Laboratório de Redes

UFPB

Centro de Informática
UFPB

Caps. 20 e 21: Algoritmos criptográficos

Iguatemi E. Fonseca

iguatemi@ci.ufpb.br

Agenda

- Cifração simétrica e confidencialidade de mensagens
  - Estrutura de cifra de Feistel
  - Algoritmo DES e Triplo DES (3DES)
  - Algoritmo AES
  - Modos de operação de cifra de bloco
- Criptografia de chave pública e autenticação de mensagem
  - Funções de hash seguras
  - Algoritmo RSA
  - Algoritmo Diffie-Hellman

Cifração simétrica e confidencialidade de mensagens

Algoritmos Simétricos de Cifração de Bloco

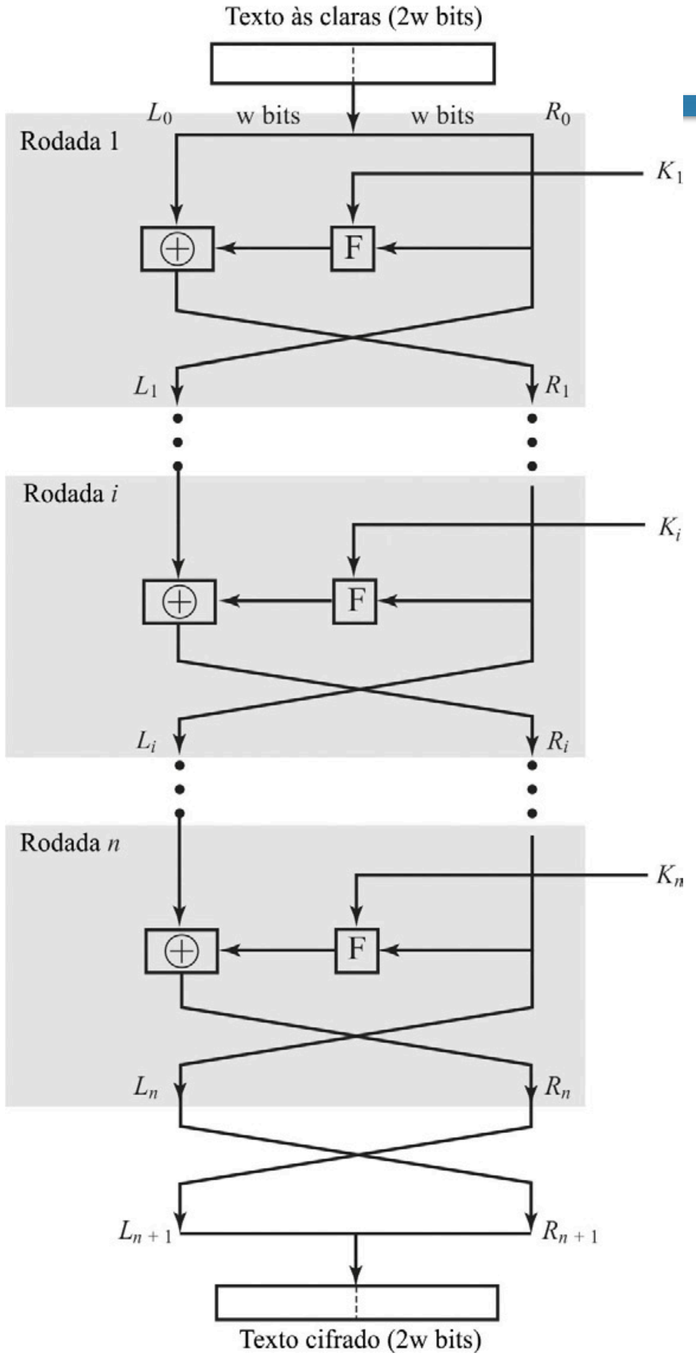
- Os algoritmos de cifração simétricos mais comumente usados são cifras de bloco
- Uma cifra de bloco processa o texto às claras fornecido como entrada em blocos de tamanho fixo e produz um bloco de texto cifrado de tamanho igual para cada bloco de texto às claras
- Algoritmos simétricos mais importantes:
  - Data Encryption Standard (DES)
  - Triple DES (DES triplo)
  - Advanced Encryption Standard (AES)

Estrutura de cifra de Feistel

- Muitos algoritmos simétricos de cifração de bloco, incluindo o DES, têm uma estrutura que foi descrita pela primeira vez por Horst Feistel, da IBM, em 1973
- As entradas para o algoritmo de cifração são um bloco de texto às claras de 2w bits de comprimento e uma chave K
- O bloco de texto às claras é dividido em duas metades, L_0 e R_0. As duas metades dos dados passam por n rodadas (rounds) de processamento e então se combinam para produzir o bloco de texto cifrado

Estrutura de cifra de Feistel

- Cada rodada i tem como entradas L_{i-1} e R_{i-1}, derivadas das rodadas anteriores, bem como uma subchave K, derivada da K global. Em geral, as subchaves K são diferentes de K e de cada uma das outras, e são geradas a partir da chave por um algoritmo de geração de subchaves
- Uma substituição é executada na metade esquerda dos dados. Essa substituição é feita mediante a aplicação de uma função de rodada F sobre a metade direita dos dados e depois executando a operação de OU exclusivo (XOR) entre a saída daquela função e a metade esquerda dos dados
- Depois dessa substituição, é executada uma permutação que consiste no intercâmbio das duas metades dos dados



- A exata execução de uma cifra de bloco simétrica depende da escolha dos seguintes parâmetros e aspectos de projeto:
  - Tamanho do bloco
  - Tamanho da chave
  - Número de rodadas
  - Algoritmo de geração de subchaves
  - Função de rodada
- Outras considerações no projeto de uma cifra de bloco simétrica
  - Software de cifração/decifração rápida
  - Facilidade de análise

Padrão de Cifração de Dados (DES)

- Adotado em 1977 pelo NIST
- O algoritmo DES pode ser descrito da seguinte maneira:
  - o texto às claras tem 64 bits de comprimento e a chave tem 56 bits de comprimento
  - quantidades maiores de texto às claras são processadas em blocos de 64 bits
  - a estrutura do DES é uma ligeira variação da rede de Feistel
  - há 16 rodadas de processamento. A partir da chave original de 56 bits são geradas 16 subchaves, cada uma delas usada em uma rodada
- O processo de decifração com o DES é essencialmente o mesmo que o processo de cifração. A regra é a seguinte: use o texto cifrado como entrada para o algoritmo DES, mas use as subchaves K em ordem inversa

DES Triplo (3DES)

- Foi o primeiro padronizado para uso em aplicações financeiras no padrão ANSI X9.17 em 1985
- Usa três chaves e três execuções do algoritmo DES
- Com três chaves distintas, o 3DES tem comprimento de chave efetivo de 168 bits
- O 3DES é utilizado em sistemas legados. Atualmente, o AES é o mais usado em sistemas atuais
- A tendência é o AES substituir o 3DES com o tempo

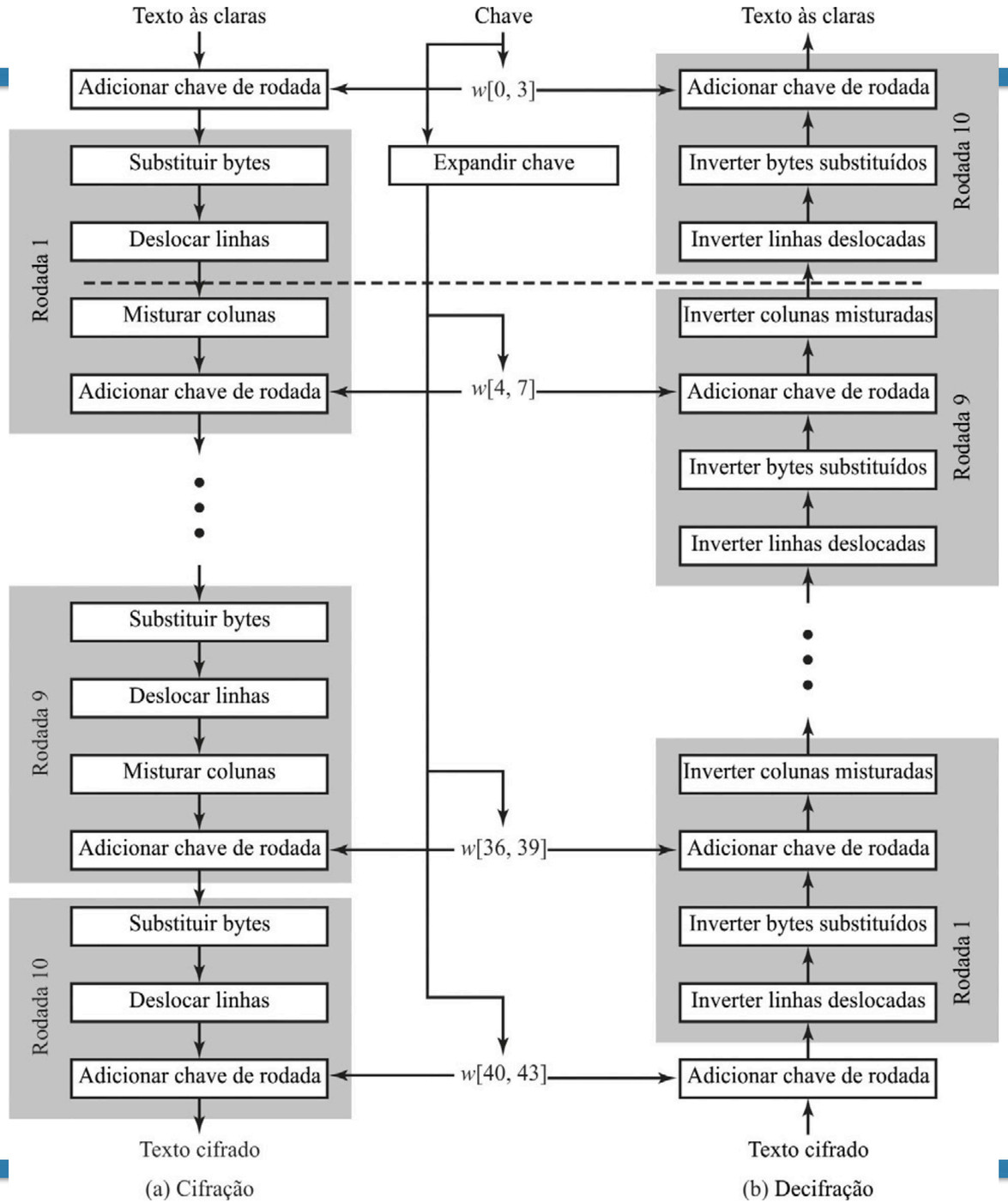
Padrão de Cifração Avançado (AES)

- O AES usa comprimento de bloco de 128 bits e comprimento de chave que pode ser de 128, 192 ou 256 bits
- A entrada para os algoritmos de cifração e decifração é um único bloco de 128 bits
- Os seguintes comentários nos dão uma percepção melhor do AES:
  - O AES não usa uma estrutura de Feistel, mas processa o bloco de dados inteiro em paralelo durante cada rodada, usando substituições e permutações
  - A chave que é passada como entrada é expandida para um vetor de 44 palavras de 32 bits, w[i]. Quatro palavras (128 bits) distintas servem como chave de rodada para cada rodada

Padrão de Cifração Avançado (AES)

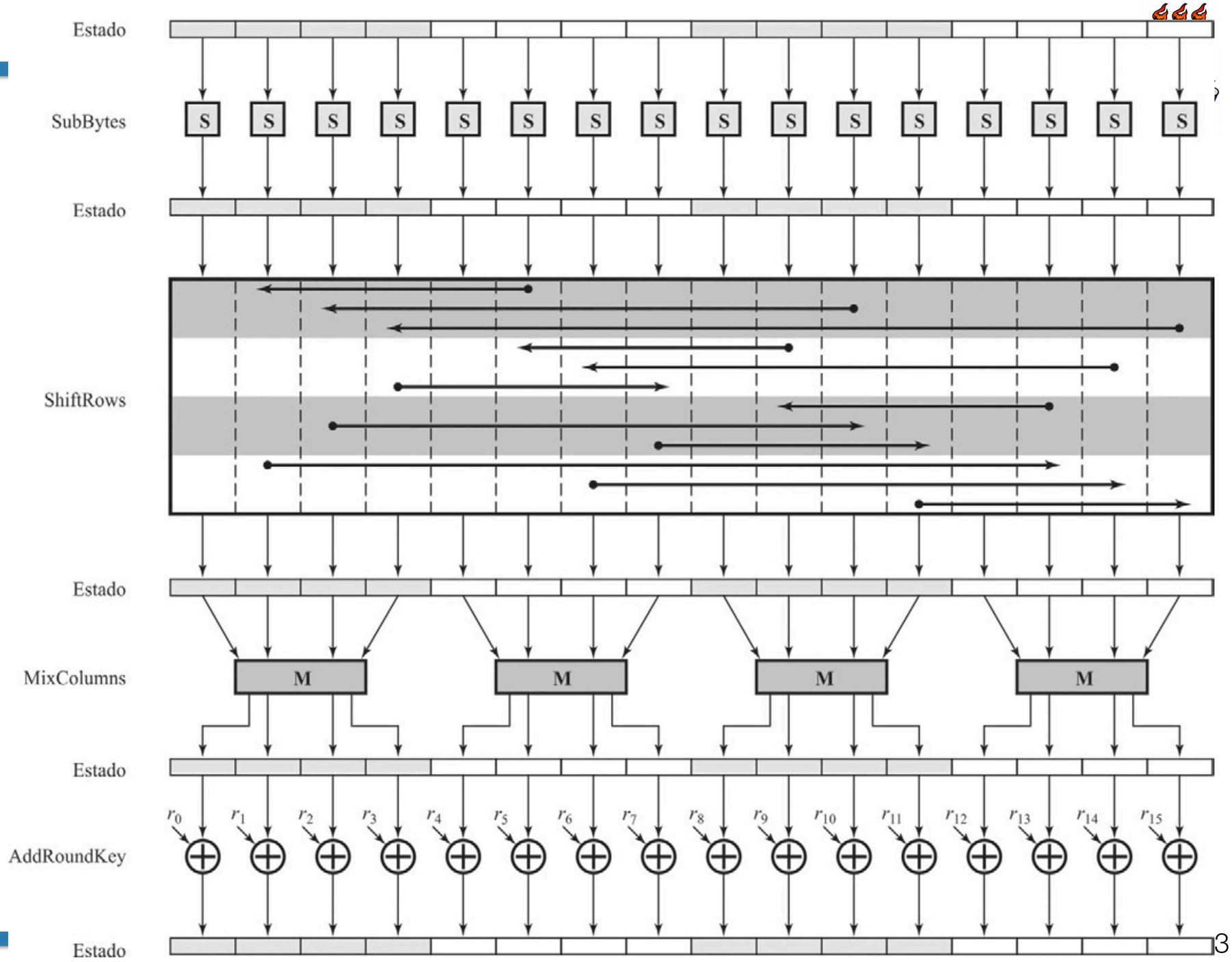
- Quatro estágios diferentes são usados, um de permutações e três de substituição:
  - **Substituir bytes (SubBytes)**: consiste em uma simples consulta a uma tabela
  - **Deslocar linhas (ShiftRows)**: consiste em deslocamento ordenado de linhas de bits
  - **Misturar colunas (MixCollumns):** Cada byte de uma coluna é mapeado para um novo valor que é função de todos os 4 bytes na coluna
  - **Adicionar chave de rodada (AddRoundKey)**: os 128 bits de Estado passam por uma operação de XOR bit a bit com os 128 bits da chave de rodada
- A estrutura é bastante simples. Para ambas, cifração e decifração, a cifra começa com um estágio de **Adicionar chave de rodada** (Add Round Key), seguido por nove rodadas, cada uma incluindo todos os quatro estágios, seguidas por uma décima rodada de três estágios

Padrão de Cifração Avançado (AES)



Laboratório de Redes

Estrutura de uma rodada completa de cifração do AES



Padrão de Cifração Avançado (AES)

- Só o estágio **Adicionar chave de rodada** (AddRoundKey) faz uso da chave. Por essa razão, a cifra começa e termina com um estágio **Adicionar chave de rodada**. Qualquer outro estágio, aplicado no início ou no final, é reversível sem conhecimento da chave e, portanto, não adicionaria qualquer segurança
- O estágio Adicionar chave de rodada por si só não seria suficientemente poderoso. Os outros três estágios juntos misturam os bits, mas por si sós não proveriam segurança porque não usam a chave. Podemos ver a cifra como operações alternadas de cifração de um bloco via XOR (Adicionar chave de rodada), seguidas pela mistura do bloco (os outros três estágios), seguidas por cifração via XOR, e assim por diante. Esse esquema é eficiente e também fornece alta segurança

Padrão de Cifração Avançado (AES)

- Cada estágio é facilmente reversível. Para os estágios Substituir byte, Deslocar linha e Misturar coluna (SubBytes, ShiftRows e MixColumns) uma função inversa é usada no algoritmo de decifração. Para o estágio Adicionar chave de rodada, consegue-se o inverso aplicando XOR entre a mesma chave de rodada e o bloco, usando a propriedade de que $A \oplus A \oplus B = B$.
- Como ocorre com a maioria das cifras de bloco, o algoritmo de decifração faz uso da chave expandida em ordem inversa. Todavia, o algoritmo de decifração não é idêntico ao algoritmo de cifração. Essa é uma consequência da estrutura particular do AES

Padrão de Cifração Avançado (AES)

- Uma vez estabelecido que todos os quatro estágios são reversíveis, é fácil verificar que a decifração realmente recupera o texto às claras. Em cada ponto horizontal (p. ex., a linha tracejada na figura), o valor de Estado é o mesmo para ambas, cifração e decifração
- A rodada final de ambas, cifração e decifração, consiste em somente três estágios. Novamente, essa é uma consequência da estrutura particular do AES e é exigida para tornar a cifra reversível

Modos de Operação de Cifra de Bloco

- Para aplicar uma cifra de bloco em uma variedade de aplicações, cinco modos de operação foram definidos

Tabela 20.4 Modos de operação de cifras de bloco

Modo	Descrição	Aplicação típica
Livro-código eletrônico (ECB)	Cada bloco de *b* bits de texto às claras é codificado independentemente usando a mesma chave.	■ Transmissão segura de valores únicos (p. ex., uma chave criptográfica)
Encadeamento de blocos de cifra (CBC)	A entrada para o algoritmo de cifração é a operação de XOR entre os próximos *b* bits do texto às claras e os *b* bits precedentes do texto cifrado.	■ Transmissão orientada a blocos de forma geral ■ Autenticação
Realimentação de cifra (CFB)	A entrada é processada *s* bits por vez. O texto cifrado precedente é usado como entrada para o algoritmo de cifração produzir uma saída pseudoaleatória, que então passa por uma operação de XOR com o texto às claras para produzir a próxima unidade de texto cifrado.	■ Transmissão orientada a fluxo de forma geral ■ Autenticação
Realimentação de saída (OFB)	Semelhante ao modo CFB, exceto que a entrada para o algoritmo de cifração é a saída da cifração precedente.	■ Transmissão orientada a fluxo por canal com ruído (p. ex., comunicação via satélite)
Contador (CTR)	Cada bloco de texto às claras passa por uma operação de XOR com valor de contador previamente cifrado. O contador é incrementado para cada bloco subsequente.	■ Transmissão orientada a fluxo de forma geral ■ Útil para requisitos de alta velocidade

Criptografia de chave pública e autenticação de mensagem

Funções de hash seguras (hash de uma via)

- É importante não apenas em autenticação de mensagem, mas também em assinaturas digitais
- Todas as funções de hash operam usando os seguintes princípios gerais. A entrada (mensagem, arquivo etc.) é vista como uma sequência de blocos de n bits. A entrada é processada um bloco por vez, de modo iterativo, para produzir um hash de n bits
- Um dos algoritmos de funções hash seguras mais utilizados é o SHA (*Secure Hash Algorithm*)

SHA (Secure Hash Algorithm)

- SHA-1 produz um valor de hash de 160 bits. Foi extinto em 2005
- SHA-2, padronizado em 2002, produz um valor de hash de 256, 384 e 512 bits, conhecidas como SHA-256, SHA-384 e SHA-512

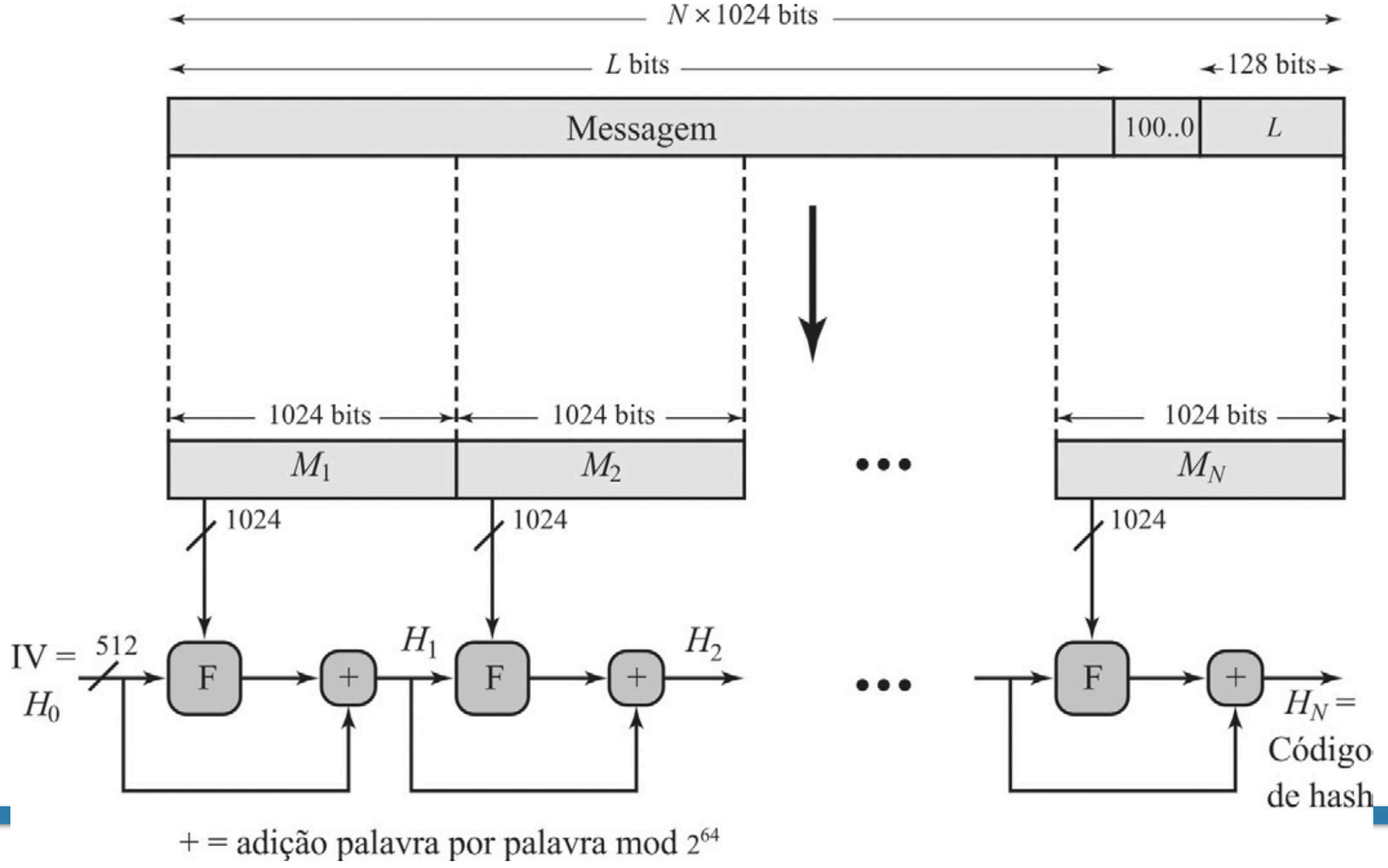
Tabela 21.1 Comparação de parâmetros do SHA

	SHA-1	SHA-256	SHA-384	SHA-512
Tamanho do resumo de mensagem	160	256	384	512
Tamanho da mensagem	$<2^{64}$	$<2^{64}$	$<2^{128}$	$<2^{128}$
Tamanho do bloco	512	512	1024	1024
Tamanho da palavra	32	32	64	64
Número de rodadas	80	64	80	80
Segurança	80	128	192	256

Notas: 1. Todos os tamanhos são medidos em bits.
2. A segurança refere-se ao fato de que um ataque baseado no paradoxo do aniversário a um resumo criptográfico de uma mensagem de tamanho n produz uma colisão com custo de trabalho de aproximadamente $2^{n/2}$.

SHA (Secure Hash Algorithm)

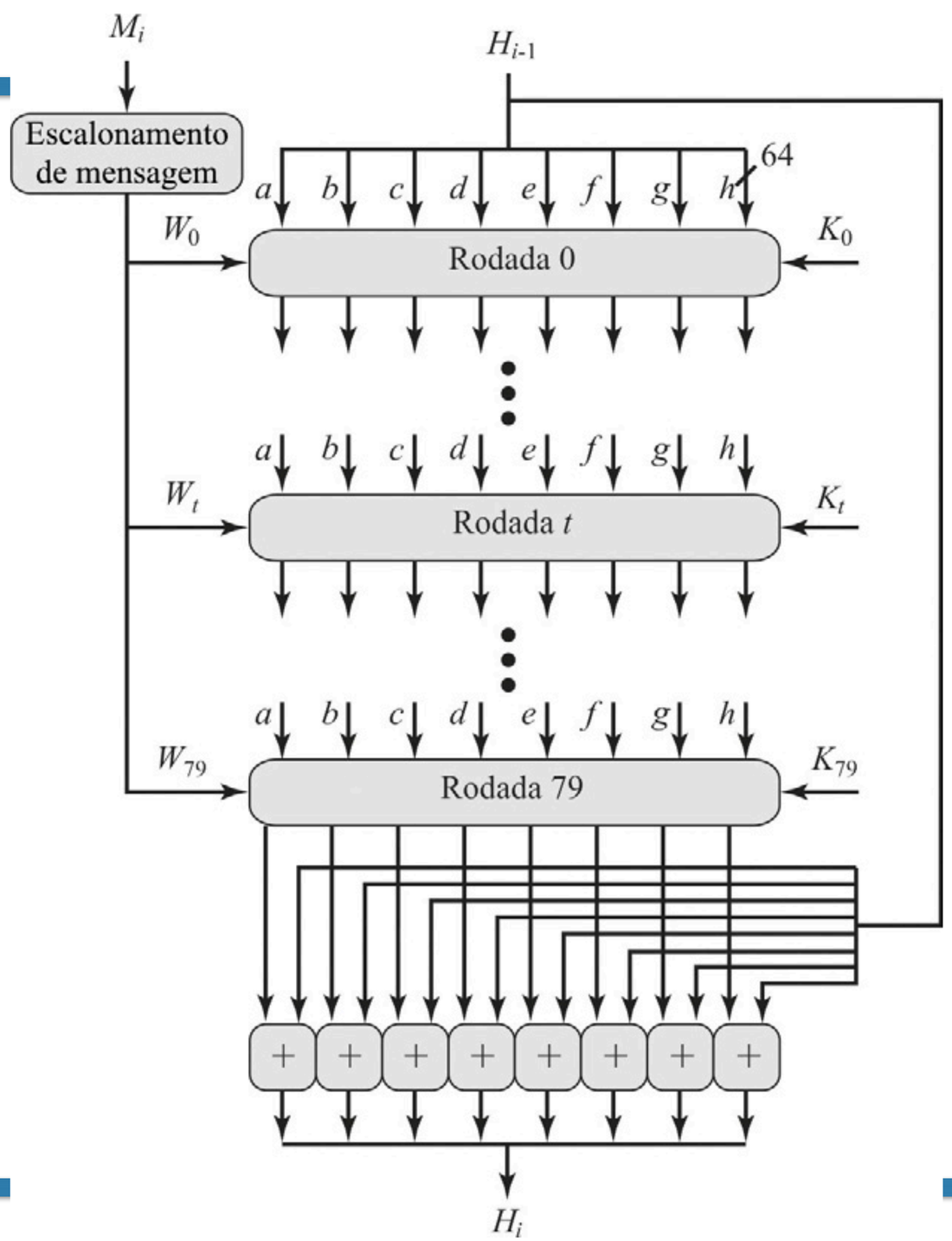
- Geração de resumo criptográfico de mensagem usando SHA-512



SHA (Secure Hash Algorithm)

- Etapas do SHA-512
  - **Etapa 1:** anexar bits de preenchimento. A mensagem é preenchida de modo que o seu comprimento seja congruente a 896 módulo 1024 [comprimento ≡ 896 (mod 1024)]
  - **Etapa 2:** concatenar comprimento. Um bloco de 128 bits é concatenado à mensagem
  - **Etapa 3:** inicializar buffer de hash. Um buffer de 512 bits é usado para guardar resultados intermediários e finais da função de hash. O buffer pode ser representado como oito registradores de 64 bits (a, b, c, d, e, f, g, h)

- Etapas do SHA-512
  - **Etapa 4:** processar mensagem em blocos de 1.024 bits (128 palavras). O coração do algoritmo é um módulo que consiste em 80 rodadas; esse módulo é indicado por **F** na figura (ver descrição no livro)
  - **Etapa 5: saída**. Depois que todos os N blocos de 1.024 bits foram processados, a saída do N-ésimo estágio é o resumo criptográfico de 512 bits da mensagem.



SHA (Secure Hash Algorithm)

- **SHA-3**
  - Em 2015, o NIST publicou a versão 3 do algoritmo SHA
  - Premissas básicas:
    - Deve ser possível substituir o SHA-2 pelo SHA-3 em qualquer aplicação mediante uma simples troca. Portanto, o SHA-3 deve suportar valores de comprimento de hash de 224, 256, 384 e 512 bits
    - O SHA-3 deve preservar a natureza on-line do SHA-2. Isto é, o algoritmo deve processar blocos comparativamente pequenos (514 ou 1.024 bits) de uma só vez, em vez de exigir que a mensagem inteira seja colocada em buffer na memória antes de seu processamento
- Possui estrutura totalmente diferente do SHA-1 e SHA-2

CIFRAÇÃO DE CHAVE PÚBLICA: ALGORITMO RSA

- Publicado em 1978 por Ron Rivest, Adi Shamir e Len Adleman
- É o algoritmo de chave pública mais utilizado
- Deve-se usar um tamanho de chave de pelo menos 2048 ou 3072 bits

CIFRAÇÃO DE CHAVE PÚBLICA: ALGORITMO RSA

Geração de chave

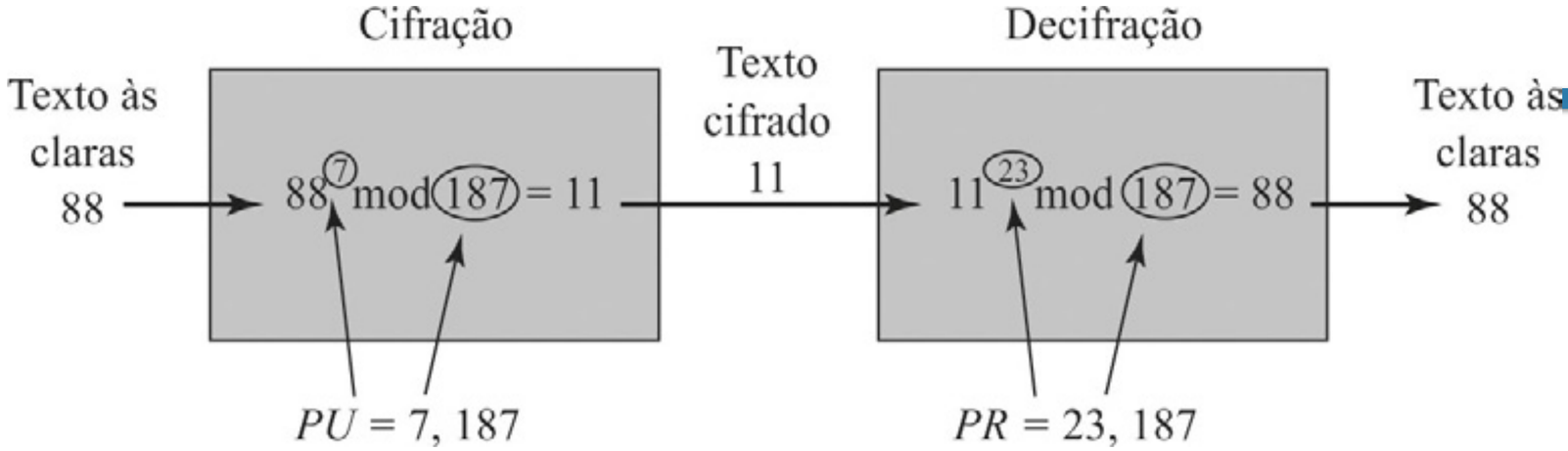
Selecionar p, q	p e q primos, $p \neq q$
Calcular $n = p \times q$	
Calcular $\phi(n) = (p-1)(q-1)$	
Selecionar inteiro e	$\gcd(\phi(n), e) = 1;\ 1 < e < \phi(n)$
Calcular d	$de \bmod \phi(n) = 1$
Chave pública	$KU = \{e, n\}$
Chave privada	$KR = \{d, n\}$

Cifração

Texto às claras:	$M < n$
Texto cifrado:	$C = M^e \pmod{n}$

Decifração

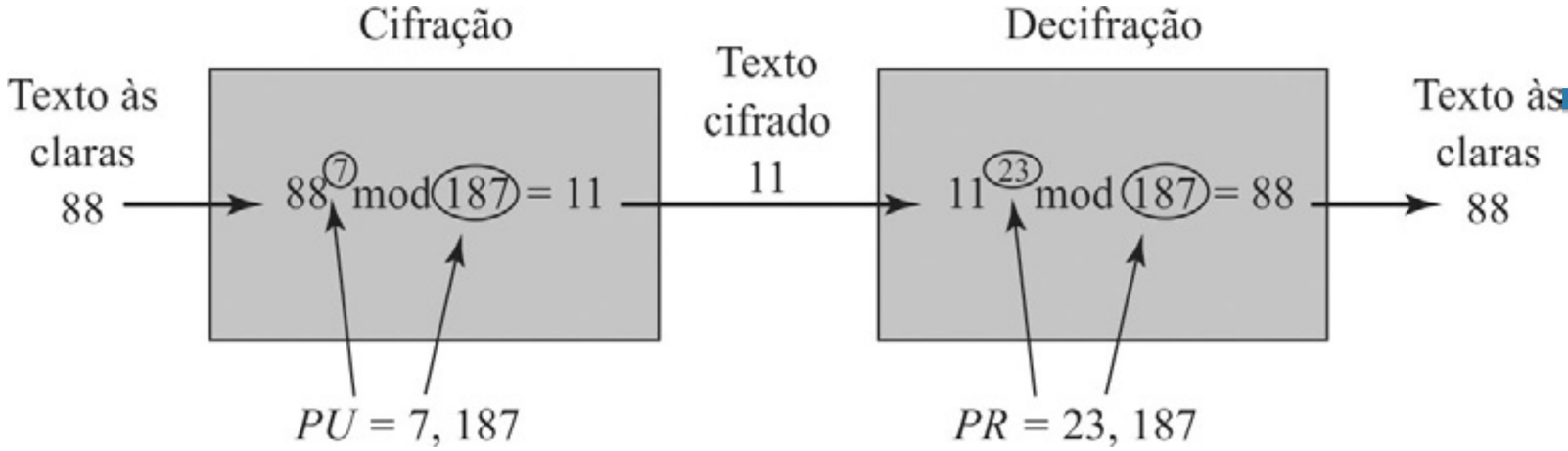
Texto cifrado	C
Texto às claras:	$M = C^d \pmod{n}$



Exemplo de uso do RSA

1. Selecionar dois números primos, $p = 17$ e $q = 11$.
2. Calcular $n = pq = 17 \times 11 = 187$.
3. Calcular $\phi(n) = (p - 1)(q - 1) = 16 \times 10 = 160$.
4. Selecionar e tal que e é relativamente primo de $\phi(n) = 160$ e menor que $\phi(n)$; escolhemos $e = 7$.
5. Determinar d tal que $de \bmod 160 = 1$ e $d < 160$. O valor correto é é $d = 23$, porque $23 \times 7 = 161 = (1 \times 160) + 1$.

As chaves resultantes são a chave pública $PU = \{7, 187\}$ e a chave privada $PR = \{23, 187\}$. O exemplo mostra a utilização dessas chaves para uma entrada de texto às claras $M = 88$. Para a cifração, precisamos calcular $C = 88^7 \bmod 187$. Explorando as propriedades da aritmética modular, podemos fazer isso da seguinte maneira:



Exemplo de uso do RSA

$$88^{7} \bmod 187 = [(88^{4} \bmod 187) \times (88^{2} \bmod 187) \times (88^{1} \bmod 187)] \bmod 187$$

$$88^{1} \bmod 187 = 88$$

$$88^{2} \bmod 187 = 7744 \bmod 187 = 77$$

$$88^{4} \bmod 187 = 59,969,536 \bmod 187 = 132$$

$$88^{7} \bmod 187 = (88 \times 77 \times 132) \bmod 187 = 894,432 \bmod 187 = 11$$

Para a decifração, calculamos $M = 11^{23} \bmod 187$:

$$11^{23} \bmod 187 = [(11^{1} \bmod 187) \times (11^{2} \bmod 187) \times (11^{4} \bmod 187) \times (11^{8} \bmod 187) \times (11^{8} \bmod 187)] \bmod 187 \times 11^{1} \bmod 187 = 11$$

$$11^{2} \bmod 187 = 121$$

$$11^{4} \bmod 187 = 14,641 \bmod 187 = 55$$

$$11^{8} \bmod 187 = 214,358,881 \bmod 187 = 33$$

$$11^{23} \bmod 187 = (11 \times 121 \times 55 \times 33 \times 33) \bmod 187 = 79,720,245 \bmod 187 = 88$$

Segurança do Algoritmo RSA

- Quatro possíveis abordagens de ataque ao algoritmo RSA são as seguintes:
  - **Força bruta**: Envolve tentar todas as possíveis chaves privadas
  - **Ataques matemáticos**: Há diversas abordagens, todas equivalentes em esforço à tarefa de fatorar o produto de dois primos
  - **Ataques de temporização**: Dependem do tempo de execução do algoritmo de decifração. Um atacante pode determinar uma chave privada monitorando o tempo que um computador leva para decifrar mensagens
  - **Ataques de texto cifrado escolhido**: Esse tipo de ataque explora propriedades do algoritmo RSA. Uma discussão desse ataque está fora do escopo deste livro

Ataques de fatoração ao RSA

- É um tipo de ataque matemático
- Usa técnicas como: *quadratic sieve* (“crivo quadrático”); *generalized number field sieve* (“crivo geral de corpos numéricos” — GNFS); *special number field sieve* (“crivo especial de corpos numéricos” — SNFS)

Tabela 21.2 Progresso na fatoração

Número de dígitos decimais	Número aproximado de bits	Data de consecução	Anos-MIPS
100	332	Abril de 1991	7
110	365	Abril de 1992	75
120	398	Junho de 1993	830
129	428	Abril de 1994	5.000
130	431	Abril de 1996	1.000
140	465	Fevereiro de 1999	2.000
155	512	Agosto de 1999	8.000
160	530	Abril de 2003	—
174	576	Dezembro de 2003	—
200	663	Maio de 2005	—

Algoritmo Diffie-Hellman

- O primeiro algoritmo de chave pública
- Foi publicado em 1976
- Também denominado "Acordo de chave de Diffie-Hellman”
- Vários produtos comerciais empregam essa técnica de acordo de chave
- A finalidade do algoritmo é habilitar dois usuários a estabelecerem uma chave secreta com segurança, que então pode ser usada para subsequente cifração de mensagens
- O algoritmo em si é limitado ao estabelecimento de chaves

Elementos públicos globais

q	Número primo
a	$a < q$ e a raiz primitiva de q

Geração de chave pelo usuário A

Selecionar X_A privada	$X_A < q$
Calcular Y_A pública	$Y_A = a^{X_A} \bmod q$

Geração de chave pelo usuário B

Selecionar X_B privada	$X_B < q$
Calcular Y_B pública	$Y_B = a^{X_B} \bmod q$

Geração de chave secreta pelo usuário A

$$K = (Y_B)^{X_A} \bmod q$$

Geração de chave secreta pelo usuário B

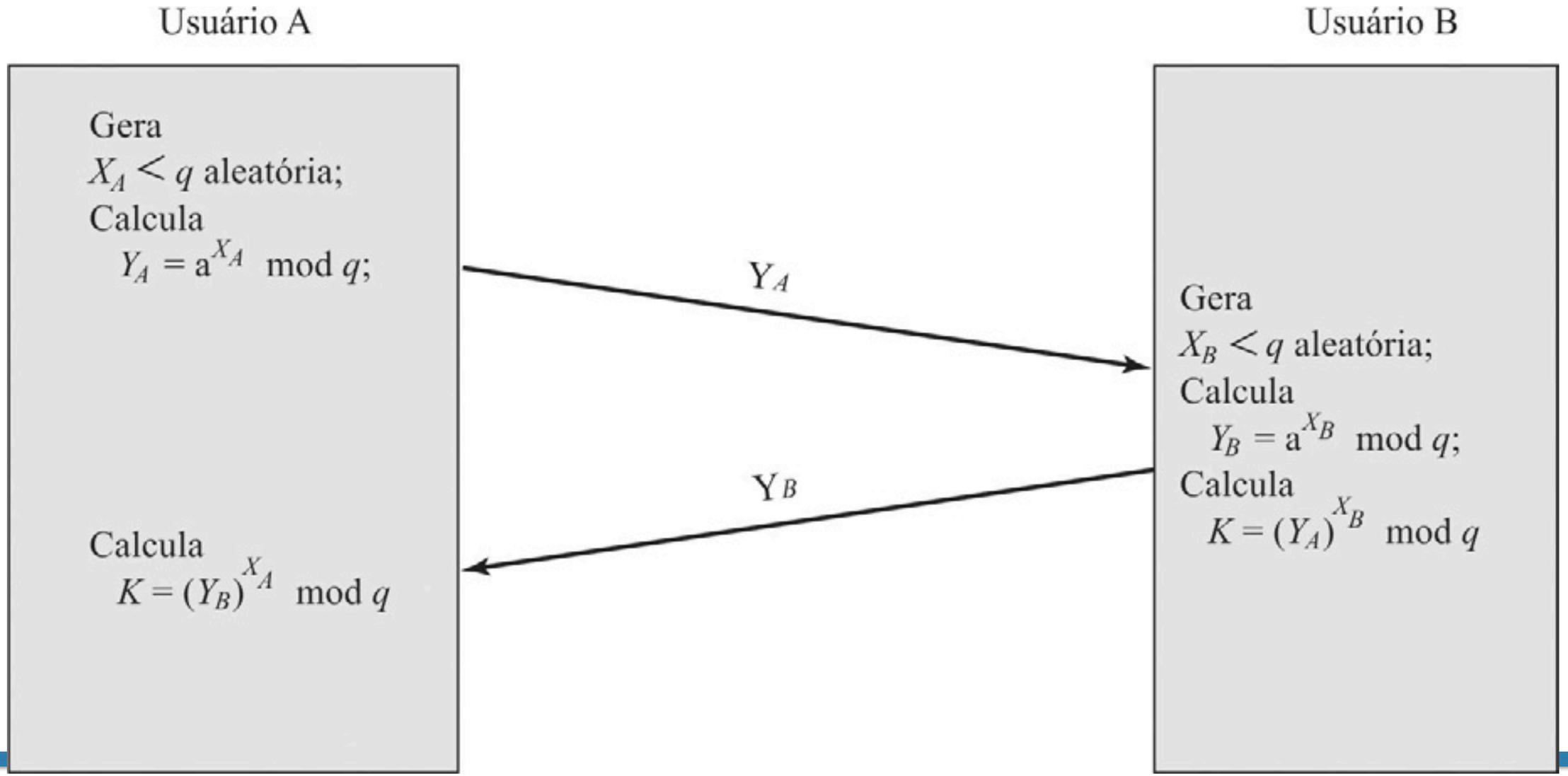
$$K = (Y_A)^{X_B} \bmod q$$

oritmo Diffie-Hellman

Algoritmo Diffie-Hellman

- A segurança do acordo de chave de Diffie-Hellman encontra-se no fato de que, embora seja relativamente fácil calcular exponenciais módulo um primo, é muito difícil calcular logaritmos discretos. Para primos grandes, a última tarefa é considerada inexequível

Exemplo de Protocolo que usa Algoritmo Diffie-Hellman



Obrigado!

Perguntas e encaminhamentos

Iguatemi E. Fonseca

iguatemi@ci.ufpb.br