One-Way Encryption and Message Authentication

Cryptographic Hash Functions

Johannes Mittmann
mittmann@in.tum.de

Zentrum Mathematik
Technische Universität München (TUM)

3rd Joint Advanced Student School (JASS)
St. Petersburg, March 30 – April 9, 2005
Course 1: Algorithms for IT Security

Outline

Hash Functions

$$h : \{0,1\}^* \rightarrow \{0,1\}^m \qquad \textit{hash function}$$

- constructs short "fingerprint" of an arbitrarily long message x
- computation of $h(x)$ should be easy
- small change of x should change $h(x)$ completely
- it should be hard to find
  - (second) preimages
  - collisions

Applications of Hash Functions

- verification of data integrity
- authentication of data origin
- optimization of digital signature schemes
- digital timestamping schemes
- hashcash
- password protection
- pseudo-random numbers generation
- ...

Definitions

Definition

A *hash family* is a four-tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, where:

1. $\mathcal{X}$ is a set of possible *messages*
2. $\mathcal{Y}$ is a finite set of possible *message digests* or *authentication tags*
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*
4. for each $K \in \mathcal{K}$, there is a *hash function* $h_K \in \mathcal{H}$, $h_K : \mathcal{X} \rightarrow \mathcal{Y}$.

Definitions

Definition

A *hash family* is a four-tuple $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$, where:

1. $\mathcal{X}$ is a set of possible *messages*
2. $\mathcal{Y}$ is a finite set of possible *message digests* or *authentication tags*
3. $\mathcal{K}$, the *keyspace*, is a finite set of possible *keys*
4. for each $K \in \mathcal{K}$, there is a *hash function* $h_K \in \mathcal{H}$, $h_K : \mathcal{X} \to \mathcal{Y}$.

- $\mathcal{X}$ finite: *compression function*
- $(x, y) \in \mathcal{X} \times \mathcal{Y}$ is *valid pair* under the key $K \iff h_K(x) = y$
- *(N, M)-hash family*, where $N = |\mathcal{X}|, M = |\mathcal{Y}|$

Cryptographic Properties

Preimage Resistance (One-Wayness)

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$ and $y \in \mathcal{Y}$.

Find: $x \in \mathcal{X}$ such that $h(x) = y$.

Cryptographic Properties

Preimage Resistance (One-Wayness)

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$ and $y \in \mathcal{Y}$.

Find: $x \in \mathcal{X}$ such that $h(x) = y$.

Second Preimage Resistance

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$ and $x \in \mathcal{X}$.

Find: $x' \in \mathcal{X}$ such that $x' \neq x$ and $h(x') = h(x)$.

Cryptographic Properties

Preimage Resistance (One-Wayness)

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$ and $y \in \mathcal{Y}$.

Find: $x \in \mathcal{X}$ such that $h(x) = y$.

Second Preimage Resistance

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$ and $x \in \mathcal{X}$.

Find: $x' \in \mathcal{X}$ such that $x' \neq x$ and $h(x') = h(x)$.

Collision Resistance

Instance: hash function $h : \mathcal{X} \to \mathcal{Y}$.

Find: $x, x' \in \mathcal{X}$ such that $x' \neq x$ and $h(x') = h(x)$.

The Random Oracle Model

- mathematical model of an "ideal" hash function
- hash function $h : \mathcal{X} \to \mathcal{Y}$ is chosen randomly from $\mathcal{Y}^{\mathcal{X}}$
- only *oracle* access
- (ϵ, q)-*algorithm*: Las Vegas algorithm with average-case success probability ϵ and at most q oracle queries

The Random Oracle Model

- mathematical model of an "ideal" hash function
- hash function $h : \mathcal{X} \to \mathcal{Y}$ is chosen randomly from $\mathcal{Y}^{\mathcal{X}}$
- only *oracle* access
- (ϵ, q)-*algorithm*: Las Vegas algorithm with average-case success probability ϵ and at most q oracle queries

Theorem (*independence* property)

Let $h \in \mathcal{Y}^{\mathcal{X}}$ be chosen at random, and let $\mathcal{X}_0 \subseteq \mathcal{X}$. Suppose that the values $h(x)$ have been determined iff $x \in \mathcal{X}_0$. Then

$$\Pr[h(x) = y] = \frac{1}{M} \qquad \forall x \in \mathcal{X} \setminus \mathcal{X}_0 \quad \forall y \in \mathcal{Y}.$$

Algorithm for Preimage

Algorithm: FindPreimage(h, y, q)

1: choose $\mathcal{X}_0 \subseteq \mathcal{X}, \quad |\mathcal{X}_0| = q$
2: **for all** $x \in \mathcal{X}_0$ **do**
3: **if** $h(x) = y$ **then return** x
4: **end for**
5: **return** failure

Algorithm for Preimage

Algorithm: FindPreimage(h, y, q)

1: choose $\mathcal{X}_0 \subseteq \mathcal{X}, \quad |\mathcal{X}_0| = q$
2: **for all** $x \in \mathcal{X}_0$ **do**
3: **if** $h(x) = y$ **then return** x
4: **end for**
5: **return** failure

Theorem

For any $\mathcal{X}_0 \subseteq \mathcal{X}$ with $|\mathcal{X}_0| = q$, the average-case success probability of FindPreimage(h, y, q) *is*

$$\epsilon = 1 - \left(1 - \frac{1}{M}\right)^q .$$

Algorithm for Second Preimage

Algorithm: FindSecondPreimage(h, x, q)

1: $y \leftarrow h(x)$
2: choose $\mathcal{X}_0 \subseteq \mathcal{X} \setminus \{x\}, \quad |\mathcal{X}_0| = q - 1$
3: **for all** $x_0 \in \mathcal{X}_0$ **do**
4: **if** $h(x_0) = y$ **then return** x_0
5: **end for**
6: **return** failure

Algorithm for Second Preimage

Algorithm: FindSecondPreimage(h, x, q)

1: $y \leftarrow h(x)$
2: choose $\mathcal{X}_0 \subseteq \mathcal{X} \setminus \{x\}, \quad |\mathcal{X}_0| = q - 1$
3: **for all** $x_0 \in \mathcal{X}_0$ **do**
4: **if** $h(x_0) = y$ **then return** x_0
5: **end for**
6: **return** failure

Theorem

For any $\mathcal{X}_0 \subseteq \mathcal{X} \setminus \{x\}$ with $|\mathcal{X}_0| = q - 1$, the success probability of FindSecondPreimage(h, x, q) *is*

$$\epsilon = 1 - \left(1 - \frac{1}{M}\right)^{q-1}.$$

Random (Second) Preimage Attack

$$1 - \epsilon = \left(1 - \frac{1}{M}\right)^q = \sum_{i=0}^{q} \binom{q}{i} \left(-\frac{1}{M}\right)^i$$
$$\approx 1 - \frac{q}{M}.$$

Random (Second) Preimage Attack

$$
\begin{aligned}
1 - \epsilon = \left(1 - \frac{1}{M}\right)^q &= \sum_{i=0}^{q} \binom{q}{i} \left(-\frac{1}{M}\right)^i \\
&\approx 1 - \frac{q}{M}. \\
q &\approx \epsilon M.
\end{aligned}
$$

Random (Second) Preimage Attack

$$
\begin{aligned}
1 - \epsilon = \left(1 - \frac{1}{M}\right)^q &= \sum_{i=0}^{q} \binom{q}{i} \left(-\frac{1}{M}\right)^i \\
&\approx 1 - \frac{q}{M}. \\
q &\approx \epsilon M.
\end{aligned}
$$

$$
\Longrightarrow \ (\epsilon, \mathcal{O}(M))\text{-algorithm}
$$

Algorithm for Collision

Algorithm: FindCollision(h, q)

1: choose $\mathcal{X}_0 \subseteq \mathcal{X}, \quad |\mathcal{X}_0| = q$
2: **for all** $x \in \mathcal{X}_0$ **do** $y_x \leftarrow h(x)$
3: **if** $y_x = y_{x'}$ for some $x' \neq x$ **then return** (x, x')
4: **else return** failure

Algorithm for Collision

Algorithm: FindCollision(h, q)

1: choose $\mathcal{X}_0 \subseteq \mathcal{X}, \quad |\mathcal{X}_0| = q$
2: **for all** $x \in \mathcal{X}_0$ **do** $y_x \leftarrow h(x)$
3: **if** $y_x = y_{x'}$ for some $x' \neq x$ **then return** (x, x')
4: **else return** failure

Theorem

For any $\mathcal{X}_0 \subseteq \mathcal{X}$ with $|\mathcal{X}_0| = q$, the success probability of Collision(h, q) *is*

$$\epsilon = 1 - \prod_{i=1}^{q-1} \left(1 - \frac{i}{M}\right).$$

Birthday (Collision) Attack

$$1 - \epsilon = \prod_{i=1}^{q-1} \left(1 - \frac{i}{M}\right) \approx \prod_{i=1}^{q-1} \exp\left(-\frac{i}{M}\right) = \exp\left(-\sum_{i=1}^{q-1} \frac{i}{M}\right)$$
$$= \exp\left(-\frac{q(q-1)}{2M}\right) \approx \exp\left(-\frac{q^2}{2M}\right).$$

Birthday (Collision) Attack

$$1 - \epsilon = \prod_{i=1}^{q-1} \left(1 - \frac{i}{M}\right) \approx \prod_{i=1}^{q-1} \exp\left(-\frac{i}{M}\right) = \exp\left(-\sum_{i=1}^{q-1} \frac{i}{M}\right)$$

$$= \exp\left(-\frac{q(q-1)}{2M}\right) \approx \exp\left(-\frac{q^2}{2M}\right).$$

$$q \approx \sqrt{2M \log \frac{1}{1-\epsilon}}.$$

Birthday (Collision) Attack

$$1-\epsilon = \prod_{i=1}^{q-1}\left(1-\frac{i}{M}\right) \approx \prod_{i=1}^{q-1}\exp\left(-\frac{i}{M}\right) = \exp\left(-\sum_{i=1}^{q-1}\frac{i}{M}\right)$$

$$= \exp\left(-\frac{q(q-1)}{2M}\right) \approx \exp\left(-\frac{q^2}{2M}\right).$$

$$q \approx \sqrt{2M\log\frac{1}{1-\epsilon}}.$$

$$\Longrightarrow \ (\epsilon, \mathcal{O}(\sqrt{M}))\text{-algorithm}$$

Birthday (Collision) Attack

$$1-\epsilon = \prod_{i=1}^{q-1}\left(1-\frac{i}{M}\right) \approx \prod_{i=1}^{q-1}\exp\left(-\frac{i}{M}\right) = \exp\left(-\sum_{i=1}^{q-1}\frac{i}{M}\right)$$
$$= \exp\left(-\frac{q(q-1)}{2M}\right) \approx \exp\left(-\frac{q^2}{2M}\right).$$
$$q \approx \sqrt{2M\log\frac{1}{1-\epsilon}}.$$

$$\Longrightarrow \ (\epsilon, \mathcal{O}(\sqrt{M}))\text{-algorithm}$$

Example (Birthday Paradox)

$\epsilon = 0.5,\ M = 365 \qquad \Longrightarrow\ q \approx 1.17\sqrt{M} \approx 22.3.$

Generic Attacks on Cryptographic Hash Functions

attack	algorithm
random (second) preimage	$(\epsilon, \mathcal{O}(M))$
birthday (collision)	$(\epsilon, \mathcal{O}(\sqrt{M}))$

Generic Attacks on Cryptographic Hash Functions

attack	algorithm
random (second) preimage	$(\epsilon, \mathcal{O}(M))$
birthday (collision)	$(\epsilon, \mathcal{O}(\sqrt{M}))$

- attacks are optimal in the random oracle model
- minimum acceptable size of a message digest: 128 bits
- 160-bit message digest (or larger) recommended

Reduce Collision to Second Preimage

Algorithm: CollisionTo2ndPreimage(h)

1: choose $x \in \mathcal{X}$ uniformly at random
2: **if** Oracle2ndPreimage$(h, x) = x'$ **and** $x' \neq x$ **and** $h(x') = h(x)$
then return (x, x')
3: **else return** failure

Reduce Collision to Second Preimage

Algorithm: CollisionTo2ndPreimage(h)

1: choose $x \in \mathcal{X}$ uniformly at random
2: **if** Oracle2ndPreimage$(h, x) = x'$ **and** $x' \neq x$ **and** $h(x') = h(x)$
then return (x, x')
3: **else return** failure

- Oracle2ndPreimage is (ϵ, q)-algorithm for Second Preimage $\Longrightarrow$ CollisionTo2ndPreimage is $(\epsilon, q + 2)$-algorithm for Collision
- collision resistance $\Longrightarrow$ second preimage resistance

Reduce Collision to Preimage

Algorithm: CollisionToPreimage(h)

Require: OraclePreimage is $(1, q)$-algorithm

1: choose $x \in \mathcal{X}$ uniformly at random
2: $y \leftarrow h(x)$
3: **if** OraclePreimage$(h, y) = x'$ **and** $x' \neq x$ **then return** (x, x')
4: **else return** failure

Reduce Collision to Preimage

Algorithm: CollisionToPreimage(h)

Require: OraclePreimage is $(1, q)$-algorithm

1: choose $x \in \mathcal{X}$ uniformly at random
2: $y \leftarrow h(x)$
3: **if** OraclePreimage$(h, y) = x'$ **and** $x' \neq x$ **then return** (x, x')
4: **else return** failure

Theorem

Let $h : \mathcal{X} \to \mathcal{Y}$ be a compression function, where $|\mathcal{X}| \geqslant 2\,|\mathcal{Y}|$. Suppose OraclePreimage *is a $(1, q)$-algorithm that solves Preimage for h. Then* CollisionToPreimage *is a $(1/2, q + 1)$-algorithm for Collision, for the fixed compression function h.*

Proof.

- $x \sim x' \iff h(x) = h(x')$ is equivalence relation on $\mathcal{X}$

Proof.

- $x \sim x' \iff h(x) = h(x')$ is equivalence relation on $\mathcal{X}$
- $\mathcal{C} := \mathcal{X}/\sim$, so $|\mathcal{C}| = |\mathcal{Y}|$

Proof.

- $x \sim x' \iff h(x) = h(x')$ is equivalence relation on $\mathcal{X}$
- $\mathcal{C} := \mathcal{X}/\sim$, so $|\mathcal{C}| = |\mathcal{Y}|$
- for $x \in \mathcal{X}$ the probability of success is $(|[x]| - 1)/|[x]|$

Proof.

- $x \sim x' \iff h(x) = h(x')$ is equivalence relation on $\mathcal{X}$
- $\mathcal{C} := \mathcal{X}/\sim$, so $|\mathcal{C}| = |\mathcal{Y}|$
- for $x \in \mathcal{X}$ the probability of success is $(|[x]| - 1)/|[x]|$
-

$$\begin{aligned}
\Pr[\text{success}] &= \frac{1}{|\mathcal{X}|} \sum_{x \in \mathcal{X}} \frac{|[x]| - 1}{|[x]|} = \frac{1}{|\mathcal{X}|} \sum_{C \in \mathcal{C}} \sum_{x \in C} \frac{|C| - 1}{|C|} \\
&= \frac{1}{|\mathcal{X}|} \sum_{C \in \mathcal{C}} (|C| - 1) = \frac{|\mathcal{X}| - |\mathcal{Y}|}{|\mathcal{X}|} \\
&\geqslant \frac{|\mathcal{X}| - |\mathcal{X}|/2}{|\mathcal{X}|} = \frac{1}{2}.
\end{aligned}$$

□

Iterated Hash Functions

$$\textsc{Compress} : \{0,1\}^{m+t} \longrightarrow \{0,1\}^{m}, \qquad t \geqslant 1.$$

Iterated Hash Functions

$$\textsc{Compress} : \{0,1\}^{m+t} \longrightarrow \{0,1\}^m, \qquad t \geqslant 1.$$

1. **preprocessing**
   input string x, $\quad |x| \geqslant m + t + 1$
   construct
   $$y = y_1 \,\|\, y_2 \,\|\, \cdots \,\|\, y_r, \qquad |y_i| = t$$
   $x \mapsto y(x)$ injection
   $$y = x \,\|\, \textsc{Pad}(x) \qquad \textit{padding function}$$

Design of Iterated Hash Functions

2. **processing**

$$
\begin{aligned}
z_0 &\leftarrow \mathrm{IV}, \qquad |\mathrm{IV}| = m \\
z_1 &\leftarrow \textsc{Compress}(z_0 \,\|\, y_1) \\
&\vdots \\
z_r &\leftarrow \textsc{Compress}(z_{r-1} \,\|\, y_r)
\end{aligned}
$$

Design of Iterated Hash Functions

2. **processing**

$$
\begin{aligned}
z_0 &\leftarrow \mathrm{IV}, \qquad |\mathrm{IV}| = m \\
z_1 &\leftarrow \textsc{Compress}(z_0 \,\|\, y_1) \\
&\vdots \\
z_r &\leftarrow \textsc{Compress}(z_{r-1} \,\|\, y_r)
\end{aligned}
$$

3. **optional output transformation**
   $g : \{0,1\}^m \rightarrow \{0,1\}^\ell$

Design of Iterated Hash Functions

2. **processing**

$$
\begin{aligned}
z_0 &\leftarrow \mathrm{IV}, \qquad |\mathrm{IV}| = m \\
z_1 &\leftarrow \textsc{Compress}(z_0 \,\|\, y_1) \\
&\vdots \\
z_r &\leftarrow \textsc{Compress}(z_{r-1} \,\|\, y_r)
\end{aligned}
$$

3. **optional output transformation**
   $g : \{0,1\}^m \to \{0,1\}^\ell$

$$
h : \bigcup_{i=m+t+1}^{\infty} \{0,1\}^i \to \{0,1\}^\ell, \qquad h(x) = g(z_r).
$$

The Merkle-Damgård Construction

Theorem (Merkle-Damgård)

Suppose $\textsc{Compress} : \{0,1\}^{m+t} \to \{0,1\}^m$ *is a collision resistant compression function, where* $t \geqslant 1$. *Then there exists a collision resistant hash function*

$$h : \bigcup_{i=m+t+1}^{\infty} \{0,1\}^i \to \{0,1\}^m.$$

The number of times $\textsc{Compress}$ *is computed in the evaluation of* h *is at most*

$$\begin{cases} 1 + \left\lceil \frac{n}{t-1} \right\rceil & \text{if } t \geqslant 2, \\ 2n + 2 & \text{if } t = 1, \end{cases}$$

where $|x| = n$.

The Secure Hash Algorithm (SHA-1)

- Ron Rivest: 128-bit hash function MD4 and strengthened version MD5
- design goals:
  - (direct) security
  - speed
  - simplicity and compactness
- collisions for MD4 (Dobbertin) and compression function of MD5 (Boer/Bosselaers)

The Secure Hash Algorithm (SHA-1)

- Ron Rivest: 128-bit hash function MD4 and strengthened version MD5
- design goals:
  - (direct) security
  - speed
  - simplicity and compactness
- collisions for MD4 (Dobbertin) and compression function of MD5 (Boer/Bosselaers)
- NIST & NSA: 160-bit hash function SHA-1
- Feb. 13, 2005: collisions with $< 2^{69}$ hash operations (Wang/Yin/Yu)

Padding Scheme

Algorithm: SHA-1-PAD(x)

Require: $|x| \leqslant 2^{64} - 1$

Ensure: $|y| = 0 \pmod{512}$

1: $d \leftarrow (447 - |x|) \mod 512$

2: $\ell \leftarrow$ the binary representation of $|x|$, where $|\ell| = 64$

3: **return** $y \leftarrow x \,\|\, 1 \,\|\, 0^d \,\|\, \ell$

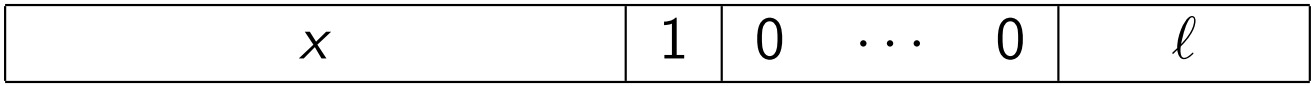

Figure: MD-Strengthening

SHA-1

$$f_t(B,C,D) = \begin{cases} BC \vee (\neg B)D, \\ B \oplus C \oplus D, \\ BC \vee BD \vee CD, \\ B \oplus C \oplus D, \end{cases} \qquad K_t = \begin{cases} \texttt{5A827999}, & \text{if } \ 0 \leqslant t \leqslant 19, \\ \texttt{6ED9EBA1}, & \text{if } 20 \leqslant t \leqslant 39, \\ \texttt{8F1BBCDC}, & \text{if } 40 \leqslant t \leqslant 59, \\ \texttt{CA62C1D6}, & \text{if } 60 \leqslant t \leqslant 79. \end{cases}$$

SHA-1

$$f_t(B,C,D) = \begin{cases} BC \vee (\neg B)D, \\ B \oplus C \oplus D, \\ BC \vee BD \vee CD, \\ B \oplus C \oplus D, \end{cases} \qquad K_t = \begin{cases} \texttt{5A827999}, & \text{if } \ 0 \leqslant t \leqslant 19, \\ \texttt{6ED9EBA1}, & \text{if } 20 \leqslant t \leqslant 39, \\ \texttt{8F1BBCDC}, & \text{if } 40 \leqslant t \leqslant 59, \\ \texttt{CA62C1D6}, & \text{if } 60 \leqslant t \leqslant 79. \end{cases}$$

Cryptosystem: SHA-1(x)

1: $y \leftarrow \text{SHA-1-PAD}(x)$
2: denote $y = M_1 \,\|\, M_2 \,\|\, \cdots \,\|\, M_n$, where each M_i is a 512-bit block
3: $H_0 \leftarrow \texttt{67452301}, \quad H_1 \leftarrow \texttt{EFCDAB89}, \quad H_2 \leftarrow \texttt{98BADCFE}$
4: $H_3 \leftarrow \texttt{10325476}, \quad H_4 \leftarrow \texttt{C3D2E1F0}$
$\ldots$

Cryptosystem: SHA-1(x) (*continued*)

...

1: **for** $i \leftarrow 1$ **to** n **do**
2: denote $M_i = W_0 \| W_1 \| \cdots \| W_{15}$, where each W_i is a word
3: **for** $t \leftarrow 16$ **to** 79 **do** $W_t \leftarrow (W_{t-3} \oplus W_{t-8} \oplus W_{t-14} \oplus W_{t-16}) \lll 1$
4: $A \leftarrow H_0, \quad B \leftarrow H_1, \quad C \leftarrow H_2$
5: $D \leftarrow H_3, \quad E \leftarrow H_4$
6: **for** $t \leftarrow 0$ **to** 79 **do**
7: $temp \leftarrow (A \lll 5) + f_t(B, C, D) + E + W_t + K_t$
8: $E \leftarrow D, \quad D \leftarrow C$
9: $C \leftarrow B \lll 30$
10: $B \leftarrow A, \quad A \leftarrow temp$
11: **end for**
12: $H_0 \leftarrow H_0 + A, \quad H_1 \leftarrow H_1 + B, \quad H_2 \leftarrow H_2 + C$
13: $H_3 \leftarrow H_3 + D, \quad H_4 \leftarrow H_4 + E$
14: **end for**
15: **return** $H_0 \| H_1 \| H_2 \| H_3 \| H_4$

Message Authentication Codes (MACs)

- "MAC = keyed hash function h_K + security properties"
- Alice and Bob share secret key K
- $(x, h_K(x))$ is transmitted over insecure channel

Message Authentication Codes (MACs)

- "MAC = keyed hash function h_K + security properties"
- Alice and Bob share secret key K
- $(x, h_K(x))$ is transmitted over insecure channel

(Existential) Forgery

Instance: valid pairs $(x_1, y_1), \ldots, (x_q, y_q)$ under unknown key K.

Find: valid pair (x, y) such that $x \notin \{x_1, \ldots, x_q\}$.

- (ϵ, q)-*forger*: forgery with worst-case success probability ϵ

Example (Iterated hash function h_K with IV $= K$)

$\textsc{Compress} : \{0,1\}^{m+t} \to \{0,1\}^m$. Given: $(x, h_K(x))$

$y = x \,\|\, \textsc{Pad}(x), \qquad |y| = rt.$

Example (Iterated hash function h_K with IV $= K$)

$\textsc{Compress} : \{0,1\}^{m+t} \to \{0,1\}^m$. Given: $(x, h_K(x))$

$y = x \,\|\, \textsc{Pad}(x), \qquad |y| = rt.$

$$x' = x \,\|\, \textsc{Pad}(x) \,\|\, w.$$
$$y' = x \,\|\, \textsc{Pad}(x) \,\|\, w \,\|\, \textsc{Pad}(x'), \qquad |y'| = r't,\ r' > r.$$

Example (Iterated hash function h_K with IV $= K$)

$\textsc{Compress} : \{0,1\}^{m+t} \to \{0,1\}^m$. Given: $(x, h_K(x))$

$y = x \,\|\, \textsc{Pad}(x), \qquad |y| = rt.$

$$x' = x \,\|\, \textsc{Pad}(x) \,\|\, w.$$
$$y' = x \,\|\, \textsc{Pad}(x) \,\|\, w \,\|\, \textsc{Pad}(x'), \qquad |y'| = r't,\ r' > r.$$

$$z_{r+1} \leftarrow \textsc{Compress}(h_K(x) \,\|\, y_{r+1})$$
$$\vdots$$
$$z_{r'} \leftarrow \textsc{Compress}(z_{r'-1} \,\|\, y_{r'}).$$

$h_K(x') = z_{r'}$.

Example (Iterated hash function h_K with IV $= K$)

$\textsc{Compress} : \{0,1\}^{m+t} \to \{0,1\}^m$. Given: $(x, h_K(x))$

$y = x \,\|\, \textsc{Pad}(x), \qquad |y| = rt.$

$$x' = x \,\|\, \textsc{Pad}(x) \,\|\, w.$$
$$y' = x \,\|\, \textsc{Pad}(x) \,\|\, w \,\|\, \textsc{Pad}(x'), \qquad |y'| = r't,\ r' > r.$$

$$z_{r+1} \leftarrow \textsc{Compress}(h_K(x) \,\|\, y_{r+1})$$
$$\vdots$$
$$z_{r'} \leftarrow \textsc{Compress}(z_{r'-1} \,\|\, y_{r'}).$$

$h_K(x') = z_{r'}$.

$$\Longrightarrow\ (1,1) - \text{forger}$$

Nested MACs

Hash families $(\mathcal{X}, \mathcal{Y}, \mathcal{L}, \mathcal{H})$ and $(\mathcal{Y}, \mathcal{Z}, \mathcal{K}, \mathcal{G})$, $\quad |\mathcal{X}| < |\mathcal{Y}| \leqslant |\mathcal{Z}|$.

$$(\mathcal{X}, \mathcal{Z}, \mathcal{M}, \mathcal{G} \circ \mathcal{H}) \qquad \textit{nested MAC},$$

$$\mathcal{M} = \mathcal{K} \times \mathcal{L}, \quad \mathcal{G} \circ \mathcal{H} = \left\{ (g \circ h)_{(K,L)} \,:\, g_K \in \mathcal{G}, h_L \in \mathcal{H} \right\}.$$

Nested MACs

Hash families $(\mathcal{X}, \mathcal{Y}, \mathcal{L}, \mathcal{H})$ and $(\mathcal{Y}, \mathcal{Z}, \mathcal{K}, \mathcal{G})$, $\quad |\mathcal{X}| < |\mathcal{Y}| \leqslant |\mathcal{Z}|$.

$$(\mathcal{X}, \mathcal{Z}, \mathcal{M}, \mathcal{G} \circ \mathcal{H}) \qquad \textit{nested MAC},$$

$$\mathcal{M} = \mathcal{K} \times \mathcal{L}, \quad \mathcal{G} \circ \mathcal{H} = \left\{ (g \circ h)_{(K,L)} \,:\, g_K \in \mathcal{G}, h_L \in \mathcal{H} \right\}.$$

Theorem

Let $(\mathcal{X}, \mathcal{Z}, \mathcal{M}, \mathcal{G} \circ \mathcal{H})$ be a nested MAC. Suppose there does not exist an $(\epsilon_1, q+1)$-collision attack for a $h_L \in \mathcal{H}$ (L secret), and there does not exist an (ϵ_2, q)-forger for a $g_K \in \mathcal{G}$. Further suppose there exists an (ϵ, q)-forger for $(g \circ h)_{(K,L)} \in \mathcal{G} \circ \mathcal{H}$. Then

$$\epsilon \leqslant \epsilon_1 + \epsilon_2.$$

HMAC

$$\begin{aligned} ipad &= 3636\cdots36, \\ opad &= 5C5C\cdots5C. \end{aligned} \qquad \text{(512-bit)}$$

512-bit key K.

Cryptosystem: $\mathrm{HMAC}(x, K)$

$$\mathrm{HMAC}_K(x) = \text{SHA-1}((K \oplus opad) \,\|\, \text{SHA-1}((K \oplus ipad) \,\|\, x))$$

CBC-MAC

$(\mathcal{P}, \mathcal{C}, \mathcal{K}, \mathcal{E}, \mathcal{D})$ endomorphic cryptosystem, $\mathcal{P} = \mathcal{C} = \{0,1\}^t$.

$K \in \mathcal{K}$, $e_K \in \mathcal{E}$.

Cryptosystem: $\mathrm{CBC\text{-}MAC}(x, K)$

1: denote $x = x_1 \| \cdots \| x_n$, $|x_i| = t$
2: $y_0 \leftarrow 00 \cdots 0$ ▷ initial value
3: **for** $i \leftarrow 1$ **to** n **do** $y_i \leftarrow e_K(y_{i-1} \oplus x_i)$
4: **return** y_n

Unconditionally Secure MACs

- a key is used to produce only *one* authentication tag
- *impersonation*: $(\epsilon, 0)$-forger
- *substitution*: $(\epsilon, 1)$-forger
- *deception probability* Pd_q: maximum value of ϵ such that (ϵ, q)-forger exists $(q = 0, 1)$

Example

$$\mathcal{X} = \mathcal{Y} = \mathbb{Z}_3, \quad \mathcal{K} = \mathbb{Z}_3 \times \mathbb{Z}_3,$$
$$\mathcal{H} = \left\{ h_{(a,b)} : \ (a,b) \in \mathcal{K} \right\}$$

$$h_{(a,b)}(x) = ax + b \pmod 3.$$

Example

$$\mathcal{X} = \mathcal{Y} = \mathbb{Z}_3, \quad \mathcal{K} = \mathbb{Z}_3 \times \mathbb{Z}_3,$$
$$\mathcal{H} = \left\{ h_{(a,b)} : (a,b) \in \mathcal{K} \right\}$$

$$h_{(a,b)}(x) = ax + b \pmod 3.$$

key	0	1	2
$(0,0)$	0	0	0
$(0,1)$	1	1	1
$(0,2)$	2	2	2
$(1,0)$	0	1	2
$(1,1)$	1	2	0
$(1,2)$	2	0	1
$(2,0)$	0	2	1
$(2,1)$	1	0	2
$(2,2)$	2	1	0

Figure: Authentication Matrix

Example

$$\mathcal{X} = \mathcal{Y} = \mathbb{Z}_3, \quad \mathcal{K} = \mathbb{Z}_3 \times \mathbb{Z}_3,$$
$$\mathcal{H} = \left\{ h_{(a,b)} : \ (a,b) \in \mathcal{K} \right\}$$

$$h_{(a,b)}(x) = ax + b \pmod 3.$$

$$\Longrightarrow \ Pd_0 = Pd_1 = \tfrac{1}{3}$$

key	0	1	2
$(0,0)$	0	0	0
$(0,1)$	1	1	1
$(0,2)$	2	2	2
$(1,0)$	0	1	2
$(1,1)$	1	2	0
$(1,2)$	2	0	1
$(2,0)$	0	2	1
$(2,1)$	1	0	2
$(2,2)$	2	1	0

Figure: Authentication Matrix

Computation of Deception Probabilities

$$\text{payoff}(x, y) = \Pr\left[y = h_{K_0}(x)\right] = \frac{\left|\{K \in \mathcal{K} : \ h_K(x) = y\}\right|}{|\mathcal{K}|}.$$

Computation of Deception Probabilities

$$\text{payoff}(x, y) = \Pr\left[y = h_{K_0}(x)\right] = \frac{\left|\{K \in \mathcal{K} : \ h_K(x) = y\}\right|}{|\mathcal{K}|}.$$

$$Pd_0 = \max\left\{\text{payoff}(x, y) : \ x \in \mathcal{X}, \ y \in \mathcal{Y}\right\}.$$

Computation of Deception Probabilities

$$\text{payoff}(x, y) = \Pr\left[y = h_{K_0}(x)\right] = \frac{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}{|\mathcal{K}|}.$$

$$Pd_0 = \max\left\{\text{payoff}(x, y) : \ x \in \mathcal{X}, \ y \in \mathcal{Y}\right\}.$$

$$\begin{aligned}
\text{payoff}(x', y';\, x, y) &= \Pr\left[y' = h_{K_0}(x') \,|\, y = h_{K_0}(x)\right] \\
&= \frac{\Pr\left[y' = h_{K_0}(x') \wedge y = h_{K_0}(x)\right]}{\Pr\left[y = h_{K_0}(x)\right]} \\
&= \frac{|\{K \in \mathcal{K} : \ h_K(x') = y', h_K(x) = y\}|}{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}.
\end{aligned}$$

Computation of Deception Probabilities

$$\text{payoff}(x, y) = \Pr\left[y = h_{K_0}(x)\right] = \frac{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}{|\mathcal{K}|}.$$

$$Pd_0 = \max\left\{\text{payoff}(x, y) : \ x \in \mathcal{X}, \ y \in \mathcal{Y}\right\}.$$

$$\begin{aligned}\text{payoff}(x', y';\, x, y) &= \Pr\left[y' = h_{K_0}(x') \,|\, y = h_{K_0}(x)\right] \\ &= \frac{\Pr\left[y' = h_{K_0}(x') \wedge y = h_{K_0}(x)\right]}{\Pr\left[y = h_{K_0}(x)\right]} \\ &= \frac{|\{K \in \mathcal{K} : \ h_K(x') = y', h_K(x) = y\}|}{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}.\end{aligned}$$

$$\mathcal{V} = \{(x, y) : \ |\{K \in \mathcal{K} : \ h_K(x) = y\}| \geqslant 1\}.$$

Computation of Deception Probabilities

$$\text{payoff}(x, y) = \Pr\left[y = h_{K_0}(x)\right] = \frac{\left|\{K \in \mathcal{K} :\ h_K(x) = y\}\right|}{|\mathcal{K}|}.$$

$$Pd_0 = \max\left\{\text{payoff}(x, y) :\ x \in \mathcal{X},\ y \in \mathcal{Y}\right\}.$$

$$\begin{aligned}\text{payoff}(x', y';\, x, y) &= \Pr\left[y' = h_{K_0}(x') \,|\, y = h_{K_0}(x)\right] \\ &= \frac{\Pr\left[y' = h_{K_0}(x') \wedge y = h_{K_0}(x)\right]}{\Pr\left[y = h_{K_0}(x)\right]} \\ &= \frac{\left|\{K \in \mathcal{K} :\ h_K(x') = y', h_K(x) = y\}\right|}{\left|\{K \in \mathcal{K} :\ h_K(x) = y\}\right|}.\end{aligned}$$

$$\mathcal{V} = \left\{(x, y) :\ \left|\{K \in \mathcal{K} :\ h_K(x) = y\}\right| \geqslant 1\right\}.$$

$$Pd_1 = \max\left\{\text{payoff}(x', y';\, x, y) :\ x \neq x' \in \mathcal{X},\ y, y' \in \mathcal{Y},\ (x, y) \in \mathcal{V}\right\}.$$

Strongly Universal Hash Families

Definition

Let $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ be an (N, M)-hash family. This hash family is *strongly universal*, if

$$\left|\{K \in \mathcal{K} : \ h_K(x) = y, \ h_K(x') = y'\}\right| = \frac{|\mathcal{K}|}{M^2}$$

for all $x, x' \in \mathcal{X}$ such that $x \neq x'$, and for all $y, y' \in \mathcal{Y}$.

Strongly Universal Hash Families

Definition

Let $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ be an (N, M)-hash family. This hash family is *strongly universal*, if

$$\left|\{K \in \mathcal{K} : \ h_K(x) = y, \ h_K(x') = y'\}\right| = \frac{|\mathcal{K}|}{M^2}$$

for all $x, x' \in \mathcal{X}$ such that $x \neq x'$, and for all $y, y' \in \mathcal{Y}$.

Lemma

Let $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ be a strongly universal (N, M)-hash family. Then

$$|\{K \in \mathcal{K} : \ h_K(x) = y\}| = \frac{|\mathcal{K}|}{M} \qquad \forall x \in \mathcal{X} \quad \forall y \in \mathcal{Y}.$$

Proof.

Let $x, x' \in \mathcal{X}$ such that $x \neq x'$, and let $y \in \mathcal{Y}$.

$$|\{K \in \mathcal{K} : h_K(x) = y\}| = \sum_{y' \in \mathcal{Y}} |\{K \in \mathcal{K} : h_K(x) = y, h_K(x') = y'\}|$$

$$= \sum_{y' \in \mathcal{Y}} \frac{|\mathcal{K}|}{M^2} = \frac{|\mathcal{K}|}{M}.$$

□

Proof.

Let $x, x' \in \mathcal{X}$ such that $x \neq x'$, and let $y \in \mathcal{Y}$.

$$|\{K \in \mathcal{K} : h_K(x) = y\}| = \sum_{y' \in \mathcal{Y}} |\{K \in \mathcal{K} : h_K(x) = y, h_K(x') = y'\}|$$
$$= \sum_{y' \in \mathcal{Y}} \frac{|\mathcal{K}|}{M^2} = \frac{|\mathcal{K}|}{M}.$$

□

Theorem

Let $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ be a strongly universal (N, M)-hash family. Then $(\mathcal{X}, \mathcal{Y}, \mathcal{K}, \mathcal{H})$ is an authentication code with

$$Pd_0 = Pd_1 = \frac{1}{M}.$$

Proof.

Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

$$\text{payoff}(x, y) = \frac{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}{|\mathcal{K}|} = \frac{|\mathcal{K}| / M}{|\mathcal{K}|} = \frac{1}{M}.$$

Proof.

Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

$$\text{payoff}(x, y) = \frac{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}{|\mathcal{K}|} = \frac{|\mathcal{K}|/M}{|\mathcal{K}|} = \frac{1}{M}.$$

Now let $x, x' \in \mathcal{X}$ such that $x \neq x'$, and let $y, y' \in \mathcal{Y}$, where $(x, y) \in \mathcal{V}$.

$$\begin{aligned}\text{payoff}(x', y'; x, y) &= \frac{|\{K \in \mathcal{K} : \ h_K(x') = y', h_K(x) = y\}|}{|\{K \in \mathcal{K} : \ h_K(x) = y\}|} \\ &= \frac{|\mathcal{K}|/M^2}{|\mathcal{K}|/M} = \frac{1}{M}.\end{aligned}$$

Proof.

Let $x \in \mathcal{X}$ and $y \in \mathcal{Y}$.

$$\text{payoff}(x, y) = \frac{|\{K \in \mathcal{K} : \ h_K(x) = y\}|}{|\mathcal{K}|} = \frac{|\mathcal{K}|/M}{|\mathcal{K}|} = \frac{1}{M}.$$

Now let $x, x' \in \mathcal{X}$ such that $x \neq x'$, and let $y, y' \in \mathcal{Y}$, where $(x, y) \in \mathcal{V}$.

$$\begin{aligned}\text{payoff}(x', y';\, x, y) &= \frac{|\{K \in \mathcal{K} : \ h_K(x') = y', h_K(x) = y\}|}{|\{K \in \mathcal{K} : \ h_K(x) = y\}|} \\ &= \frac{|\mathcal{K}|/M^2}{|\mathcal{K}|/M} = \frac{1}{M}.\end{aligned}$$

$$\Longrightarrow \ Pd_0 = Pd_1 = \frac{1}{M}$$

□

Thanks for listening!

For further reading:


Douglas R. Stinson
Cryptography: Theory and Practice
2nd ed., Chapman & Hall/CRC, 2002.


Bruce Schneier
Applied Cryptography: Protocols, Algorithms, and Source Code in C
John Wiley & Sons, Inc., 1994.


Bart van Rompay
Analysis and Design of Cryptographic Hash Functions, MAC Algorithms and Block Ciphers
Ph. D. thesis, Katholieke Universiteit Leuven, 2004.