Message Authentication Codes and Collision-Resistant Hash Functions

Yu Zhang

HIT/CST/NIS

Cryptography, Spring, 2014

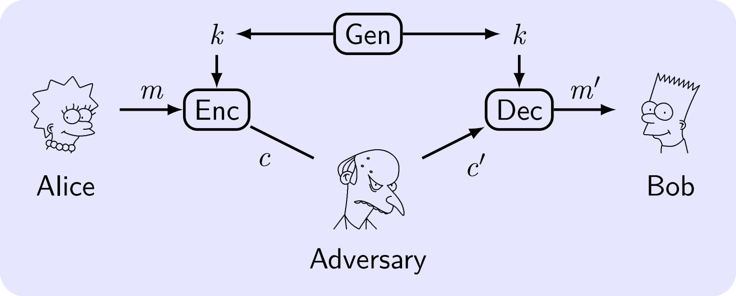
Outline

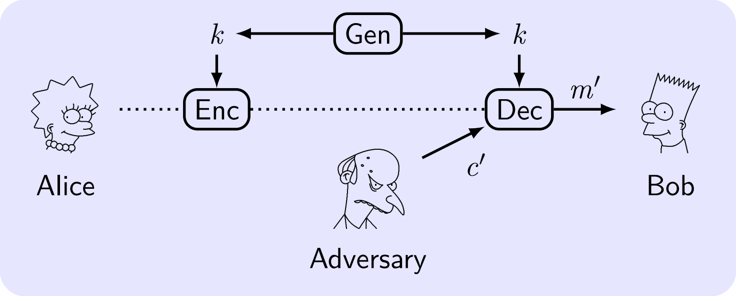
1. Message Authentication Codes (MAC) – Definitions
2. Constructing Secure Message Authentication Codes
3. CBC-MAC
4. Collision-Resistant Hash Functions
5. NMAC and HMAC

Content

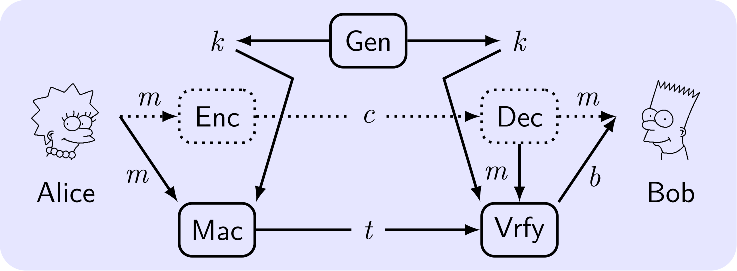
1. **Message Authentication Codes (MAC) – Definitions**
2. Constructing Secure Message Authentication Codes
3. CBC-MAC
4. Collision-Resistant Hash Functions
5. NMAC and HMAC

Integrity and Authentication





The Syntax of MAC



- key k, tag t, a bit b means valid if $b = 1$; invalid if $b = 0$.
- **Key-generation** algorithm $k \leftarrow \mathsf{Gen}(1^n), |k| \geq n$.
- **Tag-generation** algorithm $t \leftarrow \mathsf{Mac}_k(m)$.
- **Verification** algorithm $b := \mathsf{Vrfy}_k(m, t)$.
- **Message authentication code**: $\Pi = (\mathsf{Gen}, \mathsf{Mac}, \mathsf{Vrfy})$.
- **Basic correctness requirement**: $\mathsf{Vrfy}_k(m, \mathsf{Mac}_k(m)) = 1$.

Security of MAC

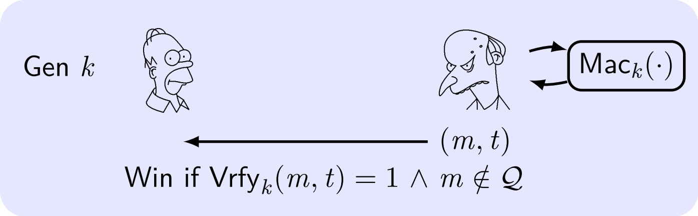
- **Intuition**: No adversary should be able to generate a **valid** tag on any "**new**" message[1] that was not previously sent.
- **Replay attack**: Copy a message and tag previously sent. (**excluded by only considering "new" message**)
  - Sequence numbers: receiver must store the previous ones.
  - Time-Stamps: sender/receiver maintain synchronized clocks.
- **Existential unforgeability**: **Not** be able to forge a valid tag on **any** message.
  - **Existential forgery**: *at least one* message.
  - **Selective forgery**: message chosen *prior* to the attack.
  - **Universal forgery**: *any* given message.
- **Adaptive chosen-message attack (CMA)**: be able to obtain tags on *any* message chosen adaptively *during* its attack.

[1] A stronger requirement is concerning *new message/tag pair*.

Definition of MAC Security

The message authentication experiment $\mathsf{Macforge}_{\mathcal{A},\Pi}(n)$:

1. $k \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given input 1^n and oracle access to $\mathsf{Mac}_k(\cdot)$, and outputs (m, t). $\mathcal{Q}$ is the set of queries to its oracle.
3. $\mathsf{Macforge}_{\mathcal{A},\Pi}(n) = 1 \iff \mathsf{Vrfy}_k(m, t) = 1 \wedge m \notin \mathcal{Q}$.



Gen k

$\mathsf{Mac}_k(\cdot)$

(m, t)

Win if $\mathsf{Vrfy}_k(m, t) = 1 \wedge m \notin \mathcal{Q}$

Definition 1

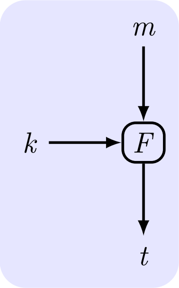
A MAC Π is **existentially unforgeable under an adaptive CMA** if $\forall$ PPT $\mathcal{A}$, $\exists$ negl such that:

$$\Pr[\mathsf{Macforge}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

Content

1. Message Authentication Codes (MAC) – Definitions
2. **Constructing Secure Message Authentication Codes**
3. CBC-MAC
4. Collision-Resistant Hash Functions
5. NMAC and HMAC

Constructing Secure MACs



Construction 2

- *F is PRF. $|m| = n$.*
- Gen(1^n)*: $k \leftarrow \{0,1\}^n$ u.a.r.*
- Mac$_k(m)$*: $t := F_k(m)$.*
- Vrfy$_k(m,t)$*: $1 \iff t \overset{?}{=} F_k(m)$.*

Theorem 3

If F is a PRF, Construction is a secure fixed-length MAC.

Lemma 4

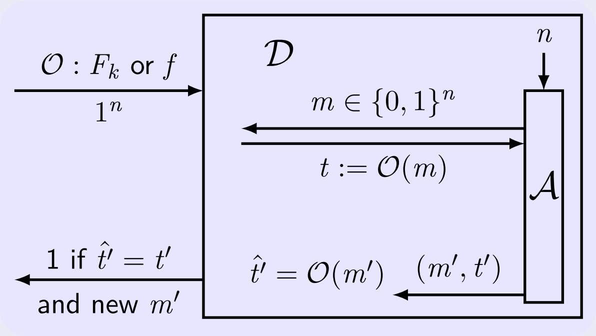
***Truncating MACs based on PRFs**: If F is a PRF, so is $F^t_k(m) = F_k(m)[1,\ldots,t]$.*

Proof of Secure MAC from PRF

Idea: Show Π is secure unless F_k is not PRF by reduction.

Proof.

D distinguishes F_k; $\mathcal{A}$ attacks Π.



□

Proof of Secure MAC from PRF (Cont.)

Proof.

(1) If true random f is used, $t = f(m)$ is uniformly distributed.

$$\Pr[D^{f(\cdot)}(1^n) = 1] = \Pr[\mathsf{Macforge}_{\mathcal{A},\tilde{\Pi}}(n) = 1] \leq 2^{-n}.$$

(2) If F_k is used, conduct the experiment $\mathsf{Macforge}_{\mathcal{A},\Pi}(n)$.

$$\Pr[D^{F_k(\cdot)}(1^n) = 1] = \Pr[\mathsf{Macforge}_{\mathcal{A},\Pi}(n) = 1] = \varepsilon(n).$$

According to the definition of PGF,

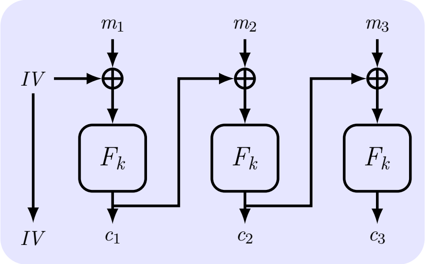
$$\left|\Pr[D^{F_k(\cdot)}(1^n) = 1] - \Pr[D^{f(\cdot)}(1^n) = 1]\right| \geq \varepsilon(n) - 2^{-n}.$$

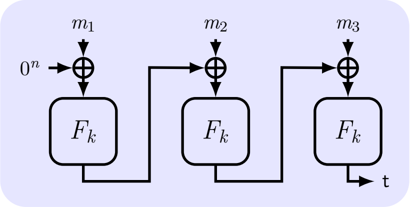
□

Content

1. Message Authentication Codes (MAC) – Definitions
2. Constructing Secure Message Authentication Codes
3. **CBC-MAC**
4. Collision-Resistant Hash Functions
5. NMAC and HMAC

Constructing Fixed-Length CBC-MAC





Modify CBC encryption into CBC-MAC:

- Change random IV to encrypted fixed 0^n,*otherwise*: query m_1 and get (IV, t_1); output $m_1' = IV' \oplus IV \oplus m_1$ and (IV', t_1).
- Tag only includes the output of the final block,*otherwise*: query m_i and get t_i; output $m_i' = t_{i-1}' \oplus t_{i-1} \oplus m_i$ and t_i.

Constructing Fixed-Length CBC-MAC (Cont.)

Construction 5

- *a PRF F and a length function ℓ. $|m| = \ell(n) \cdot n$. $\ell = \ell(n)$. $m = m_1, \ldots, m_\ell$.*
- Gen(1^n)*: $k \leftarrow \{0,1\}^n$ u.a.r.*
- Mac$_k(m)$*: $t_i := F_k(t_{i-1} \oplus m_i), t_0 = 0^n$. Output $t = t_\ell$.*
- Vrfy$_k(m,t)$*: $1 \iff t \overset{?}{=}$* Mac$_k(m)$*.*

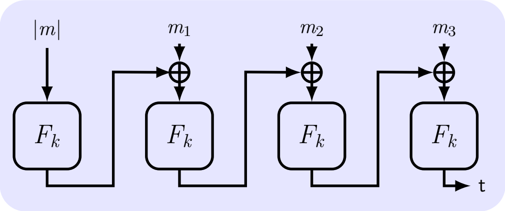
Theorem 6

If F is a PRF, Construction is a secure ***fixed-length*** *MAC.*

Not for **variable-length** message:
For one-block message m with tag t, adversary can append a block $t \oplus m$ and output tag t.

Secure Variable-Length MAC

- **Option 1**: $k_\ell := F_k(\ell)$, use k_ℓ for CBC-MAC.
- **Option 2**: Prepend m with $|m|$, then use CBC-MAC.



- **Option 3 (ECBC-MAC)**: Use two keys k_1, k_2. Get t with k_1 by CBC-MAC, then output $\hat{t} := F_{k_2}(t)$.

Lessons learned

Wrap CBC-MAC with PRF(length/tag), and only output is tag!

Content

1 Message Authentication Codes (MAC) – Definitions

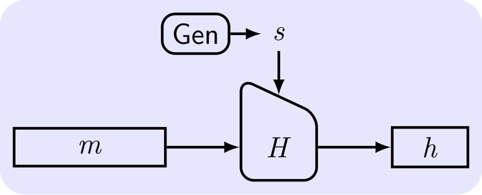
2 Constructing Secure Message Authentication Codes

3 CBC-MAC

4 **Collision-Resistant Hash Functions**

5 NMAC and HMAC

Defining Hash Function



Definition 7

A **hash function (compression function)** is a pair of PPT algorithms (Gen, H) satisfying:

- a key $s \leftarrow \text{Gen}(1^n)$, s is **not kept secret**.
- $H^s(x) \in \{0,1\}^{\ell(n)}$, where $x \in \{0,1\}^*$ and ℓ is polynomial.

If H^s is defined only for $x \in \{0,1\}^{\ell'(n)}$ and $\ell'(n) > \ell(n)$, then (Gen, H) is a **fixed-length** hash function.

Defining Collision Resistance

- **Collision** in H: $x \neq x'$ and $H(x) = H(x')$.
- **Collision Resistance**: infeasible for any PPT alg. to find.

The collision-finding experiment $\mathsf{Hashcoll}_{\mathcal{A},\Pi}(n)$:

1. $s \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given s and outputs x, x'.
3. $\mathsf{Hashcoll}_{\mathcal{A},\Pi}(n) = 1 \iff x \neq x' \wedge H^s(x) = H^s(x')$.

Definition 8

Π $(H,\ H^s)$ is **collision resistant** if $\forall$ PPT $\mathcal{A}$, $\exists$ negl such that

$$\Pr[\mathsf{Hashcoll}_{\mathcal{A},\Pi}(n) = 1] \leq \mathsf{negl}(n).$$

The "Birthday" Problem

The "Birthday" Problem

Q: "*What size group of people do we need to take such that with probability $1/2$ some pair of people in the group share a birthday?*"
A: 23.

Lemma 9

Choose q elements $y_1, \ldots, y_q$ u.a.r from a set of size N, the probability that $\exists\ i \neq j$ with $y_i = y_j$ is coll(q, N)*, then*

$$\text{coll}(q, N) \leq \frac{q^2}{2N}.$$

$$\text{coll}(q, N) \geq \frac{q(q-1)}{4N} \quad \textit{if } q \leq \sqrt{2N}.$$

$$\text{coll}(q, N) = \Theta(q^2/N) \quad \textit{if } q < \sqrt{N}.$$

A Generic "Birthday" Attack

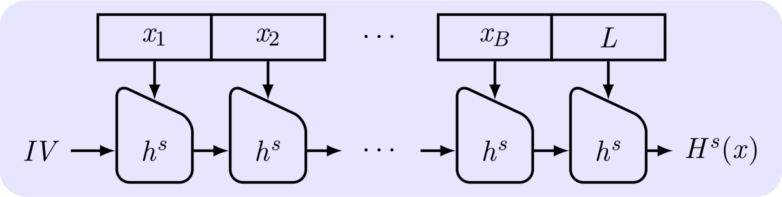
- **Birthday Attack**: $H : \{0,1\}^* \to \{0,1\}^\ell$. Choose q distinct inputs $x_1, \ldots, x_q \in \{0,1\}^{2\ell}$, check whether any of two $y_i := H(x_i)$ are equal.
- **Birthday problem**: Choose $y_1, \ldots, y_q \leftarrow \{0,1\}^\ell$ *u.a.r*, coll$(q, 2^\ell) =?$
- Collision occurs with a high probability when $\mathcal{O}(q) = \mathcal{O}(2^{\ell/2})$.
- To let time $T > 2^{\ell/2}$, then $\ell = 2\log T$ at least.
- Work only for collision resistance, no generic attacks for 2nd pre-image or pre-image resistance better than 2^ℓ.
- Require too much space $\mathcal{O}(2^{\ell/2})$.

Constructing “Meaningful” Collisions

An example with 288 different meaningful sentences

It is **hard/difficult/challenging/impossible** to **imagine/believe** that we will **find/locate/hire** another **employee/person** having similar **abilities/skills/character** as Alice. She has done a **great/super** job.

The Merkle-Damgård Transform



Construction 10

Construct ***variable-length*** *CRHF* (Gen, H) *from fixed-length* (Gen, h) *(2ℓ bits $\to \ell$ bits, $\ell = \ell(n)$):*

- Gen*: remains unchanged.*
- H*: key s and string $x \in \{0,1\}^*$, $L = |x| < 2^\ell$:*
  - $B := \lceil \frac{L}{\ell} \rceil$ *(# blocks).* ***Pad x with 0s.*** *ℓ-bit blocks $x_1, \ldots, x_B$.* $x_{B+1} := L$*, L is encoded using ℓ bits.*
  - $z_0 := IV = 0^\ell$*. For $i = 1, \ldots, B+1$, compute* $z_i := h^s(z_{i-1} \| x_i)$*.*

Security of the Merkle-Damgård Transform

Theorem 11

If (Gen, h) *is a fixed-length CRHF, then* (Gen, H) *is a CRHF.*

Proof.

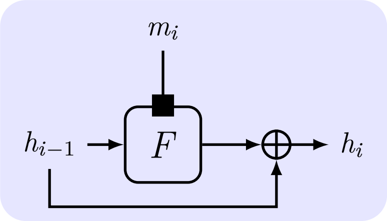
Idea: a collision in H^s yields a collision in h^s.
Two messages $x \neq x'$ of respective lengths L and L' such that $H^s(x) = H^s(x')$. $\#$ blocks are B and B'.
$x_{B+1} := L$ is necessary since **Padding with 0s** will lead to the same input with different messages.

1. $L \neq L'$: $z_B \| L \neq z_{B'} \| L'$.
2. $L = L'$: $z_{i^*-1} \| x_{i^*} \neq z'_{i^*-1} \| x'_{i^*}$.

So there must be $x \neq x'$ such that $h^s(x) = h^s(x')$. □

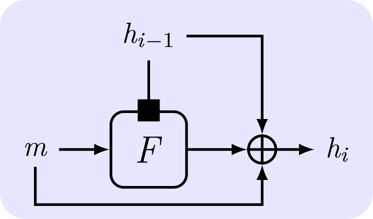
CRHF from Block Cipher

Davies-Meyer:



Used by SHA-1/2, MD5.

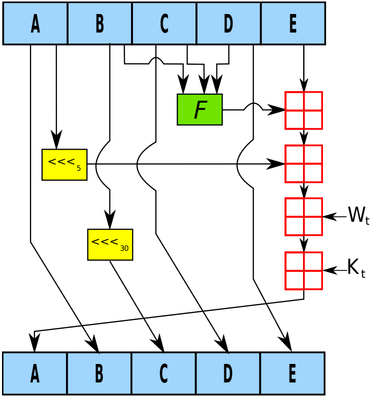
Miyaguchi-Preneel:



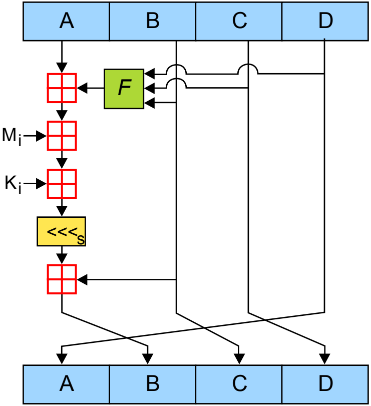
Used by Whirlpool (in ISO/IEC 10118-3).

Cryptographic Hash Functions: SHA-1 and MD5

SHA-1:



MD5:



A, B, C, D and E are 32-bit words of the state; F is a nonlinear function that varies; $\lll n$ denotes a left bit rotation by n places; W_t/M_t is the expanded message word of round t; K_t is the round constant of round t; $\boxplus$ denotes addition modulo 2^{32}.

Collision-Resistant Hash Functions in Practice

- The hash functions used in practice are generally un-keyed.
- The constructions are more heuristic in nature.
- Finding a collision in MD5 (Message Digest 5) with 128-bit output requires time $2^{20.96}$.
- Finding a collision in SHA-1 (Secure Hash Algorithm) with a 160-bit output requires time 2^{51}.

Content

1 Message Authentication Codes (MAC) – Definitions

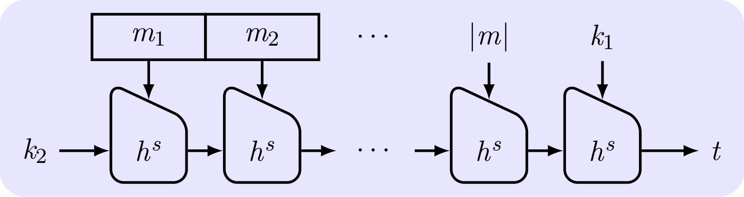
2 Constructing Secure Message Authentication Codes

3 CBC-MAC

4 Collision-Resistant Hash Functions

5 **NMAC and HMAC**

Nested MAC (NMAC)



Construction 12

$(\widetilde{\mathsf{Gen}}, h)$ *is a fixed-length CRHF.* $(\widetilde{\mathsf{Gen}}, H)$ *is Merkle-Damgård transform. NMAC:*

- $\mathsf{Gen}(1^n)$*: Output* (s, k_1, k_2)*.* $s \leftarrow \widetilde{\mathsf{Gen}}, k_1, k_2 \leftarrow \{0,1\}^n$ *u.a.r.*
- $\mathsf{Mac}_{s,k_1,k_2}(m)$*:* $t_i := h^s_{k_1}(H^s_{k_2}(m))$*.* $h^s_k \overset{\mathsf{def}}{=} h^s(k\|x)$*.*
  $H^s_{k_2}$ *is inner function;* $h^s_{k_1}$ *is outer function.*
- $\mathsf{Vrfy}_{s,k_1,k_2}(m,t)$*:* $1 \iff t \overset{?}{=} \mathsf{Mac}_{s,k_1,k_2}(m)$*.*

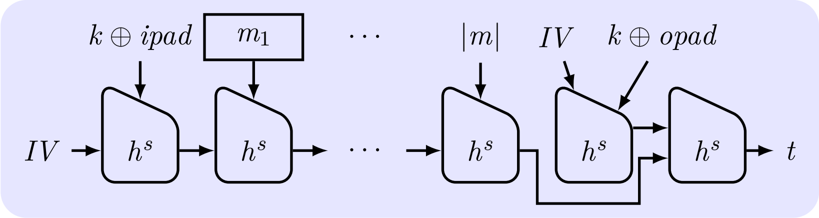
Security of NMAC

Theorem 13

If $(\widetilde{\mathsf{Gen}}, h)$ is CRHF and yields a secure MAC, then NMAC is secure. (existentially unforgeable under an adaptive CMA for arbitrary-length messages)

- k_2 is not needed once $(\widetilde{\mathsf{Gen}}, h)$ is CRHF.
  - **Weak collision resistance**: It is hard to find $(x, x'), x' \neq x$ such that $H^s_{k_2}(x) = H^s_{k_2}(x')$.
  - $H^{k_2}_s(x)$ is hidden by $h^{k_1}_s(H^{k_2}_s(x))$.
  - **Disadvantage**: IV of H must be modified.

Hash-based MAC (HMAC)



Construction 14

$(\widetilde{\mathsf{Gen}}, h)$ is a fixed-length CRHF. $(\widetilde{\mathsf{Gen}}, H)$ is the Merkle-Damgård transform. IV, opad *(0x36),* ipad *(0x5C) are fixed constants of length n. HMAC:*

- $\mathsf{Gen}(1^n)$*: Output (s, k). $s \leftarrow \widetilde{\mathsf{Gen}}, k \leftarrow \{0,1\}^n$ u.a.r.*
- $\mathsf{Mac}_{s,k}(m)$*: $t := H^s_{IV}\Big((k \oplus \mathsf{opad}) \| H^s_{IV}((k \oplus \mathsf{ipad}) \| m)\Big)$.*
- $\mathsf{Vrfy}_{s,k}(m, t)$*: $1 \iff t \overset{?}{=} \mathsf{Mac}_{s,k}(m)$.*

Security of HMAC

Theorem 15

$$G(k) \overset{\textit{def}}{=} h^s(IV \| (k \oplus \mathsf{opad})) \| h^s(IV \| (k \oplus \mathsf{ipad})) = k_1 \| k_2.$$

$(\widetilde{\mathsf{Gen}}, h)$ is CRHF. If G is a PRG, then HMAC is secure.

- HMAC is an industry standard (RFC2104) and is widely used in practice.
- HMAC is faster than CBC-MAC.
- Before HMAC, a common mistake was to use $H^s(k \| x)$.
- *Don't implement it yourself.* Verification timing attacks.

Summary

- adaptive CMA, replay attack, birthday attack.
- existential unforgeability, collision resistance.
- CBC-MAC, CRHF, Merkle-Damgård transform, NMAC, HMAC.