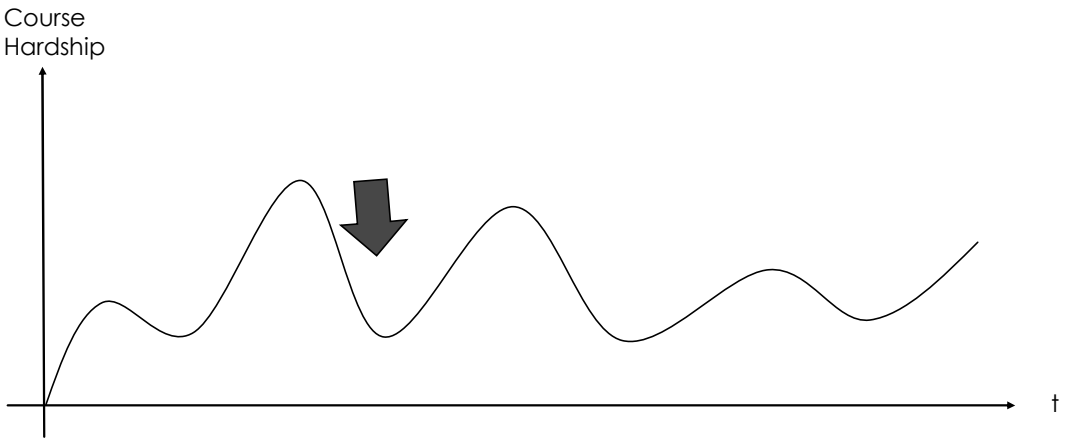
Network Security – Cryptography – Part 3
امنیت شبکه – رمزنگاری – بخش ۳

Mohammad Sayad
University of Tehran

1

How hard is this course?



2

کد احراز اصالت پیام
Message Authentication Code (MAC)

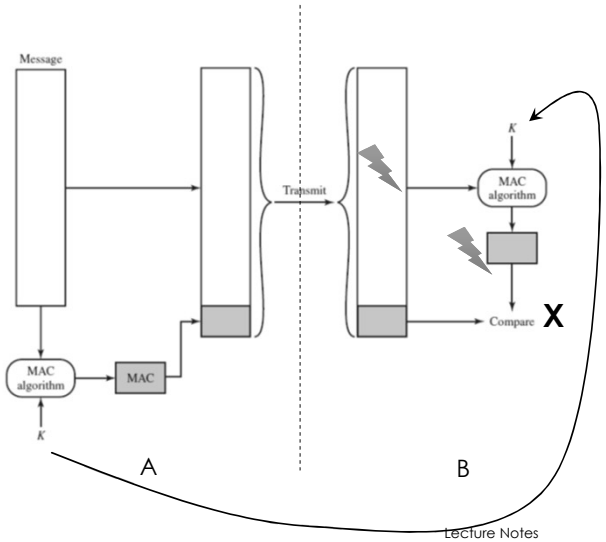
Message Authentication Code (MAC)

- MAC provides Authentication + Integrity
- But not Confidentiality
- It is sometimes called Message Integrity Code (MIC) to differentiate it from Medium Access Control (MAC).

Message Authentication Code (MAC)

This technique assumes that two communicating parties, say A and B, share a common secret key K_{AB}.

When A has a message to send to B, it calculates the message authentication code as a function of the message and the key: $MAC_M = F(K_{AB}, M)$



Points

نکات

از آنجا که کلید K_{AB} تنها دست فرستنده و گیرنده است، گیرنده B با دریافت پیام و بازگشایی چکیده با کلید K_{AB} مطمئن میشود که پیام از طرف A ارسال شده است. با رمزنگاری نامتقارن نیز چنین تضمینی ایجاد میشود. (Authentication)

اگر پیام در میانه راه توسط کسی دستکاری شود، چون در طرف B ، MAC تولید شده پیام با ضمیمه انتهای آن همخوانی ندارد، گیرنده متوجه مخدوش بودن پیام خواهد شد. بنابراین امکان دست بردن در پیام وجود ندارد (Integrity)

What's inside a MAC? داخل MAC چه چیزی است

- الگوریتم MAC شامل دو قسمت است

چون رمز کردن تمام پیام هزینه بر است

۱- محاسبه چکیده ای از پیام

۲- ارسال چکیده با کلید یا رمزی که گیرنده بتواند آنرا بازگشایی کند.

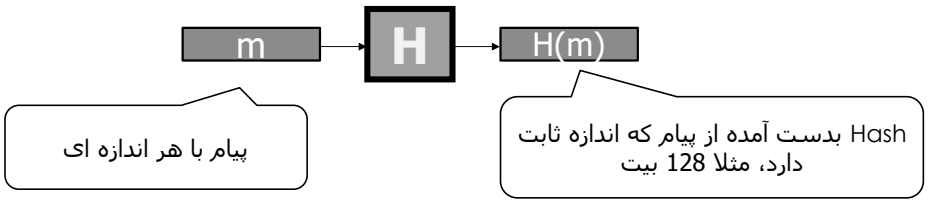
آیا لازم است طوری MAC ساخته شود که فقط گیرنده بتواند آنرا باز کند؟

چه اتفاقی می افتد اگر همه بتوانند MAC را رمزگشایی کنند؟

– پاسخ به این سوال وابسته به این است که از رمز متقارن استفاده کنیم یا نامتقارن

توابع چکیده ساز (در هم ریز) Hash Functions

توابع چکیده ساز (در هم ریز) Hash Functions



مثال الگوریتم های Hash: SHA-1، MD4، ...

توابع Hash تابع کلید نیستند و هر متنی را به طول ثابت فشرده میکنند. بنابراین یکطرفه هستند و پیام اصلی قابل بازیابی نیست. محاسبه معکوس این توابع از نظر محاسباتی غیر ممکن است.

مثال:

MD4("The quick brown fox jumps over the lazy dog")
= 1bee69a46ba811185c194762abaeae90
MD4("The quick brown fox jumps over the lazy cog")
= b86e130ce7028da59e672d56ad0113df

MD4 ("") = 31d6cfe0d16ae931b73c59d7e0c089c0
MD4 ("a") = bde52cb31de33e46245e05fbdbd6fb24
MD4 ("abc") = a448017aaf21d8525fc10ae87aa6729d
MD4 ("message digest") = d9130a8164549fe818874806e1c7014b
MD4 ("abcdefghijklmnopqrstuvwxyz")
= d79e1c308aa5bbcdeea8ed63df412da9
MD4
("ABCDEFGHIJKLMNOPQRSTUVWXYZabcdefghijklmnopqrstuvwxyz0123456789")
= 043f8582f241db351ce627e153e7f0e4

خواص تابع چکیده ساز (Hash Function Properties)

- **1.** H can be applied to a block of data of any size.
- **2.** H produces a fixed-length output.
- **3.** H(x) is relatively easy to compute for any given x
- **4.** For any given code h, it is computationally infeasible to find x such that H(x)=h.

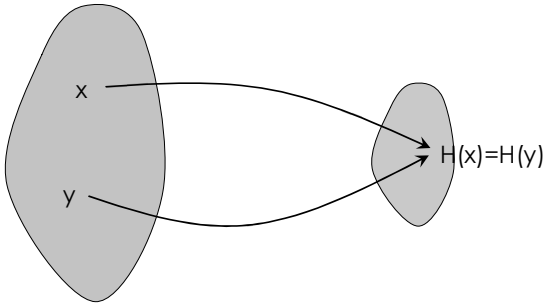
(A hash function with this property is referred to as **one-way** or **preimage resistant**)

یکطرفه بودن تابع چکیده ساز

خواص تابع چکیده ساز (Hash Function Properties)

- **5.** For any given block x, it is computationally infeasible to find $y \neq x$ with H(y)=H(x).

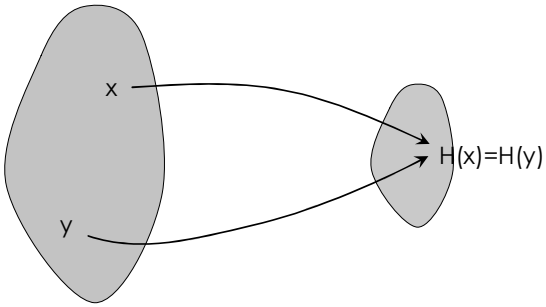
(A hash function with this property is called **second preimage resistant**. This is sometimes referred to as **weak collision resistant**)



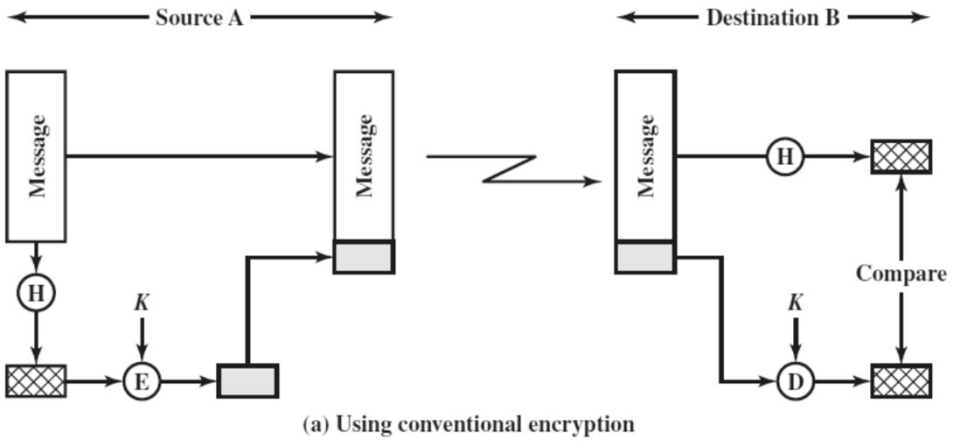
خواص تابع چکیده ساز (Hash Function Properties)

➡ **6.** It is computationally infeasible to find any pair (x, y) such that $H(x)=H(y)$.

(A hash function with this property is referred to as **collision resistant**. This is sometimes referred to as **strong collision resistant**)

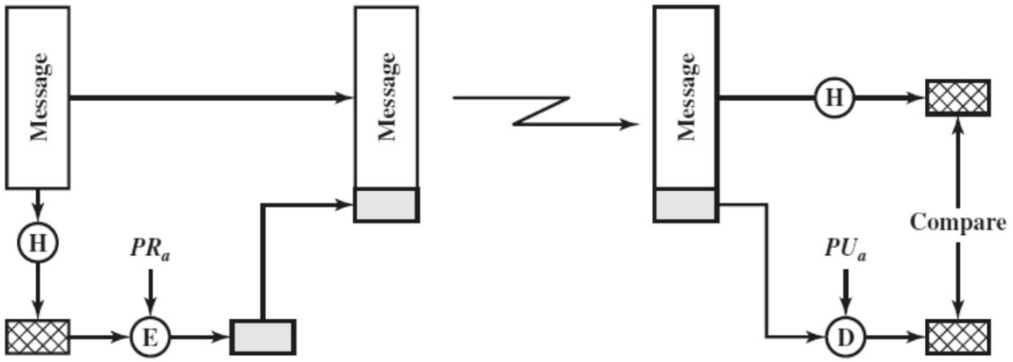


روشهای ساخت MAC با استفاده از Hash Function



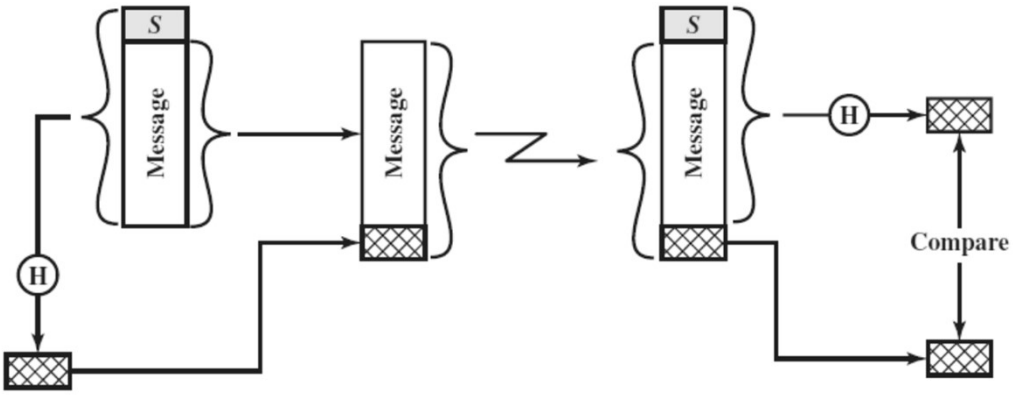
(a) Using conventional encryption

روشهای ساخت MAC با استفاده از Hash Function



(b) Using public-key encryption

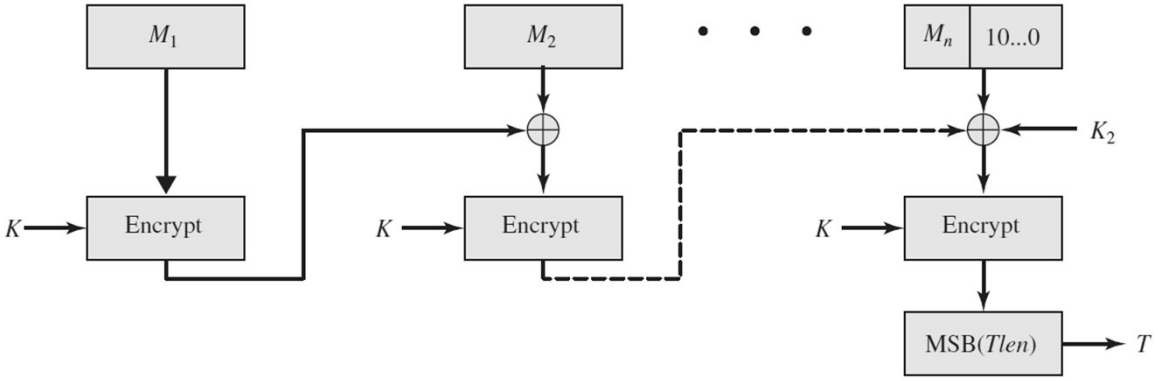
روشهای ساخت MAC با استفاده از Hash Function



(c) Using secret value

Secret value can be the key

How to make a MAC using a block cipher



$M_1 \ldots M_{n-1}$ are plain text pieces. K and K_2 are encryption and MAC keys, respectively. MSB(Tlen) takes a desired length from output (from the most significant bit).

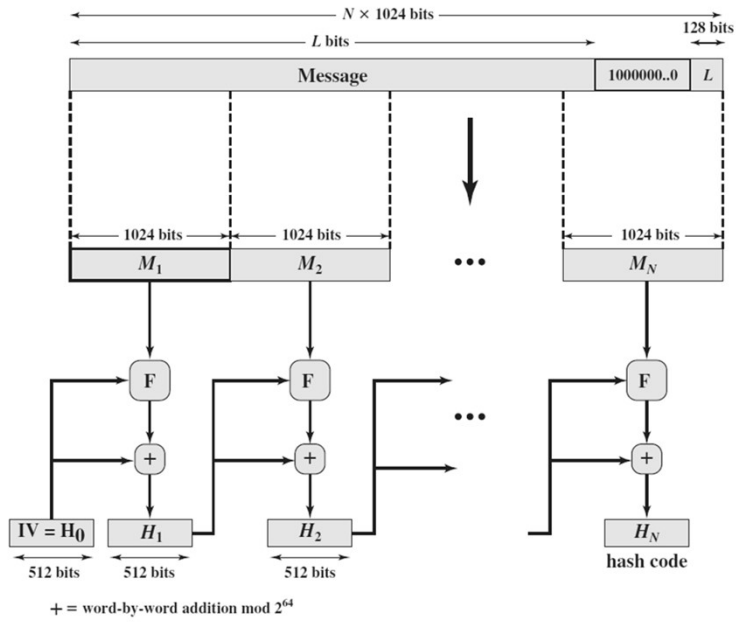
توابع **Hash** معروف کدامند

- MD4 ،MD5

  MD4 was proposed by Ronald Rivest in 1990. The digest length is 128 bits. It was broken later but it influenced the design of MD5 (RFC1321), SHA-1 and RIPEMD hashing algorithms.

- SHA-0 ،… ،SHA-3

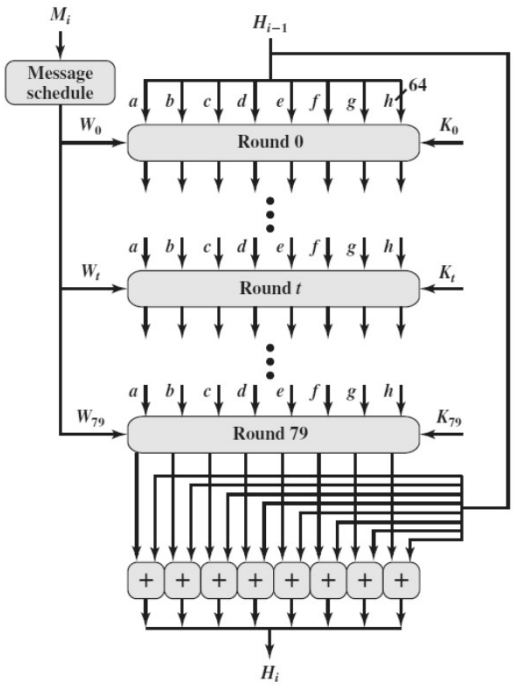
  SHA was developed by the National Institute of Standards and Technology (NIST) and published as a federal information processing standard (FIPS 180) in 1993. SHA used MDx designs.



Message Digest Generation Using SHA-512

ساختمان SHA-512

- Input is padded so that its length is 896 (mod 1024)
- 128 bits are added to the end showing the message length
- IV is a 512 bit fixed initialization vector
- F function is defined in the next slide



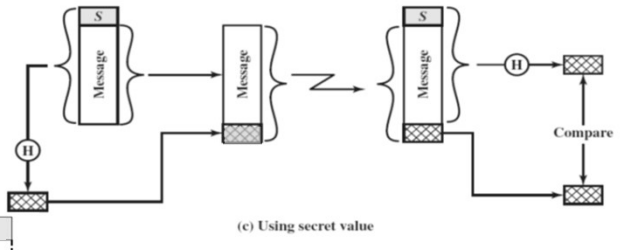
F Function in SHA-512

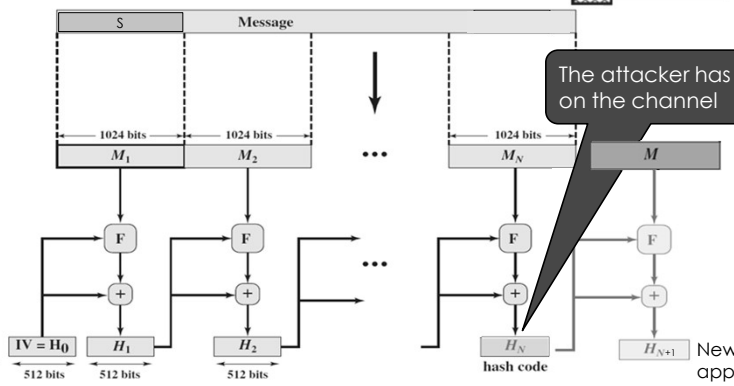
- H_{i-1} is divided into 8 64-bit registers a,b,c,d,e,f,g,h
- K_x s' are constants set by the designer

لازم به حفظ کردن نیست!

Length Extension Attack (on SHA)

- You remember this? →
- Now imagine we use SHA-2 512 in this setup.



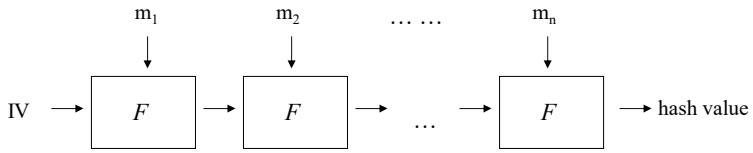


The attacker has this on the channel

New VALID hash code of the appended message

General Structure - Merkle-Damgard

Message m padded to M, a multiple of a fixed-length block
M is divided into segments $m_1, m_2, \ldots m_n$



Merkle-Damgard, 1989
F is called the compression function
- Takes inputs m_i and output of previous iteration
- Typically a series of rounds
- Output called a "chaining variable"
- Typically, a function operates on chaining variables then adds to m_i

SHA-3 – Competition

- 2004-2005 Wave of new cryptanalysis

موجی از حملات جدید در سالهای ۲۰۰۵ و ۲۰۰۴

  - Wang, Biham, Joux, Kelsey all published significant papers….
  - Cast doubt on existing hash standards and the traditional Merkle-Damgård construction
- 2005, 2006 NIST Hash Function Workshops

تقاضای صنعت و دانشگاه از NIST برای برگزاری مسابقه جدید طراحی

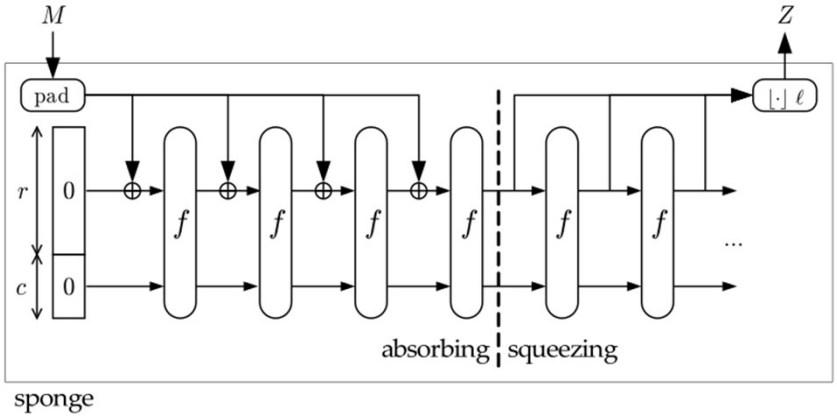
  - Industry and academia encouraged NIST to run a competition and contribute to planning
- 2007 NIST organized SHA-3 competition
  - 64 candidates submitted 31 Oct. 2008

SHA-3 Competition

- Five Finalists identified late in 2010.
  - Blake, Grøstl, JH, Keccak, Skein
- Final tweaks submitted January 2011.
- Final Workshop held in March 2012 in Washington DC
  - The winner was Keccak algorithm

SHA3

SHA-3 – Sponge Construction



- Each round, the next r bits of message is XOR'ed into the first r bits of the state, and a function f is applied to the state.
- After message is consumed, output r bits of each round as the hash output; continue applying f to get new states
- SHA-3 uses 1600 bits for state size.

Speed Comparisons

Algorithm	Speed (MiByte/s.)
AES-128 / CTR	198
MD5	335
SHA-1	192
SHA-256	139
SHA-3	~ SHA-256

Crypto++ 5.6 benchmarks, 2.2 GHz AMD Opteron 8354

➨ NIST expects SHA-2 to be used for the foreseeable future.

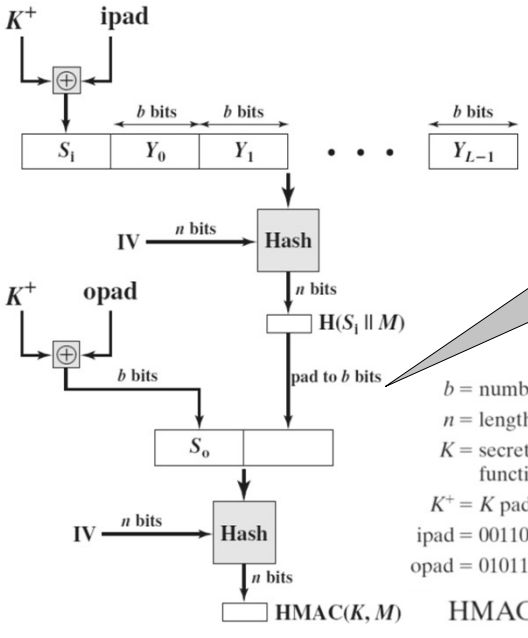
استاندارد HMAC

- هدف : ساخت یک MAC با استفاده از SHA ← RFC 2104
- محل استفاده: IP Security ، TLS، SET، ...

MAC should be designed in a way that it both has fixed length and is a function of the key.

MAC باید به نحوی تابعی ساخته شود که هم عمل چکیده سازی را انجام دهد و هم تابع کلید باشد.

ساختمان HMAC



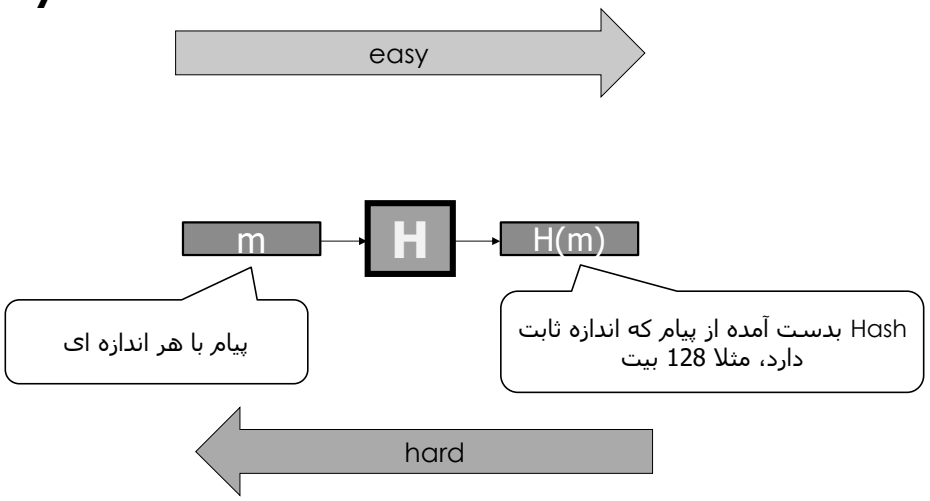
The second hash eliminates the possibility of length extension, even if a weak hash function is used.

b = number of bits in a block
n = length of hash code produced by embedded hash function
K = secret key; if key length is greater than b, the key is input to the hash function to produce an n-bit key; recommended length is $> n$
K^+ = K padded with zeros on the left so that the result is b bits in length
ipad = 00110110 (36 in hexadecimal) repeated $b/8$ times
opad = 01011100 (5C in hexadecimal) repeated $b/8$ times

$$\text{HMAC}(K, M) = \text{H}[(K^+ \oplus \text{opad}) \| \text{H}[(K^+ \oplus \text{ipad}) \| M]]$$

Summary

خلاصه



Security of Hash Functions

- Two Attack Approaches:
  - Cryptanalysis → Looking for a logical weakness
  - Brute Force → Exhaustive searching
- The strength of a hash function against brute-force attacks depends solely on **n.** The level of effort in each case is:

Preimage resistant	2^n
Second preimage resistant	2^n
Collision resistant	$2^{n/2}$

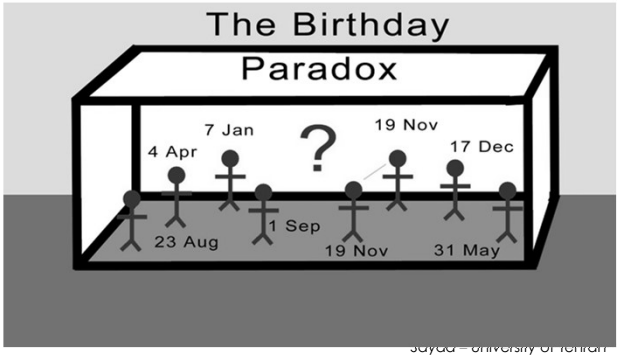
Birthday Paradox

Birthday Paradox

- If there are **n** people in a room, how much is the probability that at least 2 of them have the same birthday? (collision)

$$\bar{p}(n) = 1 \times \left(1 - \frac{1}{365}\right) \times \left(1 - \frac{2}{365}\right) \times \cdots \times \left(1 - \frac{n-1}{365}\right)$$
$$= \frac{365 \times 364 \times \cdots \times (365 - n + 1)}{365^n}$$
$$= \frac{365!}{365^n (365 - n)!}$$
$$p(n) = 1 - \bar{p}(n).$$

- For 23 people: p(23)=50.7% !



31

Hash/MAC پایان

32