Introduction to Information Security

Lecture 4: Hash Functions and MAC

2007. 6.

Prof. Byoungcheon Lee
sultan (at) joongbu . ac . kr

Information and Communications University

Contents

1. Hash Functions vs. MAC

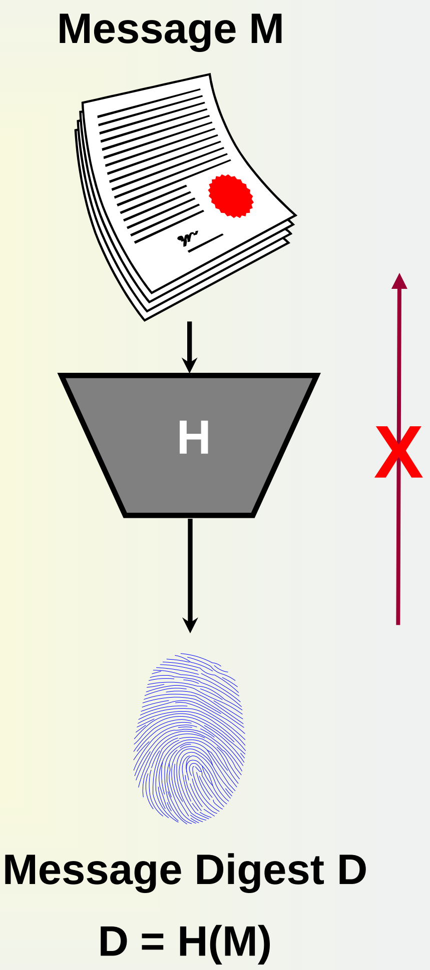
(Message Authentication Code)

Hash Functions

❖ **Hash Function**
- ✓ Generate a fixed length "Fingerprint" for an arbitrary length message
- ✓ No Key involved (public function)
- ✓ Must be at least "One-way" to be useful

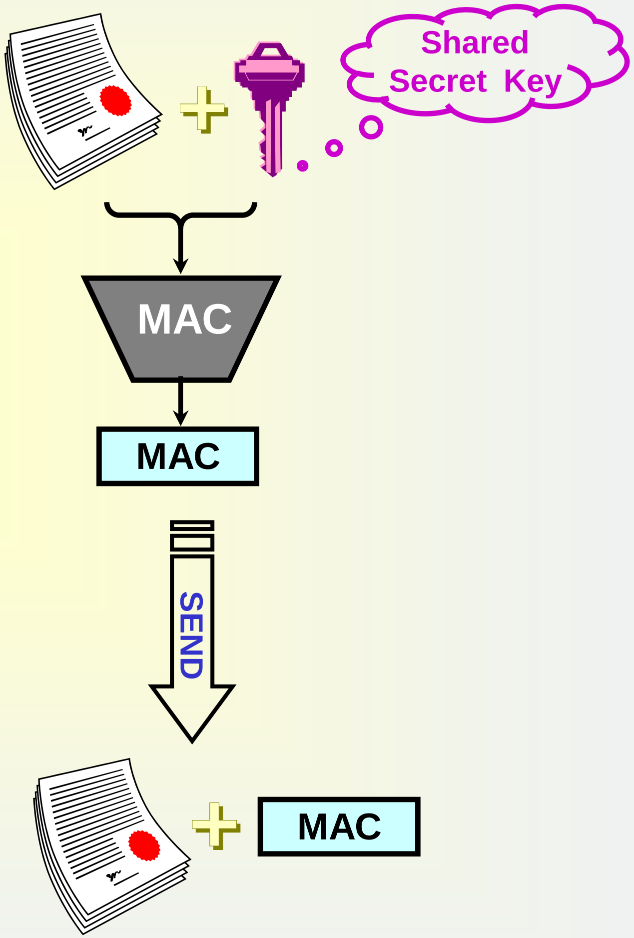
❖ **Applications**
- ✓ Unkeyed hash
  - ✓ digital signature
  - ✓ password file
  - ✓ key stream / pseudo-random number generator
- ✓ Keyed hash: MAC/ICV generation
  (Message Authentication Code, Integrity Check Value)

❖ **Constructions**
- ✓ Iterated hash functions (MD4-family hash functions): MD5, SHA1, SHA2, RMD160, HAS160
- ✓ Hash functions based on block ciphers: MDC(Manipulation Detection Code)



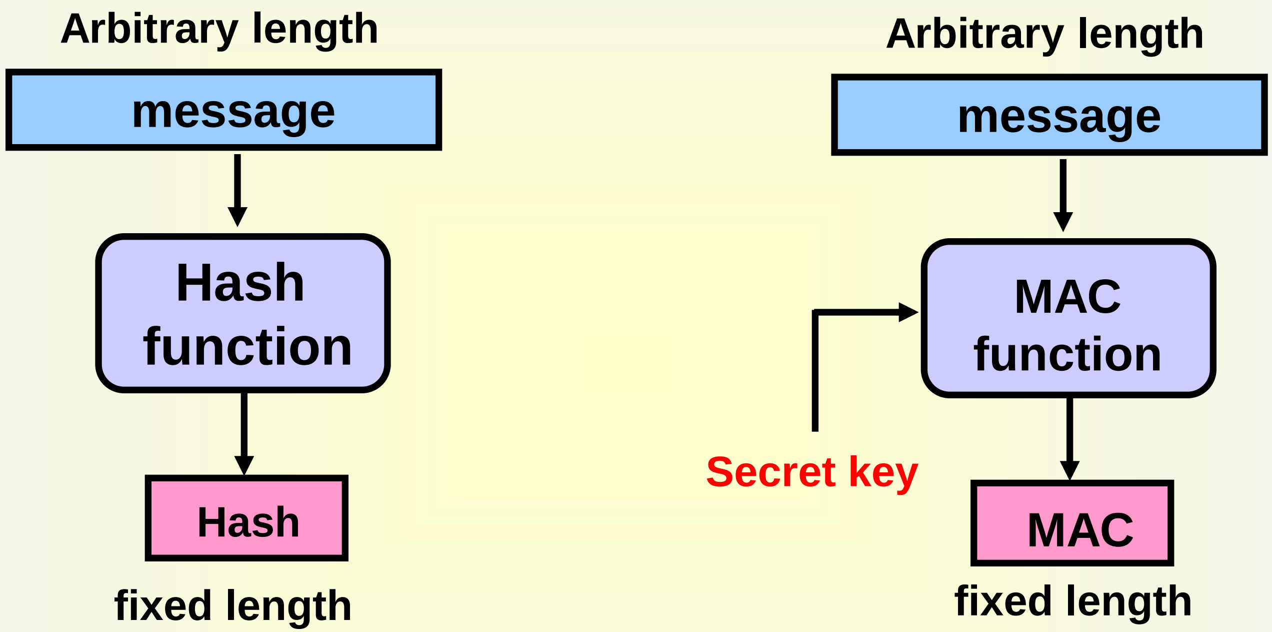
Message Authentication Codes (MACs)

- ***MAC***
  - ✓ Generate a fixed length MAC for an arbitrary length message
  - ✓ A keyed hash function
  - ✓ Provides
    - ✓ Message origin authentication
    - ✓ Message integrity
    - ✓ Entity authentication
    - ✓ Transaction authentication



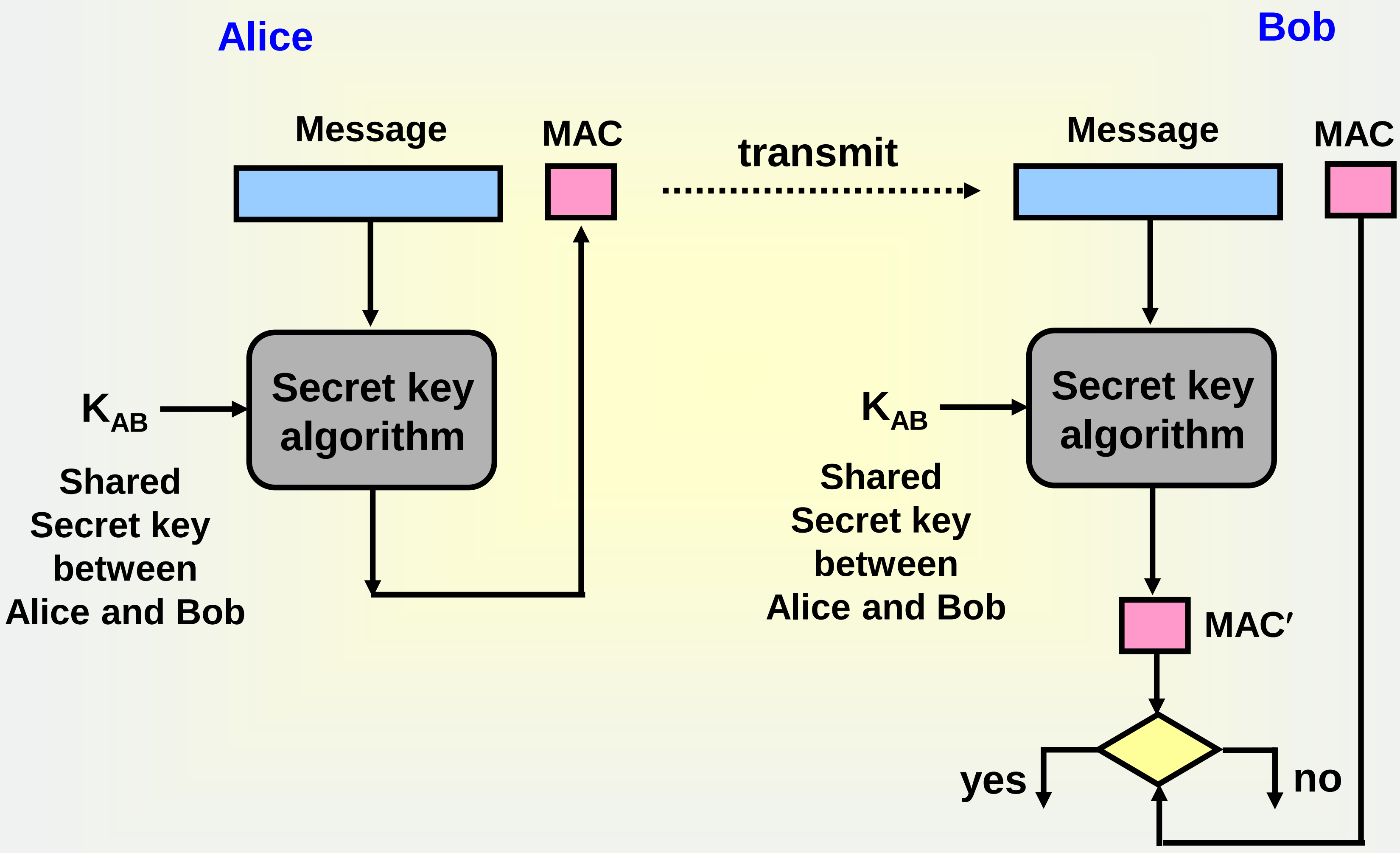
- ***Constructions***
  - ✓ Keyed hash: HMAC, KMAC
  - ✓ Block cipher: CBC-MAC
  - ✓ Dedicated MAC: MAA, UMAC

Comparison of Hash Function & MAC

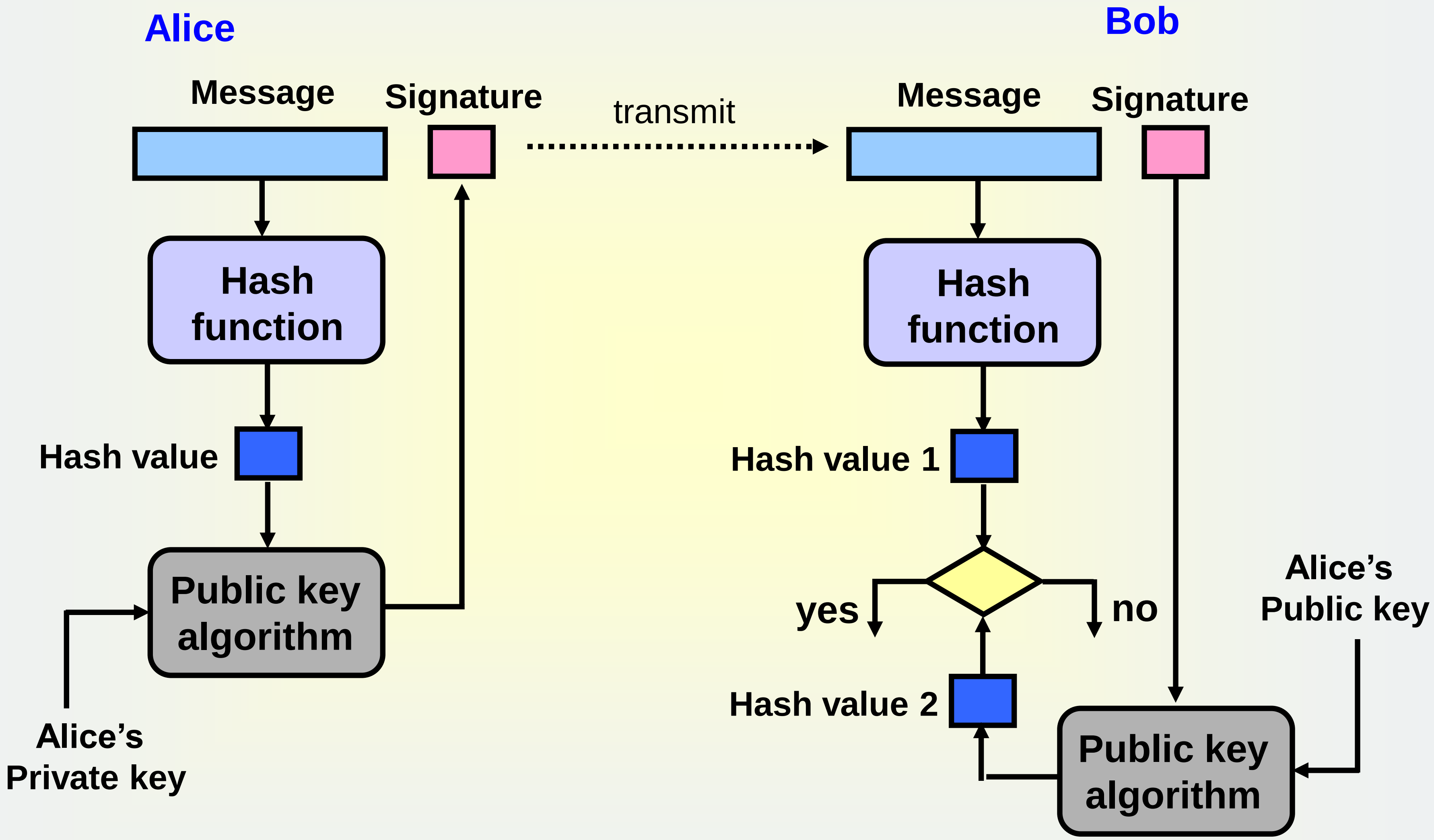


- Easy to compute
- Compression: arbitrary length input to fixed length output
- Unkeyed function vs. Keyed function

Symmetric Authentication (MAC)



Digital Signature



MAC and Digital Signature

- ❖ **MAC (Message Authentication Code)**
  - Generated and verified by a secret key algorithm
  - Message origin authentication & Message integrity
  - Cannot provide non-repudiation
  - Schemes
    - ✓ Keyed hash: HMAC
    - ✓ Block cipher: CBC-MAC, XCBC-MAC
    - ✓ Dedicated MAC: UMAC

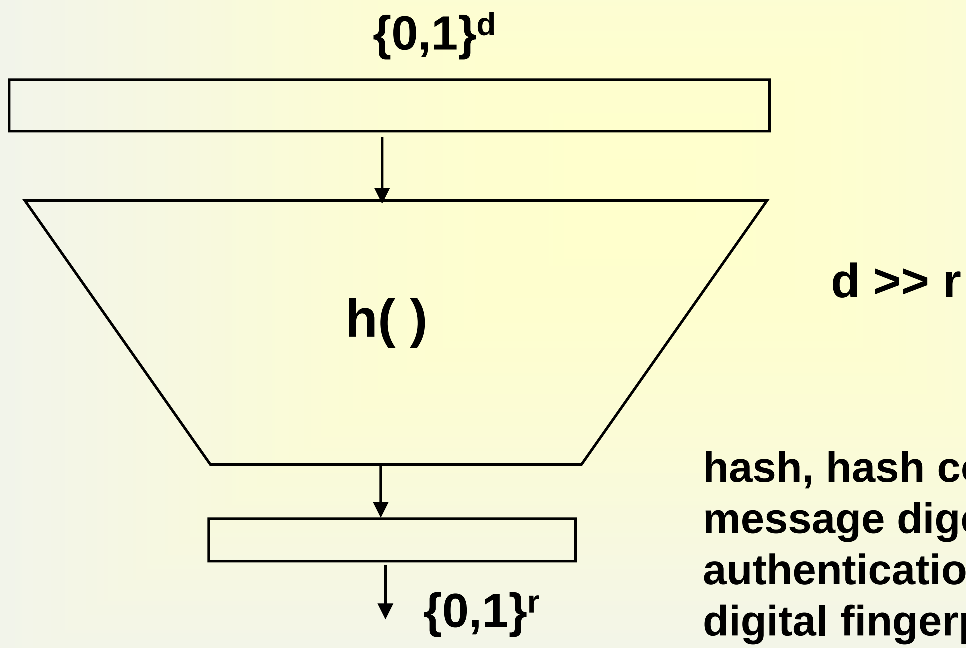
- ❖ **Digital Signature**
  - Generated and verified by a public key algorithm and a hash function
  - Message origin authentication & Message integrity
  - Non-repudiation
  - Schemes
    - ✓ Hash + Digital signature algorithm
    - ✓ RSA; DSA, KCDSA; ECDSA, EC-KCDSA

2. Hash Functions

Hash Functions

❖ **Definition**

- ➢ **Compression: arbitrary length input to fixed length output**
- ➢ **Ease of computation**

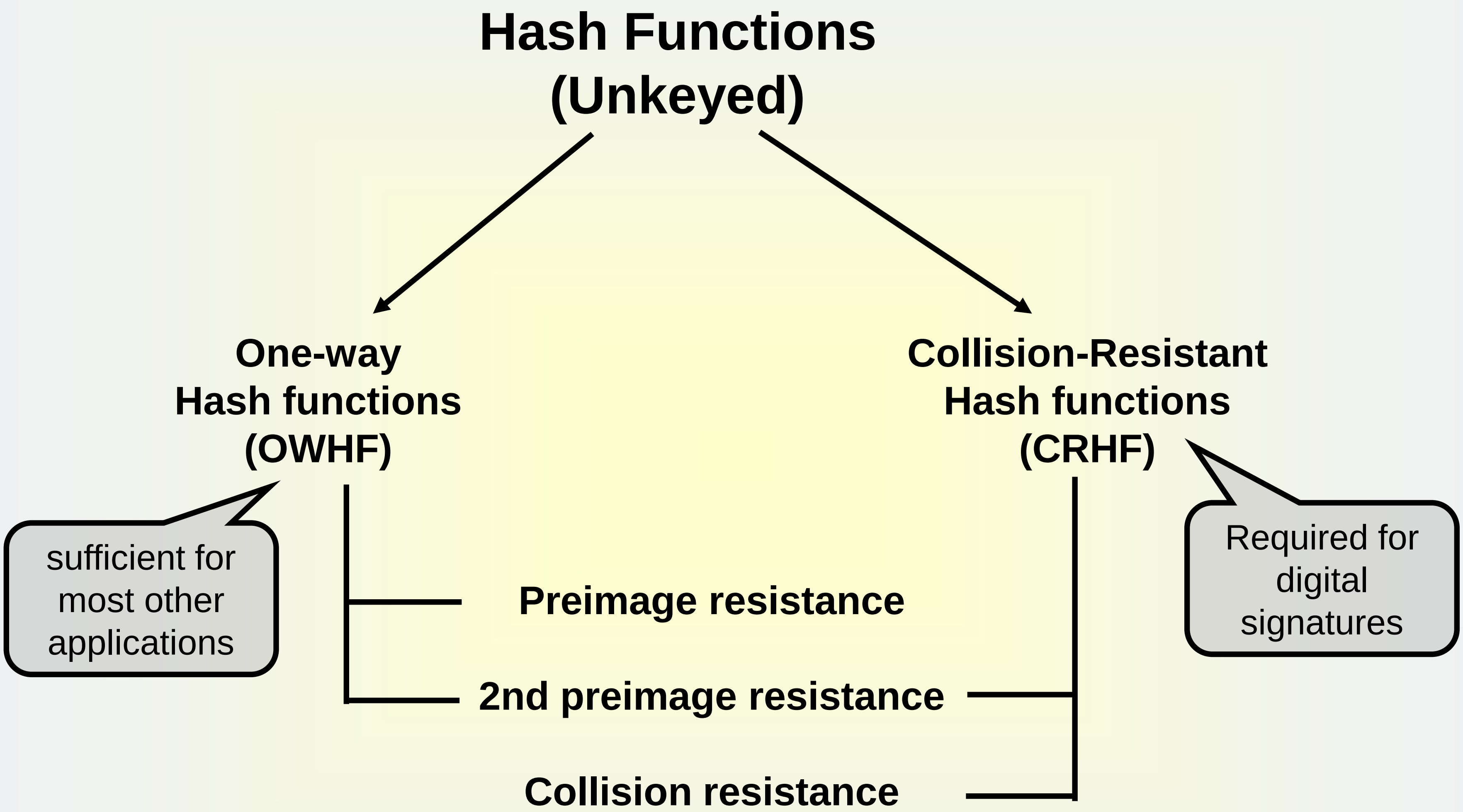


hash, hash code/value/result
message digest, checksum, MIC,
authentication tag, seal, compression
digital fingerprint, imprint

*** Note that collision is inevitable (many-to-one function).**

Hash Functions – Requirements

- **Security Properties**
  - **Preimage resistance** (One-wayness) :
    - Given y, it is computationally infeasible to find any input x such that $y = h(x)$
    - Hardest task, weakest requirement
  - **2nd preimage resistance** (Weak collision resistance) :
    - Given x, it is computationally infeasible to find another input $x' \neq x$ such that $h(x) = h(x')$
  - **Collision resistance** (Strong collision resistance) :
    - It is computationally infeasible to find any two distinct inputs x and x' such that $h(x) = h(x')$
    - Easier task, Strongest requirement

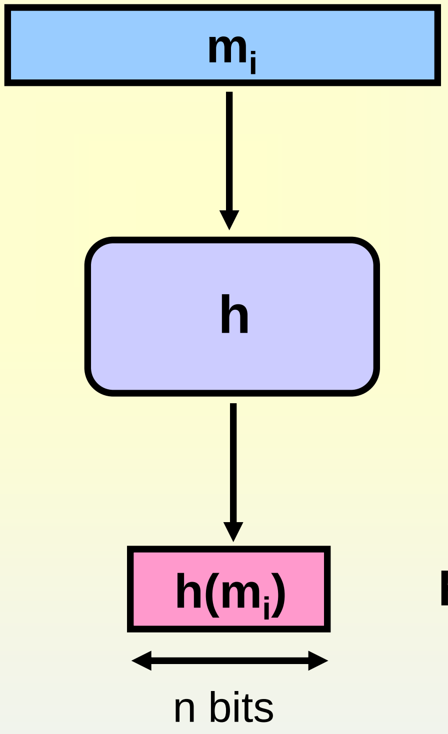

Hash Functions
(Unkeyed)
One-way
Hash functions
(OWHF)
Collision-Resistant
Hash functions
(CRHF)
sufficient for
most other
applications
Required for
digital
signatures
Preimage resistance
2nd preimage resistance
Collision resistance

Brute Force Attack on One-Way Hash Functions

Assume that a signer had signed on a hash value *y.*
→ An attacker tries to frame the signer with a wrong message. Or
→ The signer tries to repudiate his signing.

Given y,
find m such that
h(m) = y

Arbitrary message m
Or
m of the same meaning ?



for i = 1, 2, . . . 2^n

$h(m_i) = y$?

Constructing Multiple Versions of the Same Message

I **state** / **confirm**	thereby that I	**borrowed** / **received**	**$10,000** / **ten thousand dollars**	from	
Mr. / **Dr.**	**Kris** / **Krzysztof**	Gaj on	**October 15,** / **15 October**	2001. This	**money** / **amount of money**
should / **is required to**	be	**returned** / **given back**	to **Mr.** / **Dr.**	Gaj by	**November 30,** / **30 November** 2001.

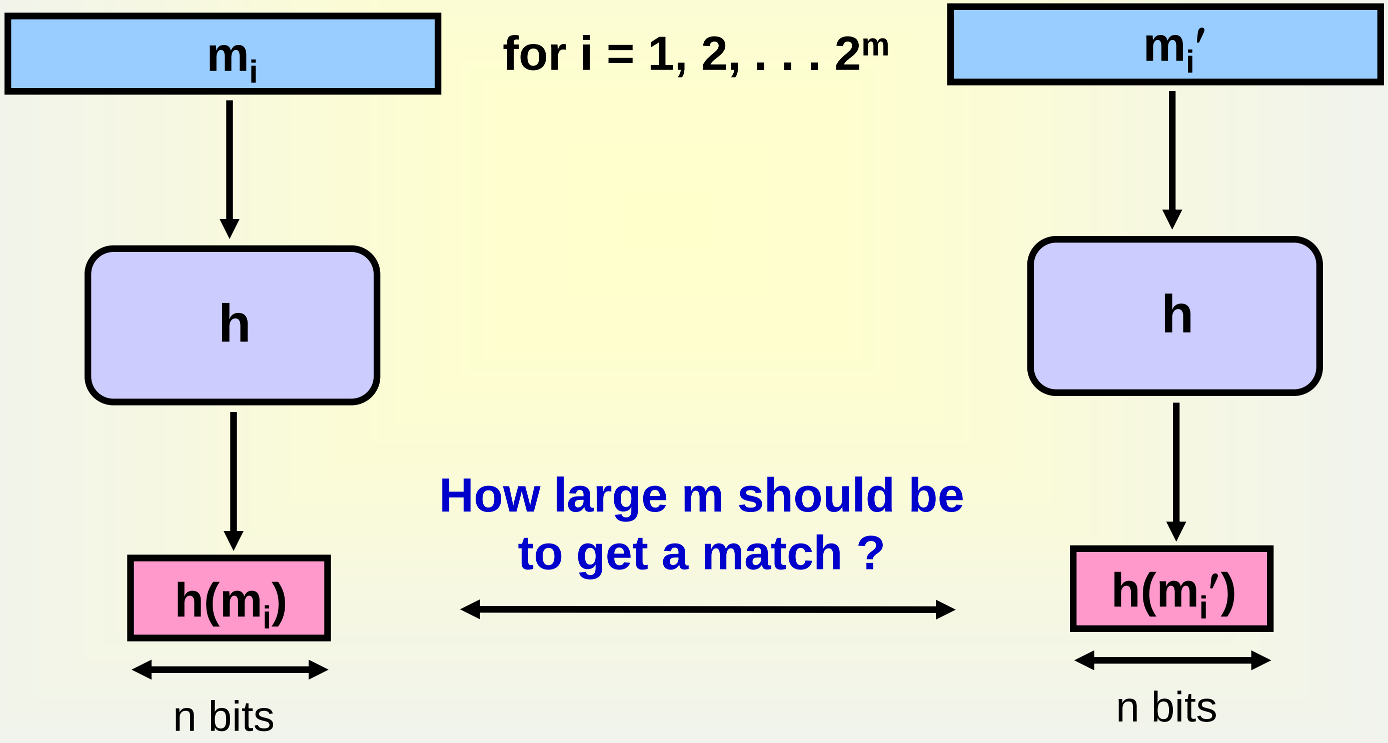
11 different positions of similar expressions

⇓

2^{11} different messages of the same meaning

Finding Collision in Collision-Resistant Hash Functions

Find any two distinct messages m, m′ such that h(m) = h(m′).



Birthday Paradox

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

1. **One of the students shares the teacher's birthday ? (complexity breaking one-wayness)**

   365/2 ≈ 188

2. **Any two of the students share the same birthday ? (complexity breaking collision resistance)**

   $1 - 365 \times 364 \times \ldots \times (365-k+1) / 365^k > 0.5 \Rightarrow k \approx 23$

In general, the probability of a match being found when k samples are randomly selected between 1 and n equals

$$1-\frac{n!}{(n-k)!\,n^k} > 1-e^{-\frac{k(k-1)}{2n}}$$

Birthday Attack

Consider two sets of k instances: X = {x_1, x_2, …, x_k} ; Y = {y_1, y_2, …, y_k},

$$\Pr[\text{no match in } Y \text{ to } x_1] = \left(1 - \frac{1}{n}\right)^k$$

$$\Pr[\text{no match in } Y \text{ to } X] = \left(\left(1 - \frac{1}{n}\right)^k\right)^k$$

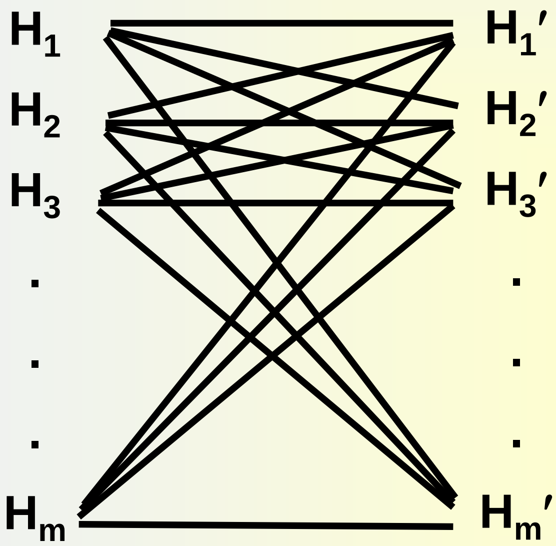
$$\Pr[\text{at least one match in } Y \text{ to } X] = 1 - \left(\left(1 - \frac{1}{n}\right)^k\right)^k > 1 - e^{-\frac{k^2}{n}}$$

The value of k making the probability of at least one match being greater than 0.5

$$\frac{1}{2} = 1 - e^{-\frac{k^2}{n}} \Rightarrow 2 = e^{\frac{k^2}{n}} \Rightarrow \ln 2 = \frac{k^2}{n}$$

$$k = \sqrt{n \ln 2} = 0.83\sqrt{n} \approx \sqrt{n}$$

Birthday Attack on Collision Search



- Number of comparisons = m^2
- Suppose that digest size = n bits
- Intuitively,
  - Hash values can take 2^n possible values
  - generate two sets of m=$2^{n/2}$ hash values
  - compare each element of the two sets
  - there is 2^n comparisons
  - probably there will be one match
- But, more exactly,
  - 1.66 x $2^{n/2}$ hash values are required to find a match with prob.>0.5 (using the birthday attack)

One Million $ Hardware Brute Force Attack

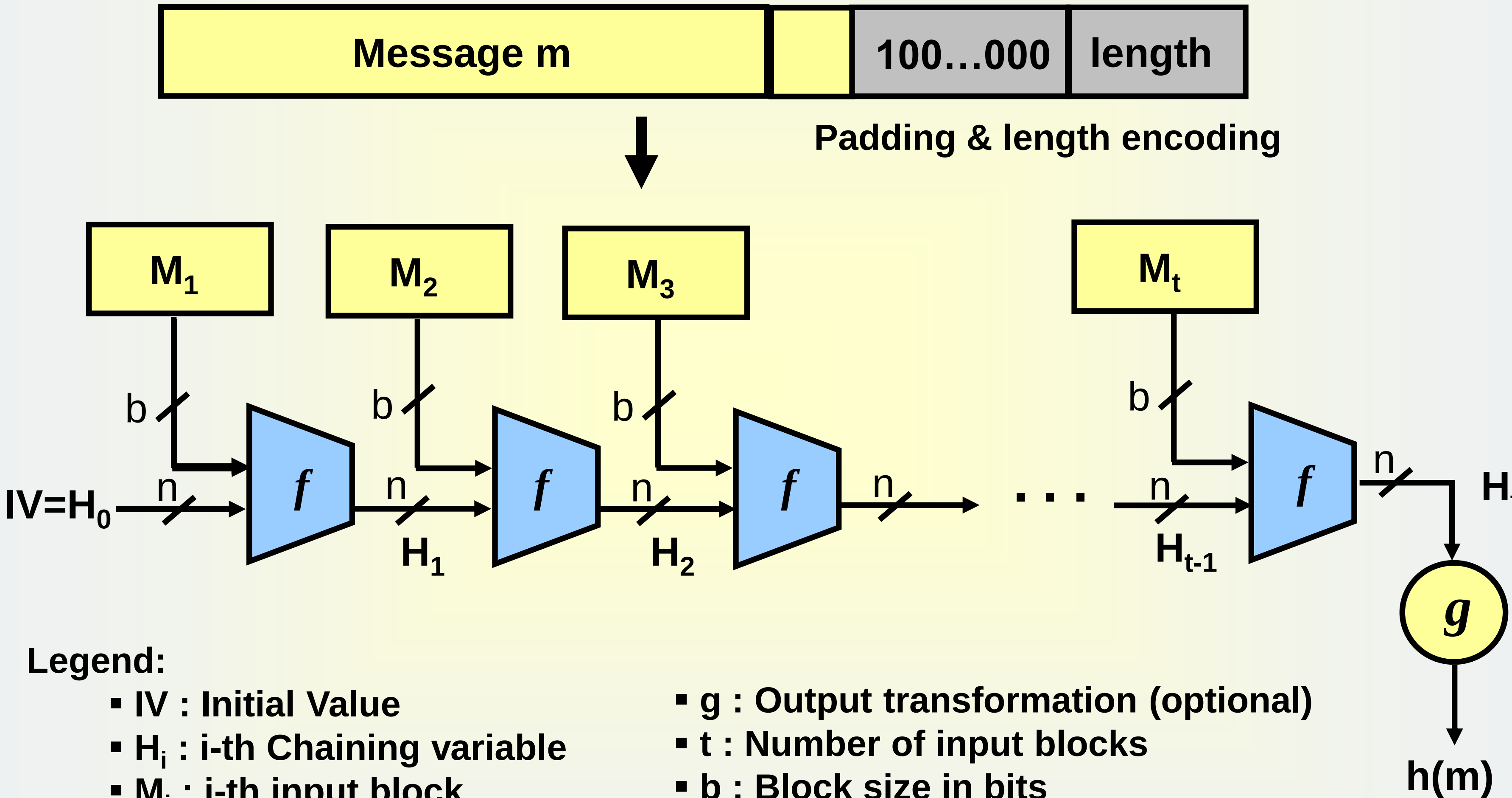
❖ **One-Way Hash Functions (complexity = 2^n)**

	n = 64	n = 80	n = 128
Year 2001	4 days	718 years	10^{17} years

❖ **Collision-Resistant Hash Functions (complexity = $2^{n/2}$)**

	n = 128	n = 160	n = 256
Year 2001	4 days	718 years	10^{17} years

General Construction of a Secure Hash Function

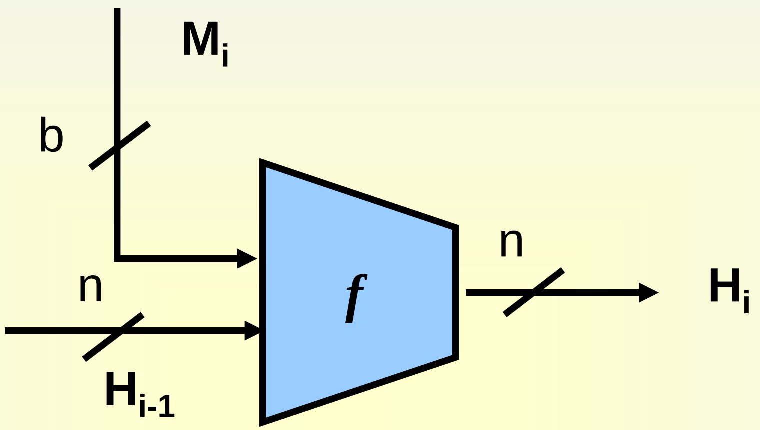


Legend:

- IV : Initial Value
- H_i : i-th Chaining variable
- M_i : i-th input block
- f : Compression function
- g : Output transformation (optional)
- t : Number of input blocks
- b : Block size in bits
- n : Hash code size in bits

General Construction of a Secure Hash Function

Compression Function
(**fixed-size hash function**)



Entire hash

$H_0 = IV$

$H_i = f\,(H_{i-1}, M_i)$ for $1 \le i \le t$

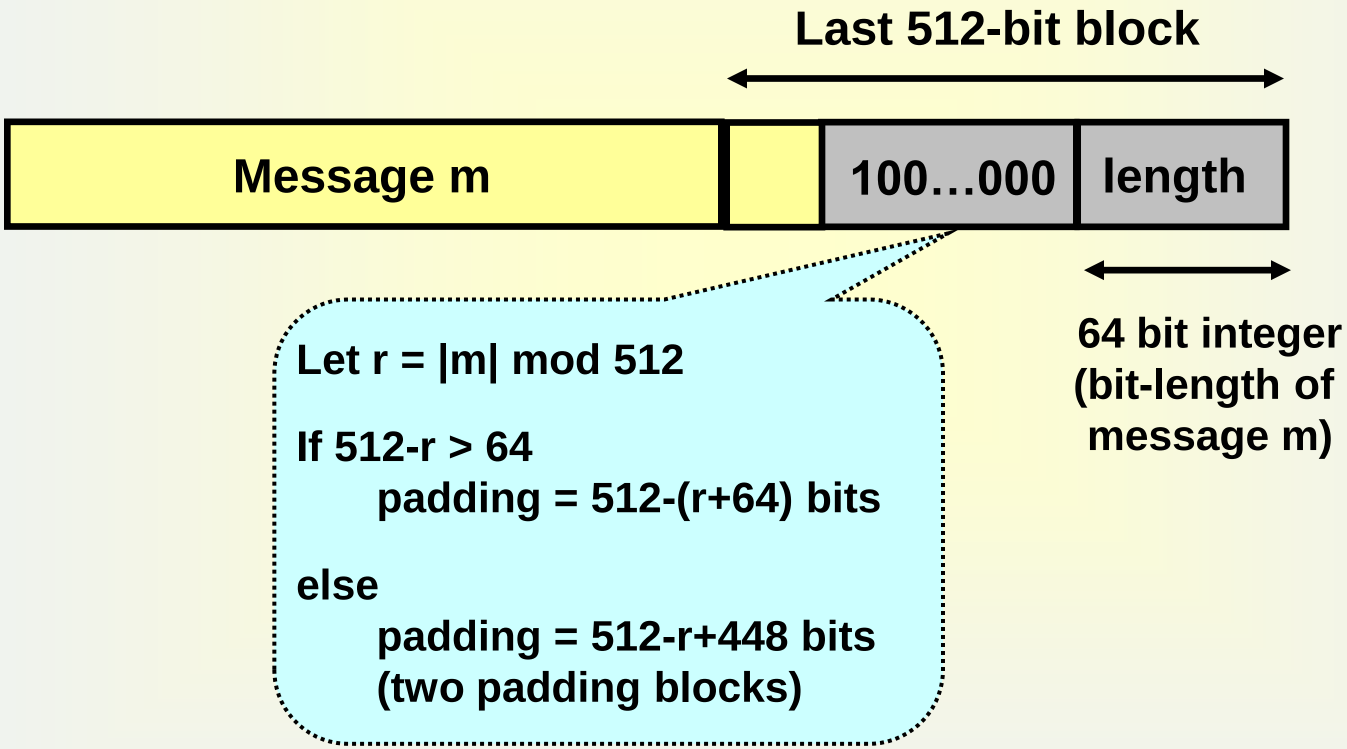
$H(m) = g(H_t)$

Fact(by Merkle-Damgård)

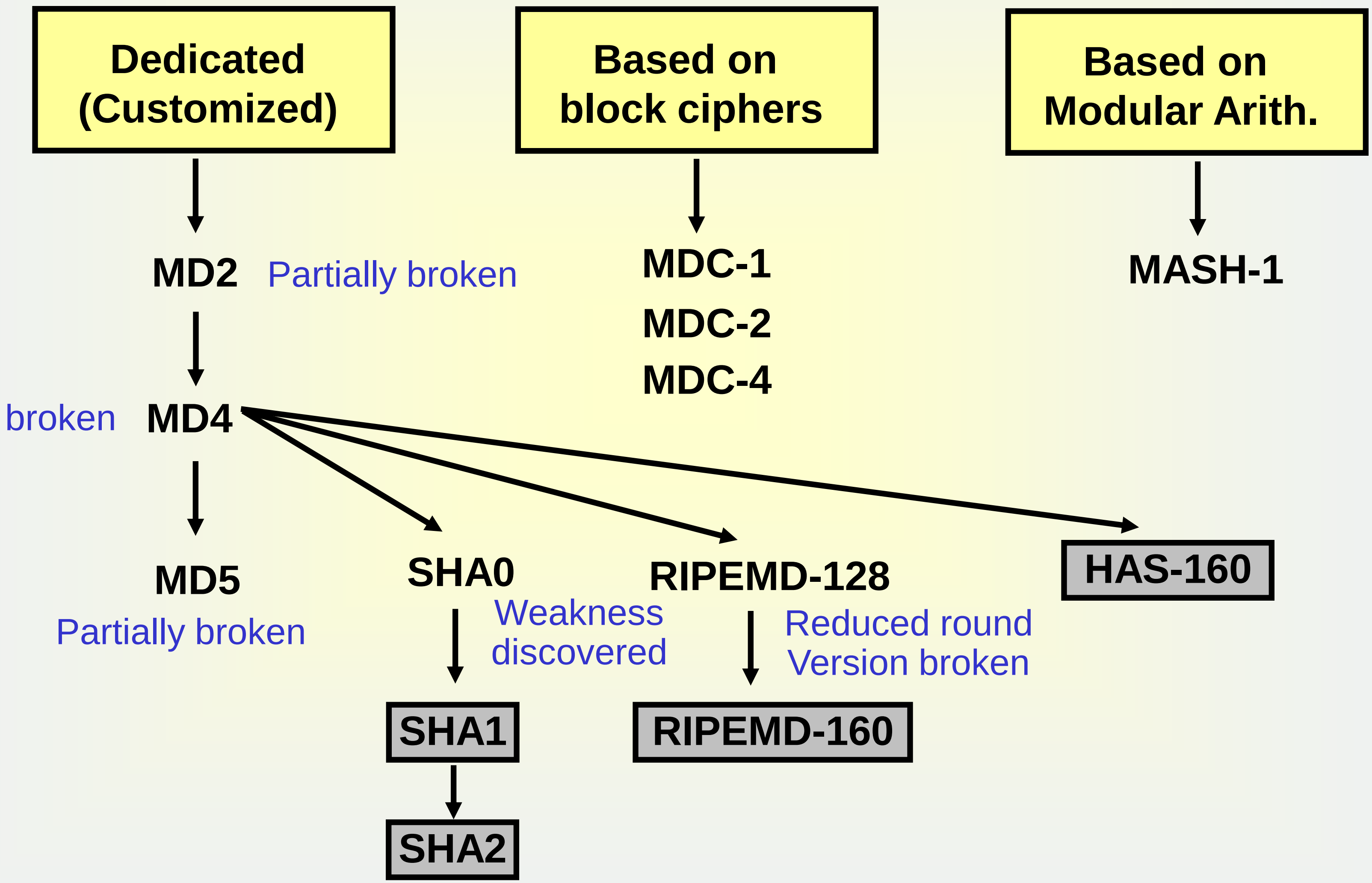
Any collision-resistant compression function f can be extended to a collision-resistant hash function h

Typical Hash Padding

❖ Assume Block size = 512 bits (MD5, SHA1, RMD160, HAS160 …)



Classification of Hash Functions



SHA (Secure Hash Algorithm)

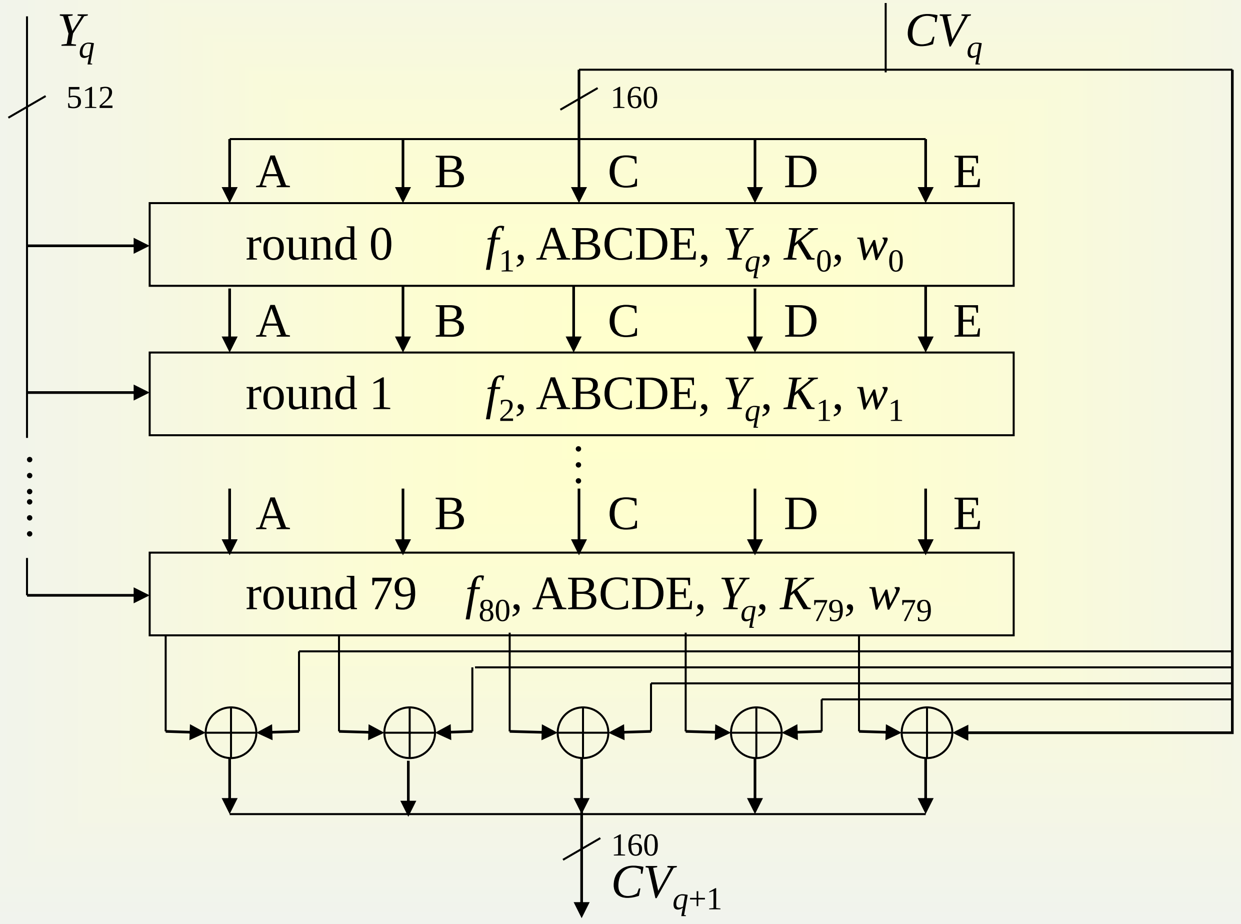
- **SHA was designed by NIST (national institute of standards and technology) & NSA (National Security Agency) in 1993, and revised as SHA-1 in 1995**
  - **SHA: FIPS PUB 180, 1993**
  - **SHA-1 : FIPS Pub 180-1, 1995**

* Federal Information Processing Standard

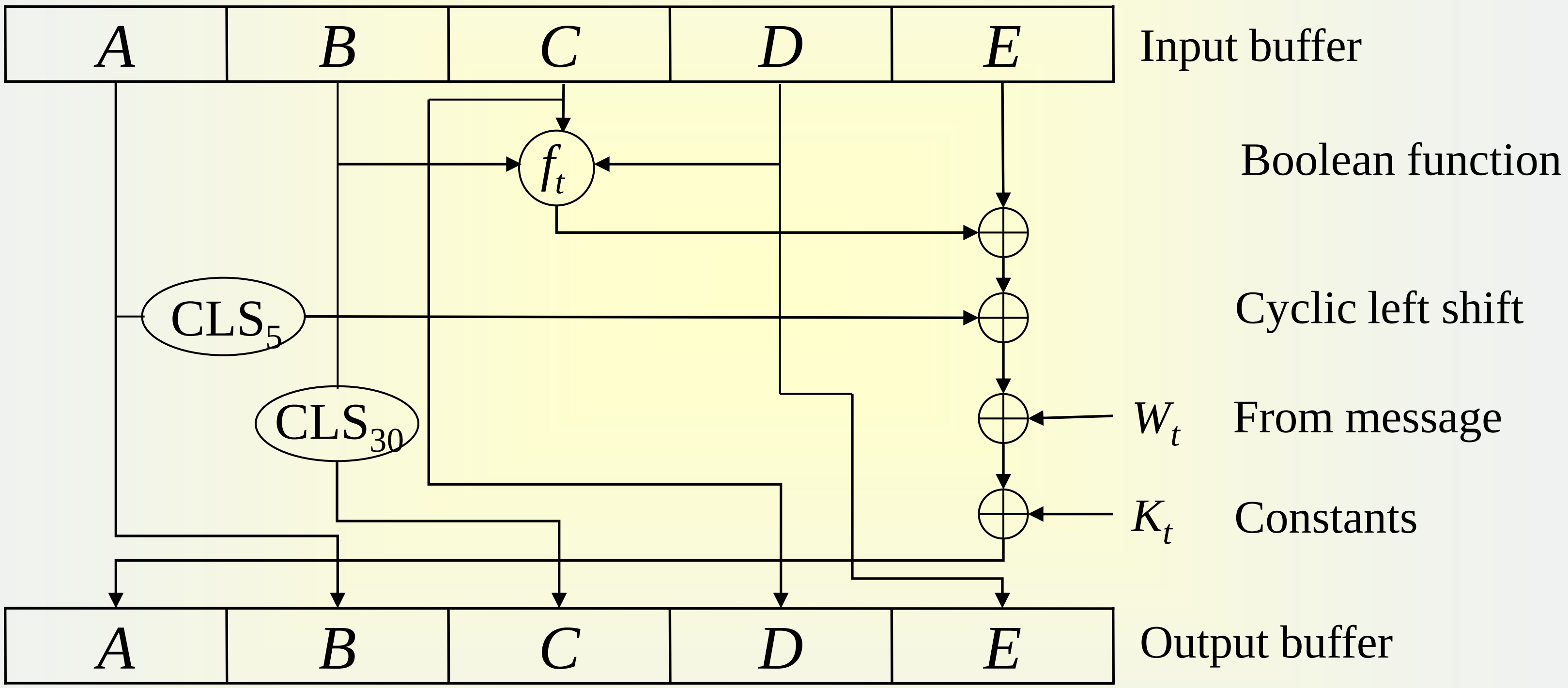
- **US standard for use with DSA signature scheme**
- **The algorithm is SHA, the standard is SHS**
- **Based on the design of MD4 with key differences**

- **SHA-1 : *Secure Hash Standard (SHS),* FIPS Pub 180-1, 1995**
  - **160-bit hash value (5 words Big Endian)**
  - **512-bit block size**
  - **4 round hash, each round has 20 steps, total 80 steps**

SHA-1 Overview



SHA-1 round function



SHA-1

Initial values

A = 6 7 4 5 2 3 0 1

B = E F C D A B 8 9

C = 9 8 B A D C F E

D = 1 0 3 2 5 4 7 6

E = C 3 D 2 E 1 F 0

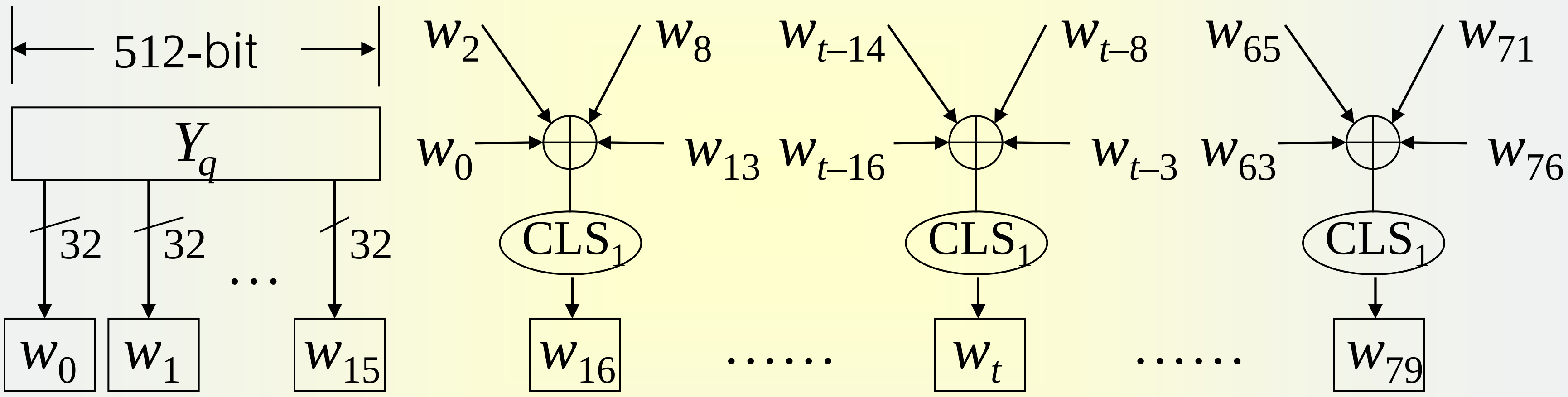
Constants K_t

$t = 0 \sim 19$	K_t = 5 A 8 2 7 9 9 9
$t = 20 \sim 39$	K_t = 6 E D 9 E B A 1
$t = 40 \sim 59$	K_t = 8 F 1 B B C D C
$t = 60 \sim 79$	K_t = C A 6 2 C 1 D 6

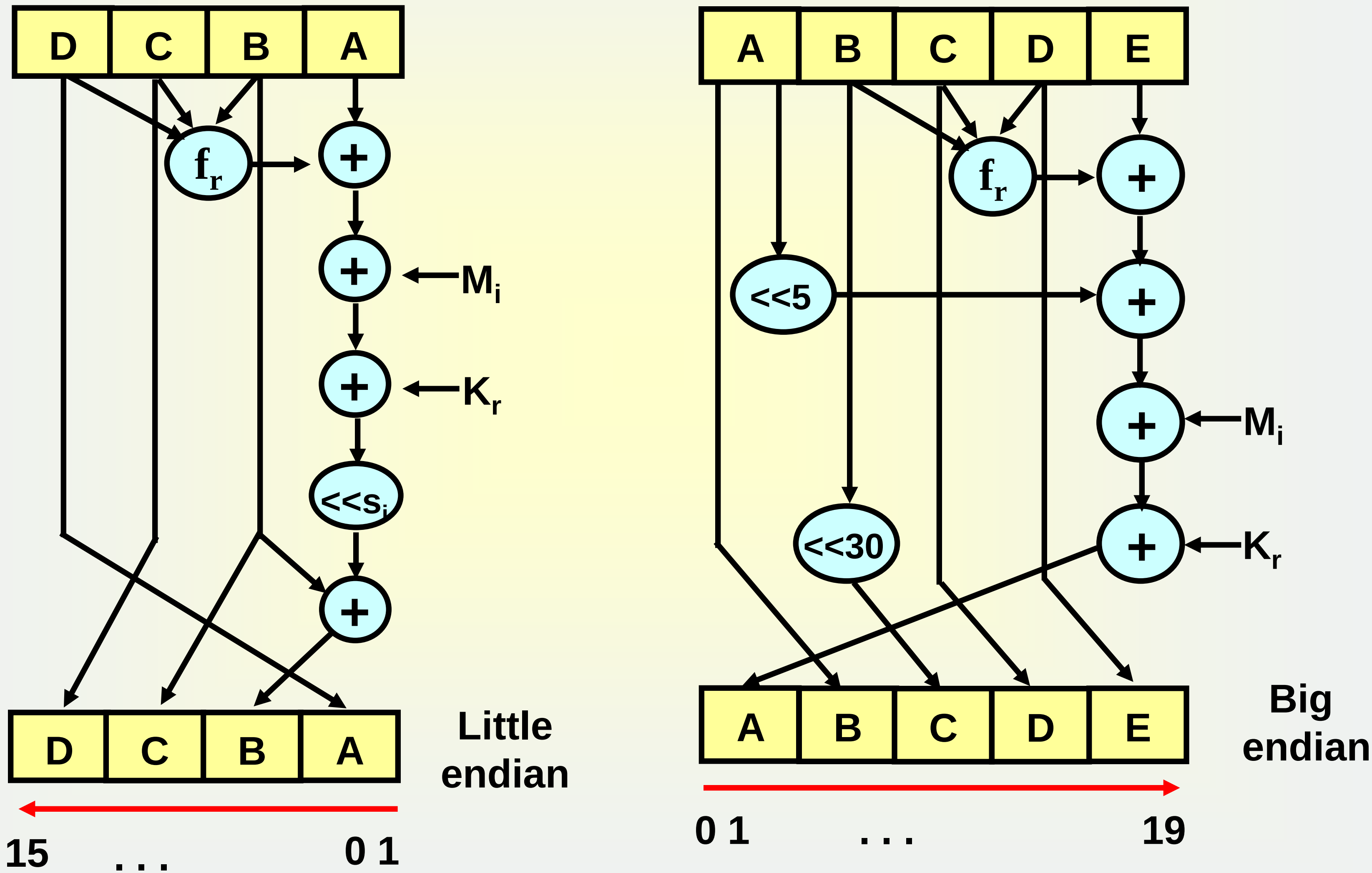
Boolean function f_t

$t = 0 \sim 19$	$f_t (B, C, D) = B \cdot C + \overline{B} \cdot D$
$t = 20 \sim 39$	$f_t (B, C, D) = B \oplus C \oplus D$
$t = 40 \sim 59$	$f_t (B, C, D) = B \cdot C + B \cdot D + C \cdot D$
$t = 60 \sim 79$	$f_t (B, C, D) = B \oplus C \oplus D$

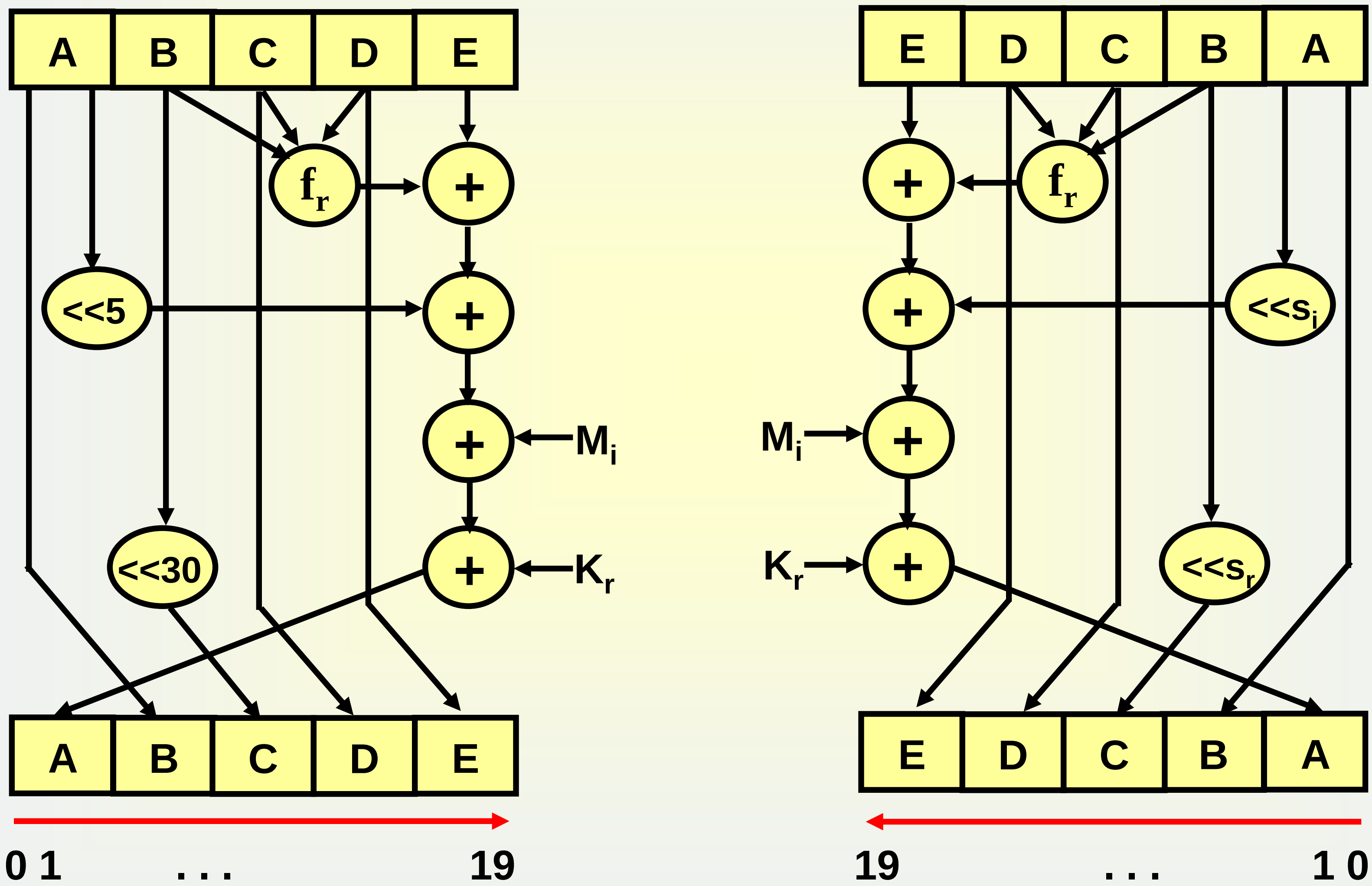
SHA-1 message inputs



Step Operations of MD5 & SHA1



Step Operations of SHA1 & HAS160

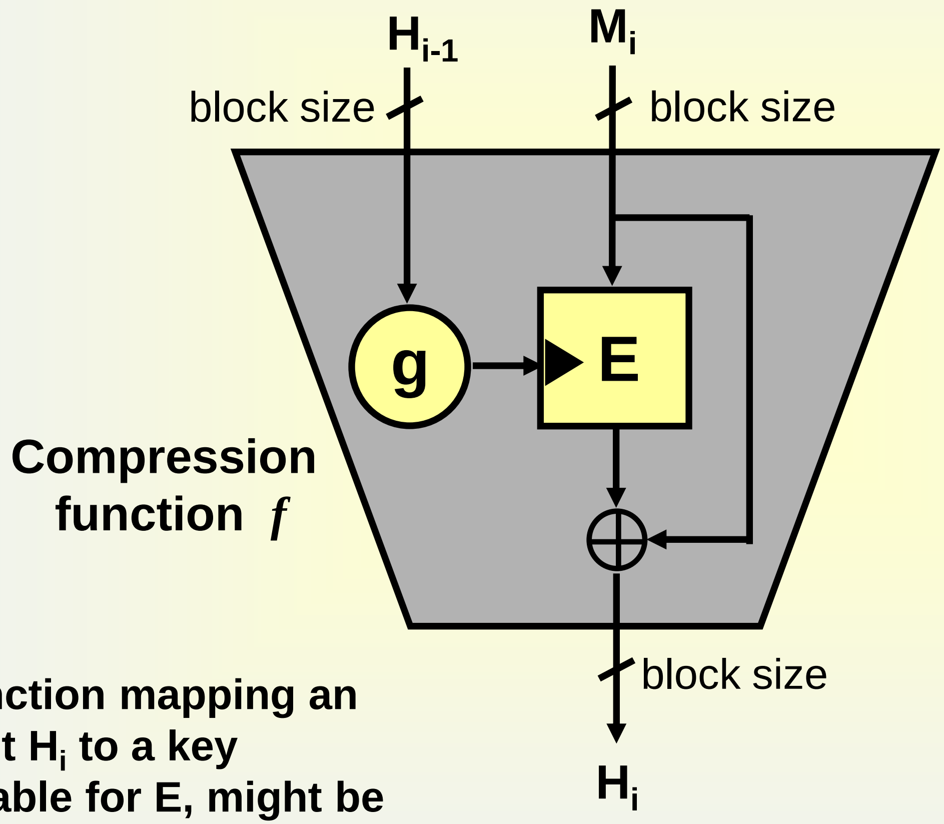


Comparison of Popular Hash Functions

Hash Func.	MD5	SHA1	RMD160	HAS160
Digest size(bits)	128	160	160	160
Block size(bits)	512	512	512	512
No of steps	64(4x16)	80(4x20)	160(5x2x16)	80(4x20)
Boolean func.	4	4(3)	5	4(3)
Constants	64	4	9	4
Endianness	Little	Big	Little	Little
Speed ratio	1.0	0.57	0.5	0.94

Hash Functions Based on Block Ciphers: MDC1

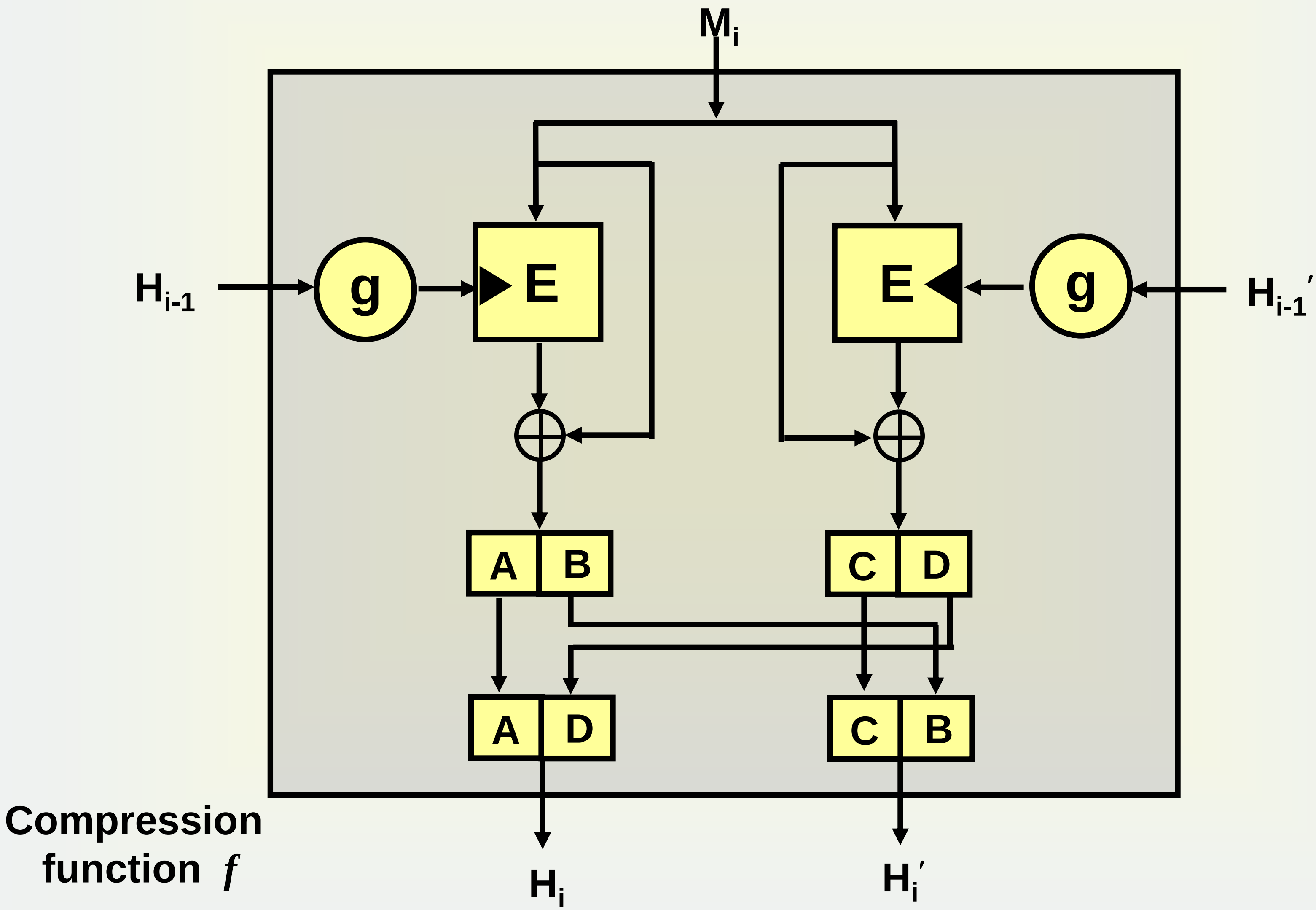
Matyas-Meyer-Oseas Scheme



g: a function mapping an input H_i to a key suitable for E, might be the identity function

- Provably Secure under an appropriate black-box model
- But produces too short hash codes for use in most applications

Hash Functions Based on Block Ciphers: MDC2



Hash Functions – Implementation results

PIII 450MHz : Widows 98 : MSVC++ 6.0

	Output len.	64 bytes	1K bytes	1M bytes
SHA1	160 bits	101.7 Mbps	200.8 Mbps	214.9 Mbps
SHA-256	256 bits	48.2 Mbps	97.6 Mbps	104.1 Mbps
SHA-512	512 bits	16.0 Mbps	28.8 Mbps	32.8 Mbps
RMD160	160 bits	91.1 Mbps	174.9 Mbps	188.1 Mbps
HAS160	160 bits	158.6 Mbps	328.7 Mbps	353.0 Mbps
Tiger	192 bits	51.0 Mbps	98.8 Mbps	106.3 Mbps
MD5	128 bits	176.5 Mbps	349.8 Mbps	376.3 Mbps

- **Remarks**
  - ✓ Theoretical strength of hash functions with k-bit output : about $2^{k/2}$
  - ✓ 128-bit Hash function does not offer sufficient protection
    - ➔ Use of MD5 not recommended
  - ✓ SHA1 only provides 80-bit security
  - ✓ AES offers three key sizes (128, 192, 256)
    - ➔ Need for companion hash algorithms to give similar security
    - ➔ SHA-(256, 384, 512) have been proposed (draft FIPS)

3. Message Authentication Code

MAC Functions

- **MAC algorithms**
  - **Keyed hash functions whose specific purpose is message authentication**

- **Requirements**
  - **Ease of computation**
  - **Compression**

    arbitrary length input to fixed length output
  - **Computation resistance**

    Given zero or more text-MAC pairs $(m_i, MAC_K(m_i))$,
    It's computationally infeasible to find any new text-MAC pair $(m, MAC_K(m))$ for $m \neq m_i$

Attacks & Forgeries on MAC Algorithms

- **Adversary's goal**
  - **MAC key recovery**
  - **MAC forgery**

- **Attacks on MAC algorithms**
  - **Known-text attack**
  - **Chosen text attack**
  - **Adaptively chosen text attack**

- **MAC Forgeries**
  - **Selective forgery**
  - **Existential forgery**

Classification of MAC Algorithms

Based on Hash functions
- HMAC
- KMAC

Based on block ciphers
- CBC-MAC
- CFB-MAC
- RIPE-MAC

Based on Stream ciphers
- CRC-MAC

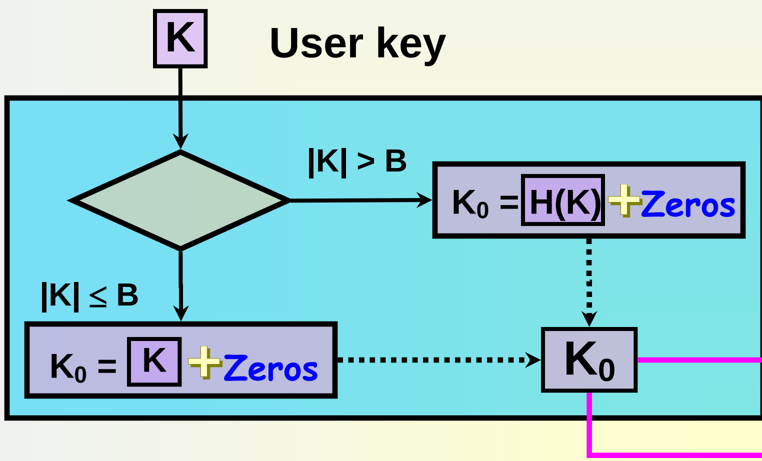
Dedicated
- MAA

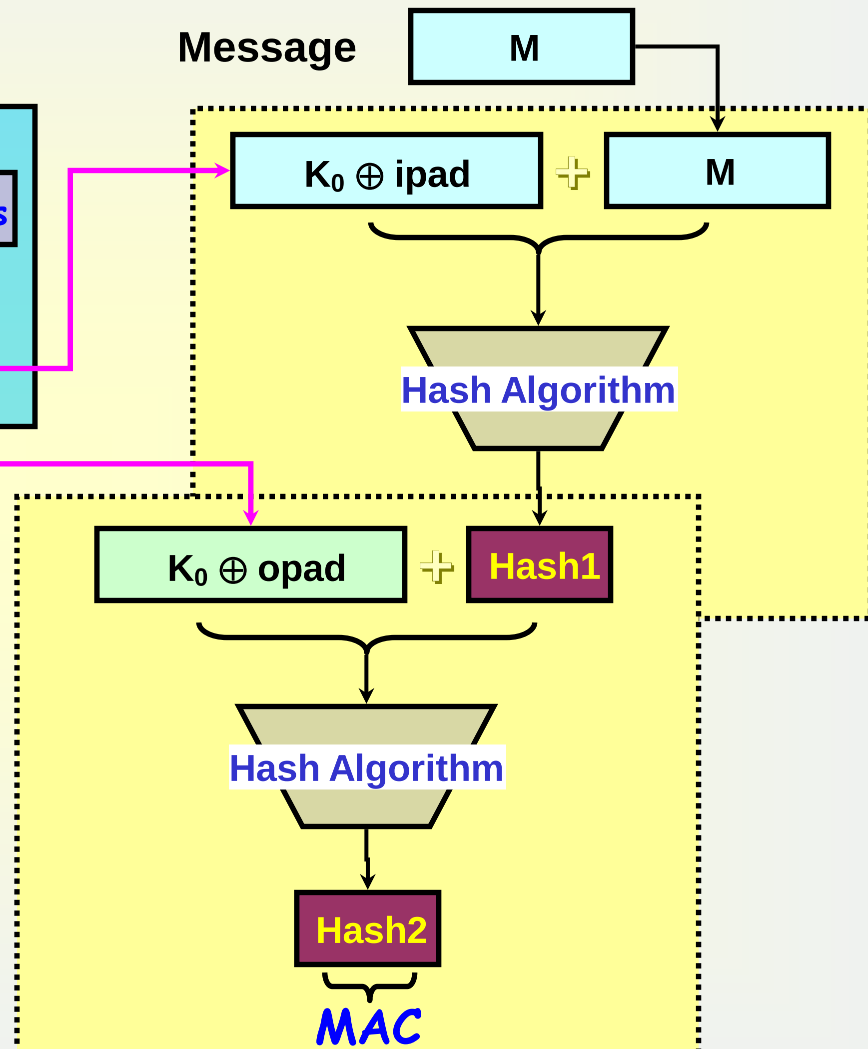
Constructing MAC from Hash Functions

- **Secret Prefix Method**
  - **$MAC_K(M) = h(K \| M)$**
  - **Extension attack: easy to generate MAC for $\underline{M}$ = M || P || M′ (P: hash padding)**
- **Secret Suffix Method**
  - **$MAC_K(M) = h(M \| K)$**
  - **Birthday attack applies: if h(M) = h(M′), then $MAC_K(M) = MAC_K(M')$**
- **Envelope Method**
  - **$MAC_K(M) = h(K \| P \| M \| K)$ (P=padding to make K || P one block)**
  - **No known weakness yet**
- **HMAC**
  - **Provably secure under some ideal assumptions on the underlying hash function**

http://csrc.nist.gov/publications/fips/fips198/fips-198a.pdf

HMAC : Keyed-Hash MAC



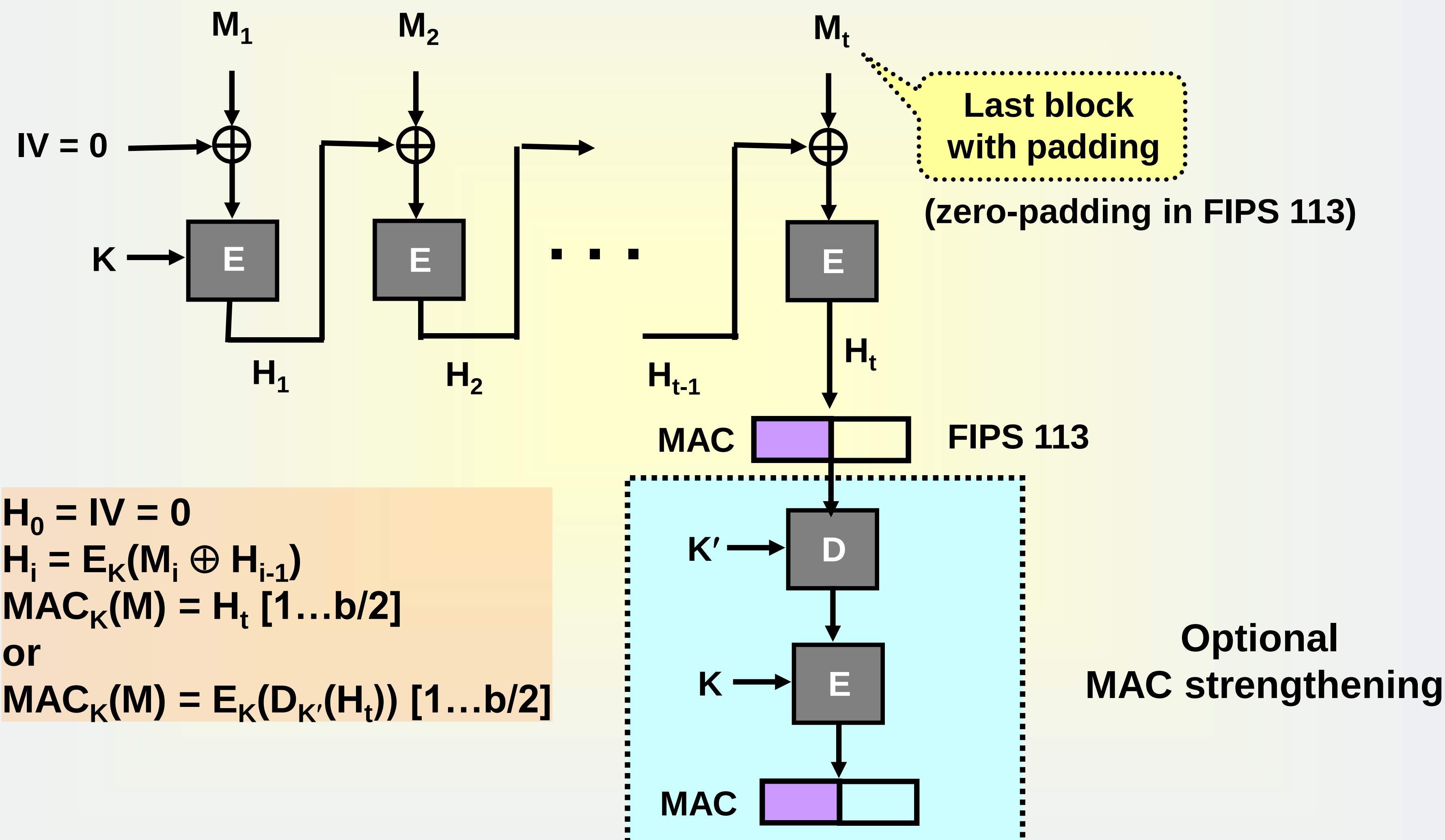


- ➕ : concatenation
- B : Block size of Hash function
  K_0 is ‘B’ bytes
- ipad : 0x36 repeated B times
- opad : 0x5C repeated B times

MAC =
$H(K_0 \oplus opad \,||\, H(K_0 \oplus ipad \,||\, M))$

❖ Used in SSL/TLS and IPsec

MAC Based on Block Ciphers: CBC-MAC



$H_0 = IV = 0$

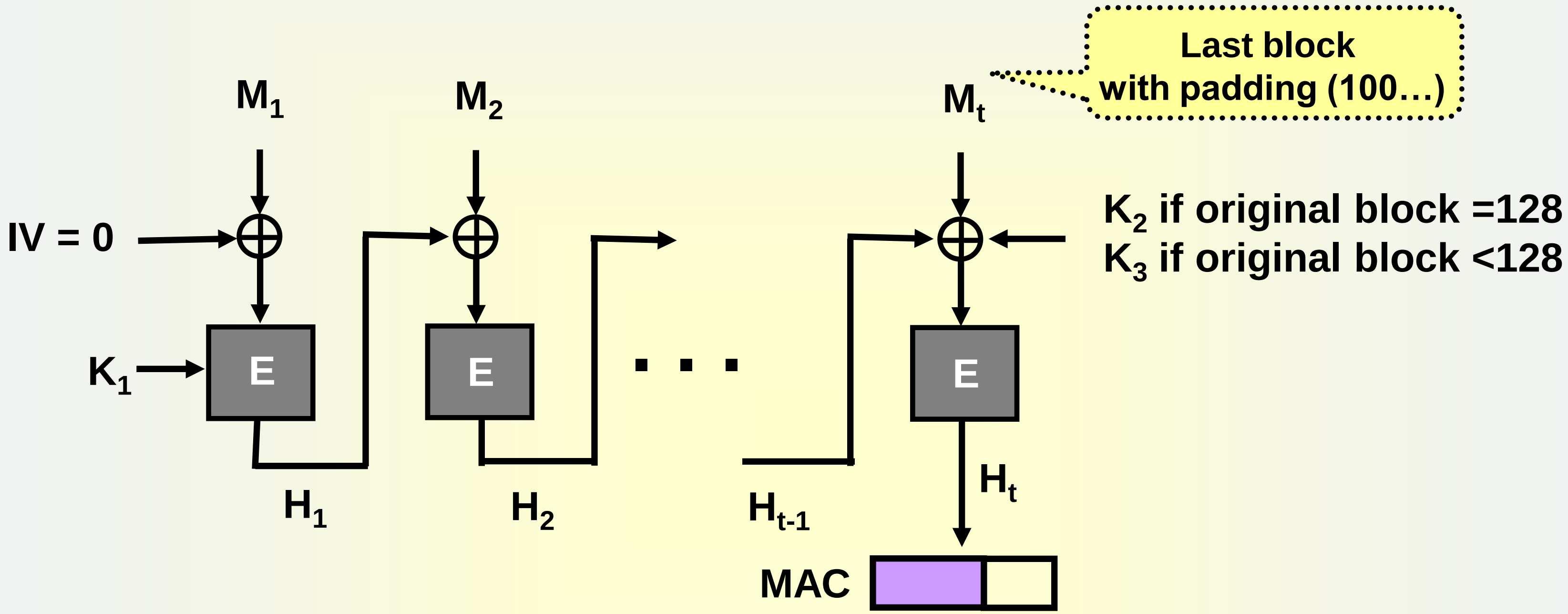
$H_i = E_K(M_i \oplus H_{i-1})$

$MAC_K(M) = H_t\ [1 \ldots b/2]$

or

$MAC_K(M) = E_K(D_{K'}(H_t))\ [1 \ldots b/2]$

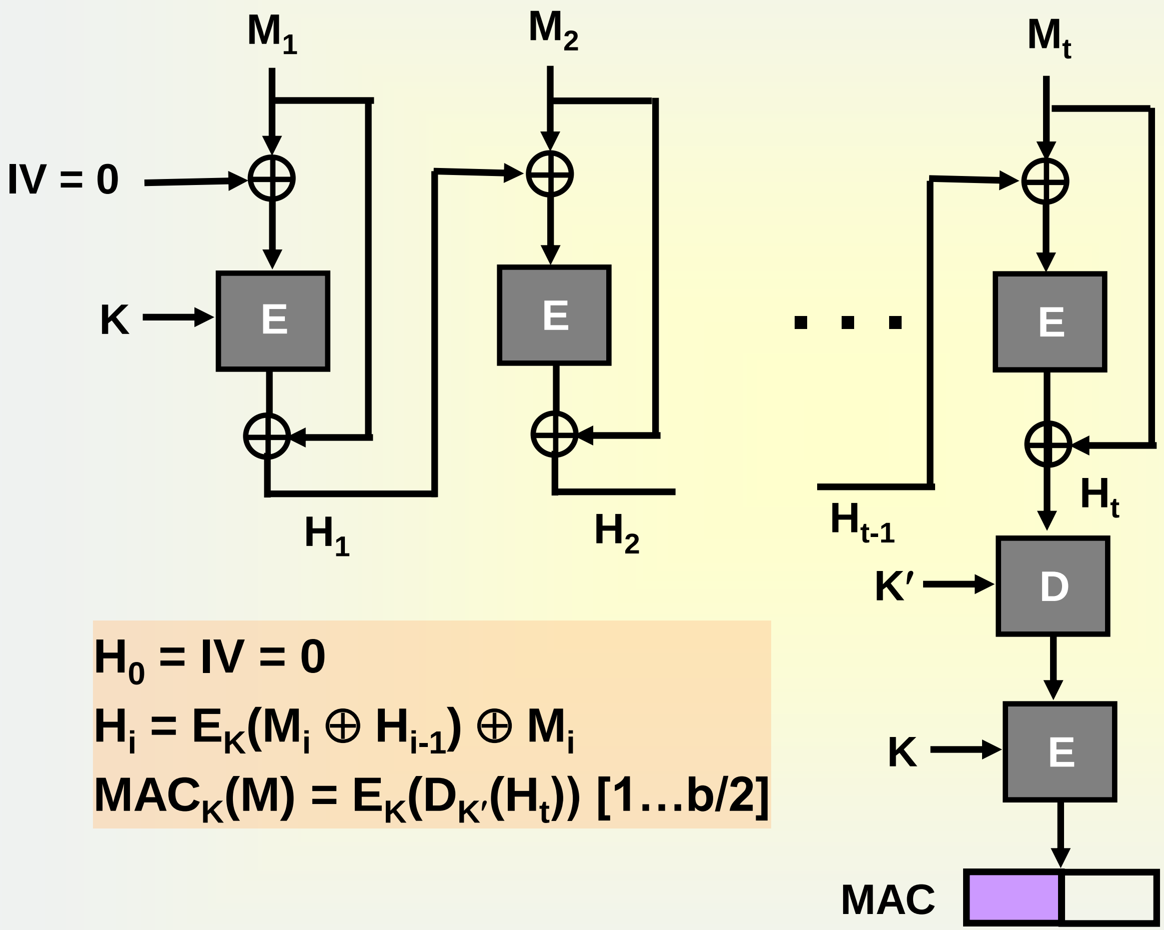
MAC Based on Block Ciphers: XCBC-MAC



Key derivation from the secret key K (Ipsec:AES-XCBC-MAC-96)

$K_1 = E_K(0x01010101010101010101010101010101)$
$K_2 = E_K(0x02020202020202020202020202020202)$
$K_3 = E_K(0x03030303030303030303030303030303)$

MAC Based on Block Ciphers: RIPE-MAC



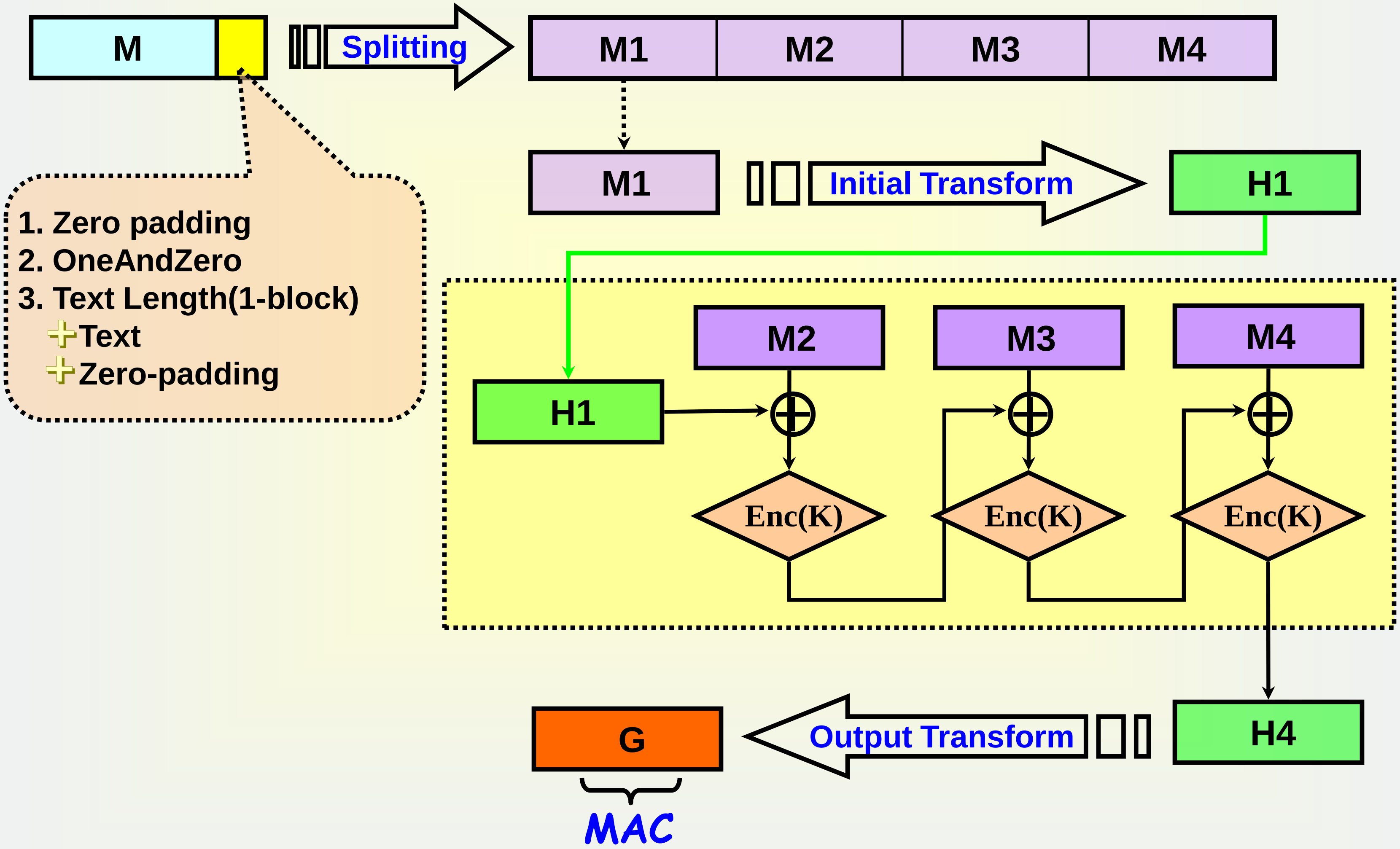
$H_0 = IV = 0$

$H_i = E_K(M_i \oplus H_{i-1}) \oplus M_i$

$MAC_K(M) = E_K(D_{K'}(H_t))\ [1 \ldots b/2]$

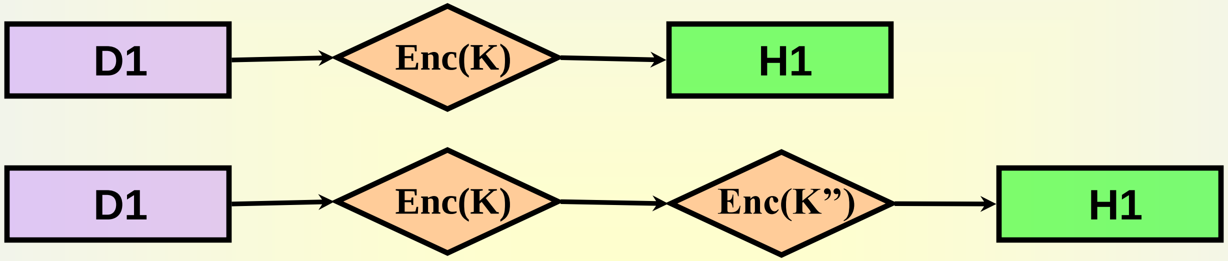
- Padding:
  one-zero padding (possible none) + length block
- $K' = K \oplus$ 0x 0f0f…0f

CBC-MAC : ISO 9797-1

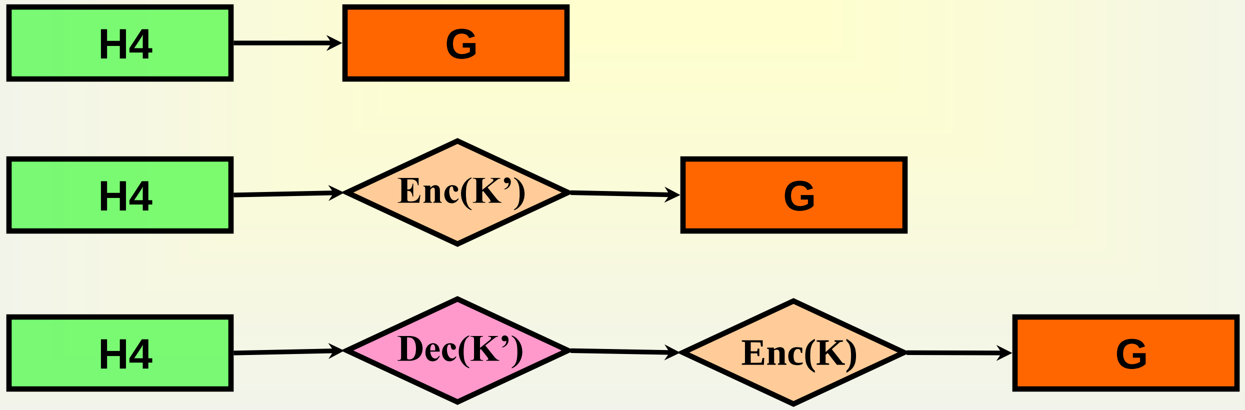


CBC-MAC : Transformation

- **Initial Transformation**



- **Output Transformation**



Homework #4

- **Birthday Paradox**

A professor posts the grades for a class using the last four digits of the social security number of each student. Assume that the social security numbers are randomly distributed.
In a class of 200 students, what is the probability that at least two students have a collision problem (the same four digits)?