Hash functions & MACs

Required Reading
W. Stallings, "Cryptography and Network-Security,"

Chapter 11 Cryptographic Hash Functions
Appendix 11A Mathematical Basis of Birthday Attack
Chapter 12 Message Authentication Codes

Recommended Reading
SHA-3 Competition 2007-2012
http://csrc.nist.gov/groups/ST/hash/sha-3

Digital Signature

Alice

Message
Hash function
Hash value
Public key algorithm
Alice's private key

Signature

Bob

Message
Hash function
Hash value
Public key algorithm
Alice's public key

Signature

Hash value 1

yes

Hash value 2

no
Hash function

arbitrary length

message


hash function


hash value

fixed length

Vocabulary

<table>
<thead>
<tr>
<th>hash function</th>
<th>hash value</th>
</tr>
</thead>
<tbody>
<tr>
<td>message digest</td>
<td>message digest</td>
</tr>
<tr>
<td></td>
<td>hash total</td>
</tr>
<tr>
<td></td>
<td>fingerprint</td>
</tr>
<tr>
<td></td>
<td>imprint</td>
</tr>
<tr>
<td></td>
<td>cryptographic checksum</td>
</tr>
<tr>
<td></td>
<td>compressed encoding</td>
</tr>
<tr>
<td></td>
<td>MDC, Message Digest Code</td>
</tr>
</tbody>
</table>

Hash functions

*Basic requirements*

1. Public description, NO key
2. Compression
   - arbitrary length input → fixed length output
3. Ease of computation
Hash functions

Security requirements

It is computationally infeasible

Given | To Find
---|---
1. Preimage resistance | $x$, such that $h(x) = y$
   $y$ | $x$
2. 2nd preimage resistance | $x'$ ≠ $x$, such that $h(x') = h(x) = y$
   $x$ and $y = h(x)$ | $x'$
3. Collision resistance | $x'$ ≠ $x$, such that $h(x') = h(x)$

Dependence between requirements

2nd preimage resistant

collision resistant

Hash functions

(unkeyed)

One-Way Hash Functions

OWHF

preimage resistance

2nd preimage resistance

collision resistance

Collision-Resistant Hash Functions

CRIHF
Given $y$

\[ h(m'_i) = y \]

i=1..2^n

2^n messages with the contents required by the forger

---

Brute force attack against Collision Resistant Hash Function

Yuval

r messages acceptable for the signer

m_i

h

h(m_i)

\[ h(m_i) = h(m'_j) \]

n - bits

r messages required by the forger

m'_j

h

h(m'_j)

\[ h(m'_j) = y \]

\[ n - \text{bits} \]
Creating multiple versions of the required message

I state confirm thereby that I borrowed received

$10,000 (ten thousand dollars) from Mr. Dr. Kris Krzysztof

Gaj on December 2, 2014. This money sum of money

should be returned given back to Mr. Dr. Gaj

by the 16th day of December 2014.

Message acceptable for the signer

I state confirm thereby that on December 2, 2014

I borrowed received from Mr. Dr. Kris Krzysztof a paper manuscript

on security of biometric passports. security of text messaging. This item

should be returned given back to Mr. Dr. Gaj

by the 16th day of December 2014.

Birthday paradox

How many students must be in a class so that there is a greater than 50% chance that

1. one of the students shares the teacher’s birthday (up to the day and month)?

2. any two of the students share the same birthday (up to the day and month)?
Birthday paradox

How many students must be in a class so that there is a greater than 50% chance that

1. one of the students shares the teacher’s birthday (day and month)?
   \[ \approx \frac{366}{2} = 188 \]
2. any two of the students share the same birthday (day and month)?
   \[ \sqrt[3]{366} \approx 19 \]

Brute force attack against Collision Resistant Hash Function

Probability \( p \) that two different messages have the same hash value:

\[
p = 1 - \exp \left( -\frac{r^2}{2^n} \right)
\]

For \( r = 2^{n/2} \) \( p = 63\% \)

J.J. Quisquater

collision search algorithm

Number of operations: \( 2 \sqrt{\pi/2} \cdot 2^{n/2} \approx 2.5 \cdot 2^{n/2} \)

Storage: Negligible
### Hash value size

**One-Way** | **Collision-Resistant**
---|---
Older algorithms:  
$n \geq 64$ & $n \geq 128$  
8 bytes & 16 bytes

Old standards (e.g., SHA-1):  
$n \geq 80$ & $n \geq 160$  
10 bytes & 20 bytes

Current standards (e.g., SHA-2, SHA-3):  
$n = 128, 192, 256$ & $n = 256, 384, 512$  
16, 24, 32 bytes & 32, 48, 64 bytes

---

### Hash function algorithms

<table>
<thead>
<tr>
<th>Customized (dedicated)</th>
<th>Based on block ciphers</th>
<th>Based on modular arithmetic</th>
</tr>
</thead>
</table>
| MD2  
Rivest 1988 | MDC-2  
MDC-4 | MASH-1  
IBM, Brachtel, Meyer, Schilling, 1988 |
| MD4  
Rivest 1990 | SHA-0  
NSA, 1992 | RIPEMD  
European RACE Integrity Primitives Evaluation Project, 1992 |
| MD5  
Rivest 1990 | SHA-1  
NSA, 1995 | RIPEMD-160  
NSA, 2000 |
| SHA-256, SHA-384, SHA-512 |

---

### Attacks against dedicated hash functions
**known by 2004**

MD2

<table>
<thead>
<tr>
<th>partially broken</th>
</tr>
</thead>
</table>

MD4

| broken, H. Dobbertin, 1995  
(one hour on PC; 20 free bytes at the start of the message) |

MD5

| partially broken,  
collisions for the compression function,  
Dobbertin, 1996  
(10 hours on PC) |

| SHA-1  
NSA, 1995  
1998 France  
weakness discovered, 
Dobbertin, 1995  
reduced round version broken,  
Dobbertin 1995 |

RIPEMD

| partially broken |

SHA-256, SHA-384, SHA-512
What was discovered in 2004-2005?

MO4

broken; Wang, Feng, Lai, Yu, Crypto 2004 (manually, without using a computer)

MO5

broken; Wang, Feng, Lai, Yu, Crypto 2004 (1 hr on a PC)

attack with 2^40 operations

Crypto 2004

RIPEMD

broken; Wang, Feng, Lai, Yu, Crypto 2004 (manually, without using a computer)

attack with 2^63 operations

Wang, Yin, Yu, Aug 2005

RIPEMD-160

SHA-256, SHA-384, SHA-512

2^63 operations

Schneier, 2005

In hardware:

Machine similar to the one used to break DES:

Cost = $50,000-$70,000

Time: 18 days

or

Cost = $0.9-$1.26M

Time: 24 hours

In software:

Computer network similar to distributed.net used to break DES (~331,252 computers):

Cost = ~ $0

Time: 7 months

Recommendations of NIST (1)

NIST Brief Comments on Recent Cryptanalytic Attacks on SHA-1

Feb 2005

The new attack is applicable primarily to the use of hash functions in digital signatures.

In many cases applications of digital signatures introduce additional context information, which may make attacks impracticle.

Other applications of hash functions, such as Message Authentication Codes (MACs), are not threatened by the new attacks.
Recommendations of NIST (2)

NIST was already earlier planning to withdraw SHA-1 in favor of SHA-224, SHA-256, SHA-384 & SHA-512 by 2010.

New implementations should use new hash functions.

NIST encourages government agencies to develop plans for gradually moving towards new hash functions, taking into account the sensitivity of the systems when setting the timetables.

---

SHA-3 Contest Timeline

2007
• publication of requirements
• 29.X.2007: request for candidates

2008
• 31.X.2008: deadline for submitting candidates
• 9.XII.2008: announcement of 51 candidates accepted for Round 1

2009
• 25-28.II.2009: 1st SHA-3 Candidate Conference, Leuven, Belgium
• 24.VII.2009: 14 Round 2 candidates announced

2010
• 23-24.VIII.2010: 2nd SHA-3 Candidate Conference, Santa Barbara, CA
• 9.XII.2010: 5 Round 3 candidates announced

2012
• 22-23.III.2012: 3rd SHA-3 Candidate Conference, Washington, D.C.
• 2.X.2012: selection of the winner

2013: draft version of the standard published

---

Number of Submissions

• Number of submissions received by NIST: 64
• Number of submissions publicly available: 56
• Number of submissions qualified to the first round: 51
Basic Requirements for a new hash function

- Must support hash values of 224, 256, 384 and 512 bits
- Available worldwide without licensing fees
- Secure over tens of years
- Suitable for use in:
  - digital signatures FIPS 186
  - message authentication codes, HMAC, FIPS 198
  - key agreement schemes, SP 800-56A
  - random number generators, SP 800-90
- At least the same security level as SHA-2 with increased efficiency

Hash functions

Applications (1)

1. Digital Signatures

   Advantages
   1. Shorter signature
   2. Much faster computations
   3. Larger resistance to manipulation
      (one block instead of several blocks of signature)
   4. Resistance to the multiplicative attacks
   5. Avoids problems with different sizes of the sender and the receiver moduli

Hash functions

Applications (2)

2. Fingerprint of a program or a document
   (e.g., to detect a modification by a virus or an intruder)
3. Storing passwords

Instead of:
ID, password

System stores:
ID, hash(password)

UNIX password scheme

"00000000"

ID, salt, hash(password, salt)

salt modifies the expansion function E of DES

Hash functions
Applications (4)

4. Fast encryption

\[ k_0 = \text{hash}(K_{AB} \ || \ IV) \]
\[ k_i = \text{hash}(K_{AB} \ || \ k_{i-1}) \]
\[ k_n = \text{hash}(K_{AB} \ || \ c_{n-1}) \]

PRNG

m_i \rightarrow k_i \rightarrow c_i
General scheme for constructing a secure hash function

Message m
Padding, appending bit length, M

\[ \begin{align*}
H_0 & \xrightarrow{f} H_1 \\
H_1 & \xrightarrow{f} H_2 \\
& \ldots \\
H_t & \xrightarrow{g} h(m)
\end{align*} \]

compression function
output transformation

Merkle-Damgard Scheme

\[ \begin{align*}
M_0 & \mapsto h_1 \\
M_1 & \mapsto h_2 \\
& \ldots \\
M_{t-1} & \mapsto h_t \\
\text{Entire hash} & = h
\end{align*} \]

Parameters of the Merkle-Damgard Scheme

<table>
<thead>
<tr>
<th>Compression function</th>
<th>SHA-1</th>
<th>SHA-256</th>
<th>SHA-512</th>
</tr>
</thead>
<tbody>
<tr>
<td>( f )</td>
<td>n=160, r=512</td>
<td>n=256, r=512</td>
<td>n=512, r=1024</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Entire hash</th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>( H_0 = IV )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( H_i = f(H_{i-1}, M_i) )</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>( h(m) = g(H_t) )</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>
Sponge Scheme

Hash padding – SHA-1 & SHA-256

<table>
<thead>
<tr>
<th>message</th>
<th>100000000000</th>
</tr>
</thead>
<tbody>
<tr>
<td>length</td>
<td></td>
</tr>
</tbody>
</table>

Length of the entire message in bits

All zero padding: \[ XXX00000 \]

Correct padding: \[ XXX00100 \]

Hash padding – SHA-3 Candidates

<table>
<thead>
<tr>
<th>Method</th>
<th>D</th>
<th>1000</th>
<th>0001</th>
<th>len64</th>
</tr>
</thead>
<tbody>
<tr>
<td>BLAKE256</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Grøstl</td>
<td></td>
<td></td>
<td></td>
<td>#blocks</td>
</tr>
<tr>
<td>JH42</td>
<td></td>
<td></td>
<td></td>
<td>len128</td>
</tr>
<tr>
<td>Keccak</td>
<td></td>
<td></td>
<td></td>
<td>len64</td>
</tr>
<tr>
<td>Skein</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>SHA-2 (256)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

D Data M Minimum Padding P Padding C Counter
## Parameters of new hash functions

### Features affecting security and functionality

<table>
<thead>
<tr>
<th></th>
<th>SHA-1</th>
<th>SHA-256</th>
<th>SHA-384</th>
<th>SHA-512</th>
</tr>
</thead>
<tbody>
<tr>
<td>Size of hash value</td>
<td>160</td>
<td>256</td>
<td>384</td>
<td>512</td>
</tr>
<tr>
<td>Complexity of the birthday attack</td>
<td>$2^{160}$</td>
<td>$2^{128}$</td>
<td>$2^{192}$</td>
<td>$2^{256}$</td>
</tr>
<tr>
<td>Equivalently secure secret-key cipher</td>
<td>Skipjack</td>
<td>AES-128</td>
<td>AES-192</td>
<td>AES-256</td>
</tr>
<tr>
<td>Message size</td>
<td>$&lt; 2^{64}$</td>
<td>$&lt; 2^{64}$</td>
<td>$&lt; 2^{128}$</td>
<td>$&lt; 2^{128}$</td>
</tr>
</tbody>
</table>

### Parameters of new hash functions

### Features affecting implementation speed

<table>
<thead>
<tr>
<th></th>
<th>SHA-1</th>
<th>SHA-256</th>
<th>SHA-384</th>
<th>SHA-512</th>
</tr>
</thead>
<tbody>
<tr>
<td>Message block size</td>
<td>512</td>
<td>512</td>
<td>1024</td>
<td>1024</td>
</tr>
<tr>
<td>Number of digest rounds</td>
<td>80</td>
<td>64</td>
<td>80</td>
<td>80</td>
</tr>
</tbody>
</table>

### Hardware implementations

#### Conceptual comparison

- SHA-1
- SHA-256
- SHA-512, SHA-384

**Speed**

**Area**
### Hash functions

**15 years ago**

- **U.S. Government standards:**
  - SHA-1

- **Other popular hash functions:**
  - MD5, RIPEMD

- **Security status:**
  - MD4 broken (1995)
  - SHA-1 replaced SHA-0 (1995)
  - MD5 partially broken (collisions in compression function, 1996)

**Present**

- **U.S. Government standards:**
  - SHA-1,
  - SHA-224, SHA-256, SHA-384, SHA-512
  - SHA-3 (draft FIPS)

- **Other popular hash functions:**
  - Whirlpool – winner of NESSIE

- **Security status:**
  - MD5 broken (1 hr on PC)
  - SHA-0 broken
  - RIPEMD broken (without a need for computer)
  - SHA-1 practically broken, best attack – \(2^{63}\) operations – only 128 x more than breaking DES

### Hash functions

**Timeline**

- **U.S. Government standards:**
  - SHA-1
  - SHA-224, SHA-256, SHA-384, SHA-512
  - SHA-3 (draft FIPS)

- **Contests:**
  - I. 2000
  - XII. 2003
  - NESSIE
  - SHA-256, SHA-384, SHA-512

- **Attacks:**
  - MD5 – collisions for compression function, 10 yrs on PC
  - VIII. 1998
  - SHA-0 – attack with \(2^{41}\) operations
  - FIFO-2005
  - VIII. 2004
  - attack on SHA-1
  - 2\(^{64}\) operations
MAC - Message Authentication Codes (keyed hash functions)

- **Arbitrary length**
- **Message** \( m \)
- **Secret key** \( K \)
- **MAC function**
- **Fixed length**

**MAC functions**

*Basic requirements*

1. Public description, SECRET key parameter
2. Compression
   - arbitrary length input → fixed length output
3. Ease of computation
MAC functions

Security requirements

Given zero or more pairs
\[ m_i, \text{MAC}_K(m_i) \quad i = 1..k \]

it is computationally impossible to find any new pair
\[ m', \text{MAC}_K(m') \]

Such that
\[ m' \neq m_i \quad i = 1..k \]

MAC functions

Security requirements

Resistance against

1. Known-text attack
2. Chosen-text attack
3. Adaptive chosen-text attack

CBC-MAC (1)
CBC-MAC (1)

\[ H_0 = IV = 0 \]
\[ H_i = DES_K(m_i \oplus H_{i-1}) \quad i = 1..t \]

MAC(m) = \[ H_t[1..32] \]
or
MAC(m) = \[ E_K(E_K^{-1}(H_t))[1..32] \]

MAC functions

- Based on block ciphers
  - CBC-MAC
  - CFB-MAC
  - RIPE-MAC
- Based on hash functions
  - HMAC
  - MD5-MAC
- Dedicated
  - MAA
- Based on stream ciphers
  - CRC-MAC

CMAC

Figure 12.12 Cipher-Based Message Authentication Code (CMAC)
RIPE-MAC

\[ H_0 = IV = 0 \]
\[ H_i = \text{DES}_k(m_i \oplus H_{i-1}) \oplus m_i \quad i = 1..t \]
\[ \text{MAC}(m) = E_k(E_k^{-1}(H_t))[0..31] \]
\[ K' = K \oplus 0xf0f0...f0 \]

HMAC

Bellare, Canetti, Krawczyk, 1996

Used in SSL and IPSec

\[ \text{HMAC}(m) = h(K \oplus \text{ipad} \parallel h(K \oplus \text{opad} \parallel m)) \]

ipad, opad - constant padding strings of the length of the message block size in the hash function h

ipad = repetitions of 0x36 = 00110110
opad = repetitions of 0x5A = 01011010
**Message Authentication Codes - MACs**

**15 years ago**
U.S. Government standards:
- MAC (DAC) based on DES (since 1985)

Other MACs in use:
- RIPE-MAC3, CRC-MAC, MAA

**Present**
U.S. Government standards:
- MAC (DAC) based on DES
- HMAC – based on hash functions used in SSL and IPSec
- CMAC – block cipher mode (AES, Triple DES, Skipjack)

Other MACs in use:
- UMAC, TTMAC, EMAC
- winners of the NESSIE contest

---

**NESSIE: Winners of the contest: 2002**

Message Authentication Codes, MACs

<table>
<thead>
<tr>
<th>Security level</th>
<th>Key size</th>
<th>Output width</th>
</tr>
</thead>
<tbody>
<tr>
<td>high</td>
<td>≥ 256</td>
<td>32-k</td>
</tr>
<tr>
<td>normal</td>
<td>≥ 128</td>
<td>32-k</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. UMAC</td>
<td>UC Davis</td>
</tr>
<tr>
<td>2. TTMAC</td>
<td>K.U. Leuven</td>
</tr>
<tr>
<td>3. EMAC</td>
<td>U. of Toronto</td>
</tr>
<tr>
<td>4. HMAC</td>
<td>NIST &amp; NSA</td>
</tr>
</tbody>
</table>

---

**Message Authentication Codes**

**Timeline**

<table>
<thead>
<tr>
<th>U.S. standards:</th>
<th>V. 2004</th>
</tr>
</thead>
<tbody>
<tr>
<td>MAC (DAC)</td>
<td>FIPS 113 (based on DES)</td>
</tr>
<tr>
<td>HMAC</td>
<td>FIPS 198 (based on hash functions)</td>
</tr>
<tr>
<td></td>
<td>V. 2005</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Contests:</th>
</tr>
</thead>
<tbody>
<tr>
<td>NESSIE</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Attack:</th>
</tr>
</thead>
<tbody>
<tr>
<td>2002 RMAC – practical attack against MAC proposed by NIST and based on Triple DES</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Origin</th>
</tr>
</thead>
<tbody>
<tr>
<td>UMAC, TTMAC, EMAC</td>
<td></td>
</tr>
</tbody>
</table>

---

**Timeline**

<table>
<thead>
<tr>
<th>Year</th>
<th>U.S. standards:</th>
</tr>
</thead>
<tbody>
<tr>
<td>1995</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>1996</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>1997</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>1998</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>1999</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2000</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2001</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2002</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2003</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2004</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2005</td>
<td>MAC (DAC)</td>
</tr>
<tr>
<td>2006</td>
<td>MAC (DAC)</td>
</tr>
</tbody>
</table>
Confidentiality & Authentication
Authenticated Ciphers

Bob

IV Message

K_{AB} - Secret key of Alice and Bob

IV Ciphertext Tag

Authenticated Cipher

Alice

IV Ciphertext Tag

K_{AB} - Secret key of Alice and Bob

IV Message Tag'

Authenticated Cipher

Confidentiality & Authentication
Authenticated Ciphers with Associated Data

Bob

IV AD Message

K_{AB} - Secret key of Alice and Bob

IV AD Ciphertext Tag

Authenticated Cipher

Alice

IV AD Ciphertext Tag

K_{AB} - Secret key of Alice and Bob

IV AD Message Tag'

Authenticated Cipher

Examples of Most Commonly Used Authenticated Ciphers

• AES-GCM
• AES-OCB3
• AES-OCB
• AES-CCM
• AES-EAX