

Week 12: Hash Functions and MAC

1. Introduction

Hash Functions vs. MAC

Hash Functions

❖ Hash Function

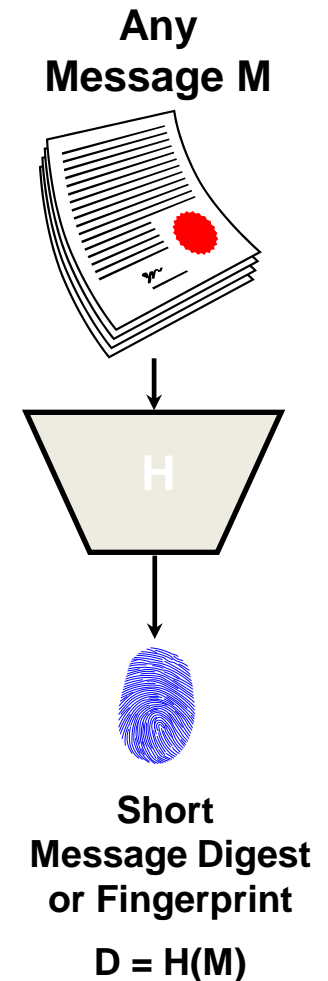
- ✓ Generate a fixed length "Fingerprint" for an arbitrary length message.
- ✓ No Key involved.
- ✓ Must be at least One-way to be useful.

❖ Applications

- ✓ Keyed hash: MAC/ICV generation.
- ✓ Unkeyed hash: digital signature, password file, key stream / pseudo-random number generator, etc.

❖ Constructions

- ✓ Iterated hash functions (MD4-family hash functions): MD5, SHA-1, SHA-2, HAVAL, HAS160, etc.
- ✓ Hash functions based on block ciphers: MDC(Manipulation Detection Code)



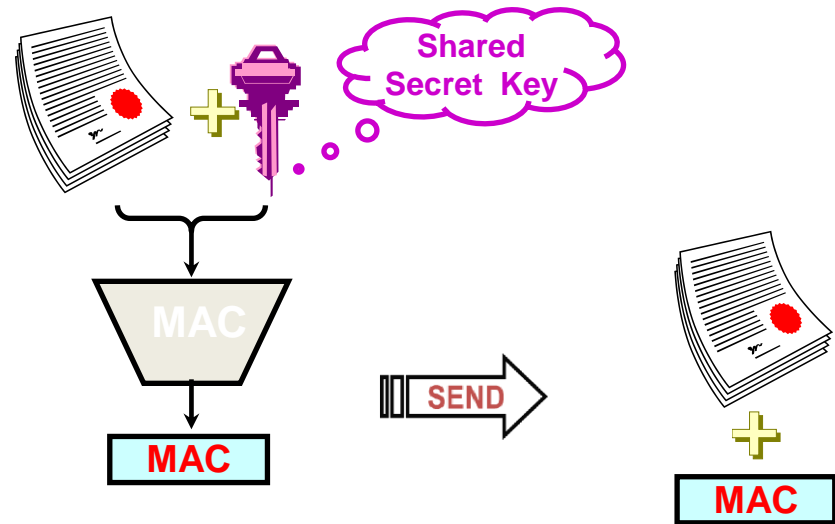
Message Authentication Code (MAC)

➤ MAC

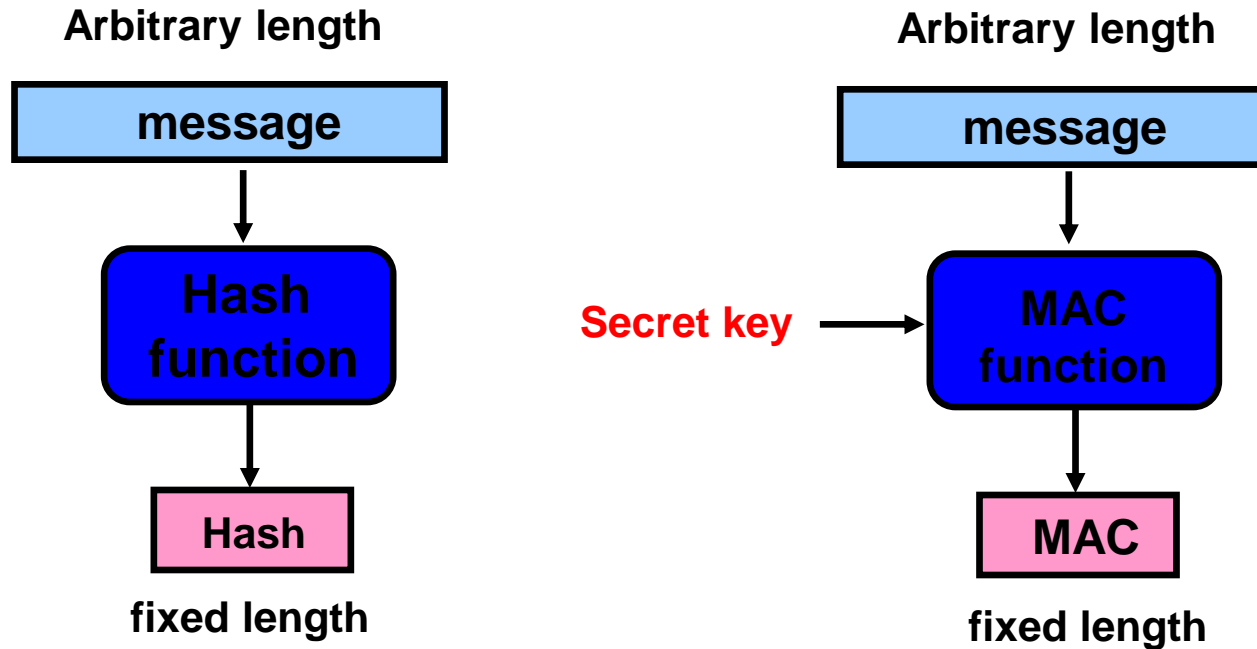
- ✓ Generate a fixed length MAC for an arbitrary length message
- ✓ A **keyed** hash function
- ✓ Message origin authentication
- ✓ Message integrity
- ✓ Entity authentication

➤ Constructions

- ✓ Keyed hash: [HMAC](#)
- ✓ Block cipher: CBC-MAC



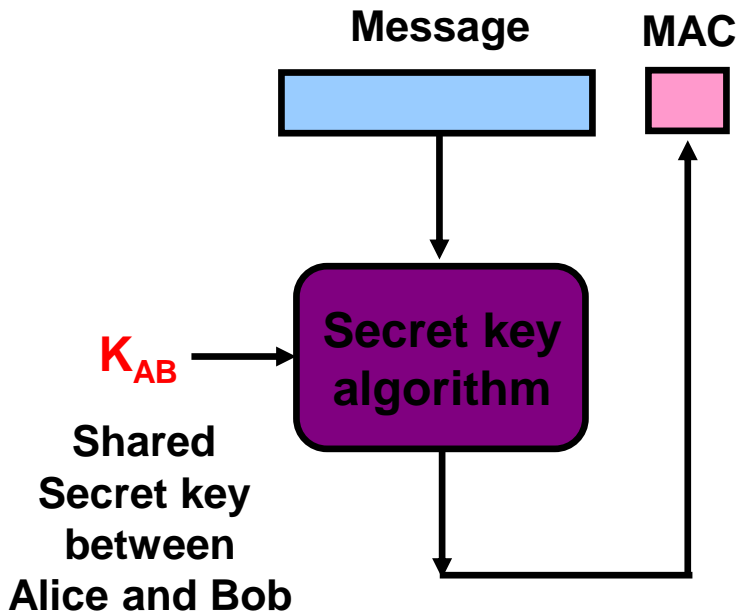
Comparison of Hash Function & MAC



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

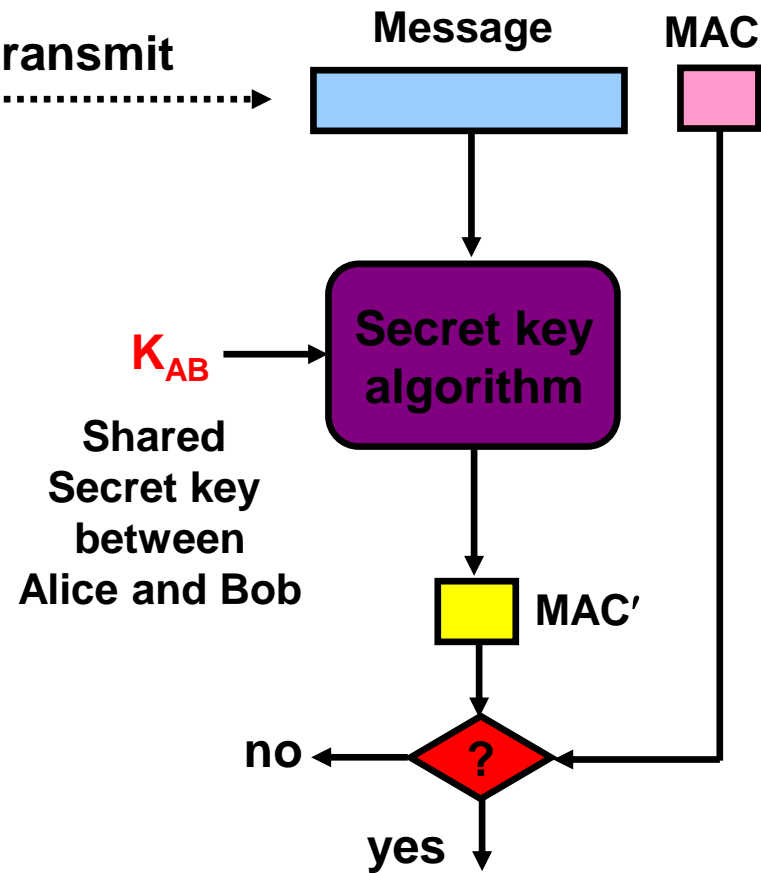
Message Authentication using MAC

Alice



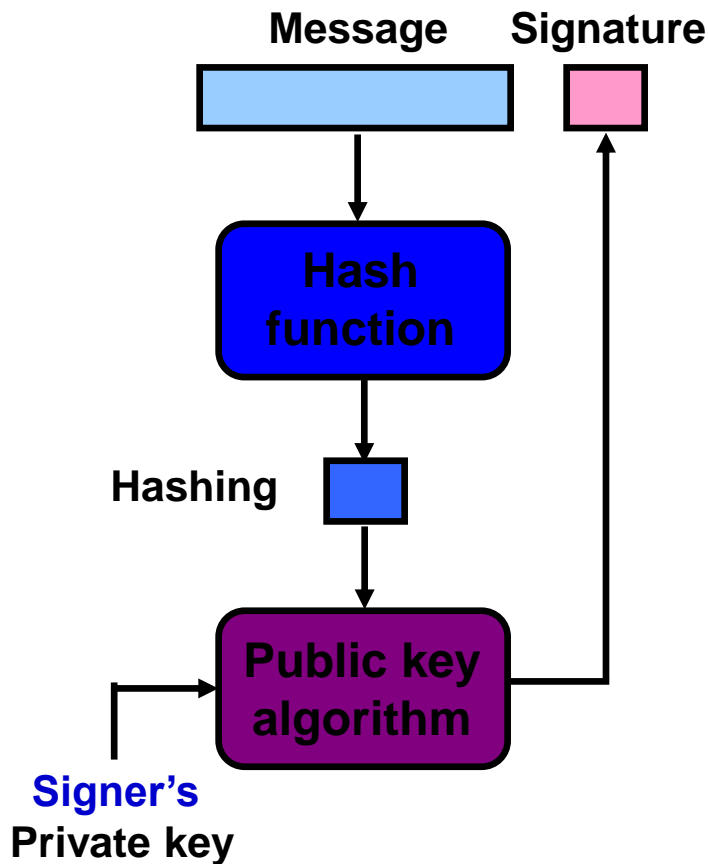
transmit

Bob

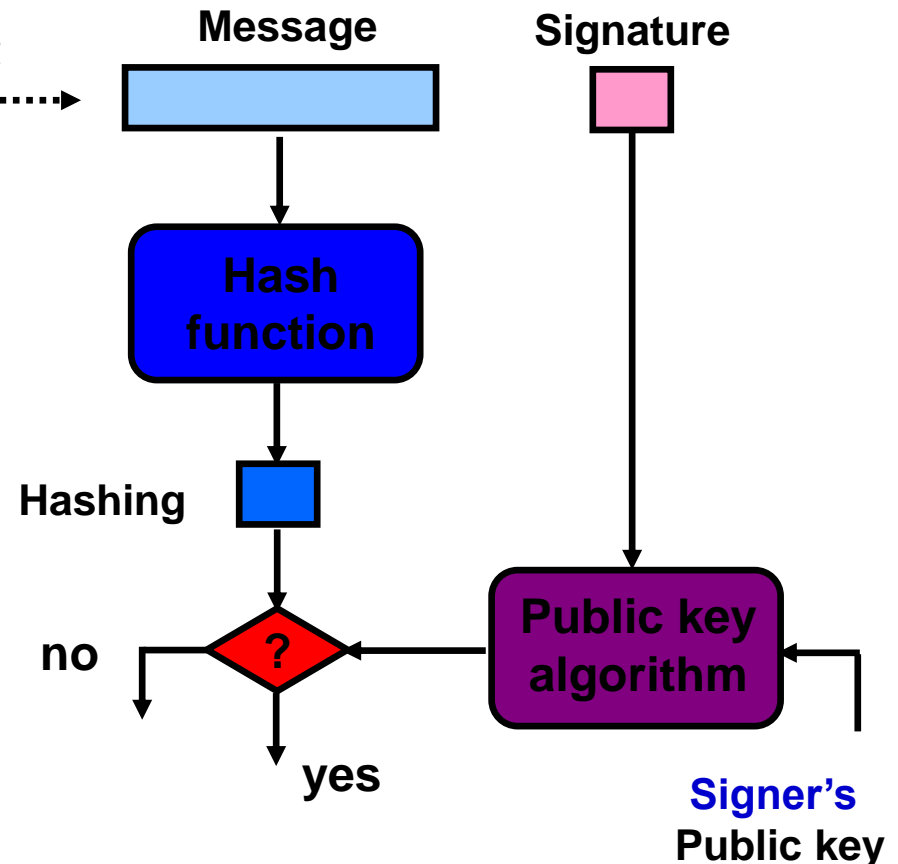


Digital Signature with Hash Function

Signer



Verifier



MAC and Digital Signature (Summary)

❖ MAC (Message Authentication Code)

- Generated and verified by a **secret key** algorithm
- Message origin authentication & Message integrity
- Schemes
 - ✓ Keyed hash: HMAC
 - ✓ Block cipher: CBC-MAC,

❖ 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-PSS, DSA, etc.

2. Hash Functions

Hash Functions – Requirements

❖ Efficient Computation

❖ Security Properties

➤ **Preimage resistance** (One-wayness) :

- Given y , it is computationally infeasible to find any input x such that $y = h(x)$

➤ **2nd preimage resistance** (Weak collision resistance) :

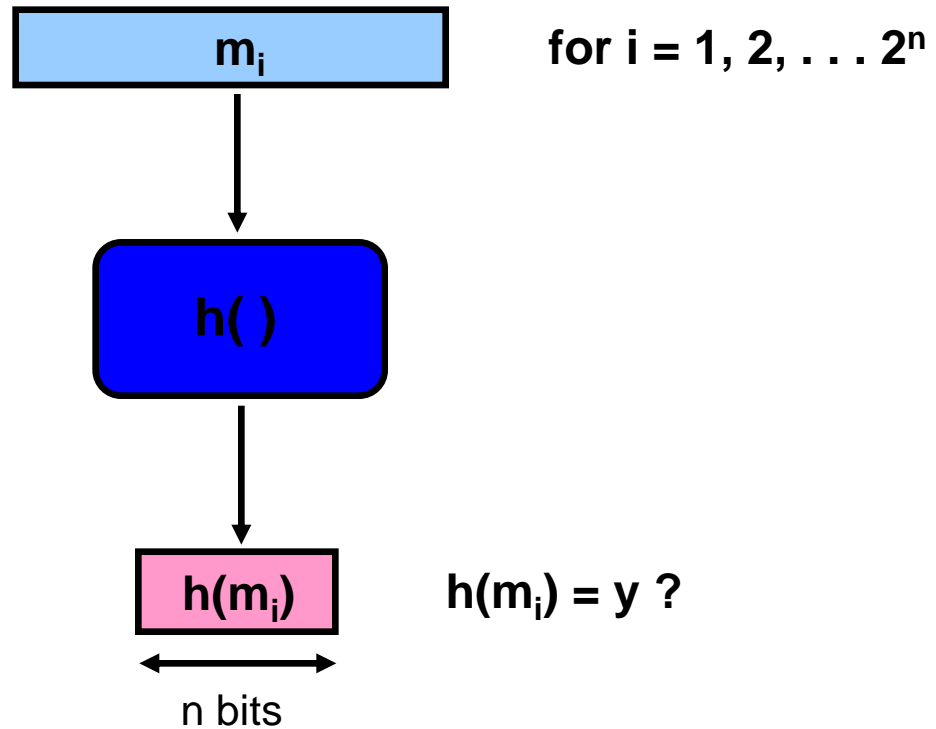
- Given x , it is computationally infeasible to find another input $x' \neq x$ such that $h(x) = h(x')$

➤ **Collision resistance** (Strong collision resistance) :

- It is computationally infeasible to find any two distinct inputs x and x' such that $h(x) = h(x')$

BFA on OW Hash Function (Pre-image Attack)

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



Arbitrary message, m_i
or
 m_j of the same meaning ?

Multiple Messages with Same Meaning

I **state** thereby that I **borrowed** **\$10,000** from
confirm **received** **ten thousand dollars**

Mr. Kris Gaj on **October 15,** 2001. This **money**
Dr. Krzysztof **15 October** **amount of money**

should be **returned** to **Mr. Gaj** by **November 30,** 2001.
is required to **given back** **Dr.** **30 November**

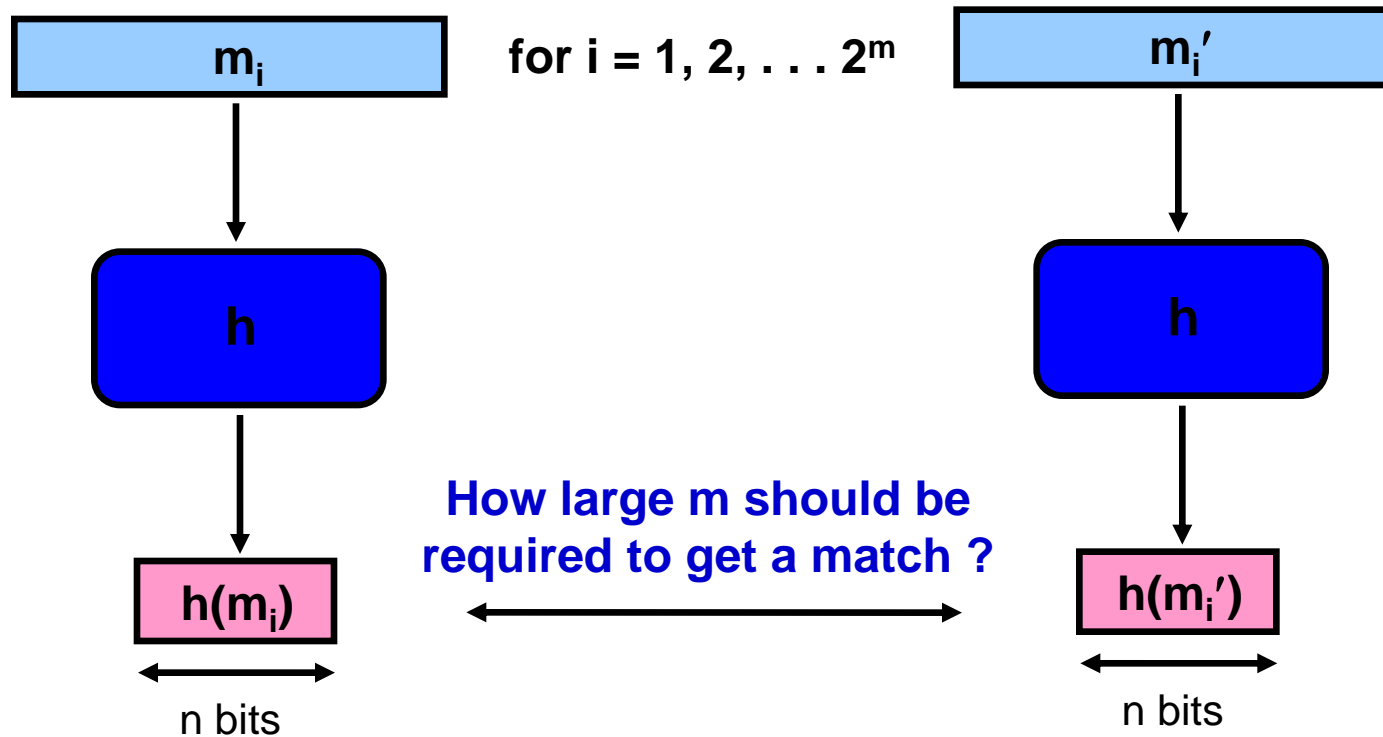
11 different positions of similar expressions



2¹¹ different messages of the same meaning

Collision in Collision-Resistant Hash Function

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**)

$$365/2 \approx 188$$

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

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

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

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

One Million \$ Hardware Brute Force Attack

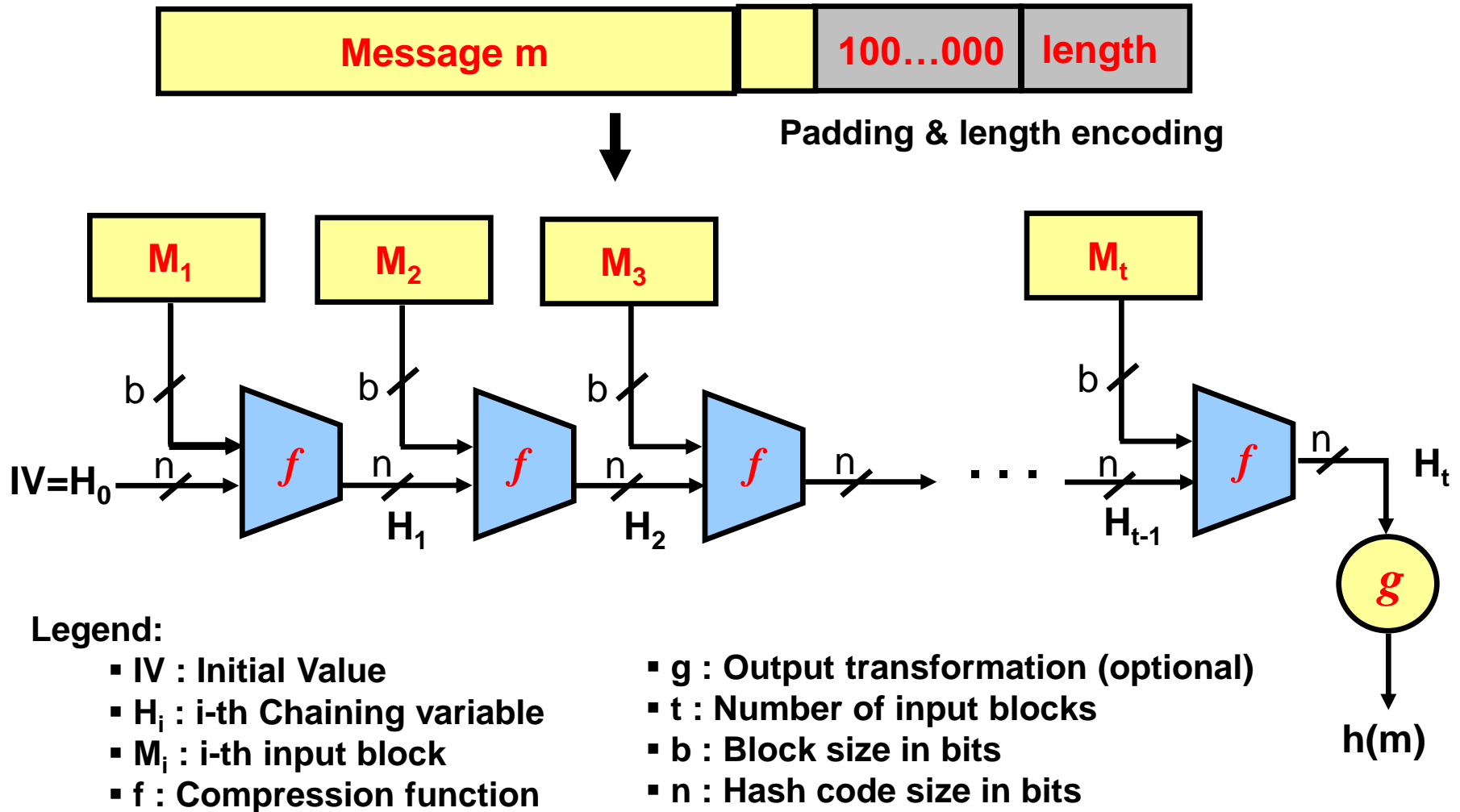
❖ One-Way Hash Functions (complexity = 2^n)

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

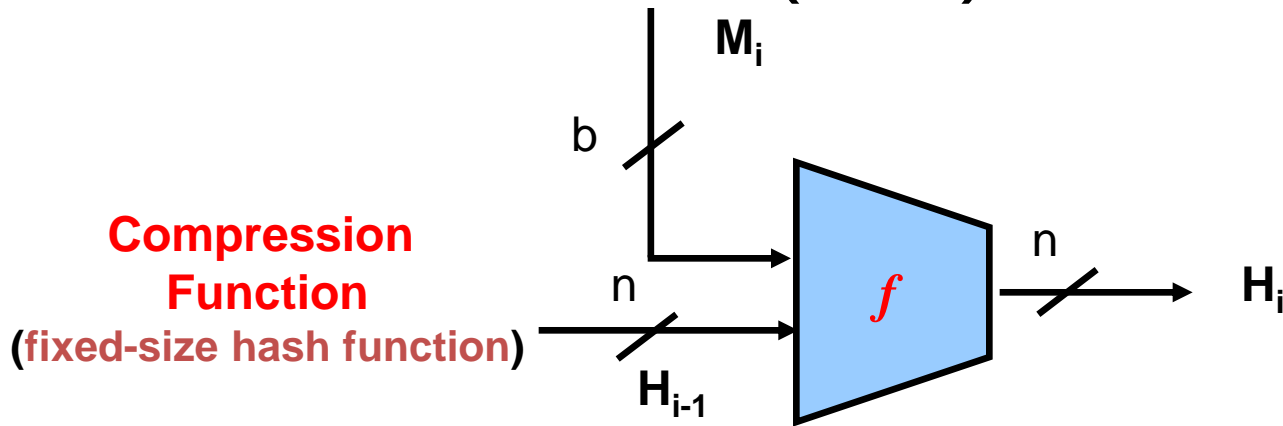
❖ Collision-Resistant Hash Functions (complexity = $2^{n/2}$)

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

Construction of Hash Function (1/2)



Construction of Hash Function (2/2)



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

Entire hash

$$H_0 = IV$$

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

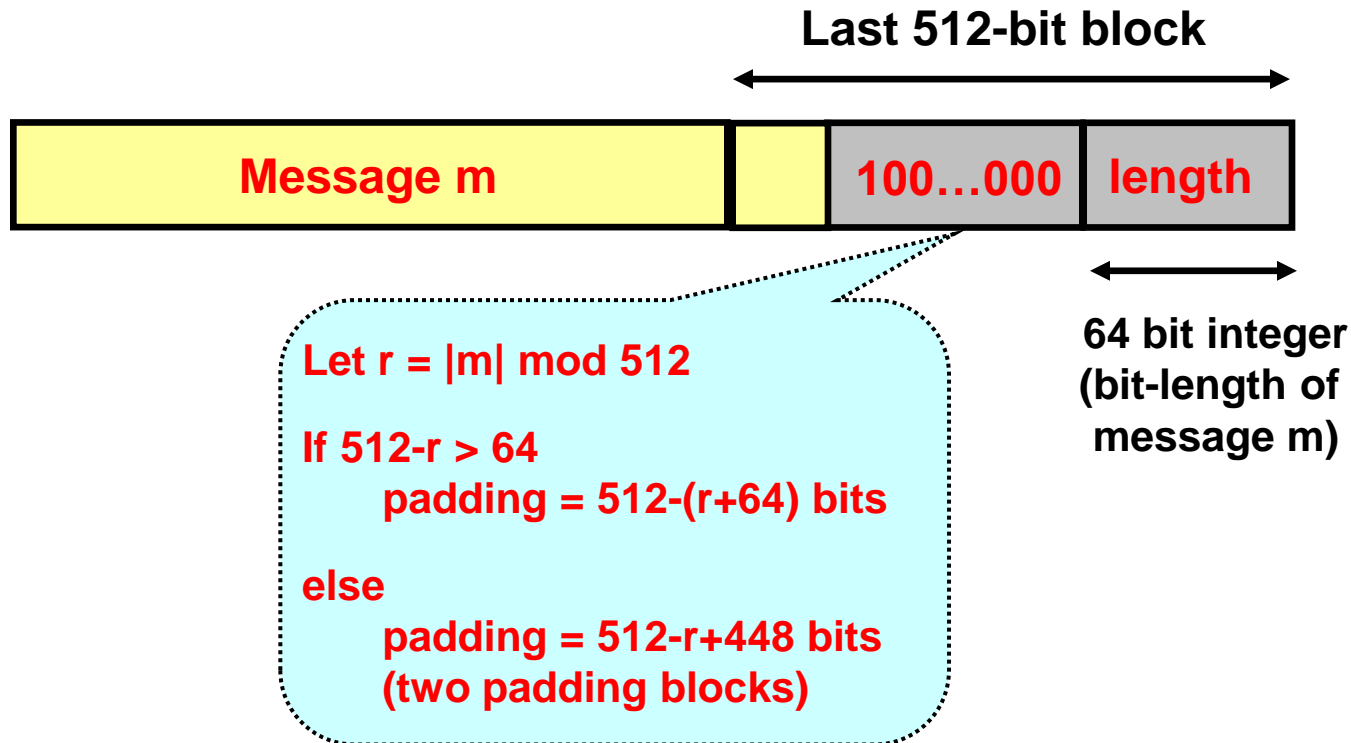
$$H(m) = g(H_t)$$

Fact(by Merkle-Damgård) **MD-strengthening.**

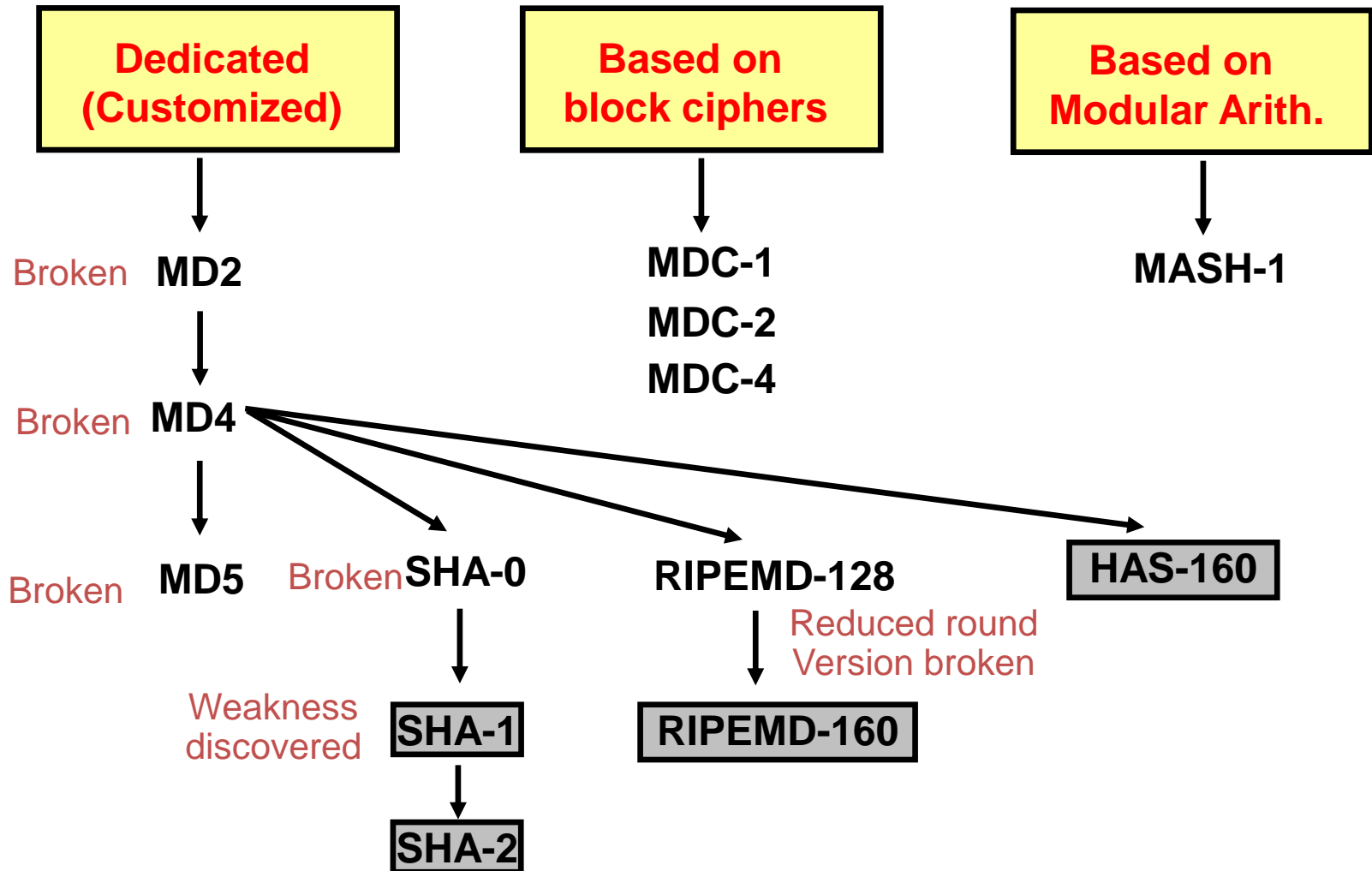
Any collision-resistant compression function f can be extended to a collision-resistant hash function h

Typical Padding

- ❖ Assume Block size = 512 bits (MD5, SHA-1, RMD160, HAS160 ...)



Hash Function Family



SHA (Secure Hash Algorithm) (1/2)

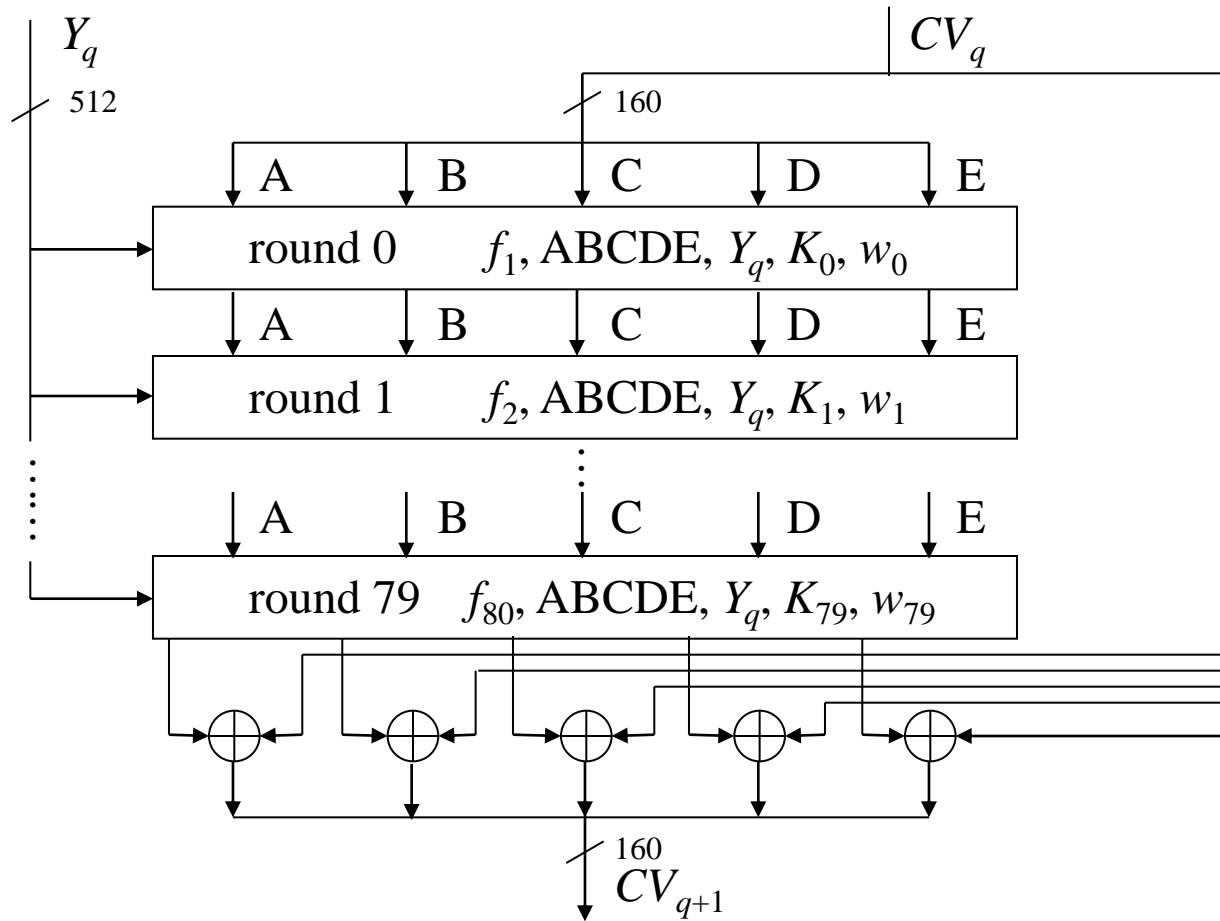
- ❖ SHA was designed by NIST (National Institute of Standards and Technology) & NSA (National Security Agency)
- ❖ US standard for use with DSA signature scheme
- ❖ The algorithm is SHA and the standard is SHS.
- ❖ Based on the design of MD4 and MD5 by Rivest@MIT

- ❖ SHA-0: FIPS PUB 180, 1993
- ❖ **SHA-1: FIPS Pub 180-1, 1995**
 - ✓ bitwise rotation of message schedule of SHA-0 changed
 - ✓ widely-used security applications and protocols such as TLS and SSL, PGP, SSH, S/MIME, and IPsec
- ❖ SHA-2: FIPS Pub 180-2, 2001
 - ✓ SHA-224, SHA-256, SHA-384, and SHA-512
 - ✓ Not so popular as SHA-1

SHA (Secure Hash Algorithm) (2/2)

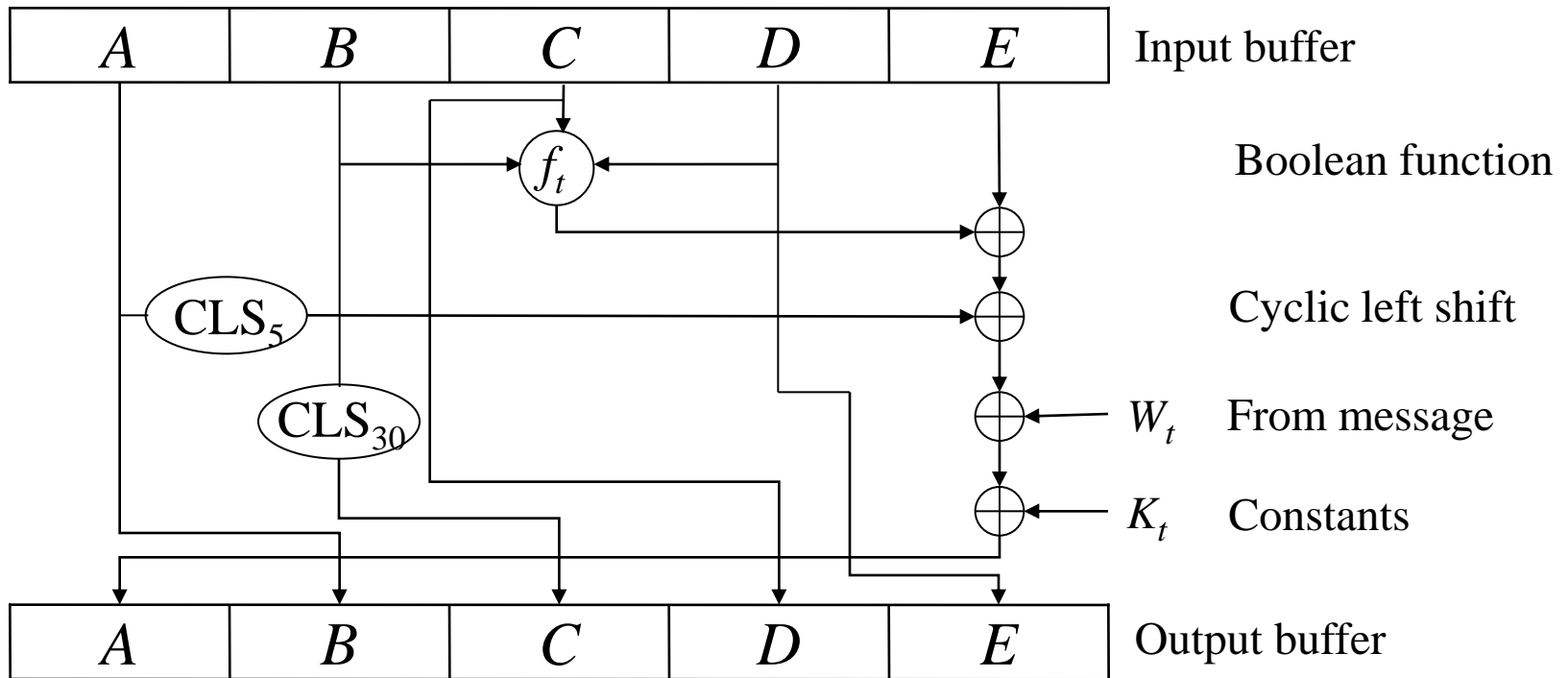
Algorithm and variant	Output size (bits)	Internal state size (bits)	Block size (bits)	Max message size (bits)	Word size (bits)	Rounds	Operation	Collisions Found
SHA-0	160	160	512	$2^{64} - 1$	32	80	+,and,or, xor,rot	Yes
SHA-1	160	160	512	$2^{64} - 1$	32	80	+,and,or, xor,rot	Yes 2^{52} attack (*)
SHA-2	<i>SHA-2</i> 256/224	256	512	$2^{64} - 1$	32	64	+,and,or, xor,shr,rot	None
	<i>SHA-5</i> 512/384	512	1024	$2^{128} - 1$	64	80	+,and,or, xor,shr,rot	None

SHA-1 Overview



<http://www.itl.nist.gov/fipspubs/fip180-1.htm>

SHA-1 Round Function



SHA-1 Constants & Boolean Functions

Initial values

$A = 67452301$

$B = EFCDA B89$

$C = 98BADCFE$

$D = 10325476$

$E = C3D2E1F0$

Constants K_t

$t = 0 \sim 19$ $K_t = 5A827999$

$t = 20 \sim 39$ $K_t = 6ED9EBA1$

$t = 40 \sim 59$ $K_t = 8F1BBCDC$

$t = 60 \sim 79$ $K_t = CA62C1D6$

Boolean function f_t

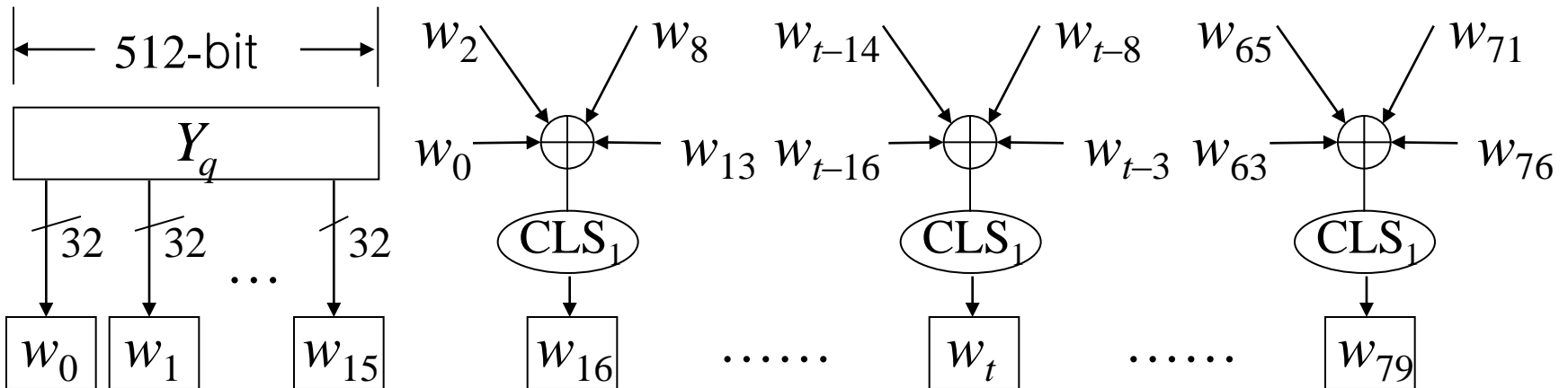
$t = 0 \sim 19$ $f_t(B, C, D) = B \cdot C + \overline{B} \cdot D$

$t = 20 \sim 39$ $f_t(B, C, D) = B \oplus C \oplus D$

$t = 40 \sim 59$ $f_t(B, C, D) = B \cdot C + B \cdot D + C \cdot D$

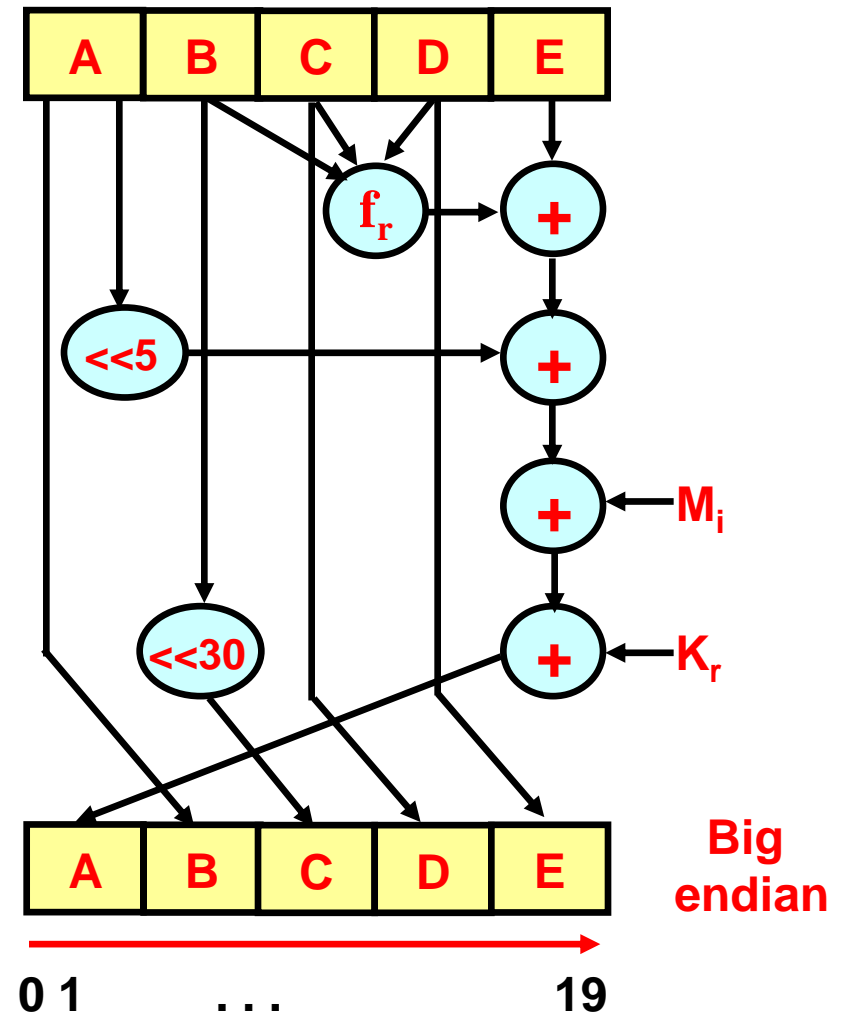
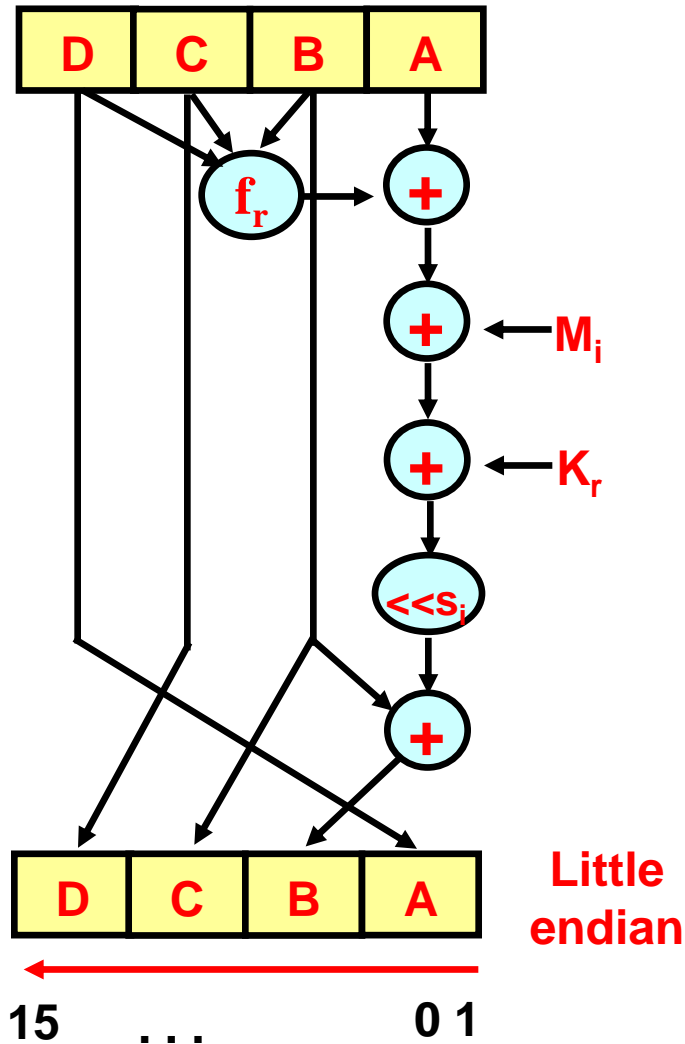
$t = 60 \sim 79$ $f_t(B, C, D) = B \oplus C \oplus D$

SHA-1 Message Inputs

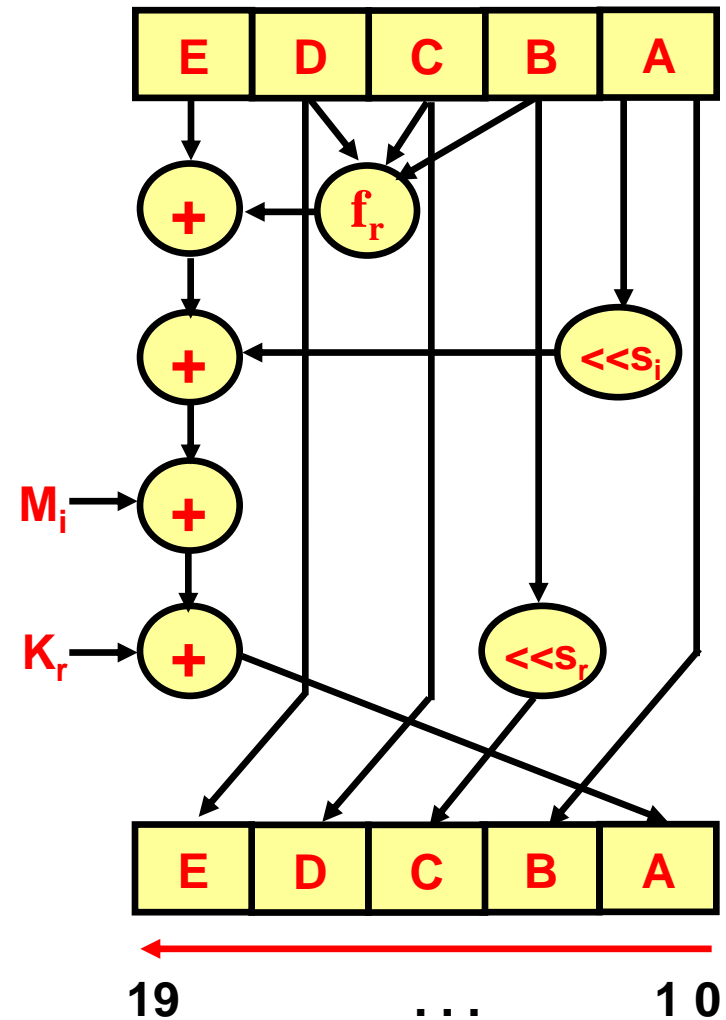
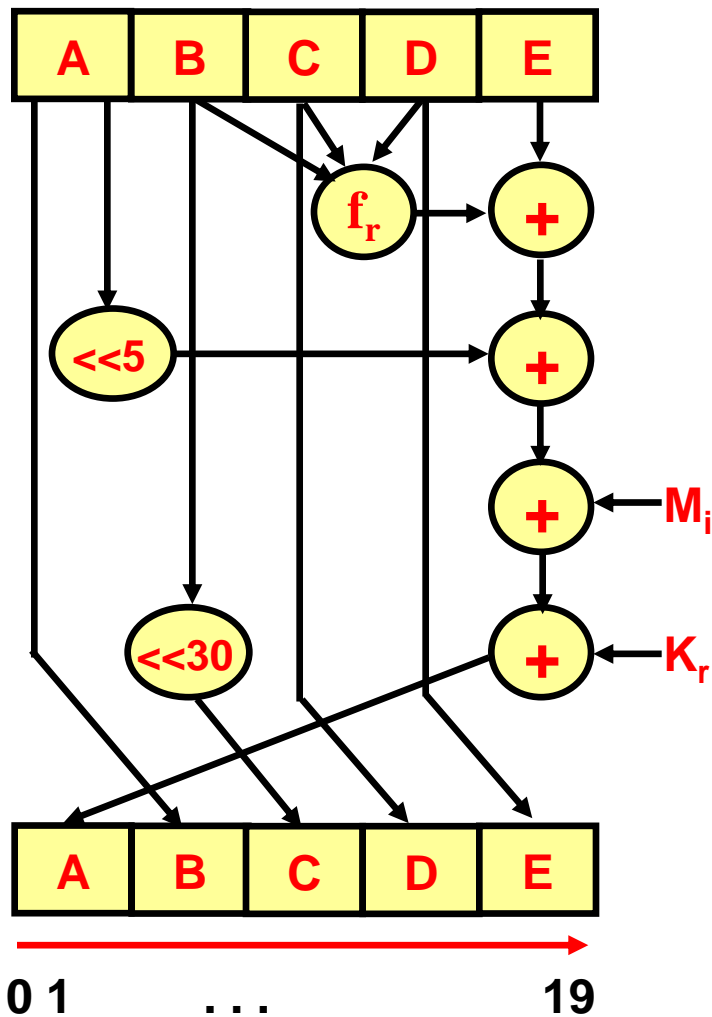


CLS: Cyclic Left Shift

Step Operations of MD5 & SHA-1



Step Operations of SHA1 & HAS160

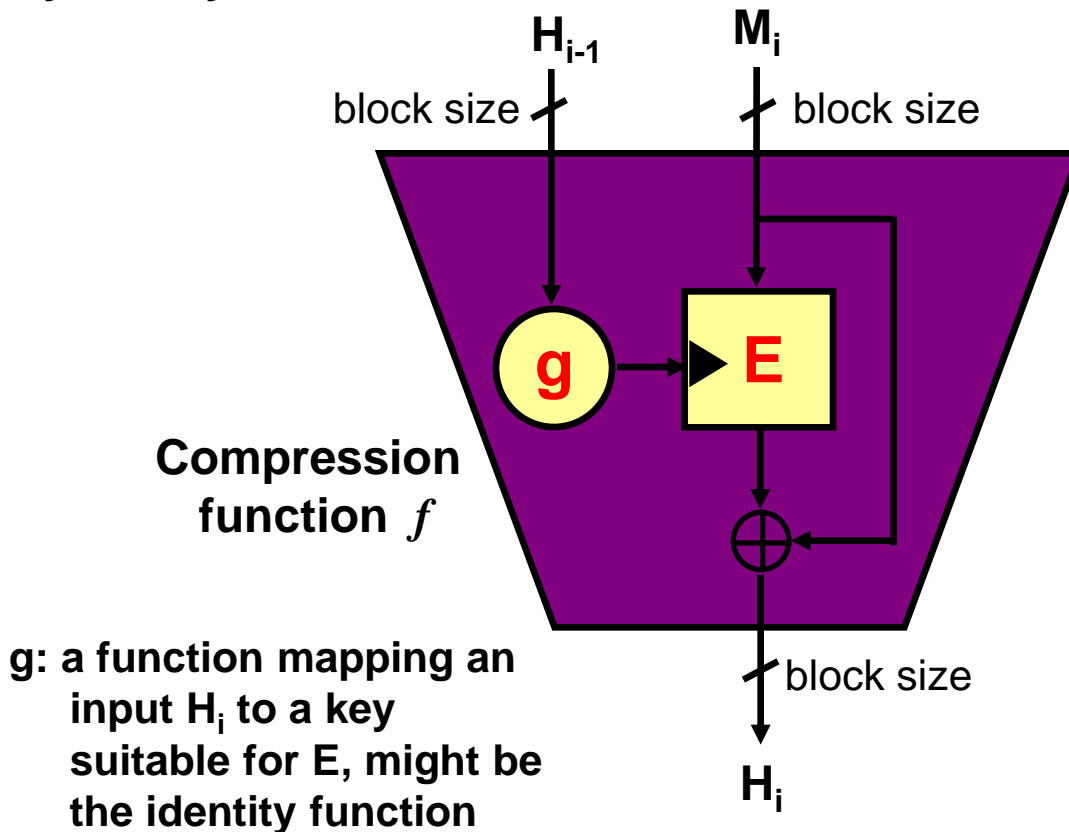


Comparison

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 Ft based on Block Ciphers : MDC1

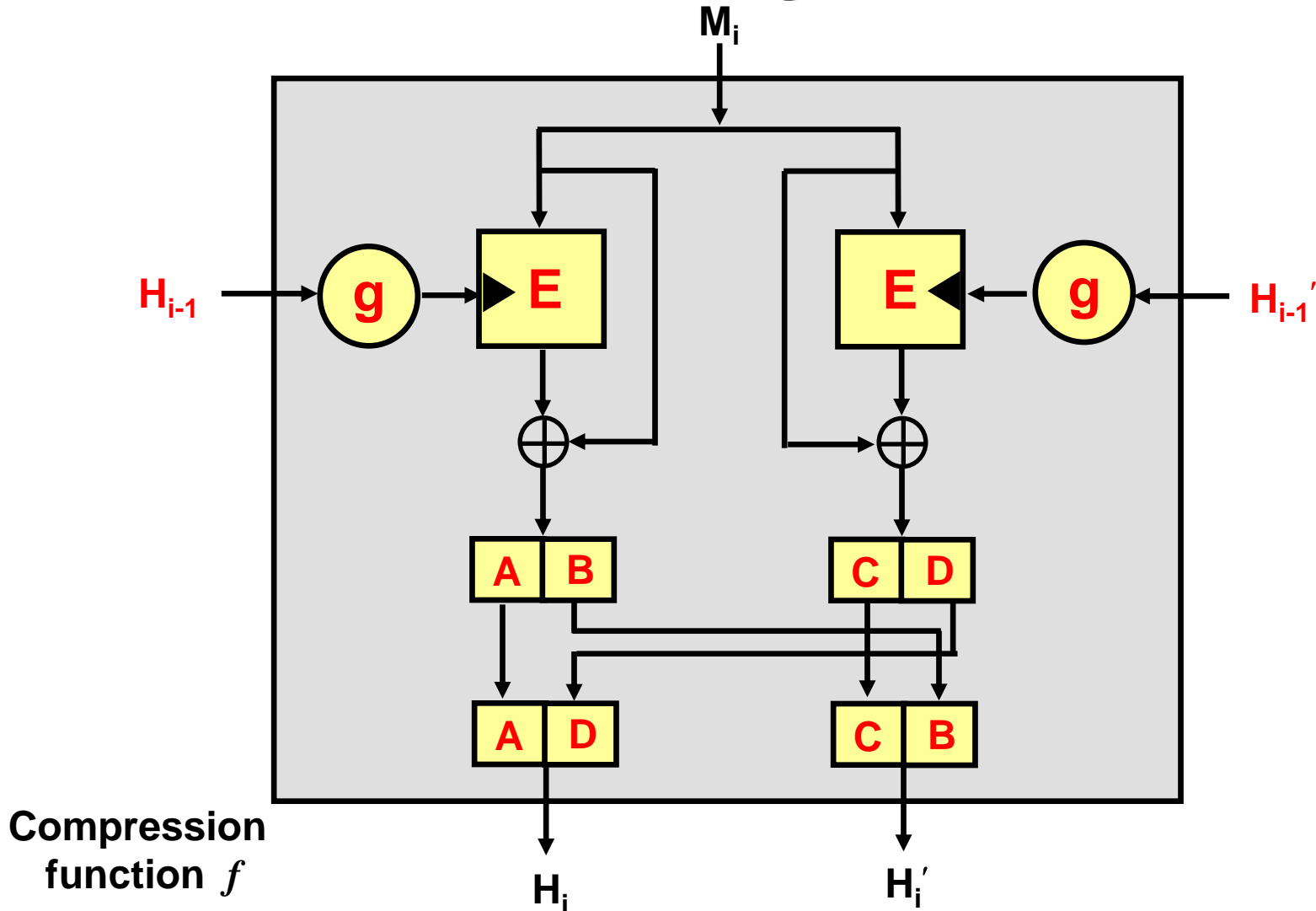
Matyas-Meyer-Oseas Scheme



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

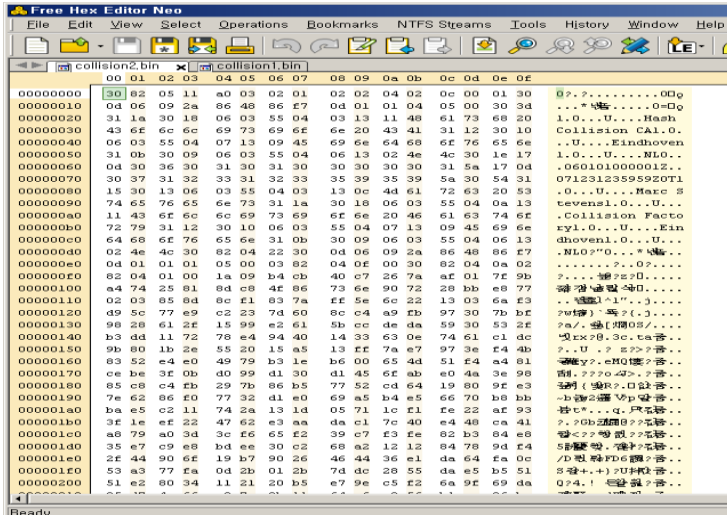
Hash Ft based on Block Ciphers :

MDC2

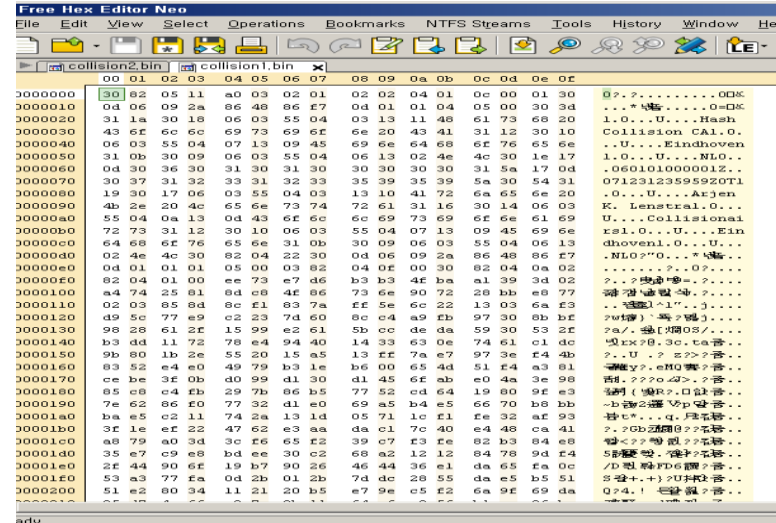


Ex. of MD5 Collisions

Collision1.bin



Collision2.bin



http://www.win.tue.nl/hashclash/ChosenPrefixCollisions/md5sumsrecipe.txt - Windows Internet Explorer

http://www.win.tue.nl/hashclash/ChosenPrefixCollisions/md5sumsrecipe.txt

Use of md5sums.exe: the command

```
md5sums -b -s collision1.bin collision2.bin
```

should produce the following output:

```
MD5sums 1.2 freeware for Win9x/ME/NT/2000/XP+
Copyright (C) 2001-2005 Jem Berkes - http://www.pc-tools.net/

collision1.bin          c6b2fe88912770fc612db71f58c7d251
collision2.bin          c6b2fe88912770fc612db71f58c7d251

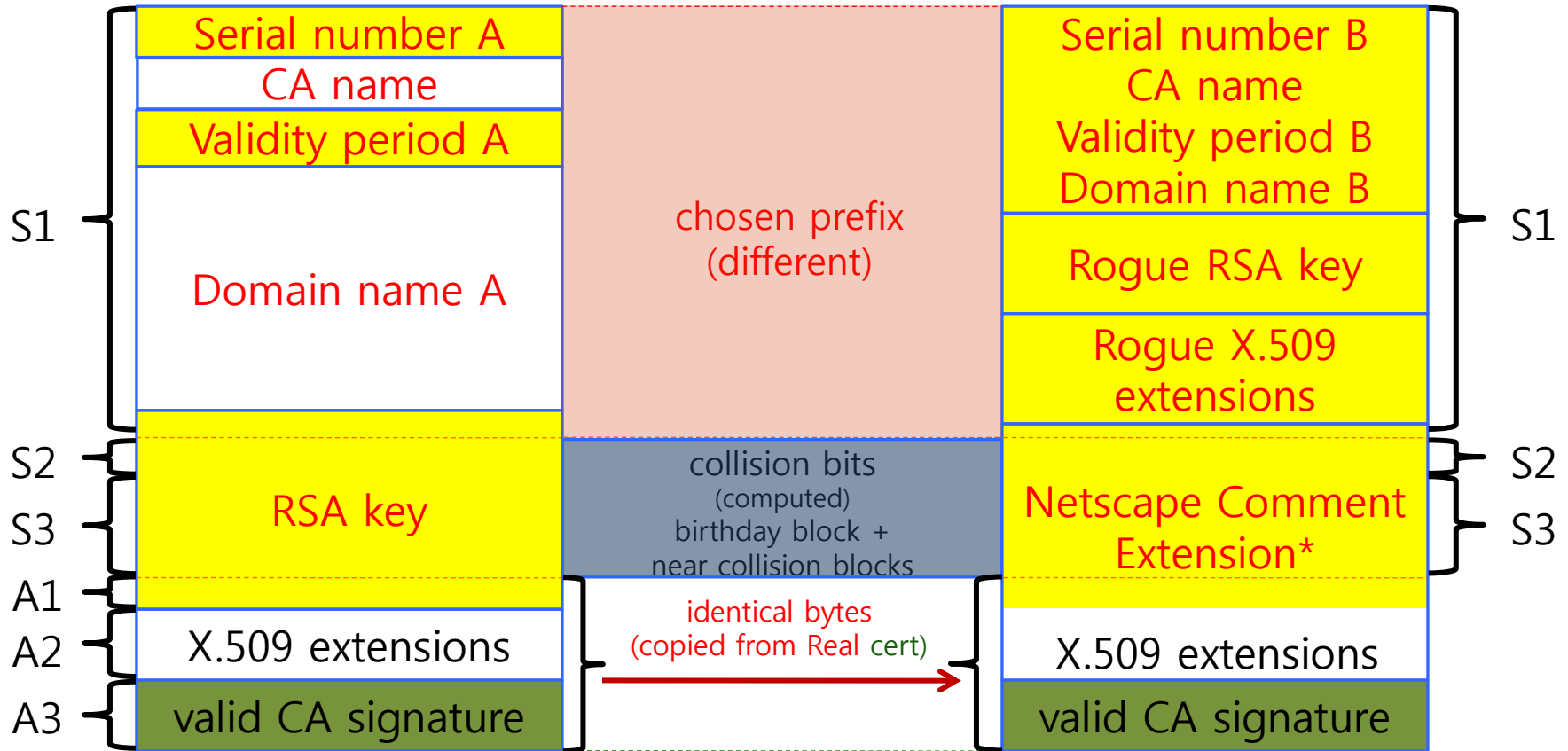
2602 bytes, 0 ms = 0.00 MB/sec
```

Same MD5 Hashed Value !!

MD5 Collision Attacks

- Colliding valid X.509 certificates
 - Lenstra, Wang, Weger, forged X.509 certificates, <http://eprint.iacr.org/2005/067.pdf>
 - Same owner with different public keys (2048 bits)
 - Stevens, Lenstra, Weger, Eurocrypt 2007
 - 8192-bit public key (8-block collision)
 - Stevens *etc.* Crypto 2009
 - Pass the browser authentication, different owners, different public keys (See next)

X.509v3 Real and Fake Certificates using MD-5



* contents ignored by browsers

SHA-3 Project

The screenshot shows the NIST website's Computer Security Resource Center. The header includes the NIST logo, the text 'National Institute of Standards and Technology Information Technology Laboratory', a search bar for CSRC, and navigation links for ABOUT, MISSION, CONTACT, STAFF, and SITE MAP. The main navigation bar lists CSRC HOME, GROUPS, PUBLICATIONS, DRIVERS, NEWS & EVENTS, and ARCHIVE. A left sidebar menu is expanded to show the 'Cryptographic Hash Project' section, which includes links to the competition, timeline, federal register notices, NIST policy on HASH functions, NIST comments on SHA-1 cryptanalysis, two workshops, a hash forum, contacts, and other links. The main content area is titled 'CRYPTOGRAPHIC HASH PROJECT' and contains 'Background Information' with three paragraphs detailing the project's history, goals, and recent developments. The footer includes the NIST logo, CSRC webmaster information, a disclaimer, and dates for the last update and page creation.

NIST National Institute of Standards and Technology
Information Technology Laboratory

SEARCH CSRC: GO

ABOUT MISSION CONTACT STAFF SITE MAP

Computer Security Division Computer Security Resource Center

CSRC HOME GROUPS PUBLICATIONS DRIVERS NEWS & EVENTS ARCHIVE

Cryptographic Hash Project

- Cryptographic Hash Algorithm Competition
- Timeline for Hash Algorithm Competition
- Federal Register Notices
- NIST Policy on HASH Functions
- NIST Comments on SHA-1 Cryptanalysis
- 2005 Cryptographic Hash Workshop
- 2006 Cryptographic Hash Workshop
- Hash Forum
- Contacts
- Other Links

CSRC HOME > GROUPS > ST > HASH PROJECT

CRYPTOGRAPHIC HASH PROJECT

Background Information

A hash function takes binary data, called the message, and produces a condensed representation, called the message digest. A cryptographic hash function is a hash function that is designed to achieve certain security properties. The [Federal Information Processing Standard 180-2, Secure Hash Standard](#), specifies algorithms for computing five cryptographic hash functions — SHA-1, SHA-224, SHA-256, SHA-384, and SHA-512. FIPS 180-2 was issued in August, 2002, superseding FIPS 180-1.

In recent years, several of the non-NIST approved cryptographic hash functions have been successfully attacked, and serious attacks have been published against SHA-1. In response, NIST held two public workshops (see menu at left) to assess the status of its approved hash functions and to solicit public input on its cryptographic hash function policy and standard. As a result of these workshops, NIST has decided to develop one or more additional hash functions through a public competition, similar to the development process of the [Advanced Encryption Standard \(AES\)](#). NIST has proposed a [tentative timeline](#) for the competition, and also published a [policy on the use of the current hash functions](#).

NIST issued [draft minimum acceptability requirements, submission requirements, and evaluation criteria](#) for candidate hash algorithms [in January, 2007 \[Federal Register Notice \(January 23, 2007\)\]](#) for public comments; the comment period ended on April 27, 2007. Based on the [public feedback](#), NIST has revised the requirements and evaluation criteria and issued a [Call for a New Cryptographic Hash Algorithm \(SHA-3\) Family](#) on November 2, 2007 [\[Federal Register Notice \(November 2, 2007\)\]](#) to launch the hash algorithm competition. Details of the competition are available at [www.nist.gov/hash-competition](#).

NIST CSRC Webmaster, [Disclaimer Notice & Privacy Policy](#)
NIST is an Agency of the U.S. Department of Commerce

Last updated: December 10, 2008
Page created: April 15, 2005