Disciplina: MDI0082 - SEGURANÇA EM REDES DE COMPUTADORES (08050771) - T01

Assinatura Digital

AULA 07

Requisitos de autenticação de mensagem

- No contexto das comunicações por uma rede, os seguintes ataques podem ser identificados:

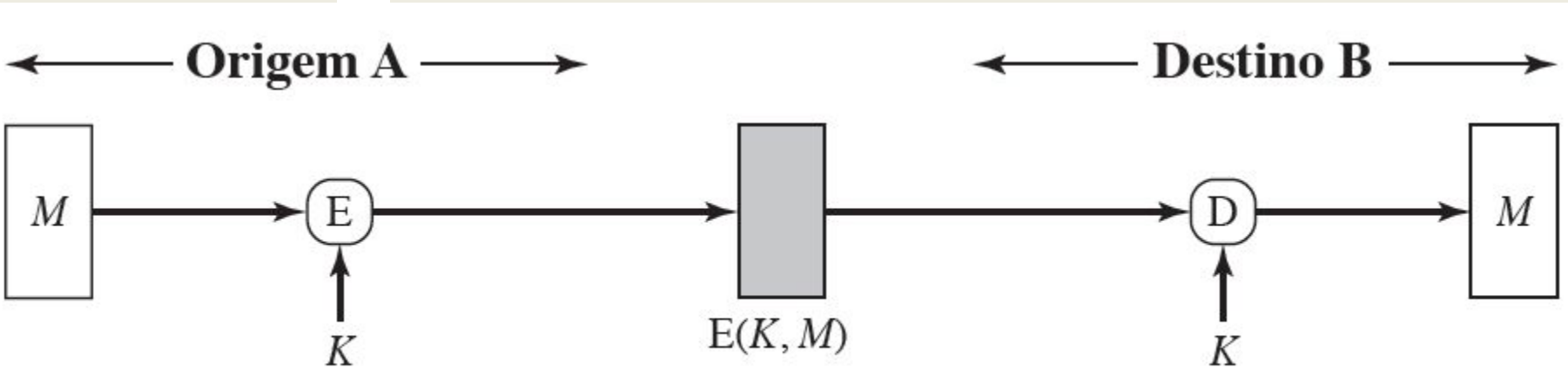
1. Divulgação
2. Análise de tráfego
3. Máscara
4. Modificação de conteúdo
5. Modificação de sequência
6. Modificação de tempo
7. Não reconhecimento na origem
8. Não reconhecimento no destino

Ataque	Descrição Resumida
Divulgação	Interceptação de dados por terceiros sem alterar o tráfego.
Análise de Tráfego	Observação de padrões e volume de tráfego, mesmo que criptografado.
Máscara (Masquerade)	Alguém se passa por outro usuário para obter acesso não autorizado.
Modificação de Conteúdo	Alteração do conteúdo de mensagens durante a transmissão.
Modificação de Sequência	Reordenação de pacotes para interferir no funcionamento da comunicação.
Modificação de Tempo	Atraso ou adiantamento deliberado na entrega de pacotes.
Não Reconhecimento na Origem	O remetente nega ter enviado a mensagem (repúdio).
Não Reconhecimento no Destino	O receptor nega ter recebido a mensagem (repúdio).

Funções de autenticação de mensagem

- A encriptação de mensagem por si só pode oferecer uma medida de autenticação.
- Uma mensagem *M* transmitida da origem A para o destino B é encriptada usando uma chave secreta *K* compartilhada por A e B.
- Se nenhuma outra parte souber a chave, então a confidencialidade é fornecida: nenhuma outra parte pode recuperar o texto claro da mensagem.

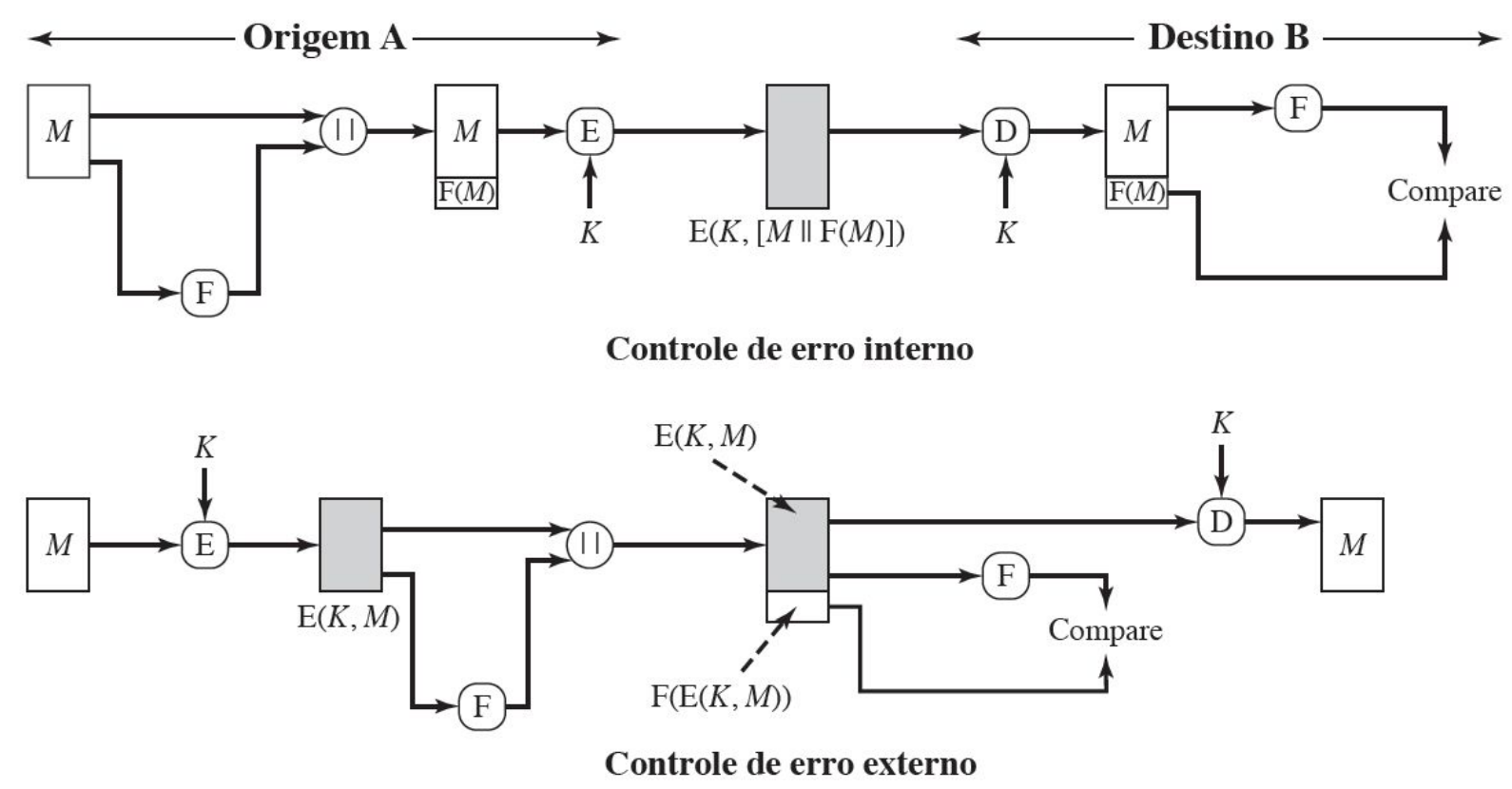
Funções de autenticação de mensagem



Encriptação simétrica: confidencialidade e autenticação

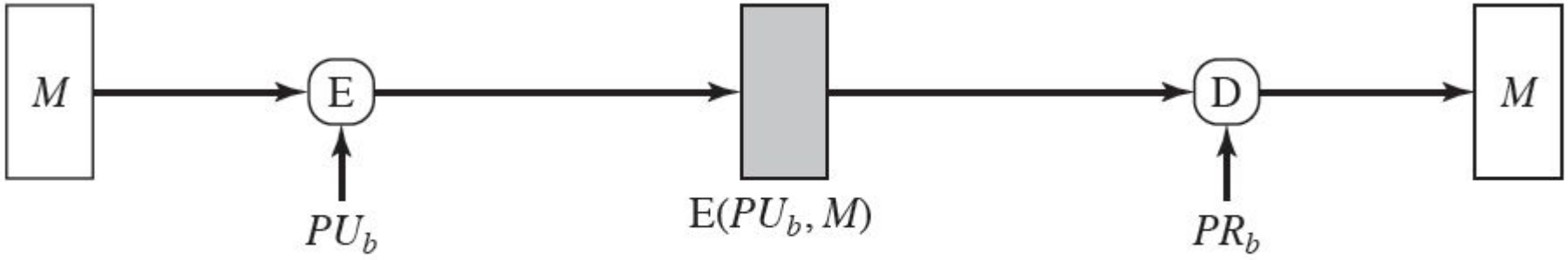
Funções de autenticação de mensagem

- Controle de erro interno e externo:



Funções de autenticação de mensagem

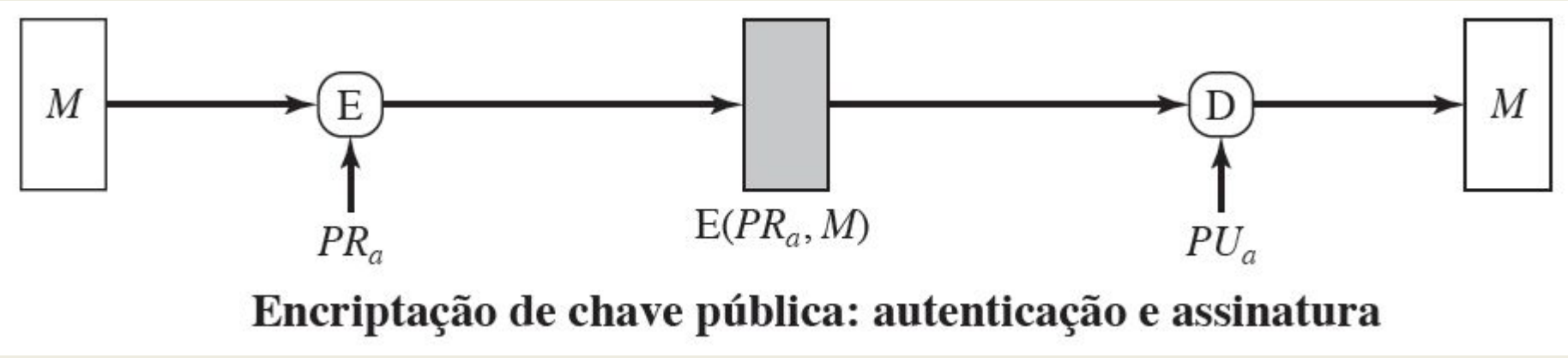
- O uso direto da encriptação de chave pública (figura abaixo) oferece confidencialidade, mas não autenticação:



Encriptação de chave pública: confidencialidade

Funções de autenticação de mensagem

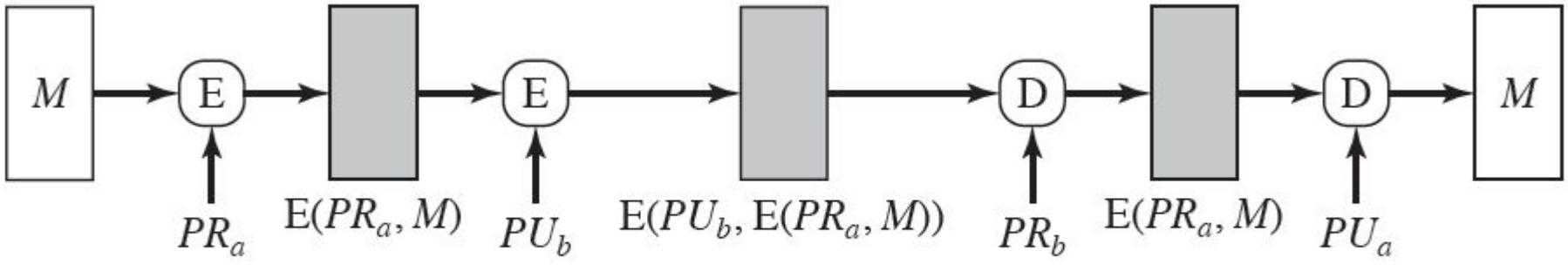
- Para oferecer autenticação, A utiliza sua chave privada para encriptar a mensagem, e B usa a chave pública de A para decriptar:



Encriptação de chave pública: autenticação e assinatura

Funções de autenticação de mensagem

- Para oferecer confidencialidade e autenticação, A pode encriptar M primeiro usando sua chave privada, que oferece a assinatura digital, e depois usando a chave pública de B, que oferece confidencialidade:



Encriptação de chave pública: confidencialidade, autenticação e assinatura

Requisitos para códigos de autenticação de mensagem

- Na avaliação da segurança de uma função MAC (Message Authentication Code), precisamos considerar os tipos de ataques que podem ser montados contra ela.
- Com isso em mente, vamos declarar os requisitos para a função.
- Suponha que um oponente saiba a função MAC, mas não saiba *K*.
- Então, a função MAC deverá satisfazer os seguintes requisitos:
  - **Forjamento impossível:** Não deve ser possível criar um MAC válido sem conhecer a chave secreta.
  - **Resistência a mensagens escolhidas:** Mesmo escolhendo mensagens e vendo seus MACs, não se deve conseguir forjar novos pares válidos.
  - **Resistência a colisões:** Não deve ser possível encontrar duas mensagens diferentes com o mesmo MAC.
  - **Imprevisibilidade:** O MAC deve parecer aleatório sem a chave, e pequenas mudanças na mensagem devem gerar MACs totalmente diferentes.

Requisitos para códigos de autenticação de mensagem

- Se um oponente observar M e MAC(K, M), deverá ser computacionalmente inviável para ele construir uma mensagem M' tal que

$$\text{MAC}(K, M') = \text{MAC}(K, M).$$

- MAC(K, M) deve ser distribuído uniformemente no sentido de que, para mensagens escolhidas de forma aleatória, M e M', a probabilidade de que MAC(K, M) = MAC(K, M') será 2^{-n}, onde n é o número de bits no tag.

Requisitos para códigos de autenticação de mensagem

- Considere que *M* seja igual a alguma transformação conhecida sobre *M*.
- Ou seja, $M' = f(M)$.
- Por exemplo, f pode envolver a inversão de um ou mais bits específicos.
- Nesse caso,

$$\Pr[\text{MAC}(K, M) = \text{MAC}(K, M')] = 2^{-n}$$

Segurança de MACs

Ataques por força bruta

- O nível de esforço para o ataque por força bruta sobre um algoritmo MAC pode ser expresso como $\min(2^k, 2^n)$.
- A avaliação da força é semelhante à dos algoritmos de encriptação simétrica.
- Pareceria razoável exigir que o tamanho da chave e o tamanho MAC satisfaçam um relacionamento como $\min(k, n) \geq N$, onde N talvez esteja no intervalo de 128 bits.

MACs baseados em funções de hash: HMAC

RFC 2104 lista os seguintes objetivos de projeto para o HMAC:

- Usar, sem modificações, as funções de hash disponíveis.
- Permitir a substituição fácil da função de hash embutida caso sejam encontradas ou exigidas funções de hash mais rápidas ou mais seguras.
- Preservar o desempenho original da função de hash sem incorrer em uma degradação significativa.

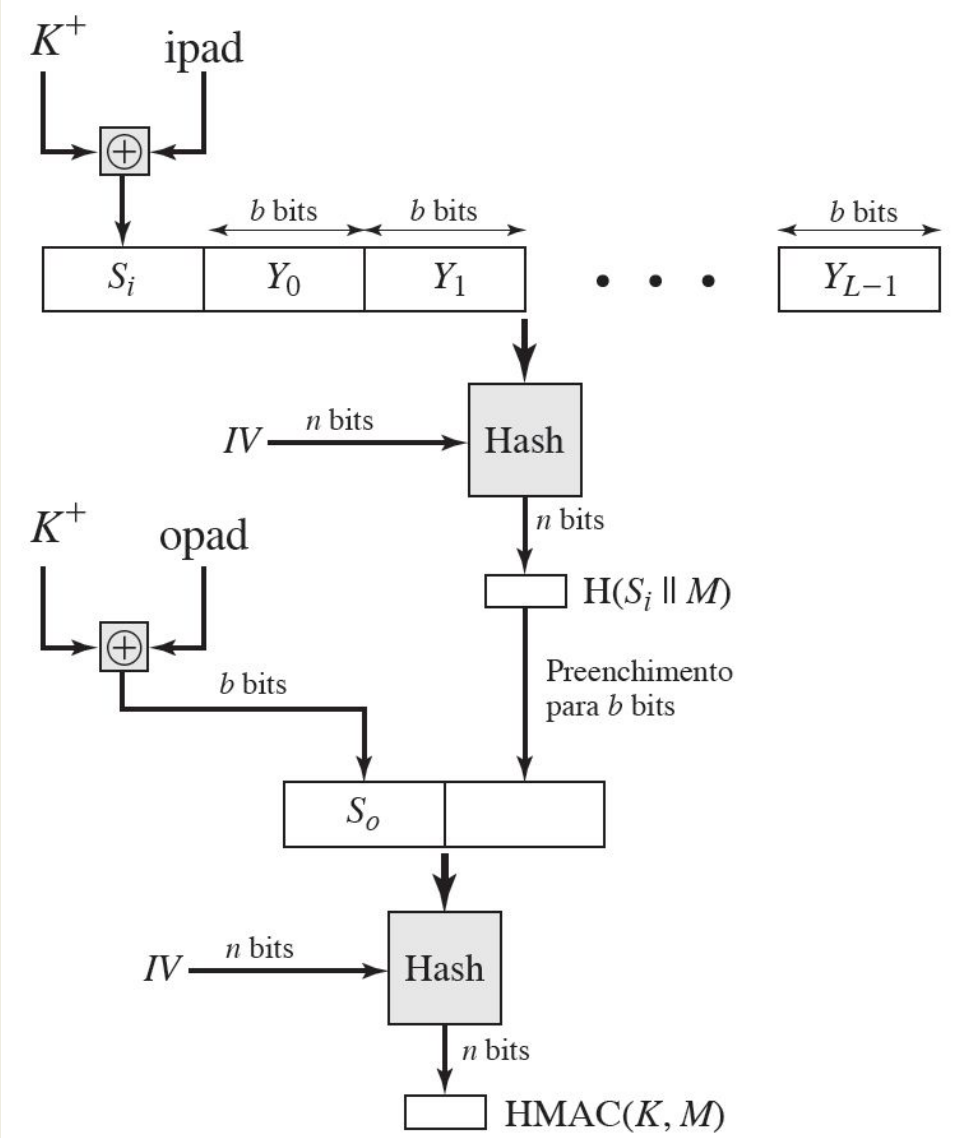
MACs baseados em funções de hash: HMAC

RFC 2104 lista os seguintes objetivos de projeto para o HMAC:

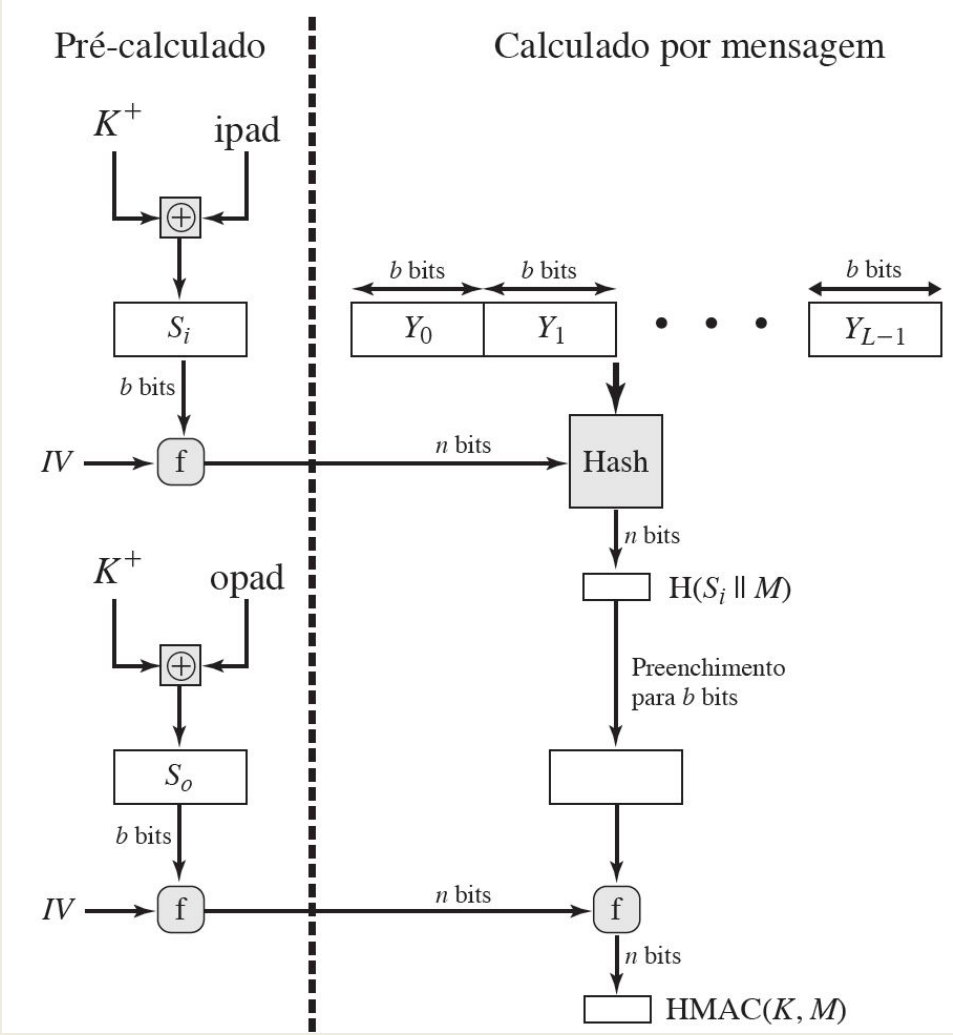
- Usar e tratar das chaves de uma forma simples.
- Ter uma análise criptográfica bem compreendida da força do mecanismo de autenticação com base em suposições razoáveis sobre a função de hash embutida.

A figura a seguir ilustra a operação geral do HMAC.

MACs baseados em funções de hash: HMAC



MACs baseados em funções de hash: HMAC



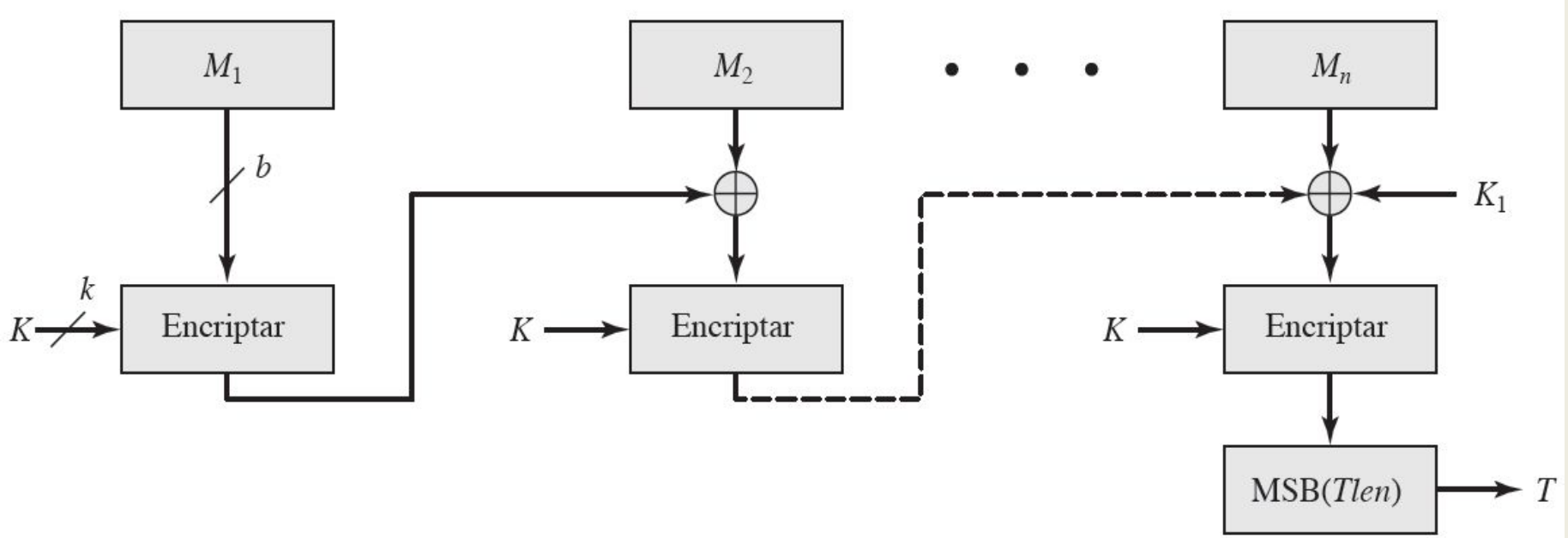
Implementação eficiente do HMAC:

MACs baseados em cifras de bloco: DAA e CMAC

- O **Data Authentication Algorithm** (**DAA**), baseado no DES, foi um dos MACs mais utilizados por muitos anos.
- O algoritmo é uma publicação do FIPS (FIPS PUB 113) e um padrão ANSI (X9.17).
- Porém, foram descobertas deficiências na segurança e ele está sendo substituído por algoritmos mais novos e fortes.
- O algoritmo pode ser definido como usando o modo de operação cipher block chaining (CBC) do DES com um vetor de inicialização de zeros.

MACs baseados em cifras de bloco: DAA e CMAC

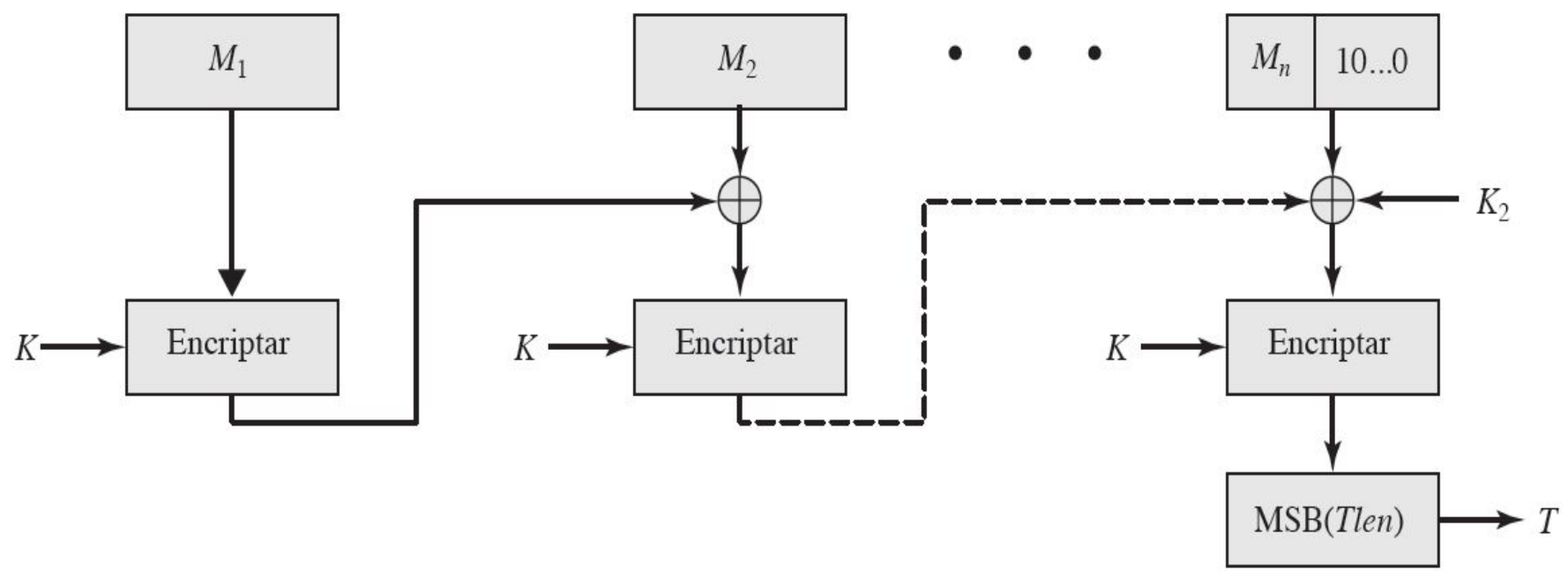
- Cipher-Based Message Authentication Code (CMAC):



Tamanho da mensagem é múltiplo inteiro do tamanho do bloco

MACs baseados em cifras de bloco: DAA e CMAC

- Cipher-Based Message Authentication Code (CMAC):



Tamanho da mensagem não é múltiplo inteiro do tamanho do bloco

MACs baseados em cifras de bloco: DAA e CMAC

- Cipher-Based Message Authentication Code (CMAC):

Gerar subchaves K_1 e K_2 a partir da chave secreta K usando AES.

- São usadas para tratar o último bloco da mensagem (com ou sem padding).

Dividir a mensagem M em blocos de tamanho fixo (por exemplo, 128 bits para AES).

- Se o último bloco for incompleto → aplica padding (como em CBC-MAC).
- Se for completo → não precisa de padding.

XOR no último bloco com K_1 ou K_2, dependendo se teve padding ou não.

Processar os blocos com AES em modo CBC, usando um vetor de inicialização (IV) igual a zero.

O último bloco cifrado é o CMAC.

MACs baseados em cifras de bloco: DAA e CMAC

- Cipher-Based Message Authentication Code (CMAC):

Característica	CMAC	HMAC
Baseado em	Cifras (ex: AES)	Hash (ex: SHA-256)
Tamanho da chave	Tamanho da cifra (ex: 128 bits para AES)	Qualquer tamanho razoável
Desempenho	Melhor em hardware	Mais comum em software
Aplicações típicas	Sistemas embutidos, criptografia em hardware	Web, APIs, software geral

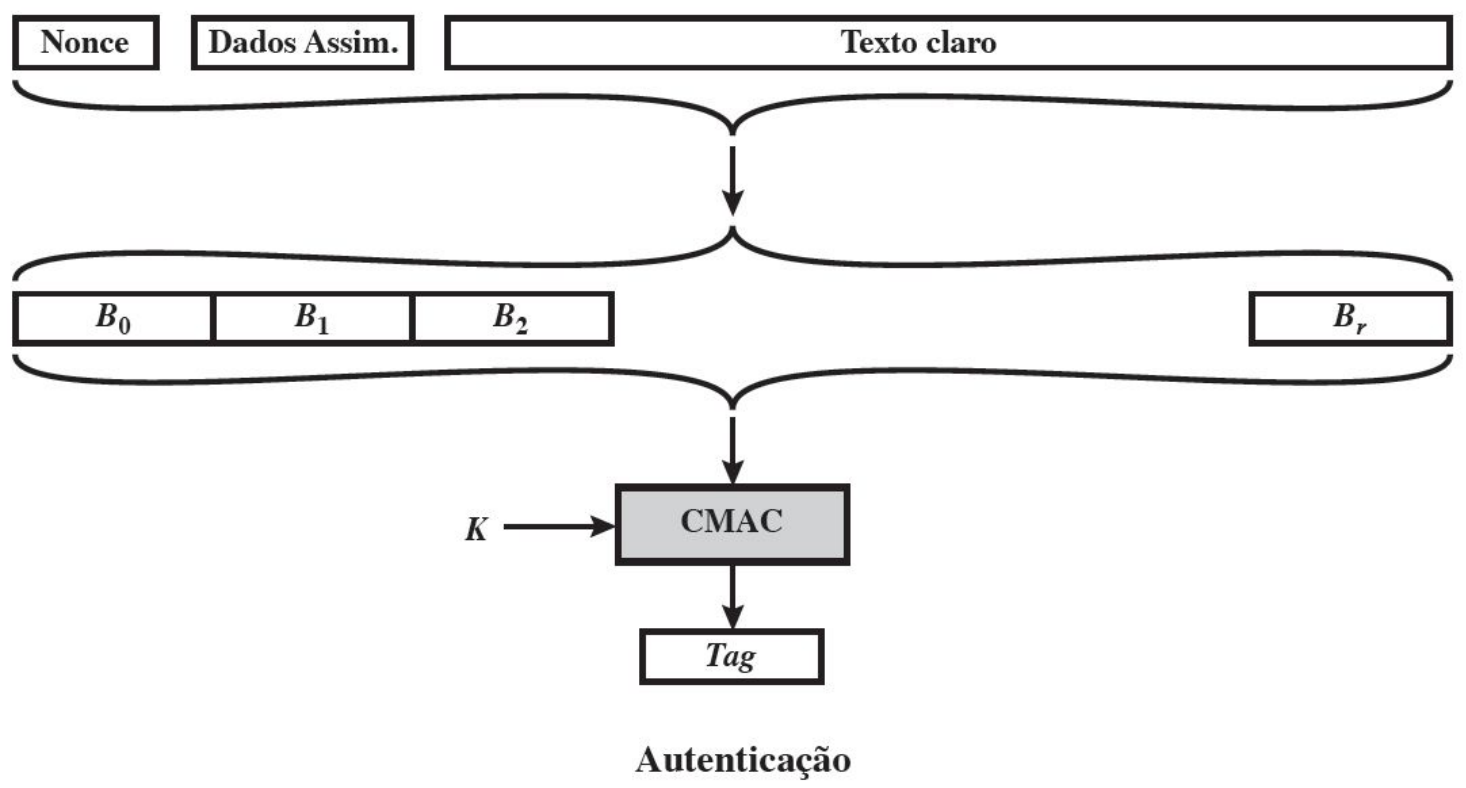
Encriptação autenticada: CCM e GCM

Existem quatro técnicas comuns para fornecer confidencialidade e encriptação para uma mensagem *M*.

- Hashing seguido por encriptação (H → E)
- Autenticação seguida por encriptação (A → E)
- Encriptação seguida por autenticação (E → A)
- Encriptação e autenticação independentes (E + A)

Encriptação autenticada: CCM e GCM

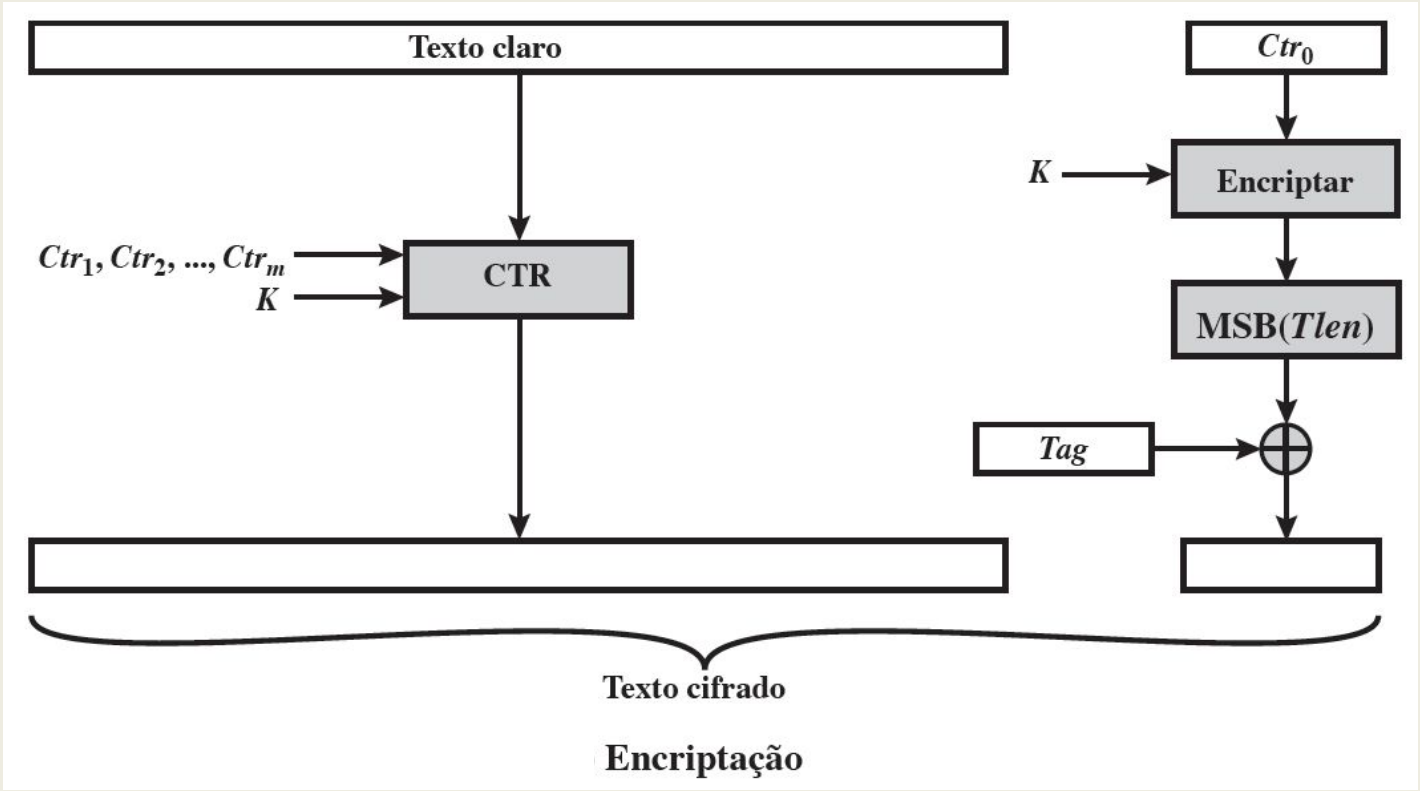
- Contador com Cipher Block Chaining-Message authentication Code (CCM):



Autenticação

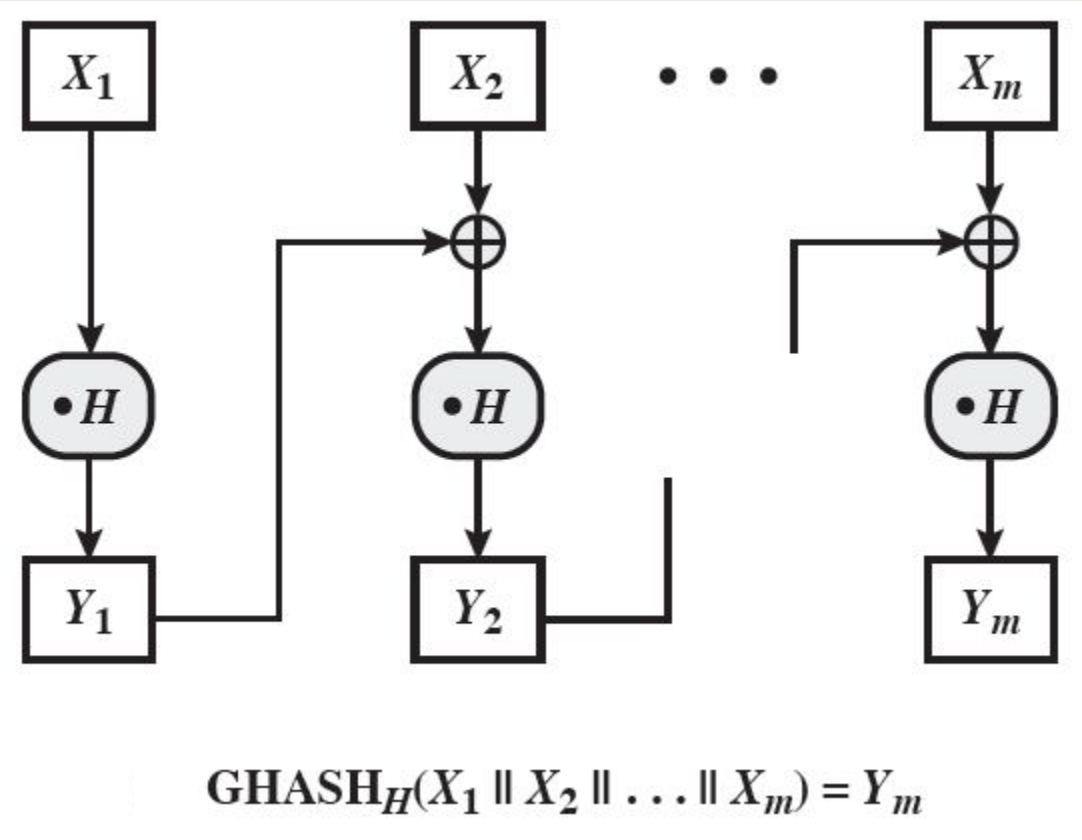
Encriptação autenticada: CCM e GCM

- Contador com Cipher Block Chaining-Message authentication Code (CCM):



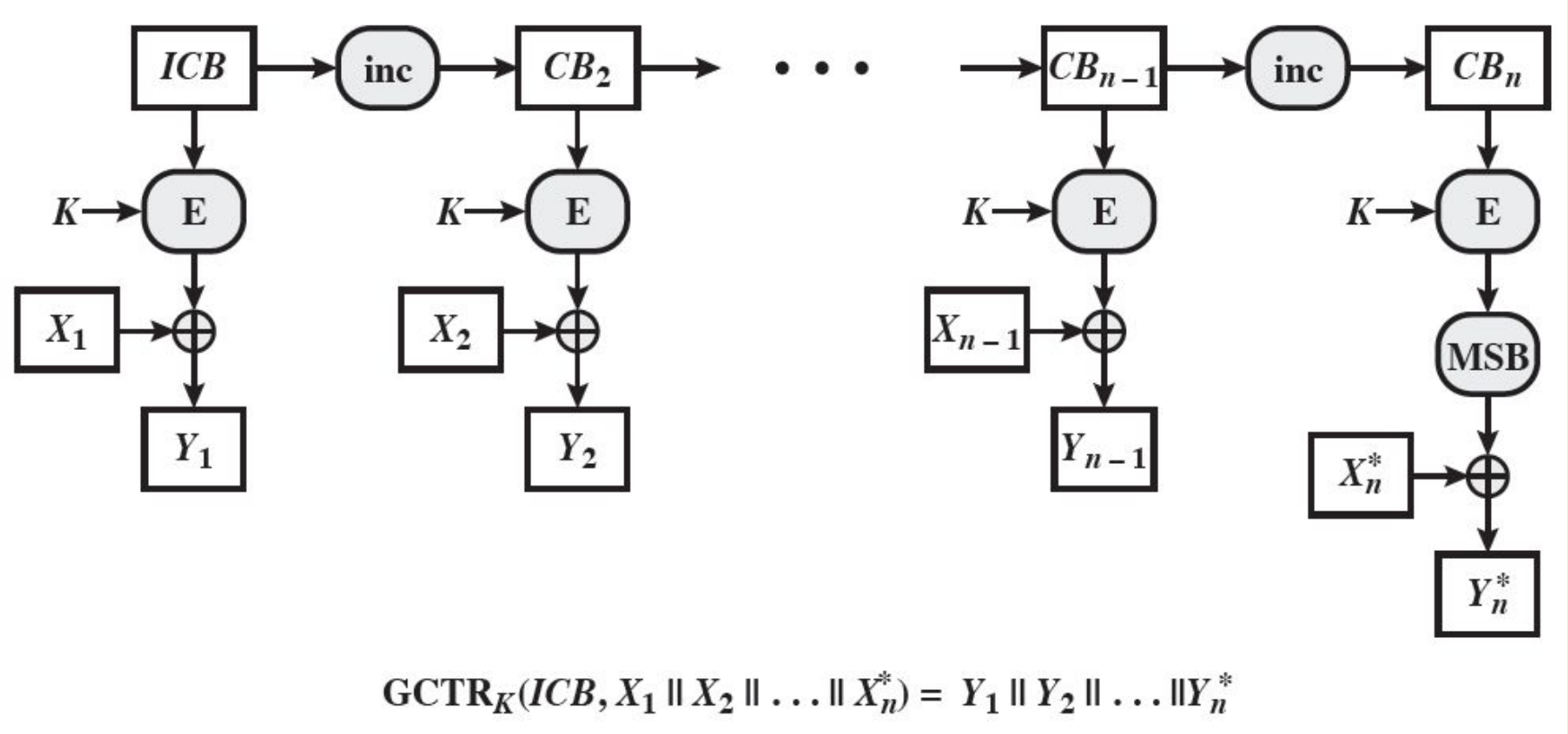
Encriptação autenticada: CCM e GCM

- Funções de autenticação e encriptação GCM:



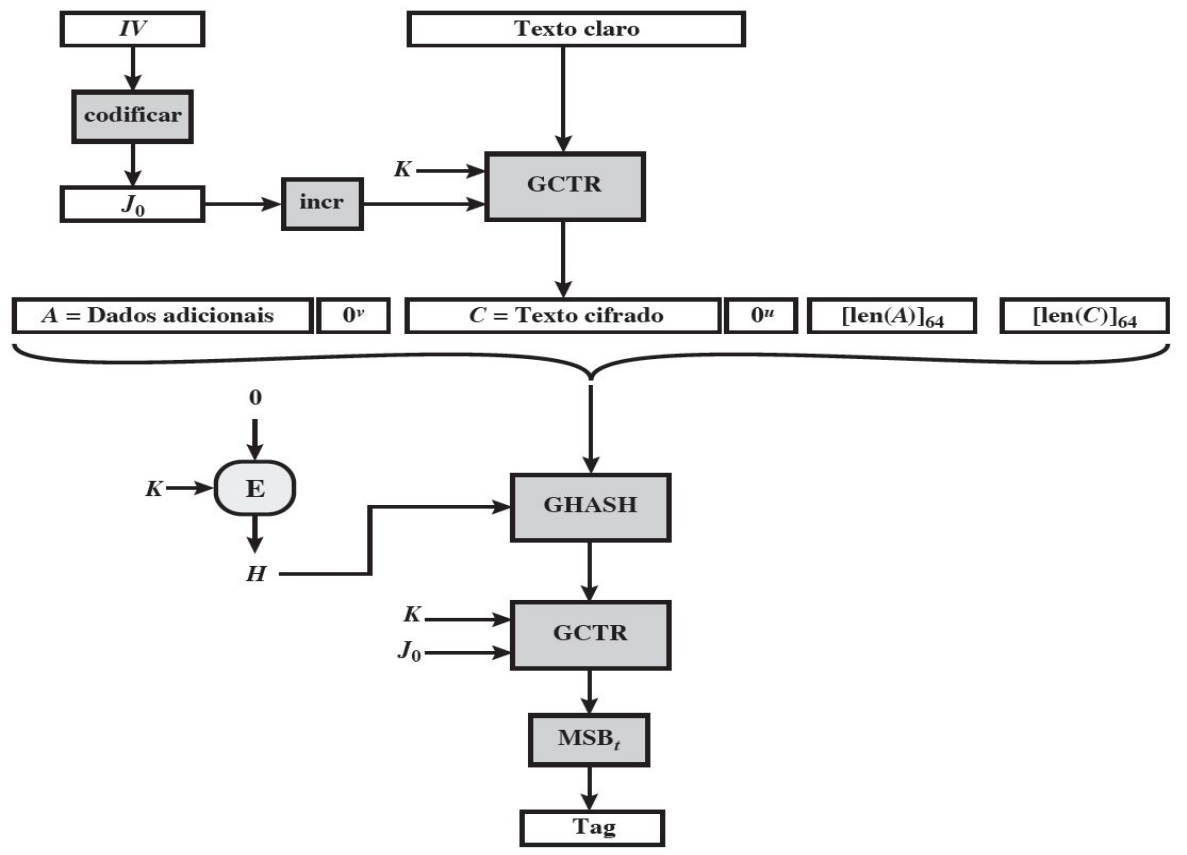
Encriptação autenticada: CCM e GCM

- Funções de autenticação e encriptação GCM:



Encriptação autenticada: CCM e GCM

- Código de autenticação Galois Counter-Message (GCM):



Key Wrapping RFC 3394

- A finalidade do **key wrapping** é trocar com segurança uma chave simétrica a ser compartilhada por duas partes, usando uma chave simétrica já compartilhada por elas.
- A segunda chave é denominada **chave de encriptação de chave** (**KEK**).
- O algoritmo key wrapping opera sobre blocos de 64 bits.
- Entradas: Texto claro, *n* valores de 64 bits (P_1, P_2, ... , P_n)
- Chave de encriptação de chave, *K*

Key Wrapping

- Saídas: Texto cifrado, (n + 1) valores de 64 bits (C_0, C_1, ... , C_n)

1. **Inicializar variáveis.**

$A(0) = \text{A6A6A6A6A6A6A6A6}$

```
for i = 1 to n
  R(0, i) = P_i
```

2. **Calcular valores intermediários.**

for $t = 1$ to s

```
W = E(K, [A(t-1) || R(t-1, 1)])
A(t) = t ⊕ MSB_64(W)
R(t, n) = LSB_64(W)
for i = 1 to n-1
  R(t, i) = R(t-1, i+1)
```

3. **Gerar resultados.**

```
C_0 = A(s)
for i = 1 to n
  C_i = R(s, i)
```

Key Unwrapping

O algoritmo key unwrapping pode ser definido da seguinte forma:

- Entradas: Texto cifrado, (n + 1) valores de 64 bits (C_0, C_1, . . . , C_n)
- Chave de encriptação de chave, K
- Saídas: Texto claro, n valores de 64 bits (P_1, P_2, . . . , P_n), ICV

Key Unwrapping

1. **Inicializar variáveis.**

   $A(s) = C_0$

   for $i = 1$ **to** n

   $R(s, i) = C_i$

2. **Calcular valores intermediários.**

   for $t = s$ **to** 1

   $W = D(K, [(A(t) \oplus t) \| R(t, n)])$

   $A(t-1) = MSB_{64}(W)$

   $R(t-1, 1) = LSB_{64}(W)$

   for $i = 2$ **to** n

   $R(t-1, i) = R(t, i-1)$

3. **Gerar resultados.**

   if $A(0) =$ A6A6A6A6A6A6A6A6

   then

   for $i = 1$ **to** n

   $P(i) = R(0, i)$

   else

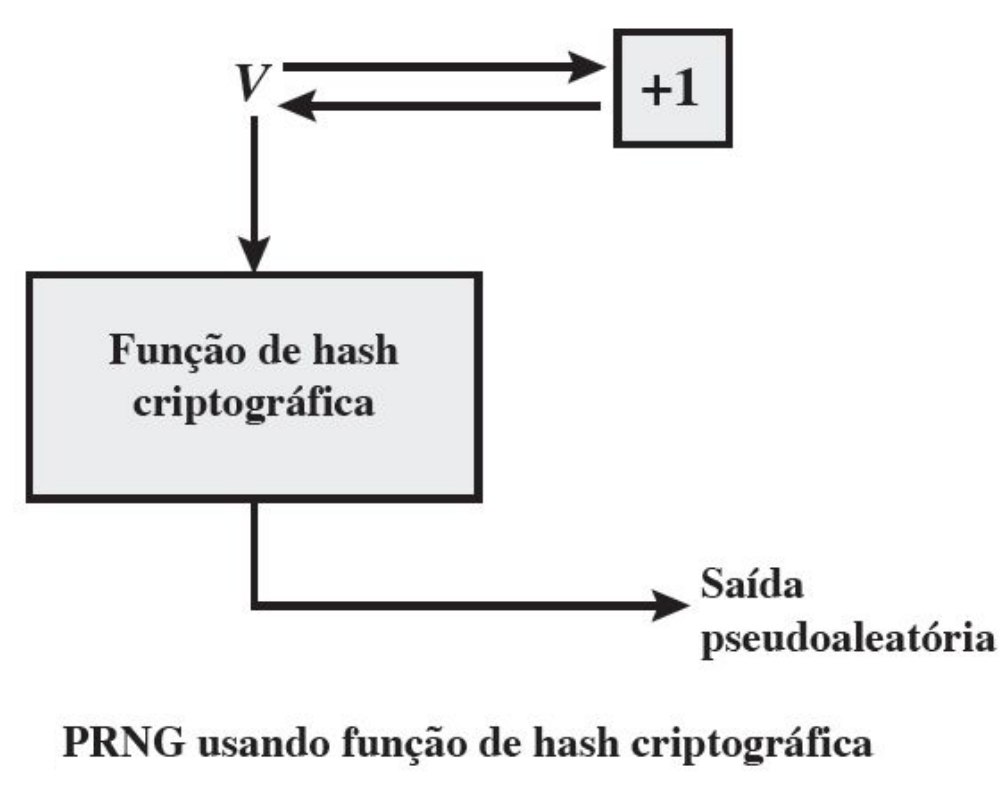
   return error

Geração de número pseudoaleatório usando funções de hash e MACs

- Uma função de hash ou MAC produz saída aparentemente aleatória e pode ser usada para criar um PRNG.
- Tanto o padrão ISO 18031 (Random Bit Generation) quanto o NIST SP 800-90 (Recommendation for Random Number Generation Using Deterministic Random Bit Generators) definem um método para geração de número aleatório usando uma função de hash criptográfica.
- SP 800-90 também define um gerador de número aleatório baseado em HMAC.

Geração de número pseudoaleatório usando funções de hash e MACs

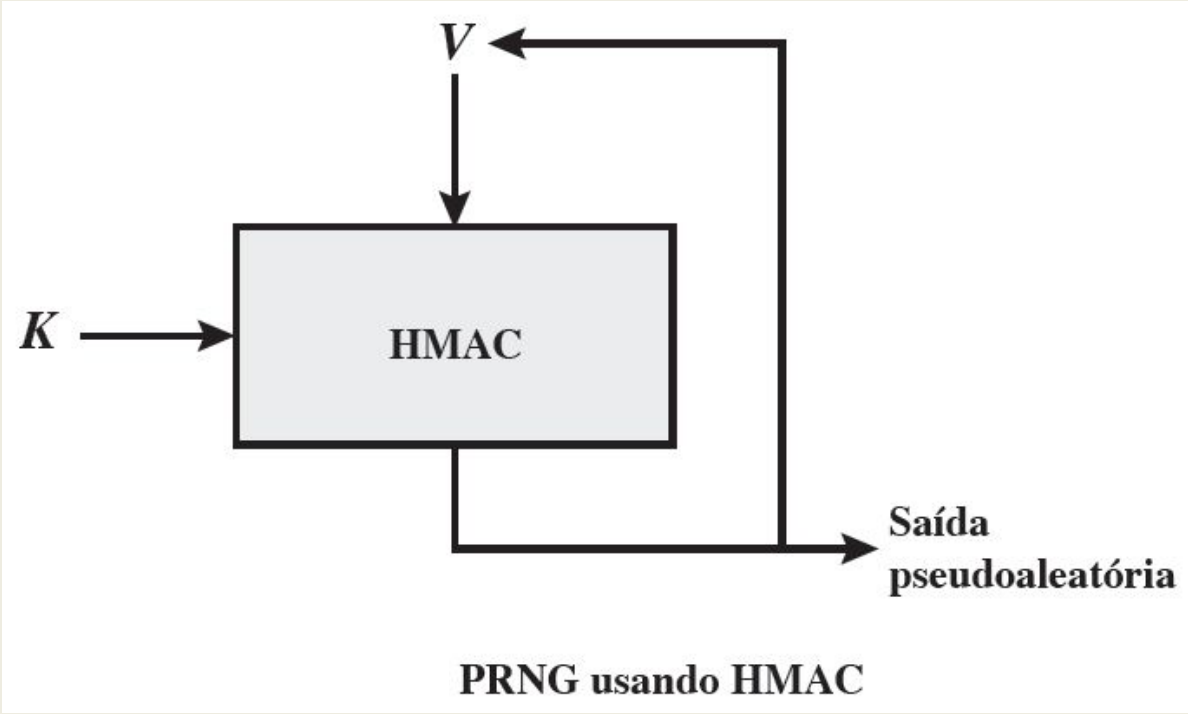
- A figura abaixo mostra a estratégia básica para um PRNG baseado em hash, especificado no SP 800-90 e no ISO 18031:



PRNG usando função de hash criptográfica

Geração de número pseudoaleatório usando funções de hash e MACs

- A figura abaixo mostra a estrutura básica do mecanismo PRNG, e a coluna mais à esquerda da figura a seguir mostra a lógica:



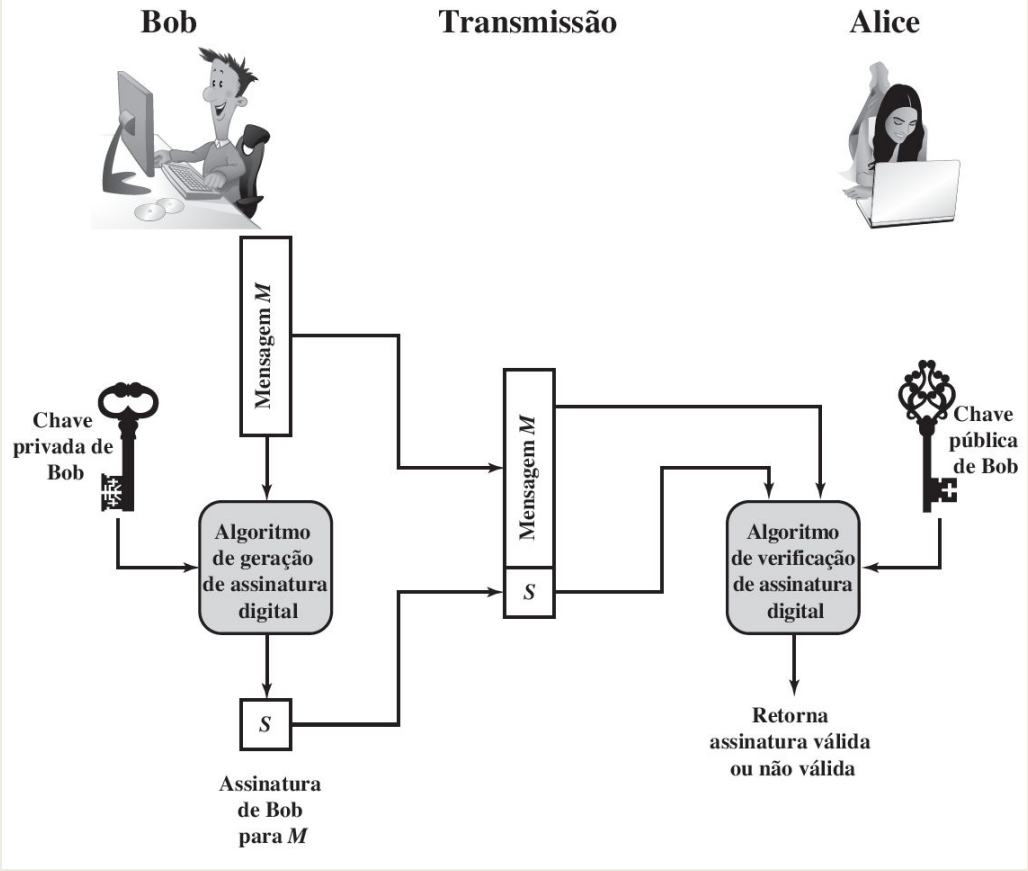
PRNG usando HMAC

Geração de número pseudoaleatório usando funções de hash e MACs

NIST SP 800-90	IEEE 802.11i	TLS/WTLS
$m = \lceil n/\text{outlen} \rceil$ $w_0 = V$ $W =$ a string nula For $i = 1$ to m $w_i = \text{MAC}(K, w_{i-1})$ $W = W \| w_i$ Retorna n bits mais à esquerda de W	$m = \lceil n/\text{outlen} \rceil$ $W =$ a nula string For $i = 1$ to m $w_i = \text{MAC}(K, (V \| i))$ $W = W \| w_i$ Retorna n bits mais à esquerda de W	$m = \lceil n/\text{outlen} \rceil$ $A(0) = V$ $W =$ a string nula For $i = 1$ to m $A(i) = \text{MAC}(K, A(i-1))$ $w_i = \text{MAC}(K, (A(i) \| V)$ $W = W \| w_i$ Retorna n bits mais à esquerda de W

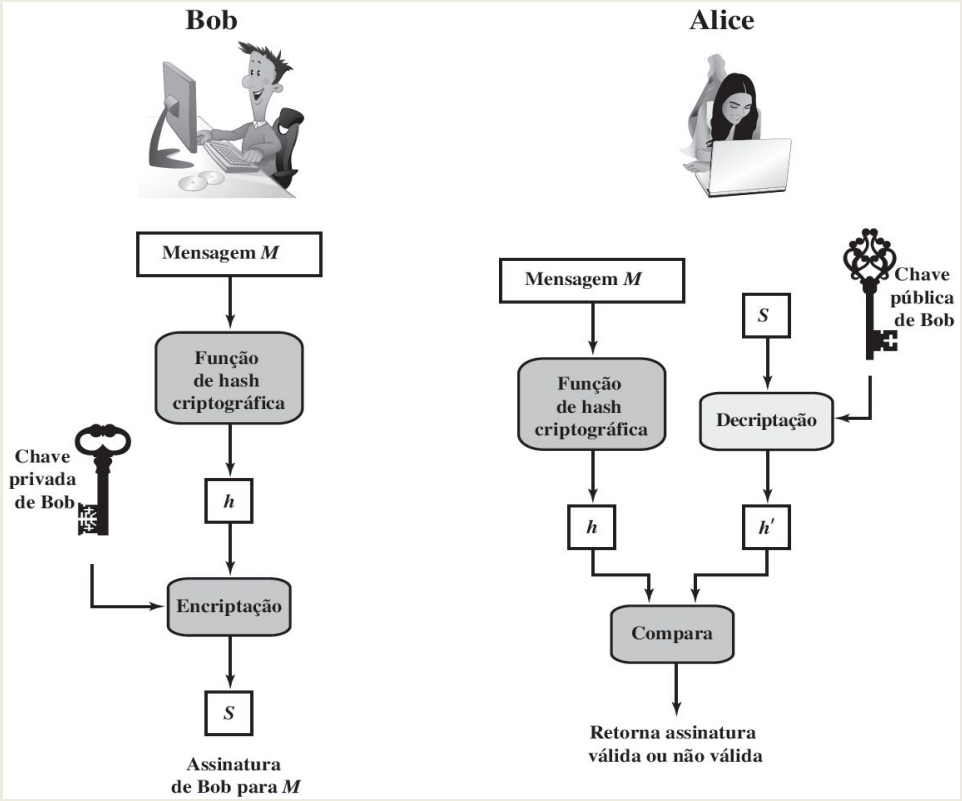
Assinaturas digitais

- Modelo genérico do processo de assinatura digital:



Assinaturas digitais

- Representação simplificada dos elementos essenciais do processo de assinatura digital:



Ataques e falsificações

Ataques, em ordem crescente de severidade:

- Ataque somente de chave
- Ataque de mensagem conhecida
- Ataque de mensagem escolhida genérica
- Ataque de mensagem escolhida direcionada
- Ataque de mensagem escolhida adaptativa

Ataques e falsificações

[GOLD88] define então o sucesso na quebra de um esquema de assinatura como resultado em que C pode fazer qualquer um dos seguintes com uma probabilidade não insignificante:

- Quebra total
- Falsificação universal
- Falsificação seletiva
- Falsificação existencial

Requisitos de assinatura digital

Com base nessas propriedades e ataques discutidos, podemos formular os seguintes requisitos para uma assinatura digital:

- A assinatura precisa ser um padrão de bits que depende da mensagem sendo assinada.
- A assinatura precisa usar alguma informação exclusiva do emissor, para impedir falsificação e negação.
- É preciso ser relativamente fácil produzir a assinatura digital.

Requisitos de assinatura digital

- É preciso ser relativamente fácil reconhecer e verificar a assinatura digital.
- É preciso ser computacionalmente inviável falsificar uma assinatura digital, seja construindo uma nova mensagem para uma assinatura digital existente ou uma assinatura digital fraudulenta para determinada mensagem.
- É preciso ser prático reter uma cópia da assinatura digital em termos de armazenamento.

Esquema de assinatura digital Elgamal

- O esquema de assinatura Elgamal envolve o uso da chave privada para encriptação e a chave pública para decriptação.
- Os elementos globais da assinatura digital Elgamal são um número primo q e a, que é uma raiz primitiva de q.
- O usuário A gera um par de chaves pública/privada.
- Para assinar uma mensagem M, o usuário A primeiro calcula o hash $m = H(M)$, tal que m seja um inteiro na faixa $0 \leq m \leq q - 1$.

Esquema de assinatura digital Elgamal

- Suponha que o usuário A deseje assinar uma mensagem m.
- **Parâmetros públicos e chaves:**
  - Escolhe-se um número primo grande p.
  - Escolhe-se um gerador g (raiz primitiva módulo p).
  - Usuário A escolhe uma chave privada $x \in \{1, ..., p-2\}$.
  - Calcula a chave pública: $y = g^x \bmod p$.
- **Para assinar a mensagem m:**
  - Escolhe-se um valor aleatório k tal que $1 < k < p-1$ e $\gcd(k, p-1) = 1$.
  - Calcula:
    - $r = g^k \bmod p$
    - $s = k^{-1}(H(m) - x * r) \bmod (p-1)$,
    - onde H(m) é o hash da mensagem,
    - e k^{-1} é o inverso multiplicativo de k mod (p-1).
- **A assinatura da mensagem é o par (r, s).**

Esquema de assinatura digital Elgamal

- Qualquer usuário B pode verificar a assinatura.
- A assinatura é válida se $V_1 = V_2$.
- Seja a assinatura composta pelo par (r, s), e:
  - p: um primo grande
  - g: uma raiz primitiva módulo p
  - y: chave pública do remetente, onde y = g^x mod p
  - x: chave privada do remetente
  - H(m): valor de hash da mensagem m
- A assinatura é válida se:
  - V1 = y^r * r^s mod p
  - V2 = g^H(m) mod p

ELGamal vs RSA

Aleatoriedade em ElGamal

- Cada vez que você assina, mesmo a mesma mensagem terá uma assinatura diferente por causa do valor aleatório k.
- No RSA básico, a assinatura é sempre igual para a mesma mensagem (o que pode ser um risco de segurança se não for combinada com **hash + padding** seguro).

Base matemática

- ElGamal depende do **logaritmo discreto** (como o algoritmo Diffie-Hellman).
- RSA depende da dificuldade de fatorar grandes números compostos.

Segurança estrutural

- ElGamal é naturalmente semelhante a esquemas d**e chave efêmera**, o que dificulta certos ataques baseados em padrões.
- RSA precisa de técnicas adicionais (**como padding OAEP ou PSS**) para garantir segurança contra ataques.

Algoritmo de assinatura digital do NIST

- O DSA utiliza um algoritmo que é projetado para oferecer apenas a função de assinatura digital.
- Diferente do RSA, ele não pode ser usado para encriptação ou troca de chave.
- Apesar disso, essa é uma técnica de chave pública.
- A técnica do DSA também usa uma função de hash.
- A função de assinatura é tal que somente o emissor, com conhecimento da chave privada, poderia ter produzido a assinatura válida.

Algoritmo de assinatura digital do NIST

- O algoritmo de assinatura digital (DSA):

Componentes globais da chave pública

p número primo entre $2^{L-1} < p < 2^L$ para $512 \le L \le 1024$ e L um múltiplo de 64; ou seja, o tamanho entre 512 e 1024 bits em incrementos de 64 bits

q divisor primo de $(p-1)$, onde $2^{N-1} < q < 2^N$; ou seja, tamanho de N bits

g $= h(p-1)/q \bmod p$, onde h é qualquer inteiro em $1 < h < (p-1)$, tal que $h^{(p-1)/q} \bmod p > 1$

Chave privada do usuário

x inteiro aleatório ou pseudoaleatório com $0 < x < q$

Chave pública do usuário

$y = g^x \bmod p$

Número secreto por mensagem do usuário

k inteiro aleatório ou pseudoaleatório com $0 < k < q$

Assinatura

$r = (g^k \bmod p) \bmod q$

$s = [k^{-1}(H(M) + xr)] \bmod q$

Assinatura = (r, s)

Verificação

$w = (s')^{-1} \bmod q$

$u_1 = [H(M')w] \bmod q$

$u_2 = (r')w \bmod q$

$v = [(g^{u1}\, y^{u2}) \bmod p] \bmod q$

TESTE: $v = r'$

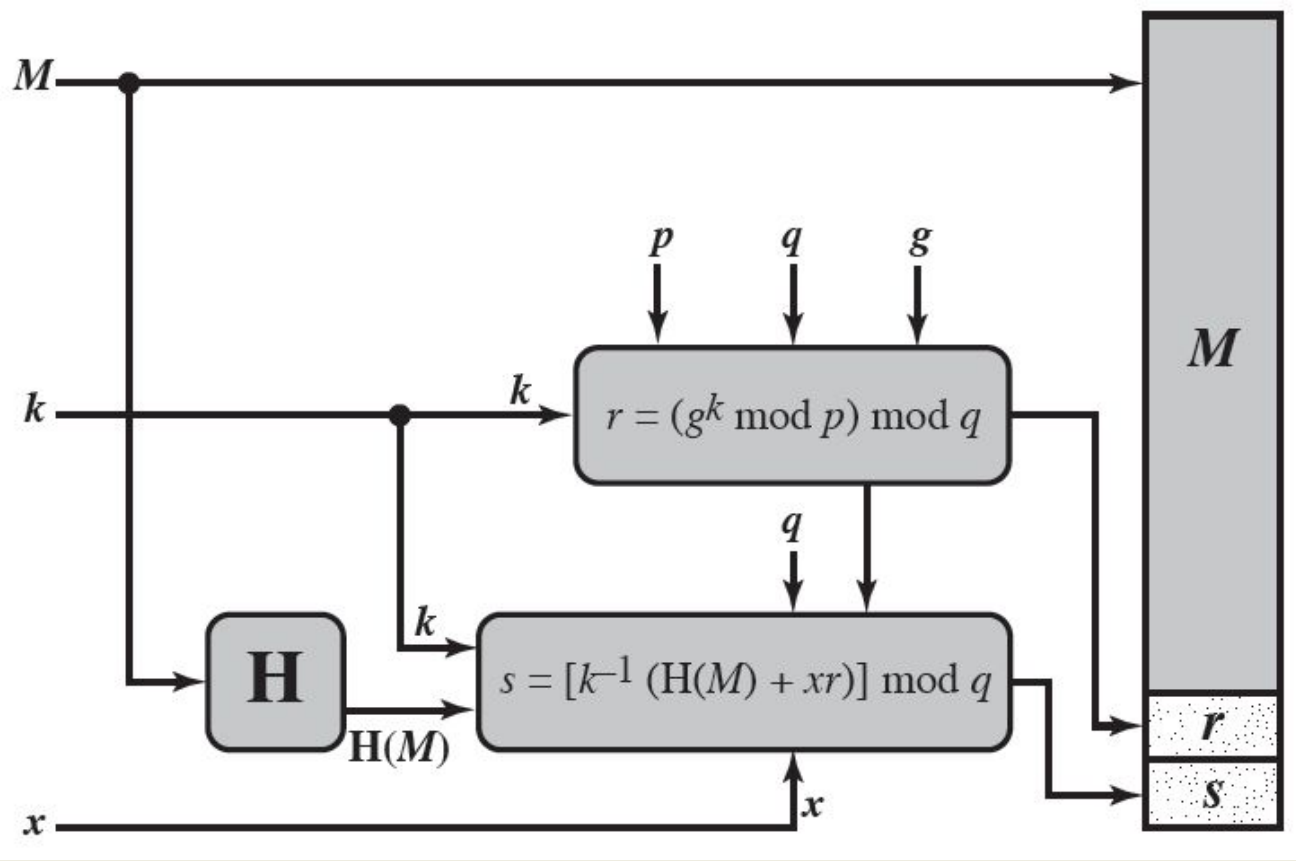
M = mensagem a ser assinada

H(M) = hash de M usando SHA-1

M', r', s' = versões recebidas de M, r, s

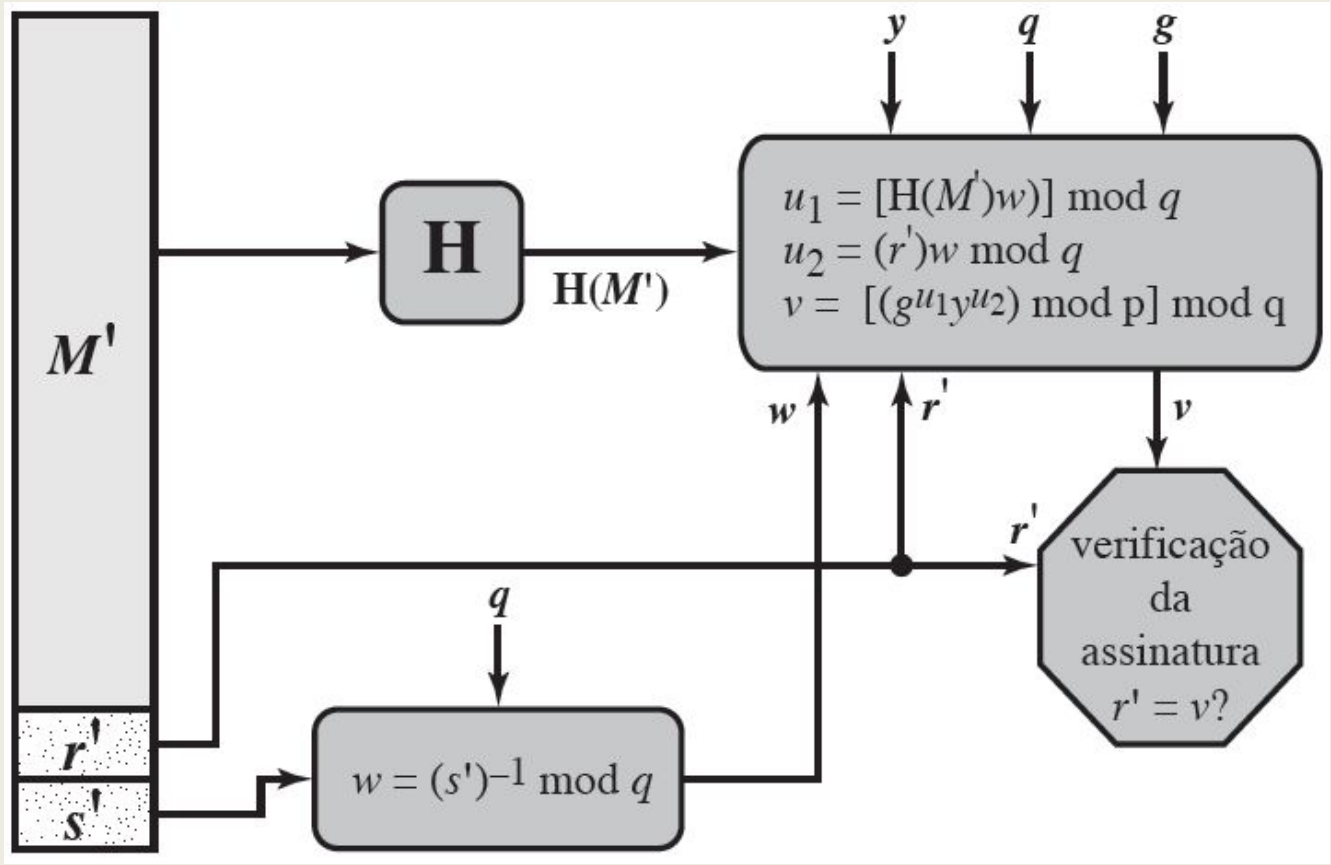
Algoritmo de assinatura digital do NIST

- Assinatura do DSA:



Algoritmo de assinatura digital do NIST

- Verificação do DSA:



Algoritmo de assinatura digital de curva elíptica

Segue uma breve visão geral do processo envolvido no ECDSA (Elliptic Curve Digital Signature Algorithm):

- Todos aqueles que participam do esquema de assinatura digital usam os mesmos parâmetros de domínio global.
- Um assinante precisa primeiro gerar um par de chaves: **pública e privada.**
- Um valor de **hash é gerado** para a mensagem ser assinada.

Algoritmo de assinatura digital de curva elíptica

- Para verificar a assinatura, o verificador usa como entrada a chave pública do assinante, os parâmetros do domínio e o inteiro *s*. A saída é um valor *v* que é comparado com *r*. A assinatura **é válida se *v* = *r*.**

Os parâmetros de domínio global para ECDSA são os seguintes:

- *q* um número primo
- *a*, *b* inteiros que especificam a equação de curva elíptica definida sobre Z_q com a equação $y^2 = x^3 + ax + b$

Resumo

Característica	RSA	RSA-PSS	ECC (ECDSA)	ElGamal	DSA (NIST)
Tipo de algoritmo	Assinatura e criptografia	Assinatura digital	Assinatura digital (sobre ECC)	Criptografia e assinatura	Assinatura digital
Base matemática	Fatoração	Fatoração + padding probabilístico	Logaritmo discreto sobre curvas	Logaritmo discreto	Logaritmo discreto (modular)
Chave pública/privada	(n,e)/d	(n,e)/d	(G,P)/k	(p,g,y)/x	(p,q,g,y)/x
Assinatura	S=H(M)^d mod n	S=RSA^−1(EM) com codificação	(r,s)	(r,s)	(r,s)
Verificação da assinatura	M′=S^e mod n	Decodifica EM, compara hashes	Verifica com operações em curva	Verifica modularmente	Verifica modularmente
Determinismo/Aleatoriedade	Determinístico	Aleatório (usa salt)	Aleatório (usa nonce k)	Aleatório (usa nonce k)	Aleatório (usa nonce k)
Segurança baseada em	Fatoração	Fatoração com resistência CCA	Logaritmo elíptico	Logaritmo discreto	Logaritmo discreto
Tamanho da chave (exemplo)	2048 bits	2048 bits	256 bits (mesma segurança que RSA 3072)	2048 bits	2048 bits
Tamanho da assinatura	Igual ao módulo (ex: 256 bytes)	Igual ao módulo	Pequena (ex: 64 bytes para P-256)	2x tamanho de q	2x tamanho de q
Eficiência (assinatura/verificação)	Assinatura rápida, verificação lenta	Assinatura e verificação seguras	Assinatura lenta, verificação rápida	Assinatura lenta	Assinatura lenta, verificação rápida
Padrões relacionados	PKCS#1	PKCS#1 v2.2 / RFC 8017	NIST FIPS 186-4 / SECG	Teórico / raramente usado	NIST FIPS 186-4
Vulnerabilidades conhecidas	Fraca contra (Chosen Ciphertext Attack) e má implementação	Muito mais segura que o RSA Tradicional	Quebra com reuse ou má escolha de k	Quebra com reuse ou má escolha de k	Quebra se k for reutilizado
Uso moderno	Amplamente usado em TLS e certificados	Assinaturas digitais seguras	Criptomoedas (Bitcoin), sistemas móveis	Pouco usado	Documentos governamentais (EUA)

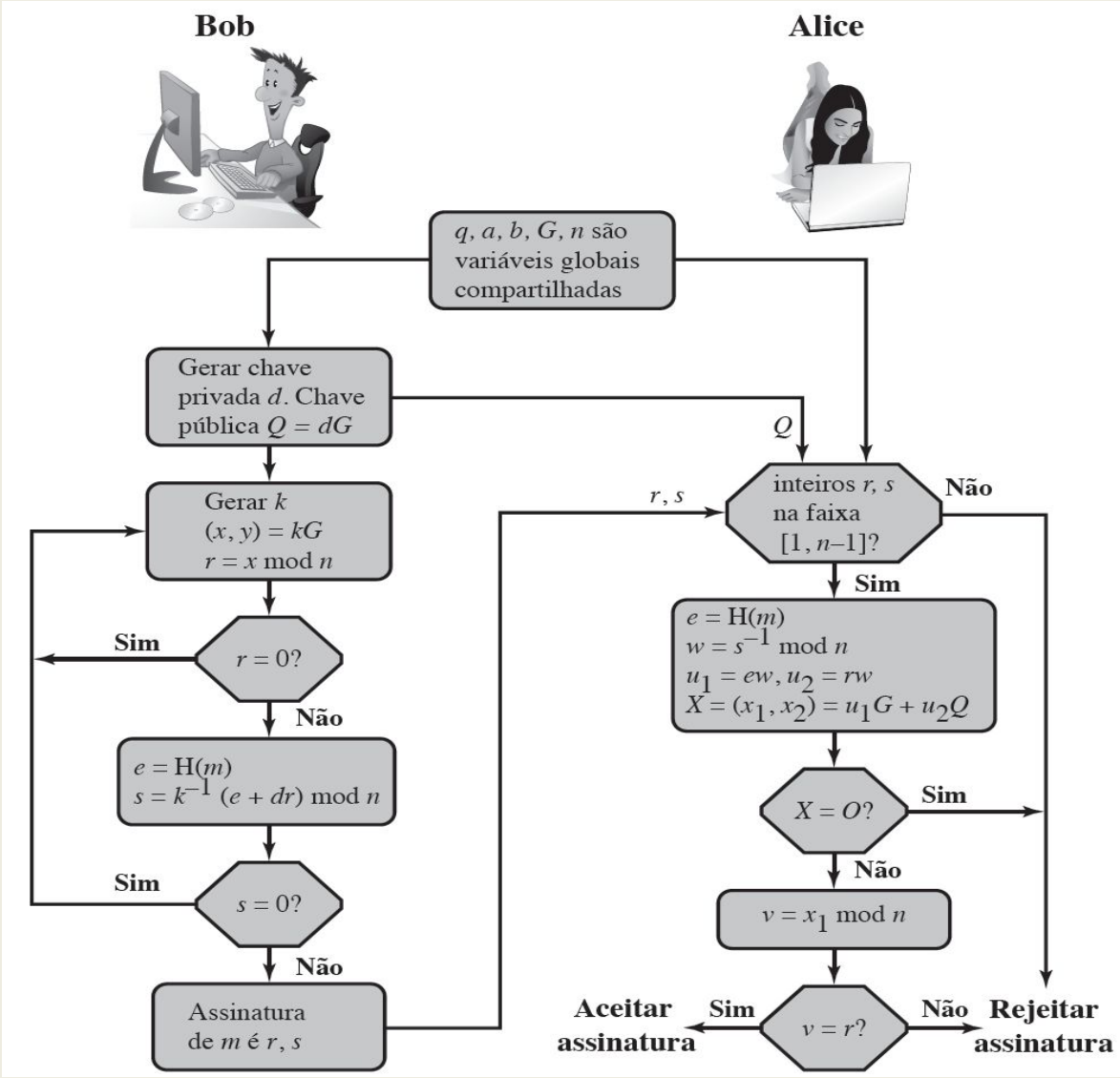
Algoritmo de assinatura digital de curva elíptica

- G um ponto de base representado por $G = (x_g, y_g)$ sobre a equação da curva elíptica
- n ordem do ponto G; ou seja, n é o menor inteiro positivo tal que $nG = O$. Este também é o número de pontos na curva.
- Cada assinante precisa gerar um par de chaves, uma privada e uma pública.
- Com os parâmetros de domínio público e uma chave privada em mãos, Bob gera uma assinatura digital de 320 bytes para a mensagem m.

Algoritmo de assinatura digital de curva elíptica

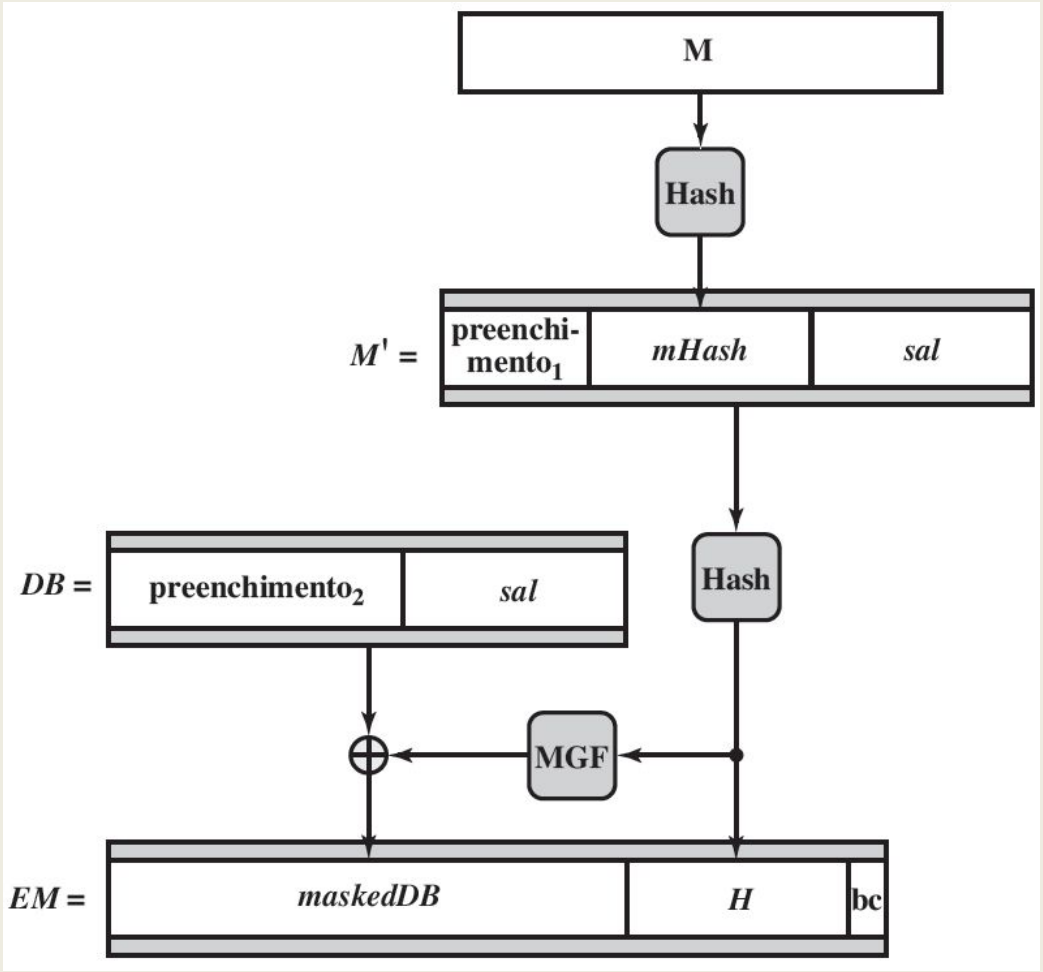
- Alice conhece os parâmetros de domínio público e a chave pública de Bob.
- Alice recebe a mensagem e a assinatura digital de Bob e a verifica.
- A figura a seguir ilustra o processo de autenticação de assinatura.

Algoritmo de assinatura digital de curva elíptica



Algoritmo de assinatura digital RSA-PSS

O primeiro estágio na geração de uma assinatura **RSA-PSS (*RSA-PSS Probabilistic Signature Scheme)*** de uma mensagem *M* é gerar a partir de *M* um resumo da mensagem de tamanho fixo, chamado de mensagem codificada (*EM – Encoded Message*):



Algoritmo de assinatura digital RSA-PSS

- A assinatura *s* é formada encriptando *m* da seguinte forma:

$$s = m^d \bmod n$$

- Para a verificação da assinatura, trate a assinatura *S* como um inteiro binário *s* sem sinal, não negativo.

- A figura a seguir ilustra o processo.

Algoritmo de assinatura digital RSA-PSS

- Verificação de EM no RSA-PSS: