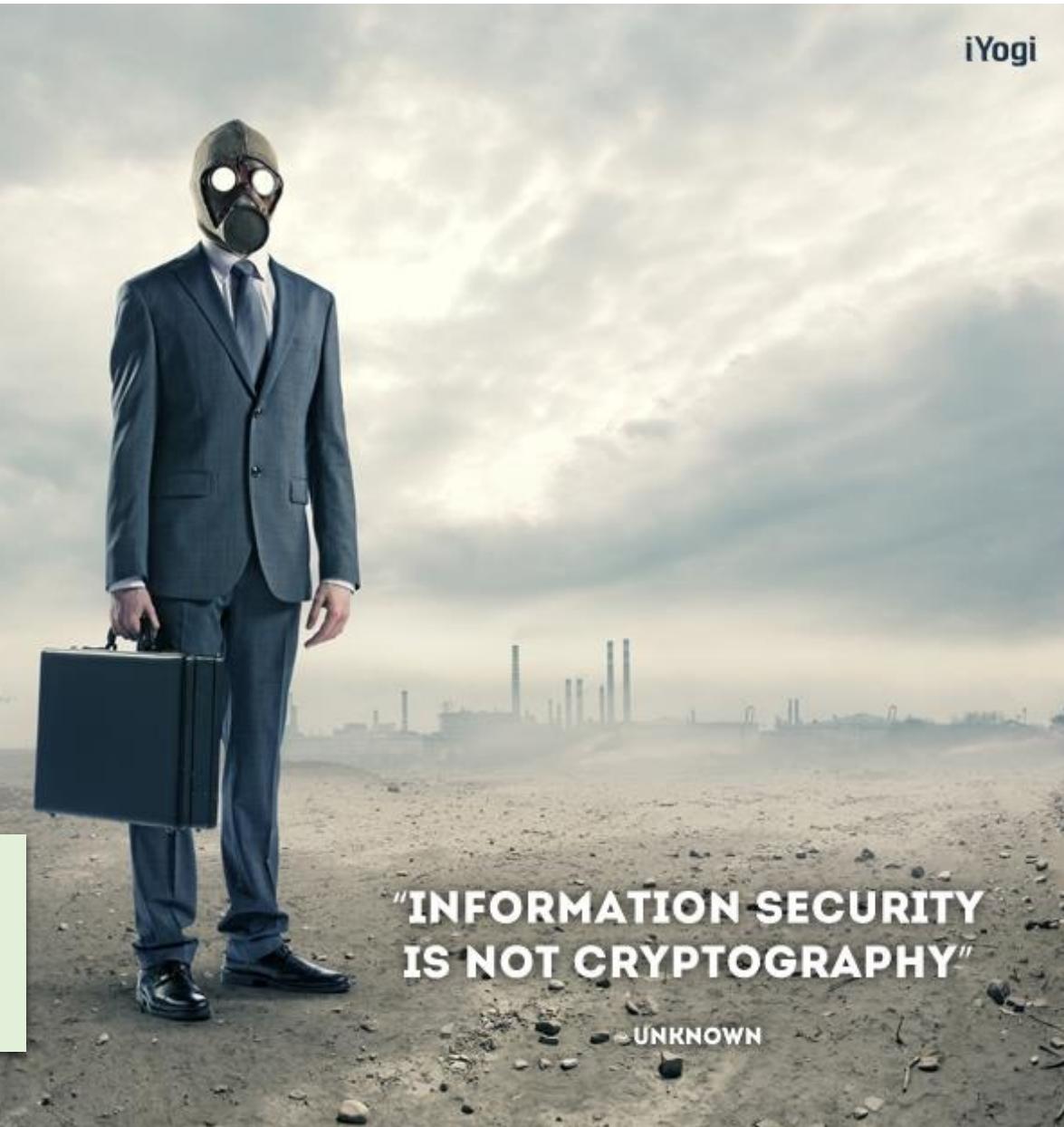


IMD0703 – Segurança de Redes

Criptografia Simétrica. Cifra de Bloco. DES. AES.





Objetivo desta aula (Cont.)

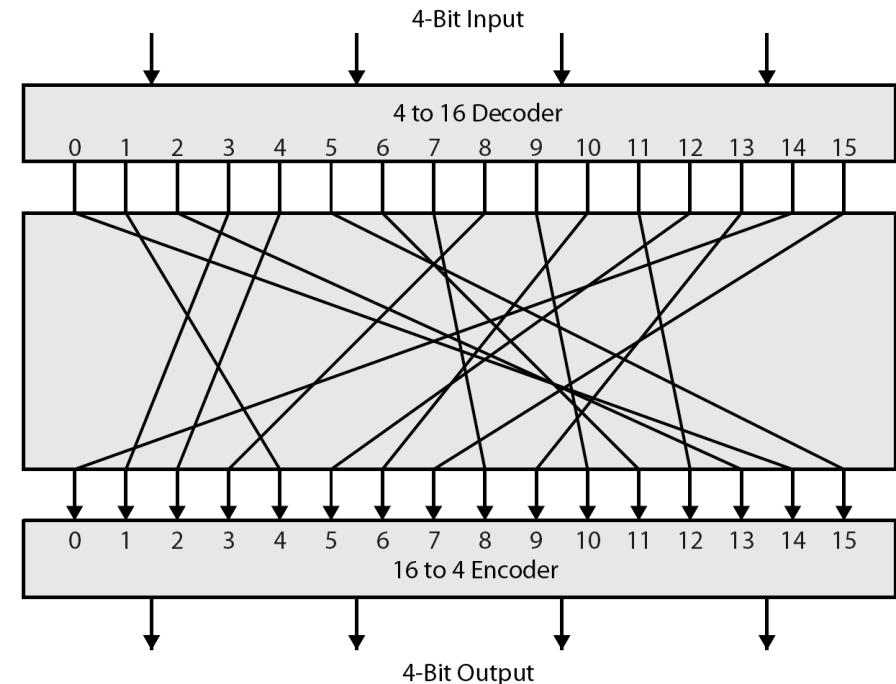
- Geral
 - Contextualizar historicamente o desenvolvimento da criptografia, apresentando as técnicas de criptografia simétrica e assimétrica, os principais algoritmos e suas aplicações
- Específicos
 - Conhecer os fundamentos da criptografia em um contexto histórico;
 - Entender os principais conceitos relacionados à criptografia tradicional;
 - Compreender os princípios de funcionamento das técnicas de criptografia;
 - Conhecer os principais algoritmos criptográficos de criptografia simétrica e suas aplicações

Cifras de Bloco (*Block Ciphers*)

- Enquanto cifras orientadas a fluxo (*stream ciphers*) operam sobre um bit ou byte por vez, numa cifra de bloco o texto claro dado como entrada é **processado em blocos de tamanho fixo**, produzindo um **bloco de cifra de mesmo tamanho**
 - Funciona como uma substituição em uma quantidade grande de bits (64 ou mais)
 - Têm maior aplicabilidade que as cifras de fluxo
- Muitas das cifras modernas operam em bloco
 - DES - Data Encryption Standard
 - TDES - Triple DES
 - AES - Advanced Encryption Standard

Cifra de Bloco ideal e Cifra de Feistel

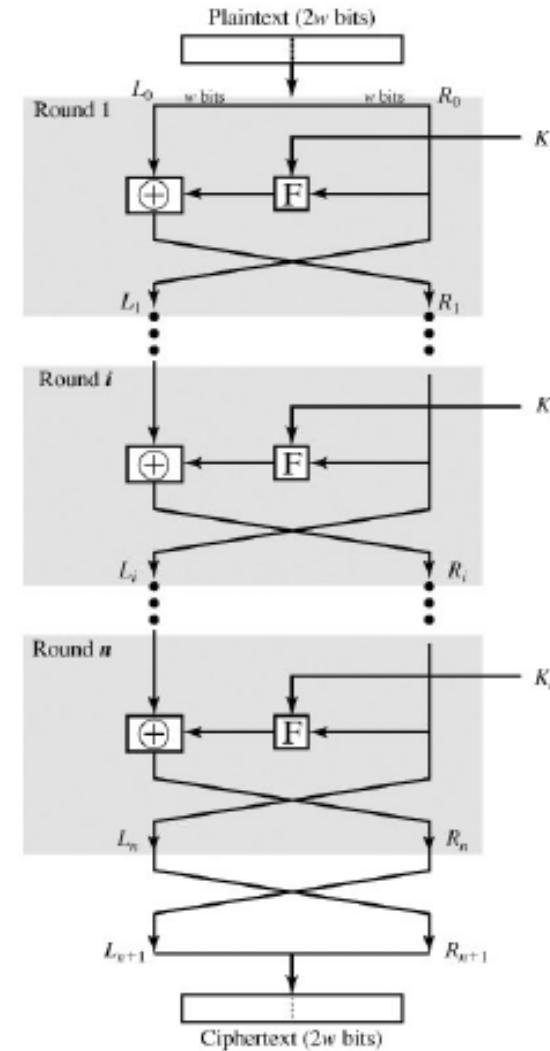
- Na cifra de bloco ideal utiliza o conceito de cifra de produto, apresentando todas as transformações possíveis para um conjunto de bits n (ou seja, 2^n)
- Feistel propôs uma aproximação para a cifra de bloco ideal, conhecida como cifra de produto
 - Uma cifra de bloco com chave de tamanho k bits e blocos de n bits que permite um total de 2^k transformações
 - Baseia-se na execução de duas ou mais cífras em sequência de tal forma que o resultado final seja criptograficamente mais forte do que qualquer uma das cífras intermediárias



Cifra de Bloco Ideal

Cifra de Feistel

- Esse modelo é utilizado pela maioria das cifras de bloco simétrico utilizados
- Funcionamento
 - Entrada de $2w$ bits e uma chave K
 - Bloco de Texto claro é dividido em 2 metades (L_0 , R_0)
 - Duas metades passam por n rodadas e são combinados para formar o texto cifrado
 - A realimentação de cada rodada é feita com L e R da rodada anterior + uma chave K_i derivada da chave K
 - F é a função complexa que é aplicada a metade direita dos dados R e depois realiza um OU exclusivo com L



Cifra de Feistel

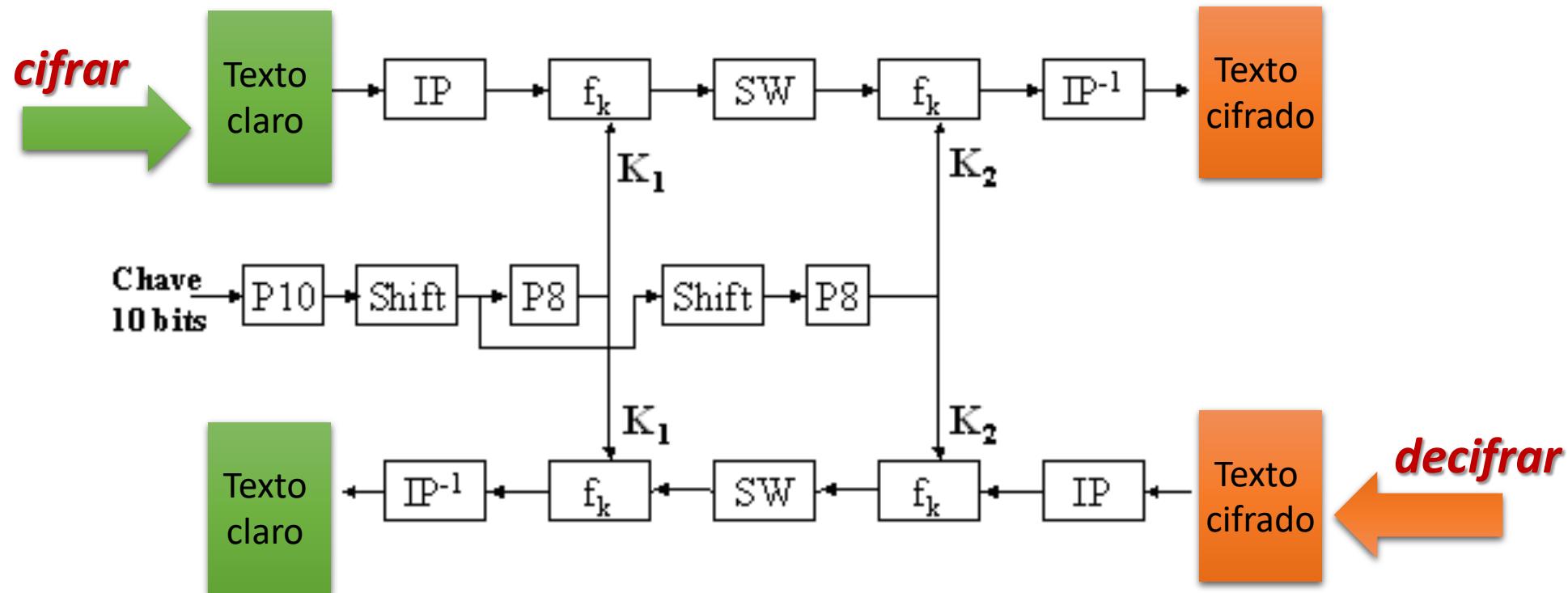
- Parâmetros:
 - **Tamanho do Bloco:** quanto maior mais seguro porém mais lento (**64 no DES e 128 no AES**)
 - **Tamanho da chave:** quanto maior, mais seguro, porém mais lento
 - Chaves de 64bits já não são consideradas seguras, sendo 128 bits o tamanho comum, mas não ideal.
 - **Numero de rodadas:** esse é o **fator crítico** desse tipo de cifra (**N=16 no DES**)
 - **Algorítmo de geração de subchaves:** maior complexidade neste algoritmo levará a maior dificuldade de criptosanálise
 - **Função F:** quanto mais complexa geralmente mais resistente à criptoanálise
- Utiliza o conceito de Difusão e Confusão para dificultar a criptoanálise estatística
 - Difusão: a estatística do texto claro é dissipada em estatísticas de longa duração no texto cifrado (ex: um caracter do texto claro deve afetar a estatística de mais de um caracter do texto cifrado)
 - Confusão: estatísticas do texto cifrado e o valor da chave de encriptação devem ser complexos

Simple DES (S-DES)

- O S-DES foi criado pelo professor Edward Shaefer da Universidade de Santa Clara com o objetivo de simplificar o ensino do funcionamento do DES
- O algoritmo de encriptação envolve cinco funções:
 - permutação inicial (**IP**)
 - a função complexa chamada de f_k , que envolve permutação e substituição dependente da chave;
 - a simples permutação de troca de duas metade dos dados (**SW**)
 - a função f_k novamente
 - finalmente é aplicada a função inversa da permutação inicial (**IP⁻¹**)

Simple DES (S-DES)

- Esquema do S-DES :
 - IP - Permutação Inicial, f_k - Função complexa, SW - Permutação simples, IP^{-1} - Inversa da permutação



Simple DES (S-DES)

- Formulação:

$$\text{texto cifrado} = \text{IP}^{-1} (f_{k_2} (\text{SW}(f_{k_1}(\text{IP}(\text{texto claro})))))$$

$$\text{texto claro} = \text{IP}^{-1} (f_{k_1} (\text{SW}(f_{k_2}(\text{IP}(\text{texto cifrado})))))$$

Onde:

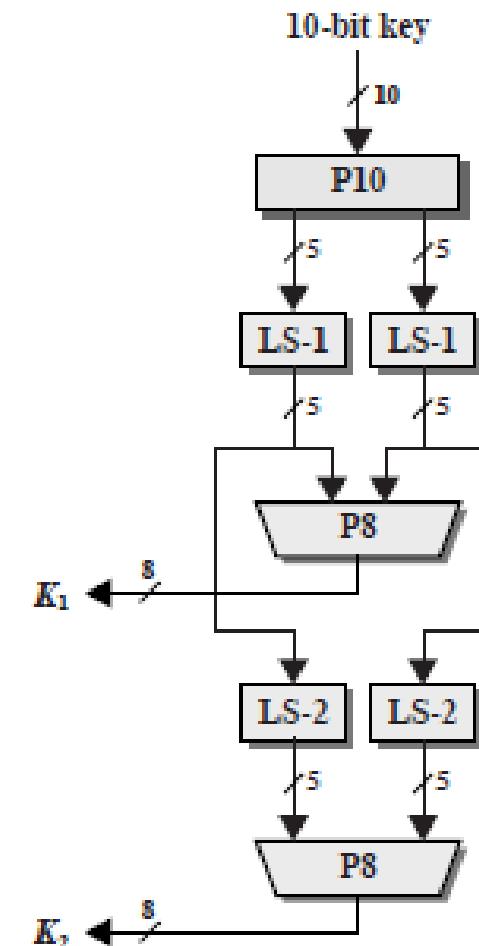
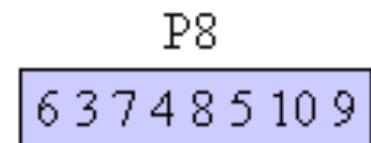
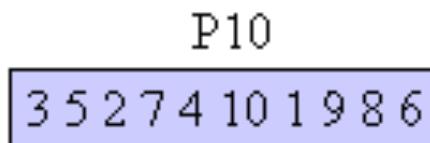
$$\left\{ \begin{array}{l} K_1 = P8(\text{Shift}(P10(\text{Key}))) \\ K_2 = P8(\text{Shift}(\text{Shift}(P10(\text{Key})))) \end{array} \right.$$

Simple DES (S-DES)

- Geração da chave

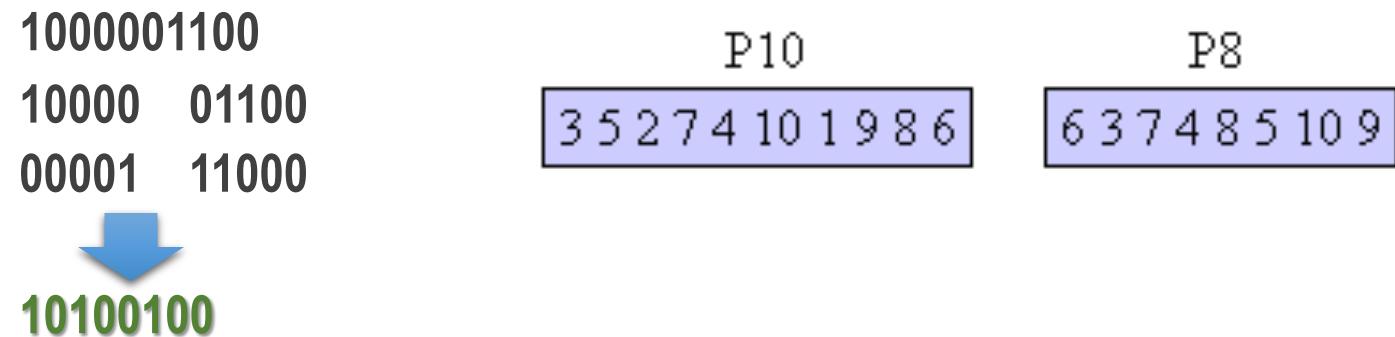
Chave = (k₁, k₂, k₃, k₄, k₅, k₆, k₇, k₈, k₉, k₁₀)

P₁₀(Chave) = (k₃, k₅, k₂, k₇, k₄, k₁₀, k₁, k₉, k₈, k₆)



Simple DES (S-DES)

- Exemplo de geração de chaves
- Chave K : **1010000010**
- K_1 :
 - Permutação inicial (P10) 1000001100
 - Separação 10000 01100
 - Rotação a esquerda LS-1: 00001 11000
 - Permutação SW (P8) ↓
 - O resultado é a subchave K_1 10100100



Simple DES (S-DES)

- Exemplo de geração de chaves

- Para K_2 ,

- pega-se o resultado de LS-1
- Rotação a esquerda LS-2 duas posições
- Permutação SW (P8)
- O resultado é a subchave K_2 é

00001 11000

00100 00011

01000011

P10

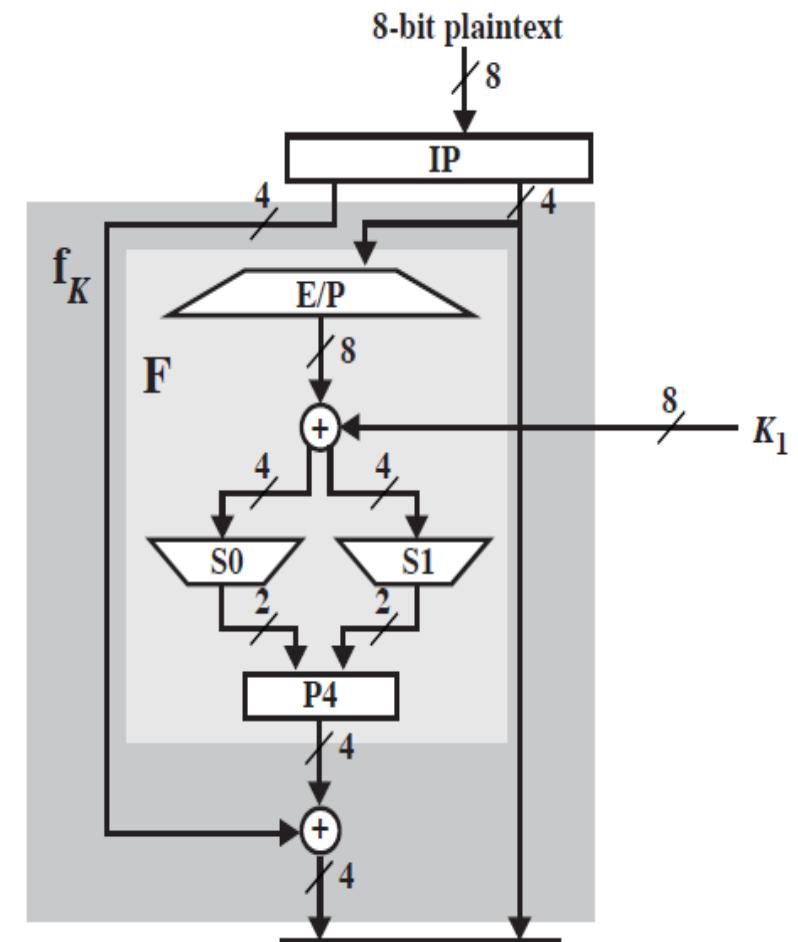
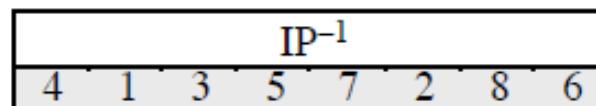
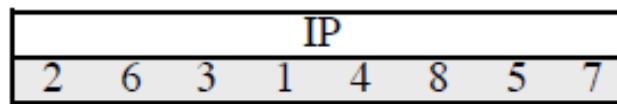
3 5 2 7 4 10 1 9 8 6

P8

6 3 7 4 8 5 10 9

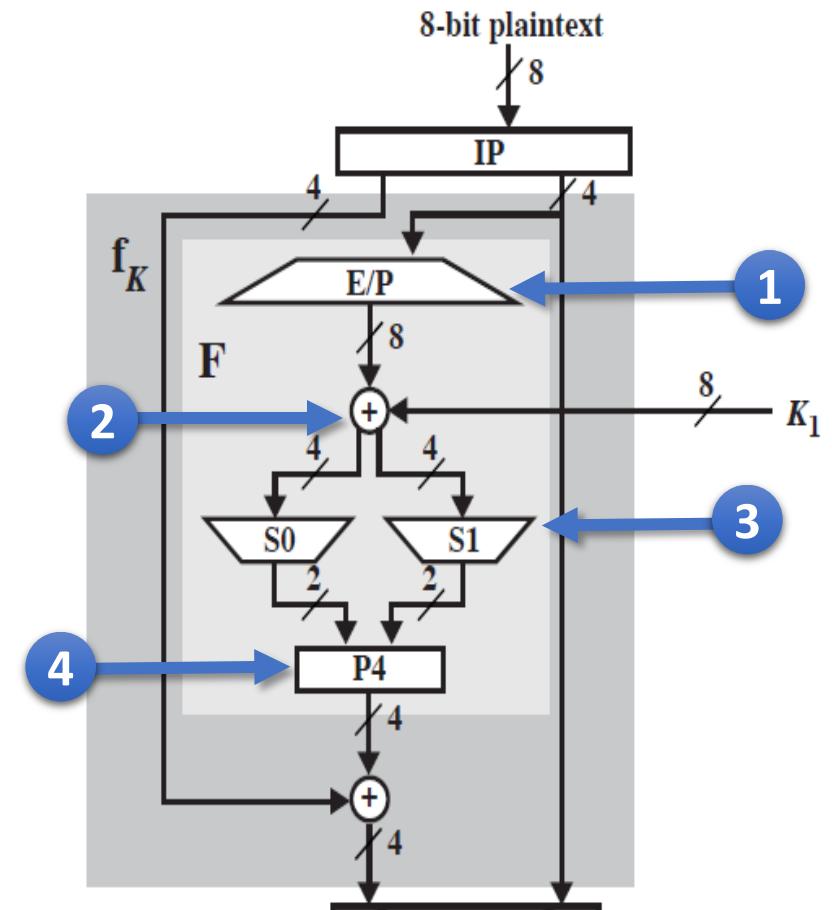
Simple DES (S-DES)

- O componente complexo do S-DES é a função f_K
 - Combinação de funções de permutação e substituição
 - L e R são os quatro bits a esquerda e os quatro bits a direita dos 8 bits que entraram na função
 - F é a função que executa as operações com os dados R e a sub-chave SK_n
- A permutação inicial (IP) e final (IP^{-1}) que ocorre durante o processo de cifragem e decifragem, dos 8 bits processados, obedece a seguinte tabela:



Simple DES (S-DES)

- A função $F(R, SK)$
 1. Ocorre uma operação de **expansão/permutação nos 4 bits de R**, transformando-o em 8 bits
 2. É realizada uma operação de **OU exclusivo com a subchave (K_1)**
 3. É então **separado em dois grupos de 4 bits cada**, que passam por uma operação na **caixa S (S0 e S1)**
 - A caixa S tem uma entrada de 4 bits e uma saída de 2 bits
 4. O **produto da caixa S** é concatenado e sofre uma **permutação**, de acordo com a **régua P4**



Simple DES (S-DES)

- Função $F(R, SK)$

1) 4 bits de entrada: 0101

$$E/P = 10101010$$

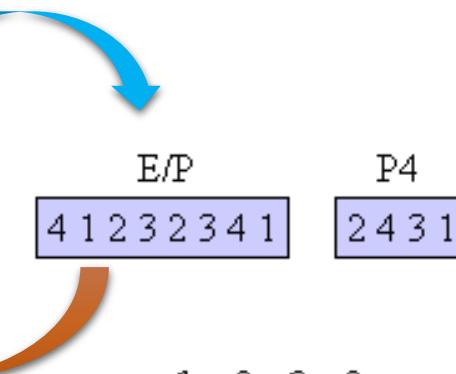
2) XOR

$$E/P = 10101010$$

$$K_1 = \underline{10000001}$$

$$R = \underline{00101011}$$

Separado: **0010** **1011**

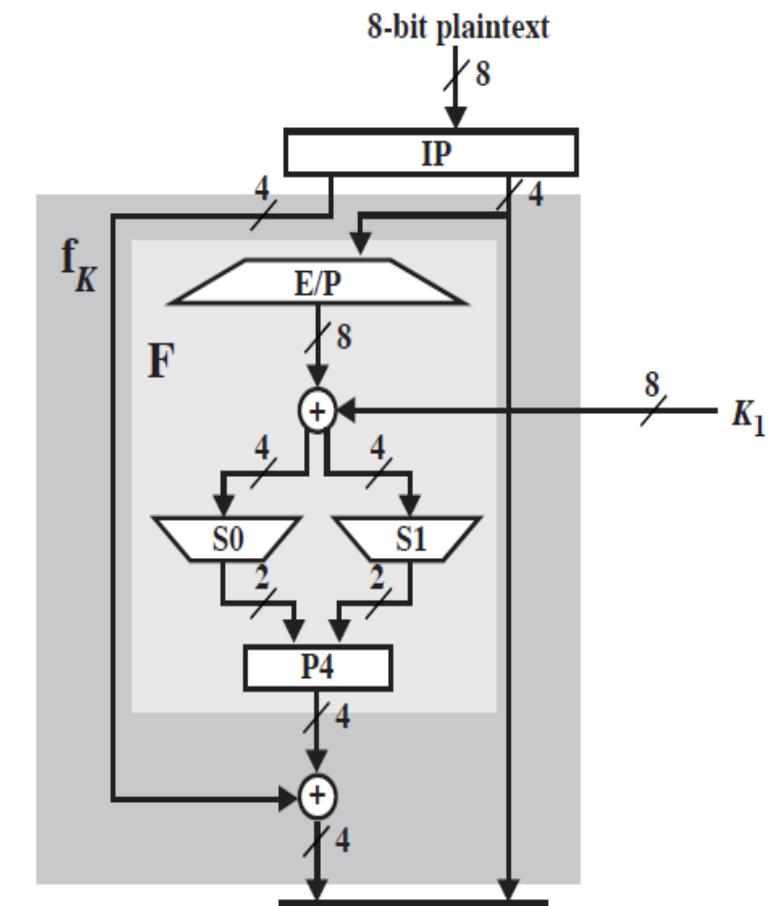


1	0	3	2
S0 = 3	2	1	0
0	2	1	3
3	1	3	2

XOR	1	0
1	0	1
0	1	0

1	1	2	3
S1 = 2	0	1	3
3	0	1	0
2	1	0	3

XOR	1	0
1	0	1
0	1	0



Simple DES (S-DES)

- Função $F(R, SK)$

3) 0010 1011

Linha ($b_1 \& b_4$)

00 = 0 11 = 3

Coluna ($b_2 \& b_3$)

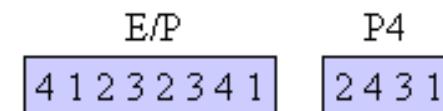
01 = 1 01 = 1

S0=00 S1=01

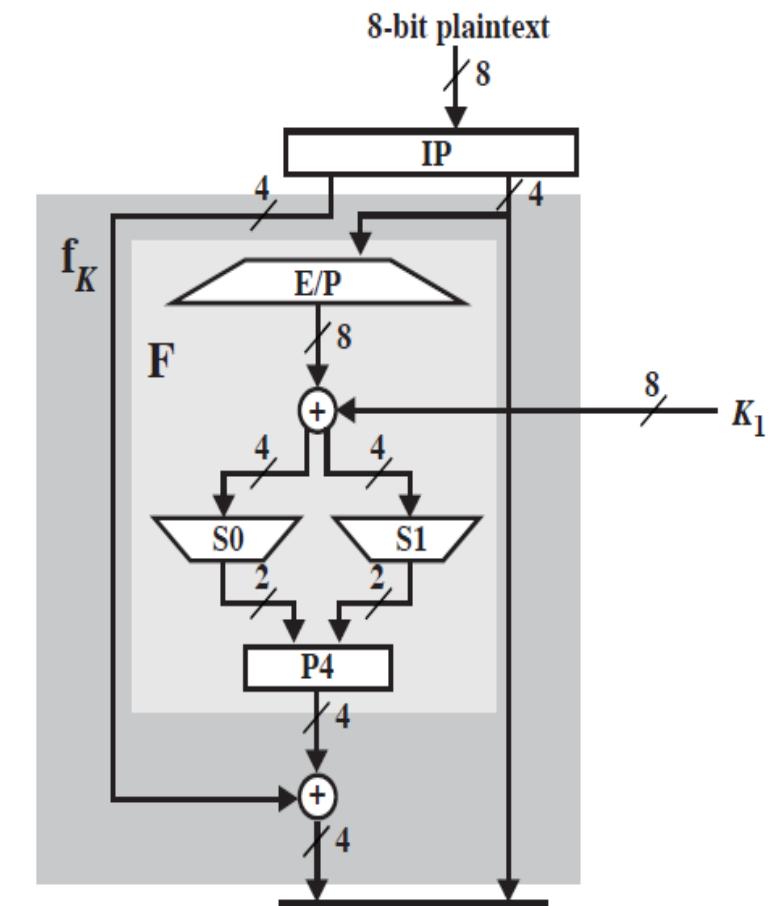
Resultado: 0001

4) Entra : 0001

Saída(P4) = 0100

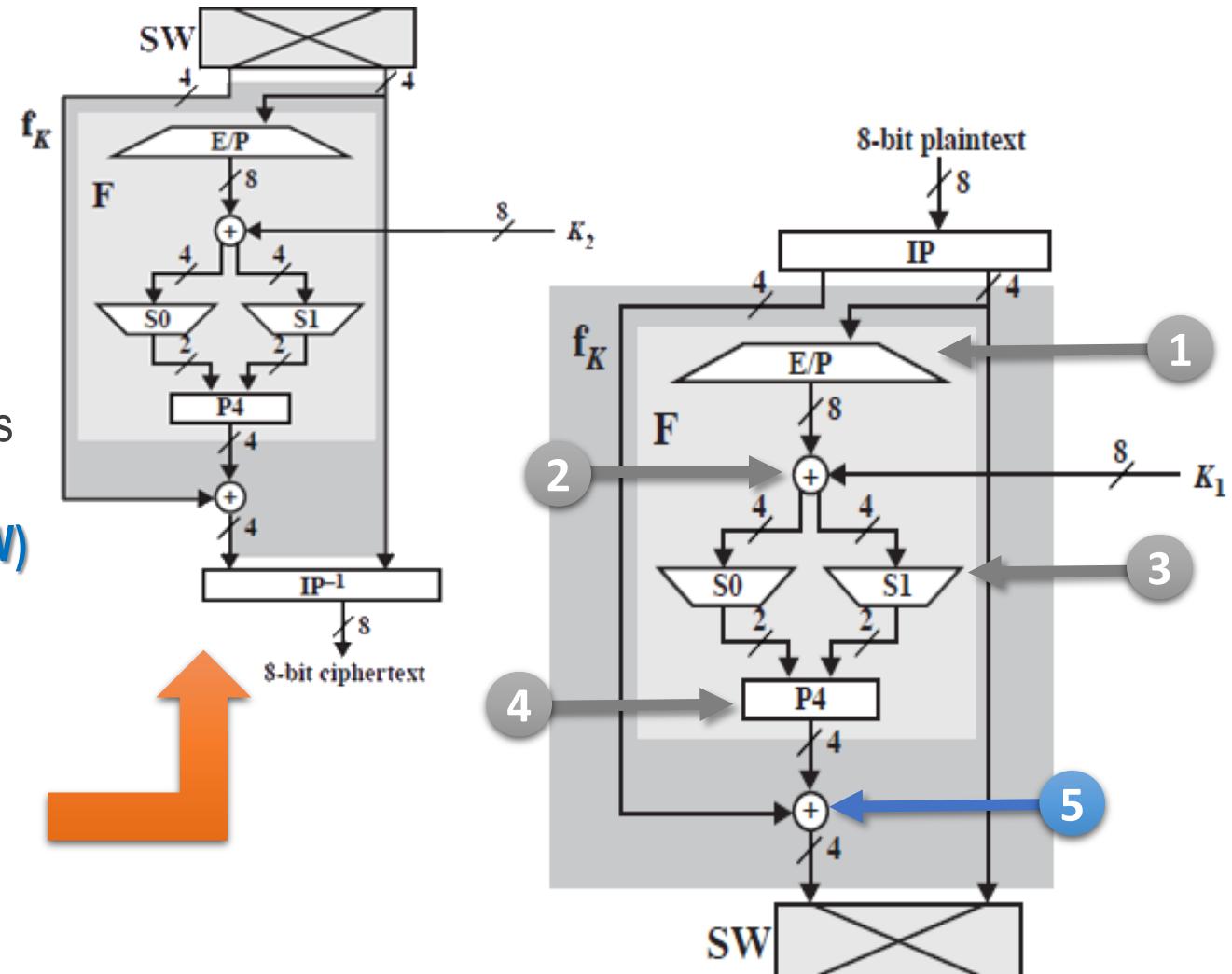


1	0	3	2
S0 = 3	2	1	0
0	2	1	3
3	1	3	2
1	1	2	3
S1 = 2	0	1	3
3	0	1	0
2	1	0	3



Simple DES (S-DES)

- A função f_k
 - O resultado de P4 sofre uma operação de OU exclusivo (**XOR**) com L
 - A função **f_k somente altera os 4 bits a esquerda**, deixando inalterados os 4 bits da direita
 - O próximo passo é a função **Switch (SW)**
 - A função SW executa uma transposição, onde os 4 bits da direita serão os quatro bits da esquerda e vice versa.
 - Esta **mesma sequência de operações são realizadas novamente, utilizando desta vez a chave K_2**

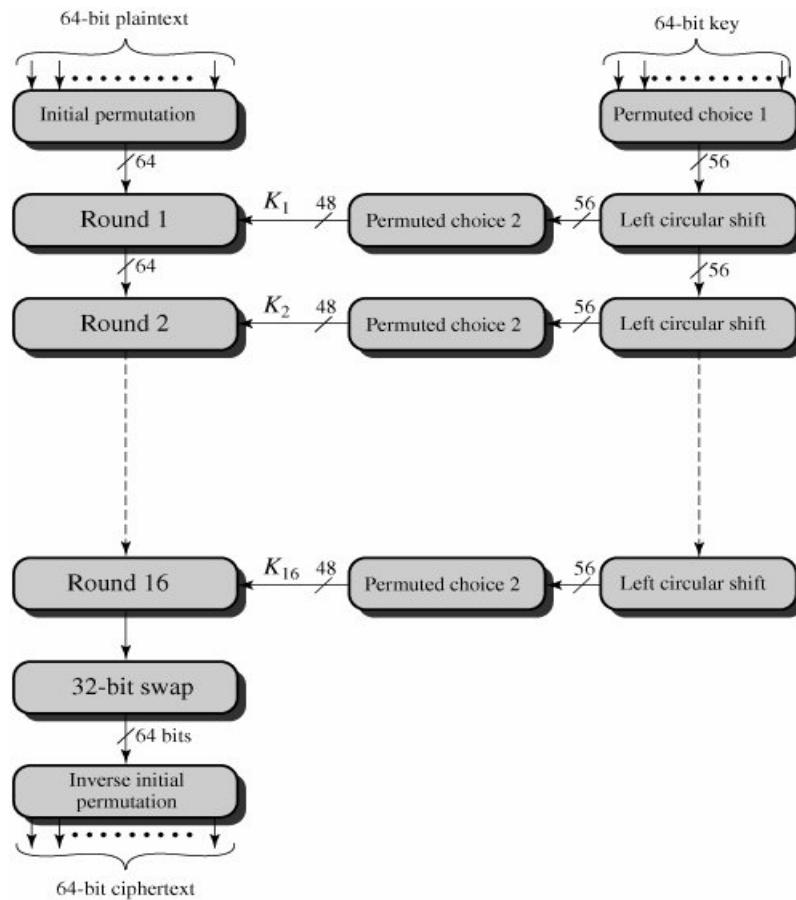


Data Encryption Standard – DES/DEA

- **DEA** (*Data Encryption Algorithm*) foi adotado pelo *National Institute of Standards and Technology* (NIST) em 1977
 - Passou então a ser chamado de **DES** (*Data Encryption Standard*)
- **Dados** são codificados em **blocos de 64 bits**
 - (8 bits do S-DES)
- **Chave** tem tamanho **56 bits**
 - (10 bits do S-DES)
- **16 Rodadas** com **16 subchaves** geradas

Data Encryption Standard – DES/DEA

- Visão Geral



(a) Initial Permutation (IP)

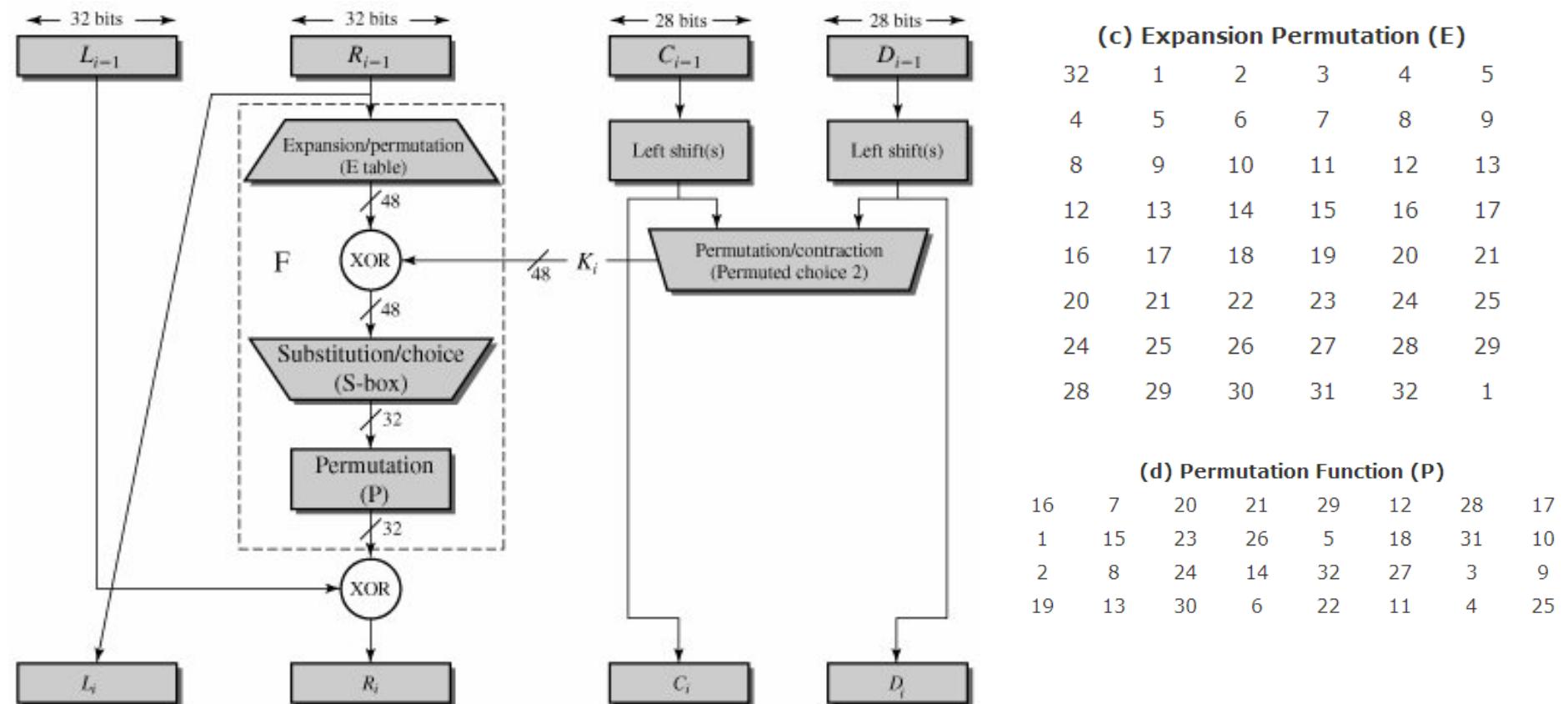
58	50	42	34	26	18	10	2
60	52	44	36	28	20	12	4
62	54	46	38	30	22	14	6
64	56	48	40	32	24	16	8
57	49	41	33	25	17	9	1
59	51	43	35	27	19	11	3
61	53	45	37	29	21	13	5
63	55	47	39	31	23	15	7

(b) Inverse Initial Permutation (IP¹)

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

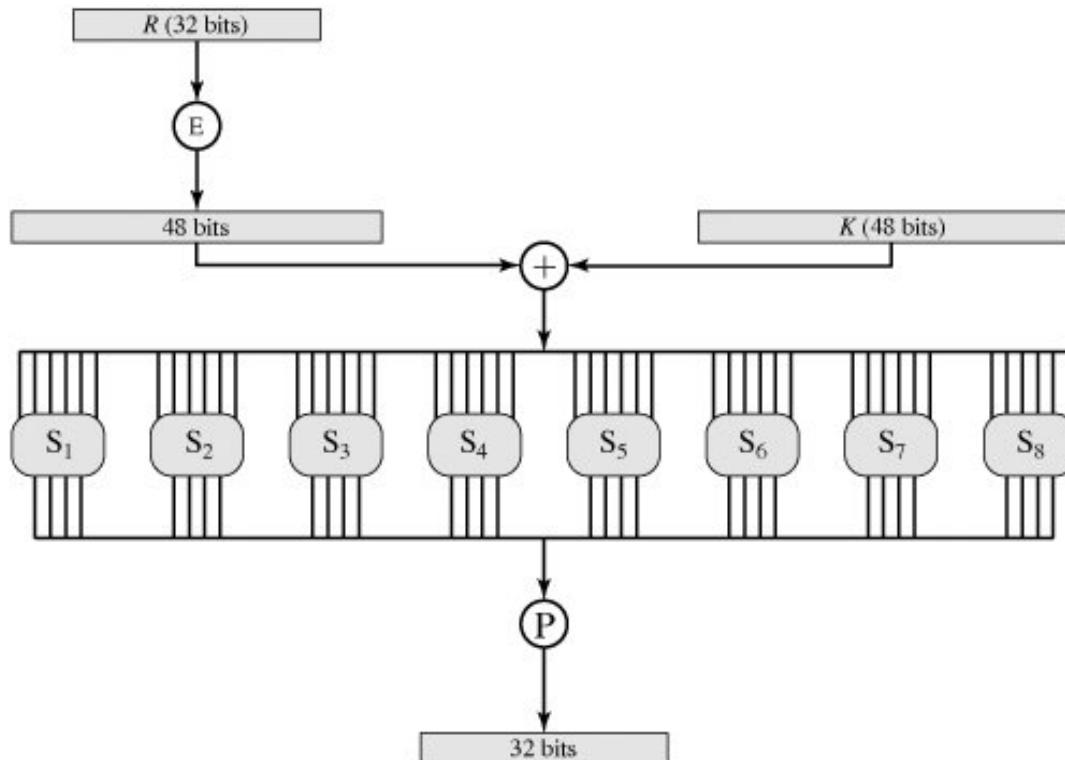
Data Encryption Standard – DES/DEA

- Visão Geral



Data Encryption Standard – DES/DEA

- Função $F(R, SK)$



S_1	<table border="1"> <tbody> <tr><td>14</td><td>4</td><td>13</td><td>1</td><td>2</td><td>15</td><td>11</td><td>8</td><td>3</td><td>10</td><td>6</td><td>12</td><td>5</td><td>9</td><td>0</td><td>7</td></tr> <tr><td>0</td><td>15</td><td>7</td><td>4</td><td>14</td><td>2</td><td>13</td><td>1</td><td>10</td><td>6</td><td>12</td><td>11</td><td>9</td><td>5</td><td>3</td><td>8</td></tr> <tr><td>4</td><td>1</td><td>14</td><td>8</td><td>13</td><td>6</td><td>2</td><td>11</td><td>15</td><td>12</td><td>9</td><td>7</td><td>3</td><td>10</td><td>5</td><td>0</td></tr> <tr><td>15</td><td>12</td><td>8</td><td>2</td><td>4</td><td>9</td><td>1</td><td>7</td><td>5</td><td>11</td><td>3</td><td>14</td><td>10</td><td>0</td><td>6</td><td>13</td></tr> </tbody> </table>	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
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																																																		
S_2	<table border="1"> <tbody> <tr><td>15</td><td>1</td><td>8</td><td>14</td><td>6</td><td>11</td><td>3</td><td>4</td><td>9</td><td>7</td><td>2</td><td>13</td><td>12</td><td>0</td><td>5</td><td>10</td></tr> <tr><td>3</td><td>13</td><td>4</td><td>7</td><td>15</td><td>2</td><td>8</td><td>14</td><td>12</td><td>0</td><td>1</td><td>10</td><td>6</td><td>9</td><td>11</td><td>5</td></tr> <tr><td>0</td><td>14</td><td>7</td><td>11</td><td>10</td><td>4</td><td>13</td><td>1</td><td>5</td><td>8</td><td>12</td><td>6</td><td>9</td><td>3</td><td>2</td><td>15</td></tr> <tr><td>13</td><td>8</td><td>10</td><td>1</td><td>3</td><td>15</td><td>4</td><td>2</td><td>11</td><td>6</td><td>7</td><td>12</td><td>0</td><td>5</td><td>14</td><td>9</td></tr> </tbody> </table>	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
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																																																		
S_3	<table border="1"> <tbody> <tr><td>10</td><td>0</td><td>9</td><td>14</td><td>6</td><td>3</td><td>15</td><td>5</td><td>1</td><td>13</td><td>12</td><td>7</td><td>11</td><td>4</td><td>2</td><td>8</td></tr> <tr><td>13</td><td>7</td><td>0</td><td>9</td><td>3</td><td>4</td><td>6</td><td>10</td><td>2</td><td>8</td><td>5</td><td>14</td><td>12</td><td>11</td><td>15</td><td>1</td></tr> <tr><td>13</td><td>6</td><td>4</td><td>9</td><td>8</td><td>15</td><td>3</td><td>0</td><td>11</td><td>1</td><td>2</td><td>12</td><td>5</td><td>10</td><td>14</td><td>7</td></tr> <tr><td>1</td><td>10</td><td>13</td><td>0</td><td>6</td><td>9</td><td>8</td><td>7</td><td>4</td><td>15</td><td>14</td><td>3</td><td>11</td><td>5</td><td>2</td><td>12</td></tr> </tbody> </table>	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
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																																																		
S_4	<table border="1"> <tbody> <tr><td>7</td><td>13</td><td>14</td><td>3</td><td>0</td><td>6</td><td>9</td><td>10</td><td>1</td><td>2</td><td>8</td><td>5</td><td>11</td><td>12</td><td>4</td><td>15</td></tr> <tr><td>13</td><td>8</td><td>11</td><td>5</td><td>6</td><td>15</td><td>0</td><td>3</td><td>4</td><td>7</td><td>2</td><td>12</td><td>1</td><td>10</td><td>14</td><td>9</td></tr> <tr><td>10</td><td>6</td><td>9</td><td>0</td><td>12</td><td>11</td><td>7</td><td>13</td><td>15</td><td>1</td><td>3</td><td>14</td><td>5</td><td>2</td><td>8</td><td>4</td></tr> <tr><td>3</td><td>15</td><td>0</td><td>6</td><td>10</td><td>1</td><td>13</td><td>8</td><td>9</td><td>4</td><td>5</td><td>11</td><td>12</td><td>7</td><td>2</td><td>14</td></tr> </tbody> </table>	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
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																																																		
S_5	<table border="1"> <tbody> <tr><td>2</td><td>12</td><td>4</td><td>1</td><td>7</td><td>10</td><td>11</td><td>6</td><td>8</td><td>5</td><td>3</td><td>15</td><td>13</td><td>0</td><td>14</td><td>9</td></tr> <tr><td>14</td><td>11</td><td>2</td><td>12</td><td>4</td><td>7</td><td>13</td><td>1</td><td>5</td><td>0</td><td>15</td><td>10</td><td>3</td><td>9</td><td>8</td><td>6</td></tr> <tr><td>4</td><td>2</td><td>1</td><td>11</td><td>10</td><td>13</td><td>7</td><td>8</td><td>15</td><td>9</td><td>12</td><td>5</td><td>6</td><td>3</td><td>0</td><td>14</td></tr> <tr><td>11</td><td>8</td><td>12</td><td>7</td><td>1</td><td>14</td><td>2</td><td>13</td><td>6</td><td>15</td><td>0</td><td>9</td><td>10</td><td>4</td><td>5</td><td>3</td></tr> </tbody> </table>	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
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_6	<table border="1"> <tbody> <tr><td>12</td><td>1</td><td>10</td><td>15</td><td>9</td><td>2</td><td>6</td><td>8</td><td>0</td><td>13</td><td>3</td><td>4</td><td>14</td><td>7</td><td>5</td><td>11</td></tr> <tr><td>10</td><td>15</td><td>4</td><td>2</td><td>7</td><td>12</td><td>9</td><td>5</td><td>6</td><td>1</td><td>13</td><td>14</td><td>0</td><td>11</td><td>3</td><td>8</td></tr> <tr><td>9</td><td>14</td><td>15</td><td>5</td><td>2</td><td>8</td><td>12</td><td>3</td><td>7</td><td>0</td><td>4</td><td>10</td><td>1</td><td>13</td><td>11</td><td>6</td></tr> <tr><td>4</td><td>3</td><td>2</td><td>12</td><td>9</td><td>5</td><td>15</td><td>10</td><td>11</td><td>14</td><td>1</td><td>7</td><td>6</td><td>0</td><td>8</td><td>13</td></tr> </tbody> </table>	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
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_7	<table border="1"> <tbody> <tr><td>4</td><td>11</td><td>2</td><td>14</td><td>15</td><td>0</td><td>8</td><td>13</td><td>3</td><td>12</td><td>9</td><td>7</td><td>5</td><td>10</td><td>6</td><td>1</td></tr> <tr><td>13</td><td>0</td><td>11</td><td>7</td><td>4</td><td>9</td><td>1</td><td>10</td><td>14</td><td>3</td><td>5</td><td>12</td><td>2</td><td>15</td><td>8</td><td>6</td></tr> <tr><td>1</td><td>4</td><td>11</td><td>13</td><td>12</td><td>3</td><td>7</td><td>14</td><td>10</td><td>15</td><td>6</td><td>8</td><td>0</td><td>5</td><td>9</td><td>2</td></tr> <tr><td>6</td><td>11</td><td>13</td><td>8</td><td>1</td><td>4</td><td>10</td><td>7</td><td>9</td><td>5</td><td>0</td><td>15</td><td>14</td><td>2</td><td>3</td><td>12</td></tr> </tbody> </table>	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
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_8	<table border="1"> <tbody> <tr><td>13</td><td>2</td><td>8</td><td>4</td><td>6</td><td>15</td><td>11</td><td>1</td><td>10</td><td>9</td><td>3</td><td>14</td><td>5</td><td>0</td><td>12</td><td>7</td></tr> <tr><td>1</td><td>15</td><td>13</td><td>8</td><td>10</td><td>3</td><td>7</td><td>4</td><td>12</td><td>5</td><td>6</td><td>11</td><td>0</td><td>14</td><td>9</td><td>2</td></tr> <tr><td>7</td><td>11</td><td>4</td><td>1</td><td>9</td><td>12</td><td>14</td><td>2</td><td>0</td><td>6</td><td>10</td><td>13</td><td>15</td><td>3</td><td>5</td><td>8</td></tr> <tr><td>2</td><td>1</td><td>14</td><td>7</td><td>4</td><td>10</td><td>8</td><td>13</td><td>15</td><td>12</td><td>9</td><td>0</td><td>3</td><td>5</td><td>6</td><td>11</td></tr> </tbody> </table>	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
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/DEA

- O efeito avalanche do DES
 - Uma pequena modificação no texto claro ou na chave produz significativa alteração no texto cifrado
 - Ex: Modificação de um bit da chave

Texto Claro:

01101000 10000101 00101111 01111010 00010011 01110110 11101011 10100100

Chave

1110010 1111011 1101111 0011000 0011101 0000100 0110001 11011100

0110010 1111011 1101111 0011000 0011101 0000100 0110001 11011100

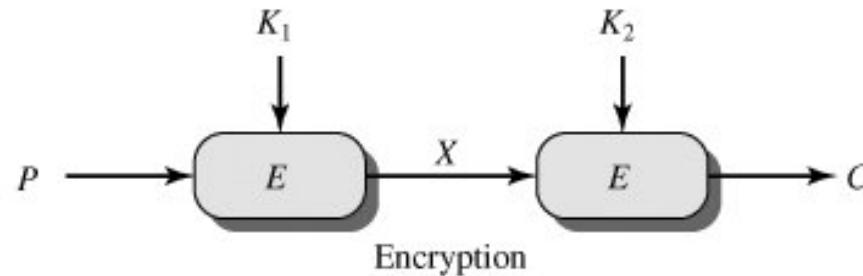
(a) Change in Plaintext		(b) Change in Key	
Round	Number of bits that differ	Round	Number of bits that differ
0	1	0	0
1	6	1	2
2	21	2	14
3	35	3	28
4	39	4	32
5	34	5	30
6	32	6	32
7	31	7	35
8	29	8	34
9	42	9	40
10	44	10	38
11	32	11	31
12	30	12	33
13	30	13	28
14	26	14	26
15	29	15	34
16	34	16	35

DES Triplo (3DES)

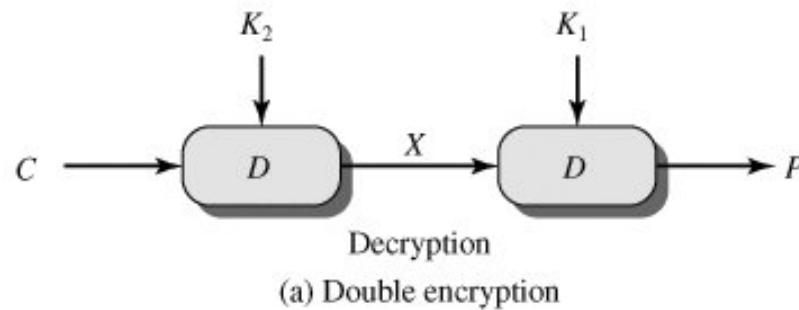
- DES, apesar de ser resistente a criptoanálise linear diferencial ainda é relativamente fraco contra ataques de força bruta
- Apesar de um domínio de chaves de 2^{256} , o DES foi provado ser inseguro em 1998
 - Ataque com menos de 3 dias para decifrar uma chave através de uma máquina decifradora de DES com custo de 250 mil dólares
- Em alternativa ao DES, foi proposto o **DES duplo**, que consiste em dois estágios do DES com duas chaves K1 e K2 aumentando o tamanho de **chave** para **112 bits**

DES Duplo

- Cifragem **DES duplo**: $C = E(K_2, E(K_1, P))$



- Decifragem **DES duplo**: $P = D(K_1, D(K_2, C))$

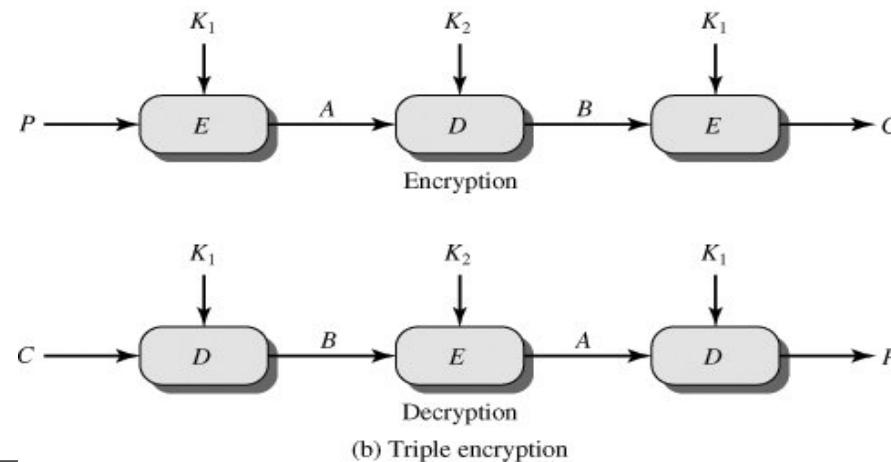


DES Triplo (3DES)

- Apesar de ser melhor que o DES, o **DES duplo** pode ser atacado por um modelo que reduz o problema da descoberta da chave com esforço muito próximo ao DES normal
- Ataque ***meet-in-the-middle*** (encontro no meio) baseia-se nas seguintes observações:
 - Sendo: $C = E(K_2, E(K_1, P)) \text{ // } P \text{ e } C \text{ conhecidos!!}$
 - então $X = E(K_1, P) = D(K_2, C)$
- Dado um par conhecido **(P,C)** podemos proceder com o ataque de texto conhecido gerando todos os valores possíveis de $E(P, K_1)$ e todos os valores possíveis de $D(C, K_2)$
 - Agora basta comparamos os resultados com um novo par **(P,C)**.

DES Triplo (3DES)

- Uma contramedida óbvia para o ataque **meet-in-the-middle** é usar um estágio adicional com três chaves diferentes
 - Aumenta o custo do ataque para 2^{112} (aprox 5×10^{33})
 - Exige uma chave de $56 \times 3 = 168$ bits, que pode ser muito pesado
- Alternativa: DES Triplo com duas chaves
 - Chave de 112 bits, melhor desempenho, boa segurança apesar dos trabalhos demonstrarem vulnerabilidades
 - $C = E(K_1, D(K_2, E(K_1, P)))$
 - $P = D(K_1, E(K_2, D(K_1, C)))$



DES Triplo (3DES)

- Finalmente o **DES triplo com 3 Chaves**:
 - $C = E(K_3, D(K_2, E(K_1, P)))$
 - $P = D(K_3, E(K_2, D(K_1, C)))$
- Motivação: desconfiança na segurança do DES triplo usando 2 chaves apenas
- Torna-se compatível com DES quando utilizamos $k_3=k_2$ ou $k_2=k_1$

AES – Advanced Encryption Standard

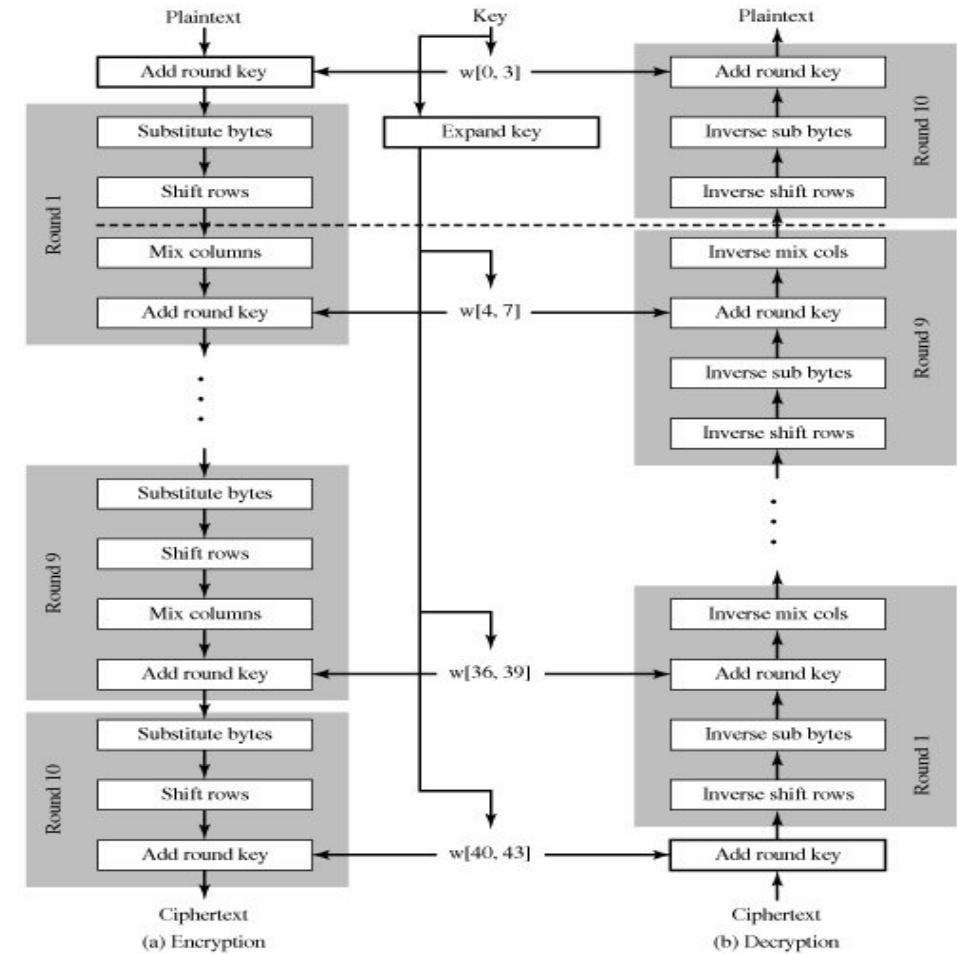
- 3DES possui grande resistência a diversos ataques sendo inclusive aos ataques de força bruta, o grande problema do seu sucessor DES
- A grande desvantagem do 3DES é o tamanho de chave de 168 bits que em conjunto com o modelo de cifra de Feistel o torna muito lento em software
- Por este motivo, em 1997 o NIST abriu um concurso para propostas de um novo padrão de segurança o **Advanced Encryption Standard** ou **AES**
 - Esse novo algoritmo deveria ser tão forte quanto o 3DES porém com eficiência e chaves maiores

AES – Advanced Encryption Standard

- Ao final do concurso (2001) o algoritmo escolhido foi o Rijndael desenvolvido pelos belgas Dr. Joan Daemen e Dr. Vicent Rijmen
- Em resumo, o novo **AES** usa um tamanho de **blocos de 128 bits** e **chaves** de tamanho **128,192 ou 256 bits**
- **Não usa a estrutura de Feistel** e a cada rodada inclui 4 funções:
 - Substituição de bytes
 - Permutação
 - Operações sobre um corpo finito
 - E operação XOR

AES – Advanced Encryption Standard

- Estrutura Detalhada
 - Chave de Entrada é expandida em um vetor de 44 words
 - Sub Bytes
 - ShiftRows
 - MixColumns
 - AddRoundKey
 - Modelo reversível porém com estruturas diferentes para cada modo E e D



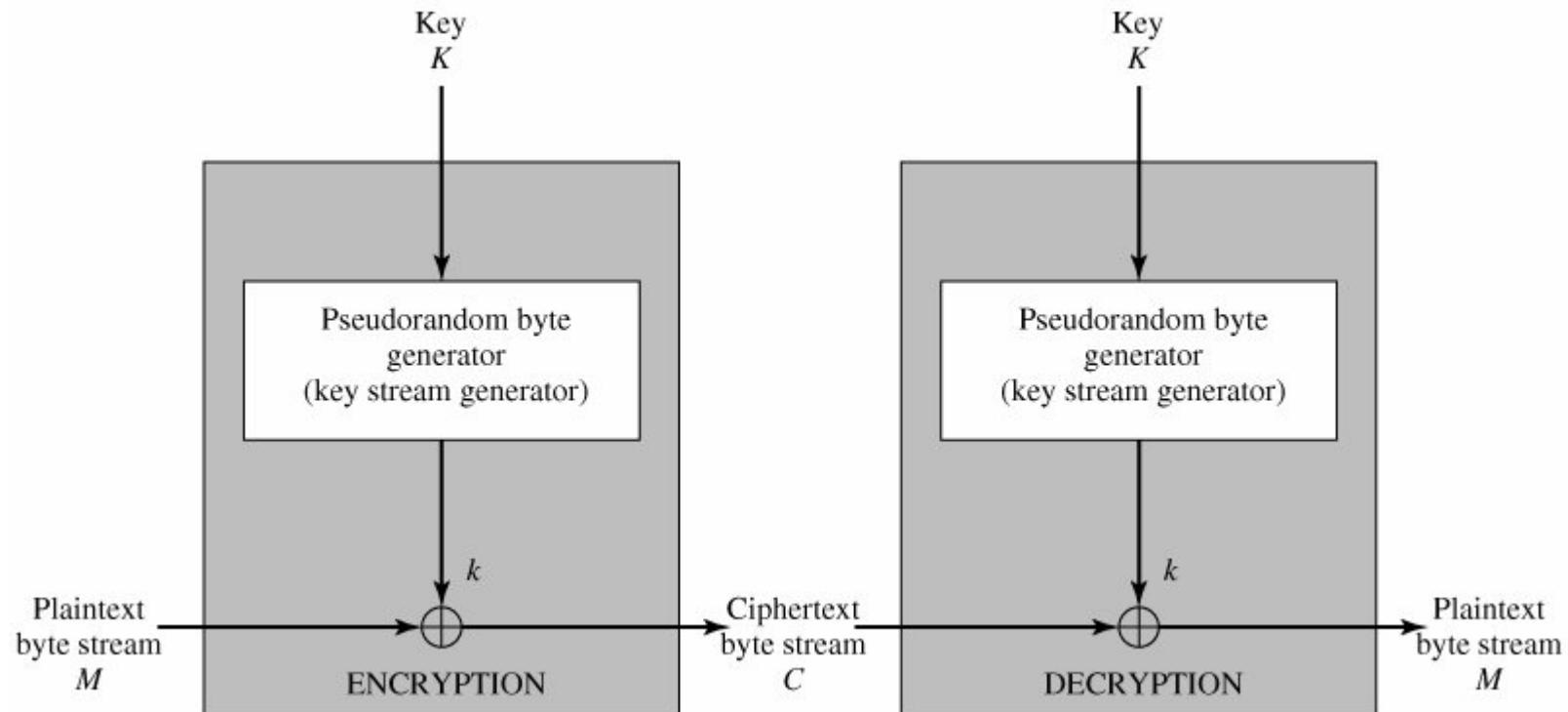
Cifras de Fluxo

- Cifras de fluxo codificam um fluxo de dados por bit ou bytes de cada vez
 - Sua qualidade está ligada ao tamanho da chave que deve, quanto maior a chave maior a dificuldade de revelar a mensagem
- Características
 - Muito parecida com one-time-pad (números aleatórios genuínos)
 - Cifras de fluxo usam números pseudo-aleatórios
 - Sequência de criptografia deverá ter um período muito grande
 - Fluxo determinístico de bits com repetição
 - Quanto maior o período melhor
 - Fluxo de chave deverá aproximar ao máximo do fluxo de números aleatórios

11001100	plaintext
\oplus	key stream
<u>01101100</u>	
10100000	ciphertext
10100000	ciphertext
\oplus	key stream
<u>01101100</u>	
11001100	plaintext

Cifras de Fluxo

- Chave precisa ser suficientemente longa para evitar ataques de força bruta (Ex: 128 bits)



Cifras de Fluxo

- Vantagem:
 - Quase sempre mais rápida em software do que a cifra de bloco
 - Poucas linhas de código para implementar
- Pode ser usado:
 - Aplicações de criptografia sobre canais de comunicação
 - Necessário confrontar, desempenho e nível de confidencialidade desejado (valor da informação)

Algoritmo RC4

- “**Rivest Cipher 4**” ou **RC4** é uma cifra de fluxo projetada em 1987 por Rivest para a *RSA Security*
 - Possui tamanho de chave variável
 - Baseado em Permutação P-Aleatória
 - Período de repetição muito grande na ordem de 10^{100}
- Era segredo até 1994

Algoritmo RC4

- Onde é utilizado:
 - Padrões SSL/TLS (Secure Sockets Layer/Transport Layer Security) → HTTPS
 - WEP (Wired Equivalence Privacy)
 - WPA (Wifi Protected Access) -> WPA2 usa AES.

Cipher	Key Length	Speed (Mbps)
DES	56	9
3DES	168	3
RC2*	variable	0.9
RC4	variable	45

* RC2: cifrador de bloco criado por Ron Rivest em 1987.

Algoritmo RC4

- Funcionamento Geral:
 - **Chave K** variável de 1 a 256 bytes (8 a 2048 bits)
 - **S = vetor de estados** de 256 posições (0 a 255) inicializado por permutação de posições definidas pela chave **K**
 - Para cifrar usamos um **k** gerado a partir de **S**, byte a byte
 - Para decifrar realizamos o mesmo processo

Algoritmo RC4

- Inicialização de **S**:
 - **S** é inicializado com os valores crescentes de sua posição 0 a 255 ($S[0]= 0 \dots S(2)=2 \dots$)
 - Um vetor **T** auxiliar armazena os bytes da **chave K**, com repetição ou não, sendo **K** com tamanho 256 então **T=K** senão repete-se **K** para preencher cada posição de **T**

```
/* Inicialização de S */  
for i = 0 to 255 do  
    S[i] = i;  
    T[i] = K[i mod keylen];
```

Algoritmo RC4

- Permutação Inicial de **S**
 - T é usado para permutar S de 0 a 255 para cada S_i trocar S_i por S_j onde j é ditado por T

```
/* Permutação Inicial de S */  
j = 0;  
for i = 0 to 255 do  
    j = (j + S[i] + T[i]) mod 256;  
    Swap (S[i], S[j]);
```

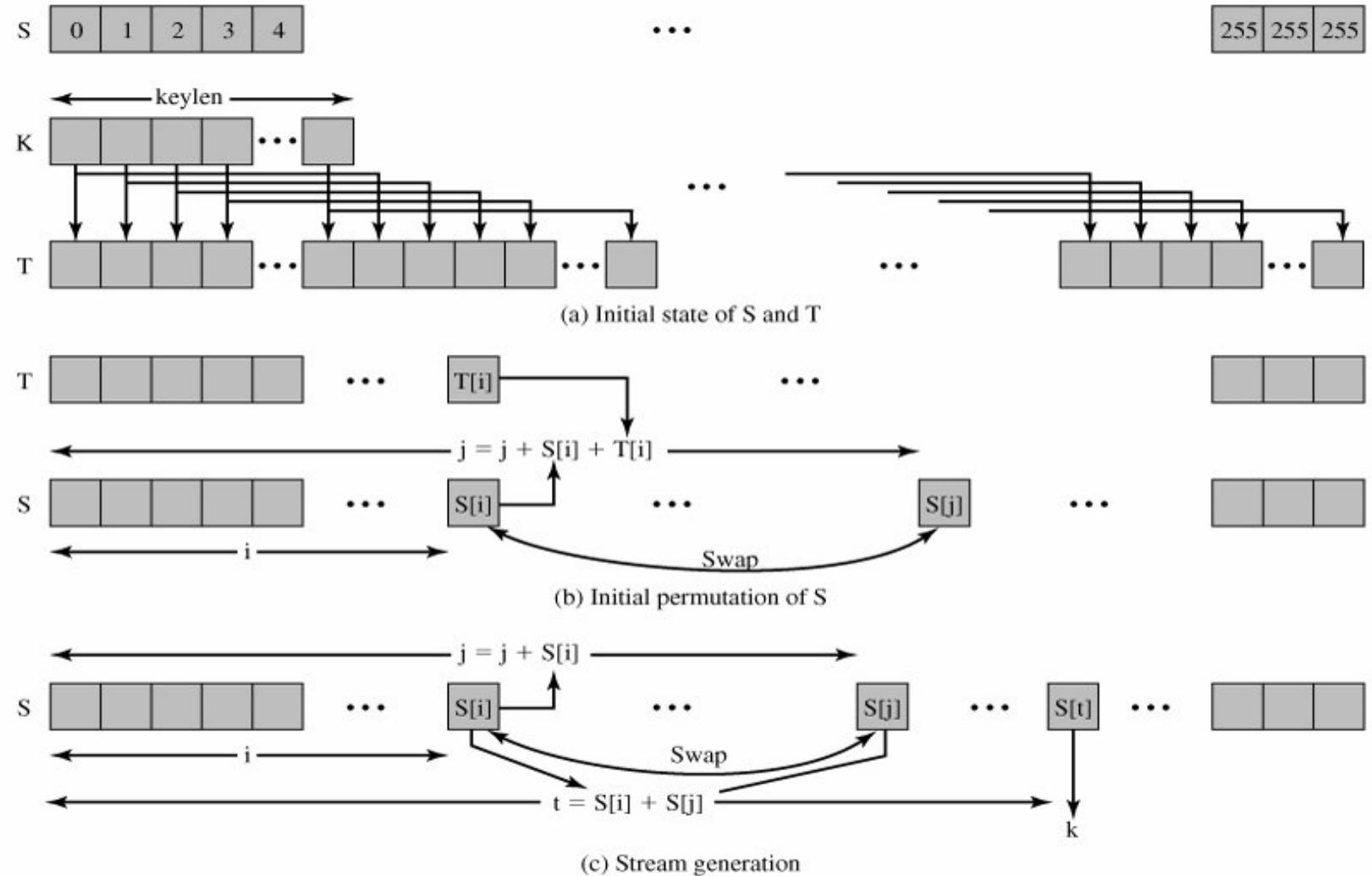
Algoritmo RC4

- Geração de Fluxo
 - Percorrer todos os elementos de S para cada S_i trocar por outro elemento ditado por S_i
 - Ao fim fazemos o **XOR** do valor k com o próximo byte de texto plano

```
/* Geração de Fluxo */  
i, j = 0;  
while (true)  
    i = (i + 1) mod 256;  
    j = (j + S[i]) mod 256;  
    Swap (S[i], S[j]);  
    t = (S[i] + S[j]) mod 256;  
    k = S[t];
```

Algoritmo RC4

- Visão Geral



Atividade Teórica

- 1. Aprofunde os seus estudo sobre o algoritmo de criptografia **DES** apresentado nesta aula e explique os diferentes modos de operação de cifras de bloco, deixando claro as vantagens de cada um:
 - Eletronic CodeBook (ECB)
 - Chipher Block Chaining (CBC)
 - Chipher FeedBack (CFB)
 - Output FeedBack (OFB)
 - CounTeR (CTR)

Atividade Prática

- 1. Implemente o algoritmo de criptografia **S-DES** apresentado em aula para cifrar/decifrar um arquivo texto passado por parâmetro em linha de comando
 - Tarefa em dupla
 - Em seus testes, vocês deverão definir a chaves
 - Pode ser implementado na linguagem de seu domínio
 - Deverá rodar no laboratório (usar máquina virtual no caso de precisar de pacotes específicos) e ser apresentado ao professor
- 2. Implementar o protótipo do algoritmo **RC4** para cifra qualquer texto usando uma chave de tamanho variável entre 1 a 256 bytes

Apresentar um relatório simples, sobre as implementações (como executar, as dificuldades no desenvolvimento, etc) e os testes realizados. Incluir a Atividade Teórica no relatório!!!

Atividade Prática

- 3. Desenvolva uma aplicação para troca de mensagens de texto (estilo chat) entre você e seus colegas de maneira que seja possível trocar mensagens de texto entre pares utilizando criptografia com o **S-DES** e o **RC4**, desenvolvidos por você
 - Assuma que a comunicação será feita pela **porta 5354**, com **socket TCP**
 - Assuma também que ambos os pares já conhecem **a chave de segurança** e essa **poderá ser modificada** pelo usuário **em tempo de execução** da aplicação, **bem como o algoritmo de criptografia utilizado (S-DES ou RC4)**

Atividade Prática

- Observações importantes
 - Testem as suas implementações com outras duplas para verificar se a sua codificação está correta ou “só funciona entre o seu cifrador/decifrador” (neste caso, você implementou um algoritmo próprio e não o pedido!)
 - O professor poderá requisitar que a dupla apresente pessoalmente o seu trabalho, para fins de esclarecer quaisquer dúvidas que possam surgir
 - Não é permitido o uso de bibliotecas para a realização de todo ou parte dos algoritmos criptográficos

Atividade Prática

- Referências
 - S-DES
 - <http://mercury.webster.edu/aleshunas/COSC%205130/G-SDES.pdf>
 - <ftp://180.211.120.110/04%20IT%20Department/KUK/Simplified%20DES.pdf>
 - RC4
 - <http://www.vanilla47.com/PDFs/Cryptography/RC4%20Stream%20Cipher/Tutorials/THE%20RC4%20STREAM%20ENCRYPTION%20ALGORITHM.pdf>
 - Sockets em C++
 - Slides em anexo
 - https://www.freebsd.org/doc/en_US.ISO8859-1/books/developers-handbook/sockets-essential-functions.html
 - Multithreading em C++
 - <https://www.tutorialcup.com/cplusplus/multithreading.htm>

Alguma Questão?

