hash function relationships

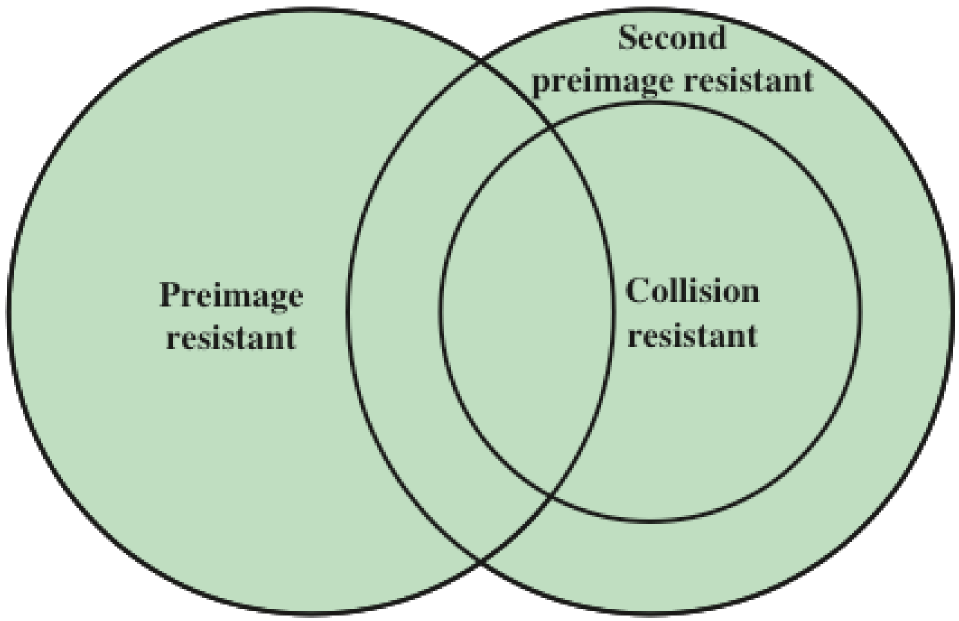


Figure 11.6 Relationship Among Hash Function Properties

Attacks on Hash Functions

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

What is the Birthday Attack?

- Consider a set of events of N different outcomes (think of this as numbers from 1 to N). Assume that they are equally likely. P(i) = 1/N for all i;
- Question: What is the probability that after n trials at least two of the outcomes will be the same?
- Example: Consider 128 bit hash values that are used in the digital signature algorithm for messages of size 1000 bits. There are 2^1000 possibilities for messages. The hash values are of length 128 bits and it is drawn uniformly from the space of 0; 1; 2^128; N here is 2^128.

What is the Birthday Attack?

- Result: The probability that after n trials at least two outcomes will be the same is at least
  - $1 – e^{-(1(n-1)n / 2N)} = \varepsilon$
- $(1 - \varepsilon) = e^{-(1(n-1)n / 2N)}$
- $\ln(1 - \varepsilon) = \ln(e^{-(1(n-1)n / 2N)}) = -n(n-1)/2N$
- $n^2 – n = 2N \ln(1/(1- \varepsilon))$
- $n \cong \mathrm{Sqrt}\left(2N \ln(1/(1- \varepsilon)) \right)$
- Let $\varepsilon = 0.5$, $n \cong 1.17\ \mathrm{Sqrt}(N)$.
- For Birthday problem, N = 365, n = 23.

Birthday Attacks

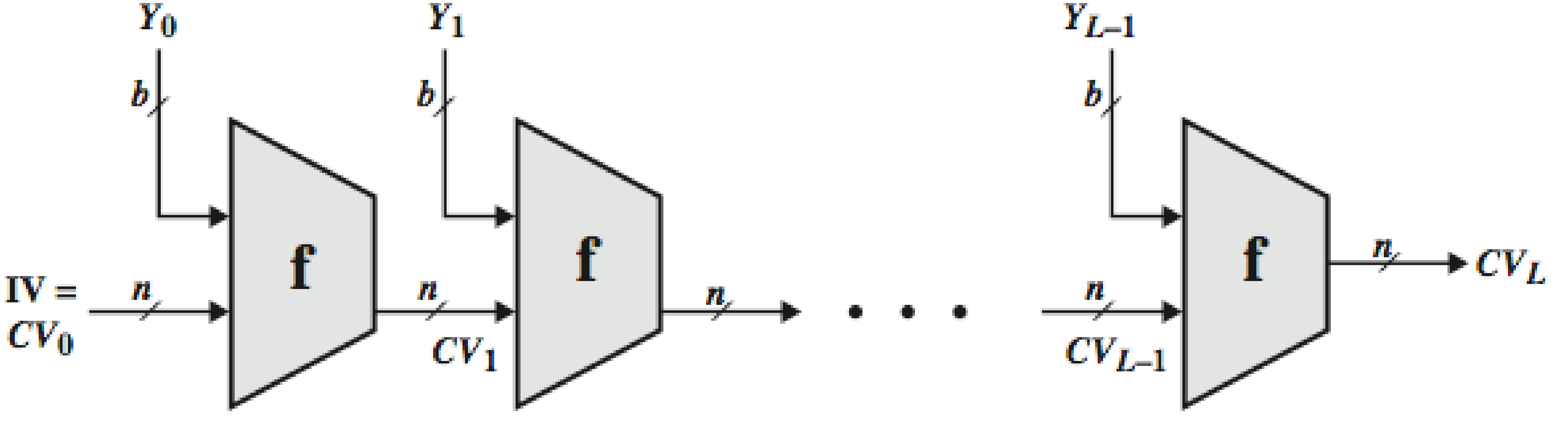
- You miight think a 64-bit hash is secure
- but by **Birthday Paradox** is not
- **birthday attack** works thus:
  - given user prepared to sign a valid message x
  - opponent generates $2^{m/2}$ variations x’ of x, all with essentially the same meaning, and saves them
  - opponent generates $2^{m/2}$ 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

Birthday Attacks

- You miight 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

Hash Function Cryptanalysis

- Cryptanalytic attacks exploit some property of algorithm so faster than exhaustive search
- Hash functions use iterative structure
  - process message in blocks (including length)
- Attacks focus on collisions in function f



Block Ciphers as Hash Functions

- Block ciphers as hash functions (Ex DES)
  - 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 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
  - nb. the algorithm is SHA, the standard is SHS
- Based on design of MD4 with key differences
- Produces 160-bit hash values
- Recent 2005 results on security of SHA-1 have raised concerns on its use in future applications

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 & detail is similar to SHA-1
- Hence analysis should be similar
- But security levels are rather higher

Comparison

	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

SHA Versions

Message Digest Size	224	256	384	512
Message Size	no maximum	no maximum	no maximum	no maximum
Block Size (bitrate *r*)	1152	1088	832	576
Word Size	64	64	64	64
Number of Rounds	24	24	24	24
Capacity *c*	448	512	768	1024
Collision resistance	2^{112}	2^{128}	2^{192}	2^{256}
Second preimage resistance	2^{224}	2^{256}	2^{384}	2^{512}

SHA-512 Compression Function

- heart of the algorithm
- processing message in 1024-bit blocks
- consists of 80 rounds
  - 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-3

- SHA-1 not yet "broken”
  - but similar to broken MD5 & SHA-0
  - so considered insecure
- 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
  - http://csrc.nist.gov/groups/ST/hash/sha-3/

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

- We have considered:
  - hash functions
    - uses, requirements, security
  - hash functions based on block ciphers
  - SHA-1, SHA-2, SHA-3