

---

# Criptografia Aplicada

Jean Everson Martina

---

# Nota sobre a autoria

- Material por Jean Everson Martina
- Adição de conteúdo
  - Dayana Spagnuelo
  - Lucas Perin

Cryptography and Network Security  
Principles and Practice (second edition)  
William Stallings

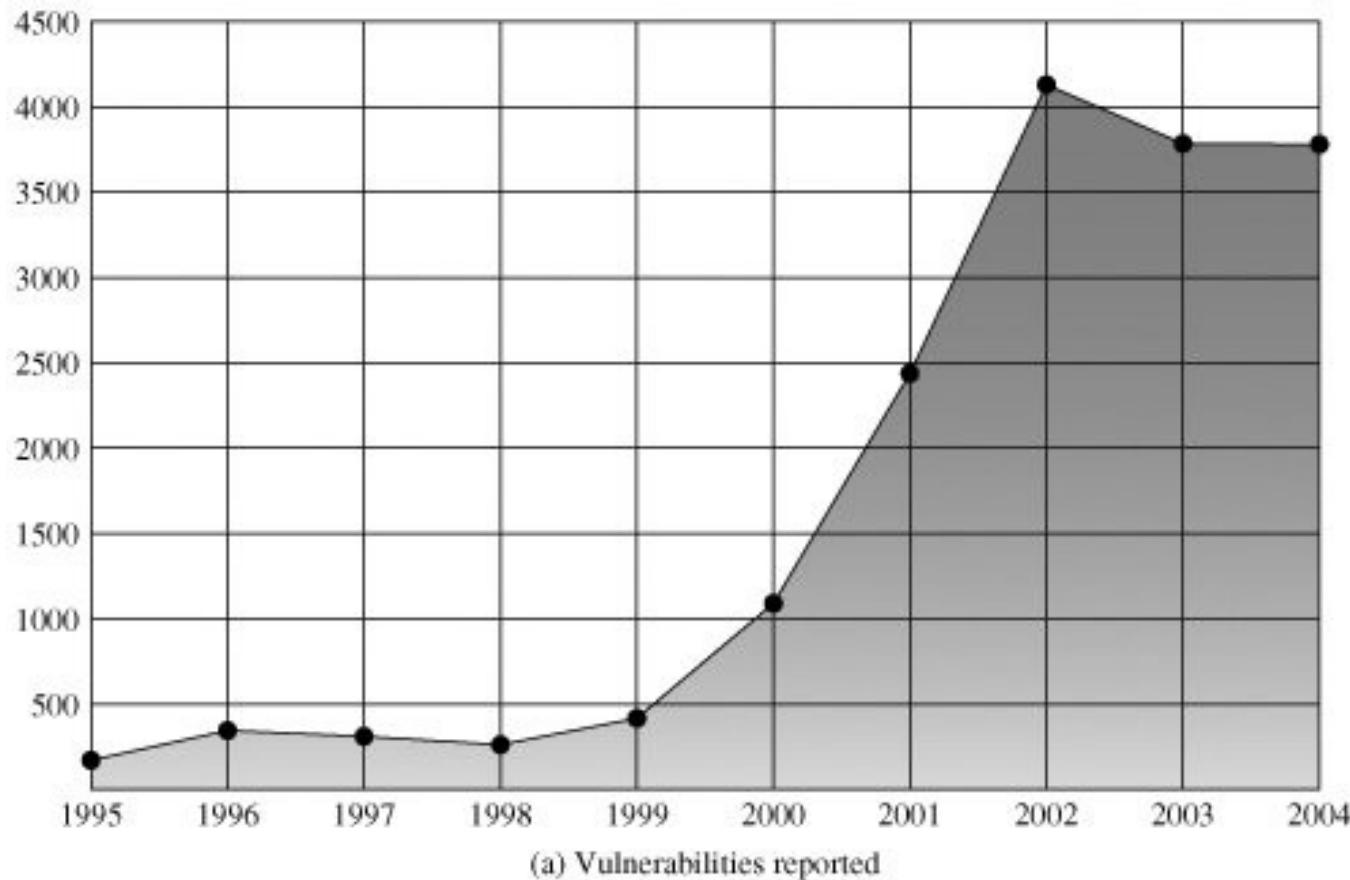
# Por que estudar segurança?

- RFC 1636 (Security in the internet architecture)  
lançado em 1994
  - Constatava que a internet precisava de mais segurança
  - Proteção de infra estrutura de rede
  - Monitoramento e controle de tráfego
  - Segurança user-end-user (autenticação e cifras)



*"On the Internet, nobody knows you're a dog."*

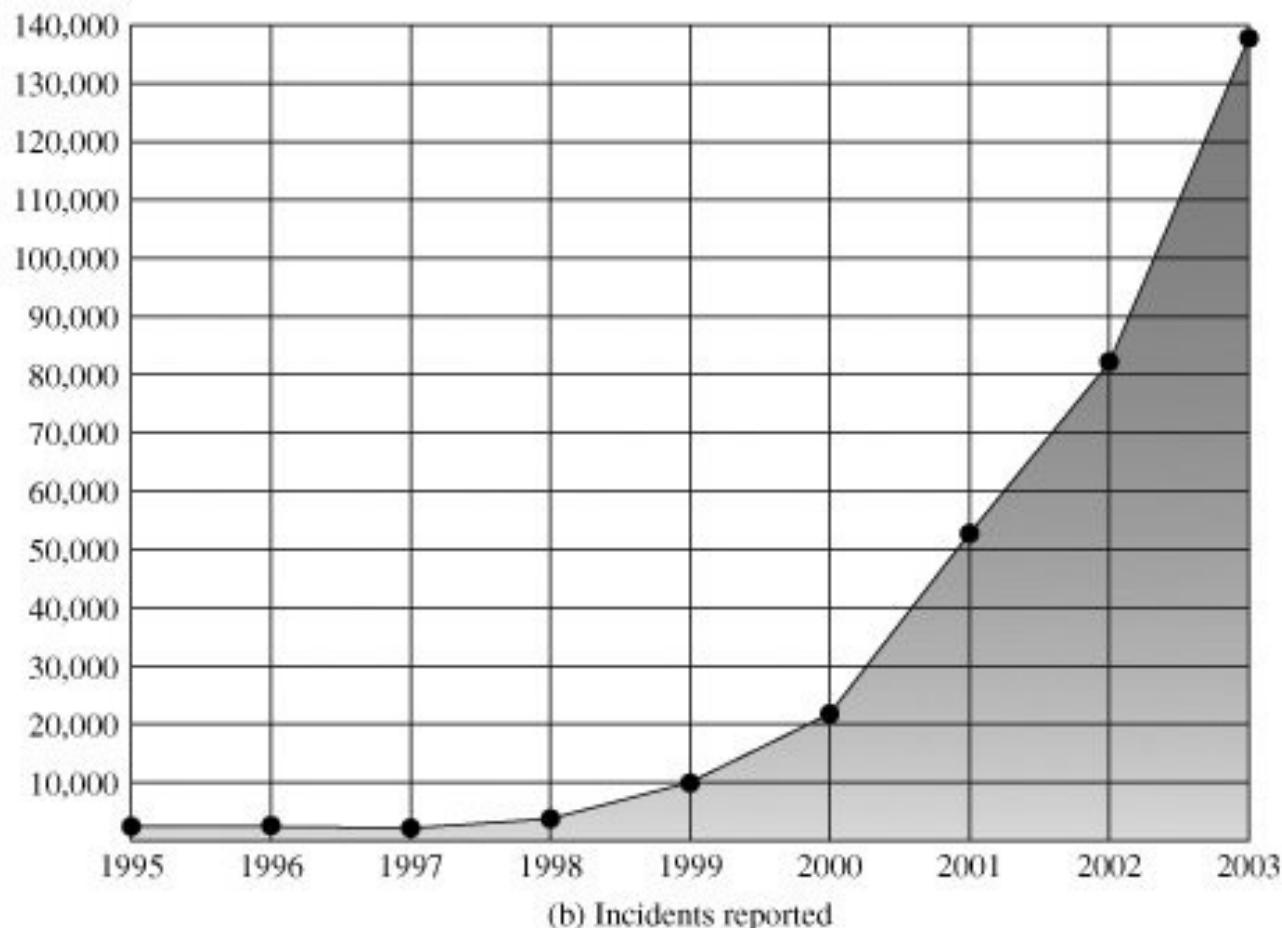
# Tendências em (In)Segurança



# Tendências em (In)Segurança

- Rede
- Aplicação
- Sistemas Operacionais
- Roteadores

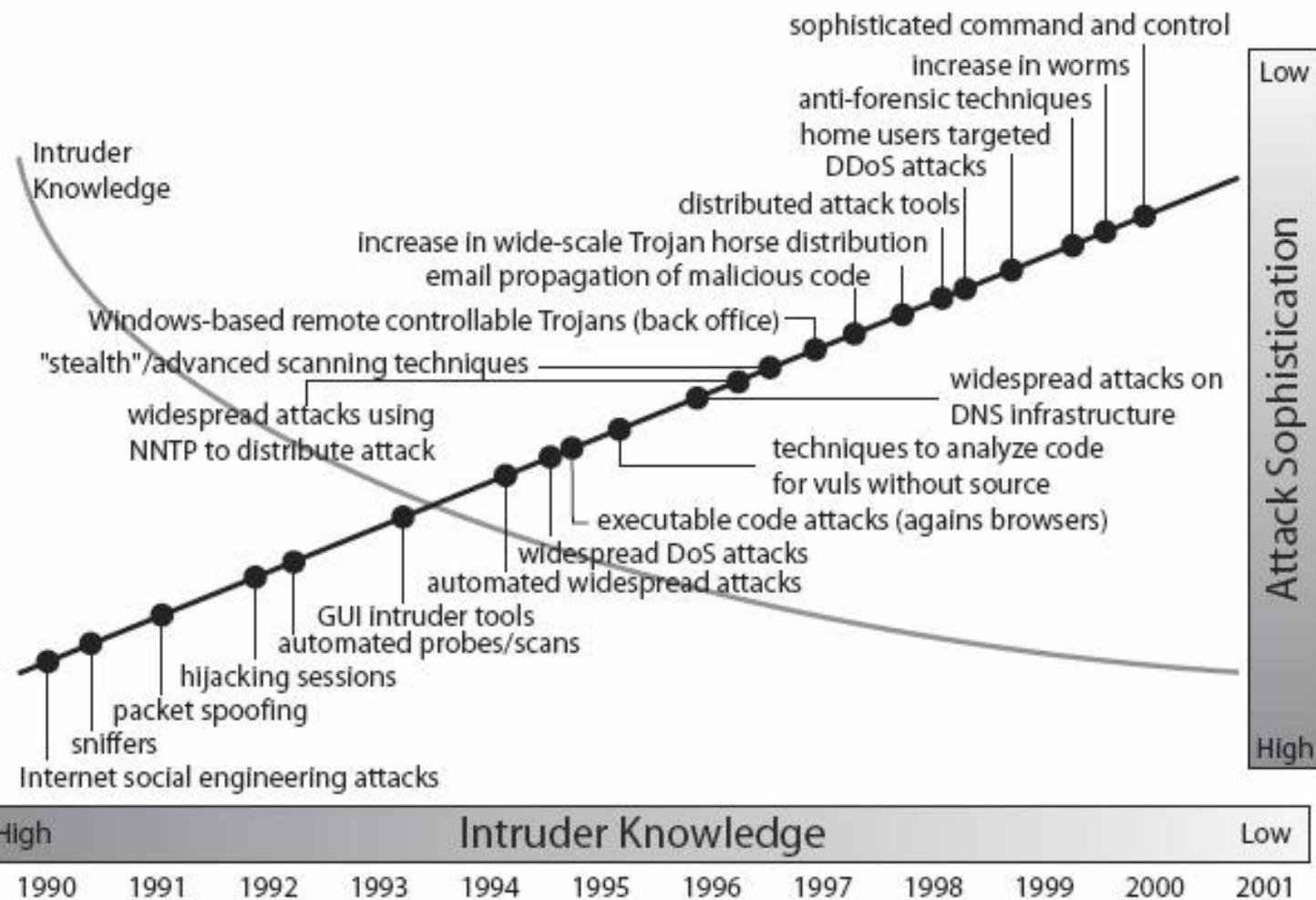
# Tendências em (In)Segurança



# Tendências em (In)Segurança

- Negação de serviço (DoS)
- IP Spoofing - pacotes com IP falso para explorar aplicativos que usam identificação a partir do IP
- Escuta de pacotes

# Tendências em (In)Segurança



Source: CERT

# Arquitetura de Segurança OSI

- Recomendação lançada em meados de 1991
- Área de segurança precisava de mais organização
- Modelo sistemático
  - Definir requisitos de segurança
  - Cumprir os requisitos definidos
  - Escolha de mecanismos e políticas de segurança
- Definiu conceitos utilizados até hoje

# Arquitetura de Segurança OSI

- Ameaça:
  - Potencial de violação
  - Vulnerabilidade
  - Brecha de segurança
- Ataque:
  - Investida em uma ameaça
  - Ação que compromete a segurança
  - Tentativa deliberada de explorar uma brecha

# Arquitetura de Segurança OSI

- Mecanismo de Segurança:
  - Processo ou dispositivo
  - Detectar, prevenir, ou recuperar ataques
- Serviços de Segurança:
  - Aumenta a segurança dos dados
  - Processo ou serviço
  - Contra ataques
  - 1 ou mais mecanismos de segurança

# Ataques Passivos

Ataques passivos tem objetivo de obter/ler informações do sistema sem afetar os recursos do mesmo.

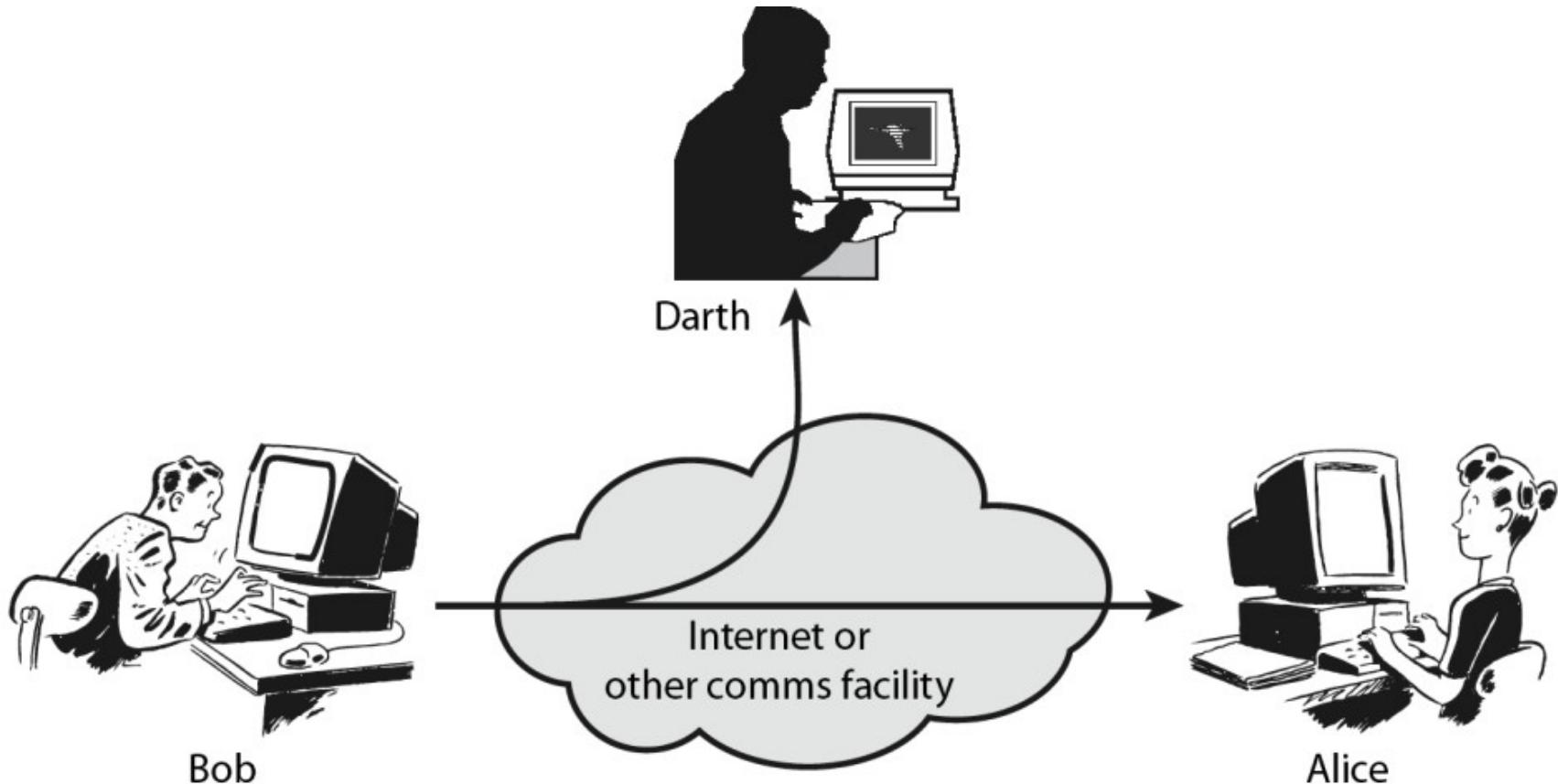
# Ataques Passivos

- Vazamento de Informação:
  - Engenharia Social
  - Descuido
- Analise de Trafego:
  - Inferência para obter a informação
  - Análise de freqüência e tamanho de mensagens

# Ataques Passivos

Ataques passivos são difíceis de detectar porque  
não alteram os dados

# Ataques Passivos



# Ataques Ativos

- Mascaramento
  - A acredita que C é B
  - Impersonating
- Repetição
  - Re-uso de informação trocada por A e B
- Modificação de Mensagens
  - Alteração do conteúdo da mensagem entre A e B
- Negação de Serviços
  - Prevenção da comunicação entre A e B

# Ataques Ativos x Passivos

- Ataques passivos são difíceis de detectar, mas fáceis de prevenir
- Ataques ativos são fáceis de detectar, mas difíceis de prevenir
  - Se aproveitam de vulnerabilidades muitas vezes não conhecidas
  - Defesas focam em detectar e recuperar, ao invés de prevenir

# Serviços de Segurança (RFC 2828)

- De acordo com a RFC 2828:

*“Um serviço provido por um sistema para prover determinado tipo de proteção ao recursos do sistema; **serviços de segurança implementam políticas de segurança e são implementados por mecanismos de segurança.**”*

# Serviços de Segurança (RFC 2828)

- Autenticação
  - Autenticação da contra-parte
  - Autenticação dos dados (garantia da fonte)
- Controle de Acesso
- Confidencialidade
- Integridade
- Não Repúdio (Origem e Destino)
- Disponibilidade

# Mecanismos de Segurança

## (RFC 2828)

- Cifragem
  - Dados não legíveis
  - Algoritmo e 0 ou mais chaves
- Assinatura Digital
  - Prova de fonte
  - Prova de integridade
  - Proteção quanto a forjamento
- Controle de Acesso
  - Mecanismos no servidor

# Mecanismos de Segurança (RFC 2828)

- Controle de Integridade
- Autenticação
  - De entidades
  - Mútua
    - A autentica-se perante B
    - B autentica-se perante A
- “Padding” de trafego
  - Preenchimento de lacunas com informação inútil
  - Contra análise de trafego

# Mecanismos de Segurança (RFC 2828)

- Controle de roteamento
  - Seleciona rotas físicas seguras para determinados dados
  - Mudança de rota quando há suspeita de ameaça
- Notarização
  - Terceira parte confiável

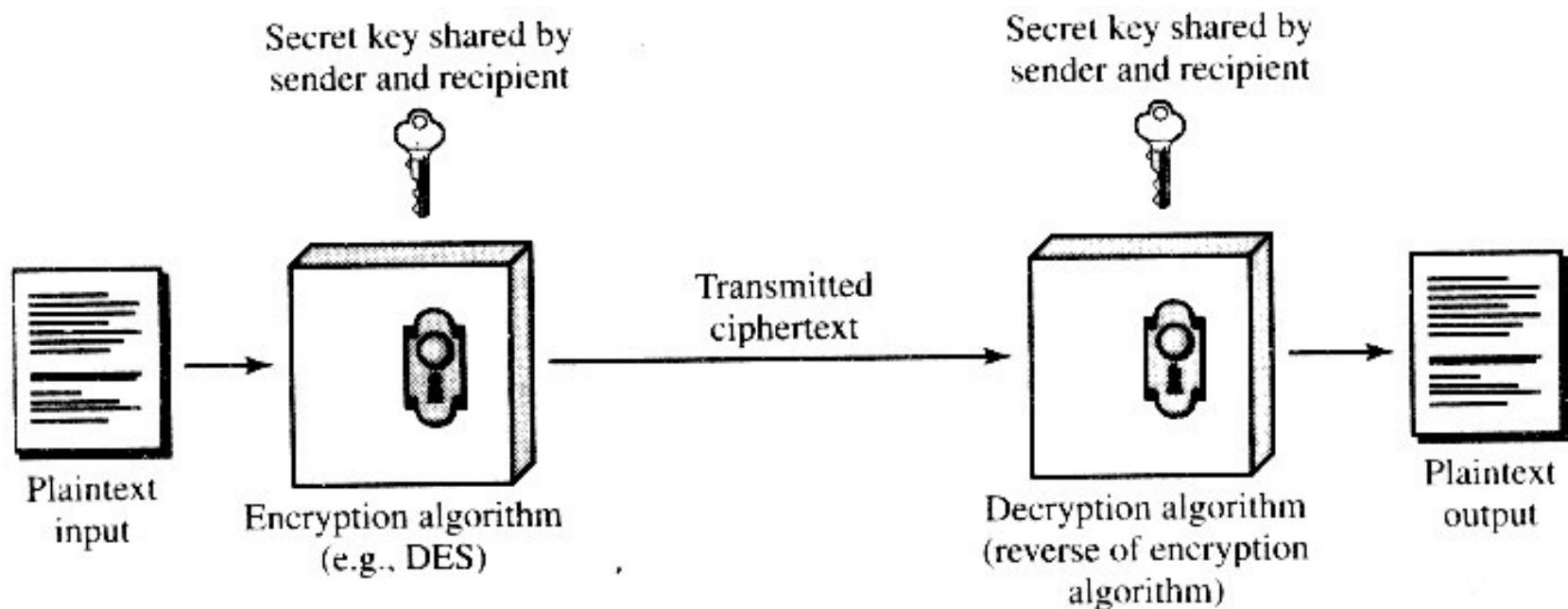
# Cifragem – Técnicas Clássicas

- Modelo de Cifragem Simétrica
- Técnicas de Substituição
- Técnicas de Transposição
- Maquinas de Rotores
- Esteganografia

# Modelo de Cifragem Simétrica

- Elementos:
  - Texto Claro
  - Algoritmo de Cifração
  - Chave Secreta
  - Texto Cifrado
  - Algoritmo de Decifração
- Algoritmo não é secreto
- Somente conhecendo a chave para conseguir obter informações

# Modelo de Cifragem Simétrica



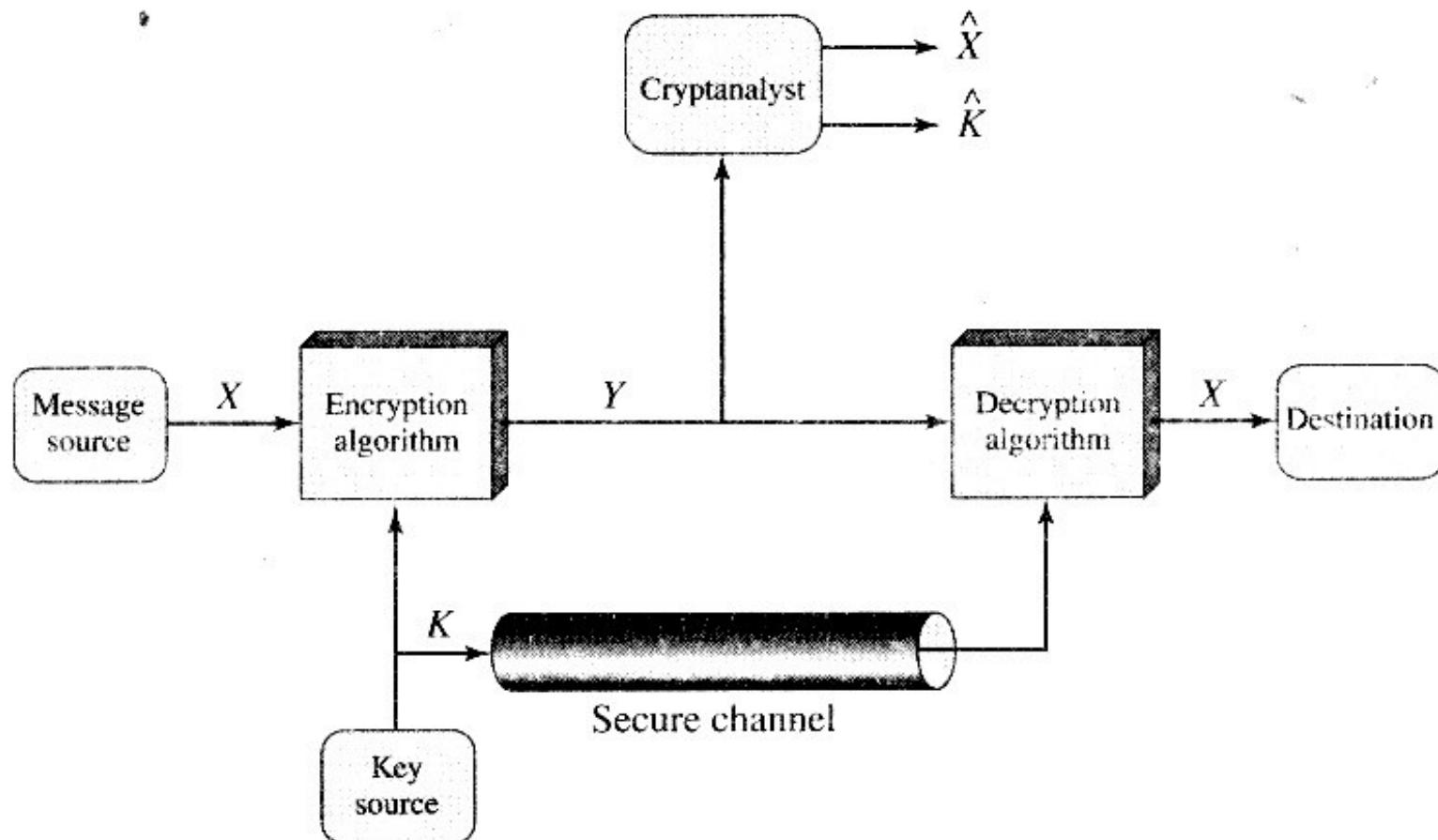
# Criptografia x Criptoanálise

- Criptografia
  - Operações no texto claro para texto cifrado
    - Substituição de elementos (mapeamento para outros)
    - Transposição - embaralhamento
  - Número de Chaves
    - Mesma chave – simétrica
    - Chaves diferentes - assimétrica
  - A forma como o texto claro é processado
    - Bloco
    - Fluxo

# Criptografia x Criptoanálise

- Criptoanálise
- Normalmente o objetivo é recuperar a chave
  - Ataque na natureza do algoritmo
    - Características do texto (claro e cifrado)
  - Força Bruta
    - Em média é necessário tentar metade das possíveis chaves antes de suceder

# Modelo de Criptosistema simétrico



# Criptoanálise

Tipo de Ataque	Acessível ao Criptoanalista
Texto Cifrado Somente	Algoritmo, texto cifrado
Texto Claro Conhecido	Algoritmo Pares texto claro - texto cifrado
Texto Claro Escolhido	Algoritmo Pares texto claro - texto cifrado Texto claro escolhido
Texto Cifrado Escolhido	Algoritmo Pares texto claro-texto cifrado Texto cifrado escolhido
Texto Escolhido	Algoritmo Pares texto claro-texto cifrado Texto claro escolhido Texto cifrado escolhido

# Incondicionalmente Seguro

X

# Computacionalmente Seguro

- Incondicional
  - Texto cifrado não contém informação suficiente para determinar o texto claro
- Computacional
  - Custo de quebrar excede o valor do conteúdo
  - O tempo requerido é maior que a vida útil do conteúdo

# Incondicionalmente Seguro

X

## Computacionalmente Seguro

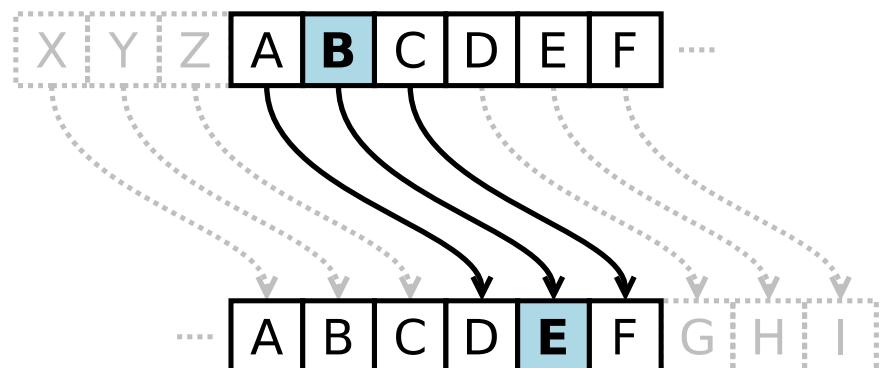
Key size (bits)	Number of alternative keys	Time required at 1 decryption/ $\mu$ s	Time required at $10^6$ decryption/ $\mu$ s
32	$2^{32} = 4.3 \times 10^9$	$2^{31} \mu\text{s} = 35.8$ minutes	2.15 milliseconds
56	$2^{56} = 7.2 \times 10^{16}$	$2^{55} \mu\text{s} = 1142$ years	10.01 hours
128	$2^{128} = 3.4 \times 10^{38}$	$2^{127} \mu\text{s} = 5.4 \times 10^{24}$ years	$5.4 \times 10^{18}$ years
168	$2^{168} = 3.7 \times 10^{50}$	$2^{167} \mu\text{s} = 5.9 \times 10^{36}$ years	$5.9 \times 10^{30}$ years
26 characters (permutation)	$26! = 4 \times 10^{26}$	$2 \times 10^{25} \mu\text{s} = 6.4 \times 10^{12}$ years	$6.4 \times 10^6$ years

# Técnicas de Substituição

- Cifrador de Cezar
- Cifradores Mono-alfabéticos
- Playfair
- Cifradores Poli-alfabéticos
- Cifrador de Viginère
- Cifrador de Vernam
- One-time pad

# Cifrador de Cesar

- Claro: Me encontre depois da aula
- Cifrado: PH HQFRQWUH GHSRLV GD DXOD
- $C = ( p + 3 ) \text{ mod } 26$
- Chave = 3
- Criptoanálise:
  - Força Bruta
    - 25 chaves para tentar



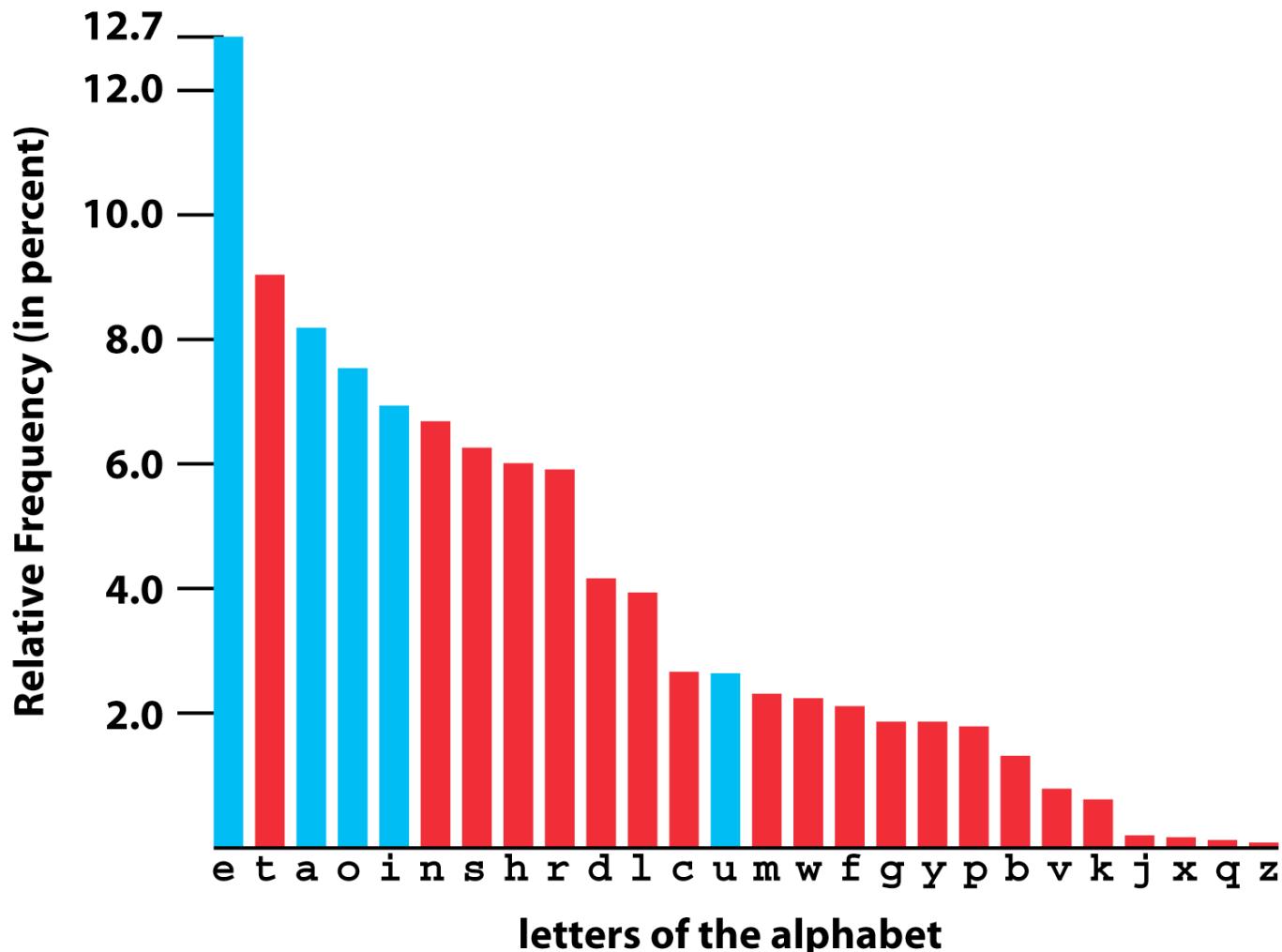
# Cifrador de Cesar

- Características que permitem o uso de força bruta:
  - Os algoritmos de cifração e decifração são conhecidos
  - Somente 25 possíveis chaves
  - Linguagem do texto claro conhecida e facilmente reconhecida

# Cifradores Mono-alfabéticos

- Mapeia de um alfabeto para outro alfabeto
- Troca de uma letra por outra letra qualquer
- Espaço de Chaves:
  - $26! > 4 \times 10^{26}$
  - Maior que DES
- Criptoanálise:
  - Análise de freqüência
  - Analise de duplas, triplas

# Freqüênci Relativa das Letras



# Playfair

- Cifra pares de letras
- Mesma linha, coluna do par – CH -> AK
- Pares na mesma linha → Direita
- Pares na mesma coluna → Abaixo
- Esconde digramas (análise de freq. mais difícil)

S	E	G	U	R
O	A	B	C	D
F	H	I/J	K	L
M	N	P	Q	T
V	W	X	Y	Z

# Cifradores Poli-Alfabéticos

- Usam um conjunto de substituições monoalfabéticas
- Uma chave determina como a transformação é dada
- Ofusca as informações de freqüência
- Nem toda a estrutura é perdida

# Cifrador de Viginère

- Chave: segurosegurosegu
- Claro: aulanosabadoebom
- Cifrado: SYRUECJEHUUWFUG
- Ataque:
  - Determinar o tamanho da chave
  - Distância da repetição no texto cifrado



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
A	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z
B	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A
C	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B
D	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C
E	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D
F	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E
G	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F
H	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G
I	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H
J	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I
K	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J
L	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K
M	M	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L
N	N	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M
O	O	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N
P	P	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O
Q	Q	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P
R	R	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q
S	S	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R
T	T	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S
U	U	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
V	V	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
W	W	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V
X	X	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W
Y	Y	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
Z	Z	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X	Y

# Cifrador de Veginère

- Chave: deceptive
- Texto: We are dicovered, save yourself

deceptivedeceptivedeceptive  
wearediscoveredsaveyourself  
ZICVTWQNGRZGVTAVZHC...

- Chave de tamanho 3 ou 9

# Cifrador de Vernam

- Transformação do texto em bits
- Transformação da chave em bits
- Ou-Exclusivo bit a bit
- $C_i = P_i \oplus K_i$
- Ataque:
  - Análise de frequência não funciona
  - Alfabeto pequeno para fazer inferências

# One-Time Pad

- Chave de igual tamanho ao texto claro
- Chave verdadeiramente aleatória
- Incondicionalmente seguro
- Cifrador de Viginère:

ciphertext: ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUFPLUYTS

key: pxlmvmsydoфuyrvzwc tnlebnecvgdupahfzzlmnyih

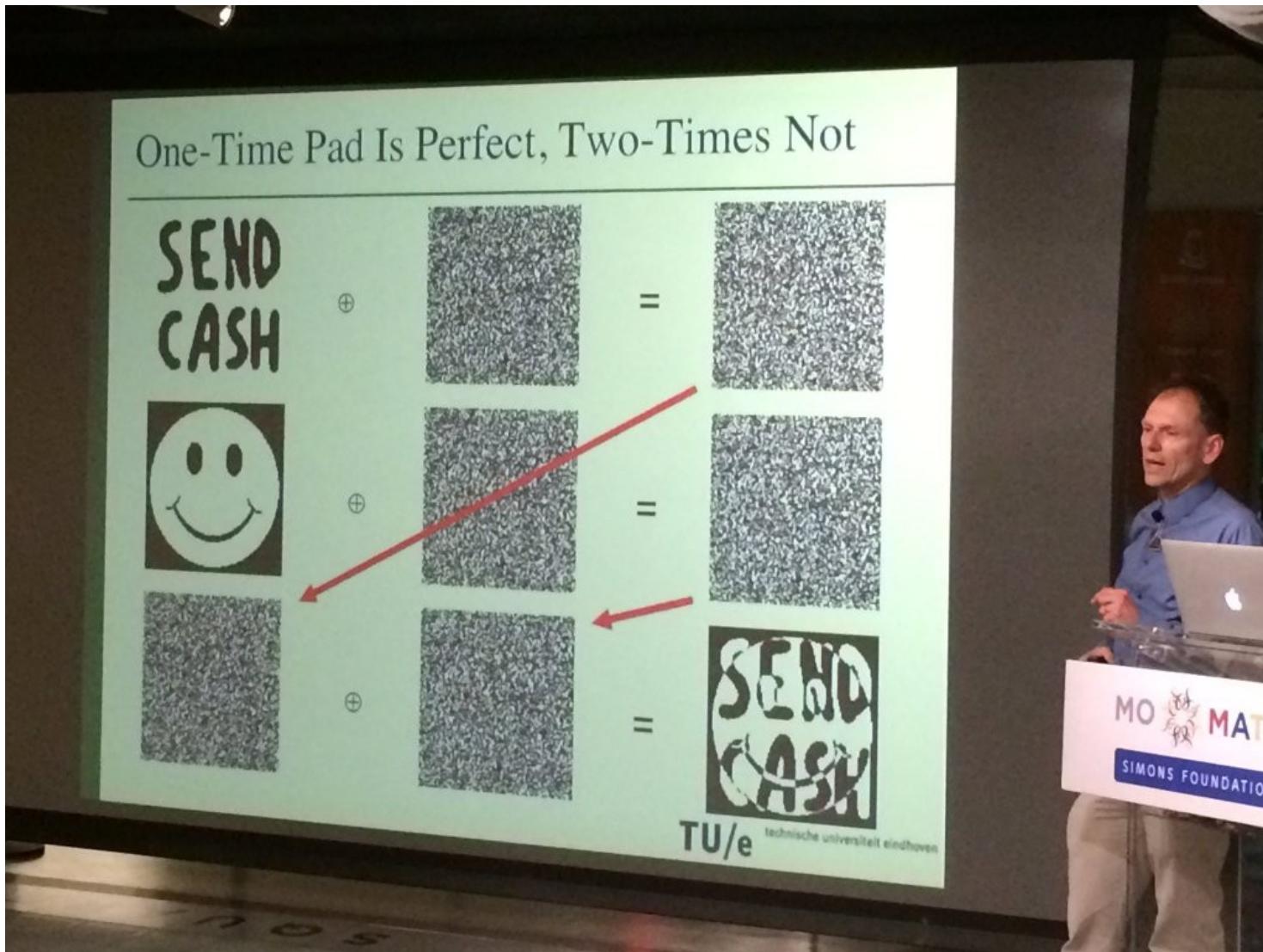
plaintext: mr mustard with the candlestick in the hall

ciphertext: ANKYODKYUREPFJBYOJDSPLREYIUNOFDOIUFPLUYTS

key: mfugpmiydgaxgoufhkllmhsqdqogtewbqfgoyuhwt

plaintext: miss scarlet with the knife in the library

# One-Time Pad (@MoMath1)



# Técnicas de Transposição

- Permutação no texto claro
- C i t g a i e a i
- R p o r f a f c l
- Matriz escrita em linha e recuperada em colunas
  - Chave pode ser a ordem das colunas
- Varias permutações confundem a Criptoanálise

# Técnicas de Transposição

4 3 1 5 2  
E S T A E  
U M A A U  
L A D E S  
E G U R A  
N C A X X



Cítala grega

TADUAEUSAXEMAGCEULENAAAERX

# Técnicas de Transposição

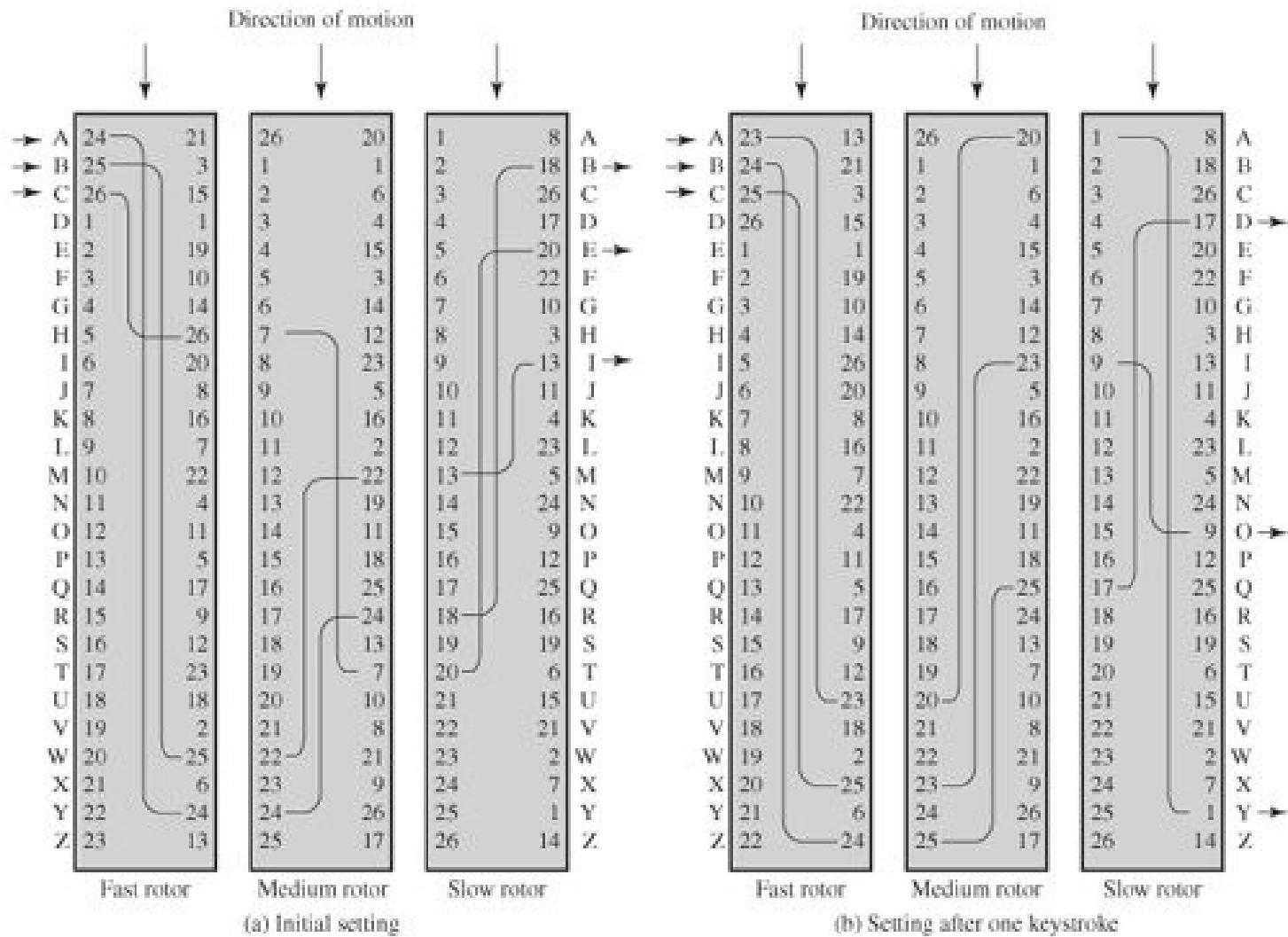
- Mesma frequência do texto claro
  - Fácil de reconhecer
- Fácil de reverter se for somente uma permutação
- Transposição do texto cifrado
  - Realizar a transposição de novo com a mesma chave

# Maquinas de Rotores

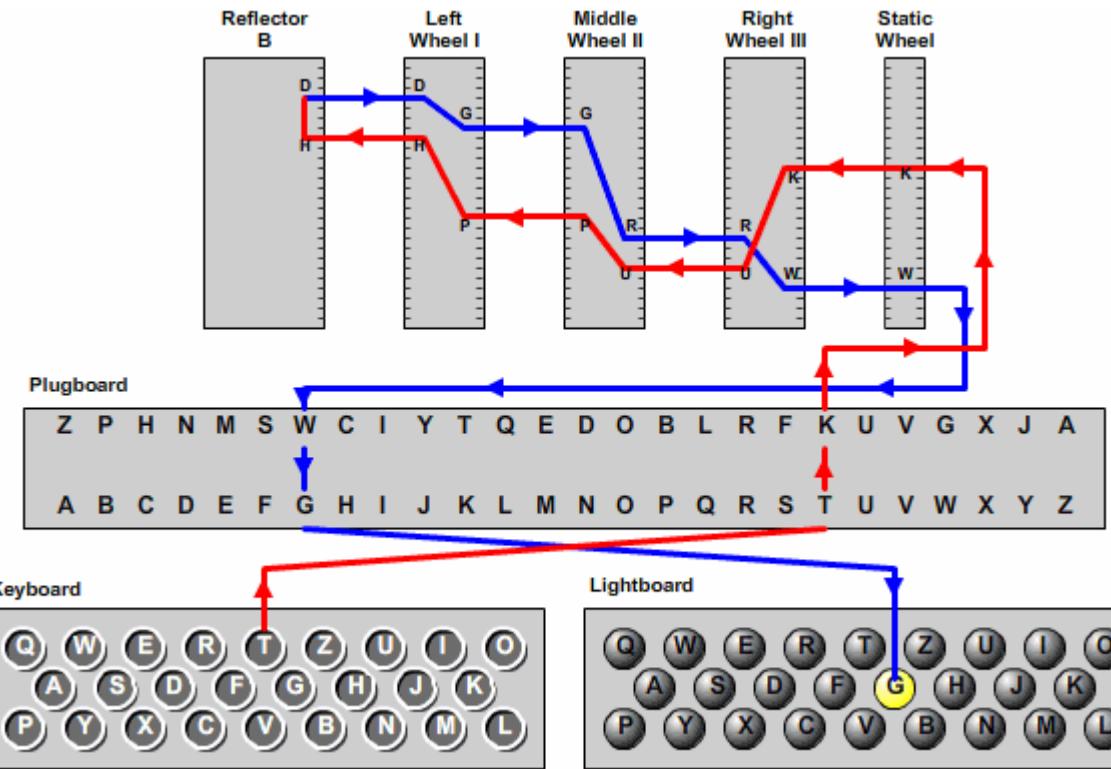
- Sistema eletro-mecânico
- Conjunto de cilindros independentes
- Cada cilindro um cifrador mono-alfabético
- Chave:
  - Posição inicial dos rotores, posição do alfabeto, retroalimentação



# Maquinas de Rotores



# Maquinas de Rotores



© 2006, by Louise Dade

# Maquinas de Rotores



# Maquinas de Rotores

- Cada volta completa do primeiro rotor faz o rotor do meio gira um pino
- Cada volta completa do rotor do meio faz o último rotor girar um pino

$$26 \times 26 \times 26 = 17.576 \text{ substituições}$$

# Esteganografia

- Mensagem escondida em mídia portadora
- Objetivo: Repúdio do Envio
- Técnicas clássicas:
  - Marcação de caracteres
  - Tinta invisível
- Técnicas Modernas
  - Imagens
  - Audio

# Esteganografia

- A imagem escondida foi obtida através dos dois últimos bits de cada componente de cor



# Esteganografia

- Qual é a mensagem escondida?

3rd March

Dear George,

Greetings to all at Oxford. Many thanks for your letter and for the Summer examination package. All Entry Forms and Fees Forms should be ready for final despatch to the Syndicate by Friday 20th or at the very latest, I'm told, by the 21st. Admin has improved here, though there's room for improvement still; just give us all two or three more years and we'll really show you! Please don't let these wretched 16+ proposals destroy your basic O and A pattern. Certainly this sort of change, if implemented immediately, would bring chaos.

Sincerely yours,

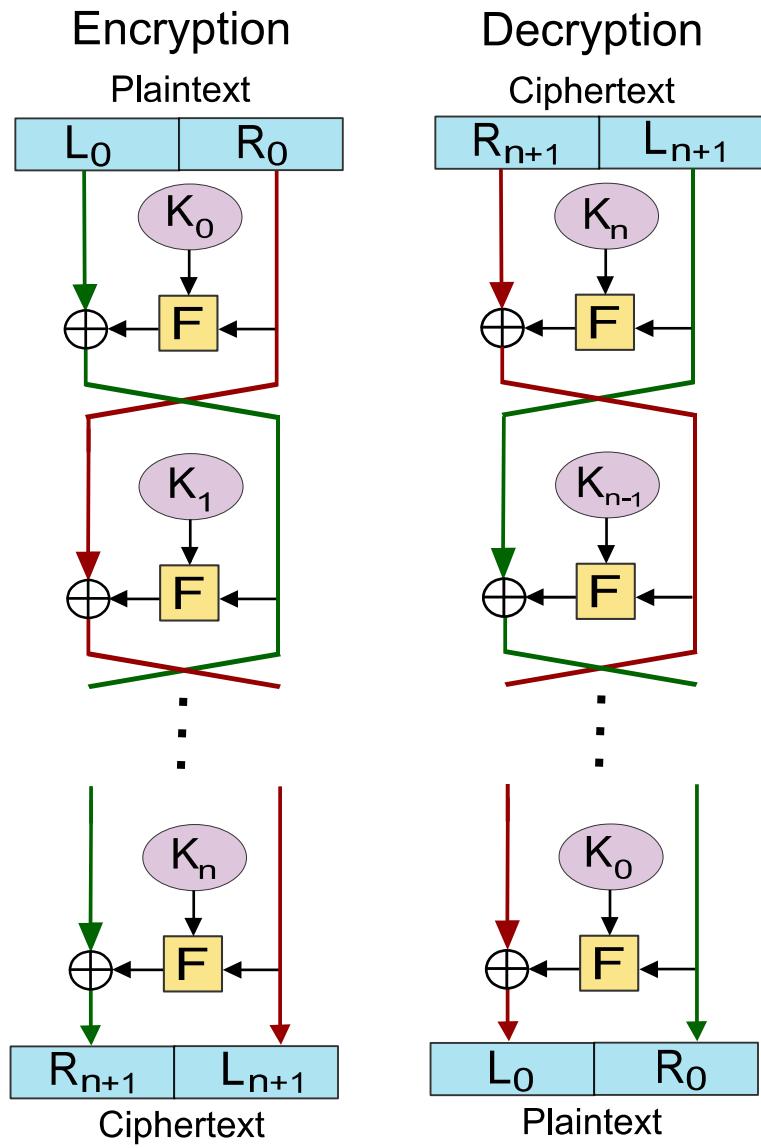
# Cifradores Simétricos

- Mesma chave para cifrar e decifrar
- 2 categorias:
  - Bloco
    - Considera um bloco como um todo
    - Cifradores de bloco ideais são impraticáveis
    - Feistel propôs componentes implementáveis
  - Stream
    - Bit a bit / Byte a byte
    - Minoria

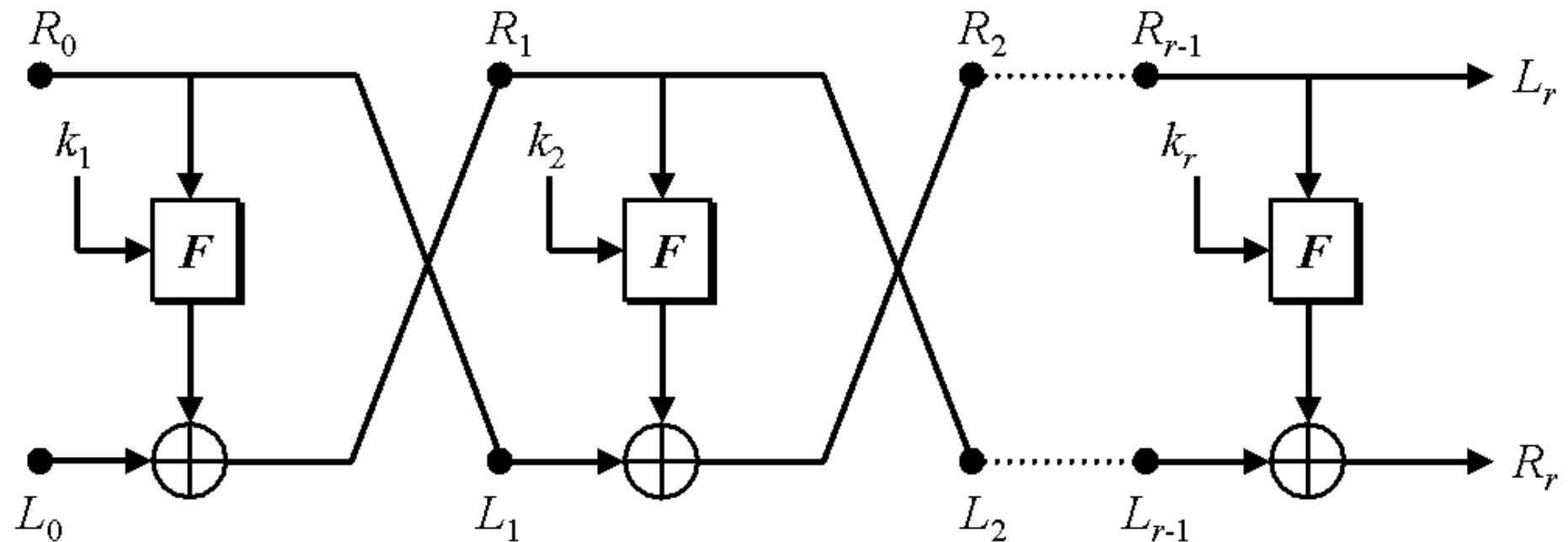
# Confusão x Difusão

- Proposto por Claude Shannon (1945)
- Confusão:
  - Complexidade da relação texto cifrado x chave
  - Protege a chave
  - Substituição deve ser complexa
- Difusão:
  - Dissipação da estrutura estatística
  - 1 dígito de entrada afeta n dígitos de saída
  - Dissimula freqüência do texto claro

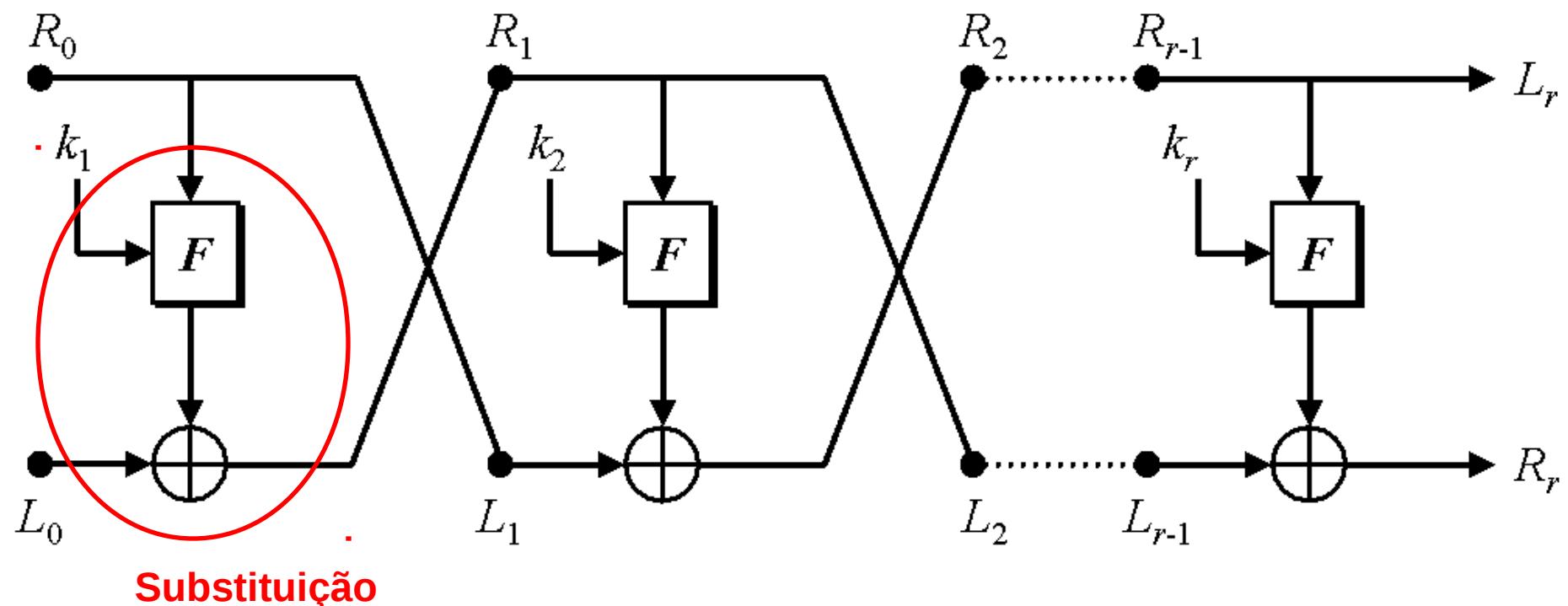
# Redes de Feistel



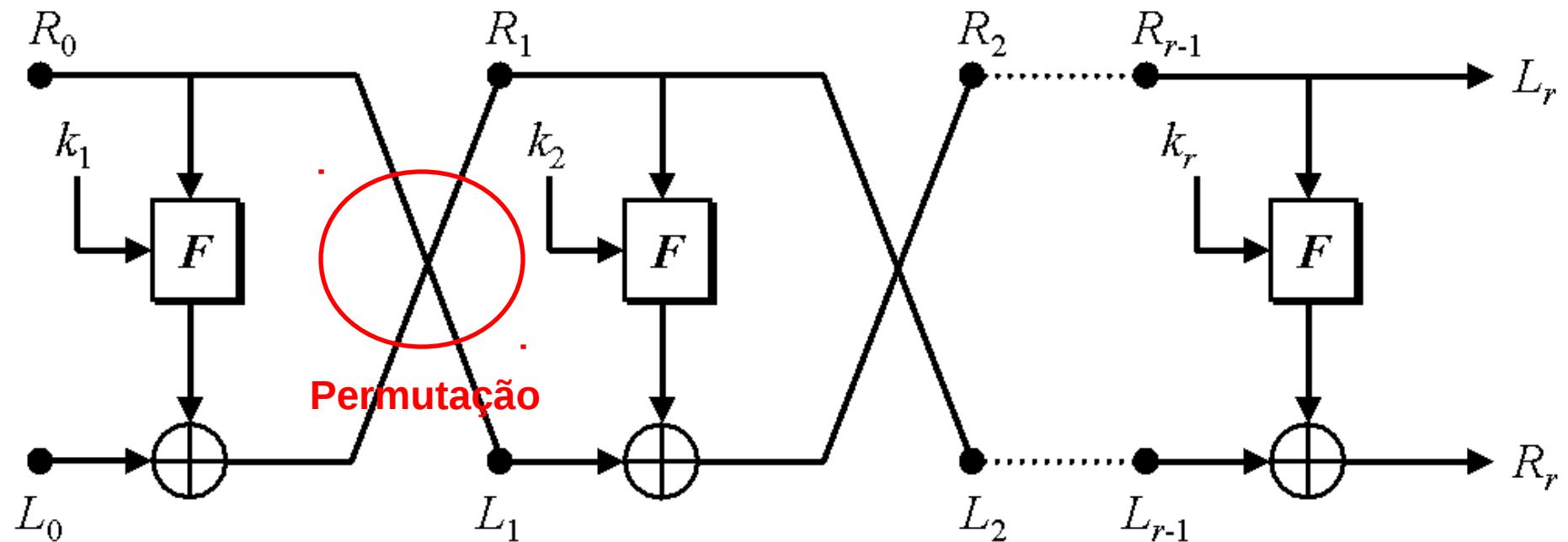
# Redes de Feistel



# Redes de Feistel



# Redes de Feistel



# Design de Redes de Feistel

- Tamanho de Bloco
  - ↑ Tamanho ↑ Segurança
  - ↑ Tamanho ↓ Velocidade
  - Relacionado com a difusão
  - 64 bits em média
- Tamanho de Chave
  - Mesmas características do bloco
  - Relacionado com a confusão
  - 128 bits pelo menos

# Design de Redes de Feistel

- Número de rodadas
  - ↑ Rodadas ↑ Segurança
  - Normalmente 16
- Geração de sub-chaves
  - ↑ Complexidade ↑ Dificuldade na criptoanalise
- Função de rodada
  - Mesma característica da geração de chaves

**Velocidade X Segurança**

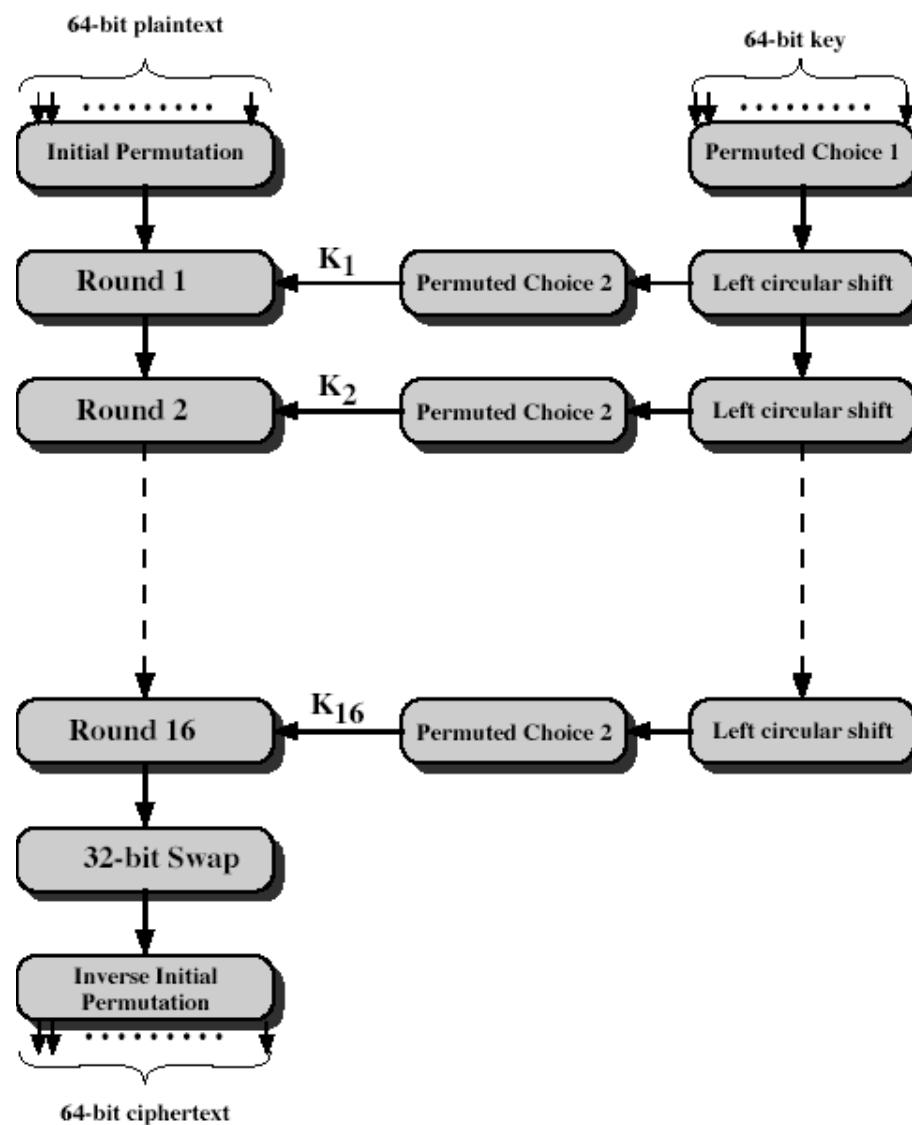
# Redes de Feistel

- $Re_i = Le_{i-1} \text{ xor } F(Re_{i-1}, K_i)$
- $Le_i = Re_{i-1}$
- Cifragem e Decifragem com o mesmo algoritimo (chaves em ordem inversa)
- Pelo menos 3 rodadas para começar a ter difusão e confusão

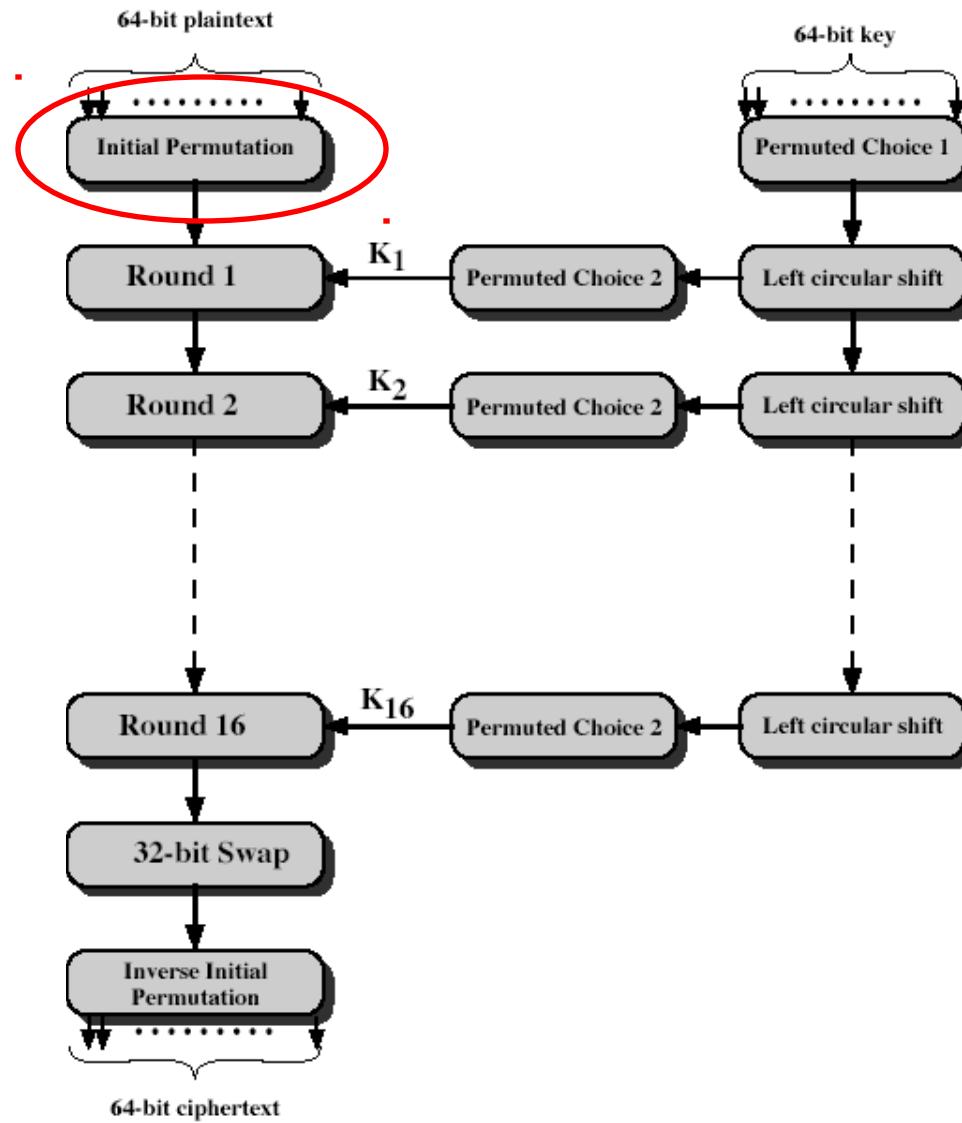
# DES – Data Encryption Standard

- Federal Information Processing Standard 46 (FIPS PUB 46) [1977]
- IBM Lucifer [1971]
- Baseado em rede de Feistel
- Chave de 56 bits (para caber em um chip)
- Blocos de 64 bits
- Ótima implementação em Hardware

# DES Ilustrado



# DES Ilustrado



# DES – Permutação inicial

(a) Initial Permutation (IP)

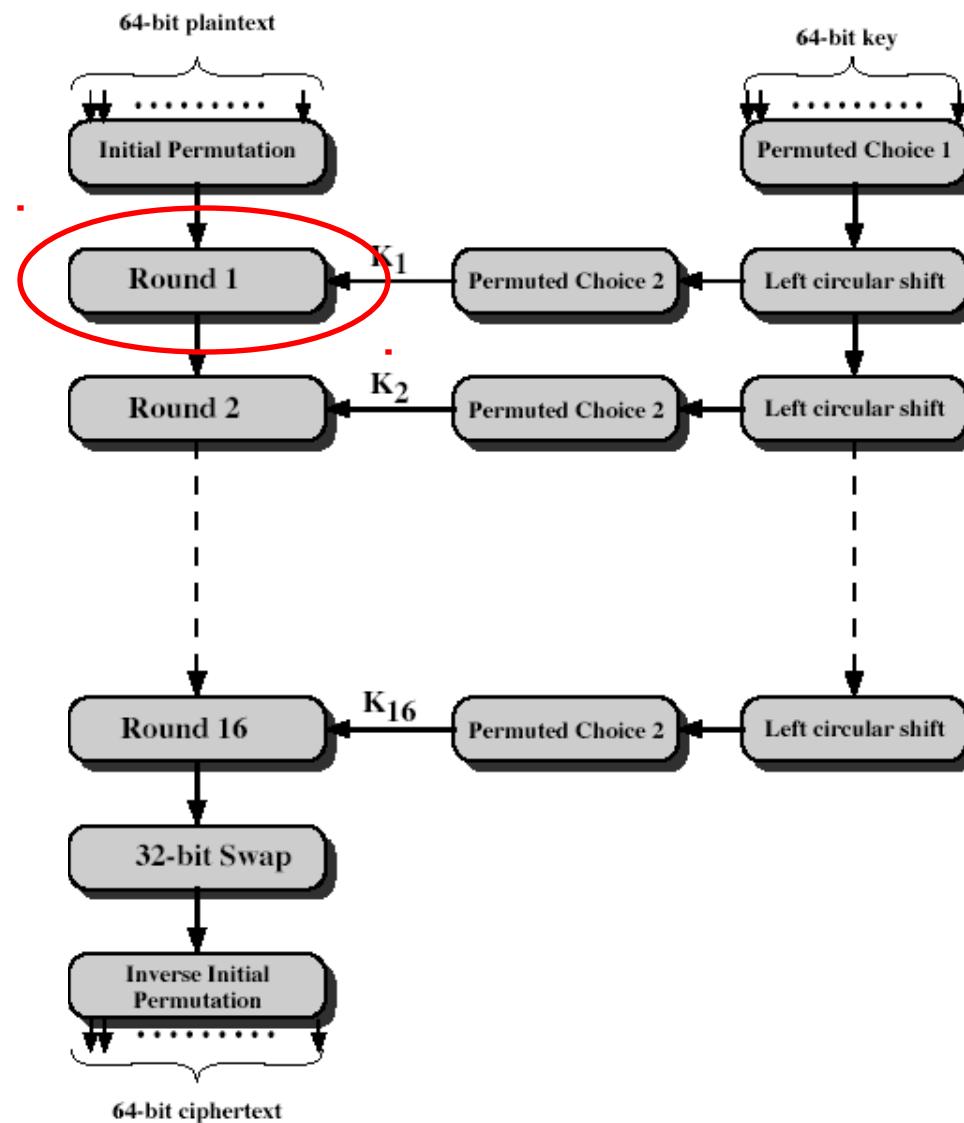
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(b) Inverse Initial Permutation ( $IP^{-1}$ )

40	8	48	16	56	24	64	32
39	7	47	15	55	23	63	31
38	6	46	14	54	22	62	30
37	5	45	13	53	21	61	29
36	4	44	12	52	20	60	28
35	3	43	11	51	19	59	27
34	2	42	10	50	18	58	26
33	1	41	9	49	17	57	25

- $IP^{-1}(IP(M))=M$
- Adicionam Difusão

# DES Ilustrado



# DES - Estrutura de 1 Rodada

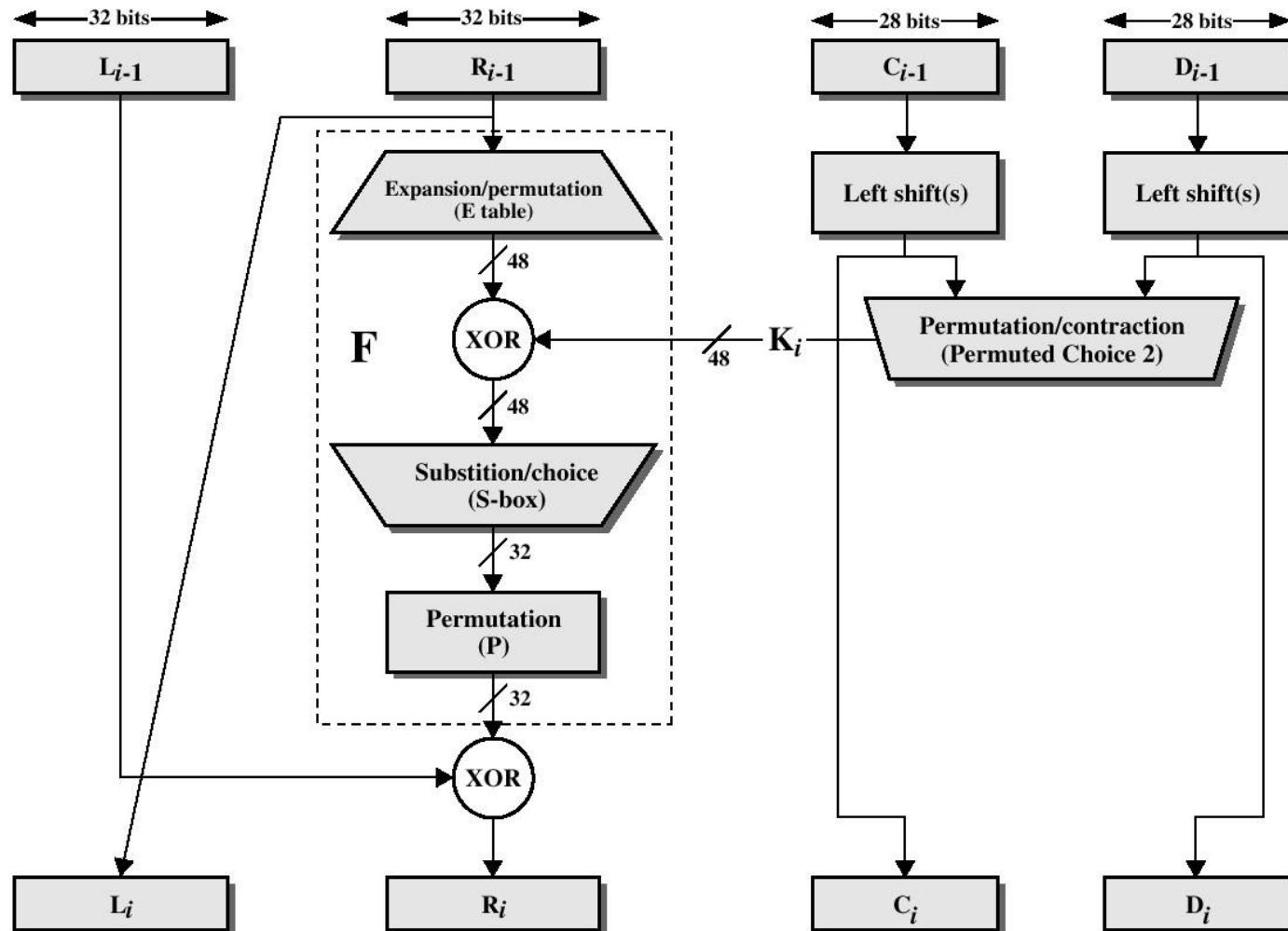


Figure 2.4 Single Round of DES Algorithm

# DES - Estrutura de 1 Rodada

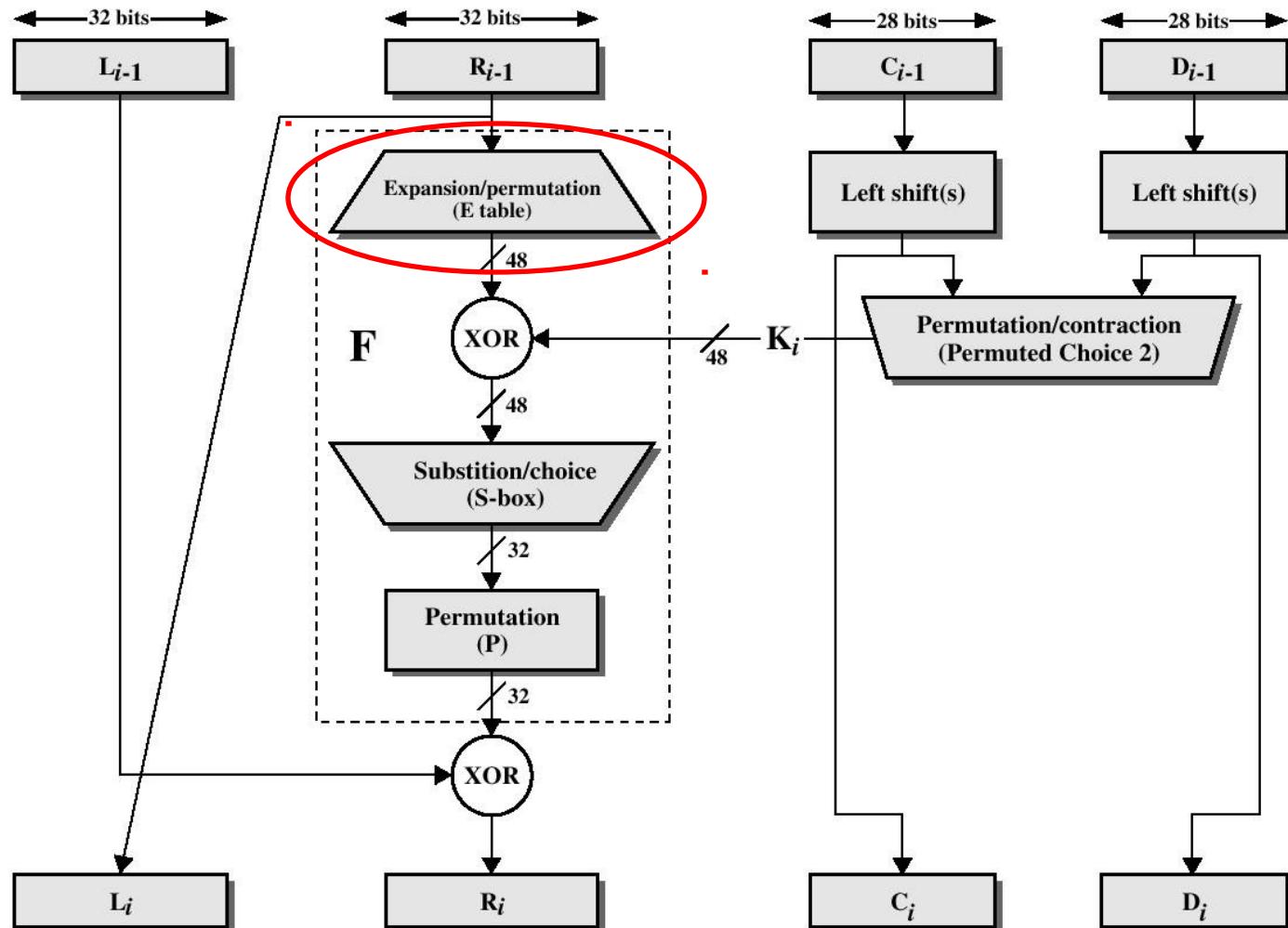


Figure 2.4 Single Round of DES Algorithm

# DES - Permutação de Expansão

**(c) Expansion Permutation (E)**

32	1	2	3	4	5
4	5	6	7	8	9
8	9	10	11	12	13
12	13	14	15	16	17
16	17	18	19	20	21
20	21	22	23	24	25
24	25	26	27	28	29
28	29	30	31	32	1

# DES - Estrutura de 1 Rodada

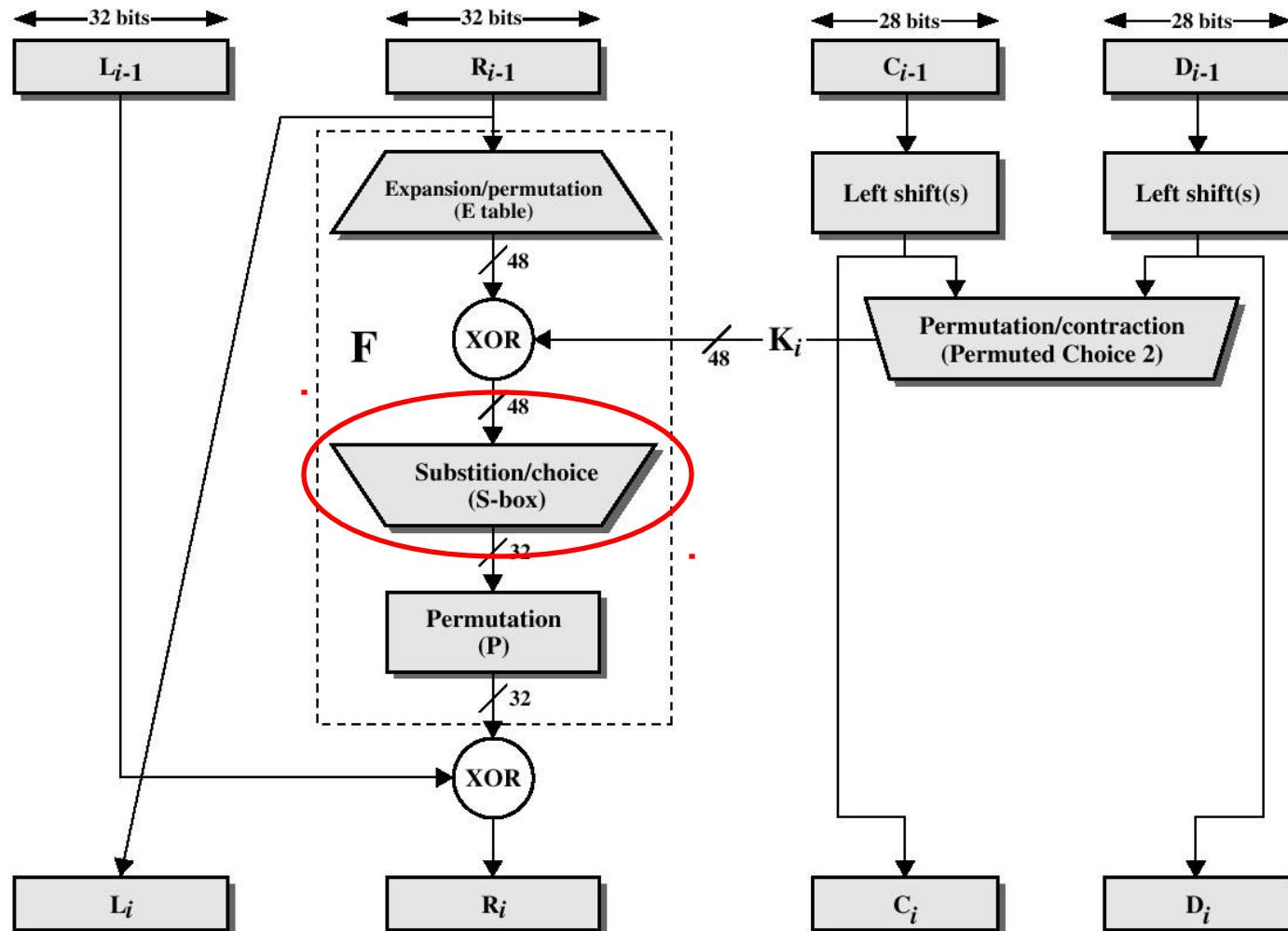


Figure 2.4 Single Round of DES Algorithm

# DES – Caixas S

Table 3.3 Definition of DES S-Boxes

S <sub>1</sub>	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13

	2	12	4	1	7	10	11	6	8	5	3	15	13	0	14	9
	14	11	2	12	4	7	13	1	5	0	15	10	3	9	8	6
	4	2	1	11	10	13	7	8	15	9	12	5	6	3	0	14
	11	8	12	7	1	14	2	13	6	15	0	9	10	4	5	3

S <sub>2</sub>	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9

	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

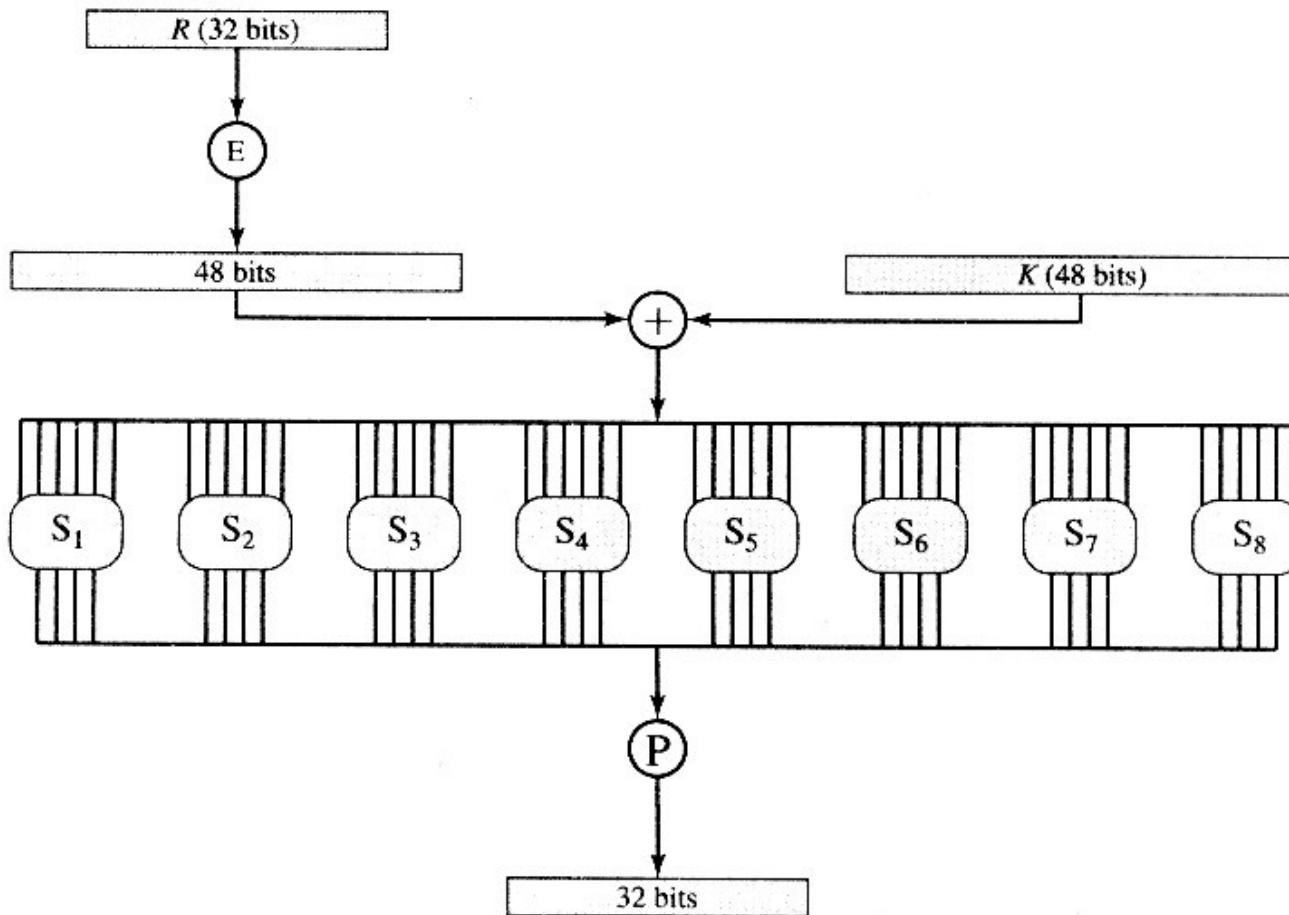
S <sub>3</sub>	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12

	4	11	2	14	15	0	8	13	3	12	9	7	5	10	6	1
	13	0	11	7	4	9	1	10	14	3	5	12	2	15	8	6
	1	4	11	13	12	3	7	14	10	15	6	8	0	5	9	2
	6	11	13	8	1	4	10	7	9	5	0	15	14	2	3	12

S <sub>4</sub>	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

	13	2	8	4	6	15	11	1	10	9	3	14	5	0	12	7
	1	15	13	8	10	3	7	4	12	5	6	11	0	14	9	2
	7	11	4	1	9	12	14	2	0	6	10	13	15	3	5	8
	2	1	14	7	4	10	8	13	15	12	9	0	3	5	6	11

# DES – Função (R,K)



# DES - Estrutura de 1 Rodada

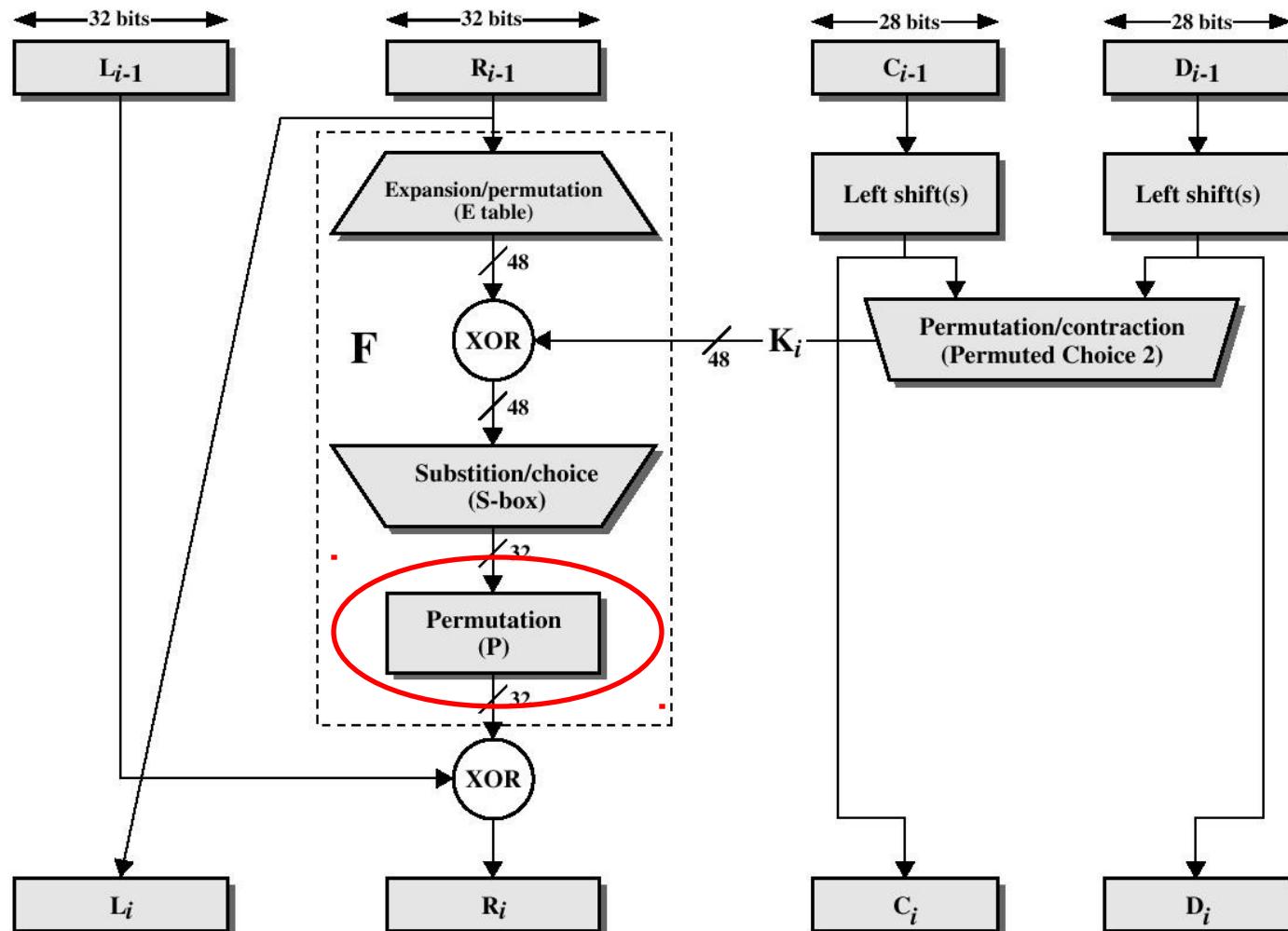


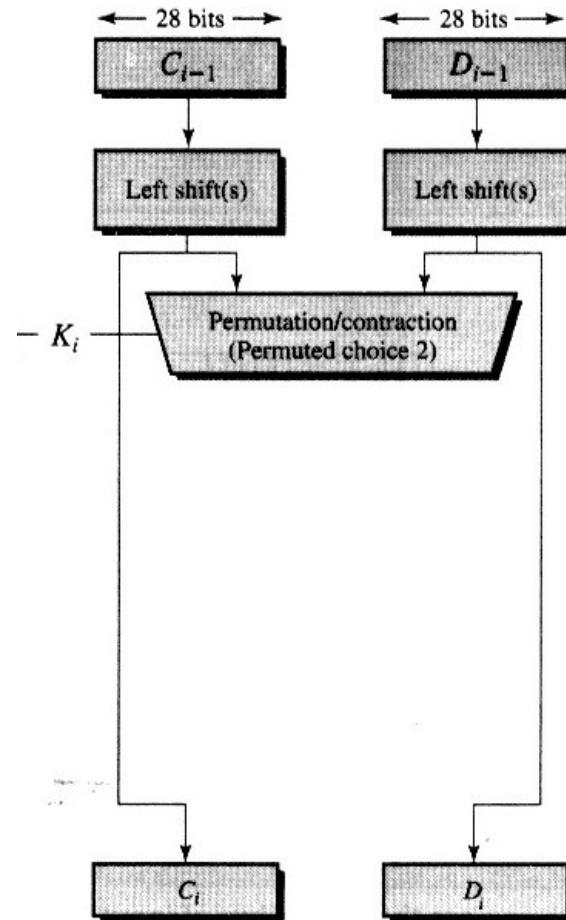
Figure 2.4 Single Round of DES Algorithm

# DES - Função de Permutação

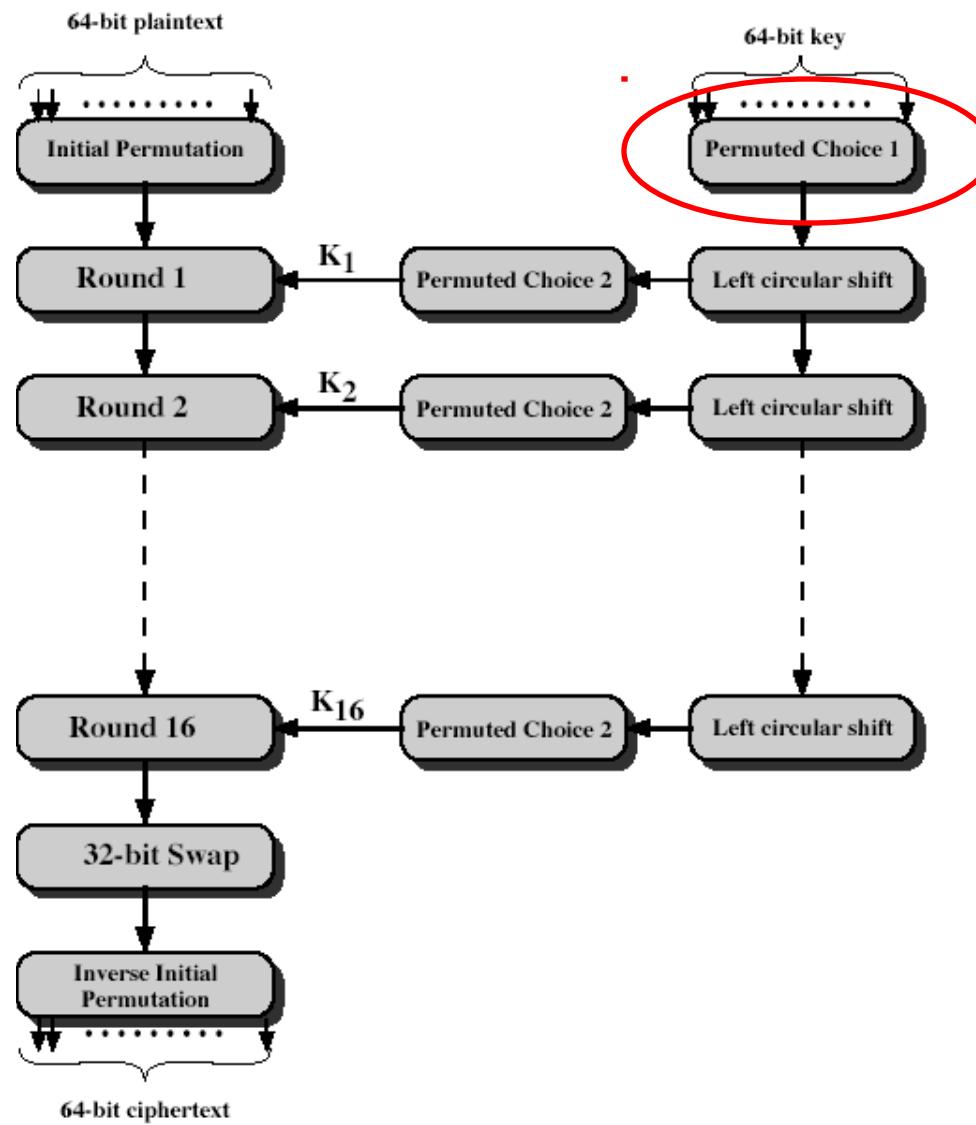
**(d) Permutation Function (P)**

16	7	20	21	29	12	28	17
1	15	23	26	5	18	31	10
2	8	24	14	32	27	3	9
19	13	30	6	22	11	4	25

# DES – Geração de Subchaves



# DES Ilustrado



# DES – Escolha Permutada 1

**(b) Permuted Choice One (PC-1)**

57	49	41	33	25	17	9
1	58	50	42	34	26	18
10	2	59	51	43	35	27
19	11	3	60	52	44	36
63	55	47	39	31	23	15
7	62	54	46	38	30	22
14	6	61	53	45	37	29
21	13	5	28	20	12	4

# DES - Geração de Subchaves

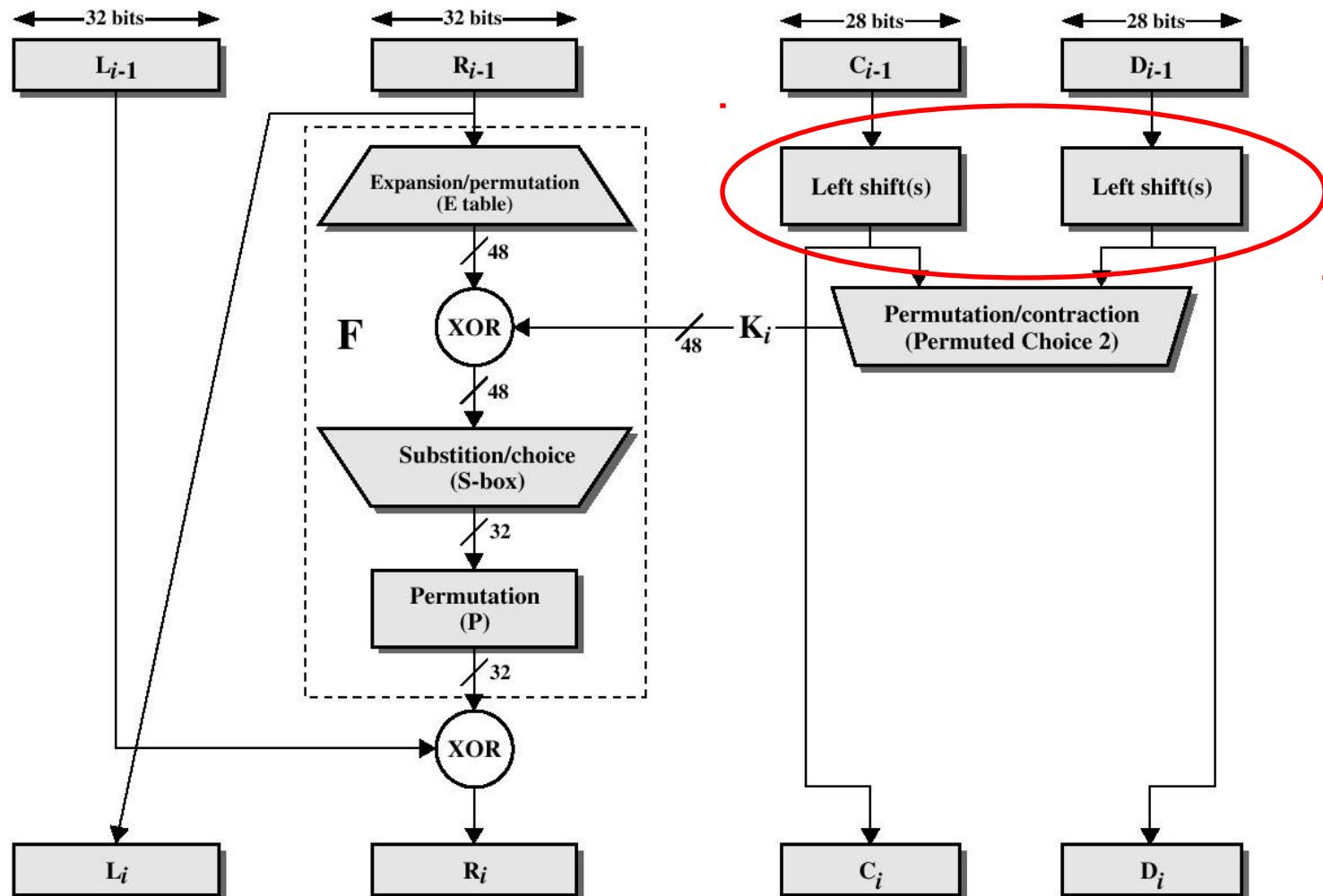


Figure 2.4 Single Round of DES Algorithm

# DES – Tabela de Rotação a Esquerda

**(d) Schedule of Left Shifts**

Round number	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Bits rotated	1	1	2	2	2	2	2	2	1	2	2	2	2	2	2	1

# DES - Geração de Subchaves

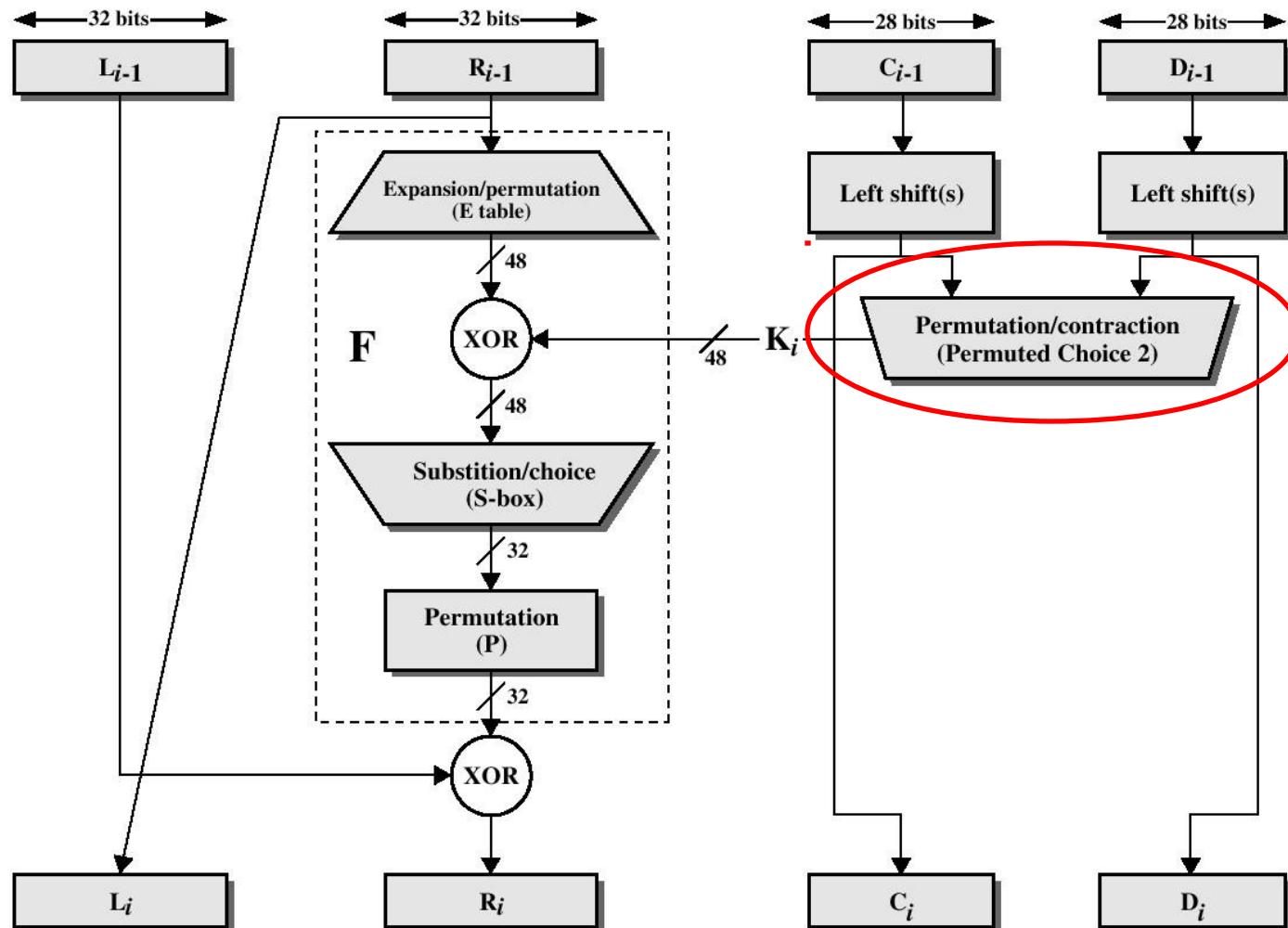


Figure 2.4 Single Round of DES Algorithm

# DES – Escolha Permutada 2

(c) Permuted Choice Two (PC-2)

14	17	11	24	1	5	3	28
15	6	21	10	23	19	12	4
26	8	16	7	27	20	13	2
41	52	31	37	47	55	30	40
51	45	33	48	44	49	39	56
34	53	46	42	50	36	29	32

# Efeito Avalanche no DES

Table 3.5 Avalanche Effect in DES

(a) Change in Plaintext		(b) Change in Key	
Round	Number of bits that differ	Round	Number of bits that differ
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
5	34	5	30
6	32	6	32
7	31	7	35
8	29	8	34
9	42	9	40
10	44	10	38
11	32	11	31
12	30	12	33
13	30	13	28
14	26	14	26
15	29	15	34
16	34	16	35

# DES - Força Criptográfica

- $2^{56} = 7.2 \times 10^{16}$
- 1977 → US\$ 20 Milhões = 10 horas
- 1998 → US\$ 250mil = 70 horas
- Hoje → US\$ 10mil = minutos
- Importante lembrar que é necessário conhecer a natureza do texto claro
- Não foram descobertas ate hoje falhas nas caixas S

# Criptoanálise Diferencial

- Não foi discutido na literatura até 1990
- Ataque no DES por Biham e Shamir (1993)
  - Redução do espaço de busca para  $2^{47}$
- Factível em versões com menos rodadas
- Conhecida pela equipe do LUCIFER em 1974
- Baseado na diferença XOR de dois textos claros
- Probabilidade de bits em caixas S

# Criptoanálise Linear

- Ataque baseado em aproximações lineares
  - Redução do espaço de busca para  $2^{43}$
- Ataque de texto claro escolhido
- Pode ser usado uma rodada de cada vez por ser linear
- Combina os resultados achados em várias rodadas

# Matemática Criptográfica

- Aritmética Modular
- Algoritmo de Euclides
- Grupos, Anéis e Corpos
- Aritmética Polinomial
- Corpos Finitos  $GF(2^n)$

# Aritmética Modular

- a é um inteiro e n é um inteiro positivo
- a mod n é o resto da divisão de a por n
- $A = \lfloor a/n \rfloor \times n + (a \text{ mod } n)$
- Congruência modulo n:
  - $(a \text{ mod } n) = (b \text{ mod } n)$
  - $a \equiv b \pmod{n}$
  - $a \equiv b \pmod{n} \rightarrow b \equiv a \pmod{n}$
  - $a \equiv b \pmod{n}$  e  $b \equiv c \rightarrow a \equiv c \pmod{n}$

# Aritmética Modular - Operações

- $[(a \bmod n) + (b \bmod n)] = (a + b) \bmod n$
- $[(a \bmod n) - (b \bmod n)] = (a - b) \bmod n$
- $[(a \bmod n) \times (b \bmod n)] = (a \times b) \bmod n$
- $(a + b) \equiv (a + c) \bmod n \rightarrow b \equiv c \bmod n$

# Algoritmo de Euclides

- Máximo Divisor Comum
  - c é um divisor de a e b
  - Qualquer divisor de a e b é divisor de c
- $\text{gcd}(a, b) = \text{gcd}(b, a \bmod b)$
- $\text{gcd}(a,b)$ 
  1.  $A \leftarrow a; B \leftarrow b$
  2. If  $B = 0$  return  $A = \text{gcd}(a,b)$
  3.  $R = A \bmod B$
  4.  $A \leftarrow B; B \leftarrow R$
  5. goto 2

# Álgebra Abstrata – Grupos

- $\{G, .\}$  são conjuntos de elementos com uma operação binária denotada por “.” que associa pares ordenados em  $G$ , onde:
  - Fechamento: Se  $a \wedge b \in G \rightarrow a . b \in G$
  - Associatividade:  $a . (b . c) = (a . b) . c$
  - Identidade:  $a . e = e . a = a \quad \forall a \in G$
  - Inverso:  $\exists a' \mid a' . a = a . a' = e$
- Exemplo: inteiros positivos e negativos

# Álgebra Abstrata - Grupos

- Se o numero de elementos é finito, chama-se Grupo finito
- Grupo Abeliano:
  - Comutatividade:  $a \cdot b = b \cdot a \quad \forall a, b \in G$
- Grupo Cíclico:
  - $a^3 = a \cdot a \cdot a$
  - $a^0 = e$
  - $G$  é cíclico se  $\forall x \in G \rightarrow x = a^k \wedge a \in G \wedge k \in \mathbb{N}$
  - Cíclico é sempre Abeliano

# Álgebra Abstrata - Anéis

- $\{R, +, \cdot\}$  é um conjunto de elementos com duas operações binárias chamadas adição e multiplicação para todos  $a$  e  $b$ , onde:
  - $R$  é um grupo Abeliano para adição
  - $R$  tem fechamento sob multiplicação
  - $R$  tem associatividade sob multiplicação
  - $R$  tem distributividade:
    - $a(b+c) = ab + ac \quad \forall a,b,c \in G$
    - $(a+b)c = ac + bc \quad \forall a,b,c \in G$

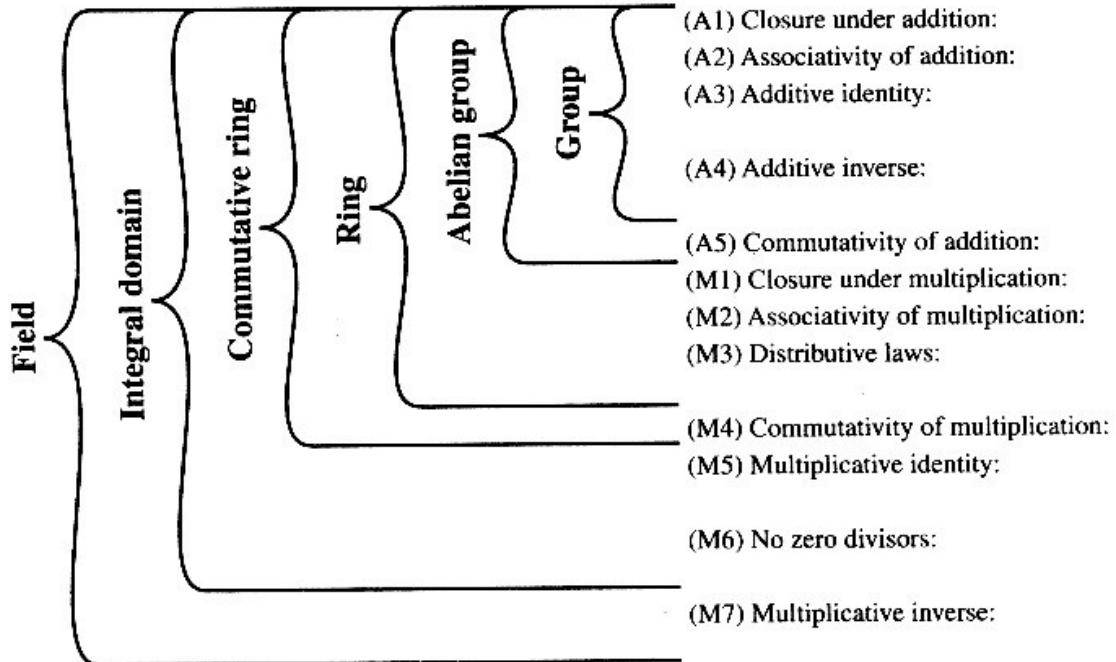
# Álgebra Abstrata - Anéis

- Anel Comutativo:
  - Comutatividade da multiplicação:  $ab = ba$
- Domínio integral:
  - Identidade Multiplicativa:  $a1 = 1a = a$
  - Divisores não-nulos:
    - $Ab = 0$  nem  $a = 0$  ou  $b = 0$
- O conjunto  $Z_n = \{0, 1, \dots, n-1\}$  é um anel comutativo junto com a operação de mod n

# Álgebra Abstrata - Corpos

- $\{F, +, \cdot\}$  é um conjunto de elementos com duas operações binárias chamadas adição e multiplicação tal que:
  - $F$  é um domínio integral
  - $F$  tem multiplicativa inversa:
    - $\forall a \in F$ , exceto 0,  $\exists a^{-1} \in F \mid aa^{-1} = (a^{-1})a = 1$
- Um Corpo é um conjunto onde se pode adicionar, subtrair , multiplicar e dividir sem deixar o conjunto

# Álgebra Abstrata - Relações



If  $a$  and  $b$  belong to  $S$ , then  $a + b$  is also in  $S$   
 $a + (b + c) = (a + b) + c$  for all  $a, b, c$  in  $S$   
There is an element  $0$  in  $R$  such that  
 $a + 0 = 0 + a = a$  for all  $a$  in  $S$   
For each  $a$  in  $S$  there is an element  $-a$  in  $S$   
such that  $a + (-a) = (-a) + a = 0$   
 $a + b = b + a$  for all  $a, b$  in  $S$   
If  $a$  and  $b$  belong to  $S$ , then  $ab$  is also in  $S$   
 $a(bc) = (ab)c$  for all  $a, b, c$  in  $S$   
 $a(b + c) = ab + ac$  for all  $a, b, c$  in  $S$   
 $(a + b)c = ac + bc$  for all  $a, b, c$  in  $S$   
 $ab = ba$  for all  $a, b$  in  $S$   
There is an element  $1$  in  $S$  such that  
 $a1 = 1a = a$  for all  $a$  in  $S$   
If  $a, b$  in  $S$  and  $ab = 0$ , then either  
 $a = 0$  or  $b = 0$   
If  $a$  belongs to  $S$  and  $a \neq 0$ , there is an  
element  $a^{-1}$  in  $S$  such that  $aa^{-1} = a^{-1}a = 1$

# Corpos Finitos GF(p)

- p é um número primo
- $\mathbb{Z}_p\{0,1,\dots,p-1\}$  e as operações são mod p
- Tem multiplicativa inversa, porque p é relativamente primo de todos elementos
- GF(p) tem p elementos
- As operações + e x são definidas no conjunto
  - Adição, subtração, multiplicação e divisão são possíveis
- Exemplo: GF(2) [+ XOR, \* AND], GF(3)

# Aritmética Polinomial

- Polinômio:
  - $f(x) = a_n x^n + a_{n-1} x^{n-1} + \dots + a_1 x + a_0$
- n é o grau do polinômio
- S: conjunto de números formados por  $a_i$
- Aritmética dividida em 3 classes:
  - Ordinária: regras básicas de álgebra
  - Modular: aritméticas no coeficientes mod p
  - Modular Polinomial: mod  $m(x)$ , onde  $m(x)$  é um polinômio irreduzível

# Aritmética Polinomial

- Em álgebra abstrata  $x$  é o indeterminado
  - Não nos interessa resolve-lo
- Aritmética Polinomial inclui adição subtração e multiplicação
- Divisão requer que o conjunto  $S$  seja um corpo
  - Números reais, Racionais,  $\mathbb{Z}_p$

# Aritmética Polinomial

- Adição e Subtração
  - Adiciona-se ou subtrai-se os expoentes
- Multiplicação
  - Multiplica-se os polinômios elemento a elemento
  - Se o grau do resultado for maior ou igual ao grau do polinômio irreduzível, então devemos tirar o módulo

# Aritmética Polinomial

- Exemplo:

- $f(x) = x^6+x^4+x^2+x+1$
- $g(x) = x^7+x+1$
- $m(x) = x^8+x^4+x^3+x+1$
- $f(x) + g(x) = x^7+x^6+x^4+x^2$
- $f(x)^*g(x) \text{ mod } m(x) = x^7+x^6+1$

# Corpos Finitos de Gallois GF( $2^n$ )

- GF(2) é o corpo finito mais simples
  - $+$   $\rightarrow$  XOR ,  $x \rightarrow$  AND
- Polinômio irreduzível
- Usamos um polinômio primo sob o Corpo
- Para algoritmos serem executados em maquinas de base 8 bits usa-se polinômios irreduzíveis de grau 8
- Ex.: GF( $2^3$ )

# AES – Advance Encryption Standard

- Cifrador de bloco para substituir o DES
- Competição em 2001, Chamada em 1997
  - 21 algoritmos, 15 candidatos, 5 finalistas, Rijndael vencedor
- Suporte a 128, 192 e 256 bits
- Não usa Feistel, processa o bloco inteiro
- Rounds:
  - Substituição de bytes, permutação, operação sobre corpo finito, e XOR com a chave

# AES – Critérios de Avaliação Inicial

- Segurança:
  - Esforço comprovado para ataque de criptoanálise
- Custo Computacional:
  - Eficiente em software e hardware. Links de alta velocidade
- Características do Algoritmo:
  - Flexibilidade, compatibilidade e simplicidade

# AES – Critérios Intermediários

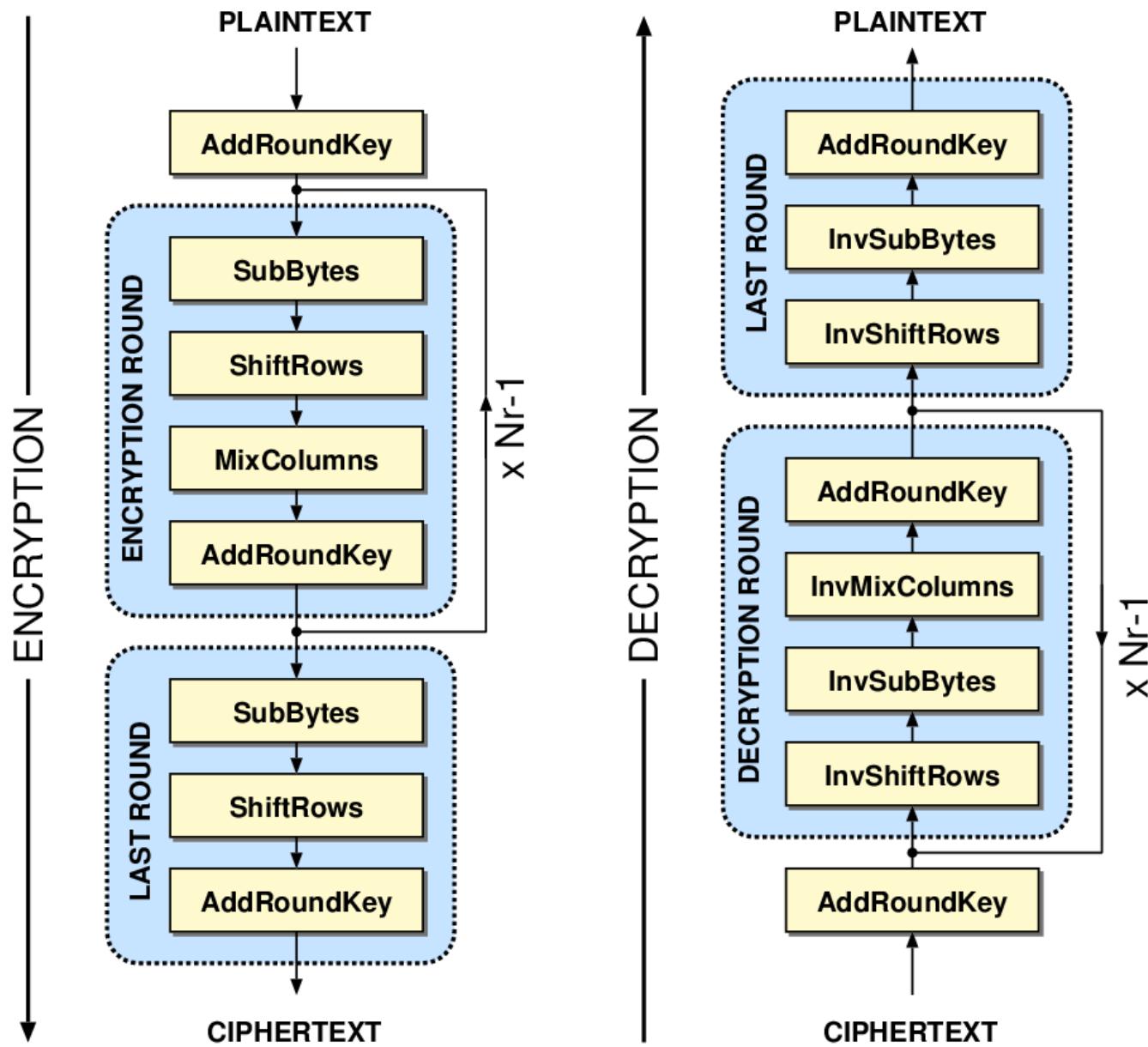
- Segurança verificada pela comunidade
- Implementação em Software
- Restrições de espaço e implementação em hardware
- Ataques nas implementações
- Cifragem x Decifragem
- Agilidade de chaves
- Paralelismo em nível de instrução

# AES - Cifrador

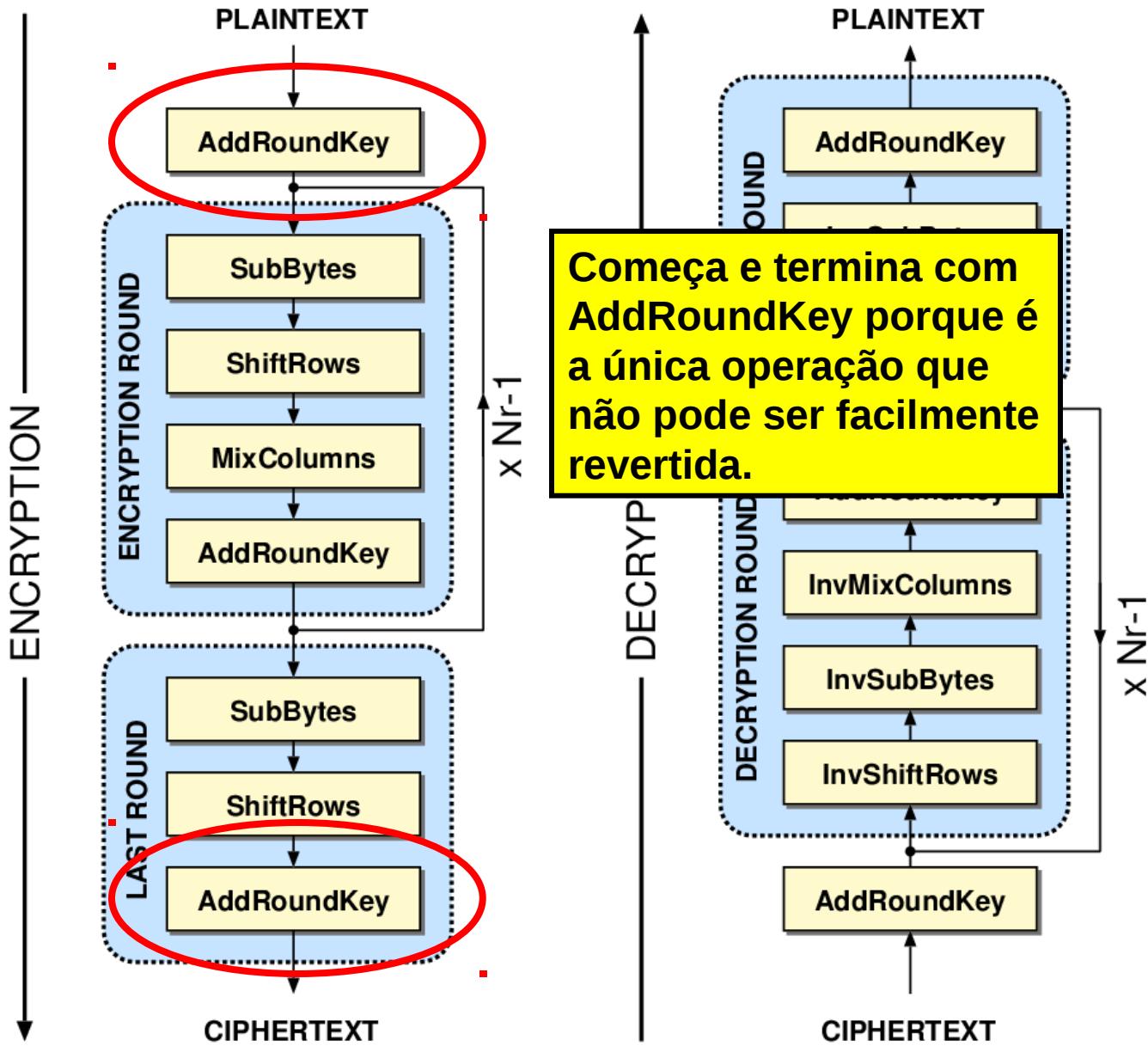
- Tamanho de bloco sempre 128 bits
- Tamanho de Chave Variável (128,192,256)
- Rijndael
  - Resistência a ataques conhecidos
  - Velocidade e tamanho em variadas plataformas
  - Simplicidade

<b>Key size (words/bytes/bits)</b>	4/16/128	6/24/192	8/32/256
<b>Plaintext block size (words/bytes/bits)</b>	4/16/128	4/16/128	4/16/128
<b>Number of rounds</b>	10	12	14
<b>Round key size (words/bytes/bits)</b>	4/16/128	4/16/128	4/16/128
<b>Expanded key size (words/bytes)</b>	44/176	52/208	60/240

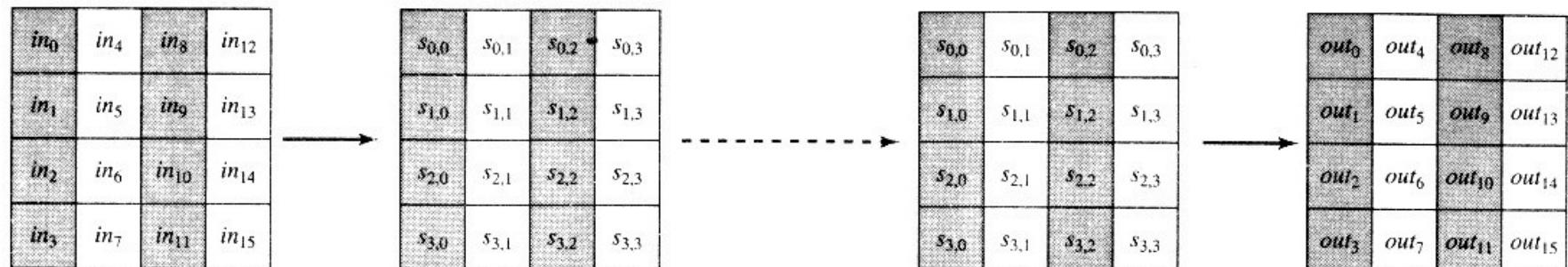
# AES - Cifragem e Decifragem



# AES - Cifragem e Decifragem



# AES - Estrutura de Dados



(a) Input, state array, and output



(b) Key and expanded key

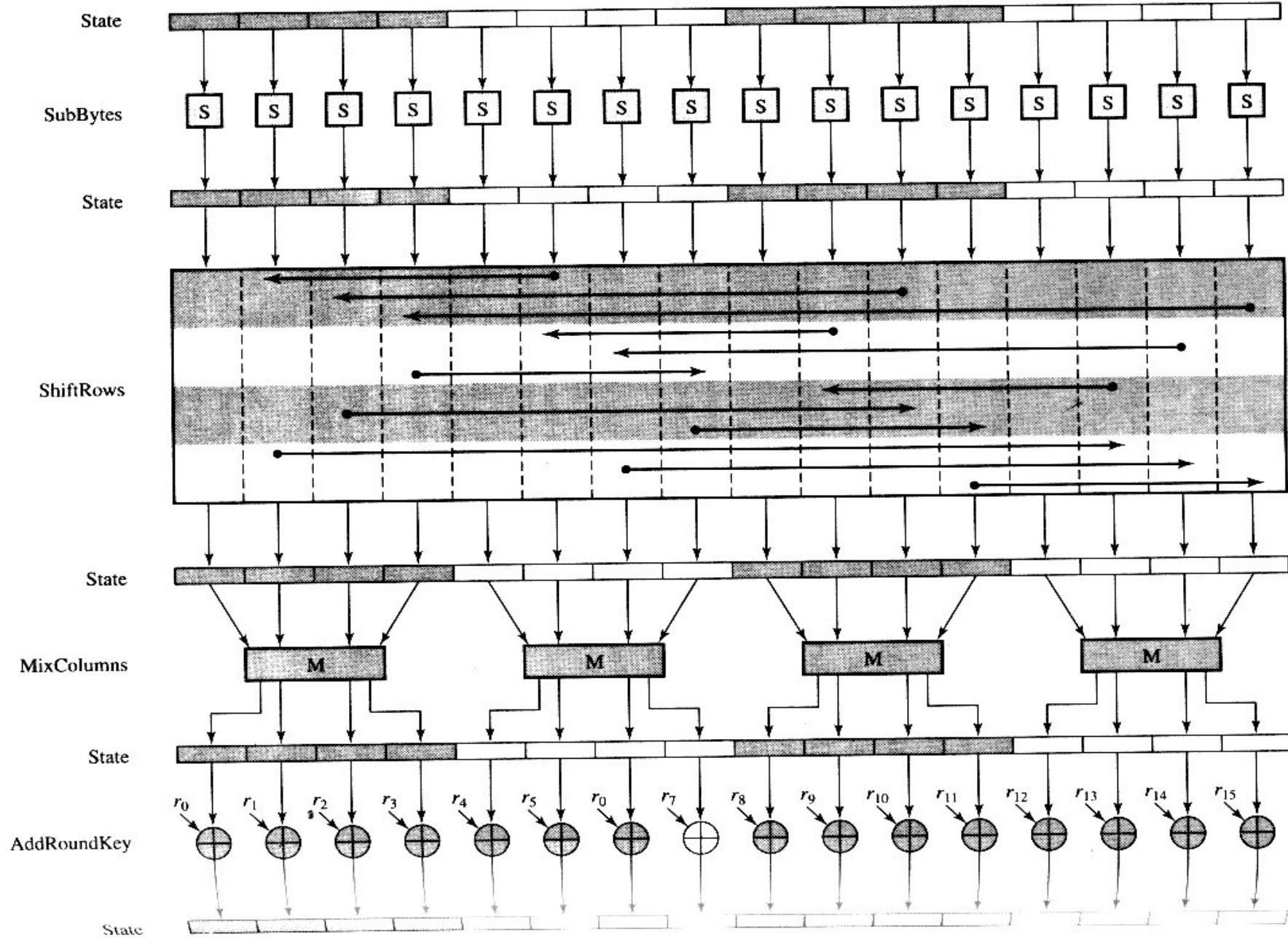
# AES - Estrutura

- Chave é expandida por matriz
- Quatro estágios por rodada:
  - Byte Sub: Caixa S GF ( $2^8$ )
  - ShiftRows: Permutação
  - MixColumns: Substituição GF ( $2^8$ )
  - AddRoundKey: XOR com chave de rodada
- XOR da chave + 9 rodadas cheias + 3 passos da ultima rodada

# AES - Estrutura

- Chave só entra em AddRoundKey
- AddRoundKey é um cifrador de Vernam
- Cada estágio é facilmente reversível
  - Chave + confusão, difusão e não linearidade
- Reversibilidade por XOR
- Decifragem usa chave na ordem invertida
- Estágio final adiciona a chave pra proteger as operações anteriores

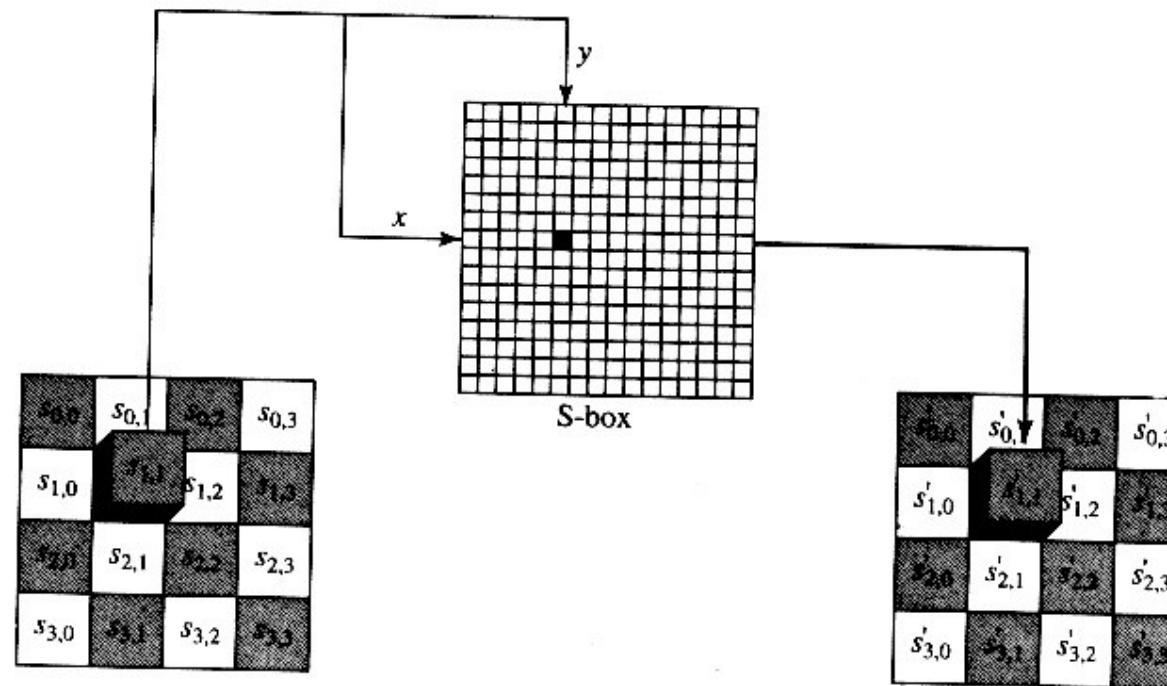
# AES - Estrutura



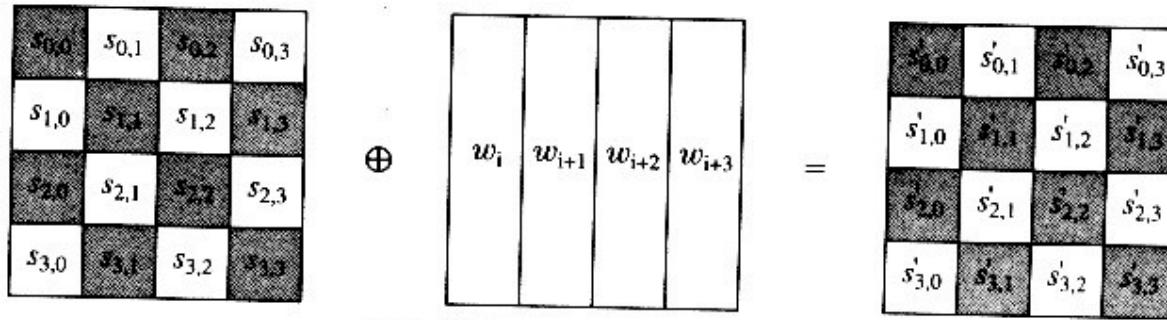
# AES - ByteSub

- Busca em Tabela
- Similar a uma Caixa S
- Resistente a todos os ataques cripto-analíticos conhecidos
- Criada com Base em aritmética no GF ( $2^8$ ), com o polinômio irredutível  $x^8 + x^4 + x^3 + x + 1$
- Funcionamento byte a byte

# AES – Funcionamento ByteSub



(a) Substitute byte transformation



(b) Add Round Key Transformation

# AES – S-Box

(a) S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	63	7C	77	7B	F2	6B	6F	C5	30	01	67	2B	FE	D7	AB	76
1	CA	82	C9	7D	FA	59	47	F0	AD	D4	A2	AF	9C	A4	72	C0	
2	B7	FD	93	26	36	3F	F7	CC	34	A5	E5	F1	71	D8	31	15	
3	04	C7	23	C3	18	96	05	9A	07	12	80	E2	EB	27	B2	75	
4	09	83	2C	1A	1B	6E	5A	A0	52	3B	D6	B3	29	E3	2F	84	
5	53	D1	00	ED	20	FC	B1	5B	6A	CB	BE	39	4A	4C	58	CF	
6	D0	EF	AA	FB	43	4D	33	85	45	F9	02	7F	50	3C	9F	A8	
7	51	A3	40	8F	92	9D	38	F5	BC	B6	DA	21	10	FF	F3	D2	
8	CD	0C	13	EC	5F	97	44	17	C4	A7	7E	3D	64	5D	19	73	
9	60	81	4F	DC	22	2A	90	88	46	EE	B8	14	DE	5E	0B	DB	
A	E0	32	3A	0A	49	06	24	5C	C2	D3	AC	62	91	95	E4	79	
B	E7	C8	37	6D	8D	D5	4E	A9	6C	56	F4	EA	65	7A	AE	08	
C	BA	78	25	2E	1C	A6	B4	C6	E8	DD	74	1F	4B	BD	8B	8A	
D	70	3E	B5	66	48	03	F6	0E	61	35	57	B9	86	C1	1D	9E	
E	E1	F8	98	11	69	D9	8E	94	9B	1E	87	E9	CE	55	28	DF	
F	8C	A1	89	0D	BF	E6	42	68	41	99	2D	0F	B0	54	BB	16	

# AES - S-Box Invertida

(b) Inverse S-box

		y															
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
x	0	52	09	6A	D5	30	36	A5	38	BF	40	A3	9E	81	F3	D7	FE
	1	7C	E3	39	82	9B	2F	FF	87	34	8E	43	44	C4	DE	E9	CF
	2	54	7B	94	32	A6	C2	23	3D	EE	4C	95	0B	42	FA	C3	4E
	3	08	2E	A1	66	28	D9	24	B2	76	5B	A2	49	6D	8B	D1	25
	4	72	F8	F6	64	86	68	98	16	D4	A4	5C	CC	5D	65	B6	92
	5	6C	70	48	50	FD	ED	B9	DA	5E	15	46	57	A7	8D	9D	84
	6	90	D8	AB	00	8C	BC	D3	0A	F7	E4	58	05	B8	B3	45	06
	7	D0	2C	1E	8F	CA	3F	0F	02	C1	AF	BD	03	01	13	8A	61
	8	3A	91	11	41	4F	67	DC	EA	97	F2	CF	CE	F0	B4	E6	73
	9	96	AC	74	22	E7	AD	35	85	E2	F9	37	E8	1C	75	DF	6F
	A	47	F1	1A	71	1D	29	C5	89	6F	B7	62	0E	AA	18	BE	1F
	B	FC	56	3E	4B	C6	D2	79	20	9A	DB	C0	FE	78	CD	5A	F4
	C	1F	DD	A8	33	88	07	C7	31	B1	12	10	59	27	80	EC	51
	D	60	51	7F	A9	19	B5	4A	0D	2D	E5	7A	9F	93	C9	9C	E1
	E	A0	E0	3B	4D	AE	2A	F5	B0	C8	EB	BB	3C	83	53	99	61
	F	17	2B	04	7E	BA	77	D6	26	E1	69	14	63	55	21	0C	71

# AES – Construção S-Box

- Cria-se uma matriz  $16 \times 16$  ordenada, com pares  $\{xy\}$  (hexadecimal), sendo  $x$  a linha e  $y$  a coluna.
- Cada byte é invertido no corpo finito  $GF(2^8)$  e 00 é mapeado para ele mesmo

$$b'_i = b_i \oplus b_{(i+4) \bmod 8} \oplus b_{(i+5) \bmod 8} \oplus b_{(i+6) \bmod 8} \oplus b_{(i+7) \bmod 8} \oplus c_i$$

$b_7$	$b_6$	$b_5$	$b_4$	$b_3$	$b_2$	$b_1$	$b_0$
1	0	1	1	0	0	0	1

# AES – Construção S-Box

- Inverse S-Box usa a inversa multiplicativa de  $b'$  em GF (2<sup>8</sup>)

$$b'_i = b_{(i+2)\text{mod}8} \oplus b_{(i+5)\text{mod}8} \oplus b_{(i+7)\text{mod}8} \oplus d_i$$

- c e d escolhidos garantem pontos opositos não fixos

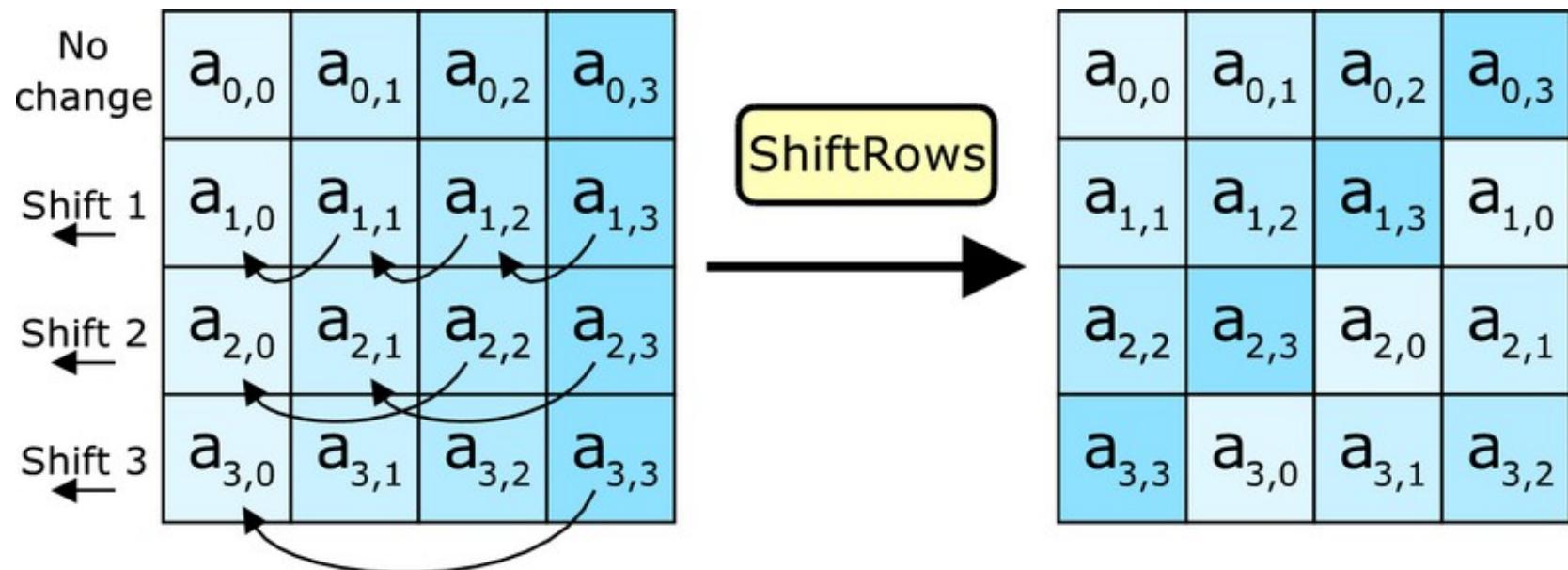
$c_7$	$c_6$	$c_5$	$c_4$	$c_3$	$c_2$	$c_1$	$c_0$
0	1	1	0	0	0	1	1

$d_7$	$d_6$	$d_5$	$d_4$	$d_3$	$d_2$	$d_1$	$d_0$
0	0	0	0	0	1	0	1

# AES - ShiftRows

- Deslocamento horizontais de n bytes por linha
  - 0 na primeira, 1 na segunda, 2 na terceira e 3 na quarta
- Cifragem gira para esquerda
- Decifragem gira para a Direita
- Garante que 4 bytes de uma coluna são dispersos para outras colunas (dados de entrada e saída lidos em colunas)

# AES - ShiftRows



# AES – MixColumns

- Multiplicação de uma coluna do estado por uma matriz pré-determinada
- Matriz  $4 \times 4$  é baseada numa inversão  $GF(2^8)$
- Cada elemento na matriz produto é a soma dos elementos de uma linha e uma coluna, tudo em  $GF(2^8)$
- Implementação prática baseada em XORs

# AES - MixColumns

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} s_{0,0} & s_{0,1} & s_{0,2} & s_{0,3} \\ s_{1,0} & s_{1,1} & s_{1,2} & s_{1,3} \\ s_{2,0} & s_{2,1} & s_{2,2} & s_{2,3} \\ s_{3,0} & s_{3,1} & s_{3,2} & s_{3,3} \end{bmatrix} = \begin{bmatrix} s'_{0,0} & s'_{0,1} & s'_{0,2} & s'_{0,3} \\ s'_{1,0} & s'_{1,1} & s'_{1,2} & s'_{1,3} \\ s'_{2,0} & s'_{2,1} & s'_{2,2} & s'_{2,3} \\ s'_{3,0} & s'_{3,1} & s'_{3,2} & s'_{3,3} \end{bmatrix}$$

$$s'_{0,j} = (2 \cdot s_{0,j}) \oplus (3 \cdot s_{1,j}) \oplus s_{2,j} \oplus s_{3,j}$$

$$s'_{1,j} = s_{0,j} \oplus (2 \cdot s_{1,j}) \oplus (3 \cdot s_{2,j}) \oplus s_{3,j}$$

$$s'_{2,j} = s_{0,j} \oplus s_{1,j} \oplus (2 \cdot s_{2,j}) \oplus (3 \cdot s_{3,j})$$

$$s'_{3,j} = (3 \cdot s_{0,j}) \oplus s_{1,j} \oplus s_{2,j} \oplus (2 \cdot s_{3,j})$$

# AES - MixColumns

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$\begin{array}{r}
 87 \quad * \quad 02 \\
 10000111 \quad * \quad 00000010 \\
 x^7+x^2+x+1 \quad * \quad x \qquad \qquad = x^8+x^3+x^2+x
 \end{array}$$

$$\begin{aligned}
 (x^8+x^4+x^3+x^2) \bmod (x^8+x^4+x^3+x+1) &= x^4+x^2+1 \\
 &\quad 000010101
 \end{aligned}$$

# AES - MixColumns

87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

→

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

87 \* 02

10000111 \* 00000010

$b_7$	$b_6$	$b_5$	$b_4$	$b_3$	$b_2$	$b_1$	$b_0$	$b_e$
1	0	0	0	0	1	1	1	0

Se  $b_7 = 0$ , então resultado =  $b_6 b_5 b_4 b_3 b_2 b_1 b_0 0$

Se  $b_7 = 1$ , então resultado =  $b_6 b_5 b_4 b_3 b_2 b_1 b_0 0 \oplus 00011011$

# AES - MixColumns



87	F2	4D	97
6E	4C	90	EC
46	E7	4A	C3
A6	8C	D8	95

47	40	A3	4C
37	D4	70	9F
94	E4	3A	42
ED	A5	A6	BC

$$6E \quad * \quad 03$$

$$01101110 \quad * \quad 00000011$$

$$01101110 \quad * \quad (00000010 \oplus 00000001)$$

$$(01101110 * 00000010) \oplus 01101110$$

$$(6E * 02) \oplus 6E$$

Se  $b_7 = 0$ , então resultado =  $b_6 b_5 b_4 b_3 b_2 b_1 b_0 0$

Se  $b_7 = 1$ , então resultado =  $b_6 b_5 b_4 b_3 b_2 b_1 b_0 0 \oplus 00011011$

# AES - MixColumns

87 \* 02 = 0001 0101 ⊕

6E \* 03 = 1011 0010 ⊕

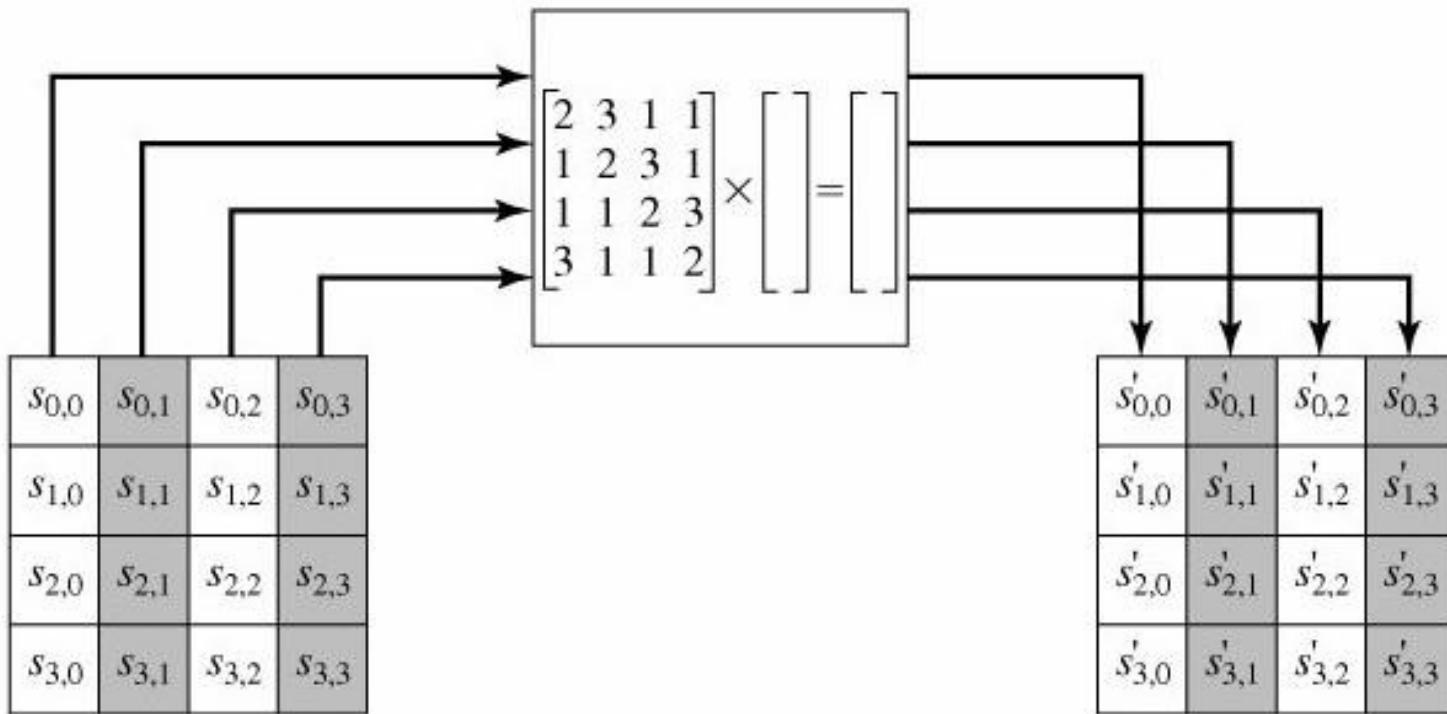
46 \* 01 = 0100 0110 ⊕

A6 \* 01 = 1010 0110 ⊕

---

0100 0111 = 47

# AES – MixColumns



# AES – MixColumns

- Matriz escolhida de forma a facilitar a implementação
  - $01 * f(x) = f(x)$
  - $02 * f(x)$ 
    - $02 = 0000\ 0010 = 0x^7 + 0x^6 + \dots + 1x + 0 = x$
    - Grau de  $f(x) < 8$ ,  $02 * f(x) = f(x) * x$  (shift a esquerda)
    - Grau de  $f(x) \geq 8$ ,  $02 * f(x) = f(x) * x \text{ XOR } 0001\ 1011$   
shift a esquerda XOR 0001 1011
  - $03 * f(x) = (x + 1) * f(x) = f(x) * x + f(x) * 1 = 02 * f(x) + f(x)$

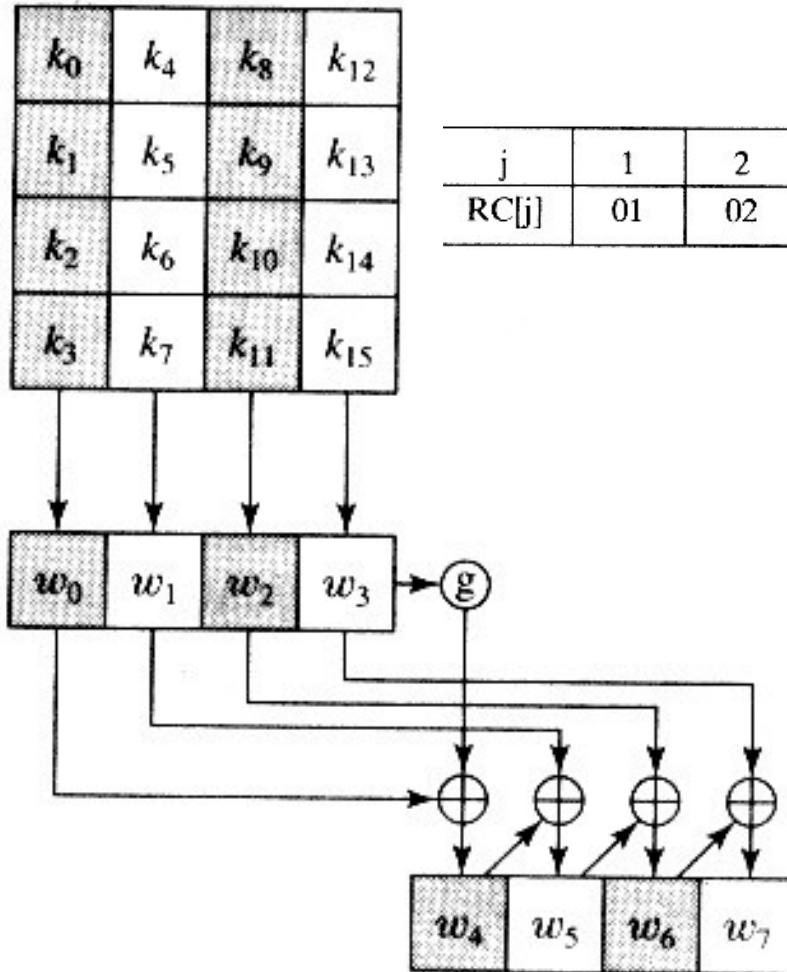
# AES - AddRoundKey

- XOR do estado com uma chave de 128 bits da rodada
- Cifrador de Vernam
- Afeta cada bit da Matriz “State”
- Simples, mas eficaz por causa dos outros passos e da expansão de chaves

# AES – Expansão de Chaves

- Entram 16 bytes e saem 176 bytes
- Produz 4 bytes para cada sub chave
- As chaves são os 4 primeiros bytes da chave expandida
- Cada byte posterior depende do byte anterior com XOR exceto o último
- $G$  é uma função complexa (rotação, substituição usando caixa S, e XOR com uma constante de rodada)

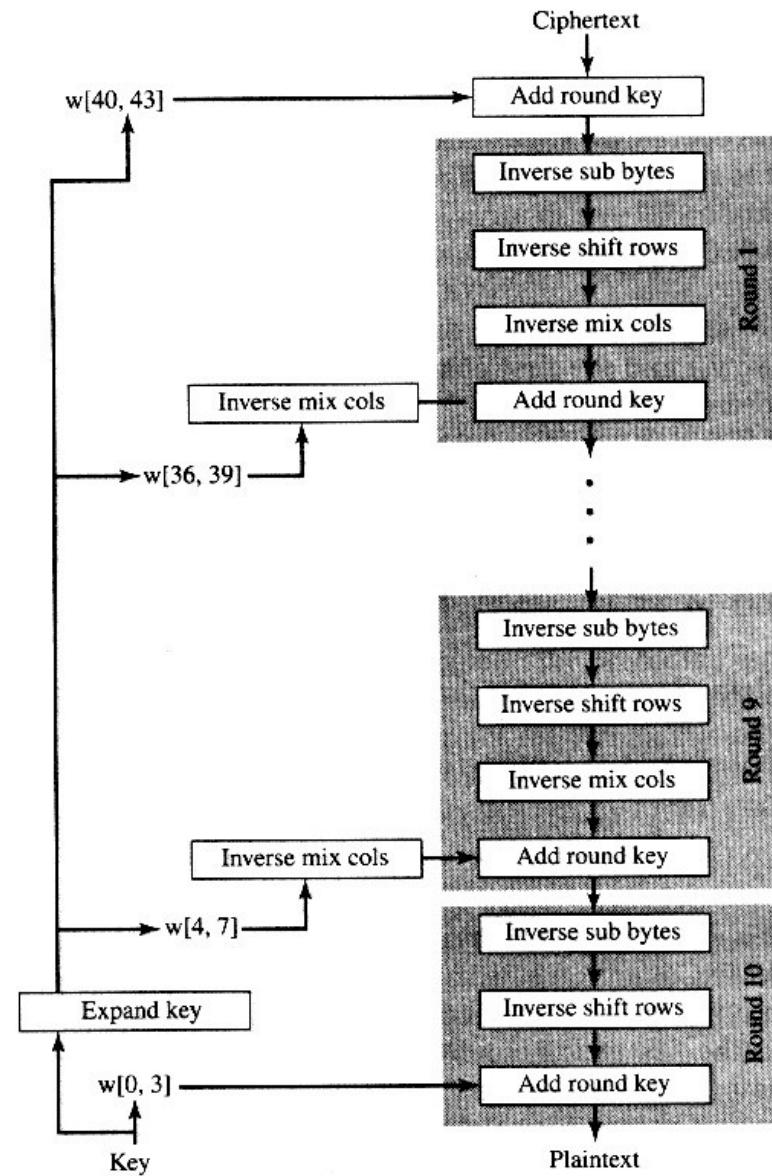
# AES – Expansão de Chaves



Função  $g$ :

- Shift left de bytes
- S-Box em cada byte
- XOR com a word  $RC(j)$  0 0 0
- $j =$  rodada

# AES – Cifrador Inverso



# AES – Cifrador Inverso

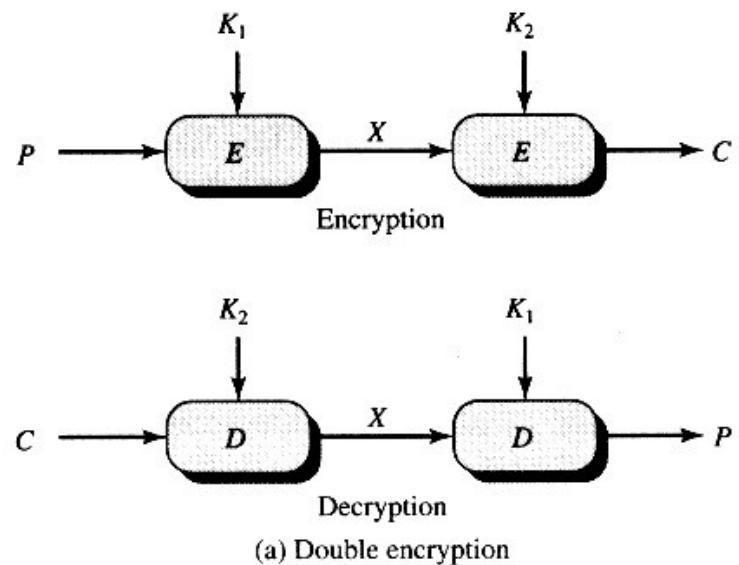
- Troca-se:
  - ShiftRows → InvShift Rows
  - SubBytes → InvSubBytes
  - MixColumns → InvMixColums
  - AddRoundKey usa as chaves em ordem invertida
- Em termos de implementação é o mesmo algoritmo, com matrizes de valores diferentes

# Cifragem Múltipla

- Aplicação do mesmo algoritmo múltiplas vezes com diferentes chaves
  - Objetivo: Aumentar o espaço de busca da chave
- Meet-in-the-Middle [DIFF77]
  - Ataque de texto claro conhecido
  - Quebra qualquer cifrador com dois pares de blocos

# Double DES

- Aumento da chave para 112 bits
- Produz mapeamentos que não são diretamente deriváveis por 1 aplicação
- Meet-in-the-middle com esforço de  $2^{56}$ , não muito maior do que  $2^{55}$  requerido pelo DES

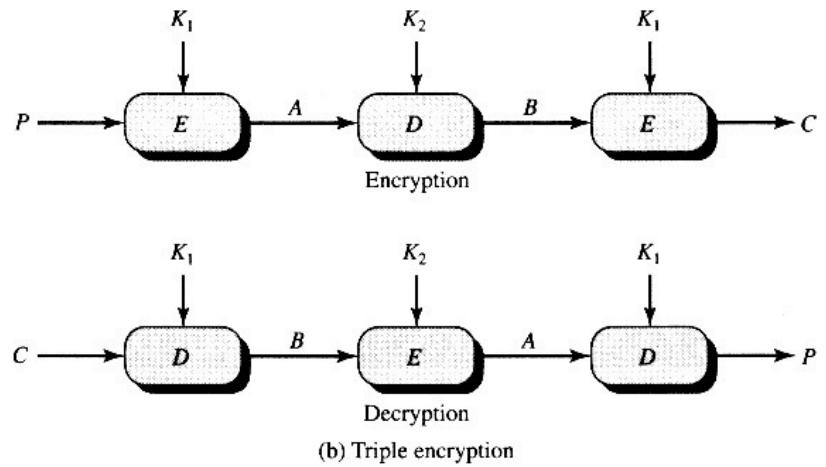


# Cifragem Múltipla

- Meet-in-the-Middle [DIFF77]
  - $C = E(K_2, E(K_1, P)) \rightarrow X = E(K_1, P) = D(K_2, C)$
  - Dois pares de  $P$  e  $C$  são conhecidos
  - Cifrar  $P$  com todas as possíveis chaves
  - Decifrar  $C$  com todas as possíveis chaves
  - Quando encontrar dois resultados iguais, testar as chaves para o outro par
  - Se os blocos coincidirem, então assume-se a chave como sendo verdadeira

# Triple DES

- Chave de 112 ou 168 bits
- Evita Meet-in-the-middle
- Pode ser EDE, ou EEE. A decifragem não muda nada criptograficamente

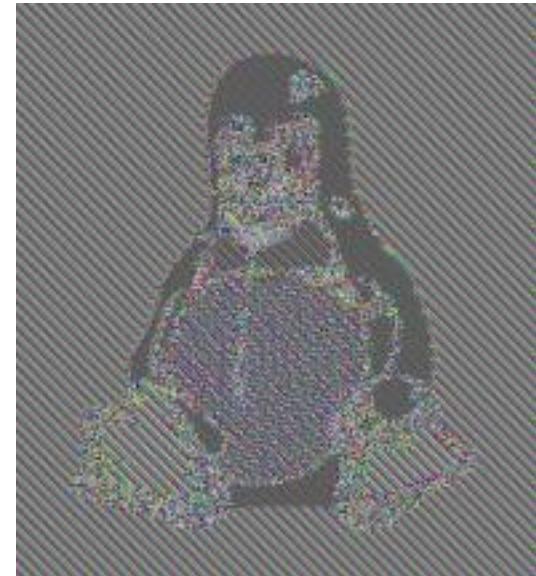


# Modos de Operação

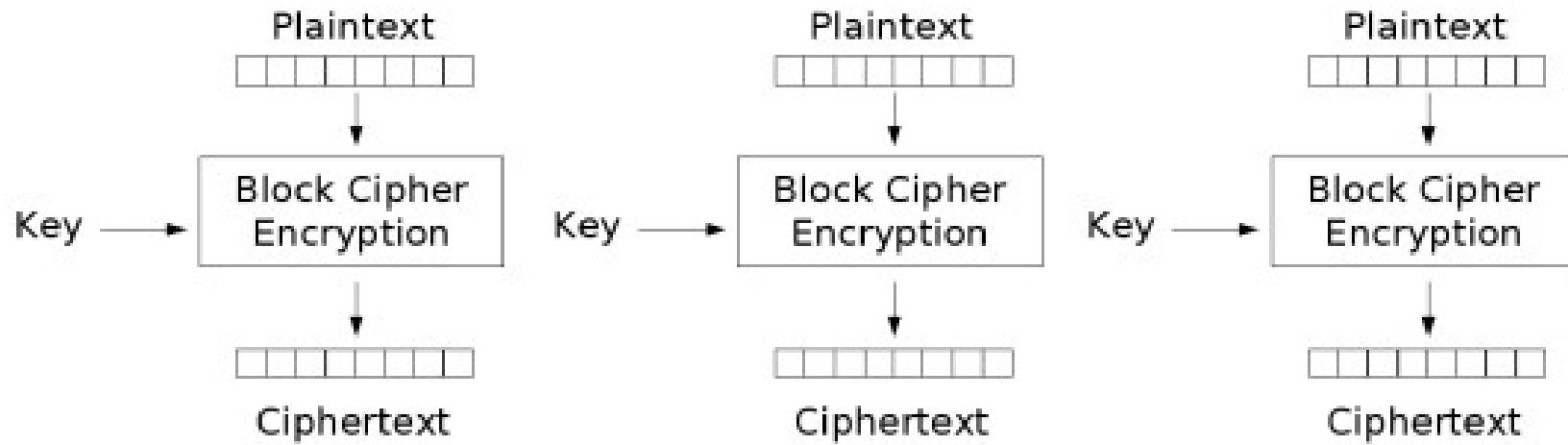
- Electronic Codebook - ECB
- Cipher Block Chaining - CBC
- Cipher Feedback - CFB
- Output Feedback - OFB
- Counter Mode - CTR

# Electronic Codebook - ECB

- Cada bloco é codificado de forma independente
- Segurança para transmissão de dados únicos e pequenos
- (senhas)



# ECB

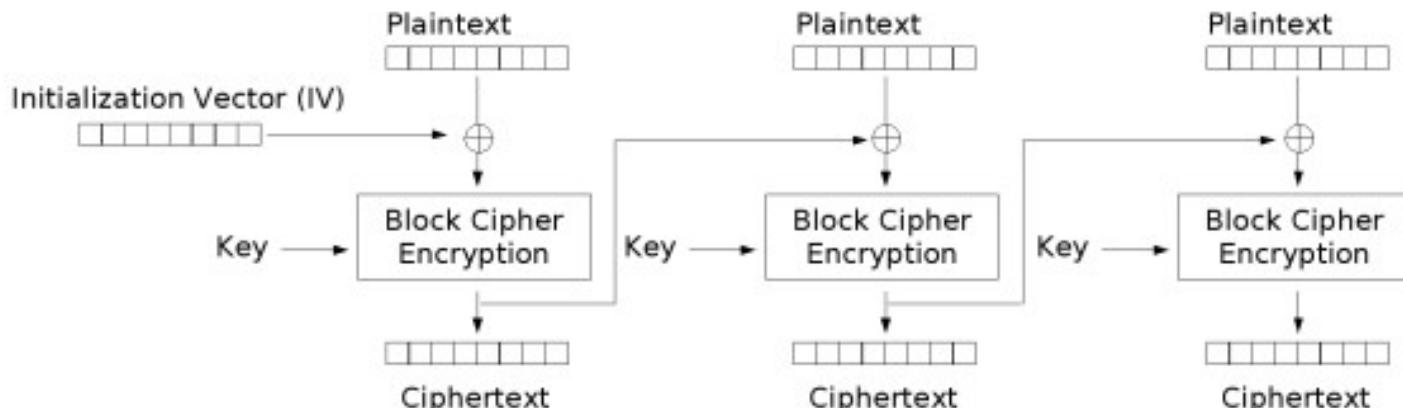


Electronic Codebook (ECB) mode encryption

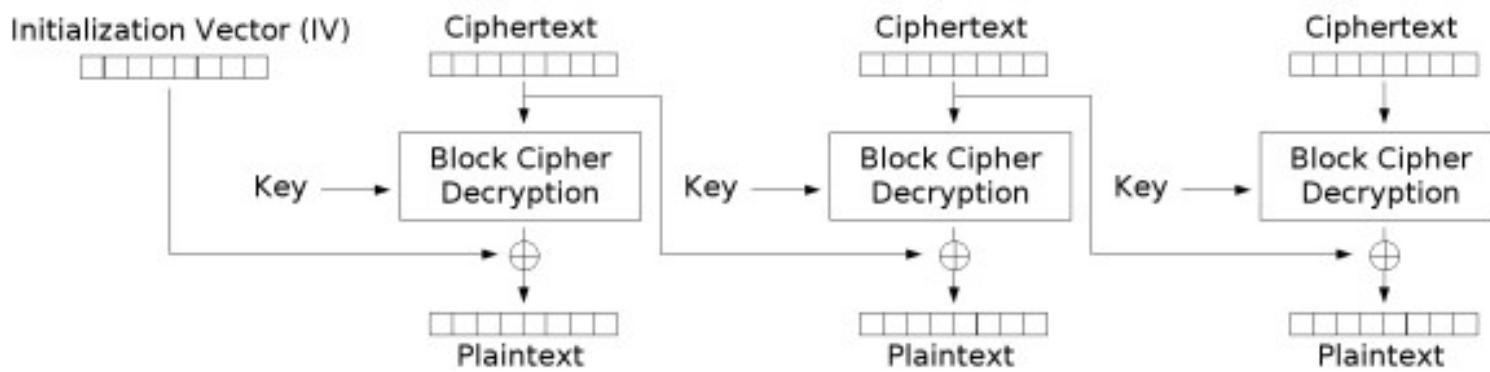
# Cipher Block Chaining - CBC

- A entrada é XOR do próximo bloco de texto claro e o bloco anterior cifrado
- Uso pra transmissão de dados gerais e autenticação
- Decifragem paralelizável

# CBC



Cipher Block Chaining (CBC) mode encryption

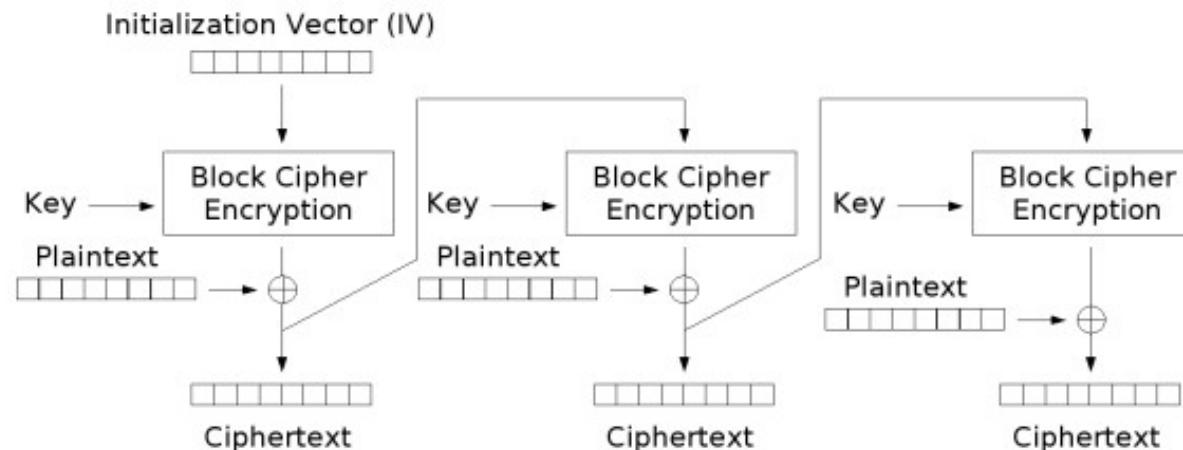


Cipher Block Chaining (CBC) mode decryption

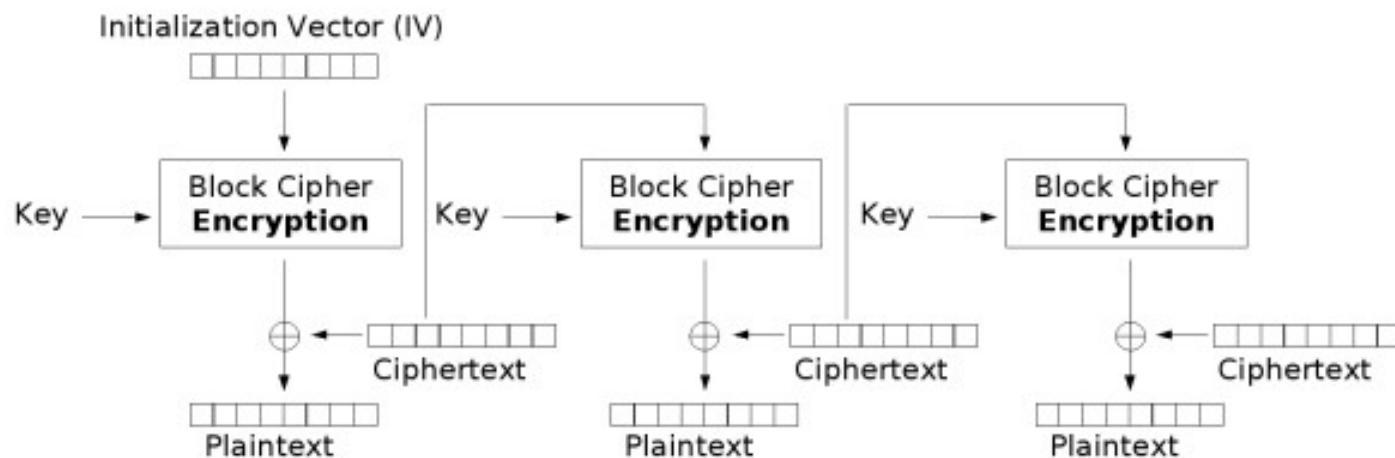
# Cipher Feedback - CFB

- O texto cifrado é XOR com o texto claro e retroalimentado no cifrador
- Transforma um cifrador de bloco em uma espécie de cifrador de fluxo
- Uso pra transmissão de dados gerais em **streaming** e autenticação
- Retroalimentação depois do XOR
- Decifragem paralelizável

# CFB



Cipher Feedback (CFB) mode encryption

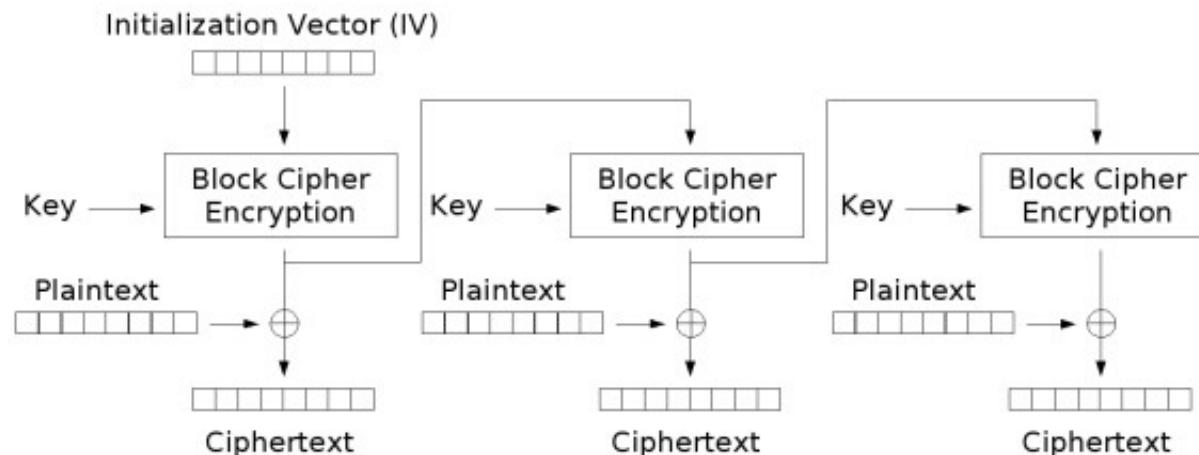


Cipher Feedback (CFB) mode decryption

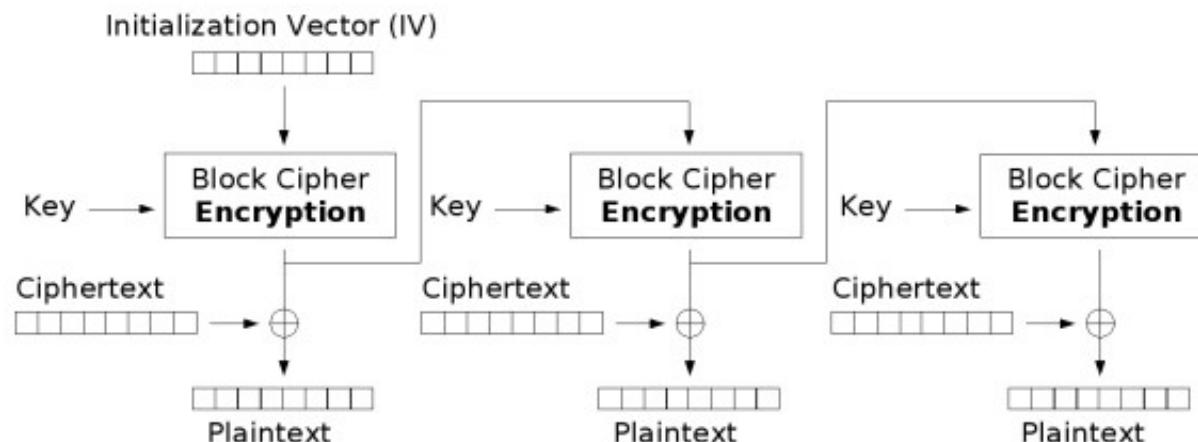
# Output Feedback - OFB

- Similar a CFB. A saída do cifrador é retroalimentada para gerar um stream de bits
- Usado em canais ruidosos
- Retroalimentação antes do XOR
- Nem cifragem nem decifragem são paralelizáveis

# OFB



Output Feedback (OFB) mode encryption

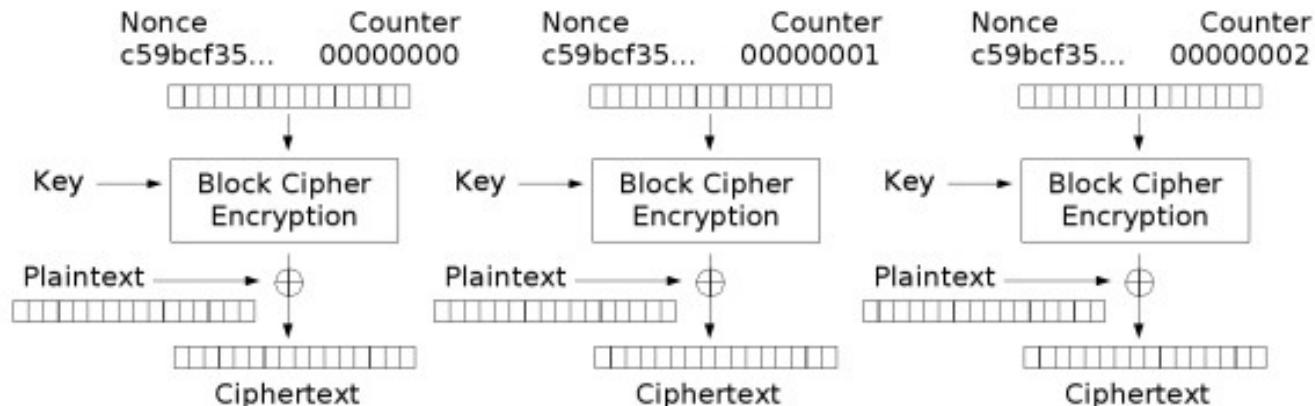


Output Feedback (OFB) mode decryption

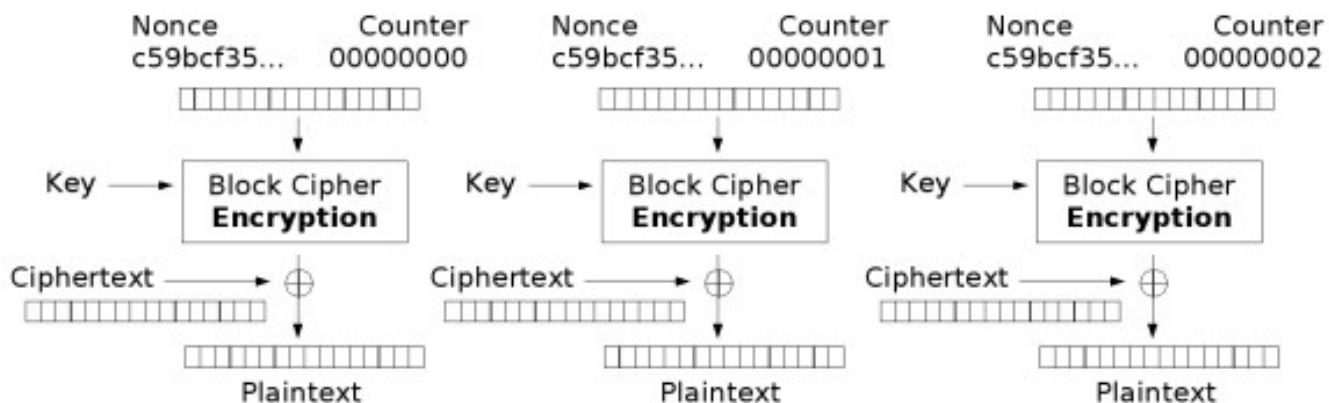
# Counter Mode - CTR

- Cada bloco é XORed com um contador cifrado
- Uso geral em transmissão de dados e em links de alta velocidade
- Cifragem e deifragem parallelizáveis

# CTR



Counter (CTR) mode encryption



Counter (CTR) mode decryption

# Teoria de Números

- Números Primos
- Teoremas de Euler e Fermat
- Teste de Primalidade
- Teorema Chinês do Resto
- Logaritmo Discreto

# Números Primos

- Primo é um inteiro que só pode ser dividido por 1 e por ele mesmo sem resto
- Todo numero inteiro pode ser representado por uma fatoração de primos
- $a = \prod_{p \in P} p^n$
- $n \geq 0$
- $12 = 2^2 * 3^1, 91 = 7^1 * 13^1$

# Números Primos

- Multiplicação de números inteiros pode ser feita pela adição de fatores primos

$$12 * 18 = (2^2 * 3^1) * (2^1 * 3^2) = 216$$

$$(2^3 * 3^3) = 8 * 27 = 216$$

# Números Primos

- Nós podemos saber que um numero divide outro se todo expoente do primo do divisor é  $\leq$  que o do dividendo
- Calcular o MDC de números expressados em notação prima é a multiplicação dos primos pelo menor expoente
- Isso só funciona facilmente para não primos

$$355 = 3 * 5^3$$

$$525 = 3 * 5^2 * 7$$

- $MDC(355, 525) = 3 * 5^2 = 75$

# Teorema de Fermat

- Se  $p$  é primo e  $a$  é um inteiro positivo não divisível por  $p$  então  $a^{p-1} \equiv 1 \pmod{p}$
- Requer que  $p$  e  $a$  sejam relativamente primos, ou seja  $\text{MDC}(a, p) = 1$
- Forma alternativa:  $a^p \equiv a \pmod{p}$
- $p=5, a=3 \rightarrow a^p = 3^5 = 243 \equiv 3 \pmod{5} = a \pmod{p}$

# Função Totiente de Euler

- A função é escrita  $\phi(n)$  e é definida como a quantidade de números menores que  $n$  e que são relativamente primos a  $n$ .
- $\Phi(1) = 1, \Phi(35) = 24 \rightarrow \{1,2,3,4,6,8,9,11,12,13,16,17,18,19,22,23,24,26,27,29,31,32,33,34\}$
- $\phi(p) = p - 1$
- $\phi(n) = \phi(pq) = \phi(p)\times \phi(q)= (p-1)(q-1)$

# Teorema de Euler

- Para todo  $a$  e  $n$  que são relativamente primos  
 $a^{\phi(n)} \equiv 1 \pmod{n}$
- $a = 3, n = 10, \phi(10) = 4$ 
  - $a^{\phi(n)} = 3^4 = 81 \equiv 1 \pmod{10} = 1 \pmod{n}$
- Requer que  $n$  e  $a$  sejam relativamente primos
- Versão alternativa:
  - $a^{\phi(n)+1} \equiv a \pmod{n}$

# Teste de Primalidade

- Saber se um numero é primo é importante para afirmar o teorema de Fermat
- Temos que trabalhar com números das ordem de grandeza de 1024 bits
- Algoritmo de Miller-Rabin
  - Determinístico
  - Probabilístico

# Background matemático

- Se  $p$  é um número primo e  $a < p$  é um inteiro positivo

$a^2 \equiv 1 \pmod{p}$  se e somente se

$a \equiv 1 \pmod{p}$  OU  $a \equiv -1 \pmod{p} = -1 = p-1$

$$a^2 \bmod p = (a \bmod p)^2 = (a \bmod p) * (a \bmod p)$$

# Background matemático

- Se  $p > 2$  é um número primo
$$p - 1 = 2^k * q, \text{ com } k > 0, q \text{ ímpar}$$
- Para todo  $1 < a < p - 1$  uma das duas condições é verdadeira:

$$a^q \bmod p = 1 \text{ OU}$$

$$a^q \bmod p, a^{2q} \bmod p, a^{4q} \bmod p, \dots,$$

$$a^{(2^{k-1})q} \bmod p = -1 = p - 1$$

# Background matemático

- Prova
$$a^q \bmod p, a^{2q} \bmod p, a^{4q} \bmod p, \dots,$$
$$a^{(2^{k-1})q} \bmod p, a^{(2^k)q} \bmod p$$
- Sabemos que  $a^{(2^k)q} \bmod p = 1$  pelo teorema de fermat

# Background matemático

- Prova
$$a^q \bmod p, a^{2q} \bmod p, a^{4q} \bmod p, \dots,$$
$$a^{(2^{k-1})q} \bmod p, a^{(2^k)q} \bmod p$$
- Sabemos que cada elemento da lista é o quadrado do anterior, então:
  - Ou  $a^q \bmod p = 1$ , assim todos os elementos da lista seriam 1
  - Ou um dos elementos que não o último é  $-1 = p-1$

# Algoritmo de Miller-Rabin

Test(n) - para n ímpar

1. ache k, q inteiros  $k > 0$ , q ímpar |  $(n-1=2^kq)$
2. rand(int a)  $\rightarrow 1 < a < n-1$
3. Se  $a^q \text{ mod } n = 1 \rightarrow \text{Inconclusivo}$
4. para j = 0 ate k -1 faca
5. se  $a^{2^{j}q} \text{ mod } n \equiv n - 1 \rightarrow \text{Inconclusivo}$
6. Senão  $\rightarrow \text{Composto}$

# Algoritmo Miller-Rabin

- Probabilidade de falha menor que  $(1/4)^t$
- $t =$  diferentes valores para  $a$
- Repetindo 10 vezes a probabilidade de ser falso primo é de  $10^{-6}$
- Tem que ser inconclusivo sempre
- Quanto maior  $t$ , mais a certeza de que  $n$  é primo

# Distribuição de Números Primos

- Todos os pares não são primos
- Primos são espalhados na ordem  $\ln(n)$
- A probabilidade de se achar um primo é  $0.5 \ln(n)$
- Para se achar um primos de 200 bits temos que tentar  $0.5 \ln(2^{200}) = 69$  na média
- A certeza do Miller-Rabin pode custar caro

# Distribuição de Números Primos

**Table 8.1** Primes under 2000

# Teorema Chinês do Resto

- “É possível reconstruir inteiros a partir de seus resíduos módulo um conjunto de número relativamente primos entre si”
- Permite manipular números potencialmente grandes mod M em termos de tuplas de números menores (relativamente primos entre si)
- $Z_{10}$ , mod 2 e mod 5 como fatores,  $r_2 = 0$  e  $r_5 = 3$  tem como solução única  $x = 8$

# Teorema Chinês do Resto

- $973 \bmod 1813 \rightarrow (\bmod 37, \bmod 49)$ 
  - $973 \bmod 37 = 11, 973 \bmod 49 = 42 \rightarrow (11, 42)$
- $678 \bmod 1813 \rightarrow (\bmod 37, \bmod 49)$ 
  - $678 \bmod 37 = 12, 678 \bmod 49 = 41 \rightarrow (12, 41)$
- $(973 + 678) \bmod 1813 = (23, 34) = 1651$

# Geradores de Números Aleatórios

- Uso:
  - Geração de chaves
  - Geração de parâmetros
  - Controles de sessão
- Aleatoriedade:
  - Distribuição uniforme de 0 e 1 → fácil
  - Independência → difícil
- Estratégia testes de independência similar a Miller-Rabin

# Geradores de Números Aleatórios

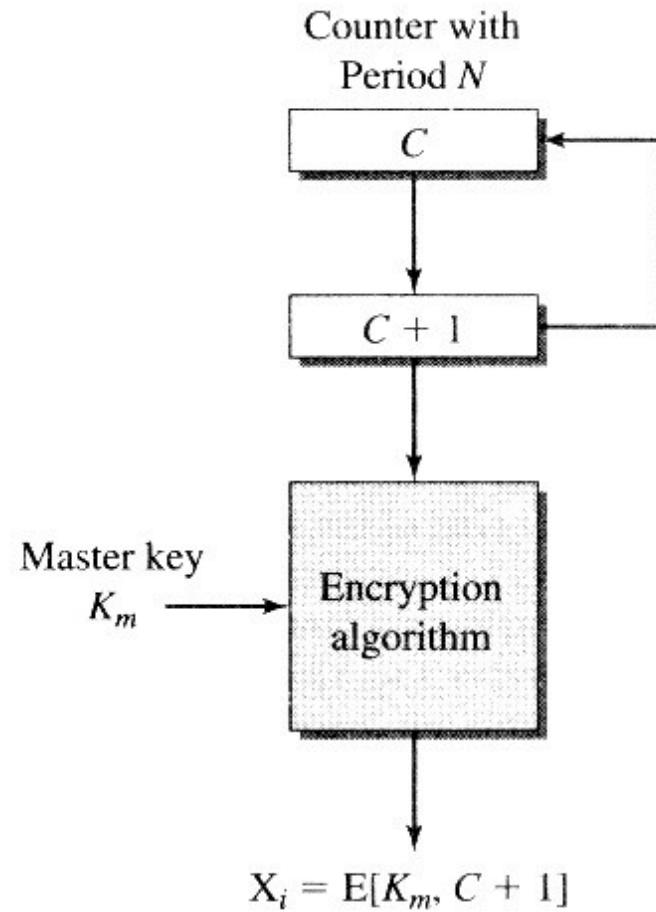
- Não previsibilidade → nonces
- Solução determinística x não determinística
- Geradores Pseudo-Aleatórios:
  - Determinístico (dada um entrada, sempre a mesma saída)
  - Passa por testes de aleatoriedade
  - Aleatoriedade relativa
    - Geradores de Congruência Linear
    - Geradores Criptográficos

# Geradores de Congruência Linear

- Modulo m, multiplicador a, incremento c e semente inicial  $X_0$
- $X_{n+1} = (aX_n + c) \text{ mod } m$ ,  $0 \leq X_n < m$
- Dependente na boa escolha de parâmetros
  - m perto ou igual a  $2^{31}$
  - Um bom a é difícil → um punhado em 2 bilhões pra ter um período próximo a m
    - bom período garante pouca repetição de valores
    - normalmente a = 16807

# Geradores Criptográficos

- Cifragem cíclica
  - Bom para chaves de sessão
- DES em OFB com a semente sendo a chave
- ANSI X9.17: 3-DES é um dos mais robustos



# Logaritmo Discreto

- Utilizado no Diffie-Hellman e DSA
- $\log_a(b) = x \rightarrow a^x = b$
- É o logaritmo calculado  $Z_p$
- $3^4 \text{ mod } 17 = 13 \rightarrow 3^k = 13 \pmod{17}$ 
  - 4 é uma solução, mas na verdade inúmeras soluções existem  $\rightarrow 4 + 16n = \log_3(13) \pmod{17}$
  - Equivalente a  $k \equiv 4 \pmod{16}$
- Não existe algoritmo eficiente pra isso

# Logaritmo Discreto

- Força bruta: elevar a base a maiores potência de k ate achar o valor certo
  - Não existe algoritmo eficiente na computação não-quântica
- Funciona para criptografia, porque é fácil fazer com a exponenciação, mas difícil fazer o logaritmo discreto
- Assimetria equivalente da multiplicação e fatoração de números primos
- Eficiente em outros grupos (curvas elípticas)

# Criptografia com Chave Pública

- Criptografia com chave pública NÃO é mais segura que simétrica
- Criptografia com chave pública NÃO surgiu para substituir a simétrica
- Distribuição de chaves NÃO é mais simples na criptografia com chave pública

# Criptografia com Chave Pública

- Proposto por Diffie-Hellman (1976)
- Revolução na criptografia
  - Baseada em funções matemáticas
  - Deixa de lado substituição e permutação

# Princípios de Cripto-sistemas de Chave Pública

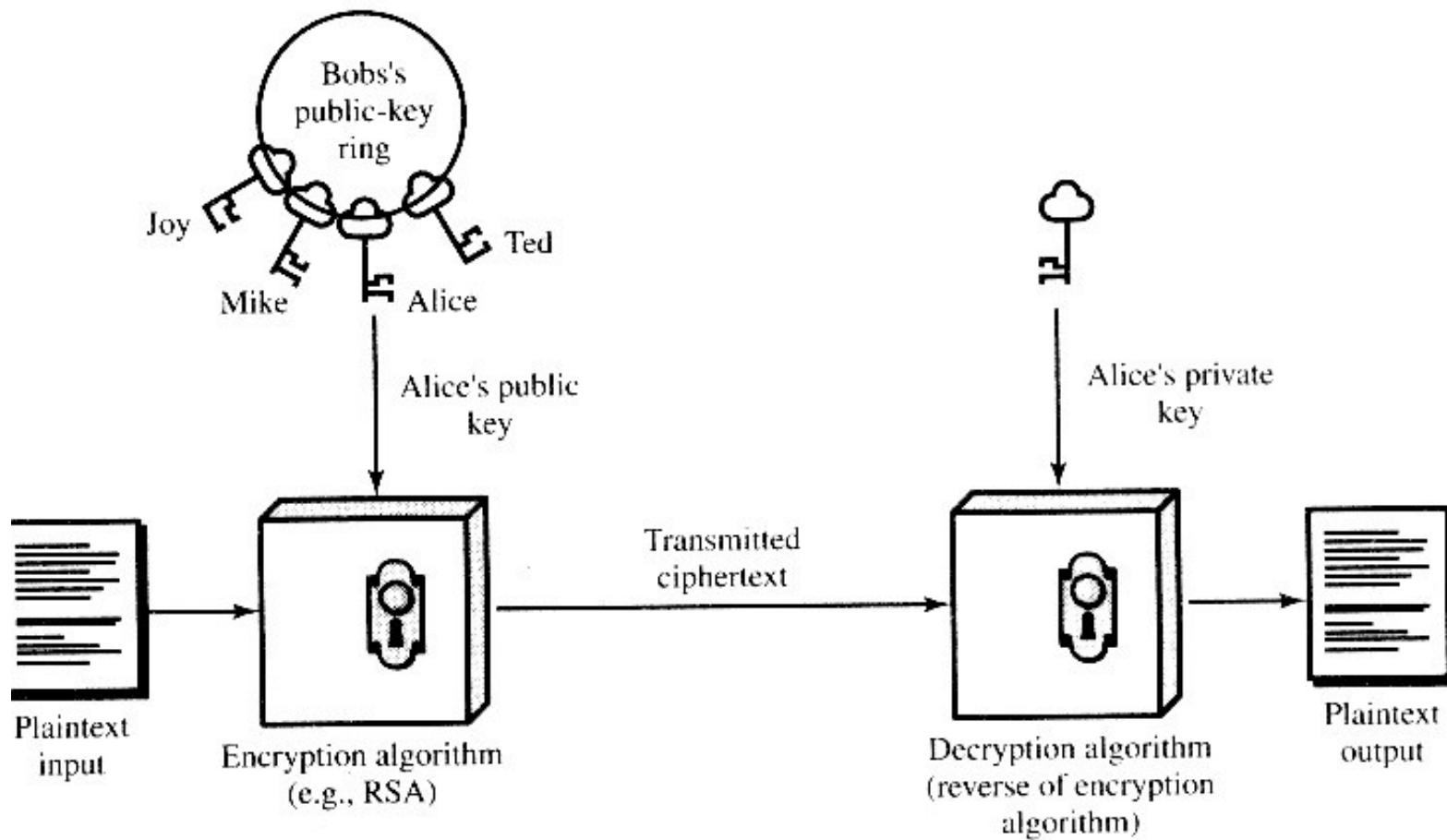
- Uma chave pública e uma privada
- O que é feito com uma chave poder ser “desfeito” com outra
- Chave assimétrica prove:
  - Confidencialidade, Autenticação, e derivados
- Foi criada para responder ao problema de distribuição de chaves
- Provê assinatura digital

# Cripto-sistemas de Chave Pública -

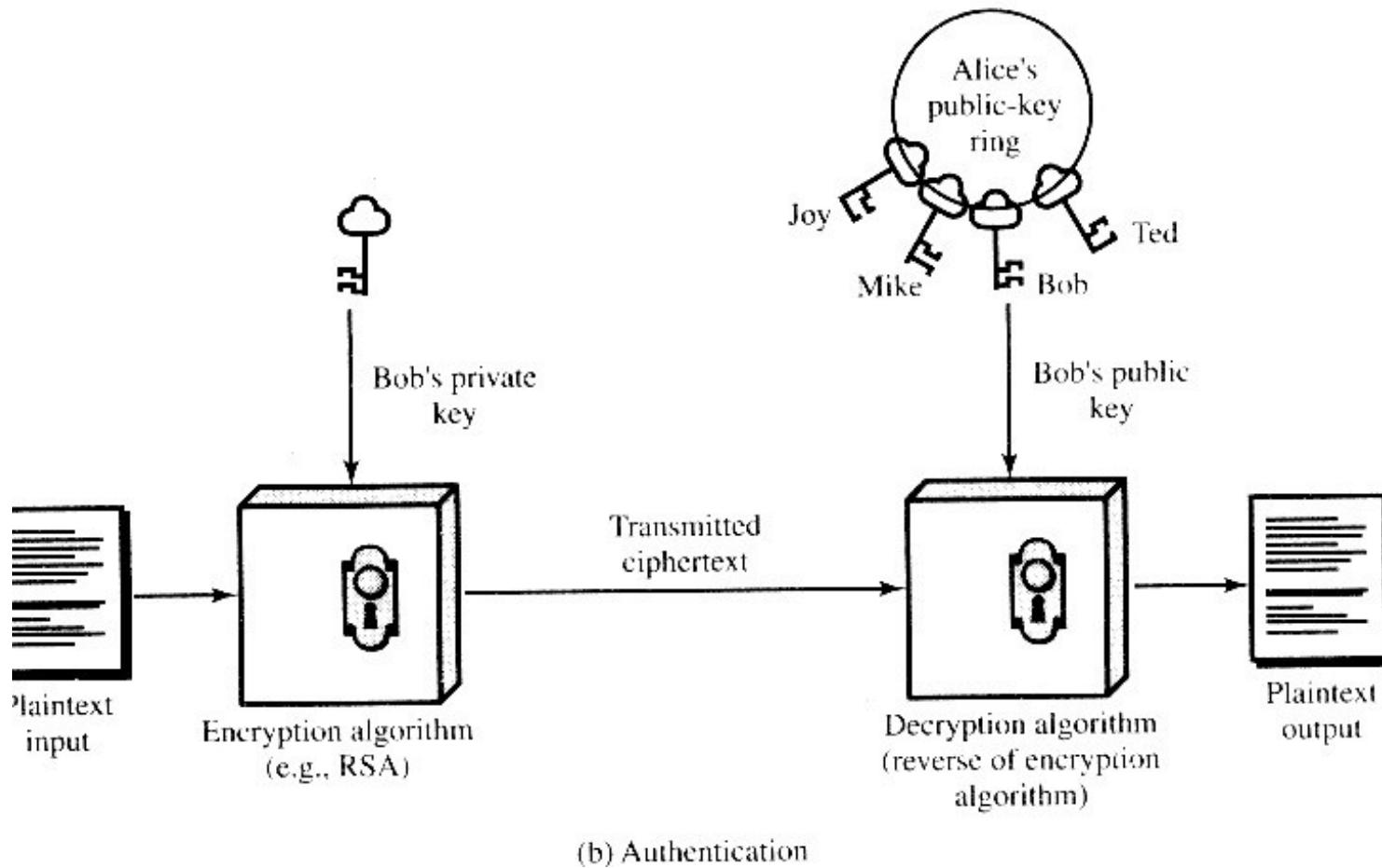
## Elementos

- Texto claro
- Algoritmo de cifragem
- Par de chaves
- Texto cifrado
- Algoritmo de decifragem
- É computacionalmente impossível determinar a chave privada através da chave pública

# Cripto-sistemas de Chave Pública - Cifragem



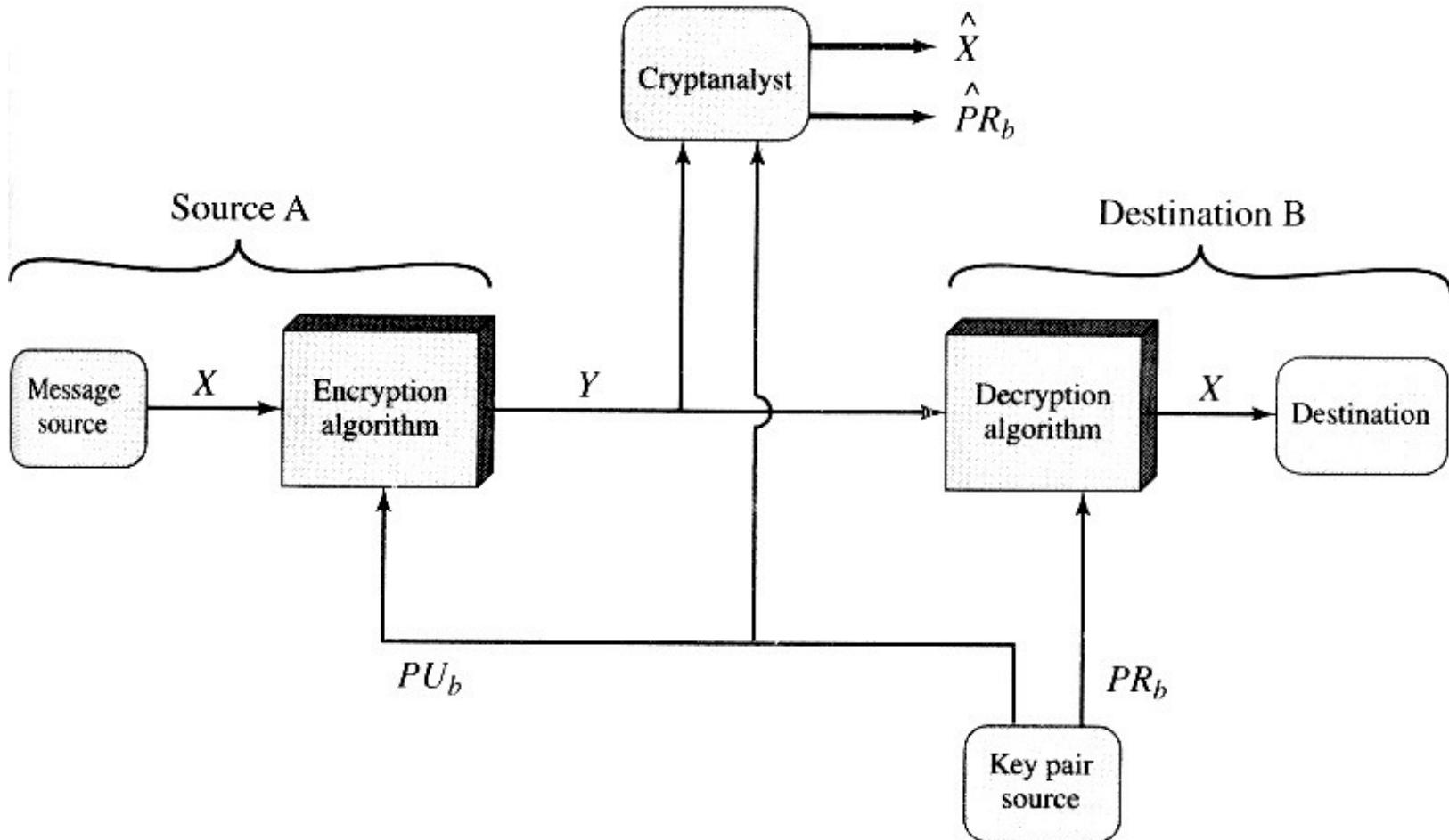
# Cripto-sistemas de Chave Pública - Autenticação



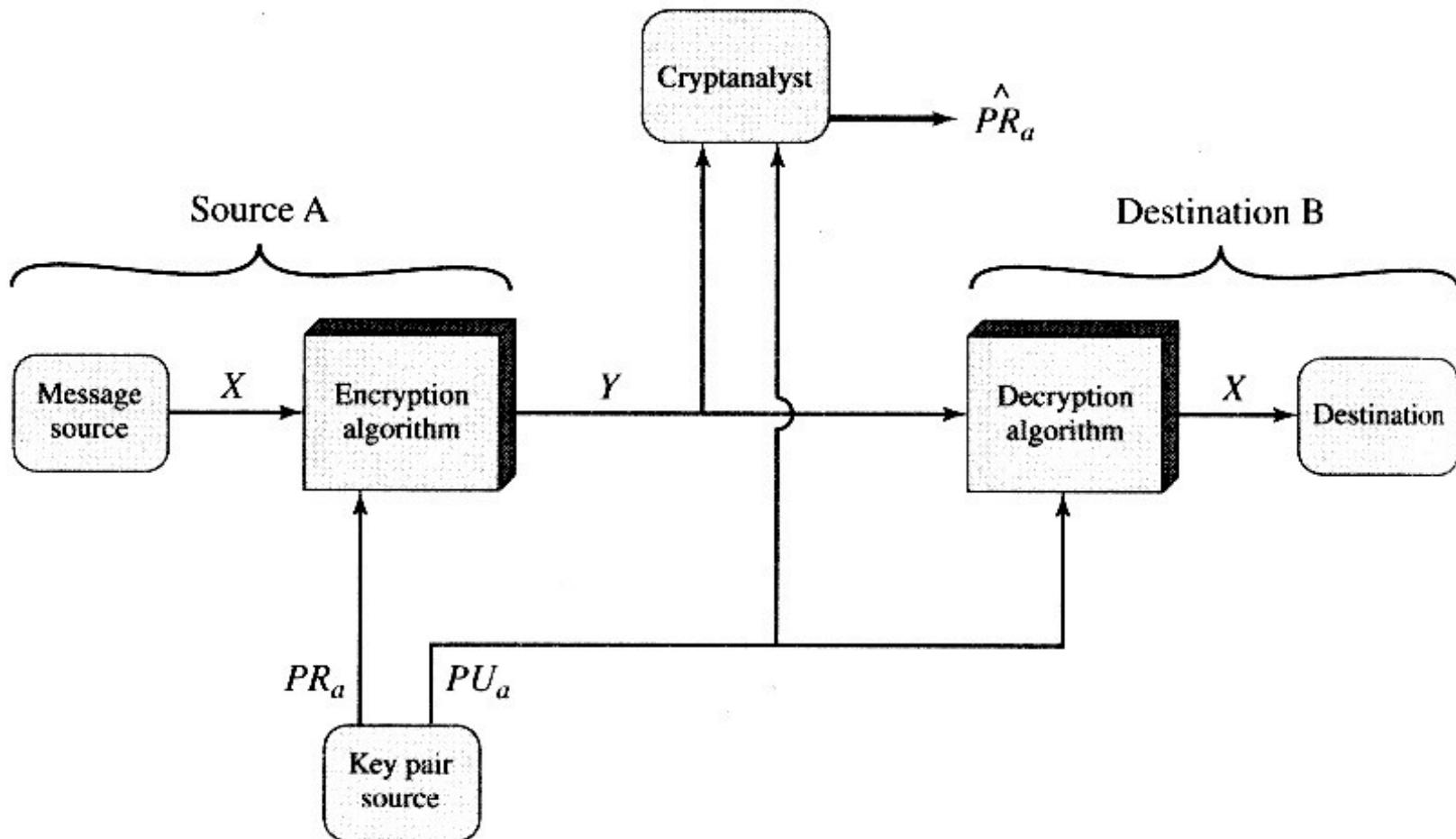
# Chave secreta x Chave pública

- Chave Secreta:
  - Funcionamento:
    - Mesmo algoritmo
    - Compartilhamento da chave
  - Segurança:
    - Chave secreta
    - Impossível quebrar sem a chave
- Chave Pública:
  - Funcionamento:
    - Diferentes algoritmos
    - Pares de chaves
  - Segurança:
    - Uma chave secreta
    - Impossível derivar a outra chave
    - Impossível quebrar com uma só chave

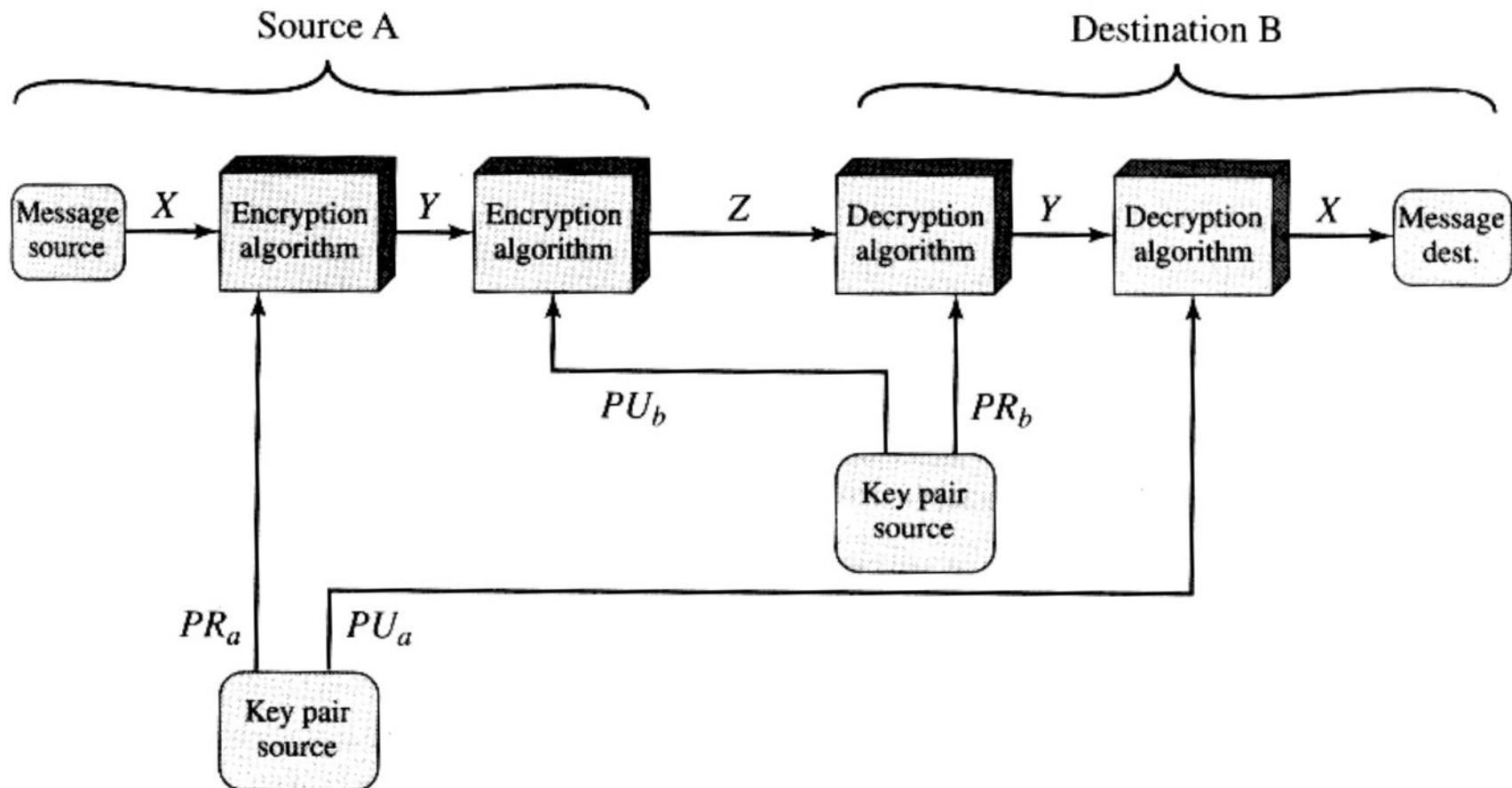
# Modelo Cripto-Analítico - Confidencialidade



# Modelo Cripto-Analítico - Autenticação



# Modelo Cripto-Analítico - Misto



# Aplicações de Chave Pública

- Cifragem/Decifragem
- Assinatura Digital
- Troca de Chaves

Algorithm	Encryption/Decryption	Digital Signature	Key Exchange
RSA	Yes	Yes	Yes
Elliptic Curve	Yes	Yes	Yes
Diffie-Hellman	No	No	Yes
DSS	No	Yes	No

# Requisitos de Chave Pública

- Fácil (computacionalmente) gerar um par de chaves
- Fácil para o remetente cifrar com a chave pública
- Fácil para o destinatário decifrar com a chave privada
- Impossível determinar  $K_r$  a partir de  $K_u$
- Impossível recuperar o texto claro conhecendo  $K_u$  e o texto cifrado

# Criptoanálise de Chave Pública

- Função de caminho único com “dica”
  - $Y=f(x) \rightarrow$  fácil ,  $X=f^{-1}(Y) \rightarrow$  impossível
- Fácil quando se conhece a “dica”
- Ataque de força bruta ainda existe
  - Chave pequena -> força bruta
  - Chave grande -> lentidão

# RSA

- 1977, Rivest, Shamir e Adelman / MIT
- É o algoritmo mais aceito
  - Base para a Web
  - Base para assinatura digital no Brasil
- Texto claro e texto cifrado são inteiros mod n
- Tamanho do bloco é normalmente 1024 bits (309 dígitos)
- É baseado em exponenciação mod p

# RSA - Algoritmo

- Blocos de com valores menores que n
- $C = M^e \text{ mod } n$
- $M = C^d \text{ mod } n = ((M^e)^d) \text{ mod } n = M^{ed} \text{ mod } n$
- Todos conhecem n, o remetente conhece e, o destinatário conhece d
- Chave Pública  $\rightarrow (n, e)$
- Chave Privada  $\rightarrow (n, d)$

# RSA - Requisitos

- $e, d, n$  são escolhidos pra satisfazer  $M^{ed} \bmod n = M$  para todo  $M < n$
- Para isso “ $e$ ” e “ $d$ ” devem ser multiplicativas inversas mod  $\phi(n)$   $\rightarrow e \cdot d \bmod \phi(n) = 1$ 
  - $e \cdot d \equiv 1 \bmod \phi(n) \rightarrow d \equiv e^{-1} \bmod \phi(n)$
  - $\gcd(\phi(n), d) = 1$
  - $\gcd(\phi(n), e) = 1$

# RSA – Requisitos Práticos

- $p, q$  primos: privados e escolhidos
- $n = p \cdot q$ : público e calculado
- $e \mid \text{gcd}(\phi(n), e) = 1 \wedge 1 < e < \phi(n)$ : público e escolhido
- $d \equiv e^{-1} \pmod{\phi(n)}$  privado e calculado
- Chave pública ( $e, n$ )
- Chave privada ( $d, n$ )

# RSA na Prática

- $p = 17$  e  $q = 11$
- $n = \text{porque} = 17 \times 11 = 187$
- $\phi(n) = (p - 1)(q - 1) = 16 \times 10 = 160$
- $e = 7$ ,  $\gcd(160, 7) = 1 \wedge 1 < 7 < 160$
- $d \mid de \equiv 1 \pmod{160} \wedge d < 160 \rightarrow d = 23$   
 $- 23 \times 7 = 161$
- $K_u = \{7, 187\}$  ,  $K_r = \{23, 187\}$

# RSA – Cifragrem/Decifragem

## Prática

- Texto Claro = 88
- $88^7 \text{ mod } 187 = 11$
- Texto cifrado = 11
- $11^{23} \text{ mod } 187 = 88$
- Computacionalmente intensivo de fazer com números grande

# RSA - Considerações Computacionais

- Exponenciação mod n requer truques matemáticos
  - $88^7 \text{ mod } n = (88^1 * 88^2 * 88^4) \text{ mod } n$
- O e acaba sendo fixo em primos como: 65537 ( $2^{16} + 1$ ), 17 ou 3, e sofre ataques se utilizado muitas vezes
- d tem que ser grande para evitar força bruta
- Gerar chaves pode ser demorado pois precisamos do M-R várias vezes em um número muito grande

# Segurança do RSA

- Força Bruta:
  - Todas as possíveis chaves
  - $\uparrow$  Tamanho  $\downarrow$  Eficiência
- Ataques Matemáticos:
  - Todos equivalentes a fatorar  $p \cdot q$  (achar o  $\phi(n)$ )
- Ataques de Tempo
  - Adivinhação da chave privada pelo tempo gasto na decifragem

# RSA - Ataques matemáticos

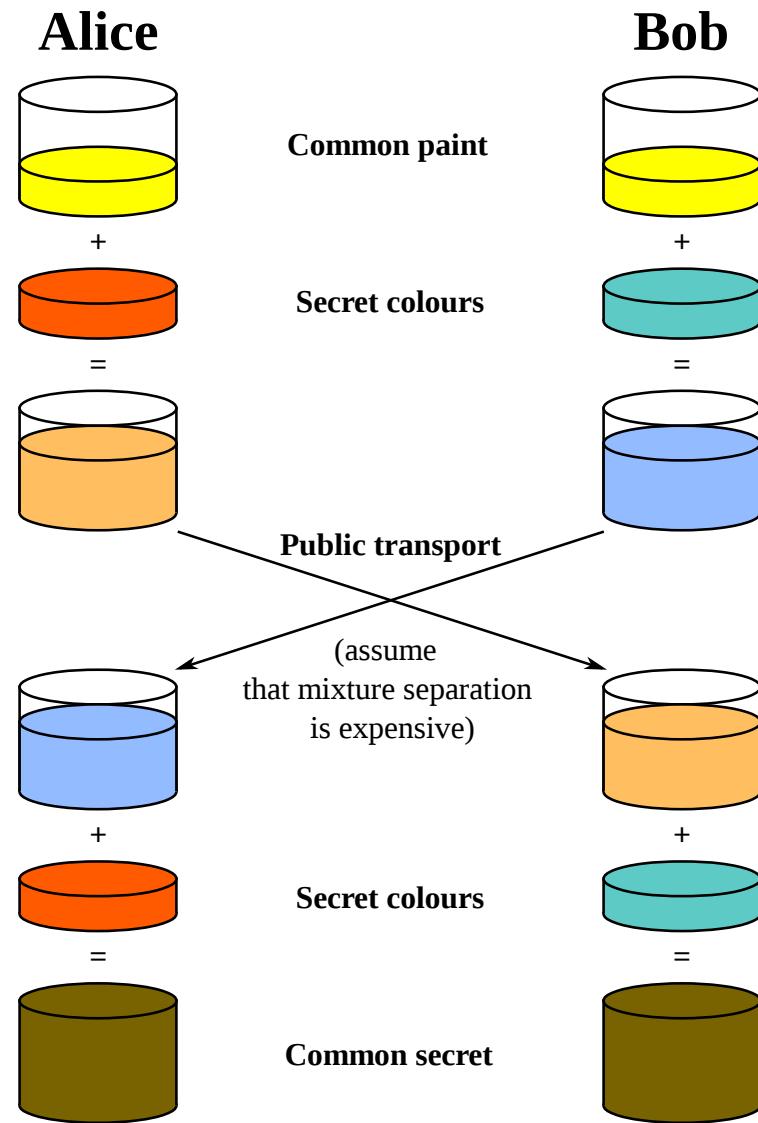
Number of Decimal Digits	Approximate Number of Bits	Date Achieved	MIPS-years	Algorithm
100	332	April 1991	7	Quadratic sieve
110	365	April 1992	75	Quadratic sieve
120	398	June 1993	830	Quadratic sieve
129	428	April 1994	5000	Quadratic sieve
130	431	April 1996	1000	Generalized number field sieve
140	465	February 1999	2000	Generalized number field sieve
155	512	August 1999	8000	Generalized number field sieve
160	530	April 2003	—	Lattice sieve
174	576	December 2003	—	Lattice sieve
200	663	May 2005	—	Lattice sieve

[http://en.wikipedia.org/wiki/RSA\\_Factoring\\_Challenge#The\\_prizes\\_and\\_records](http://en.wikipedia.org/wiki/RSA_Factoring_Challenge#The_prizes_and_records)

# Troca de Chaves Diffie-Hellman

- Primeiro algoritmo publicado de chave pública
- Objetivo: Troca segura de parâmetros para estabelecer uma chave de sessão
- O algoritmo depende da dificuldade de calcular logaritmos discretos
- Raiz primitiva  $\rightarrow a \bmod p \dots a^{p-1} \bmod p$
- $b \equiv a^i \pmod{p}$  onde  $0 \leq i \leq p \rightarrow \text{dlog}_{a,p}(b)$

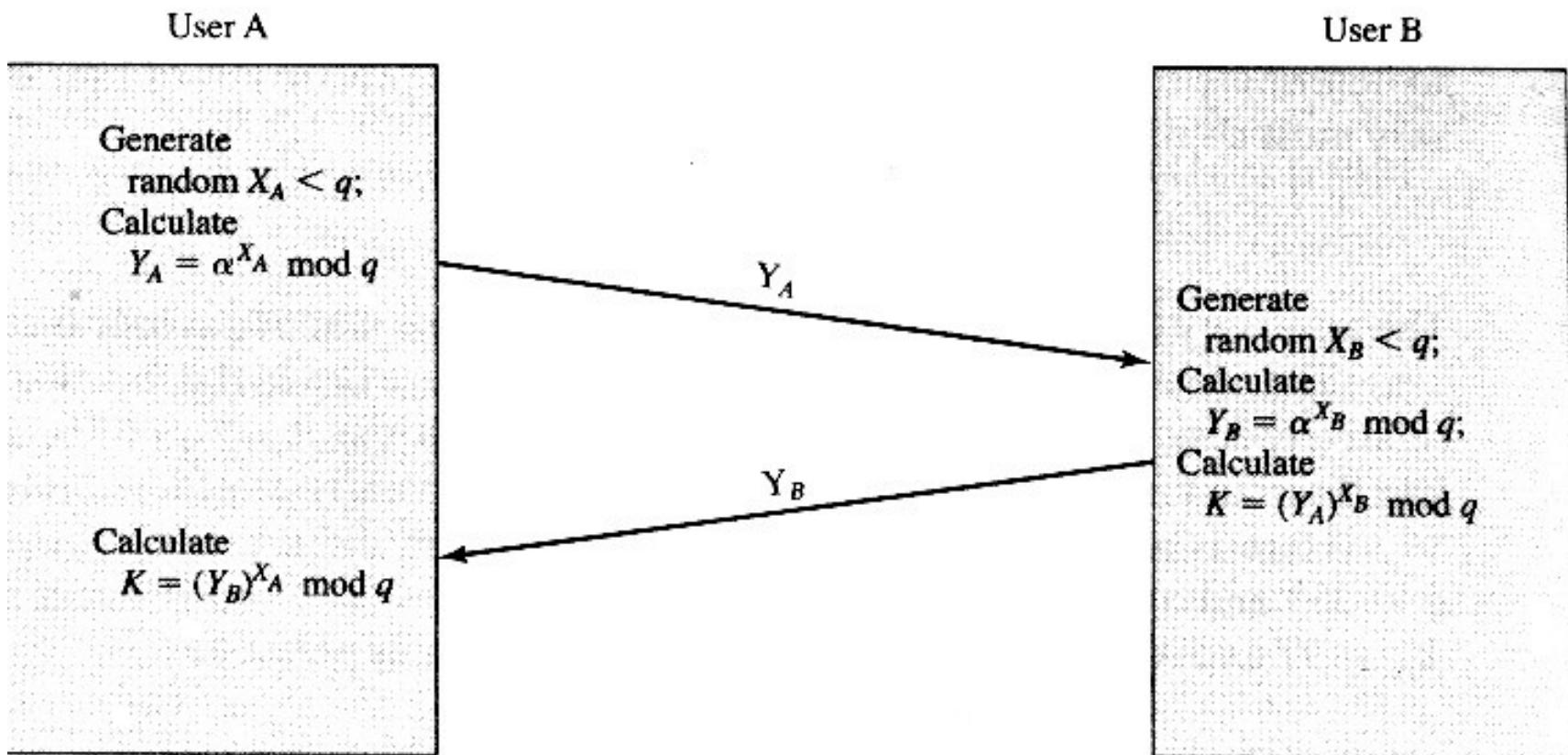
# Troca de Chaves Diffie-Hellman



# Diffie-Hellman - Algoritmo

- Parâmetros:
  - $q$  numero primo,  $a$  raiz primitiva de  $q \rightarrow$  públicos
  - $X_a$  e  $X_b < q$  números aleatórios secretos
  - Geração de chave:
    - $Y_a = a^{X_a} \text{ mod } q$  e  $Y_b = a^{X_b} \text{ mod } q$
  - Segredo:
    - $K = (Y_b)^{X_a} \text{ mod } q$
    - $K = (Y_a)^{X_b} \text{ mod } q$
- O adversários só sabe  $q$ ,  $a$  , $Y_a$  e  $Y_b$

# Protocolos de Troca da Chaves



# Diffie-Hellman - Exemplo

- $q = 353$ ,  $a = 3$ ,  $X_a = 97$  e  $X_b = 233$
- A computa:
  - $Y_a = 3^{97} \text{ mod } 353 = 40$
- B computa:
  - $Y_b = 3^{233} \text{ mod } 353 = 248$
- A deriva:
  - $K = 248^{97} \text{ mod } 353 = 160$
- B deriva:
  - $K = 40^{233} \text{ mod } 353 = 160$

# Diffie-Hellman – Ataque MITM

- C gera  $X_{c1}$ ,  $X_{c2}$  e computa  $Y_{c1}$  e  $Y_{c2}$
- C intercepta  $Y_a$  de A para B, manda como A  $Y_{c1}$  pra B e calcula  $K_2 = (Y_a)^{X_{c2}} \text{ mod } q$
- B recebe  $Y_{c1}$ , calcula  $K_1 = (Y_{c1})^{X_b} \text{ mod } q$
- B manda  $Y_b$  para A, C intercepta, manda  $Y_{c2}$  pra A e calcula  $K_1 = (Y_b)^{X_{c1}} \text{ mod } q$
- A recebe  $Y_{c2}$  e calcula  $K_2 = (Y_{c2})^{X_a} \text{ mod } q$
- C atua como proxy

# Autenticação de Mensagens

- Garantia de que a mensagem está íntegra e que foi enviada por alguém válido
- Cifragem garante autenticação
  - Somente as duas partes conhecem o segredo
  - Se B recebe uma mensagem cifrada, então A deve ter enviado
  - Não é prático quando o texto claro não é legível (seqüência aleatória de bits)

# Autenticação de Mensagens

- MAC é um algoritmo de verificação que requer uma chave e garante autenticação
  - O modelo mais popular utiliza Hash
  - Outro modelo popular utiliza cifradores de bloco
- Assinatura eletrônica garante autenticação de mensagens
  - Garante integridade e autenticidade da fonte
  - Também garante não repúdio

# Autenticação - Ataques

- Mascaramento:
  - Origem fraudulenta
- Modificação de conteúdo:
  - Alteração da carga da mensagem
- Modificação de seqüência:
  - Reordenamento de mensagens
- Modificação de tempo:
  - Replay e delay

# Funções de Autenticação

- Autenticadores:
  - Cifragem
  - Message Authentication Codes
  - Funções HASH

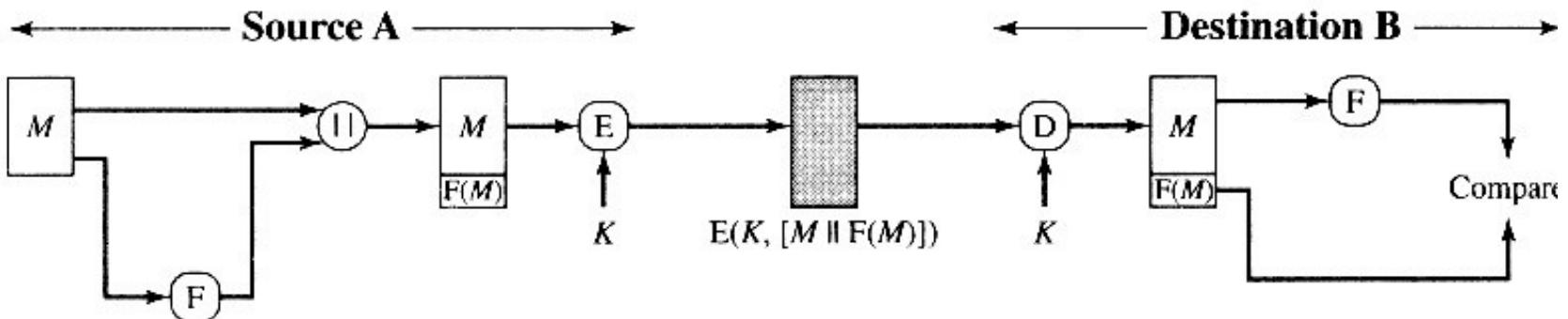
# Autenticação - Cifragem

- Provê autenticação usando algoritmos criptográficos
- Autenticação por cifragem pode ser dividida em:
  - Simétrica
  - Assimétrica
- A autenticação é baseada na manutenção dos segredos
- Chaves que devem ser protegidas

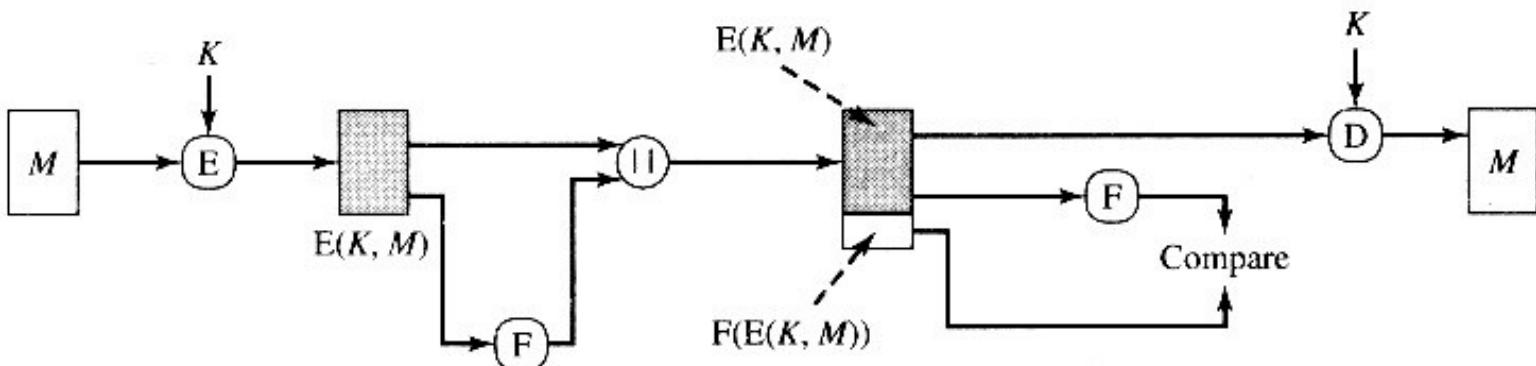
# Autenticação - Cifragem Simétrica

- Somente A e B compartilham a chave K
- Se um texto recebido por A decifra para uma mensagem inteligível usando K, A pode inferir que a mensagem veio de B
- Senão for legível, deve conter alguma estrutura que seja facilmente reconhecida:
  - Detecção de erro
  - Hash

# Autenticação - Cifragem Simétrica com Integridade



(a) Internal error control

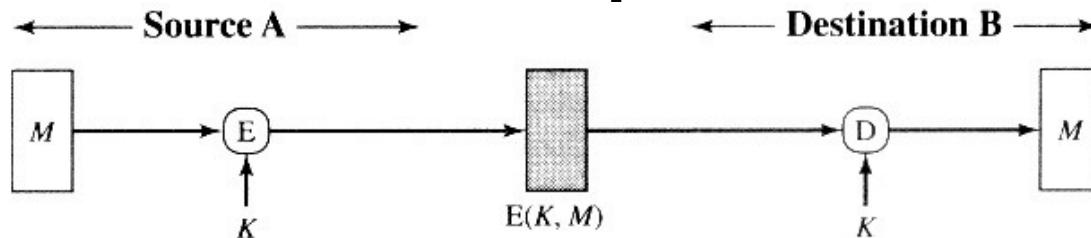


(b) External error control

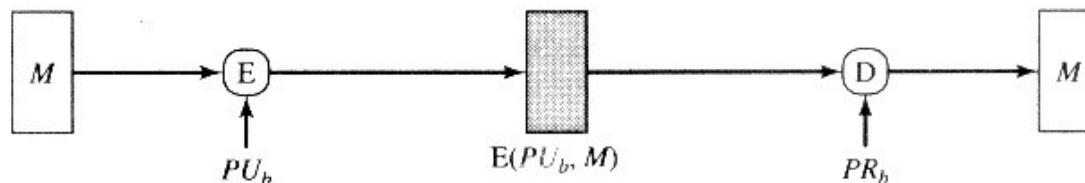
# Autenticação - Cifragem Assimétrica

- Autenticação pelo uso da chave privada
- A operação pode ser desfeita pela chave pública
- Se relacionarmos B com a chave pública que decifra uma mensagem recebida por A, este autentica B pela posse da chave privada
- Integridade normalmente feita por HASH
  - Alta complexidade de cifragem para textos grandes

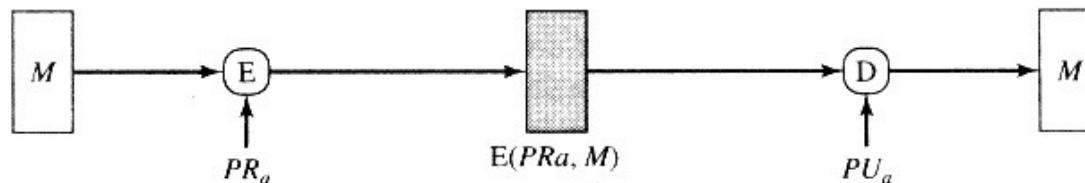
# Autenticação – Cifragem em Exemplos



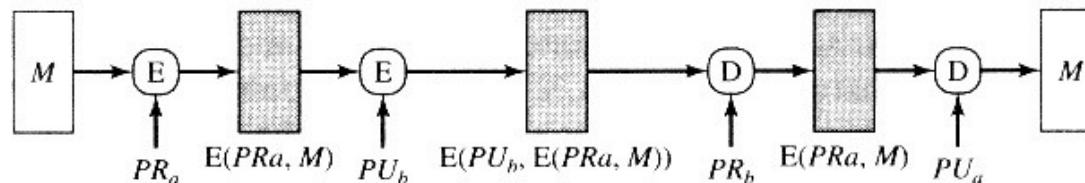
(a) Symmetric encryption: confidentiality and authentication



(b) Public-key encryption: confidentiality



(c) Public-key encryption: authentication and signature



(d) Public-key encryption: confidentiality, authentication, and signature

# Autenticação – Propriedades da Cifragem

$A \rightarrow B: E(K, M)$

- Provides confidentiality
  - Only A and B share K
- Provides a degree of authentication
  - Could come only from A
  - Has not been altered in transit
  - Requires some formatting/redundancy
- Does not provide signature
  - Receiver could forge message
  - Sender could deny message

(a) Symmetric encryption

# Autenticação – Propriedades da Cifragem

$A \rightarrow B: E(PU_b, M)$

- Provides confidentiality
  - Only B has  $PR_b$  to decrypt
- Provides no authentication
  - Any party could use  $PU_b$  to encrypt message and claim to be A

(b) Public-key (asymmetric) encryption: confidentiality

$A \rightarrow B: E(PR_a, M)$

- Provides authentication and signature
  - Only A has  $PR_a$  to encrypt
  - Has not been altered in transit
  - Requires some formatting/redundancy
  - Any party can use  $PU_a$  to verify signature

(c) Public-key encryption: authentication and signature

# Autenticação – Propriedades da Cifragem

A → B:  $E(PU_b, E(PR_a, M))$

- Provides confidentiality because of  $PU_b$
- Provides authentication and signature because of  $PR_a$

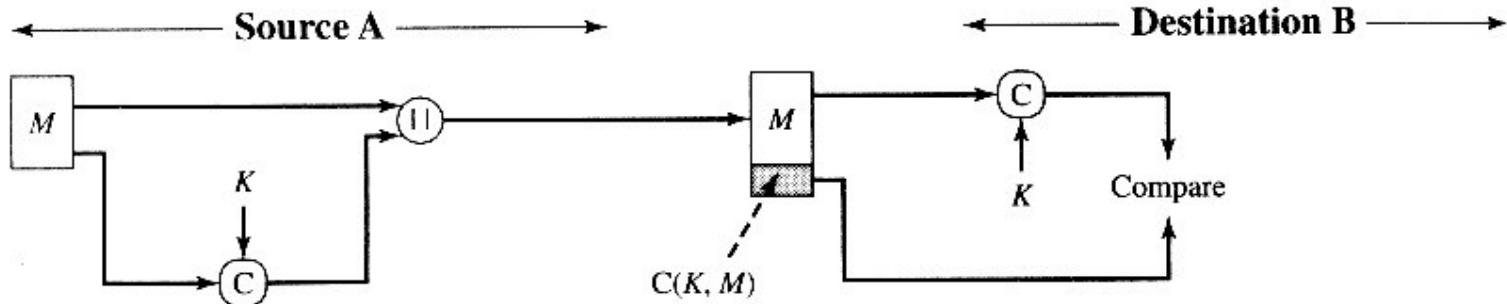
# Autenticação – Códigos de Autenticação de Mensagens

- Resumo da mensagem baseado em chave simétrica
  - $\text{MAC} = \text{C}(K, M)$
- É similar a cifragem mas não tem reversão
- Calcula-se dos dois lados usando os mesmos parâmetros para confirmar

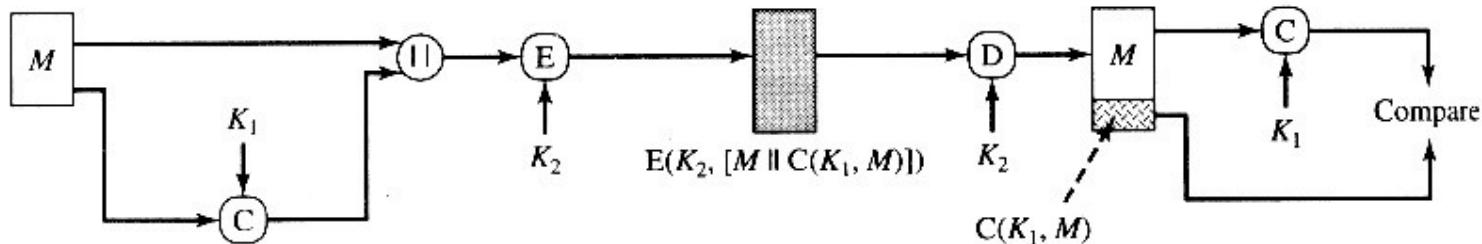
# Códigos de Autenticação de Mensagens - Quando Usar

- Mensagem enviada a vários destinatários, somente um verifica a integridade
- Mensagem muito grande para ser cifrada (processo lento), usa-se checagem MAC seletiva
- Verificação de integridade de programas
- Não é necessário sigilo
- Autenticação após decifragem
- Verificação de integridade autêntica antes da decifragem.

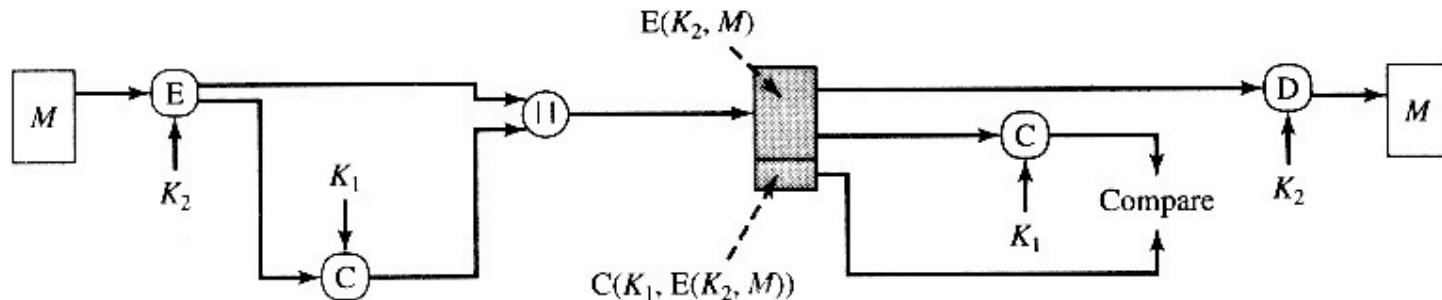
# Códigos de Autenticação de Mensagens - Uso



(a) Message authentication



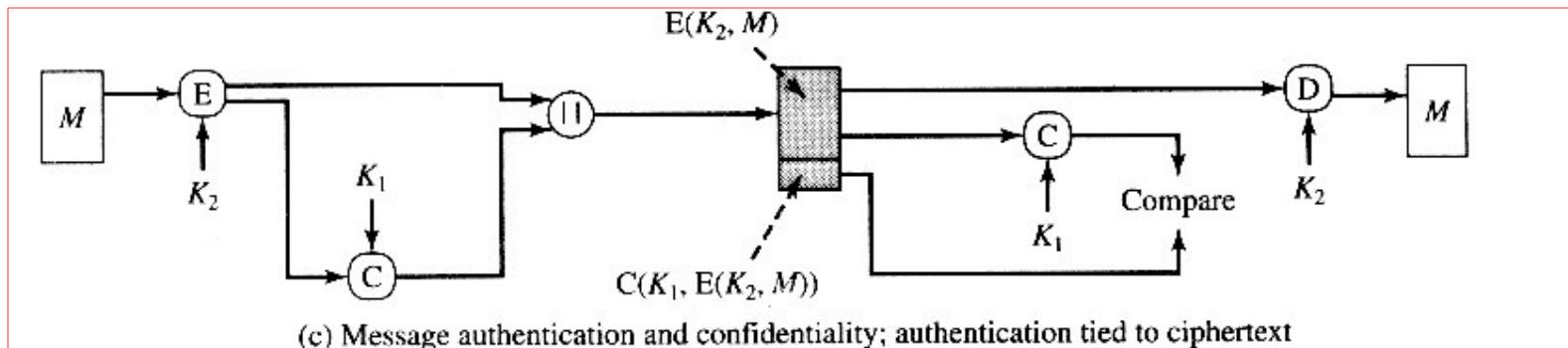
(b) Message authentication and confidentiality; authentication tied to plaintext



(c) Message authentication and confidentiality; authentication tied to ciphertext

# Códigos de Autenticação de Mensagens - Uso

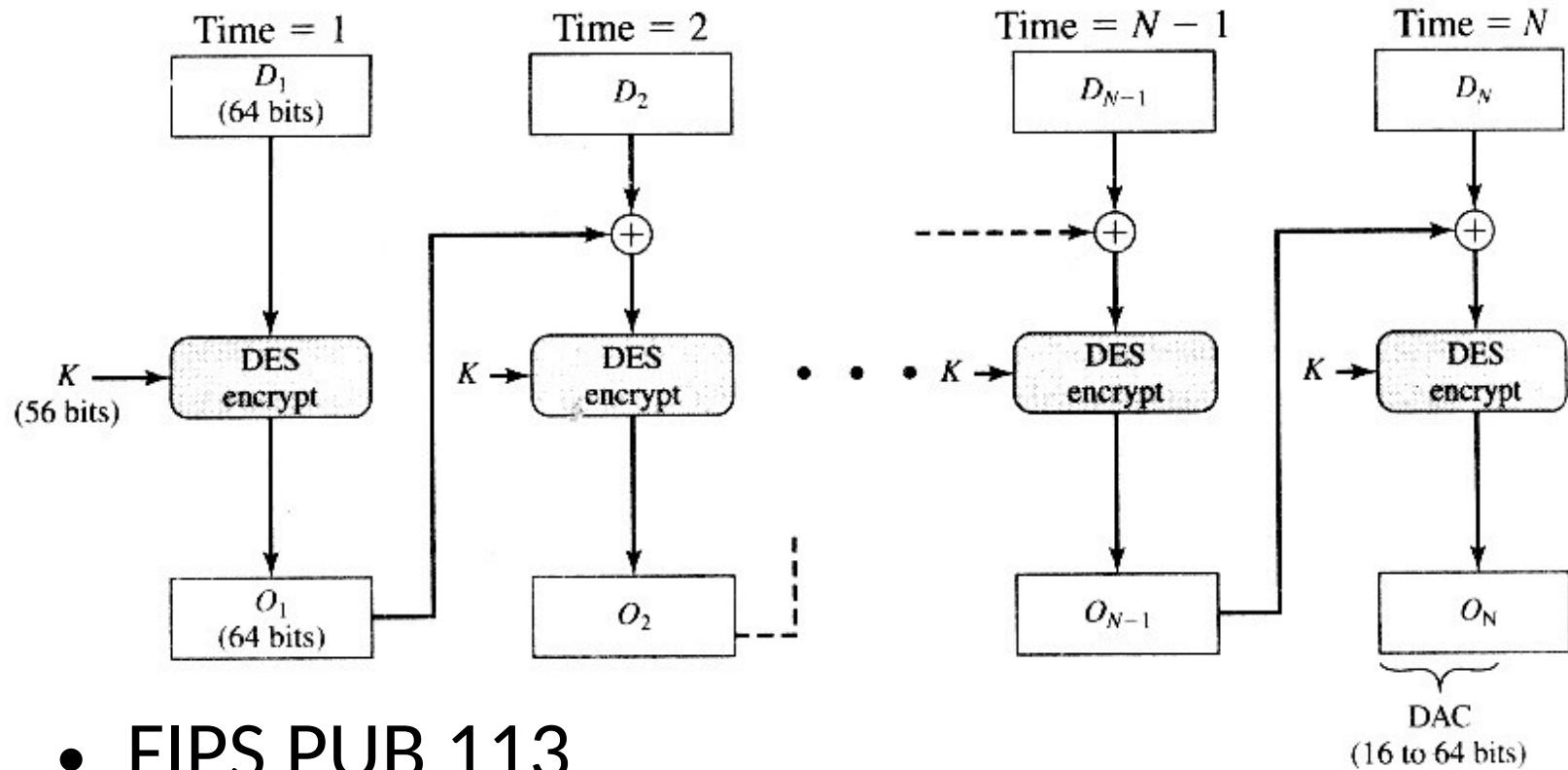
Encrypt and Mac!



# MAC – Descrição/Requisitos

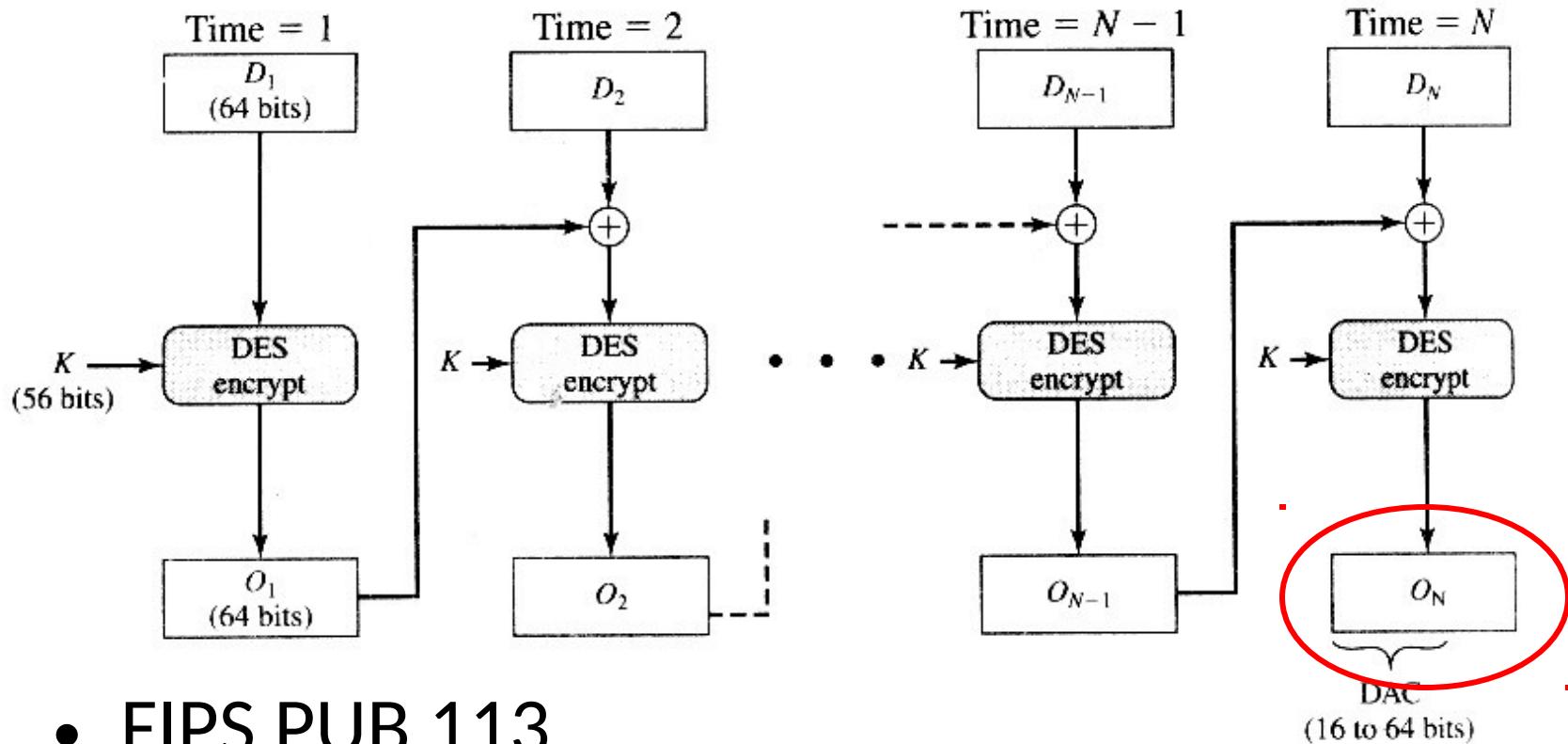
- Função de caminho único
- Requerimentos:
  - $C(K,M') = C(K,M)$  impossível para  $M, M'$  escolhido
  - Distribuição uniforme  $C(K,M') = C(K,M) \rightarrow$  Probabilidade =  $2^{-n}$ ,  $n$  = tamanho do MAC
  - Efeito avalanche

# DAC – MAC baseado em DES



- FIPS PUB 113
- Algoritmo bastante usado

# DAC – MAC baseado em DES



- FIPS PUB 113
- Algoritmo bastante usado

Resultado do DAC

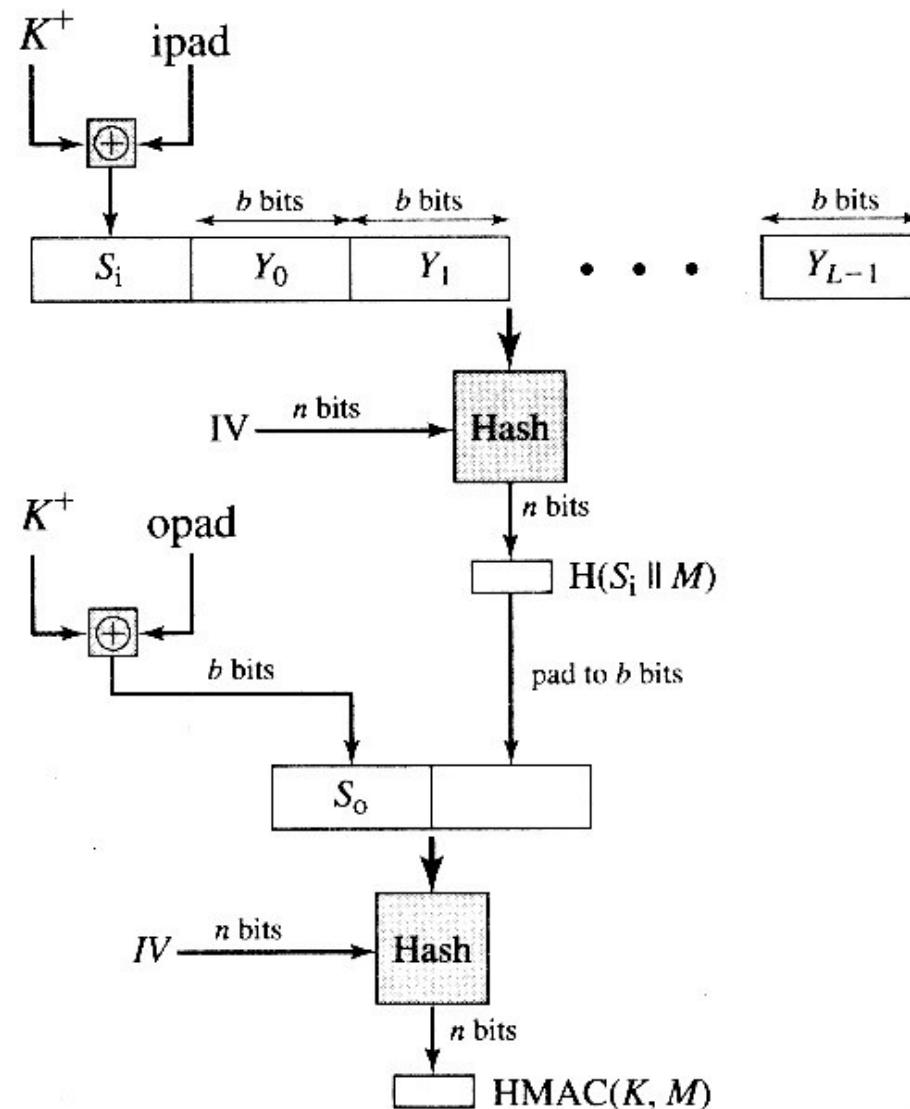
# HMAC

- MAC baseado em função HASH
- Objetivos:
  - Mais rápido que cifragem
  - Funções HASH amplamente disponíveis
- RFC 2104 /FIPS 198 → como adicionar uma chave a um HASH
- Usado em SSL e IPSEC

# HMAC – Objetivos de Projeto

- Usar funções HASH sem modificação
- Permitir trocar a função HASH
- Preservar a performance do HASH
- Usar chave de maneira simples
- Ter toda análise criptográfica baseada no função HASH

# HMAC - Estrutura



# HMAC - Estrutura

- $\text{HMAC}(K, M) = H[(K^+ \otimes \text{opad}) || H[(K^+ \otimes \text{ipad}) || M]]$ 
  - $b \rightarrow$  tamanho do bloco em bits
  - $K \rightarrow$  Chave,  $K^+ \rightarrow$  Chave estendida até  $b$
  - $\text{ipad} = 0x36$  repetido  $b/8$
  - $\text{opad} = 0x5C$  repetido  $b/8$
  - $\text{IV} \rightarrow$  valor de inicialização do HASH
  - opad e ipad geram bits alternados na chave

# HMAC Pseudo-Código

```
function hmac (key, message)
    if (length(key) > blocksize) then
        key = hash(key) // keys longer than blocksize are shortened
    end if
    if (length(key) < blocksize) then
        key = [0x00 * (blocksize - length(key)) || key] // keys shorter than blocksize are zero-padded
    end if
    o_key_pad = [0x5c * blocksize] ⊕ key // Where blocksize is that of the underlying hash function
    i_key_pad = [0x36 * blocksize] ⊕ key
    return hash(o_key_pad || hash(i_key_pad || message))
end function
```

# Assinatura Digital

- É um mecanismo de autenticação que possibilita o criador da mensagem ser identificado
- Prova de não-repúdio
- Pode ser direta ou arbitrada

# Assinatura Digital - Requisitos

- A assinatura deve depender de cada bit da mensagem
- Deve usar algo único do criador
- Deve ser fácil de produzir, reconhecer e verificar
- Dever ser computacionalmente não forjável
- Dever ser possível reter uma cópia

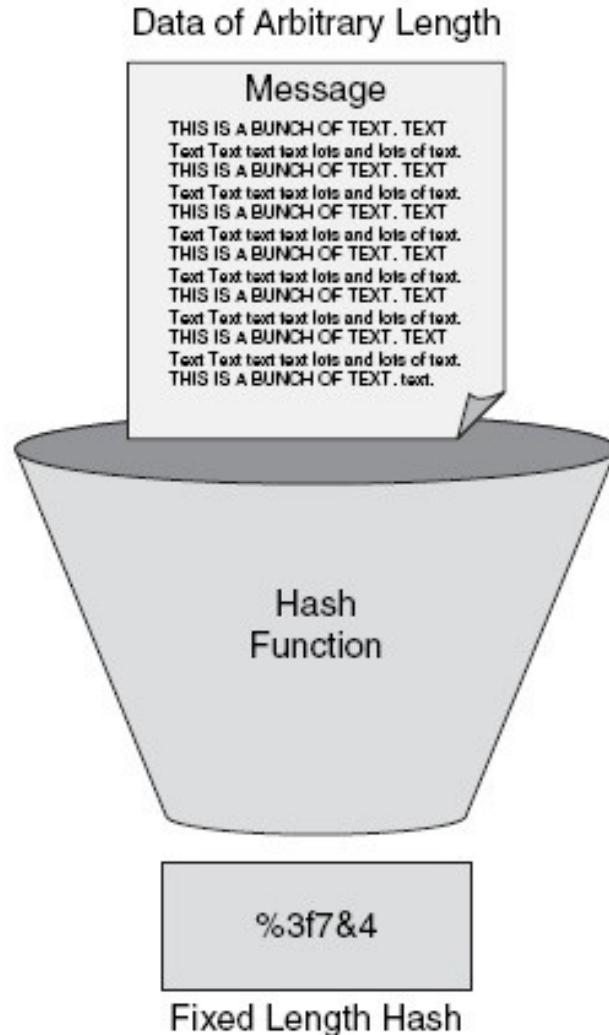
# Assinatura Digital Direta

- Envolve só origem e destino
- Cifragem do hash com a chave privada
- Validade atrelada a chave privada
- Negar é alegar a perda da chave
- É normalmente incluído carimbo de tempo

# Assinatura Digital Arbitrada

- Tentar resolver o problema da assinatura direta
- Envolve origem, destino e árbitro
- O árbitro checa a mensagem e assina junto dando o seu carimbo de tempo
- O árbitro provê uma prova de verificação
- O árbitro deve ser confiável por ambos

# Integridade – Funções HASH



# Integridade – Funções HASH

- São similares a MAC mas não tem chaves
- Provê propriedades como efeito avalanche
- Prove uma camada de integridade diferente da autenticação

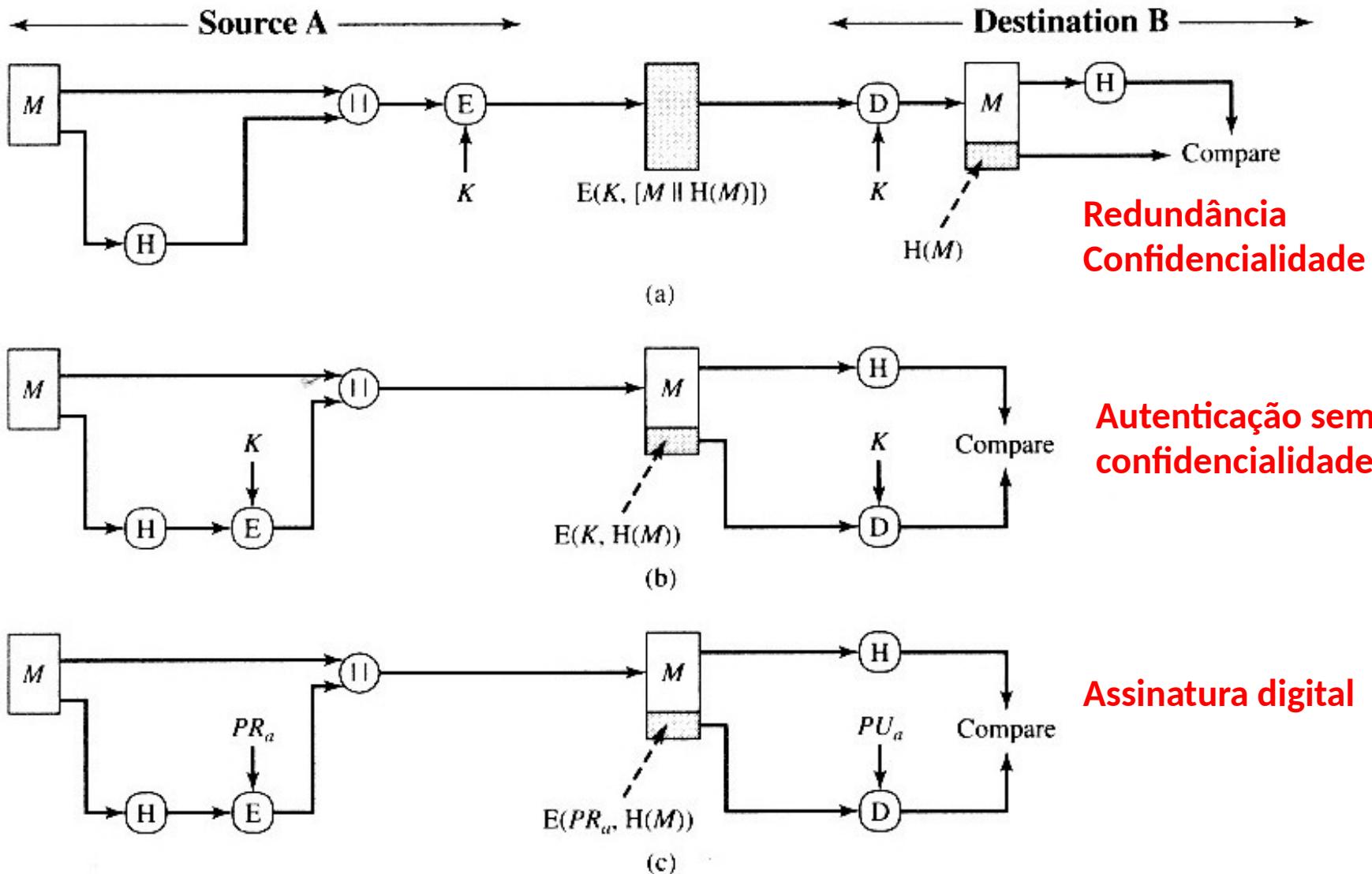
# Integridade – Funções HASH

- Computacionalmente não praticável achar:
  - Um dado que coincida com um hash pré-especificado (não é inversível)
  - Dois dados que tenham o mesmo hash (colisão)
- Melhor algoritmo para ambos: força bruta

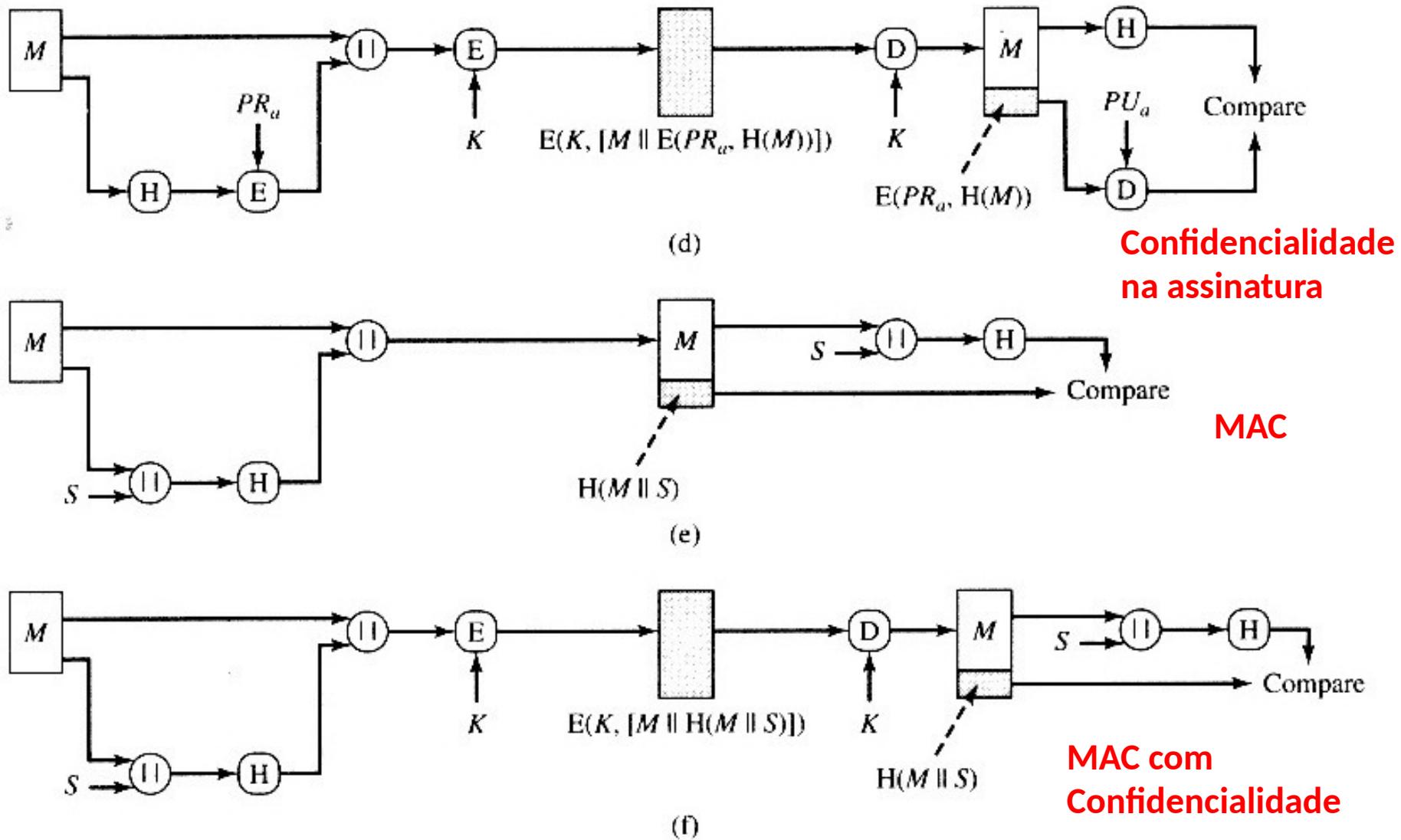
# HASH – Descrição/Requisitos

- Função de caminho único,  $M$  variável,  $H(M)$  Fixo
- Produz uma impressão digital de um arquivo
- Requisitos:
  - Fácil de computar  $H(M)$  para qualquer  $M$
  - É impossível achar  $M$  tendo  $H(M) \rightarrow$  caminho único
  - Dado  $M_1$  e  $M_2$  não deve ser possível computar  $H(M_1) = H(M_2)$  para  $M_1 \neq M_2$
- Pseudo-aleatoriedade

# Funções HASH - Usos



# Funções HASH - Usos



# Funções HASH – Quando Usar

- Uso em funções MAC
- Verificação de integridade
- Indexação de arquivos (estruturas de dados)
- Armazenamento de senhas
  - salted password hashing
  - *Token* de acesso

# Paradoxo do Aniversário

- Considere uma sala com 30 alunos
- Professor escolhe uma data
  - ~8% de chance de um aluno ter nascido nesta data.
- Professor solicita data de nascimento dos alunos
  - 70% de chance de dois alunos terem a mesma data de nascimento

Obs: Ano com 365 dias

# Paradoxo do Aniversário

- Em um grupo de 23 pessoas existe uma probabilidade de 50% para que duas destas façam aniversário no mesmo dia.
- A chance de encontrar um valor repetido em um conjunto de 0 a N-1 excede 50% depois de aprox.  $\sqrt{N}$  tentativas

# Paradoxo do Aniversário

- A está preparado para assinar x
- Atacante gera  $2^{m/2}$  variações de uma mensagem x com o mesmo significado (m é o tamanho do hash)
- Atacante gera  $2^{m/2}$  variações fraudulentas y
  - A probabilidade de encontrar algum y com hash igual ao de algum x é maior que 50%
  - Se oferece a versão variada (x) para assinatura e se usa a versão fraudulenta (algum y).

# Paradoxo do Aniversário M2<sup>37</sup>

Dear Anthony,

{ This letter is } to introduce { you to } { Mr. } Alfred { P. }  
I am writing { to you } { -- } { -- }  
Barton, the { new } { chief } jewellery buyer for { our }  
newly appointed { senior } { the }  
Northern { European } { area } He { will take } over { the }  
Europe { division } has taken { -- }  
responsibility for { all } our interests in { watches and jewellery }  
the whole of { jewellery and watches }  
in the { area } Please { afford } him { every } help he { may need }  
region { give } { all the } needs  
to { seek out } the most { modern } lines for the { top } end of the  
find { up to date } { high }  
market. He is { empowered } to receive on our behalf { samples } of the  
authorized { specimens }  
{ latest } { watch and jewellery } products, { up } to a { limit }  
{ newest } { jewellery and watch } subject { maximum }  
of ten thousand dollars. He will { carry } a signed copy of this { letter }  
hold { document }

# Resistência de Hashes

- Para tamanho de hash  $m$ :

Técnica	Esforço
Reversão ( $h \rightarrow y \mid H(y) = h$ )	$2^m$
Colisão fraca ( $x \rightarrow y \neq x \wedge H(y) = H(x)$ )	$2^m$
Colisão forte ( $x, y \mid H(x) = H(y)$ )	$2^{(m/2)}$