Cryptography and Network Security

Authentication requirement

Authentication Function

Hash Function

SHA

Session Meta Data

Author	**Dr T Sree Sharmila**
Reviewer	
Version Number	1.0
Release Date	15 July 2018

Revision History

Revision Date	Details	Version no.
		1.0

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

Introduction

- message authentication is concerned with:
  - protecting the integrity of a message
  - validating identity of originator
  - non-repudiation of origin (dispute resolution)
- will consider the security requirements
- then three alternative functions used:
  - hash function (see Ch 11)
  - message encryption
  - message authentication code (MAC)

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

Message Security Requirements

1. disclosure
2. traffic analysis
3. masquerade
4. content modification
5. sequence modification
6. timing modification
7. source repudiation
8. destination repudiation

Message Security Requirements

- The first two requirements (Disclosure: Release of message contents; and Traffic analysis: Discovery of the pattern of traffic between parties) belong in the realm of message confidentiality, and are handled using the encryption techniques already discussed.
- Measures to deal with items 3 through 6 (Masquerade: Insertion of messages into the network from a fraudulent source; Content modification: of the contents of a message; Sequence modification: to a sequence of messages between parties; and Timing modification: Delay or replay of messages) are generally regarded as message authentication.
- Mechanisms for dealing specifically with item 7 (Source repudiation: Denial of transmission of message by source) come under the heading of digital signatures. Generally, a digital signature technique will also counter some or all of the attacks listed under items 3 through 6.
- Dealing with item 8 (Destination repudiation: Denial of receipt of message by destination) may require a combination of the use of digital signatures and a protocol designed to counter this attack.

Message Security Requirements

- In summary, message authentication is a procedure to verify that received messages come from the alleged source and have not been altered.
- Message authentication may also verify sequencing and timeliness.
- A digital signature is an authentication technique that also includes measures to counter repudiation by the source.

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

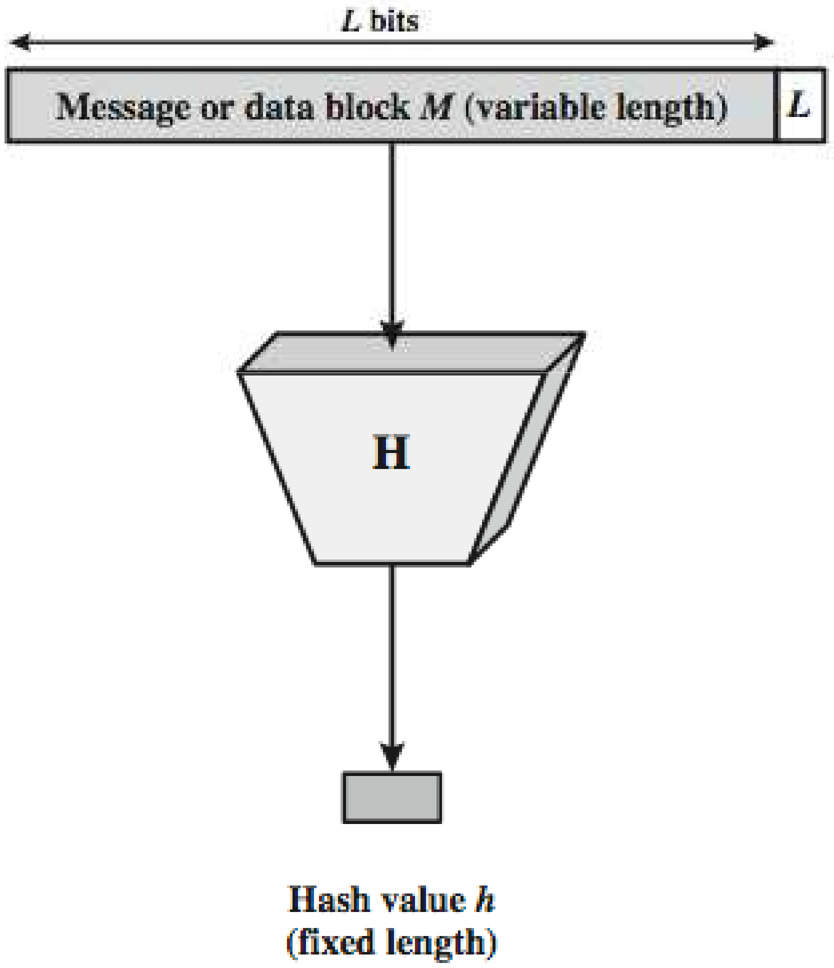
Hash Functions

- condenses arbitrary message to fixed size

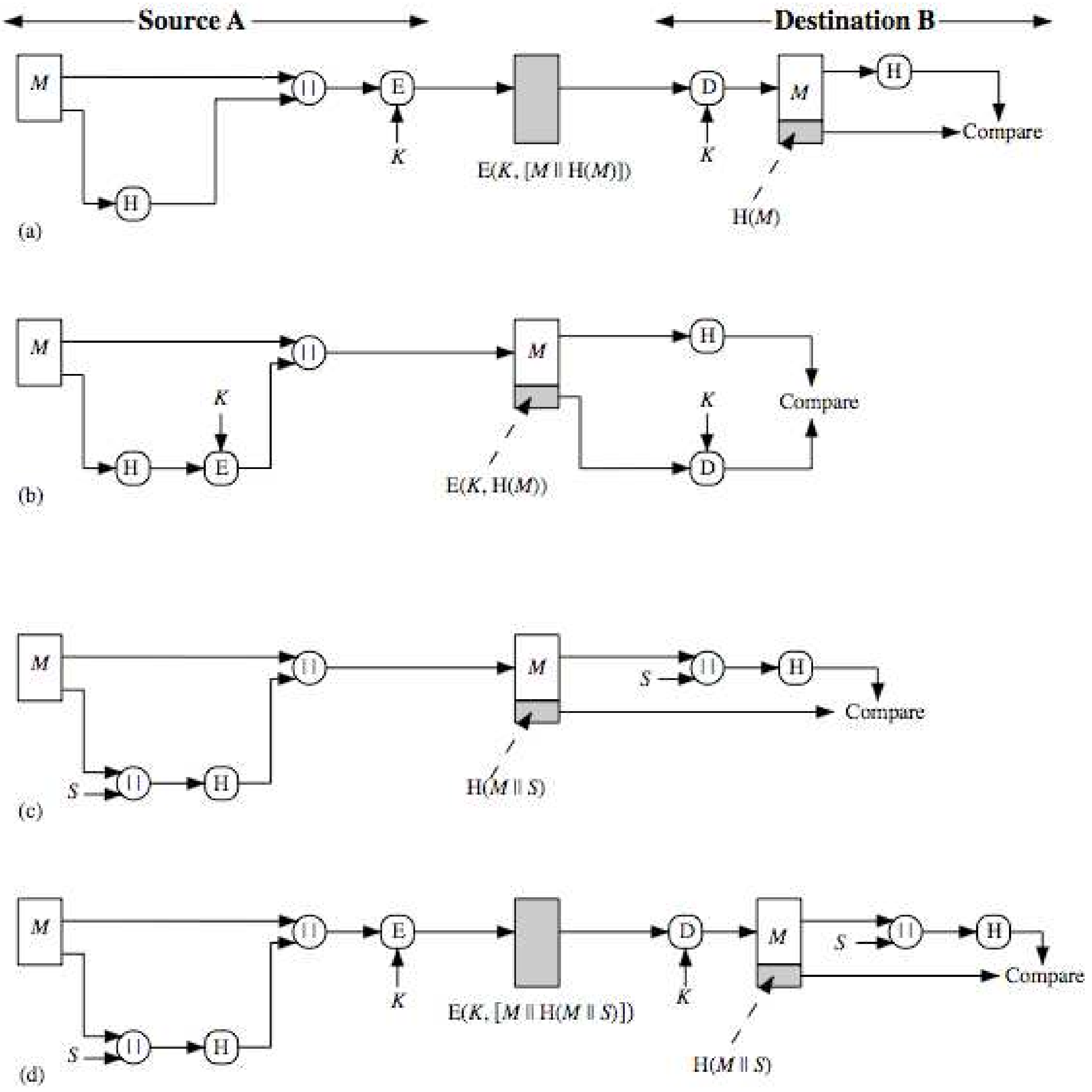
  `h = H(M)`

  - A "good" hash function has the property that the results of applying the function to a large set of inputs will produce outputs that are evenly distributed, and apparently random.
- the objective of a hash function is data integrity
- hash used to detect changes to message ie, to determine whether or not data has changed
- want a cryptographic hash function
  - computationally infeasible to find data mapping to pre-specific hash (one-way property)
  - computationally infeasible to find two data that map to same hash (collision-free property)

Cryptographic Hash Function



Hash Functions & Message Authent-ication



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.

a. The message plus concatenated hash code is encrypted using symmetric encryption. Since only A and B share the secret key, the message must have come from A and has not been altered. The hash code provides the structure or redundancy required to achieve authentication.

b. Only the hash code is encrypted, using symmetric encryption. This reduces the processing burden for those applications not requiring confidentiality.

c. Shows the use of a hash function but no encryption for message authentication. The technique assumes that the two communicating parties share a common secret value S. A computes the hash value over the concatenation of M and S and appends the resulting hash value to M. Because B possesses S, it can recompute the hash value to verify. Because the secret value itself is not sent, an opponent cannot modify an intercepted message and cannot generate a false message.

d. Confidentiality can be added to the approach of (c) by encrypting the entire message plus the hash code.

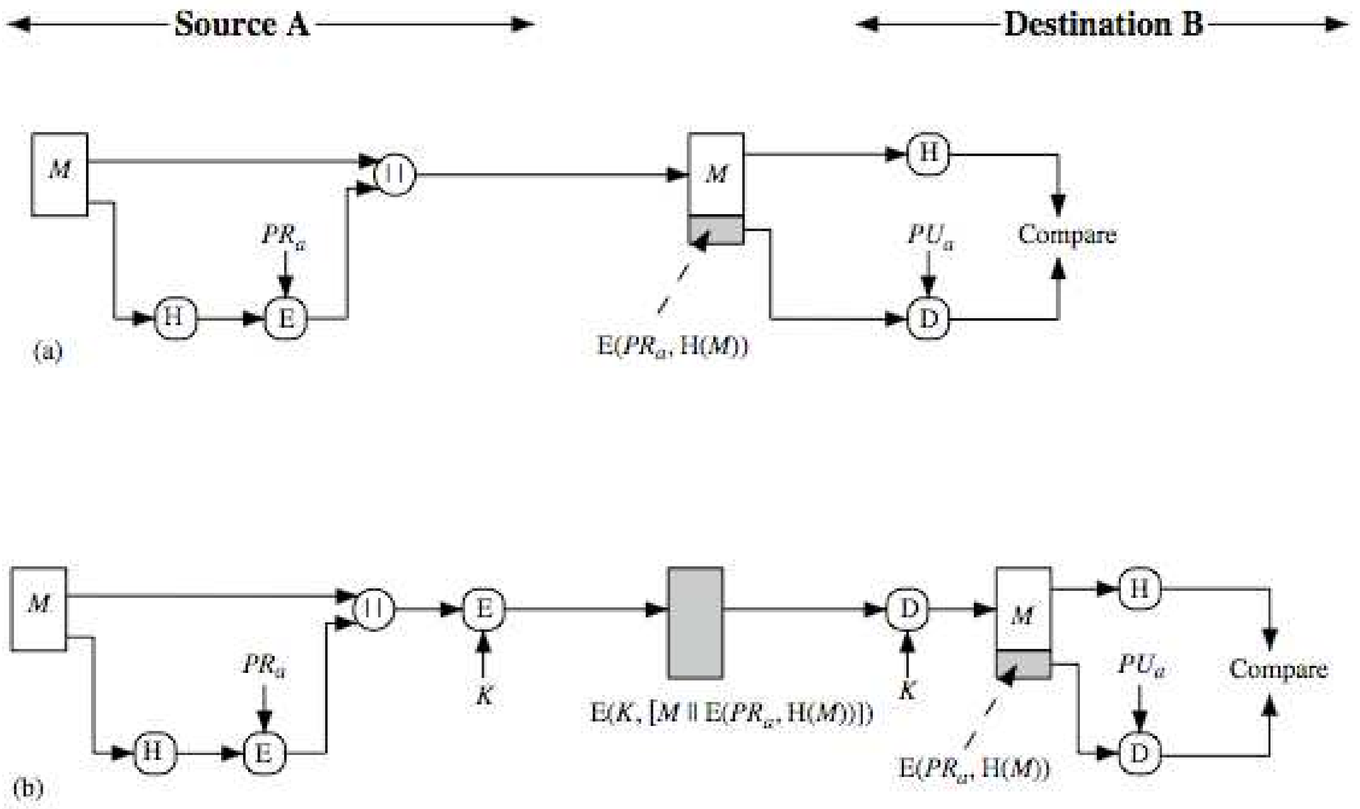
- When confidentiality is not required, method (b) has an advantage over methods (a) and (d), which encrypts the entire message, in that less computation is required.

- Another important application, which is similar to the message authentication application, is the digital signature.
- The operation of the digital signature is similar to that of the MAC.
- In the case of the digital signature, 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.
- Figure illustrates, in a simplified fashion, how a hash code is used to provide a digital signature:

a. The hash code is encrypted, using public-key encryption and using the sender's private key. It also provides a digital signature, because only the sender could have produced the encrypted hash code. In fact, this is the essence of the digital signature technique.

b. If confidentiality as well as a digital signature is desired, then the message plus the private-key-encrypted hash code can be encrypted using a symmetric secret key. This is a common technique.

Hash Functions & Digital Signatures



Other Hash Function Uses

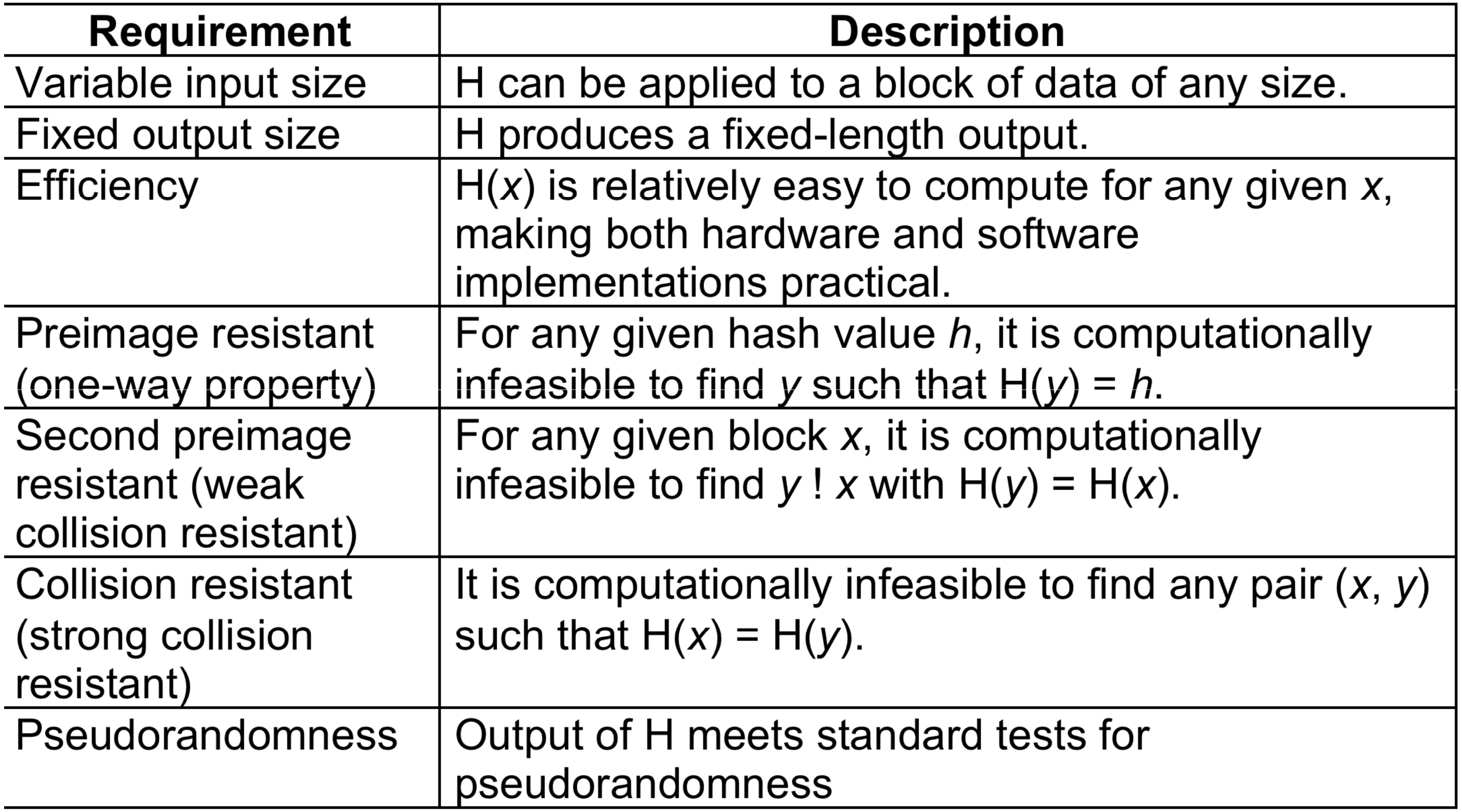
- to create a one-way password file
  - store hash of password not actual password
  - the actual password is not retrievable by a hacker who gains access to the password file
- when a user enters a password, the hash of that password is compared to the stored hash value for verification. This approach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

  - An intruder would need to change F without changing H(F).
- 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
  - $C_i = b_{i1} \text{ xor } b_{i2} \text{ xor } . . . \text{ xor } b_{im}$
  - 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* ! *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 & y with same hash so H(x) = H(y)
- If collision ressistance is required, then the value $2^{m/2}$ determines strength of hash code against brute-force attacks

- Van Oorschot and Wiener presented a design for a $10 million collision search machine for MD5, which has a 128-bit hash length, that could find a collision in 24 days.
- Thus a 128-bit code may be viewed as inadequate.
- The next step up, if a hash code is treated as a sequence of 32 bits, is a 160-bit hash length.
- With a hash length of 160 bits, the same search machine would require over four thousand years to find a collision.
- With today's technology, the time would be much shorter, so that 160 bits now appears suspect.

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

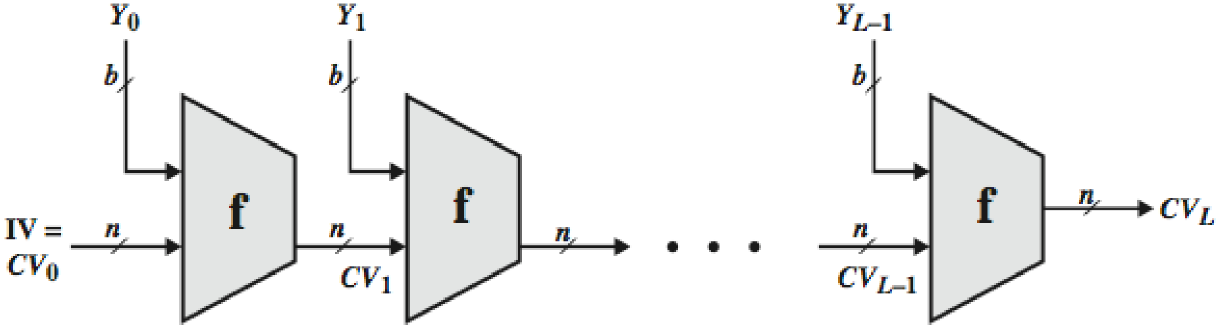
Birthday Attacks

- 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

- The Birthday Attack exploits the birthday paradox – the chance that in a group of people two will share the same birthday – only 23 people are needed for a Pr>0.5 of this. Can generalize the problem to one wanting a matching pair from any two sets, and show need $2^{m/2}$ in each to get a matching m-bit hash.
- Yuval proposed the strategy shown to exploit the birthday paradox in a collision resistant attack. Note that creating many message variants is relatively easy, either by rewording or just varying the amount of white-space in the message. All of which indicates that larger MACs/Hashes are needed.

Hash Function Cryptanalysis

- cryptanalytic attacks exploit some property of algorithm 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

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

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

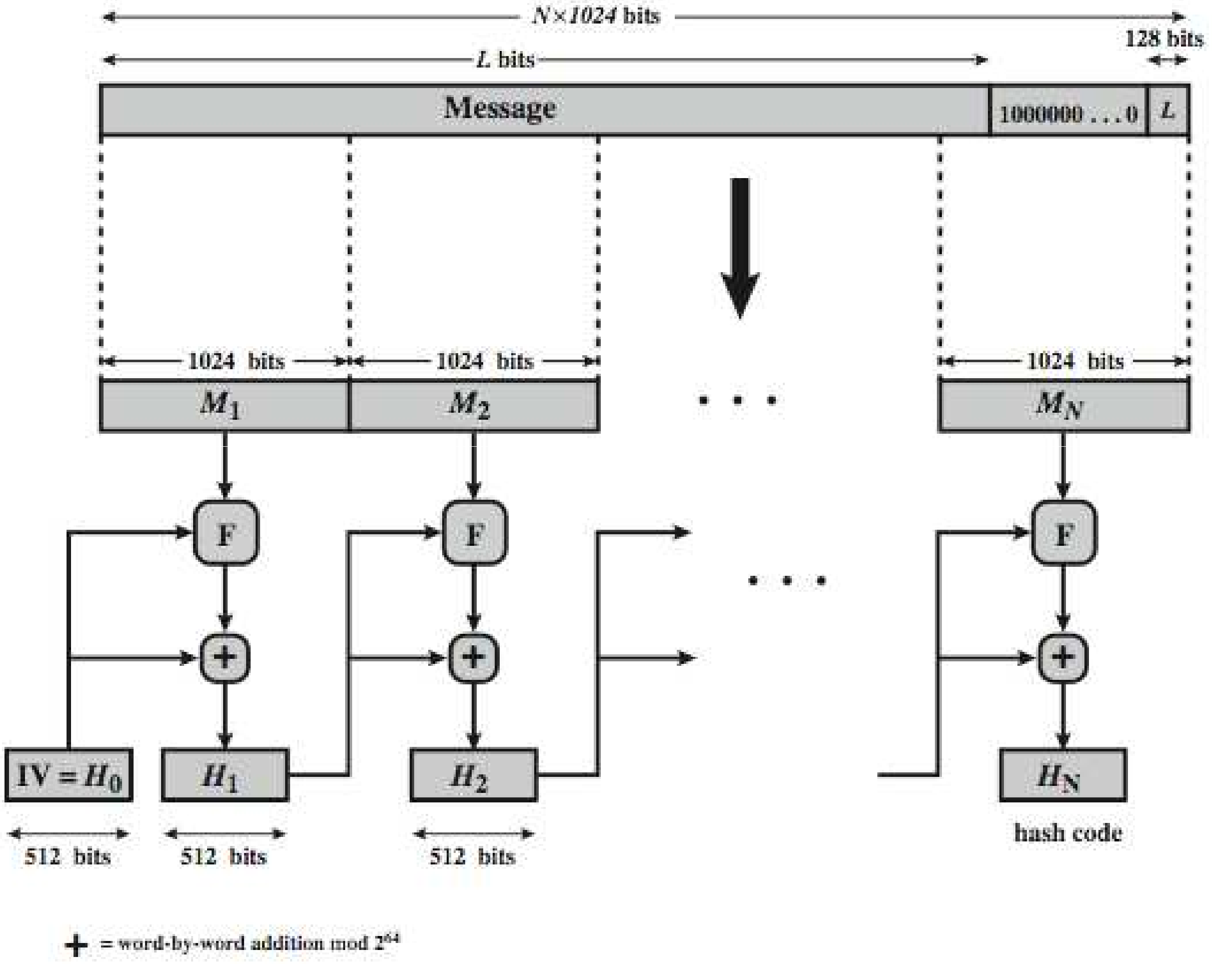
Revised Secure Hash Standard

- 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

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

SHA-512 Overview



SHA-512

- Step 1: Append padding bits
- Step 2: Append length
- Step 3: Initialize hash buffer
- Step 4: Process the message in 1024-bit (128-word) blocks, which forms the heart of the algorithm
- Step 5: Output the final state value as the resulting hash

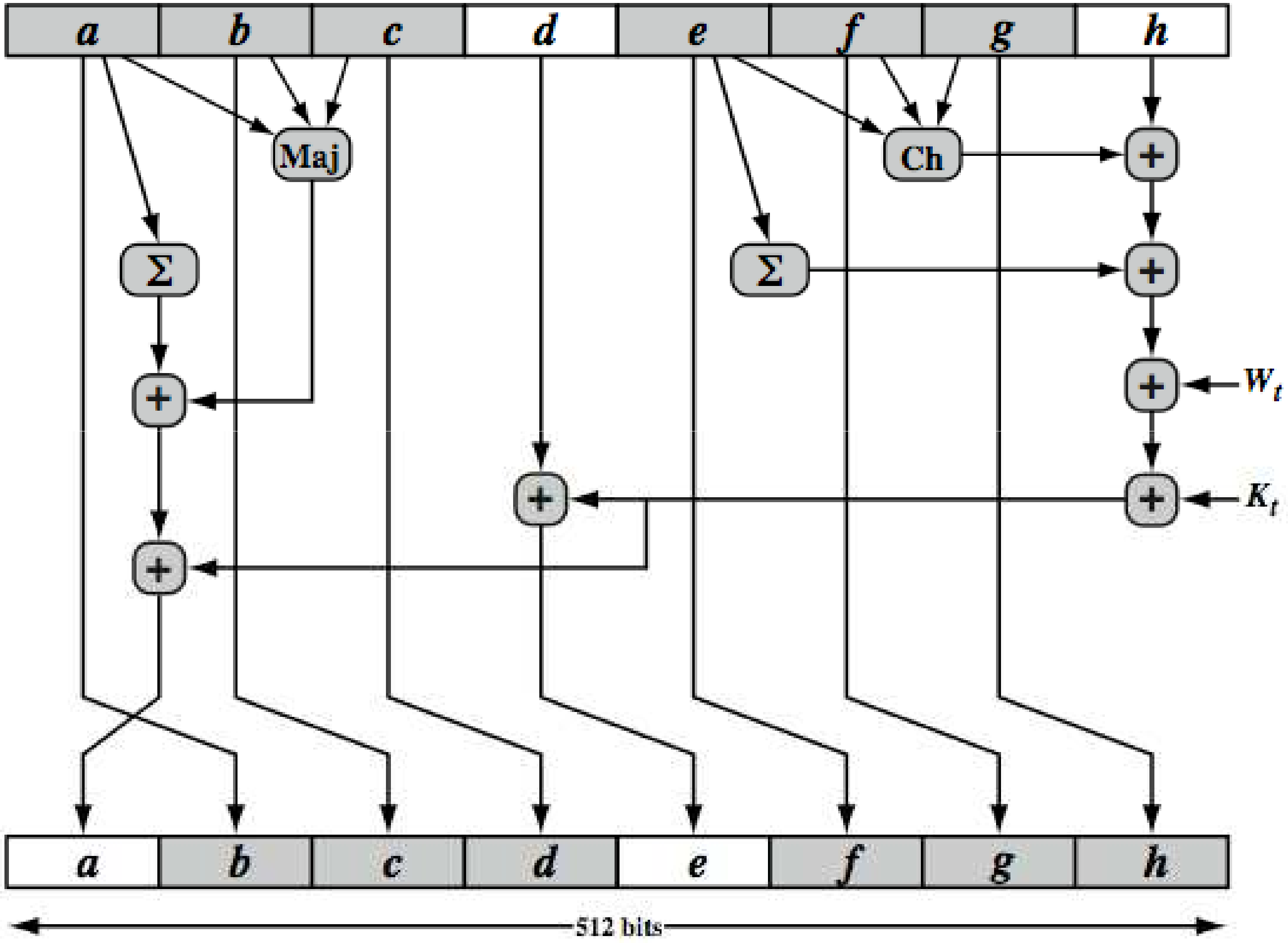
SHA-512 Compression Function

- 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-512 Compression Function

- The SHA-512 Compression Function is the heart of the algorithm.
- In this Step 4, it processes the message in 1024-bit (128-word) blocks, using a module that consists of 80 rounds, labeled F
- Each round takes as input the 512-bit buffer value, and updates the contents of the buffer.
- At input to the first round, the buffer has the value of the intermediate hash value.
- Each round *t* makes use of a 64-bit value *Wt* derived using a message schedule from the current 1024-bit block being processed.
- Each round also makes use of an additive constant *Kt*, based on the fractional parts of the cube roots of the first eighty prime numbers.
- The constants provide a “randomized” set of 64-bit patterns, which should eliminate any regularities in the input data.
- The output of the eightieth round is added to the input to the first round to produce the final hash value for this message block, which forms the input to the next iteration of this compression function.

SHA-512 Round Function



SHA-512 Round Function

- Each 64-bit word is shuffled along one place, and in some cases manipulated using a series of simple logical functions (ANDs, NOTs, ORs, XORs, ROTates), in order to provide the avalanche & completeness properties of the hash function. The elements are:

  Ch(e,f,g) = (e AND f) XOR (NOT e AND g)

  Maj(a,b,c) = (a AND b) XOR (a AND c) XOR (b AND c)

  ∑(a) = ROTR(a,28) XOR ROTR(a,34) XOR ROTR(a,39)

  ∑(e) = ROTR(e,14) XOR ROTR(e,18) XOR ROTR(e,41)

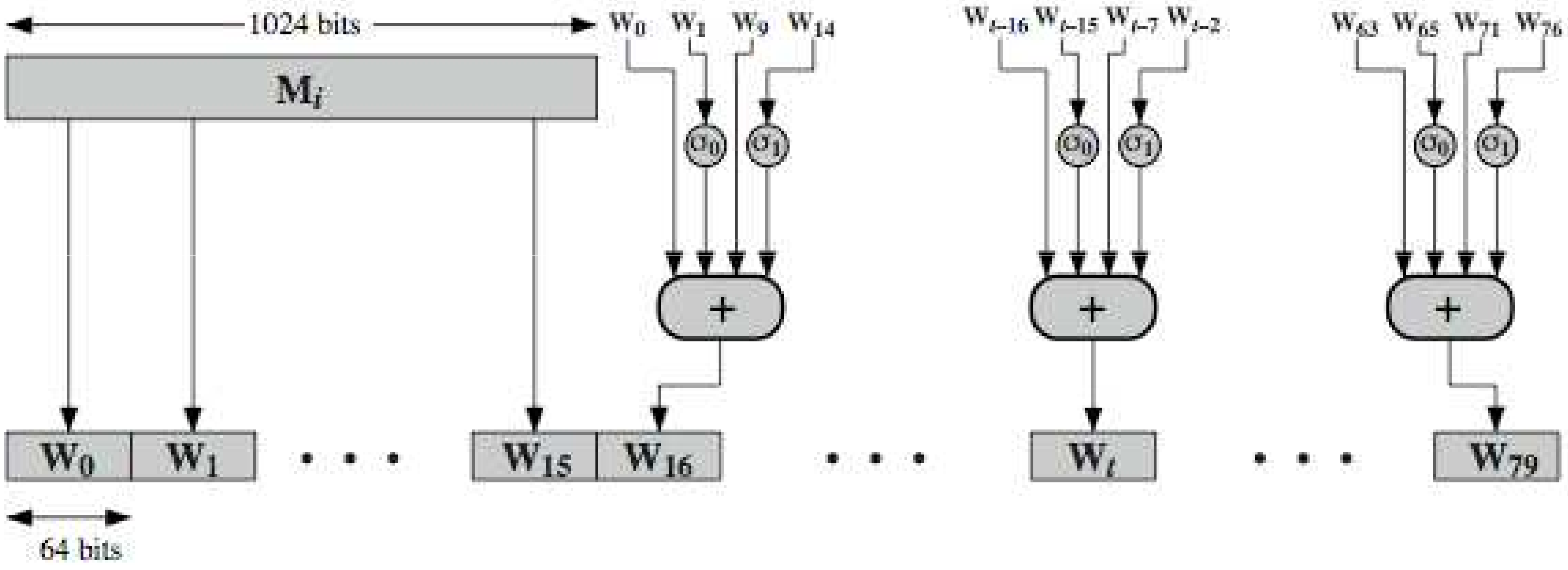
  + = addition modulo 2^64

  Kt = a 64-bit additive constant

  Wt = a 64-bit word derived from the current 512-bit input block.

- Six of the eight words of the output of the round function involve simply permutation (*b, c, d, f, g, h*) by means of rotation.
- Only two of the output words (*a, e)* are generated by substitution.
- Word e is a function of input variables *d, e, f, g, h,* as well as the round word W t and the constant Kt.
- Word a is a function of all of the input variables, as well as the round word W t and the constant Kt.

SHA-512 Round Function


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

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

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
  - goal to have in place by 2012 but not fixed

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

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

Summary

- have considered:
  - hash functions
    - uses, requirements, security
  - hash functions based on block ciphers
  - SHA 512

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

Test your understanding

1) What characteristics are needed in a secure hash function?
2) What is the difference between weak and strong collision resistance?
3) What is the role of a compression function in a hash function?
4) What is the difference between little-endian and big-endian format?
5) Explain in detail SHA 512.

Agenda

- Introduction
- Message Authentication requirement
- Hash function
- Birthday attack
- Secure Hash Algorithm (SHA)
- SHA-512
- SHA-3
- Summary
- Test your understanding
- References

References

1. William Stallings, Cryptography and Network Security, 6th Edition, Pearson Education, March 2013.
2. Charlie Kaufman, Radia Perlman and Mike Speciner, “Network Security”, Prentice Hall of India, 2002.