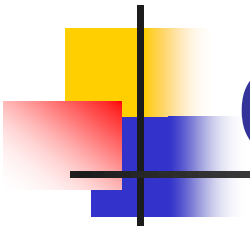
Bloque II
Sistemas de autenticación

Algoritmos generadores de autenticadores

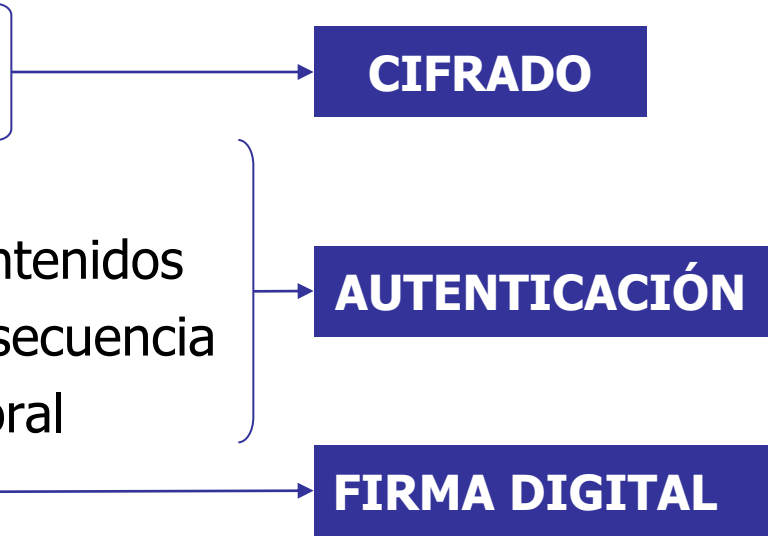
Seguridad en Redes de Comunicaciones
María Dolores Cano Baños



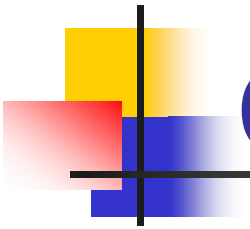
Contenidos

Introducción

- Ataques:
  - Revelación
  - Análisis del tráfico
  - Enmascaramiento
  - Modificación de contenidos
  - Modificación de la secuencia
  - Modificación temporal
  - Repudio



- Autenticación o firma digital
  - 1) Autentificador
  - 2) Protocolo de autenticación

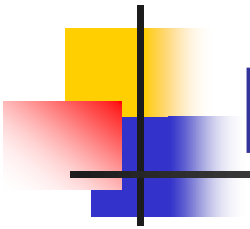


Contenidos

1.1 Introducción ✓

1.2 Funciones HASH y MAC

- 1.2.1 MD5
- 1.2.2 SHA
- 1.2.3 HMAC



Funciones HASH y MAC

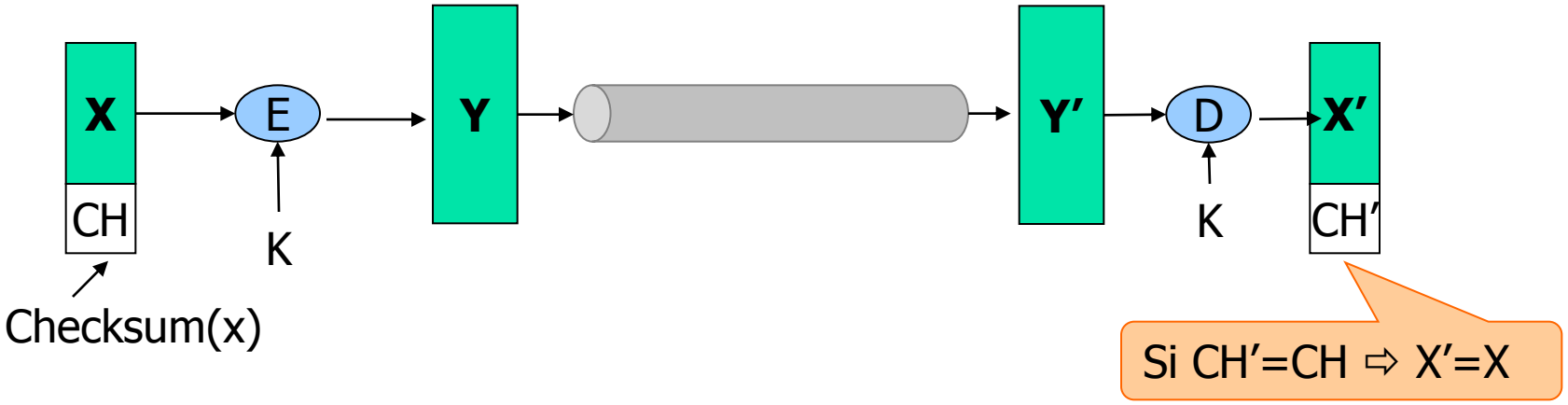
- Funciones para generar un **autentificador**:
  - Cifrado del mensaje
  - Código de autenticación del mensaje
  - Función hash

Funciones HASH y MAC

- **Cifrado del mensaje:**

CIFRADO SIMÉTRICO

- Confidencialidad ✓
- Autenticación ✓
- ¿Cómo garantizar que el texto plano es el original?

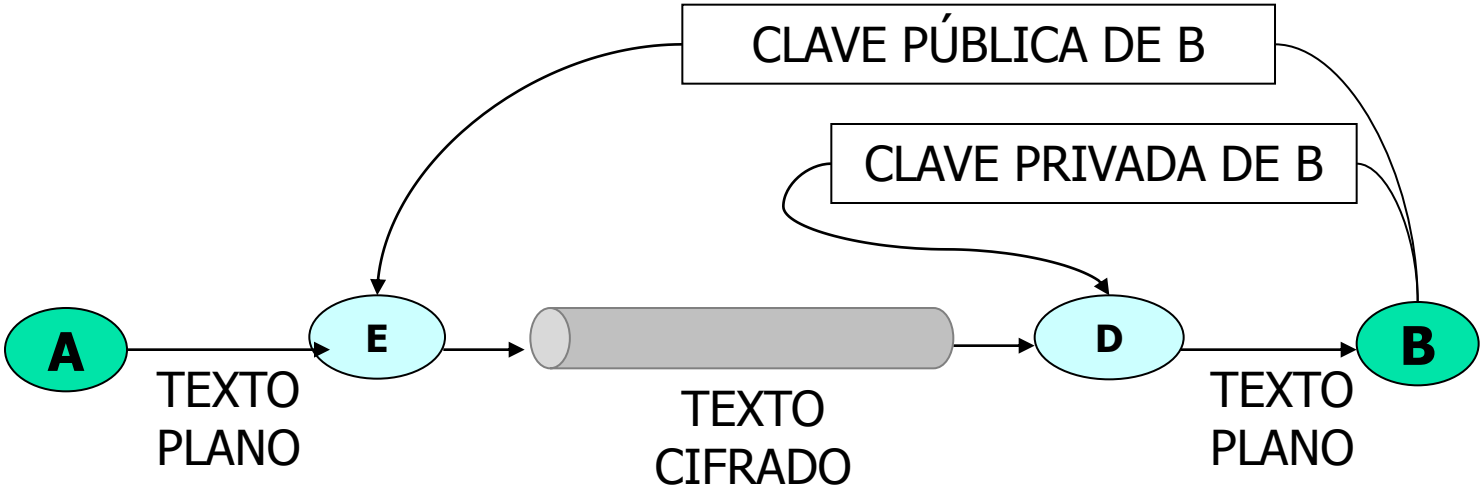


Funciones HASH y MAC

- **Cifrado del mensaje:**

CIFRADO ASIMÉTRICO

- Confidencialidad ✓
- ~~Autenticación~~



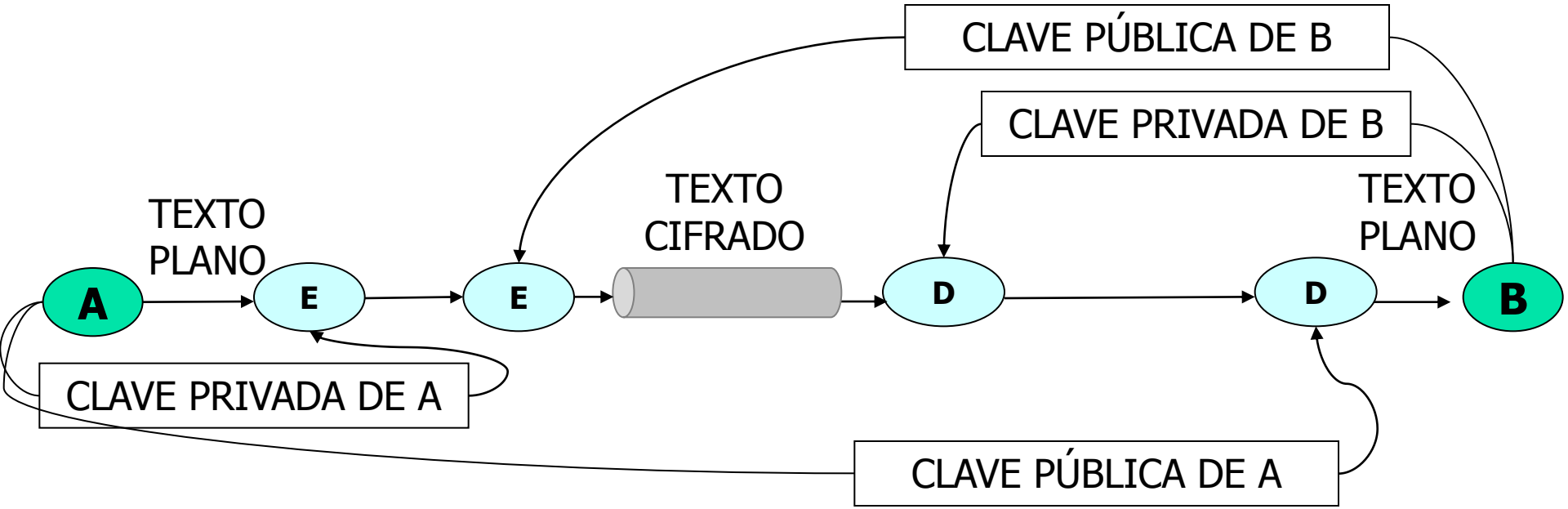
Comunicación cifrada de A a B

Funciones HASH y MAC

- **Cifrado del mensaje:**

CIFRADO ASIMÉTRICO

- Confidencialidad ✓
- Autenticación ✓
- Cómo garantizar que el texto plano es el original? Checksum



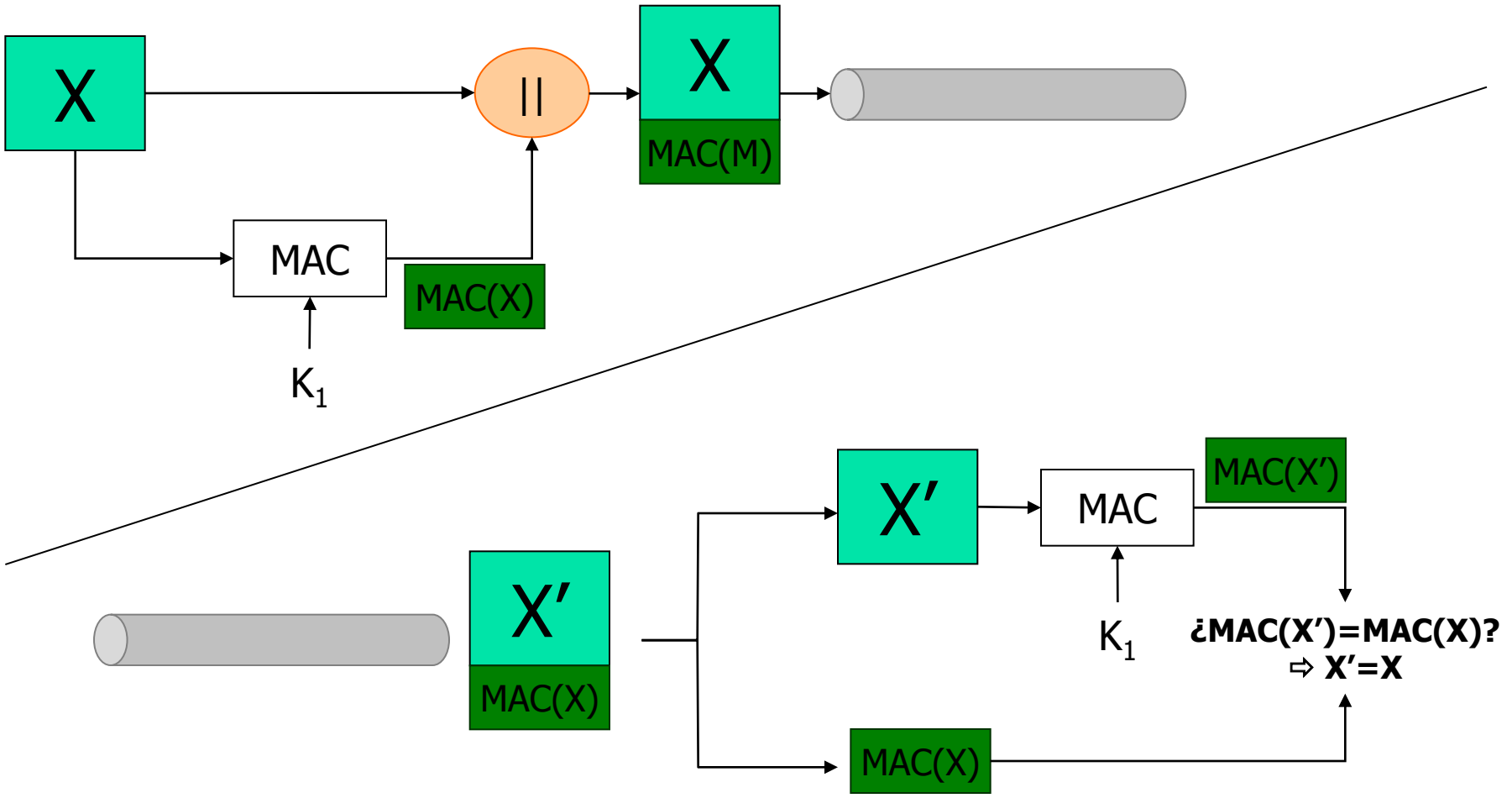
Funciones HASH y MAC

- **Código de autenticación del mensaje** (Message Authentication Code, MAC)
  - Clave privada
  - Bloque de datos de tamaño fijo ≡ MAC
  - Se añade al mensaje

AUTENTICACIÓN

- El mensaje no fue alterado
- Mensaje proviene del verdadero emisor
- Si se incluyen números de secuencia no es posible modificación temporal por parte de un atacante
- Para que haya confidencialidad hay que cifrar texto plano+MAC

Funciones HASH y MAC



Funciones HASH y MAC

- Codigo MAC=$C_k(X)$, si un atacante conoce la función C pero no conoce la clave k:
  - Computacionalmente intratable crear un mensaje X' tal que $C_k(X') = C_k(X)$
  - Para dos mensajes seleccionados de modo aleatorio X y X' la probabilidad de que $C_k(X') = C_k(X)$ es de 2^{-n} donde n es la longitud del MAC
  - Sea X' = f(X) la probabilidad de que $C_k(X')=C_k(X)$ es de 2^{-n} donde n es la longitud del MAC

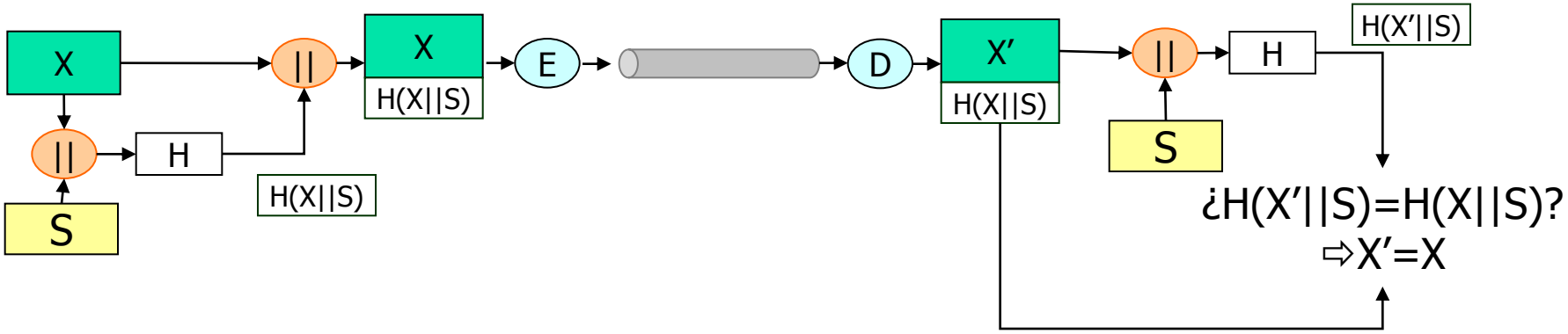
Funciones HASH y MAC

- **Función hash**, acepta un mensaje X de tamaño variable y genera un código hash de tamaño fijo H(X) ≡ Message Digest
  - H(X) ofrece detección de errores
- Posibilidades de autenticación
  1. Texto plano se concatena con código hash y se cifra todo con cifrado simétrico
  2. Se cifra sólo el código hash con cifrado simétrico
  3. Se cifra sólo código hash con cifrado asimétrico (con clave privada de emisor) ⇨ firma digital
  4. Se cifra hash con cifrado asimétrico (con clave privada de emisor) y todo se cifra con cifrado simétrico

Funciones HASH y MAC

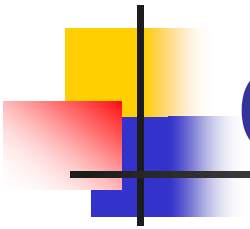
- Posibilidades de autenticación

5. Compartir dos comunicantes un valor secreto S.



Funciones HASH y MAC

- Codigo hash h=H(X)
  - H se puede usar con bloques de datos de cualquier tamaño
  - H genera salida de tamaño fijo
  - H(X) es fácil de obtener
  - Conocido h, es computacionalmente intratable encontrar X tal que H(X)=h
  - Es computacionalmente intratable encontrar X' tal que X'≠X y H(X')=H(X)
  - Es computacionalmente intratable encontrar un par (X, X') tal que H(X)=H(X')



Contenidos

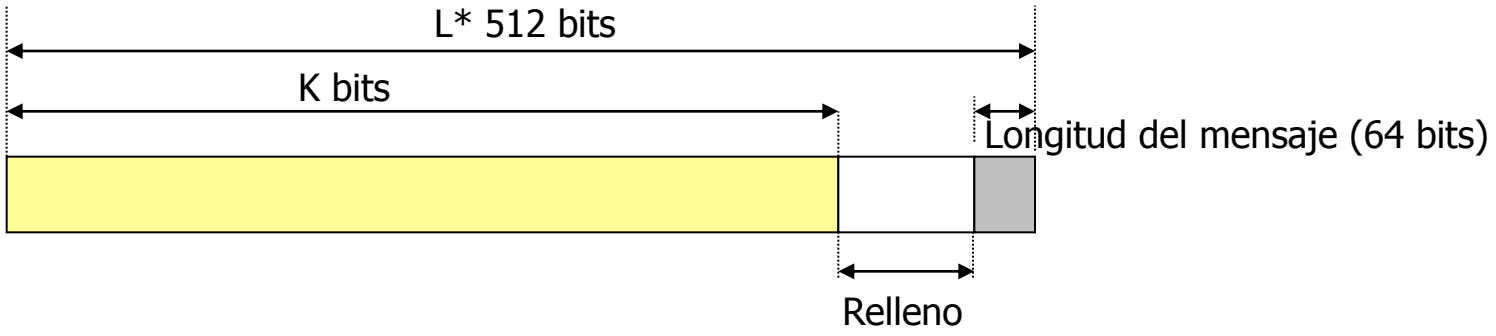
1.1 Introducción ✓

1.2 Funciones HASH y MAC ✓

- 1.2.1 MD5
- 1.2.2 SHA
- 1.2.3 HMAC

MD5

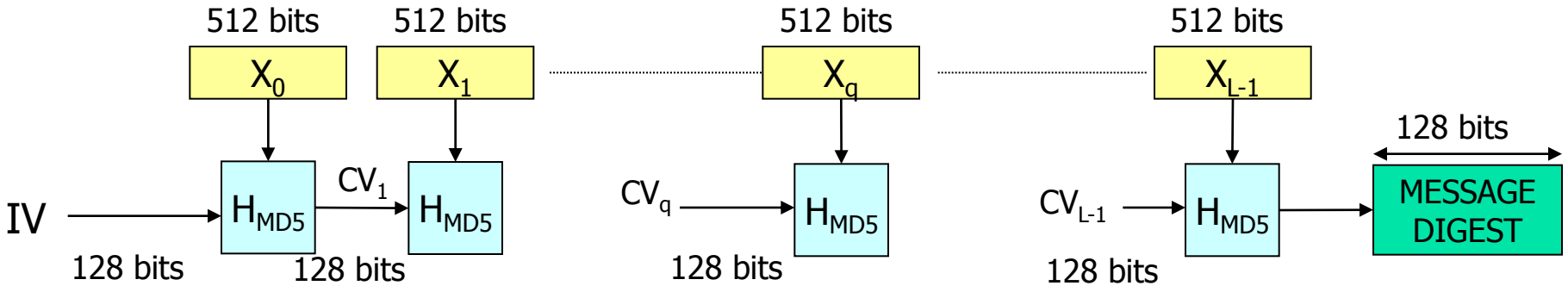
- Algoritmo **Message Digest MD5**, entrada de longitud variable (en bloques de 512 bits) y genera salida de 128 bits (message digest)
- Algoritmo:
  1. Añadir bits de relleno (long. final = 448 mod 512)
  2. Añadir longitud del mensaje original (64 bits)



MD5

Texto plano en bloques X_0, X_1, ..., X_{L-1} ⇨ longitud del mensaje 512*L bits = 16*32*L bits = N palabras de 32 bits (N=16*L)

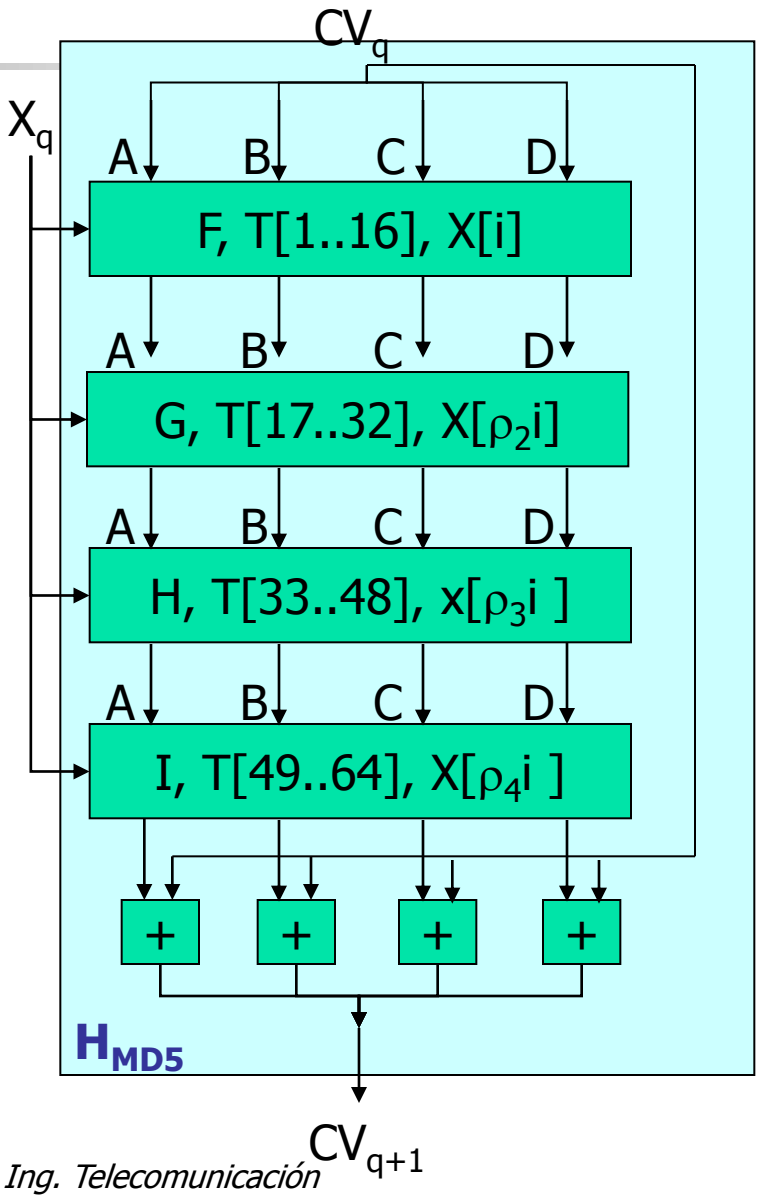
3. Inicializar buffer (128 bits) = (A,B,C,D)
   1. A=67452301; B= EFCDAB89; C=98BADCFE; D=10325476
4. Procesar el texto plano en bloques de 512 bits: comprimir el mensaje aplicando H_{MD5}



5. La salida de la etapa L es el Message Digest

MD5

- Bloques H_{MD5}
  - Cuatro rondas cada una con una función lógica diferente (F, G, H, I)
  - Entrada de ronda: X_q; 128 bits de buffer; contenido tabla T
  - T[1..64] tiene 64 entradas (32 bits) obtenidas de la función seno: T[i]=parte entera (2^{32}·abs(sin(i))) donde i son radianes



MD5

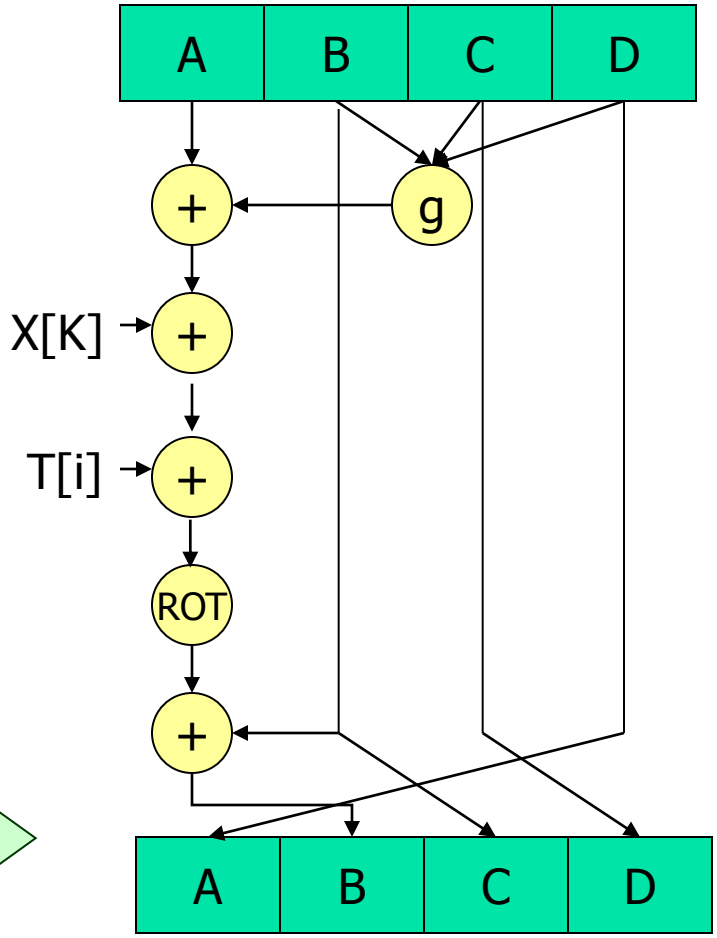
- Bloques H_{MD5}
  - Cada ronda 16 operaciones
  - Operación: b ← b+((a+g(b,c,d)+X[k]+T[i]) <<<s
  - Donde:
    - a,b,c,d son las cuatro palabras del buffer
    - g es una de las funciones F, G, H, I
    - <<<s desplazamiento circulara izquierda de s bits
    - T[i] i-ésima palabra de T

Ronda	Función g	G(b,c,d)
1	F(b,c,d)	(b AND c) OR (b AND d)
2	G(b,c,d)	(b AND d) OR (c AND d)
3	H(b,c,d)	b XOR c XOR d
4	I(b,c,d)	c XOR (b OR d)

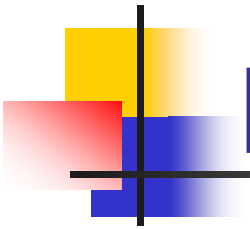
MD5

- Bloques H_{MD5}
  - X[k] = K-ésima palabra de bloque X_q

  En rondas 2, 3 y 4 $\Rightarrow$
  $\rho_2(i)=(1+5i) \bmod 16$
  $\rho_3(i)=(5+3i) \bmod 16$
  $\rho_4(i)=(1+5i) \bmod 16$

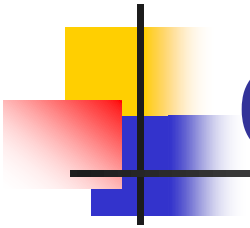


Ejemplo de una iteración dentro de una ronda



MD5

- Cada bit del *message digest* es función de todos los bits de entrada
- Probabilidad de encontrar dos mensajes que generen mismo *message digest* es del orden de 2^{64} operaciones
- Probabilidad de encontrar un mensaje conocido el message digest es del orden de 2^{128} operaciones
- Nuevas propuestas: SHA-1 y RIPEMD-168

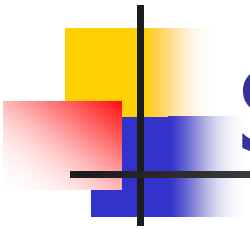


Contenidos

1.1 Introducción ✓

1.2 Funciones HASH y MAC ✓

- 1.2.1 MD5 ✓
- 1.2.2 SHA
- 1.2.3 HMAC



SHA

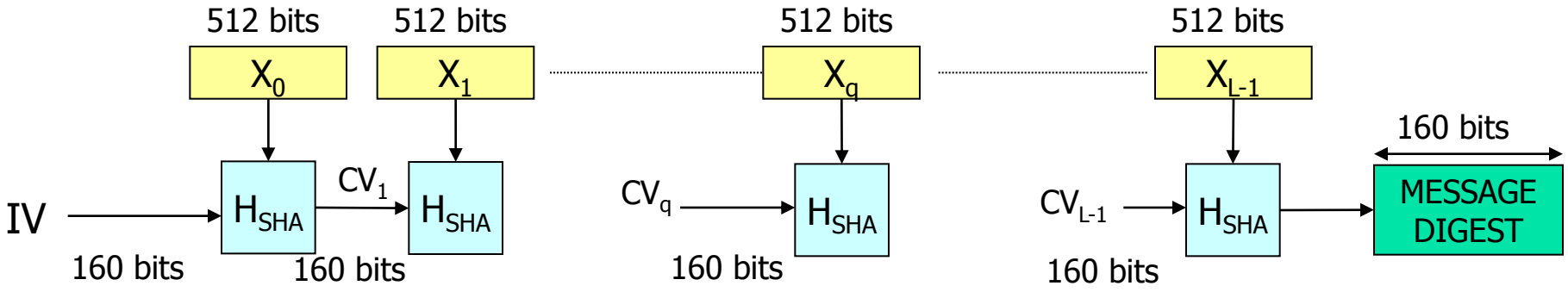
- SHA (Secure HAsh)
- Basado en el MD4
- Entrada ≡ mensaje de longitud máxima < 2^{64} bits (procesada en bloques de 512 bits)
- Message disgest ≡ 160 bits
- Estructura global similar al MD5

SHA

- Algoritmo:
  - Añadir bits de relleno (long. final 448 mod 512)
  - Añadir longitud de mensaje original (64 bits)
  - Inicializar buffer (160 bits) = (A,B,C,D,E)

A=67452301; B= EFCDAB89; C=98BADCFE; D=10325476; E=C3D2E1F0

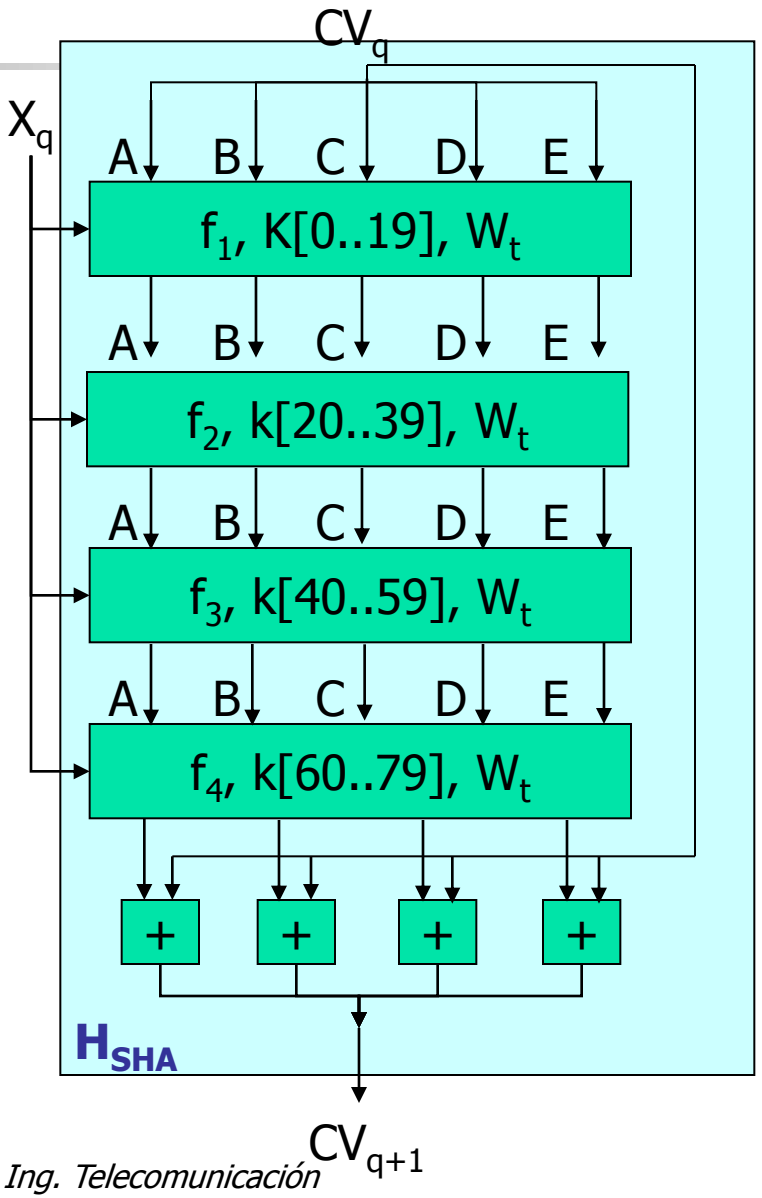
  - Procesar bloques de 512 bits a través de módulos H_{SHA}



SHA

- Bloques H_{SHA}
  - Cuatro rondas cada una con una función lógica diferente (f_1, f_2, f_3, f_4)
  - Cada ronda 20 operaciones
  - En cada ronda se usa constante K_t ($0 \le t < 80$)

Operacion nº	Hexadecimal	Tomar parte entera de
$0 \le t < 20$	K_t= 5A827999	$2^{30}\sqrt{2}$
$20 \le t < 40$	K_t= 6ED9EBA1	$2^{30}\sqrt{3}$
$40 \le t < 60$	K_t= 8F1BBCDC	$2^{30}\sqrt{5}$
$60 \le t < 80$	K_t=CA62C1D6	$2^{30}\sqrt{10}$



SHA

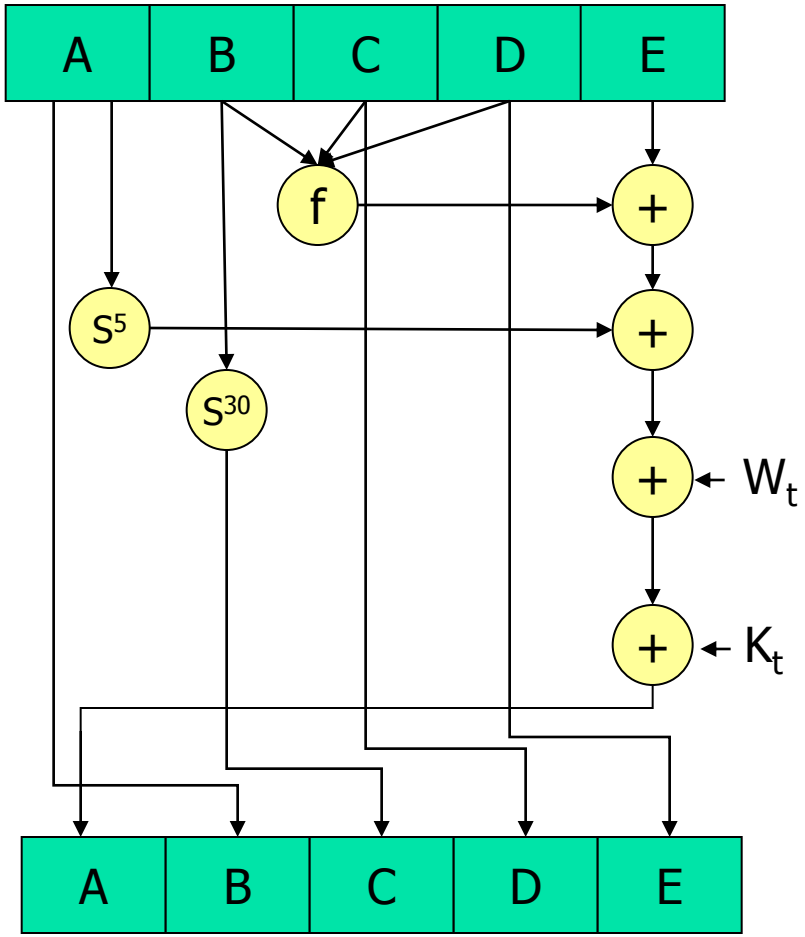
- Bloques H_{SHA}
  - Operación: $A \leftarrow (E + f(t,B,C,D) + S^5(A) + W_t + K_t$
    - A, B, C, D, E palabras de 32 bits del buffer
    - t ≡ nº de operación
    - f(t, B, C, D) ≡ función lógica de la operación t
    - S^k ≡ desplazamiento circular a izquierda k bits
    - W_t ≡ palabra de 32 bits derivada de la entrada de 512 bits

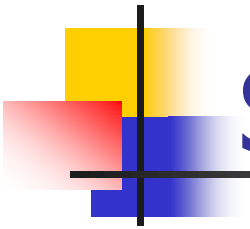
      $W_t = S^1 (W_{t-16} \oplus W_{t-14} \oplus W_{t-8} \oplus W_{t-3})$
    - K_t ≡ constante aditiva

Operación nº	Función f	Valor
0≤t<20	$f_1=f(t,B,C,D)$	(B AND C) OR (B AND D)
20≤t<40	$f_2=f(t,B,C,D)$	B XOR C XOR D
40≤t<60	$f_3=f(t,B,C,D)$	(B AND C) OR (B AND D) OR (C AND D)
60≤t<80	$f_4=f(t,B,C,D)$	B XOR C XOR D

SHA

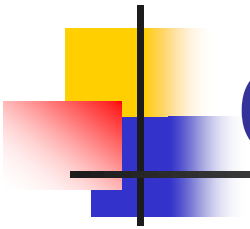
Ejemplo de una operación dentro de una ronda





SHA

- SHA vs. MD5
  - Probabilidad de encontrar dos mensajes que generen mismo *message digest* es del orden de 2^{80} operaciones
  - Probabilidad de encontrar un mensaje conocido el message digest es del orden de 2^{160}
  - No se conocen ataques por criptoanálisis a SHA
  - Ambos funcionan bien en arquitecturas de 32 bits, SHA más lento en mismo hardware
  - Sencillos



Contenidos

1.1 Introducción ✓

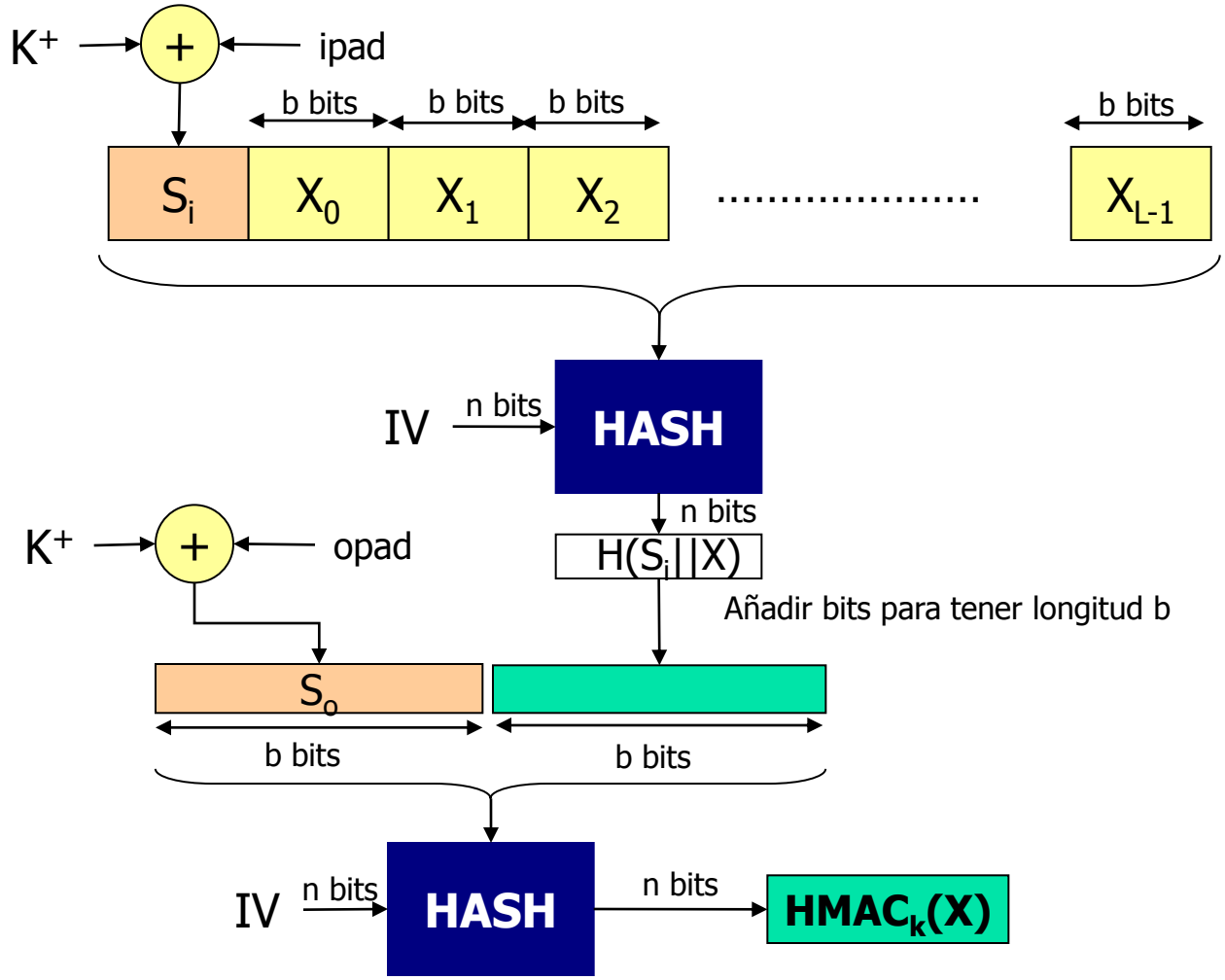
1.2 Funciones HASH y MAC ✓

- 1.2.1 MD5 ✓
- 1.2.2 SHA ✓
- 1.2.3 HMAC

HMAC

- Desarrollar un MAC a partir de un código hash
- HMAC (RFC 2104)
- La función hash se considera una caja negra
- Algoritmo:
  1. Añadir ceros a la izquierda de k para crear cadena de bits de longitud b $\rightarrow K^+$
  2. (K^+) XOR (ipad) $\rightarrow$ bloque S_i de b bits
  3. Concatenar M a S_i
  4. Aplicar función hash H al flujo generado en 3.
  5. (K^+) XOR (opad) $\rightarrow$ bloque S_o de b bits
  6. Concatenar código hash de 4 con S_o
  7. Aplicar función hash H al flujo generado en 6 $\rightarrow$ resultado HMAC

HMAC



ipad ≡ 00110110

opad ≡ 01011010