Message Authentication Codes and Collision-Resistant Hash Functions

Yu Zhang

Harbin Institute of Technology

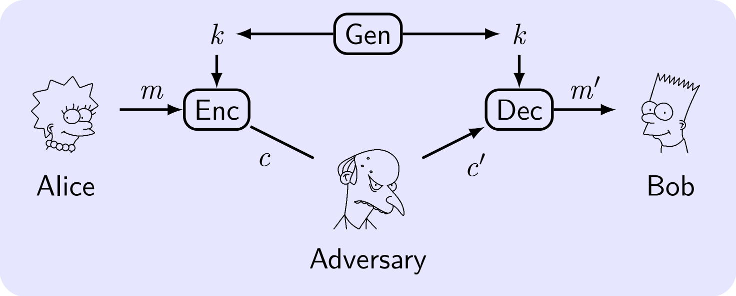
Cryptography, Autumn, 2015

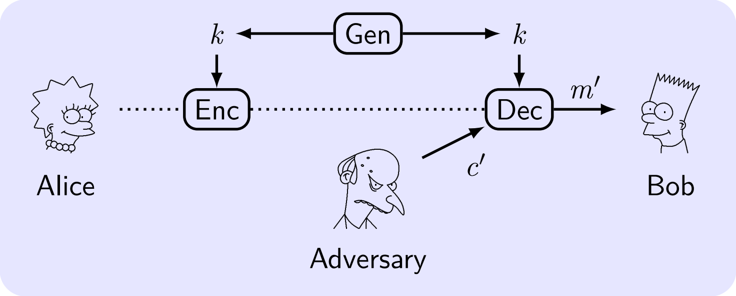
Outline

Content

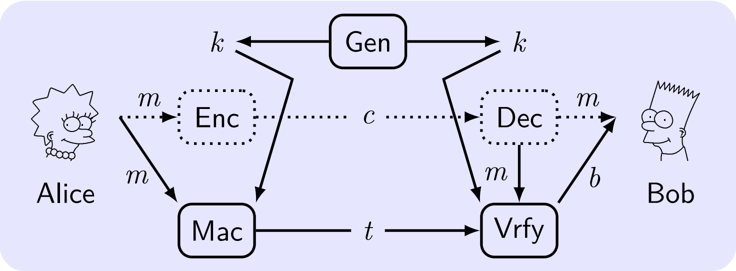
1. Message Authentication Codes (MAC) – Definitions
2. Constructing Secure Message Authentication Codes
3. CBC-MAC
4. Collision-Resistant Hash Functions
5. HMAC

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 $k \leftarrow \text{Gen}(1^n), |k| \geq n$.
- **Tag-generation** algorithm $t \leftarrow \text{Mac}_k(m)$.
- **Verification** algorithm $b := \text{Vrfy}_k(m, t)$.
- **Message authentication code**: $\Pi = (\text{Gen}, \text{Mac}, \text{Vrfy})$.
- **Basic correctness requirement**: $\text{Vrfy}_k(m, \text{Mac}_k(m)) = 1$.

Security of MAC

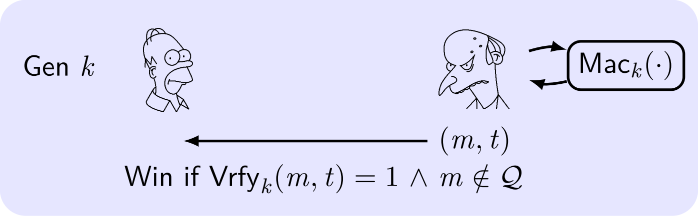
- **Intuition**: No adversary should be able to generate a **valid** tag on any "**new**" message[1] that was not previously sent.
- **Replay attack**: Copy a message and tag previously sent. (**excluded by only considering "new" message**)
  - Sequence numbers: receiver must store the previous ones.
  - 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. $\mathsf{Macforge}_{\mathcal{A},\Pi}(n) = 1 \iff \mathsf{Vrfy}_k(m, t) = 1 \wedge m \notin \mathcal{Q}$.



Definition 1

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

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

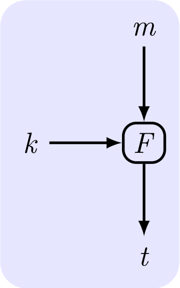
Questions

Suppose $\langle S, V\rangle$ are CMA-secure, are $\langle S', V'\rangle$ secure?

- $S'_{k_1,k_2}(m) = (S_{k1}(m), S_{k_2}(m))$
  $V'_{k_1,k_2}(m,(t_1,t_2)) = V_{k1}(m,t_1) \wedge V_{k_2}(m,t_2)$
- $S'_k(m) = (S_k(m), S_k(m))$
  $V'_k(m,(t_1,t_2)) = \begin{cases} V_k(m,t_1) & \text{if } t_1 = t_2 \\ 0 & \text{otherwise} \end{cases}$
- $S'_k(m) = (S_k(m), S_k(0^n))$
  $V'_k(m,(t_1,t_2)) = V_k(m,t_1) \wedge V_k(0^n,t_2)$
- $S'_k(m) = S_k(m),\ V'_k(m,t) = \begin{cases} V_k(m,t) & \text{if } m \neq 0^n \\ 1 & \text{otherwise} \end{cases}$
- $S'_k(m) = S_k(m)$ without the LSB
  $V'_k(m,t) = V_k(m,t\|0) \ \vee \ V_k(m,t\|1)$
- $S'_k(m) = (S_k(m), m),\ V'_k(m,(t_1,t_2)) = V_k(m,t_1) \wedge t_2 = m$

Content

Constructing Secure MACs



Construction 2

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

Theorem 3

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

Lemma 4

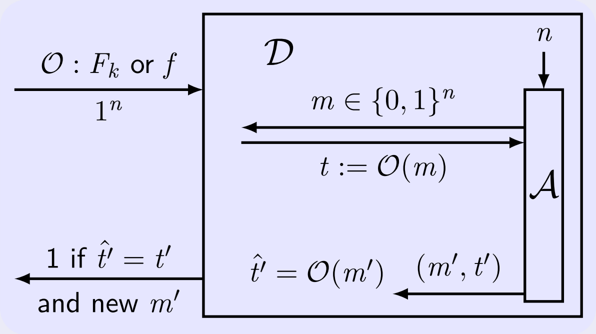
Truncating MACs based on PRFs: *If F is a PRF, so is* $F_k^t(m) = F_k(m)[1, \ldots, t]$.

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 PRF,

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

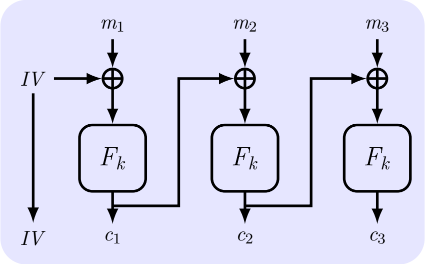
□

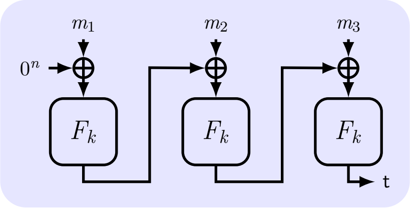
Extension to Variable-Length Messages

For variable-Length Messages, would the following suggestions be secure?

- **Suggestion 1**: XOR all the blocks together and authenticate the result. $t := \mathsf{Mac}'_k(\oplus_i m_i)$.
- **Suggestion 2**: Authenticate each block separately. $t_i := \mathsf{Mac}'_k(m_i)$.
- **Suggestion 3**: Authenticate each block along with a sequence number. $t_i := \mathsf{Mac}'_k(i \| m_i)$.

Content





Modify CBC encryption into CBC-MAC:

- Change random IV to encrypted fixed 0^n,*otherwise*:
  Q: query m_1 and get (IV, t_1); output $m_1' = IV' \oplus IV \oplus m_1$ and $t' =$ ____.
- Tag only includes the output of the final block,*otherwise*:
  Q: 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. $|m| = \ell(n) \cdot n$. $\ell = \ell(n)$. $m = m_1, \ldots, m_\ell$.*
- $\mathsf{Gen}(1^n)$*: $k \leftarrow \{0,1\}^n$ u.a.r.*
- $\mathsf{Mac}_k(m)$*: $t_i := F_k(t_{i-1} \oplus m_i), t_0 = 0^n$. Output $t = t_\ell$.*
- $\mathsf{Vrfy}_k(m,t)$*: $1 \iff t \stackrel{?}{=} \mathsf{Mac}_k(m)$.*

Theorem 6

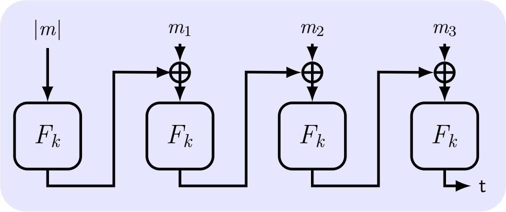
*If F is a PRF, Construction is a secure **fixed-length** MAC.*

Not for **variable-length** message:

Q: For one-block message m with tag t, adversary can append a block ____ and output tag t.

Secure Variable-Length MAC

- **Input-length key separation**: $k_\ell := F_k(\ell)$, use k_ℓ for CBC-MAC.
- **Length-prepending**: Prepend m with $|m|$, then use CBC-MAC.



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

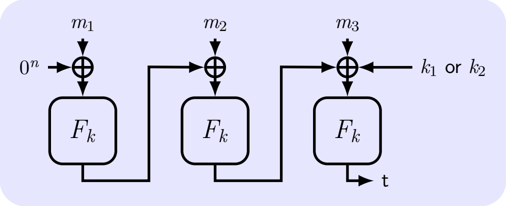
Q: to authenticate a voice stream, which approach do you prefer?

MAC Padding

Padding must be invertible!

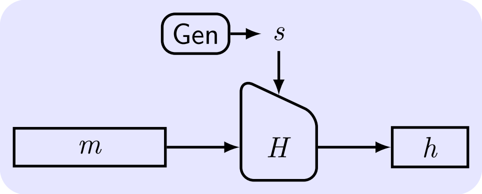
$$m_0 \neq m_1 \Rightarrow \mathsf{pad}(m_0) \neq \mathsf{pad}(m_1).$$

ISO: pad with "100...00". Add dummy block if needed.
Q: what if no dummy block?
CMAC (Cipher-based MAC from NIST): key$= (k, k_1, k_2)$.



- No final encryption: extension attack thwarted by keyed XOR.
- No dummy block: ambiguity resolved by use of k_1 or k_2.

Content



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 $\mathsf{Hashcoll}_{\mathcal{A},\Pi}(n)$:

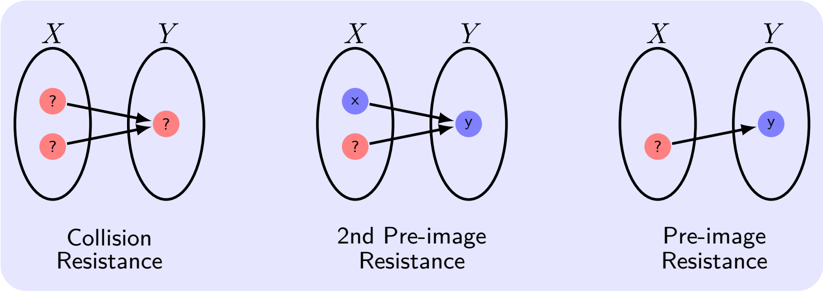
1. $s \leftarrow \mathsf{Gen}(1^n)$.
2. $\mathcal{A}$ is given s and outputs x, x'.
3. $\mathsf{Hashcoll}_{\mathcal{A},\Pi}(n) = 1 \iff x \neq x' \wedge H^s(x) = H^s(x')$.

Definition 8

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

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

Weaker Notions of Security for Hash Functions



- **Collision resistance**: It is hard to find $(x, x'), x' \neq x$ such that $H(x) = H(x')$.
- **Second pre-image resistance**: Given s and x, it is hard to find $x' \neq x$ such that $H^s(x') = H^s(x)$.
- **Pre-image resistance**: Given s and $y = H^s(x)$, it is hard to find x' such that $H^s(x') = y$.

Questions

H is CRHF. Is H' CRHF?

- $H'(m) = H(m) \oplus H(m \oplus 1^{|m|})$
- $H'(m) = H(m) \| H(0)$
- $H'(m) = H(m) \| H(m)$
- $H'(m) = H(m) \oplus H(m)$
- $H'(m) = H(m[0, \ldots, |m| - 2])$
- $H'(m) = H(m \| 0)$

Applications of Hash Functions

- **Fingerprinting and Deduplication**: $H(alargefile)$ for virus fingerprinting, deduplication, P2P file sharing
- **Merkle Trees**: $H(H(H(file1), H(file2)), H(H(file3), H(file4)))$ fingerprinting multiple files / parts of a file
- **Passward Hashing**: $(salt, H(salt, pw))$ mitigating the risk of leaking password stored in the clear
- **Key Derivation**: $H(secret)$ deriving a key from a high-entropy (but not necessarily uniform) shared secret
- **Commitment Schemes**: $H(info)$ hiding the commited info; binding the commitment to a info

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* $\mathsf{coll}(q, N)$*, then*

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

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

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

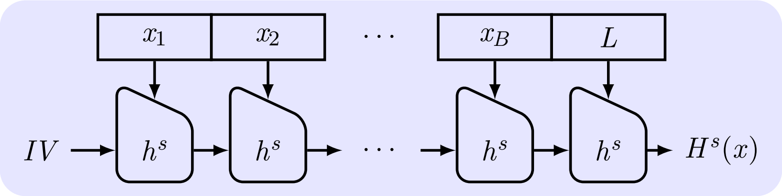
The length of hash value should be long enough.

Constructing "Meaningful" Collisions

How many different meaningful sentences are in the below paragraph?

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.

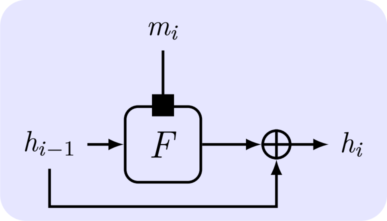
Idea: a collision in H^s yields a collision in h^s.
Two messages $x \neq x'$ of respective lengths L and L' such that $H^s(x) = H^s(x')$. # blocks are B and B'.
$x_{B+1} := L$ is necessary since **Padding with 0s** will lead to the same input with different messages.

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 $h^s(x) = h^s(x')$. □

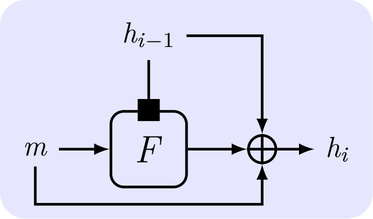
CRHF from Block Cipher

Davies-Meyer (SHA-1/2, MD5)



$$h_i = F_{m_i}(h_{i-1}) \oplus h_{i-1}$$

Miyaguchi-Preneel (Whirlpool)



$$h_i = F_{h_{i-1}}(m_i) \oplus h_{i-1} \oplus m$$

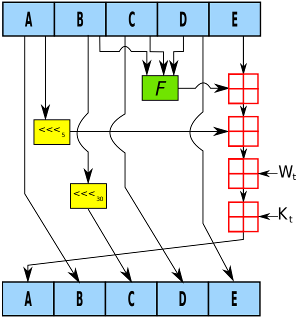
Theorem 12

If F is modeled as an ideal cipher, then Davies-Meyer construction yields a CRHF.

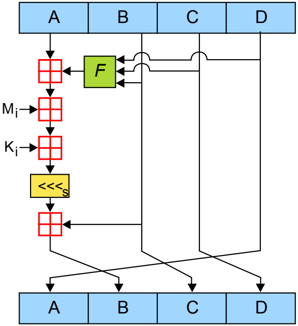
Q: what if $h_i = F_{m_i}(h_{i-1})$ without XOR with h_{i-1}?
Q: what if F is not ideal such that $\exists x, F_k(x) = x$?

Cryptographic Hash Functions: SHA-1 and MD5

SHA-1:



MD5:

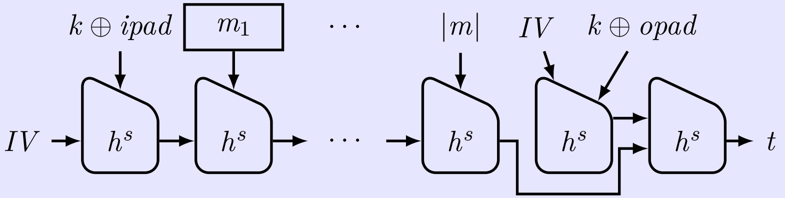


A, B, C, D and E are 32-bit words of the state; F is a nonlinear function that varies; $\lll n$ denotes a left bit rotation by n places; W_t/M_t is the expanded message word of round t; K_t is the round constant of round t; $\boxplus$ denotes addition modulo 2^{32}.

- Finding a collision in 128-bit MD5 requires time $2^{20.96}$
- Finding a collision in 160-bit SHA-1 requires time 2^{51}

Content

Hash-based MAC (HMAC)



Construction 13

$(\widetilde{\mathsf{Gen}}, h)$ *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*
- $\mathsf{Mac}_{s,k}(m)$*:* $t := H^s_{IV}\Big((k \oplus \mathsf{opad}) \| H^s_{IV}\big((k \oplus \mathsf{ipad}) \| m\big)\Big)$
- $\mathsf{Vrfy}_{s,k}(m,t)$*:* $1 \iff t \stackrel{?}{=} \mathsf{Mac}_{s,k}(m)$

Security of HMAC

Theorem 14

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

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

- HMAC is an industry standard (RFC2104)
- HMAC is faster than CBC-MAC
- Before HMAC, a common mistake was to use $H^s(k \| x)$
- Verification timing attacks: (Keyczar crypto library (Python))
  def Verify(key, msg, sig_bytes):
      return HMAC(key, msg) $==$ sig_bytes
  The problem: implemented as a byte-by-byte comparison
- *Don't implement it yourself*

Summary

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