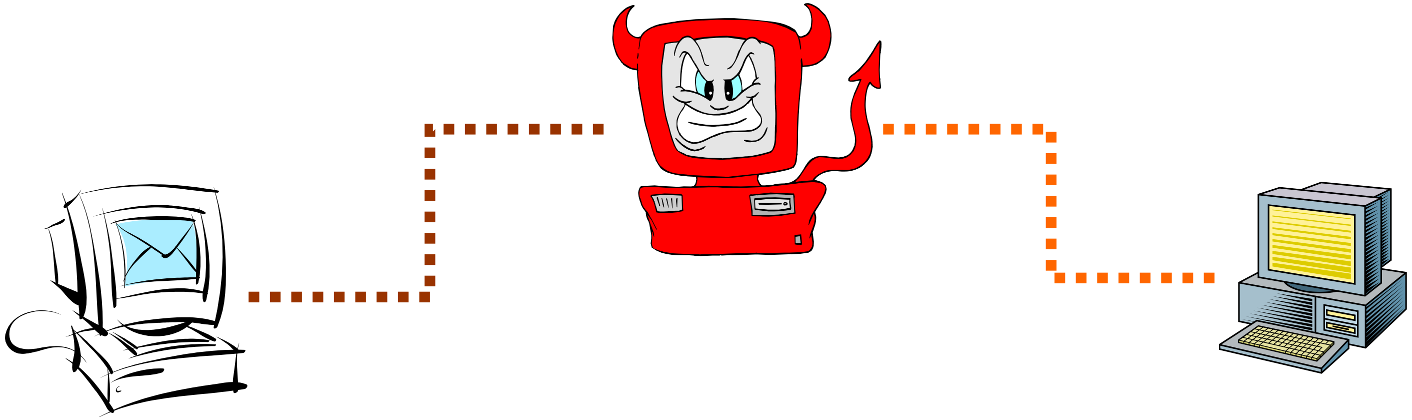
Cryptographic Hash Functions

ITC 3093 Principles of Computer Security

Based on Cryptography and Network Security by William Stallings and Lecture slides by Lawrie Brown

Data Integrity and Source Authentication

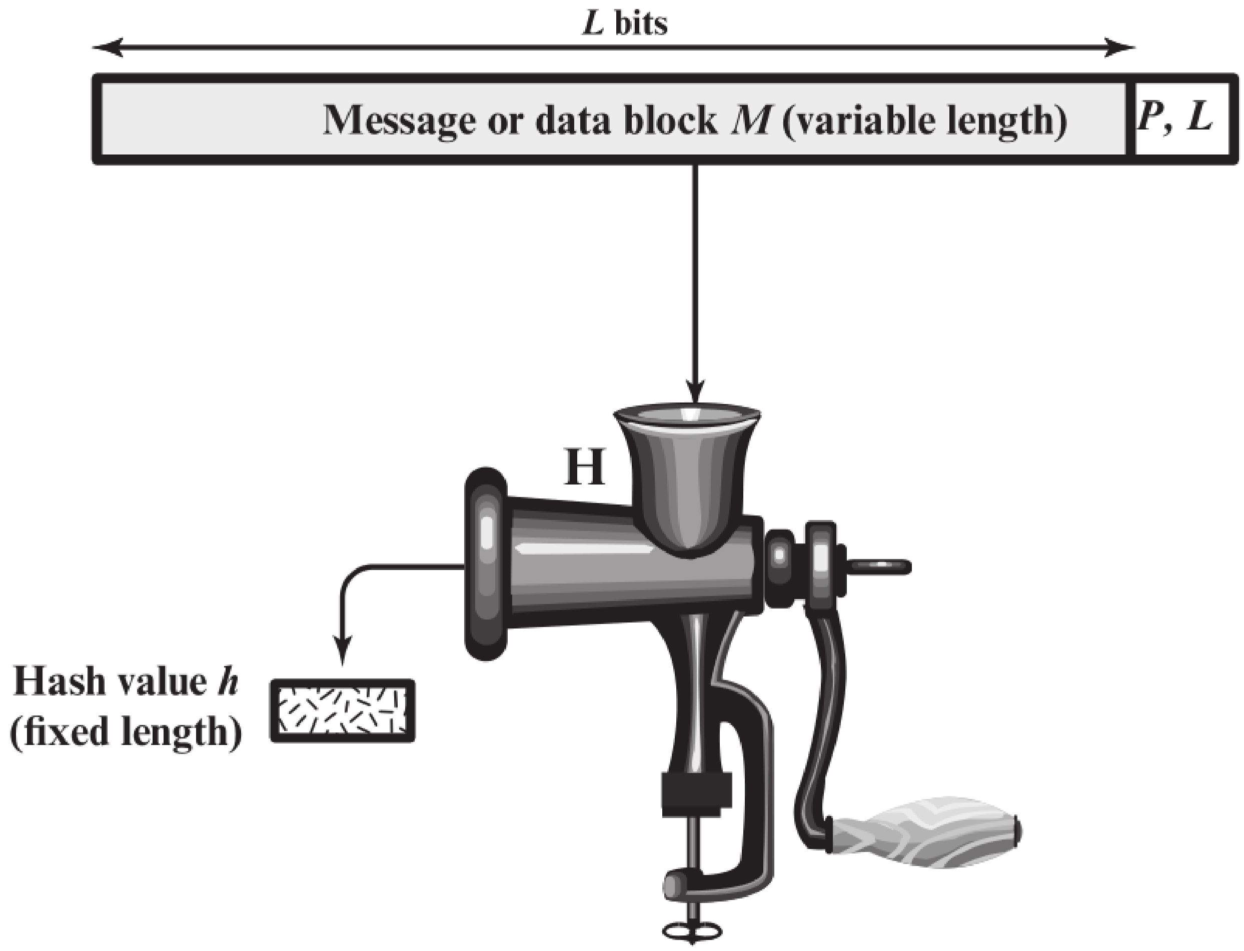
- Encryption does not protect data from modification by another party.
  - Why?
- Need a way to ensure that data arrives at destination in its original form as sent by the sender and it is coming from an authenticated source.



Hash Functions

- condenses arbitrary message to fixed size
  - `h = H(M)`
- usually assume hash function is public
- hash used to detect changes to message
- want a cryptographic hash function
  - computationally infeasible to find data mapping to specific hash (one-way property)
  - computationally infeasible to find two data to same hash (collision-free property)

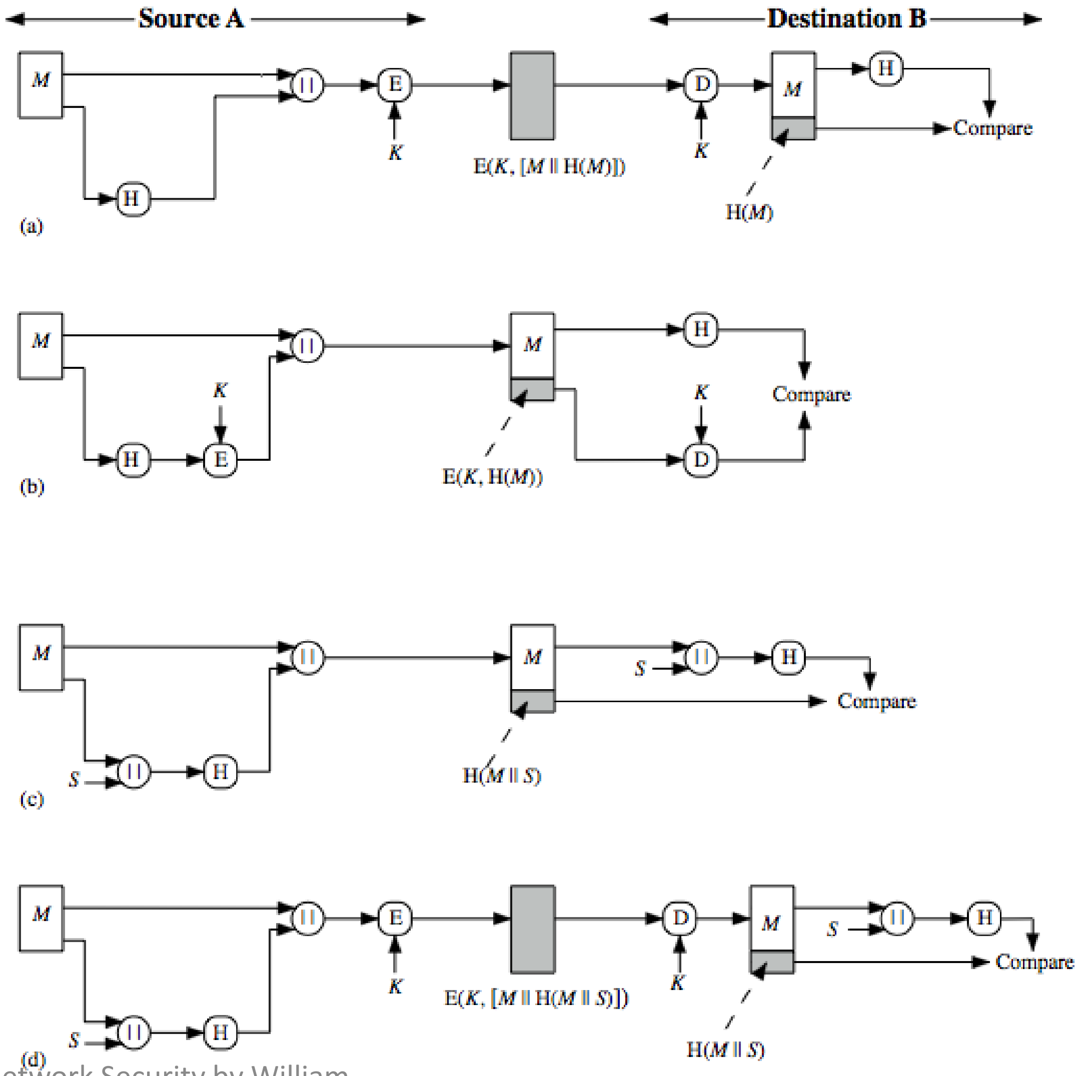
Cryptographic Hash Function



P, L = padding plus length field

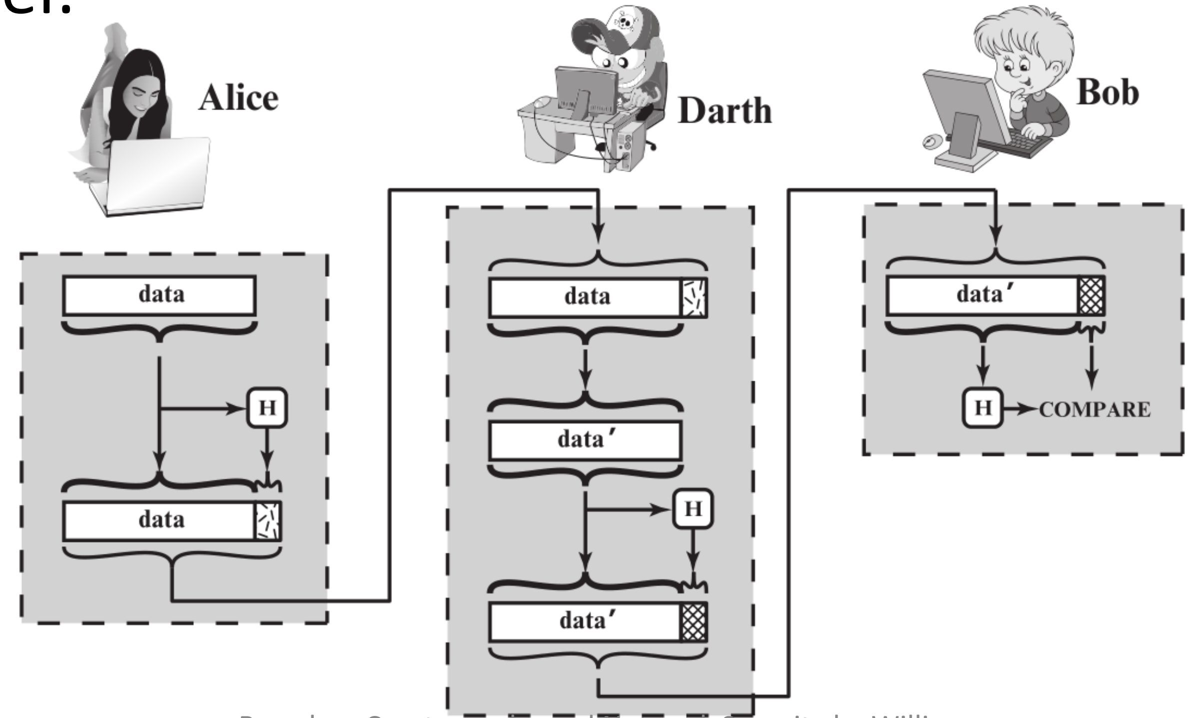
Hash Functions & Message Authentication

- Message authentication is a mechanism or service used to verify the integrity of a message, by assuring that the data received are exactly as sent.
- Some ways in which a hash code can be used to provide message authentication:
    a) The message plus concatenated hash code is encrypted using symmetric encryption.
    b) Only the hash code is encrypted, using symmetric encryption.
    c) Shows the use of a hash function but no encryption for message authentication.
    d) Confidentiality can be added to the approach of (c) by encrypting the entire message plus the hash code.



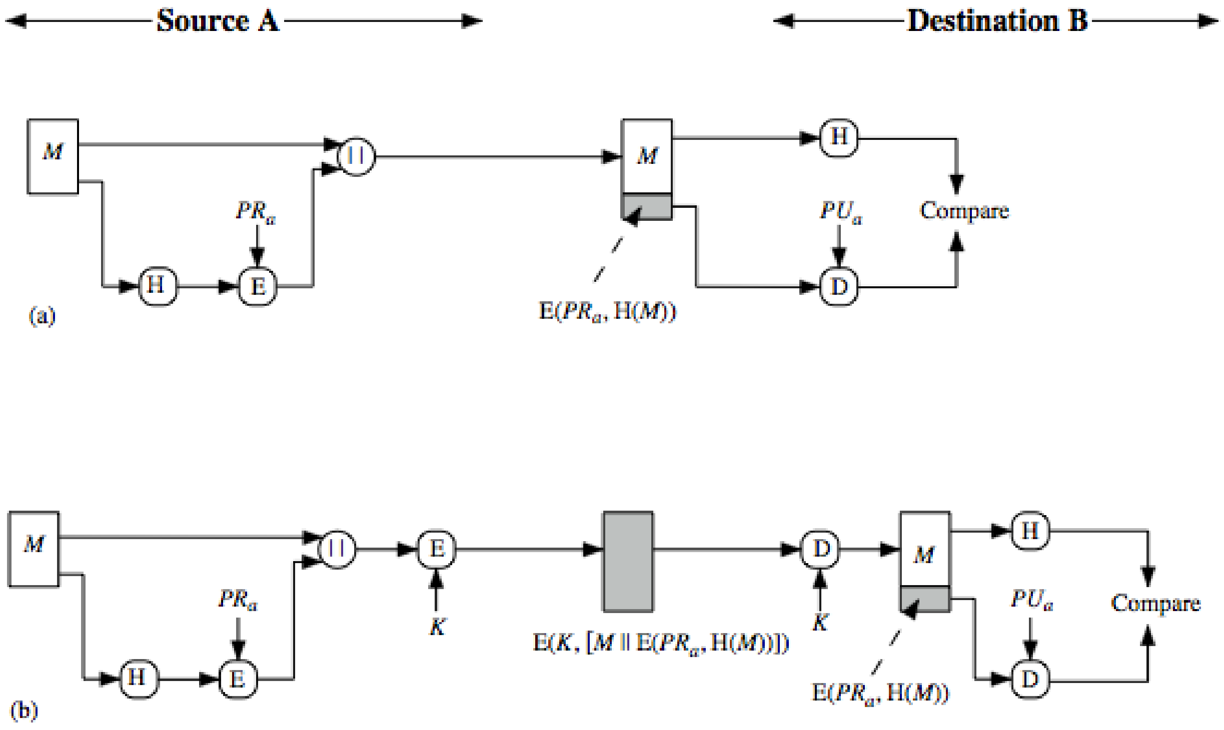
Use of hash function to check data integrity

- The hash value must be transmitted in a secure fashion.
- It should not feasible for an adversary to also alter the hash value to fool the receiver.



Hash Functions & Digital Signatures

- In digital signatures, the hash value of a message is encrypted with a user's private key.
- Anyone who knows the user's public key can verify the integrity of the message that is associated with the digital signature.
- In this case an attacker who wishes to alter the message would need to know the user's private key.



Based on Cryptography and Network Security by William Stallings and Lecture slides by Lawrie Brown

Other Hash Function Uses

- to create a one-way password file
  - store hash of password not actual password
  - the actual password is not retrievable by a hacker who gains access to the password file
  - when a user enters a password, the hash of that password is compared to the stored hash value for verification
- for intrusion detection and virus detection
  - keep & check hash of files on system

Two Simple Insecure Hash Functions

- consider two simple insecure hash functions
- bit-by-bit exclusive-OR (XOR) of every block
  - $C_i = b_{i1} \text{ xor } b_{i2} \text{ xor } . . . \text{ xor } b_{im}$
  - 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) with $x \neq y$, such that $H(x) = H(y)$.
Pseudorandomness	Output of H meets standard tests for pseudorandomness.

Attacks on Hash Functions

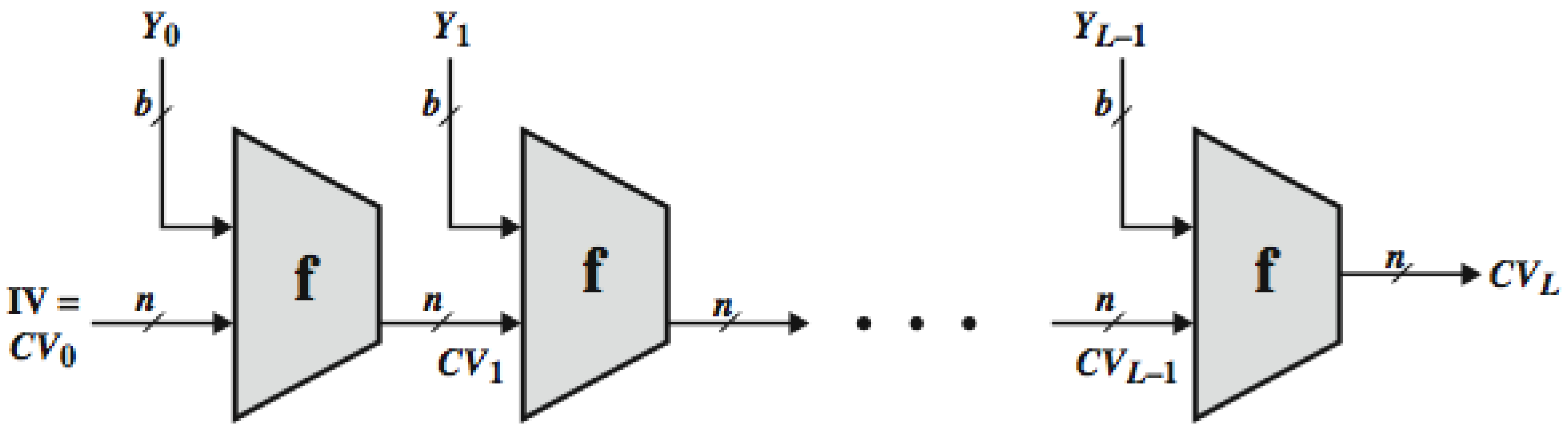
- have brute-force attacks and cryptanalysis
- a preimage or second preimage attack
  - find y s.t. $H(y)$ equals a given hash value
- collision resistance
  - find two messages x & y with same hash so $H(x) = H(y)$
- hence value $2^{m/2}$ determines strength of hash code against brute-force attacks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

Choosing the length of Hash outputs

- The Weakest Link Principle:
  - A system is only as secure as its weakest link.
- Hence all links in a system should have similar levels of security.
- Because of the birthday attack, the length of hash outputs in general should double the key length of block ciphers
  - SHA-224 matches the 112-bit strength of triple-DES (encryption 3 times using DES)
  - SHA-256, SHA-384, SHA-512 match the new key lengths (128,192,256) in AES

Hash Function Cryptanalysis

- cryptanalytic attacks exploit some property of alg so faster than exhaustive search
- hash functions use iterative structure
  - process message in blocks (incl length)
- attacks focus on collisions in function f



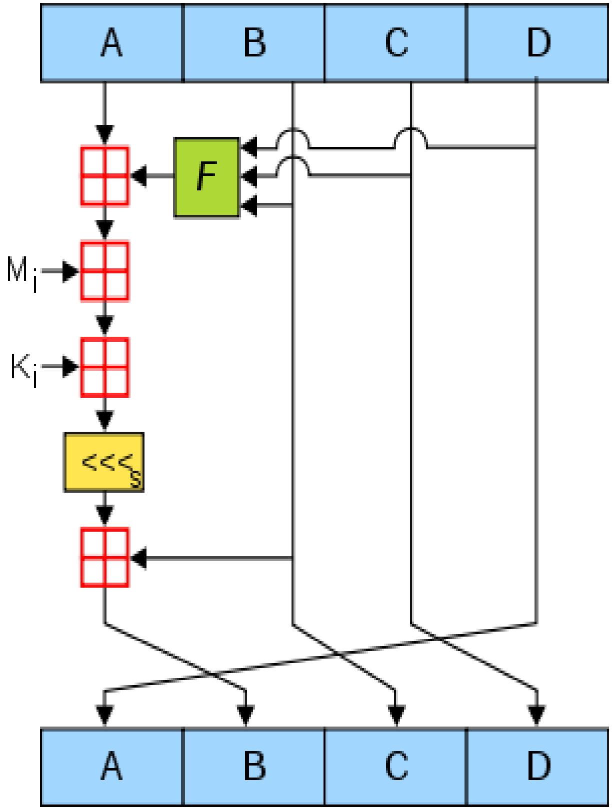
Block Ciphers as Hash Functions

- 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

MD5

- Rivest, 1991
- Based on Davies-Meyer const.
- Very popular until recently.
    - 2004: First collision attacks
    - 2008: Practical collision attack; SSL cert. with same MD5 hash.
    - ~2010: Forged Microsoft MD5 certificates used in Flame malware
- Preimage resistance: Mostly ok.

64 rounds of:



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

SHA Versions

Algorithm	Message Size	Block Size	Word Size	Message Digest Size
SHA-1	$< 2^{64}$	512	32	160
SHA-224	$< 2^{64}$	512	32	224
SHA-256	$< 2^{64}$	512	32	256
SHA-384	$< 2^{128}$	1024	64	384
SHA-512	$< 2^{128}$	1024	64	512
SHA-512/224	$< 2^{128}$	1024	64	224
SHA-512/256	$< 2^{128}$	1024	64	256

Note: All sizes are measured in bits.

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
  - NIST selected Keccak (pronounced “catch-ack”) in October 2012, created by Guido Bertoni, Joan Daemen and Gilles Van Assche, Michaël Peeters

Speed Comparisons

Algorithm	Speed (MiByte/s.)
AES-128 / CTR	198
MD5	335
SHA-1	192
SHA-256	139
SHA-3	~ SHA-256

Crypto++ 5.6 benchmarks, 2.2 GHz AMD Opteron 8354

- NIST expects SHA-2 to be used for the foreseeable future.
- SHA-3: A companion algorithm with a different structure and properties.

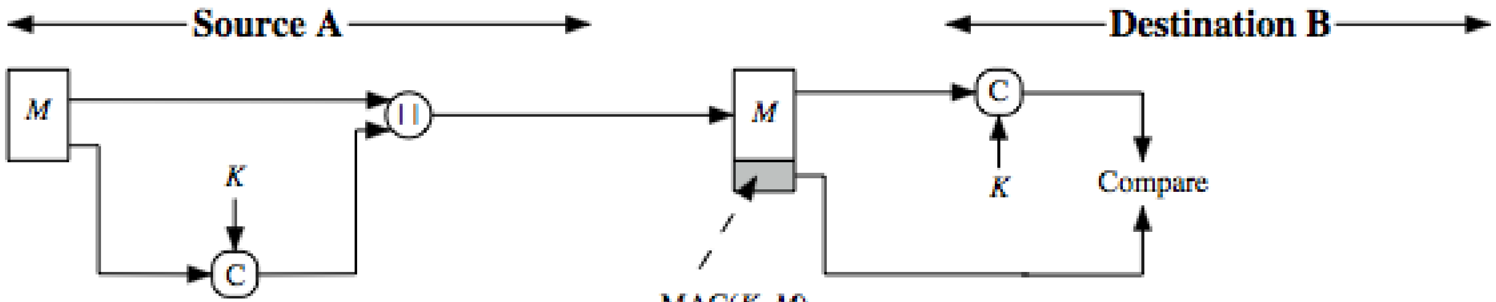
Message Authentication Code (MAC)

Message Authentication Requirements

- Disclosure
  - Release of message contents to any person or process not possessing the appropriate cryptographic key
- Traffic analysis
  - Discovery of the pattern of traffic between parties
- Masquerade
  - Insertion of messages into the network from a fraudulent source
- Content modification
  - Changes to the contents of a message, including insertion, deletion, transposition, and modification
- Sequence modification
  - Any modification to a sequence of messages between parties, including insertion, deletion, and reordering
- Timing modification
  - Delay or replay of messages
- Source repudiation
  - Denial of transmission of message by source
- Destination repudiation
  - Denial of receipt of message by destination

Message Authentication Code

- a small fixed-sized block of data
  - generated from message + secret key
  - MAC = C(K,M)
  - appended to message when sent



(a) Message authentication

Message Authentication Code

- A MAC scheme is a hash family, used for message authentication
- $MAC(K,M) = H_K(M)$
- The sender and the receiver share secret K
- The sender sends $(M, H_k(M))$
- The receiver receives (X,Y) and verifies that $H_K(X)=Y$, if so, then accepts the message as from the sender
- To be secure, an adversary shouldn't be able to come up with (X',Y') such that $H_K(X')=Y'$.

Message Authentication Codes

- as shown the MAC provides authentication
- can also use encryption for secrecy
  - generally use separate keys for each
  - can compute MAC either before or after encryption
  - is generally regarded as better done before
- why use a MAC?
  - sometimes only authentication is needed
  - sometimes need authentication to persist longer than the encryption (eg. archival use)
- note that a MAC is not a digital signature

Requirements for MACs

- Taking into account the types of attacks, the MAC needs to satisfy the following:
- The first requirement deals with message replacement attacks, in which an opponent is able to construct a new message to match a given MAC, even though the opponent does not know and does not learn the key
- The second requirement deals with the need to thwart a brute-force attack based on chosen plaintext
- The final requirement dictates that the authentication algorithm should not be weaker with respect to certain parts or bits of the message than others

MACs Based on Hash Functions: HMAC

- There has been increased interest in developing a MAC derived from a cryptographic hash function
- Motivations:
  - Cryptographic hash functions such as MD5 and SHA generally execute faster in software than symmetric block ciphers such as DES
  - Library code for cryptographic hash functions is widely available
  - HMAC has been chosen as the mandatory-to-implement MAC for IP security
  - Has also been issued as a NIST standard (FIPS 198)

Constructing MAC from Hash Functions

- Let h be a one-way hash function

- MAC(K,M) = h(K || M), where || denote concatenation
  - Insecure as MAC
  - Because of the Merkle-Damgard construction for hash functions, given M and t=h(K || M), adversary can compute M'=M||Pad(M)||X and t', such that h(K||M') = t'

Security of HMAC

- Depends in some way on the cryptographic strength of the underlying hash function
- Appeal of HMAC is that its designers have been able to prove an exact relationship between the strength of the embedded hash function and the strength of HMAC
- Generally expressed in terms of the probability of successful forgery with a given amount of time spent by the forger and a given number of message-tag pairs created with the same key

Authenticated Encryption (AE)

- A term used to describe encryption systems that simultaneously protect confidentiality and authenticity of communications
- Approaches:
  - Hashing followed by encryption
  - Authentication followed by encryption
  - Encryption followed by authentication
  - Independently encrypt and authenticate
- Both decryption and verification are straightforward for each approach
- There are security vulnerabilities with all of these approaches