Segurança da Informação

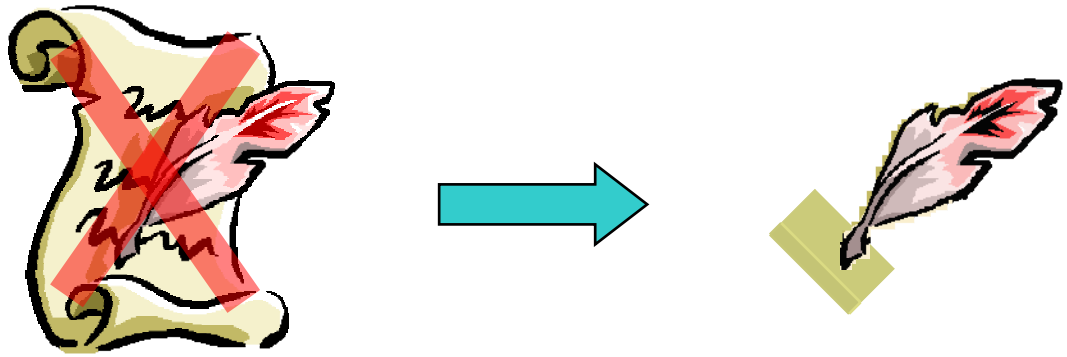
Resumos Criptográficos
(Funções de Hash)

Funções de Hash

- AKA *resumos criptográficos*.
- Redundâncias anexadas a mensagens com o propósito de detectar alterações.
- Dependem exclusivamente da mensagem (sem chave):

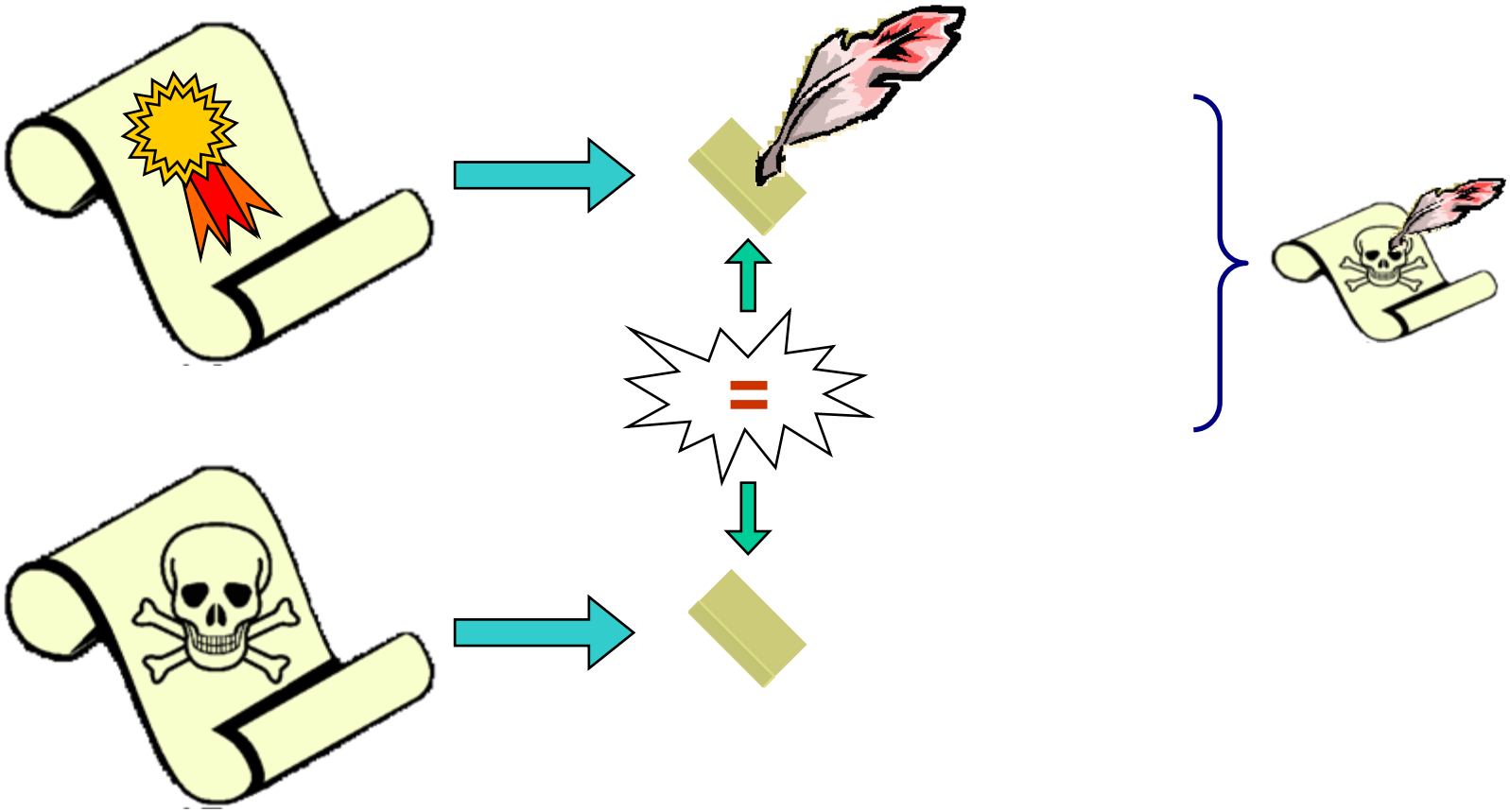
Resumos Criptográficos

- Assinaturas digitais *não* são aplicadas ao *conteúdo* de documentos eletrônicos, mas a *resumos* do conteúdo.
- Motivo: algoritmos de assinatura são muito mais *lentos* que funções de resumo (*hash*).



- Problemas?

Resumos Criptográficos



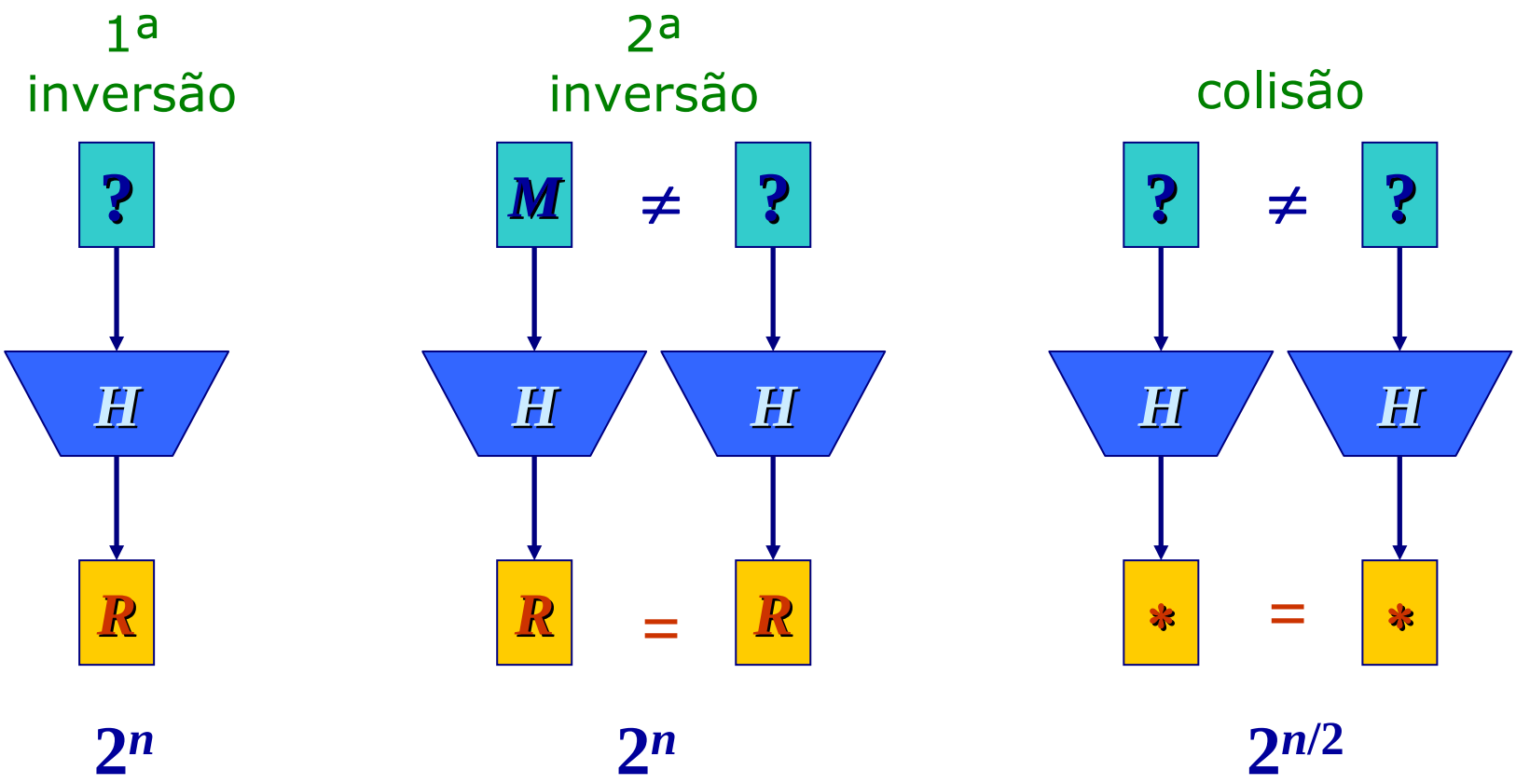
Propriedades Fundamentais

- (*Resistência a primeira inversão*) Dado um resumo R, é *inviável* encontrar uma mensagem M tal que $R = H(M)$.

- (*Resistência a segunda inversão*) Dado um resumo R e uma mensagem M_1 tal que $R = H(M_1)$, é *inviável* encontrar uma outra mensagem $M_2 \neq M_1$ tal que $R = H(M_2)$.

- (*Resistência a colisões*) É *inviável* encontrar duas mensagens M_1 e M_2 tais que $H(M_1) = H(M_2)$.

Propriedades Fundamentais



Outras Propriedades

- A alteração de 1 bit da entrada causa a alteração de cerca de metade dos bits de saída (avalanche completa).
- É inviável prever o valor de qualquer bit de saída (balanço perfeito).
- É inviável inverter a função parcialmente (recuperar apenas alguns bits da entrada).
- As propriedades fundamentais valem para qualquer sub-cadeia da saída.

Outras Propriedades

- Certas propriedades empíricas transcendem os objetivos originais de projeto das funções.
- Necessário analisar quantitativamente até que ponto essas propriedades são válidas.
- Algumas propriedades não são válidas em certas aplicações.
- Construções auxiliares estabelecem essas propriedades (algoritmos de MAC).

Projeto de Funções de Hash

- Não existe ainda uma teoria completa para o projeto sistemático de uma função de hash.
- Quase todas as funções de hash existentes, em particular as padronizadas e as mais resistentes a ataques, são derivadas de cifras de bloco.
- A estrutura genérica dessas construções utiliza a cifra de bloco subjacente como uma caixa preta (definida apenas por sua interface).

Projeto de Funções de Hash

- É possível quantificar a segurança de certas construções por *redução*: se for possível violar as propriedades fundamentais dessas construções, então é possível quebrar a cifra de bloco subjacente (qualquer que seja).

- Paradigma Merkle-Damgård: estado interno (resumo parcial) atualizado iterativamente, aplicando uma função de compressão à mensagem particionada em blocos.

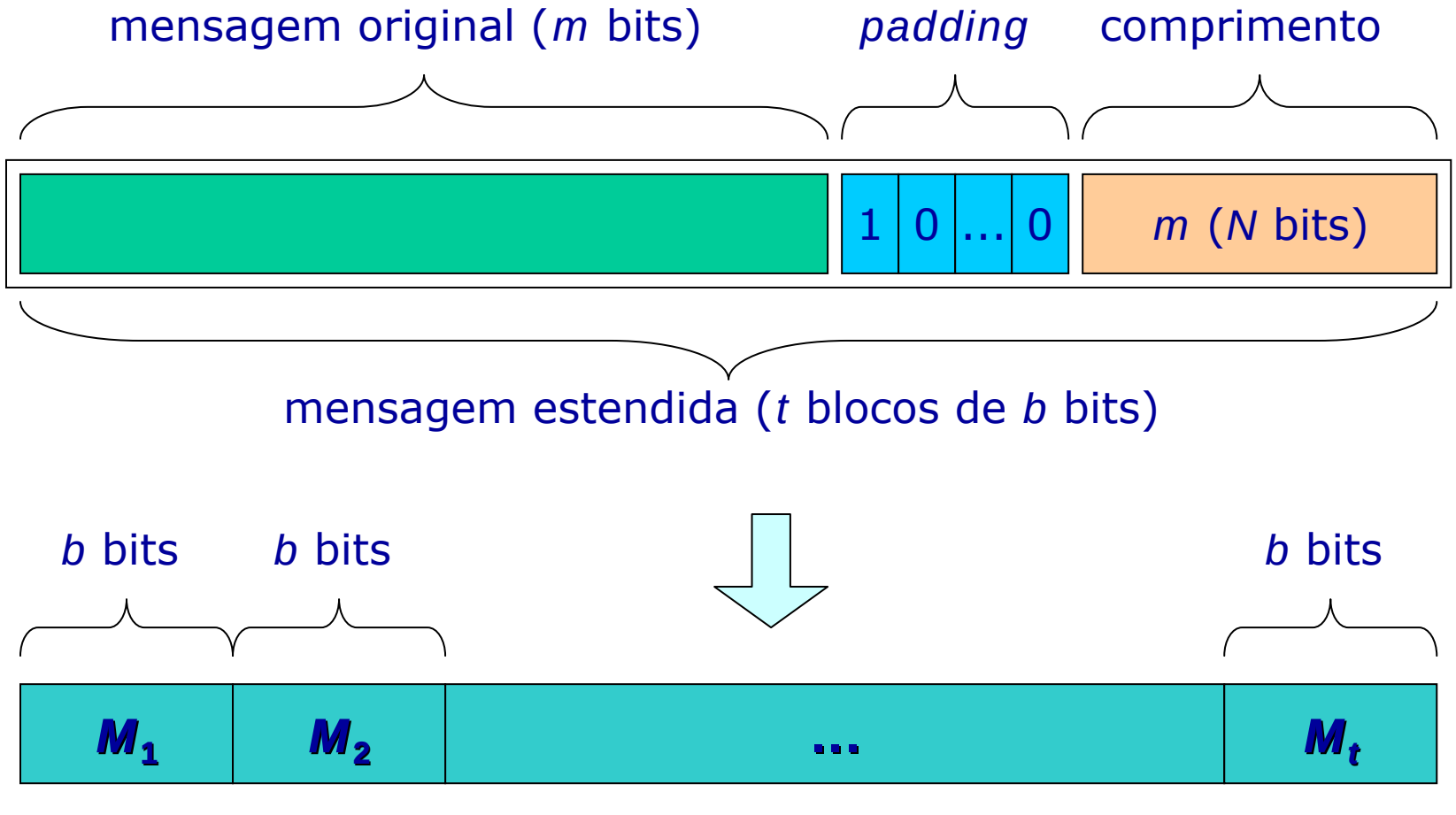
Complemento Merkle-Damgård

- Parâmetros:
  - Limite de tamanho de mensagens: $m < 2^N$ bits.
  - Tamanho de bloco: b bits.
  - Tamanho de estado interno: h bits.

- Acrescenta-se à mensagem um bit '1', seguido de tantos bits '0' quantos forem necessários para que o tamanho do resultado seja um múltiplo de b bits, exceto por um espaço final de N bits.

- Anexa-se o valor do tamanho m da mensagem original em bits na forma de um inteiro de N bits, com zeros binários à esquerda.

Complemento Merkle-Damgård



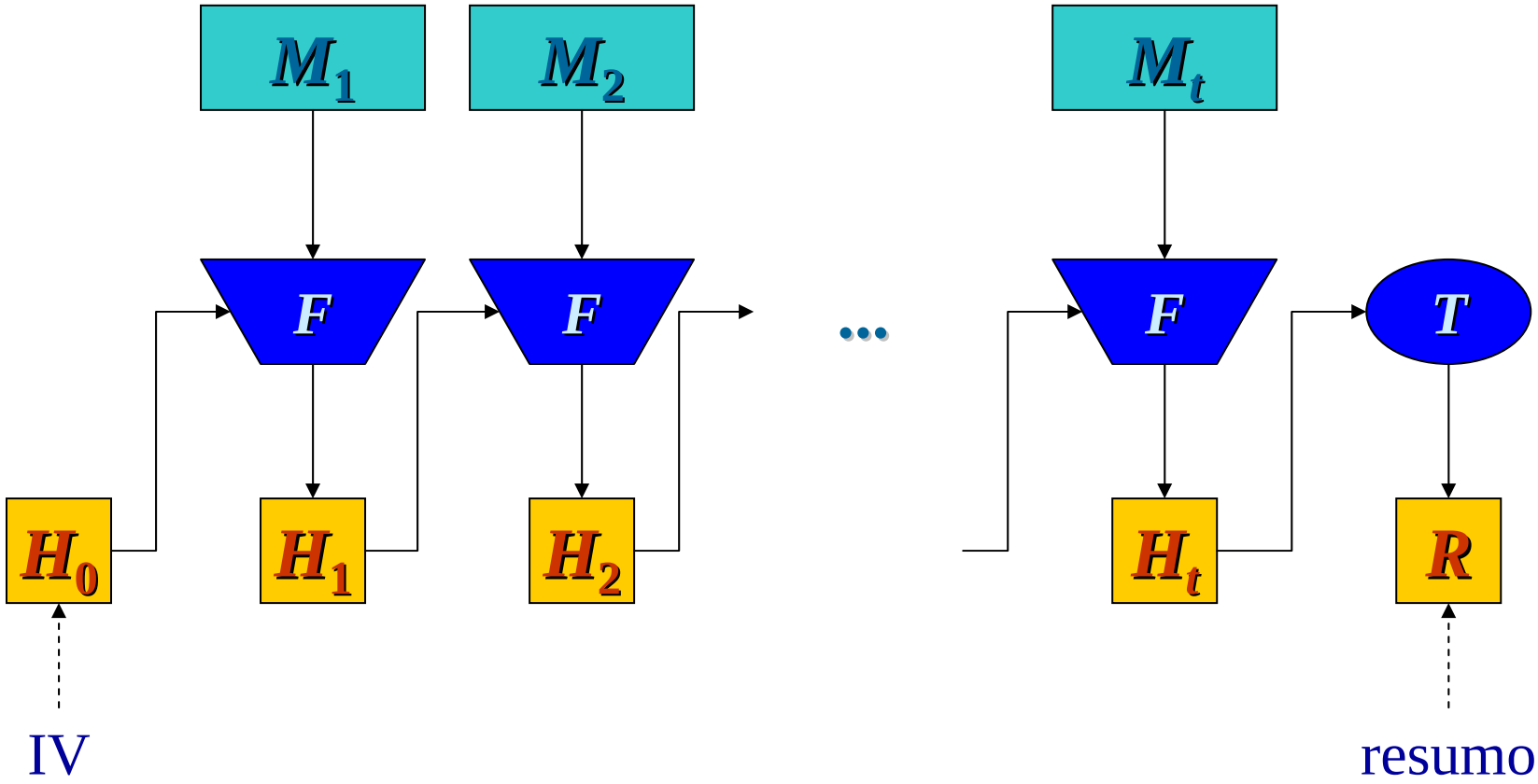
Função de Compressão

- Construção propriamente derivada da cifra de bloco.
- Atualiza o estado interno H_i da função de hash (valor parcial do resumo) a partir do estado anterior H_{i-1} e do bloco corrente Mi da mensagem:

$$(H_{i-1}, M_i) \xrightarrow{F} H_i$$

- H_{i-1} e M_i participam como entrada e/ou chave da cifra subjacente, e são combinados com a saída cifrada para tornar a construção irreversível.
- O valor R do resumo é o valor final do estado interno H_t, possivelmente transformado (e.g. truncado).

Função de Compressão

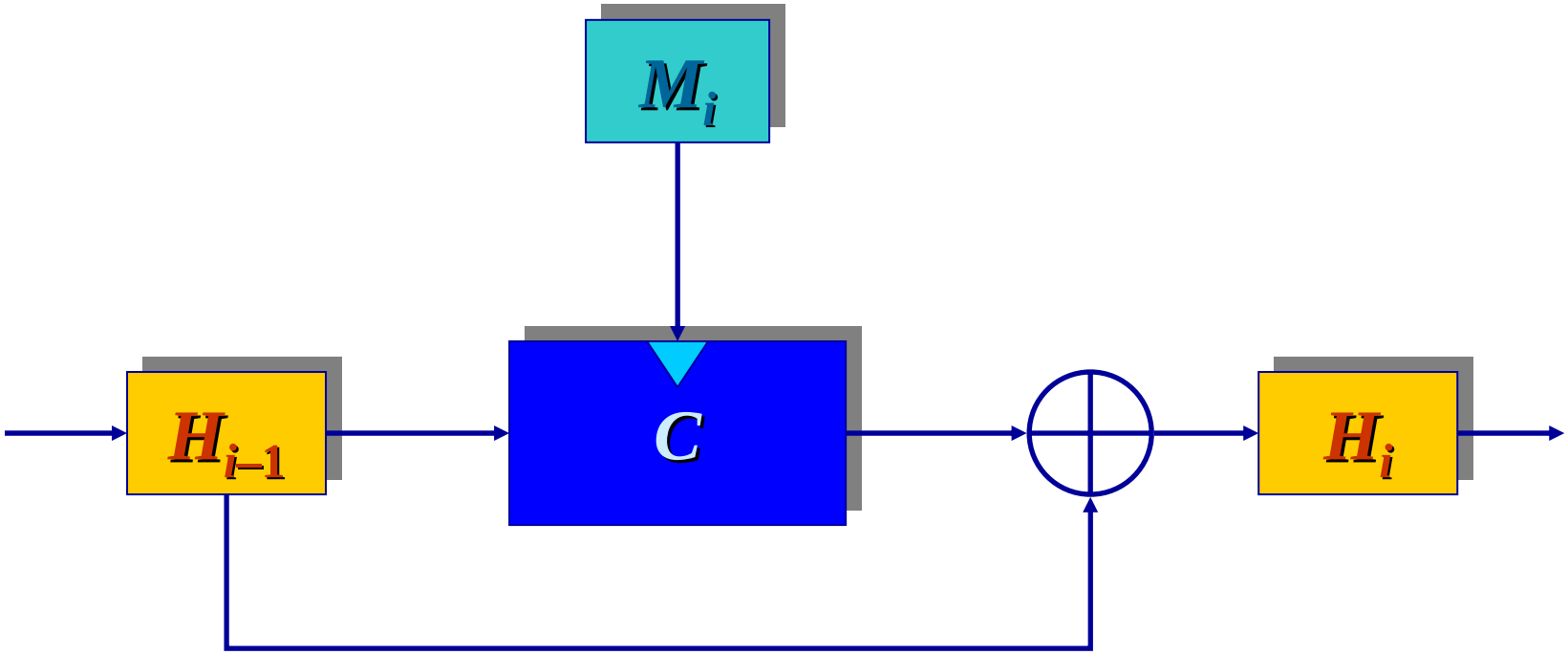


Função de Compressão

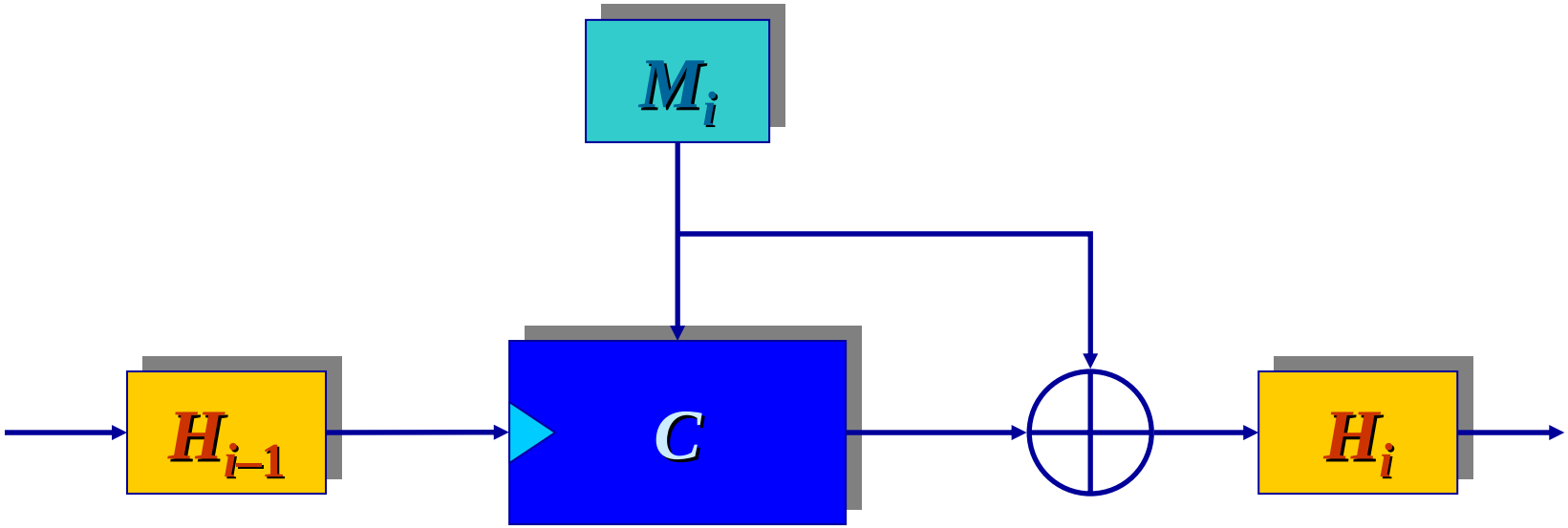
- Três dentre as funções de compressão demonstravelmente tão seguras quanto a cifra de bloco são mais amplamente empregadas:
  - Davies-Meyer;
  - Matyas-Meyer-Oseas;
  - Miyaguchi-Preneel.

- A primeira é mais popular, mas as outras duas são quantitativamente mais seguras.

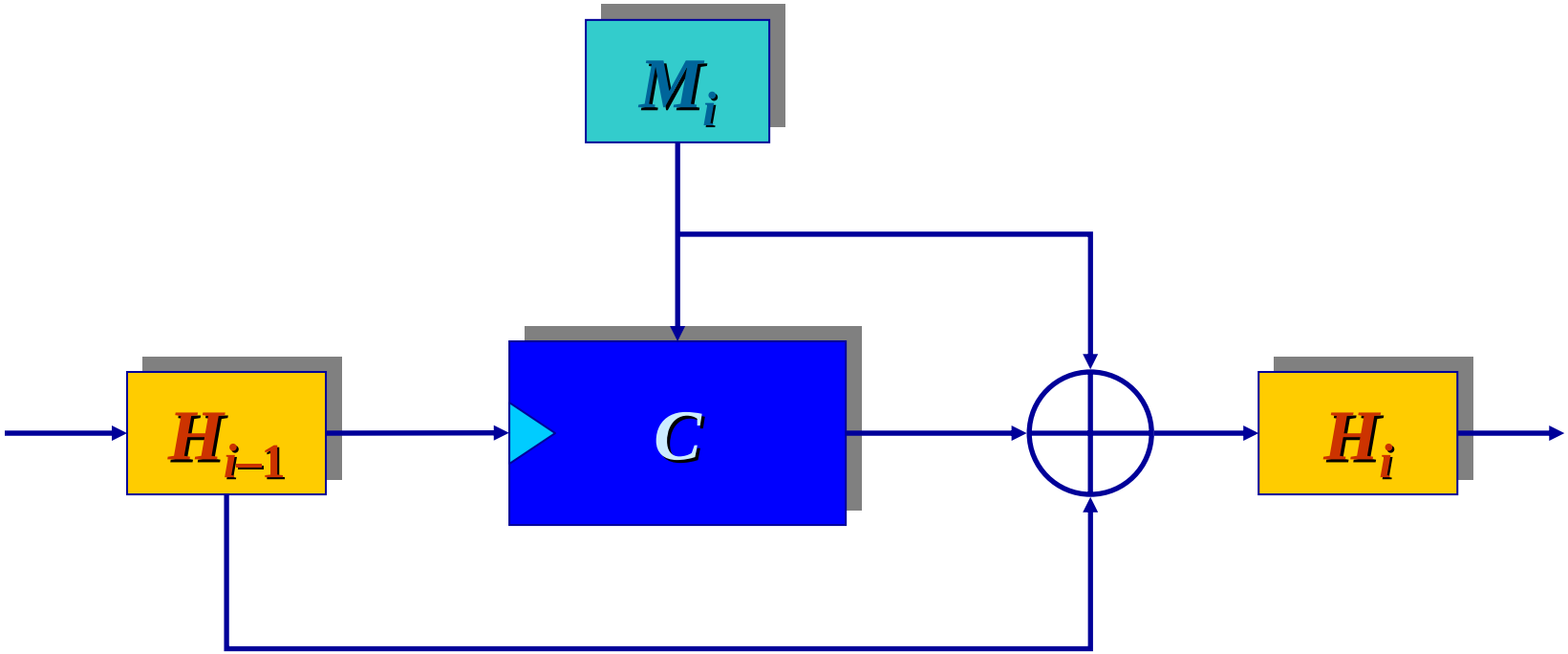
Davies-Meyer



Matyas-Meyer-Oseas



Miyaguchi-Preneel



Construções paralelizáveis

- Cálculo paralelo com hardware disponível.
- Incremental em plataformas sequenciais.
- Baixo custo para correção e atualização de resumos acumulados (pequena alteração em grande massa de dados).
- Paradigma Bellare-Micciancio: "randomize-then-combine".

Construções paralelizáveis

- Particionamento da mensagem:

  $M = M_1 M_2 ... M_m$; $|M_i| = n$ $(i < m)$, $|M_m| \leq n$.

- Função de "randomização":

  $F: \mathbb{N} \times \{0, 1\}^b \rightarrow \{0, 1\}^h$

  $R_i \leftarrow F(i, M_i)$

- Combinação: operação $\bullet$ de grupo abeliano com 2^h elementos.

  $H(M) \leftarrow R_1 \bullet ... \bullet R_m$.

Construções paralelizáveis

- *F* pode ser a função de compressão de uma função de hash comum, desde que livre de colisões.
- Operação de combinação:
  - MuHash: multiplicação módulo $p \approx 2^h$ (lenta, mas equivalente ao PLD se *F* for ideal).
  - AdHash: adição módulo $p \approx 2^h$ (rápida, e equivalente ao problema da mochila ponderada se *F* for ideal).
  - XHash: XOR (trivialmente quebrável).

Funções de Hash

- Família MD
  - Desenvolvida por Ronald Rivest.
  - MD2, MD4, MD5.
  - Completamente quebrada.

- Família SHA
  - Desenvolvida pela NSA.
  - SHA-0, SHA-1: projeto inspirado em MD4.
  - SHA-2 (quatro funções: SHA-224, SHA-256, SHA-384, SHA-512).

Família MD

****

CONTRACT

At the price of $**1**76,495 Alf Blowfish

sells his house to Ann Bonidea…

f_{MD4}(IV || "**********_CONTRACT__At_the_price_of_$**1**76,495_Alf_Blow") = ***h***

****

CONTRACT

At the price of $**2**76,495 Alf Blowfish

sells his house to Ann Bonidea…

f_{MD4}(IV || "**********_CONTRACT__At_the_price_of_$**2**76,495_Alf_Blow") = ***h***

SHA-0

- Função inicialmente proposta como padrão americano para uso em assinaturas digitais, com resumos de 160 bits.
- Vulnerabilidade anunciada (sem detalhes) em 1993 pela NSA, redescoberta em 1998 por Chabaud e Joux.
- Quebrada em 2004 (Joux et al.; Wang et al.)

SHA-0

Referência	Passos do ataque	Tempo (PC)
Chabaud&Joux (1998)	2^{61}	
Biham et al. (2004)	2^{51}	20 anos
Wang et al. (2005)	2^{39}	
Naito et al. (2006)	$2^{36} - 2^{40}$	100 h
Manuel&Peyrin (2008)	2^{33}	1 h

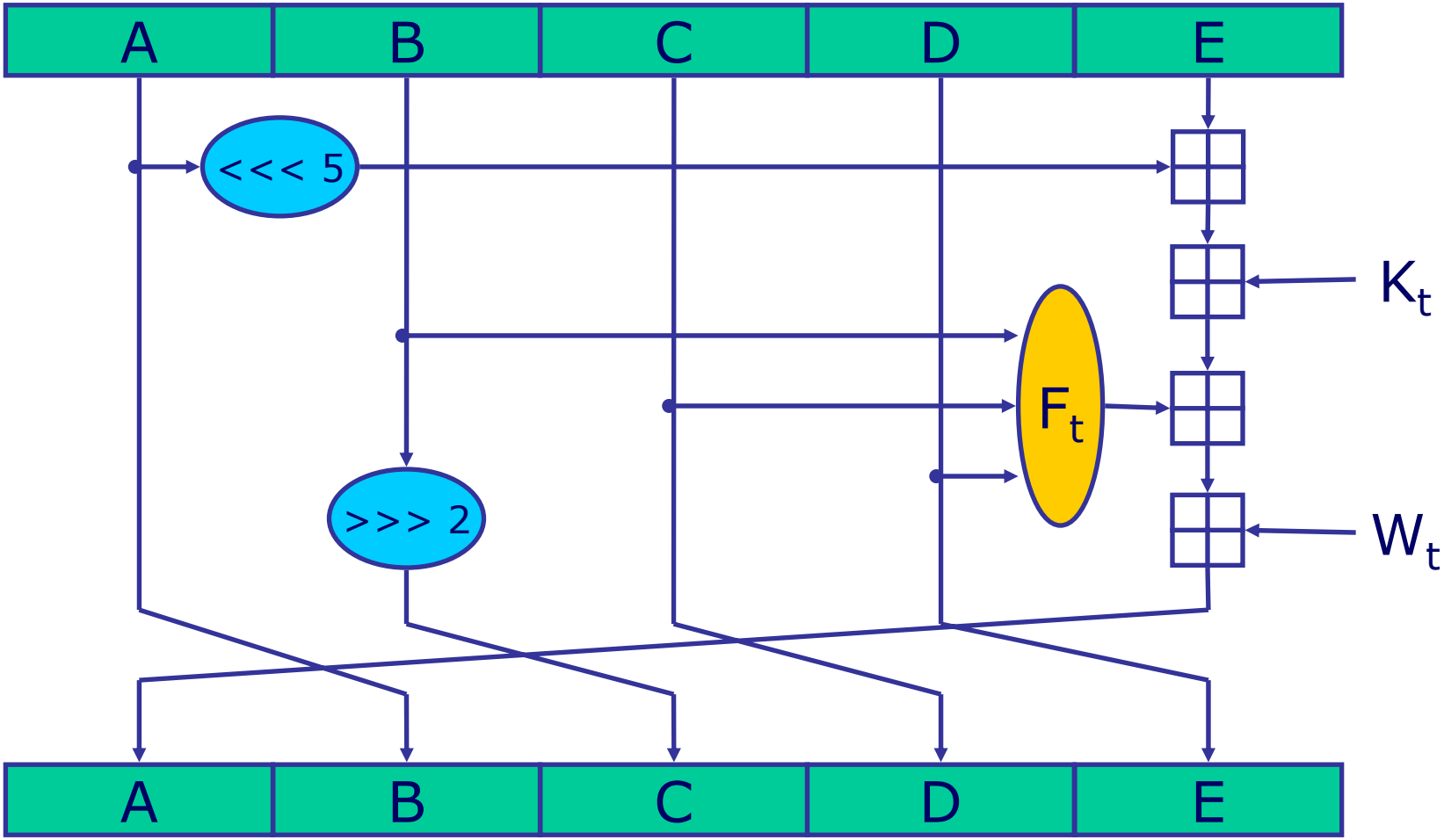
SHA-1

- Supostamente corrige a vulnerabilidade da função SHA-0.
- Função mais amplamente empregada em aplicações não legadas.
- Enfraquecida em 2005 (Wang et al.) e 2006 (de Cannière et al.): $\approx 2^{61}$ passos para obter colisões vs. 2^{80} passos projetados.
- Futuro incerto:
  - colisão viável para 70 dos 80 passos,
  - obsolescência anunciada para 2010.

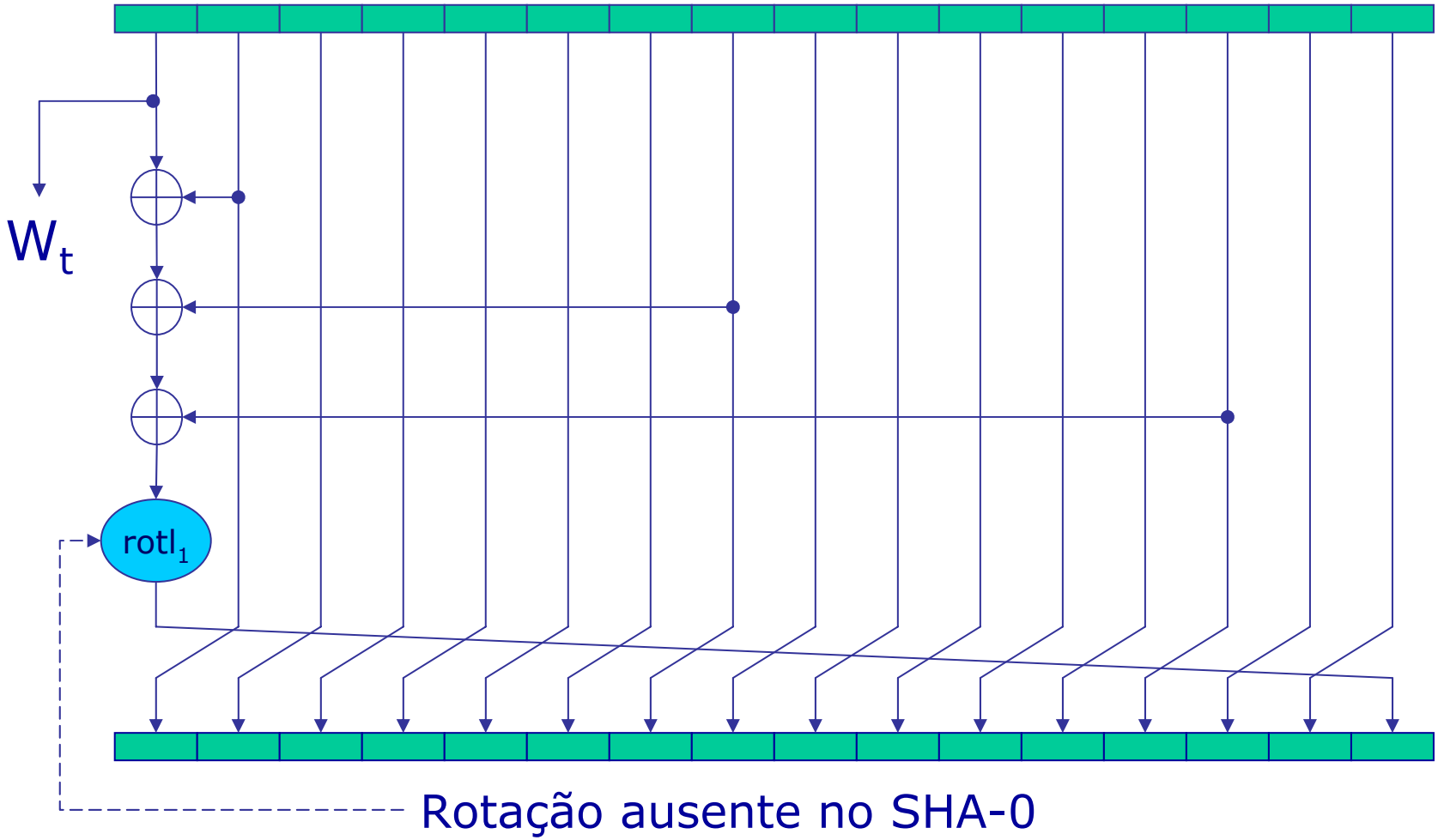
Estrutura básica

- Complemento Merkle-Damgård:
  - b = 512 bits.
  - N = 64 bits.
  - h = 160 bits.
- Compressão Davies-Meyer.
- Cifra de bloco dedicada:
  - Operações simples em processadores de 32 bits ou em hardware.
  - Função de compressão variável, distribuída em 4×20 passos.
  - Escalonamento linear de mensagem.

Estrutura básica



Escalonamento de Mensagem



SHA-2

- Família de quatro funções propostas para equiparar a segurança de hash aos tamanhos de chave do 3DES e do AES.

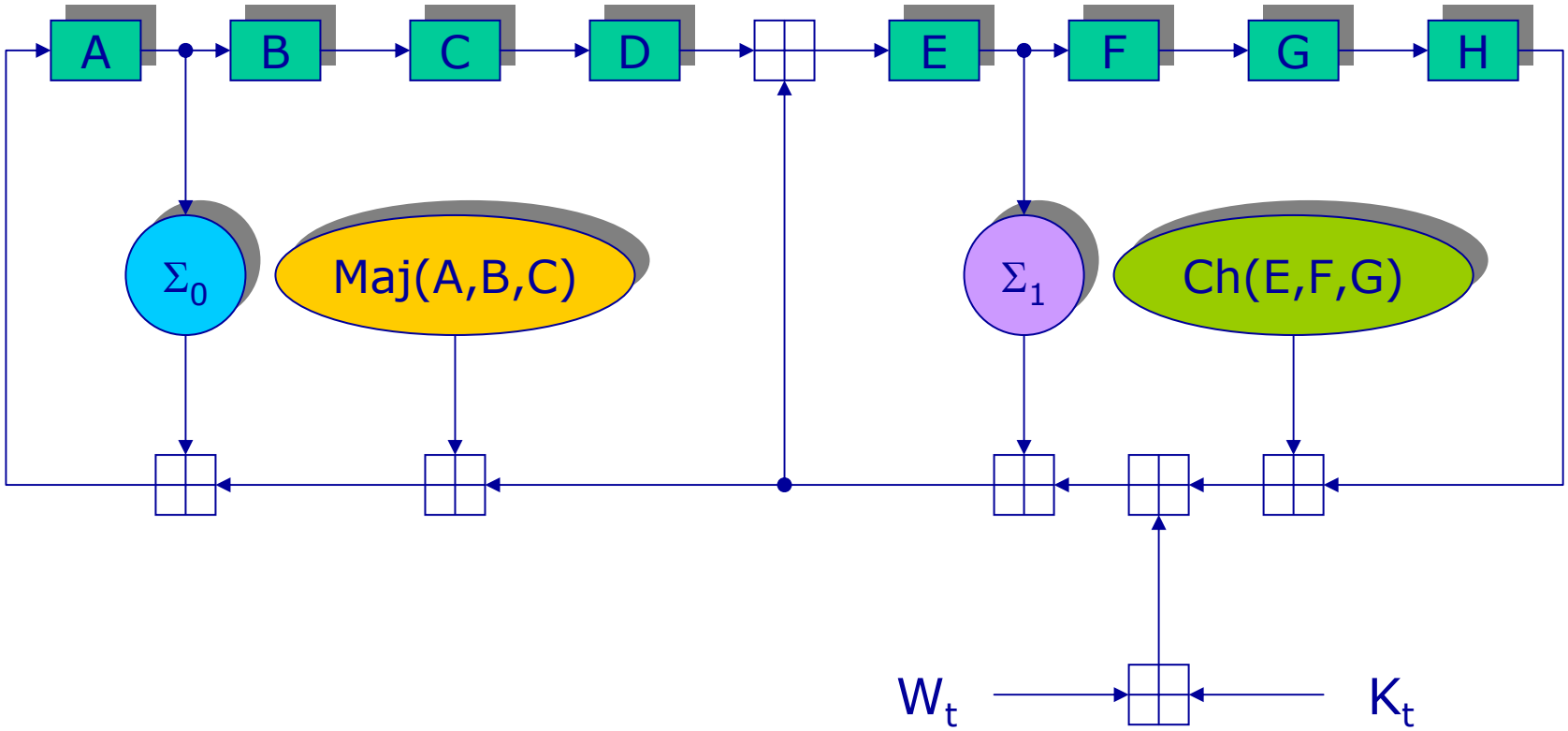
- Pela primeira vez, reconhece-se a estrutura interna como derivada explicitamente de um algoritmo de bloco dedicado.

- Projeto semelhante para as quatro funções, e mais simples que o SHA-1.

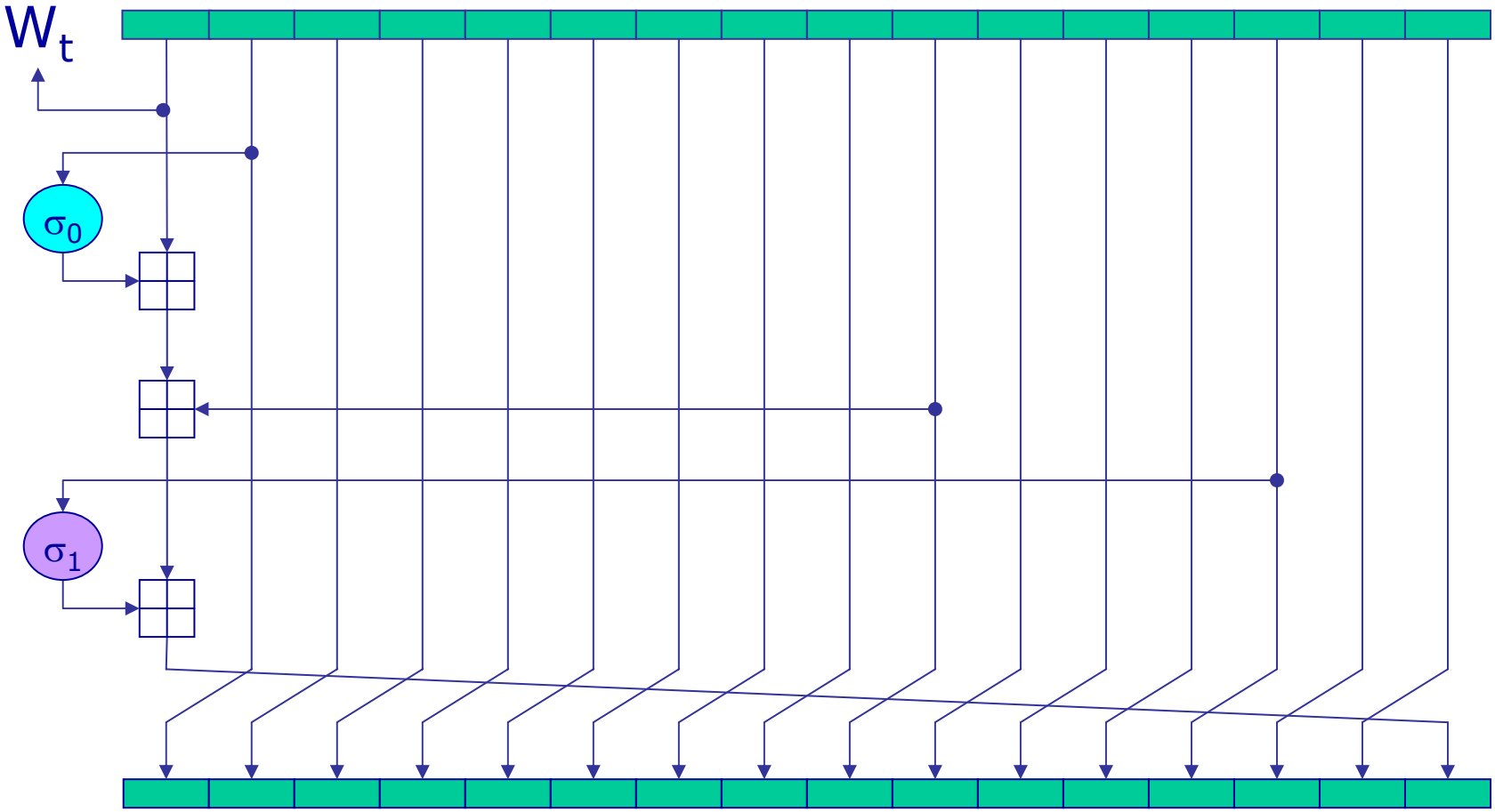
Estrutura Básica de SHA-2

- Semelhante ao SHA-1.
- Complemento Merkle-Damgård.
- Compressão Davies-Meyer.
- Cifra dedicada mais homogênea.
  - SHA-224/256: b = 512 bits, N = 64 bits.
  - SHA-384/512: b = 1024 bits, N = 128 bits.
- SHA-224 e SHA-384: essencialmente iguais, respectivamente, a SHA-256 e SHA-512 (diferem apenas em algumas constantes de inicialização e no truncamento do valor final de hash).

Cifra de bloco dedicada SHA-2



Escalonamento de Mensagem



Função de Compressão: SHA-224 e SHA-256

- Funções auxiliares:

$$\mathrm{Maj}(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

$$\mathrm{Ch}(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$

$$\Sigma_0(x) = \mathrm{rotr}_2(x) \oplus \mathrm{rotr}_{13}(x) \oplus \mathrm{rotr}_{22}(x)$$

$$\Sigma_1(x) = \mathrm{rotr}_6(x) \oplus \mathrm{rotr}_{11}(x) \oplus \mathrm{rotr}_{25}(x)$$

$$\sigma_0(x) = \mathrm{rotr}_7(x) \oplus \mathrm{rotr}_{18}(x) \oplus (x \gg 3)$$

$$\sigma_1(x) = \mathrm{rotr}_{17}(x) \oplus \mathrm{rotr}_{19}(x) \oplus (x \gg 10)$$

Função de Compressão: SHA-384 e SHA-512

- Funções auxiliares:

$$\mathrm{Maj}(x,y,z) = (x \wedge y) \oplus (x \wedge z) \oplus (y \wedge z)$$

$$\mathrm{Ch}(x,y,z) = (x \wedge y) \oplus (\neg x \wedge z)$$

$$\Sigma_0(x) = \mathrm{rotr}_{28}(x) \oplus \mathrm{rotr}_{34}(x) \oplus \mathrm{rotr}_{39}(x)$$

$$\Sigma_1(x) = \mathrm{rotr}_{14}(x) \oplus \mathrm{rotr}_{18}(x) \oplus \mathrm{rotr}_{41}(x)$$

$$\sigma_0(x) = \mathrm{rotr}_{1}(x) \oplus \mathrm{rotr}_{8}(x) \oplus (x \gg 7)$$

$$\sigma_1(x) = \mathrm{rotr}_{19}(x) \oplus \mathrm{rotr}_{61}(x) \oplus (x \gg 6)$$

Outras funções de *hash*

- Família RIPEMD
  - RIPEMD, RIPEMD-128, RIPEMD-160.
  - Projeto europeu (RIPE).
  - RIPEMD original quebrada (Wang et al.).
  - Versões modificadas: margem de segurança insuficiente.
- HAVAL
  - Parametrizada pelo comprimento (128–256 passo 32) e número de passos (3–5).
  - Todas as combinações quebradas.

Outras funções de *hash*

- Tiger
  - Hash de $h = 192$ bits, com truncamento previsto para 160 bits em sistemas legados.
  - Dedicada a processadores de 64 bits.
  - Ataques recentes: colisões contra ~~16~~ ~~19~~ 22 de um total de 24 passos.
  - Segurança limitada a 2^{96}: inferior aos valores mínimos recomendados atualmente ($\geq 2^{112} \Rightarrow h \geq 224$ bits).
- Inúmeras propostas quebradas, mesmo algumas mais recentes (Panama, SMASH, FORK, ...).

Funções "provavelmente" seguras

- VSH ("Very Smooth Hash")
  - Baseada na dificuldade de fatorar números inteiros e de calcular raízes quadradas módulo inteiros compostos.
-  LASH ("LAttice [based] Secure Hash")
  - Baseada na dificuldade de encontrar vetores próximos ou curtos em reticulados.
-  FSB ("Fast Syndrome Based [hash]")
  - Baseada na dificuldade de decodificar códigos lineares

Funções "provavelmente" seguras

- "If a [hash] is *provably* secure, it's *probably* not" (Lars Knudsen).
- Todas ≈25(!) vezes mais lentas que funções convencionais para o mesmo nível de segurança.

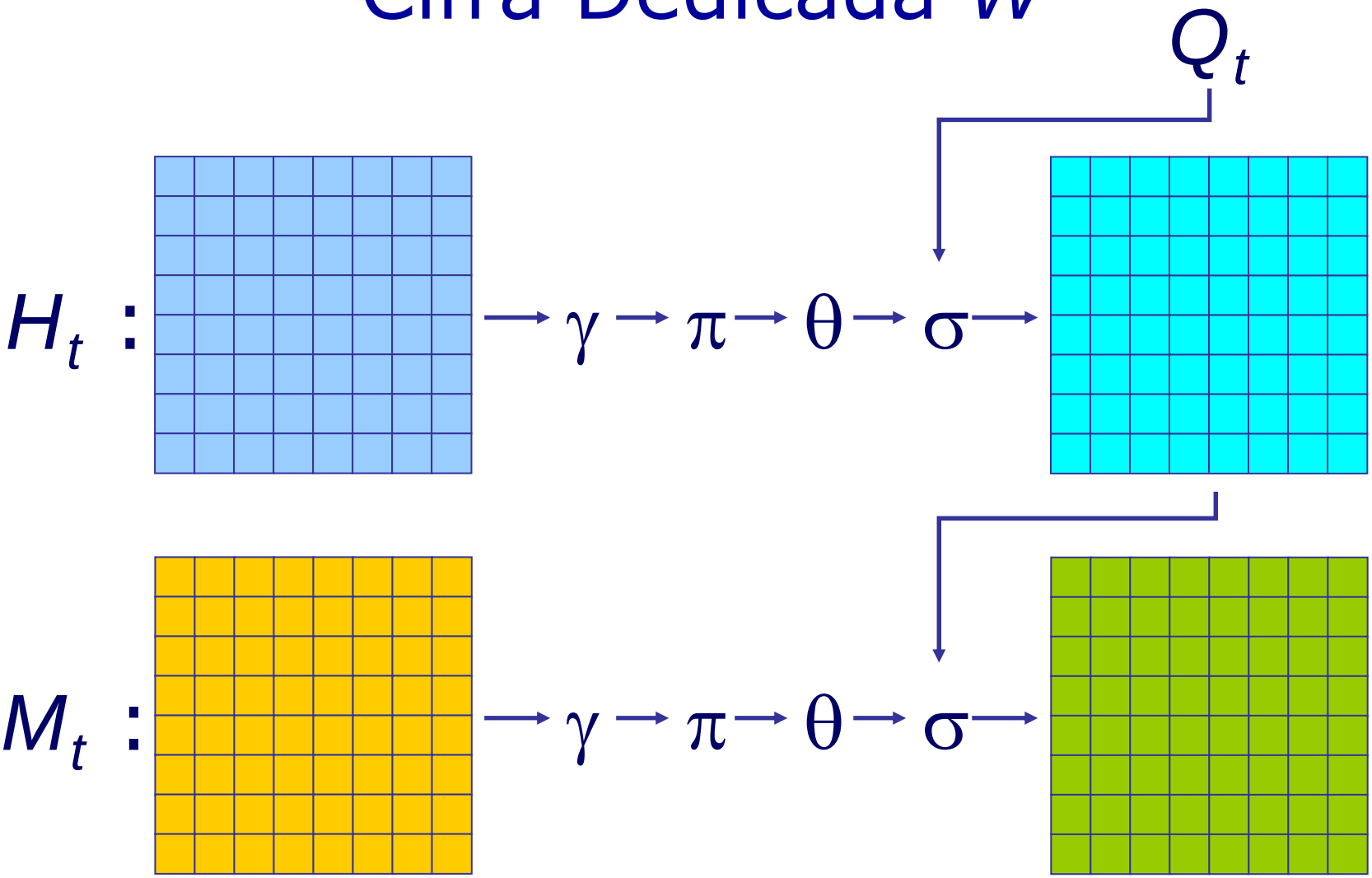
- N.B. Existem funções convencionais de hash que ***não*** foram afetadas pelo "extermínio em massa".

WHIRLPOOL

- Parâmetros:
  - b = 512 bits.
  - h = 512 bits.
  - N = 256 bits (i.e. $|M| < 2^{256}$).
- Compressão Miyaguchi-Preneel.
- Cifra dedicada W, projetada conforme a estratégia de trilha larga (mesma família do AES).

- Projeto belgo-brasileiro (B., Rijmen 2000).


Cifra Dedicada *W*

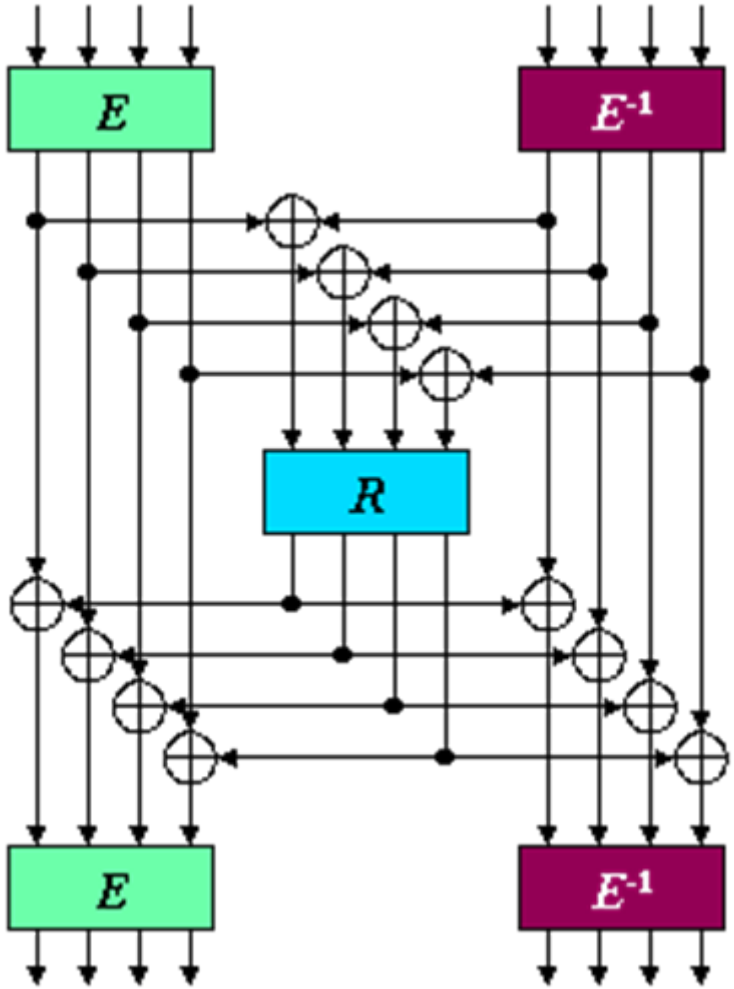


Cifra Dedicada *W*

- γ: substituição byte a byte.
  - $b_{i,j} \leftarrow S[a_{i,j}]$.
- π: permutação circular das colunas.
  - $b_{i,j} \leftarrow a_{(j-i) \bmod 8,j}$.
- θ: combinação linear.
  - $b_{i,j} \leftarrow \sum_k a_{i,k}\, c_{k,j}$ (i.e. $B = AC$).
- σ: adição (xor) de chave ou constante.
  - $b_{i,j} \leftarrow a_{i,j} \oplus h_{i,j}$ ou $a_{i,j} \oplus q_{i,j}$.

Cifra Dedicada *W*

- Tabela de substituição (S-box) 8×8: composição de mini-tabelas 4×4.
- E, E^{-1}: algébricas; R: aleatória.
- Transformação resultante altamente não linear.
- Implementação eficiente em HW e SW.
- Constantes de escalonamento de chaves derivadas diretamente da S-box.



WHIRLPOOL

- Matriz de difusão C corresponde a código corretor de erros MDS.

$$C = \begin{bmatrix} 1 & 1 & 4 & 1 & 8 & 5 & 2 & 9 \\ 9 & 1 & 1 & 4 & 1 & 8 & 5 & 2 \\ 2 & 9 & 1 & 1 & 4 & 1 & 8 & 5 \\ 5 & 2 & 9 & 1 & 1 & 4 & 1 & 8 \\ 8 & 5 & 2 & 9 & 1 & 1 & 4 & 1 \\ 1 & 8 & 5 & 2 & 9 & 1 & 1 & 4 \\ 4 & 1 & 8 & 5 & 2 & 9 & 1 & 1 \\ 1 & 4 & 1 & 8 & 5 & 2 & 9 & 1 \end{bmatrix}.$$

WHIRLPOOL

- Resiste a todas as linhas de ataque ***conhecidas***, inclusive ataques de colisão de blocos múltiplos.

- Estratégia de trilha larga aplicada não só ao laço de cifração de *W*, mas também ao escalonamento de chave.

- Eficiência relativa a SHA-512: ≈10% mais rápida em software, ≈20 vezes mais rápida em hardware.