Segurança da Informação

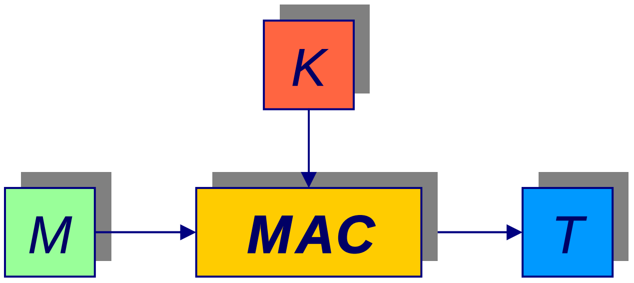
Códigos de Autenticação (MAC)
Números Aleatórios
Quotização de Segredo

Códigos de Autenticação

Códigos de Autenticação

- Redundâncias anexadas a mensagens com o propósito de detectar alterações e garantir a autenticidade do remetente.

- Dependem da mensagem e também de uma informação secreta, compartilhada entre o remetente e o destinatário.



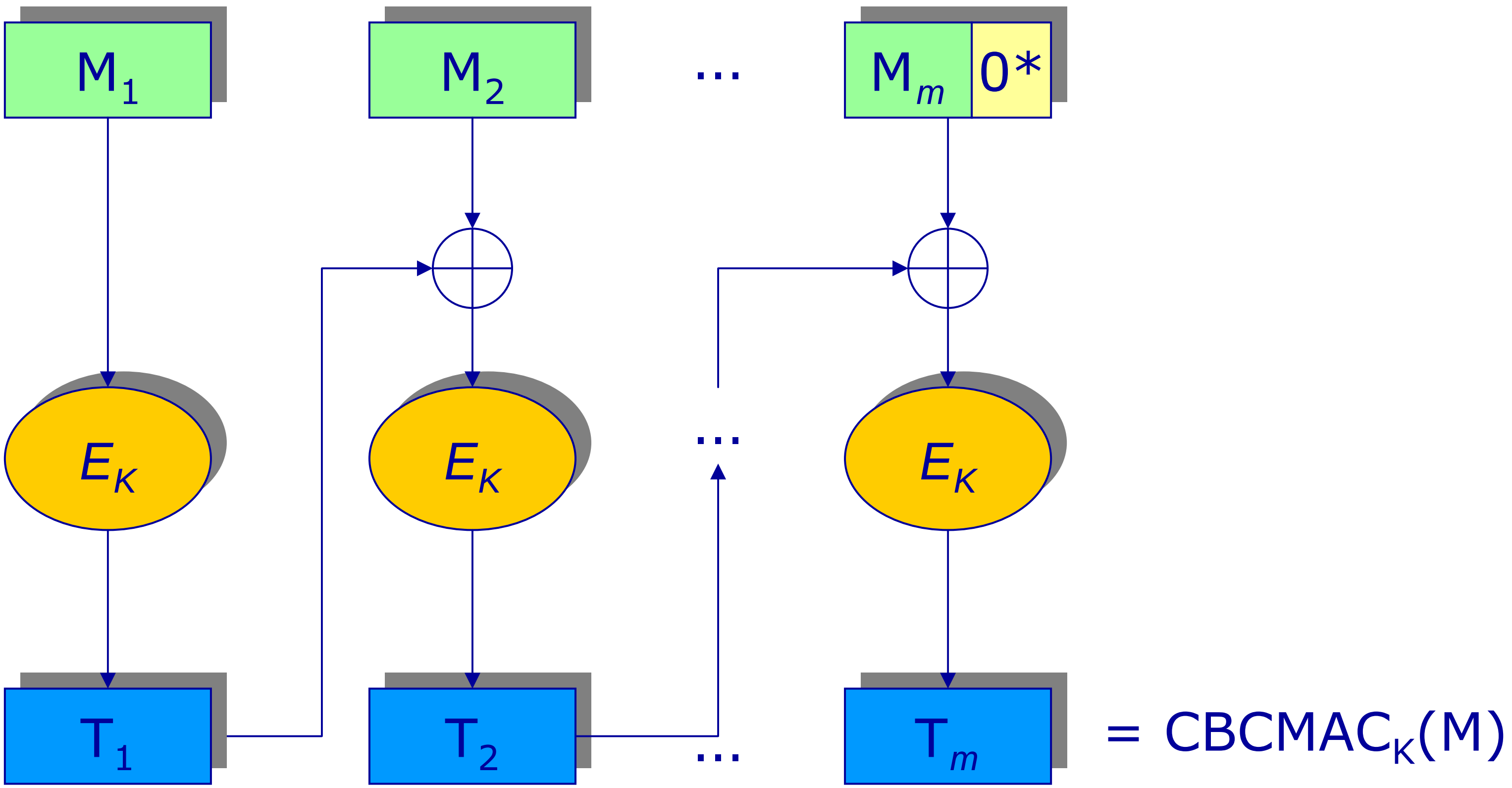
Assinaturas Digitais?

- Um código de autenticação visa a garantir:
  - Integridade.
  - Autenticidade.
- Não pode garantir *irretratabilidade*, pois tanto o remetente quanto o destinatário conhecem a mesma chave.
- Numa assinatura digital verdadeira, apenas o remetente conhece a chave de assinatura.

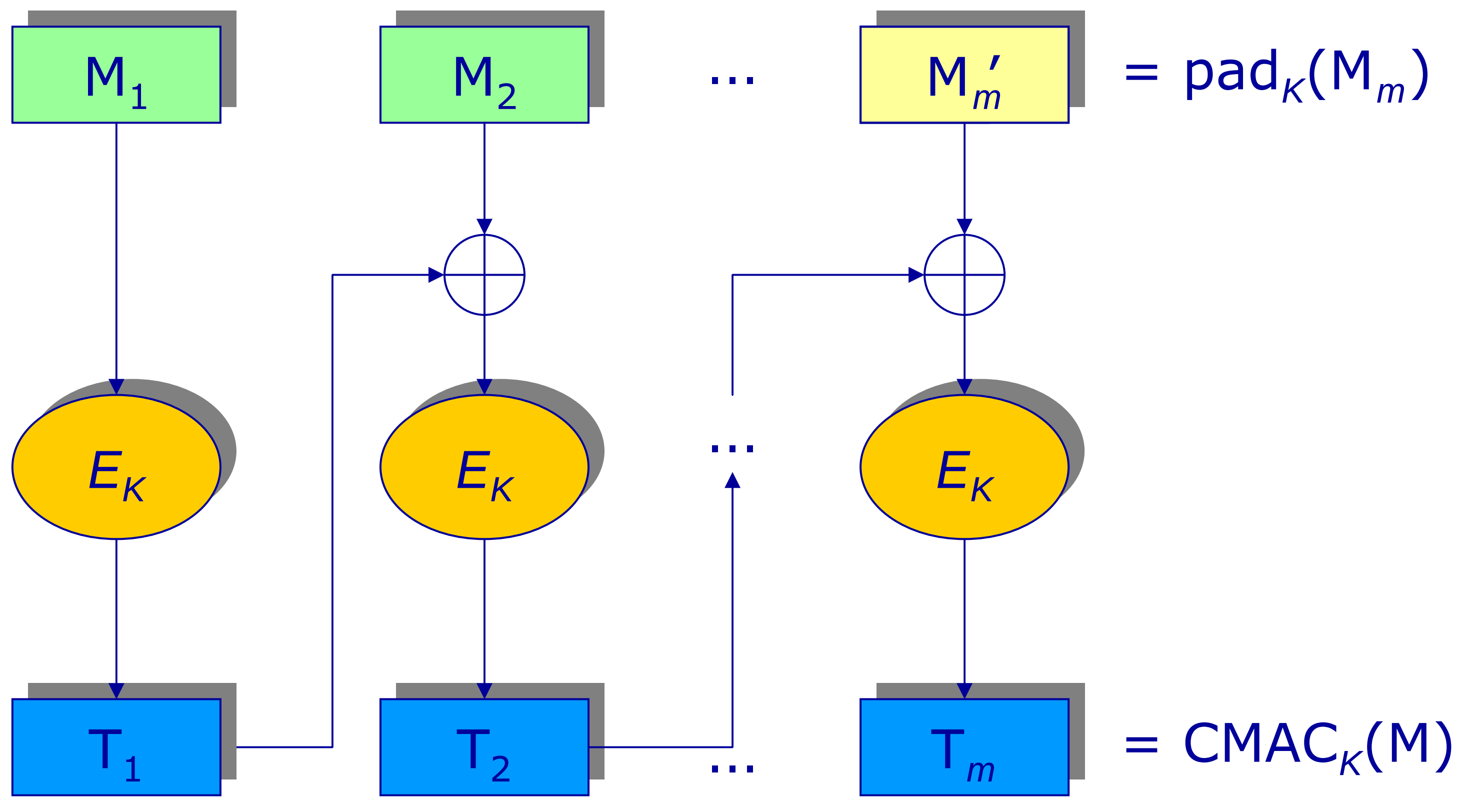
Construções Básicas

- Cifras de bloco:
  - CBCMAC (FIPS 113, ANSI X9.17).
  - CMAC (NIST SP 800-38B).
  - Vantagem: espaço reduzido de código (aproveitam implementações existentes de cifras de bloco).
- Funções de *hash*:
  - HMAC (FIPS 198).
  - Vantagem: velocidade de operação (funções de *hash* puras).

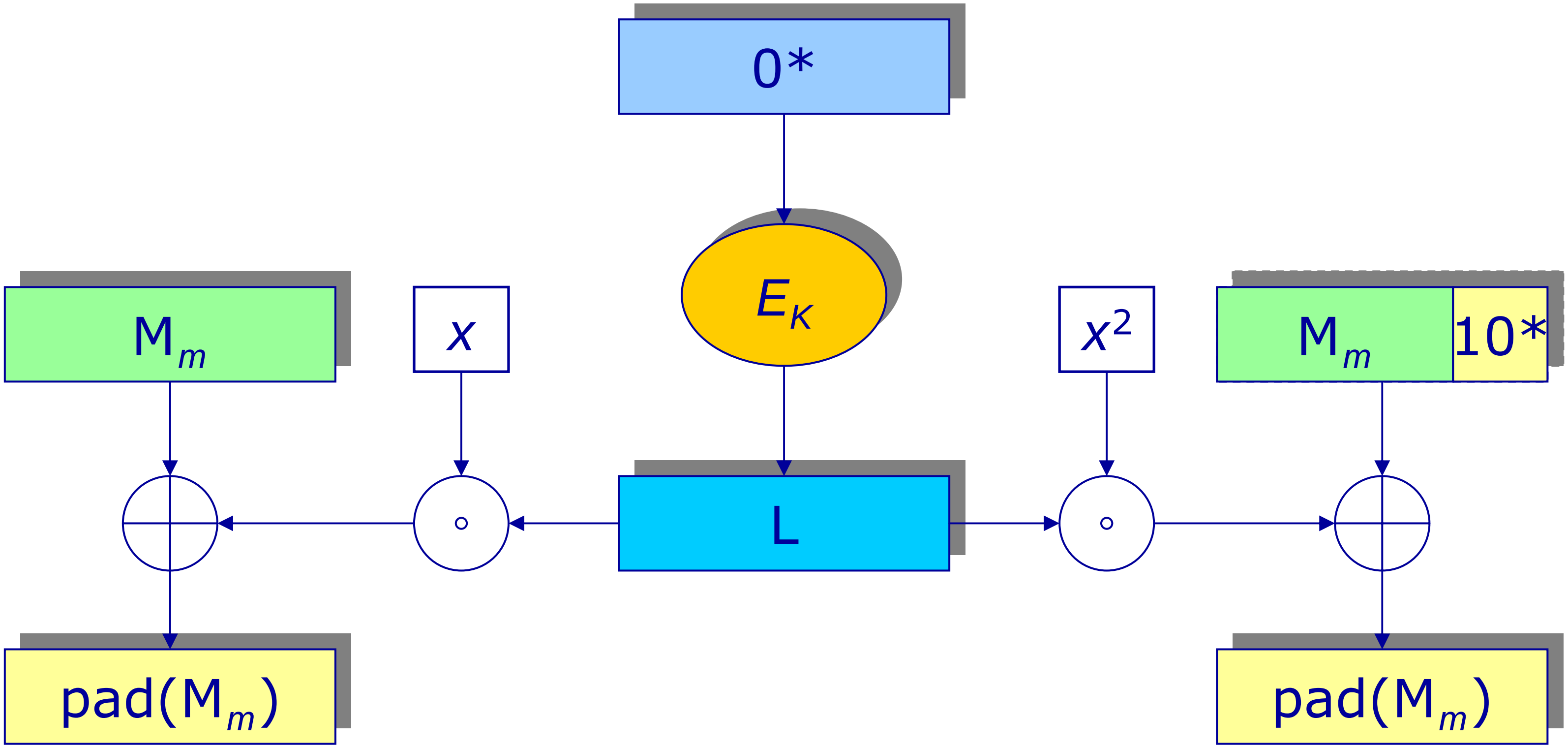
CBCMAC



CMAC



CMAC



CMAC

- CBCMAC só é seguro se aplicado exclusivamente a mensagens de um *mesmo tamanho*, previamente fixado para cada chave.
- CMAC é aplicável a mensagens de qualquer tamanho.
- Ambos são estritamente sequenciais.

Carter-Wegman

- Mensagem M_1 ... M_m interpretada como um polinômio em GF(2^n)[X]:

$$P_M(X) = \bigoplus_{k=1}^{m} M_k X^k,\ M_k \in \mathrm{GF}(2^n)$$

- Ideia básica: $\mathrm{CW}_K(M) = P_M(K)$ (caveat!)

- Variante GHASH: parte do modo híbrido GCM (NIST SP 800-38D).

Outros algoritmos de MAC

- PMAC:
  - Paralelizável.
  - Elementos do modo LRW.

- Família ALRED:
  - Processamento de cada bloco envolve apenas cifração parcial.
  - Natural para cifras da estratégia de trilha larga.
  - Cálculo sequencial; variante paralelizável.

Modos Híbridos

- Objetivo: proporcionar os serviços de confidencialidade, integridade e autenticidade.
- Composição genérica: combinação de um modo de confidencialidade e um código de autenticação sobre a mesma cifra de bloco. Exemplos: EAX, CCM (CTR + CMAC).
- Custo para cifrar n blocos: $2n + \varepsilon$ chamadas da cifra subjacente.

Modos Híbridos

- Modos integrados: código de autenticação computacionalmente mais leve (por bloco) que a cifra subjacente. Exemplos: GCM (CTR + GHASH), OCB, LetterSoup.
- Custo para cifrar *n* blocos: $(1 + \delta)n + \varepsilon$ chamadas da cifra; tipicamente $0.1 \leq \delta \leq 0.4$.

HMAC

- Utiliza funções de hash que particionam a mensagem em blocos (por exemplo, construção Merkle-Damgård).
- Tamanho da chave limitado apenas pelo tamanho dos blocos.
- Eficiente: apenas duas chamadas da função de hash, sendo a segunda sempre sobre um volume reduzido de dados (2 blocos).

HMAC

- Completar um bloco a partir da chave K com zeros binários à direita: $K \mid\mid 0^*$.

- Dois blocos iniciais:

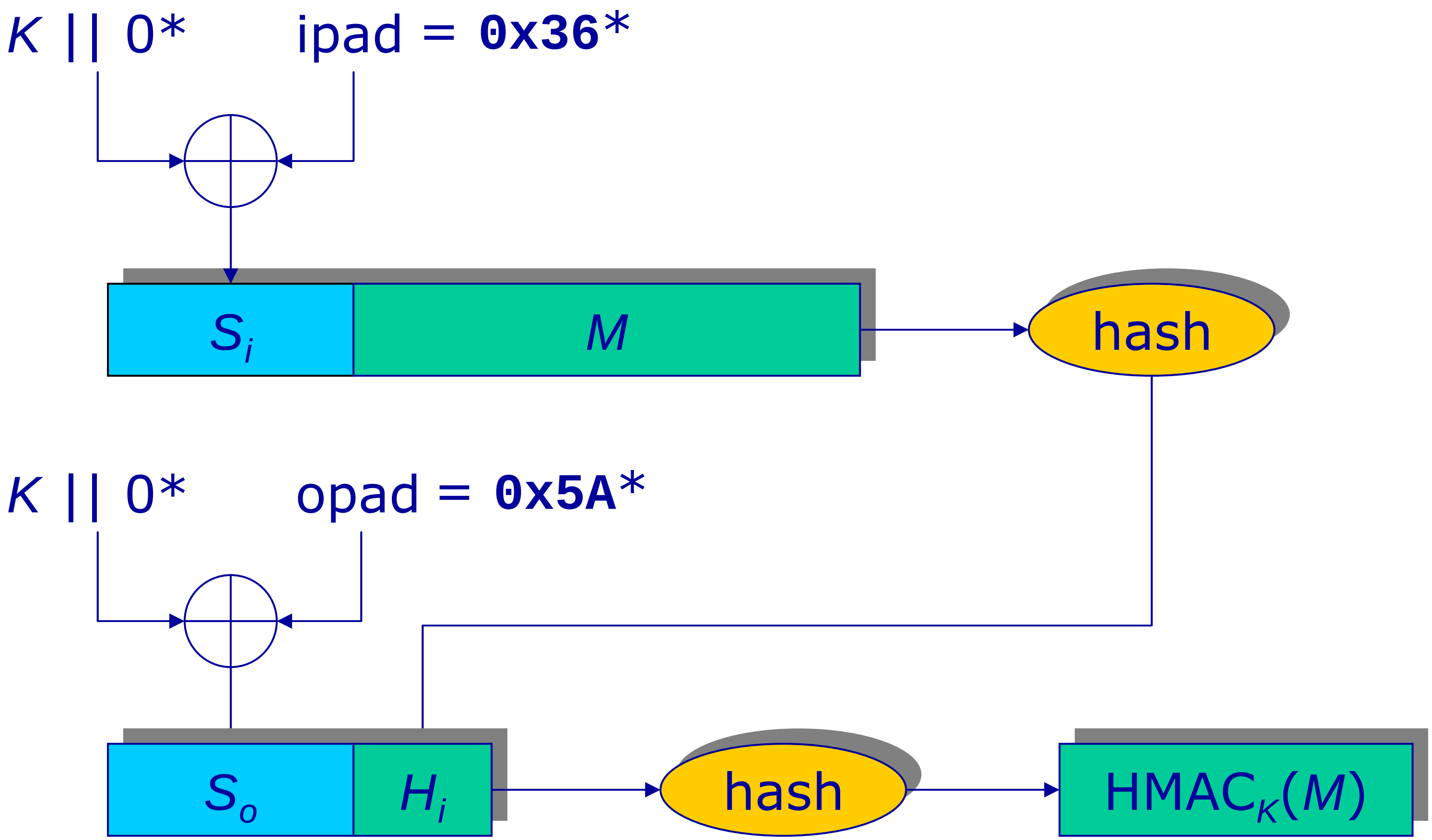
  $S_i = (K \mid\mid 0^*) \oplus \text{ipad}$, ipad = **0x36***.

  $S_o = (K \mid\mid 0^*) \oplus \text{opad}$, opad = **0x5A***.

- Hash interno: $H_i = \text{hash}(S_i \mid\mid M)$.

- Hash externo: $\text{HMAC}_K(M) = \text{hash}(S_o \mid\mid H_i)$.

HMAC



Números Aleatórios

Estudo de caso

- Relembrando: Netscape 1.x (1995).
- Dois estudantes de graduação de Berkeley descrevem como quebrar a segurança do navegador, recuperando chaves de sessão SSL em ≈25 s.
- Chaves de 128 bits?

Análise de segurança

- Baixa entropia das chaves de sessão!
- Estratégia:
  - Engenharia reversa do gerador de números aleatórios.
  - Chaves geradas a partir do clock (precisão de μs), sem acúmulo entre ativações do navegador.
  - Conhecendo o minuto em que a sessão SSL foi estabelecida, há menos de 70 milhões de chaves possíveis (2^{26} contra 2^{128}).
  - Testam-se todas elas.
- Exercício: propor uma solução.

Entropia

- Medida do *desconhecimento* que se tem sobre um sistema.
- Entropia é necessária para gerar chaves e outras informações de caráter privativo, imprevisível ou irrepetível.
- A segurança de um sistema pode depender criticamente das fontes de entropia que utiliza.

Fontes de entropia

- Fontes de entropia bruta (aleatoriedade) são todas de origem extra-criptográfica.

- As melhores fontes são *físicas* (ruído térmico, decaimento radioativo), mas podem também ter origem *comportamental* (estatísticas de rede e outros sistemas com grande número de incógnitas).

Geradores pseudo-aleatórios

- Frequentemente, a capacidade de produção de uma fonte não atende às necessidades de um sistema.
- Possível solução:
  - Coletar entropia real suficiente para uma semente de tamanho adequado.
  - Usar uma fórmula iterativa determinística para produzir uma sequência “indistinguível” de uma sequência aleatória.

- Como construir geradores pseudo-aleatórios seguros?

Geradores pseudo-aleatórios

- Como construir geradores pseudo-aleatórios seguros?

- Construções derivadas de:
  - Algoritmos simétricos (especialmente cifras de bloco).
  - Funções de hash.
  - Problemas computacionais (e.g. Blum-Blum-Shub).

Usando Cifras de Bloco

- Mantém-se um contador com o tamanho típico de um bloco (o valor inicial é irrelevante).
- A semente aleatória é usada como chave.
- Em cada passo, o contador é incrementado e cifrado.
- O valor cifrado constitui um bloco de bits pseudo-aleatórios, extraídos sob demanda.

Usando Funções de Hash

- Mantém-se um contador com o tamanho do hash produzido.
- O valor inicial é a semente aleatória.
- Em cada passo, o contador é incrementado e submetido à função de hash.
- O valor de hash constitui um bloco de bits pseudo-aleatórios, extraídos sob demanda.

Gerador DSS

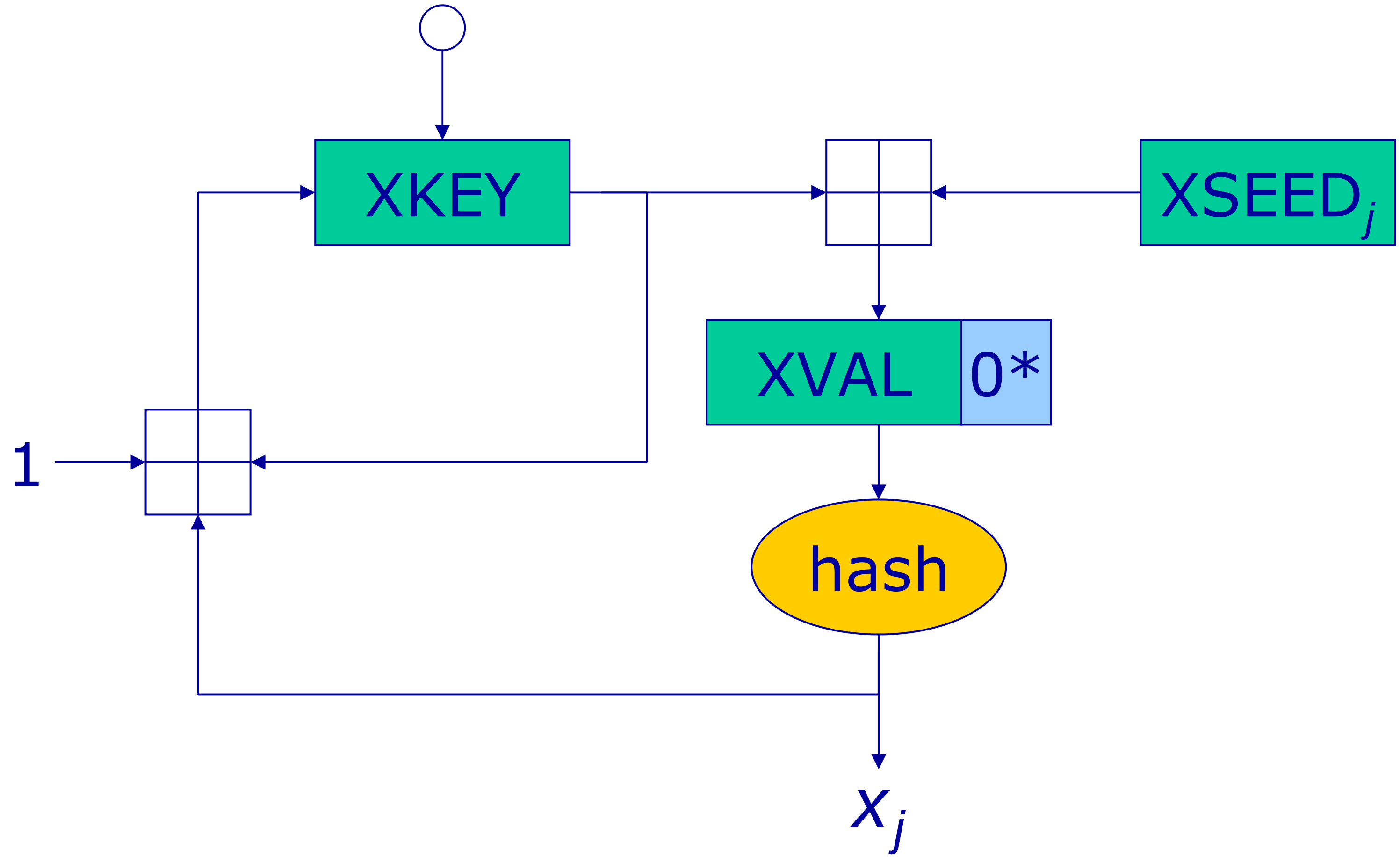
- FIPS 186-2, apêndices 3.1 – 3.3.
- Especificado para a geração de parâmetros e chaves (EC)DSA; uso muito difundido em outras aplicações.
- Produz uma sequência de inteiros de *n* bits, onde *n* é o tamanho do hash utilizado (operando em blocos de *b* bits).
- O estado interno não é um contador sequencial (valor depende da última saída produzida).

Gerador DSS

- Semente (entropia externa): XKEY.
- Geração de m valores $x_0, \ldots, x_{m-1}$ de n bits:

for $j \leftarrow 0, \ldots, m-1$ {
    $\text{XSEED}_j \leftarrow$ (entrada opcional de usuário)
    $\text{XVAL} \leftarrow (\text{XKEY} + \text{XSEED}_j) \bmod 2^n$
    $x_j \leftarrow \text{hash}(\text{XVAL} \mid\mid 0^{b-n})$
    $\text{XKEY} \leftarrow (1 + \text{XKEY} + x_j) \bmod 2^n$
}

Gerador DSS



Problemas

- Pontos fixos:
  - $x_j \equiv -1 \pmod{2^n}$, entrada de usuário constante.

- Atualização de XKEY:
  - XKEY $\leftarrow$ (1 + XKEY + x_j) mod 2^n = XKEY

- Todos os valores subsequentes $x_{j+1}, \ldots, x_{m-1}$ são *iguais*!

- Felizmente, a probabilidade é muito baixa.

Blum-Blum-Shub

- Inicialização:
  - gerar aleatoriamente (entropia física) dois números primos p, $q \equiv 3 \pmod 4$, distintos e secretos;
  - calcular $n \leftarrow pq$;
  - selecionar aleatoriamente (entropia física) uma semente $0 < s < n$ tal que $\mathrm{GCD}(s, n) = 1$;
  - calcular $x_0 \leftarrow s^2 \bmod n$.

- Geração do i-ésimo bit pseudo-aleatório z_i:
  - atualizar $x_i \leftarrow {x_{i-1}}^2 \bmod n$;
  - devolver $z_i \leftarrow x_i \bmod 2$ (bit menos significativo de x_i).

Blum-Blum-Shub

- Hipótese de segurança: o problema da fatoração de números inteiros (IFP) é computacionalmente intratável.
- O tamanho de n deve ser o mesmo de um módulo RSA com o nível de segurança desejado (no mínimo 1024–2048 bits).
- É possível acelerar o gerador BBS extraindo até $\approx \lg \lg n$ bits em cada passo. Para tamanhos práticos de n, são $\approx$10–11 bits).

Acumulando entropia

- Geradores pseudo-aleatórios via de regra armazenam amostras de entropia de um gerador entre duas ativações do sistema.

- A entropia acumula-se ao longo da operação do gerador, garantindo um mínimo de perdas (principalmente se a entropia da fonte bruta for escassa).

Cuidados especiais

- Coletar entropia do maior número possível de fontes.
- Escolher uma construção baseada em cifra de bloco ou função de hash apropriada.
- Mudar a chave periodicamente a partir da fonte de entropia bruta.
- Análise estatística (NIST SP 800-22).

Quotização de Segredo

Quotização de Segredo

- Distribuição de poderes.
- Divisão de responsabilidade.
- Segredo *distribuído*: cada custódio pode usar o segredo individualmente ☹
- Segredo *particionado*: no mínimo *k* custódios são necessários para recompor o segredo ☺

Partição por Fragmentação

- Segredo de m bits particionado em k quotas de m/k bits (recuperado por concatenação).

- Problemas:
  - As quotas precisam ser apresentadas sempre na mesma ordem para recompor o segredo.
  - Dadas $q < k$ quotas, apenas $m - q(m/k)$ bits são desconhecidos.

- Resultado: o esforço de completar o segredo diminui exponencialmente com o número de custódios.

Exemplo

- Segredo de 128 bits fragmentado em 4 quotas de 32 bits.

Custódios Corruptos	Esforço de Recuperação
1	2^{96}
2	2^{64}
3	2^{32}
4	0

Quotização absoluta

- Objetivo: particionar um segredo x de m bits entre exatamente k custódios, de modo que nenhum conluio de $q < k$ custódios possa recuperar o segredo.

- Solução: gerar e distribuir entre os custódios $k-1$ instâncias $s_1, \ldots, s_{k-1}$ de OTP de m bits, mais o valor $s_k = x \oplus s_1 \oplus \ldots \oplus s_{k-1}$.

- Forte e frágil: a falta de uma só quota si impede a recuperação do segredo (qualquer valor de m bits é equiprovável).

Hipótese de Trabalho

- A probabilidade de corrupção diminui com o número de custódios.
- Gerenciamento de risco: dimensionamento apropriado de k.
- Por robustez, é preciso admitir um universo de $n \geq k$ custódios (particionamento k de n).

Quotização de Shamir

- Caso $k = 2$.
- Idéia geométrica:
  - Por um ponto qualquer num plano passam infinitas retas.
  - Dois pontos distintos quaisquer determinam uma reta completamente.
- A própria reta é o segredo quotizado, as quotas são os pontos.

Interpolação de Lagrange

- Generalização: polinômios de grau $k-1$ são determinados por k pontos distintos.

- Por $q < k$ pontos quaisquer passam infinitos polinômios de grau $k-1$.

- A indeterminação do segredo (polinômio quotizado) é **total** se $q < k$ pontos forem conhecidos, como na quotização absoluta.

Interpolação de Lagrange

- Na prática, os polinômios têm coeficientes inteiros mod *p* (aritmética modular).
- O segredo pode ser reduzido a apenas um coeficiente do polinômio (inteiro mod *p*).
- A abscissa de cada ponto pode ser usada para identificar publicamente a quota.
- Cada quota (ordenada do ponto) tem o mesmo tamanho do segredo (inteiro mod *p*).

Exemplo

- Segredo de 128 bits dividido em 4 quotas de 128 bits.

Custódios Corruptos	Esforço de Recuperação
1	2^{128}
2	2^{128}
3	2^{128}
4	0

Outras Propriedades

- Possibilidade de estender o número de quotas (k de n).
- Quotização hierárquica simples: atribuição de mais de uma quota por custódio (pouco prático).
- Métodos mais eficientes e hierarquias mais complexas.

Epílogo

Distribuição de Chave

- Possuindo um gerador pseudo-aleatório adequado, um algoritmo simétrico seguro e um código de autenticação correspondente, é possível estabelecer comunicações seguras?
- ... Sim, se as entidades que se comunicam conhecerem as chaves utilizadas pelo algoritmo simétrico e pelo código de autenticação.
- *Como transmitir seguramente essas chaves*?
- *Como saber se as entidades são autênticas*?