

Criptografia de chave simétrica



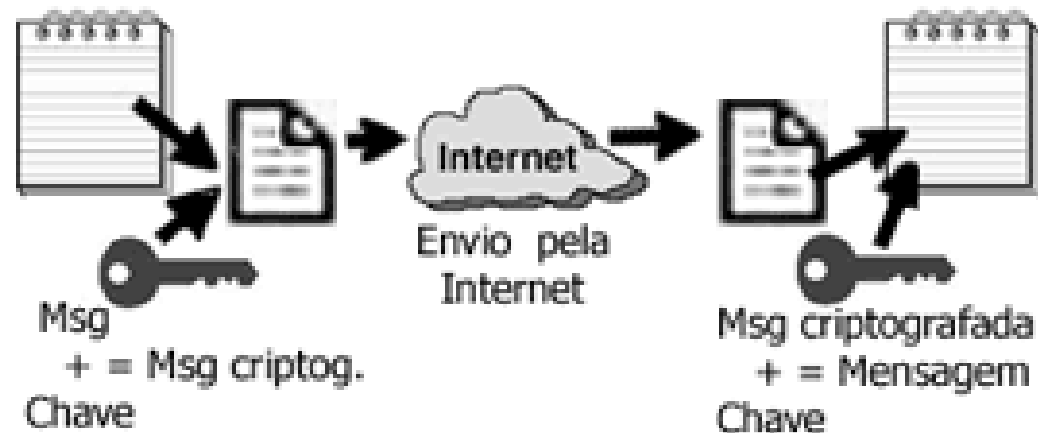
Murillo Pontual
(mbcrap@cin.ufpe.br)

Igor Vanderlei
(imv@cin.ufpe.br)

Modelos de Criptografia:

•Chaves Simétricas (Informal):

É utilizada uma única chave para cifrar e decifrar a mensagem.



- **Modelos de Chave Simétrica – Subdivisão**

- **Cifras de fluxo**

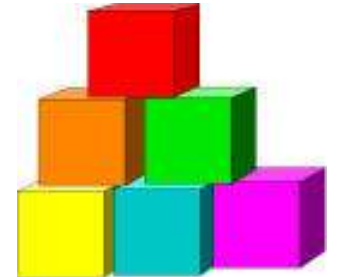
As cifras de fluxo encriptam um texto pleno bit a bit.

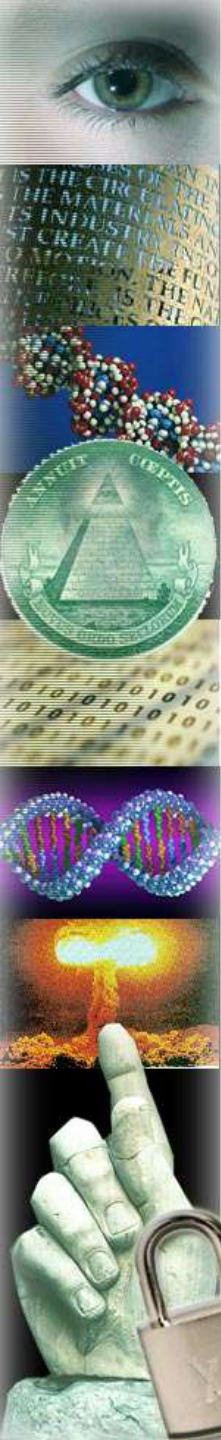


1 0 1 0 0 0 1 1

- **Cifras de bloco**

As cifras de bloco usam conjuntos com um número fixo de bits (geralmente 64 bits nas cifras modernas) como unidades de cifragem.

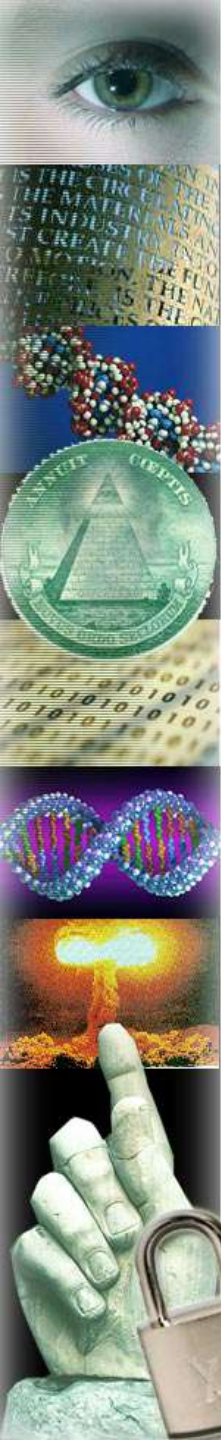




Cifra de Bloco:

Definição: É uma função matemática $E: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^l$ que recebe 2 entradas, uma chave **K** de comprimento k , e um texto pleno **M** com comprimento l , e retorna um texto cifrado **C** com comprimento l , i.e. $C=E(M,K)$.

Para cada chave $\mathbf{K} \in \{0, 1\}^k$ tem-se $E_K: \{0, 1\}^l \rightarrow \{0, 1\}^l$ como sendo a função $E_K(M) = E(K, M)$. Para qualquer cifra de bloco, e qualquer chave **K**, a função E_k é uma permutação sobre $\{0, 1\}^n$, i.e. há uma bijeção que mapeia $\{0, 1\}^l$ to $\{0, 1\}^l$. A função E_k possui uma inversa, denotada por E_k^{-1} , esta função inversa também mapeia $\{0, 1\}^l$ to $\{0, 1\}^l$.



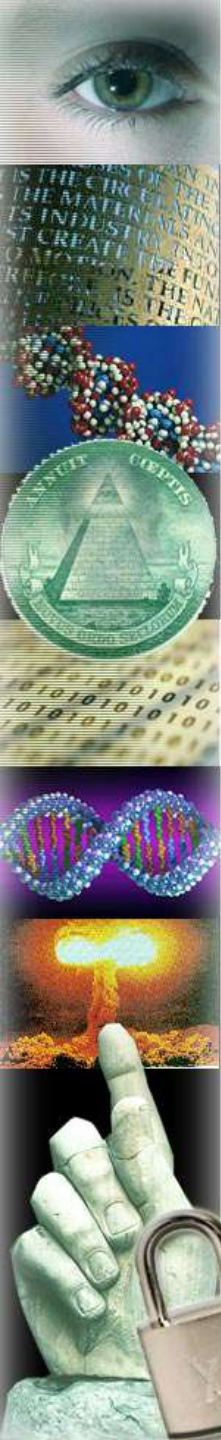
Cifra de Bloco(continuação):

Como era de se esperar, as seguintes condições são satisfeitas:

$$E_K^{-1}(E_K(M)) = M \quad E_K(E_K^{-1}(C)) = C$$

Defini-se $E^{-1}: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^l$

como sendo $E^{-1}(K, C) = E_K^{-1}(C)$, isto é, o inverso da cifra E .

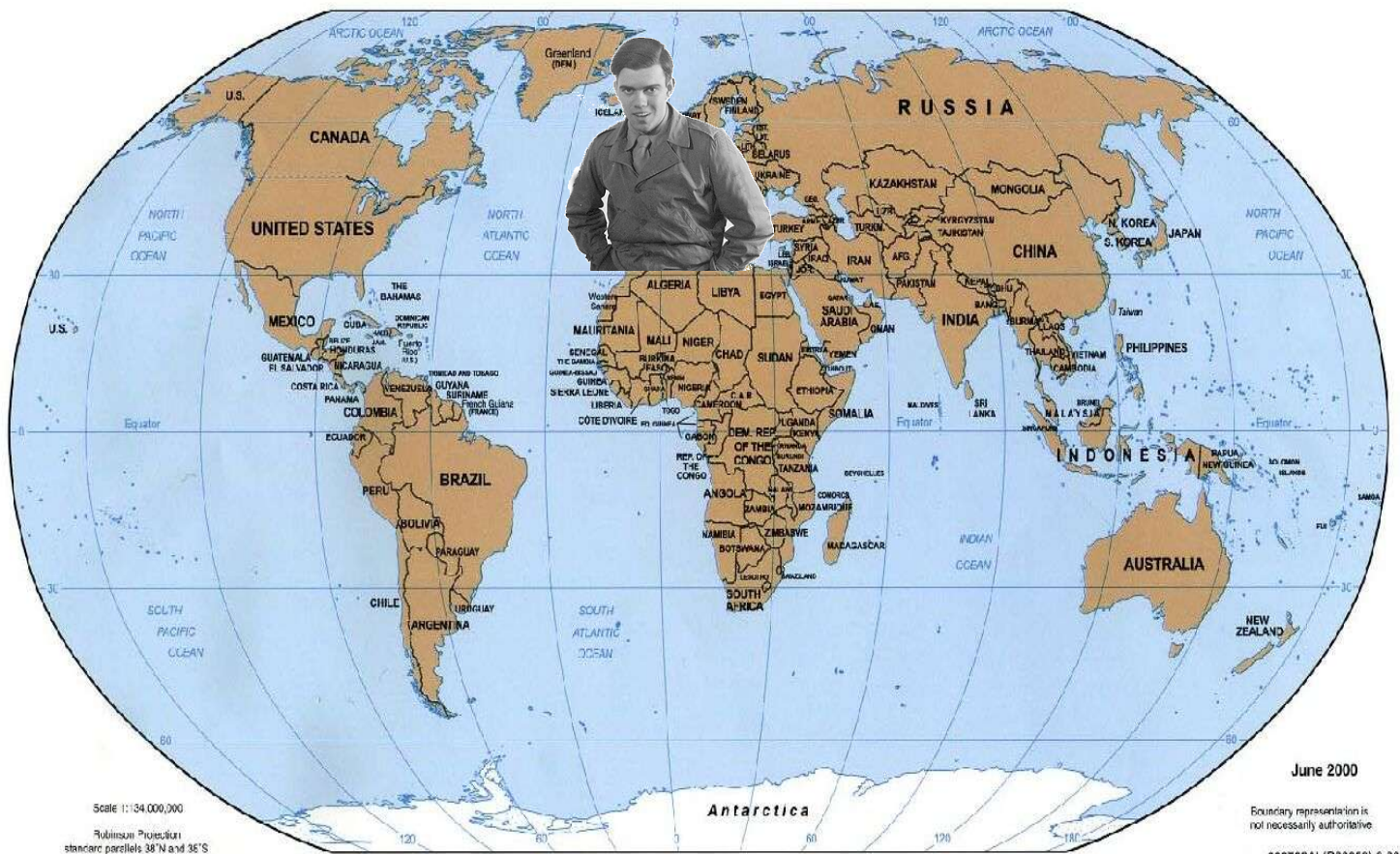


Cifra de Bloco - Algumas Considerações:

- Uma cifra de bloco deve ser um algoritmo público e bem especificado;
- Tanto a cifra E , como sua inversa E^{-1} devem ser fáceis de serem computadas, isto é, dados K, M , pode-se computar $E(K, M)$, e dados K, C , pode-se computar $E^{-1}(K, C)$.
- Na prática uma chave K é escolhida e mantida secreta por um par de usuários. Em seguida a função E_k é utilizada para processar de alguma forma as informações.
- A segurança se baseia em manter a chave secreta.

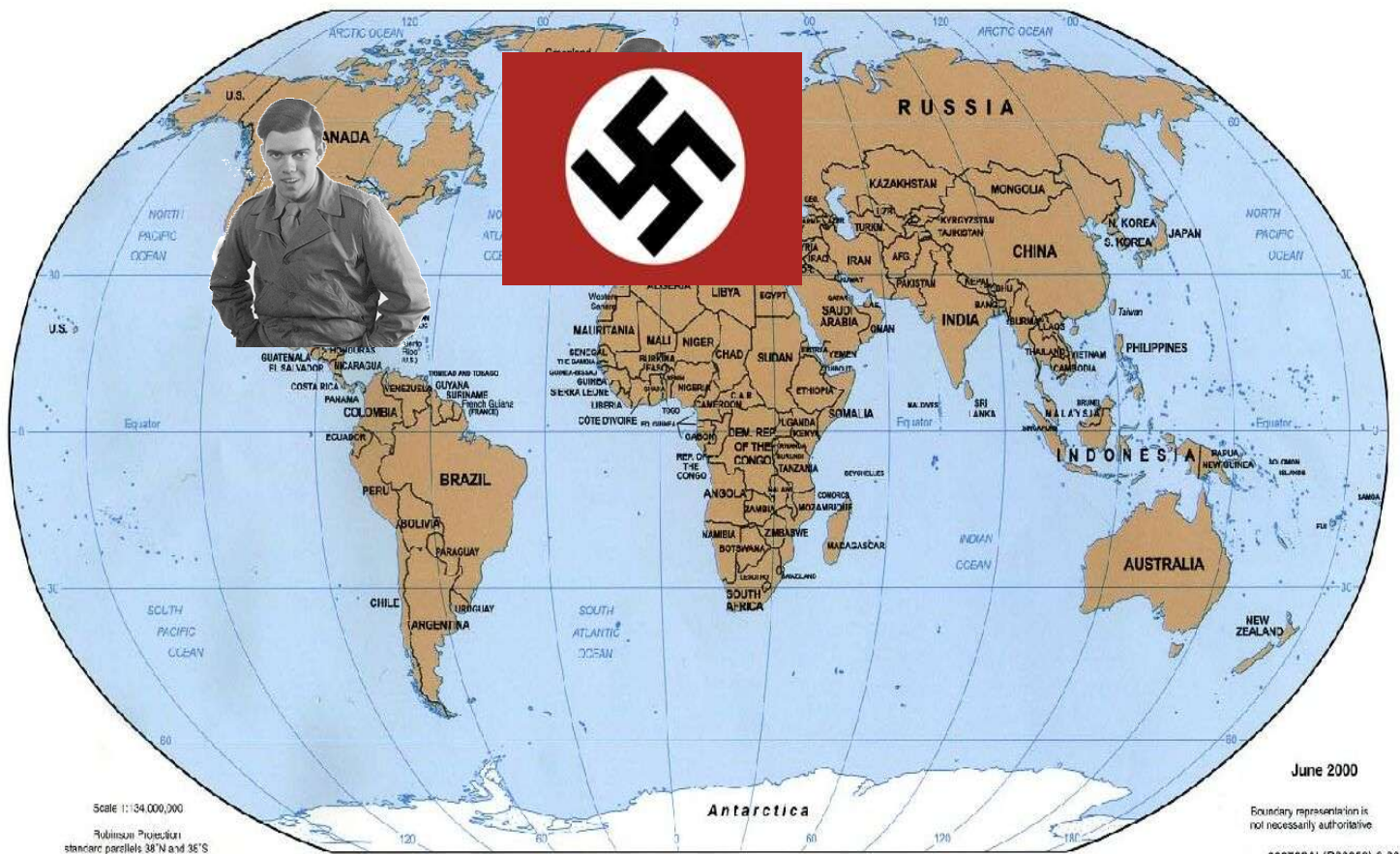
Um Pouco de História:

Década de 30



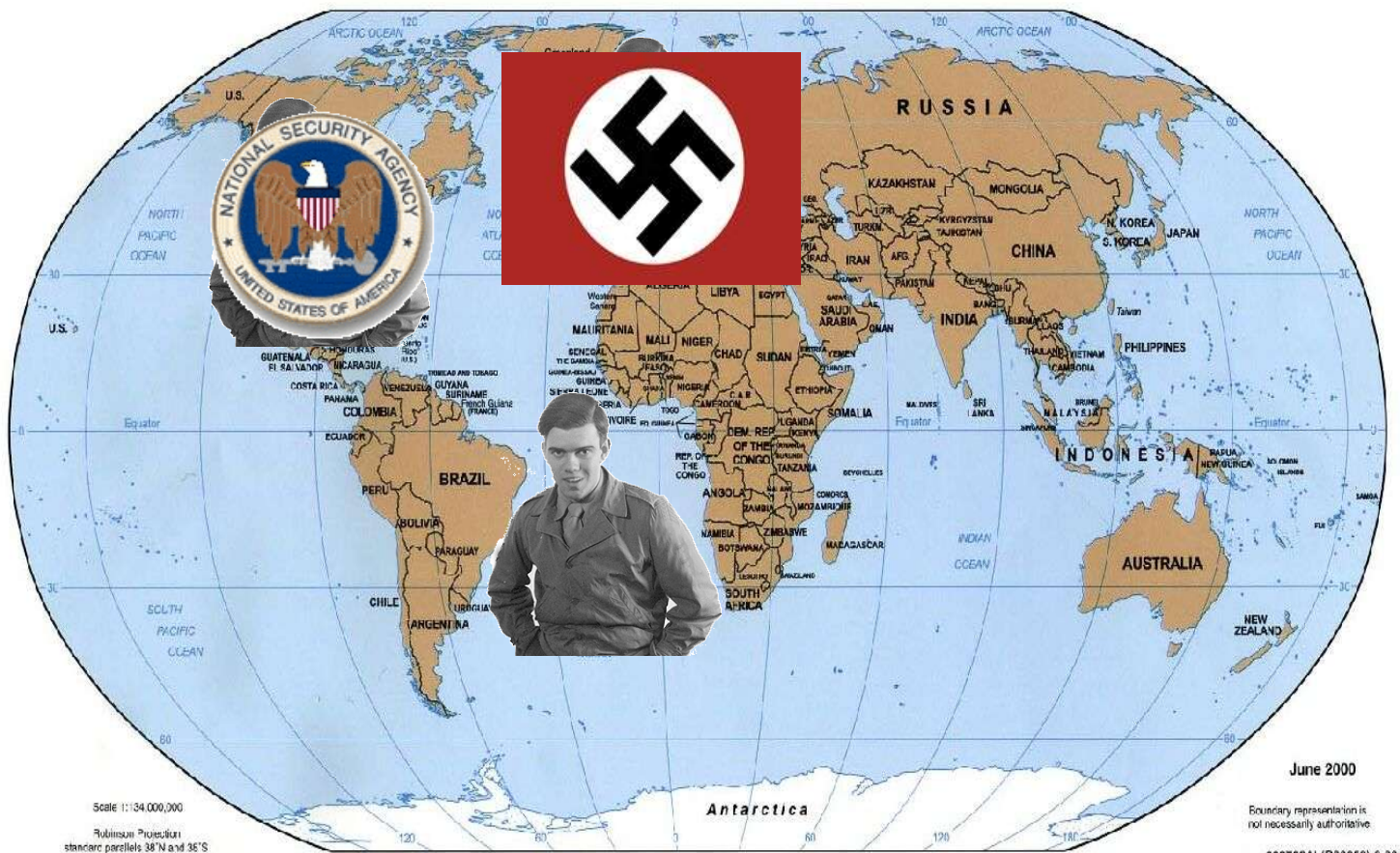
Um Pouco de História:

Década de 30



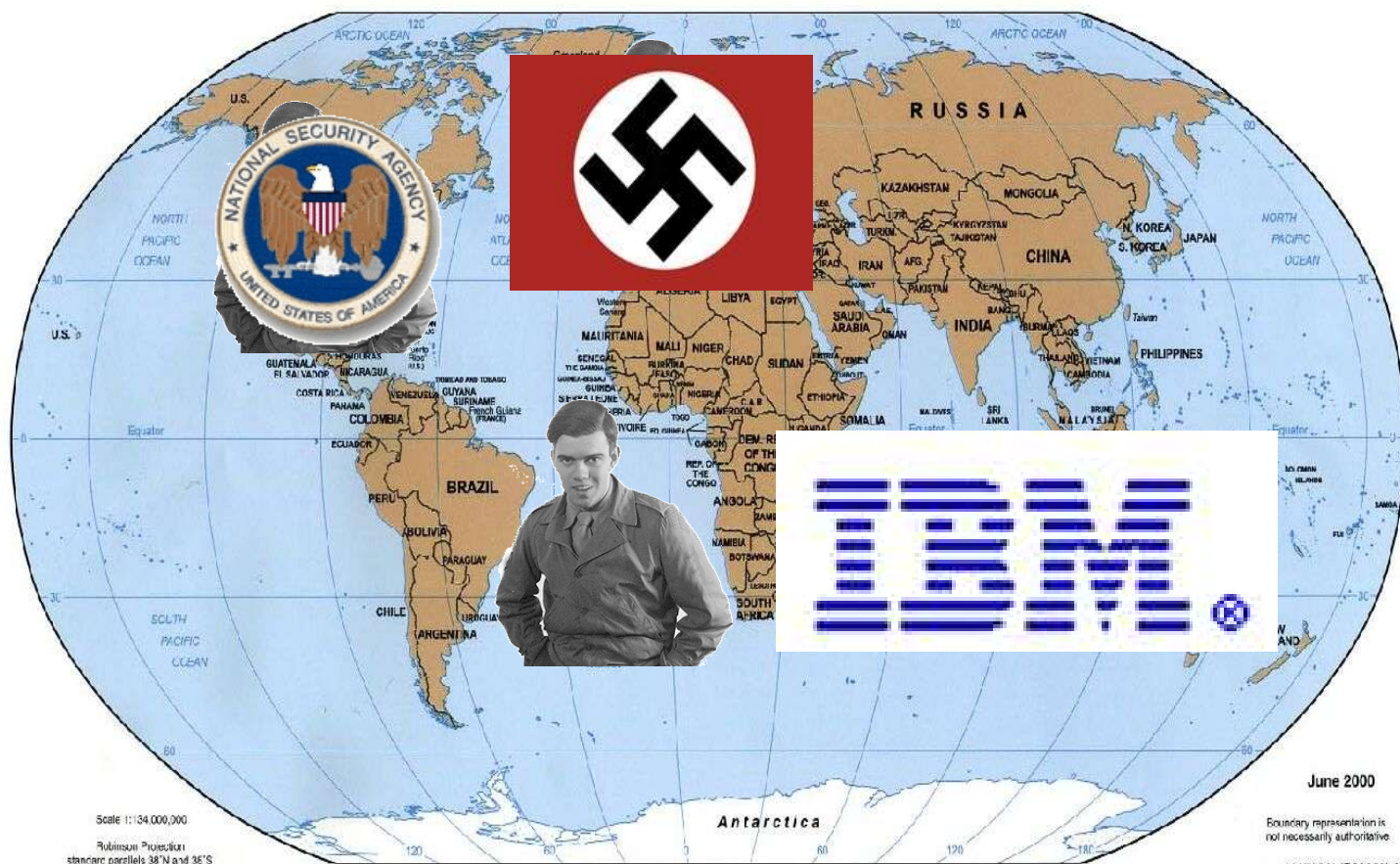
Um Pouco de História:

Década de 30



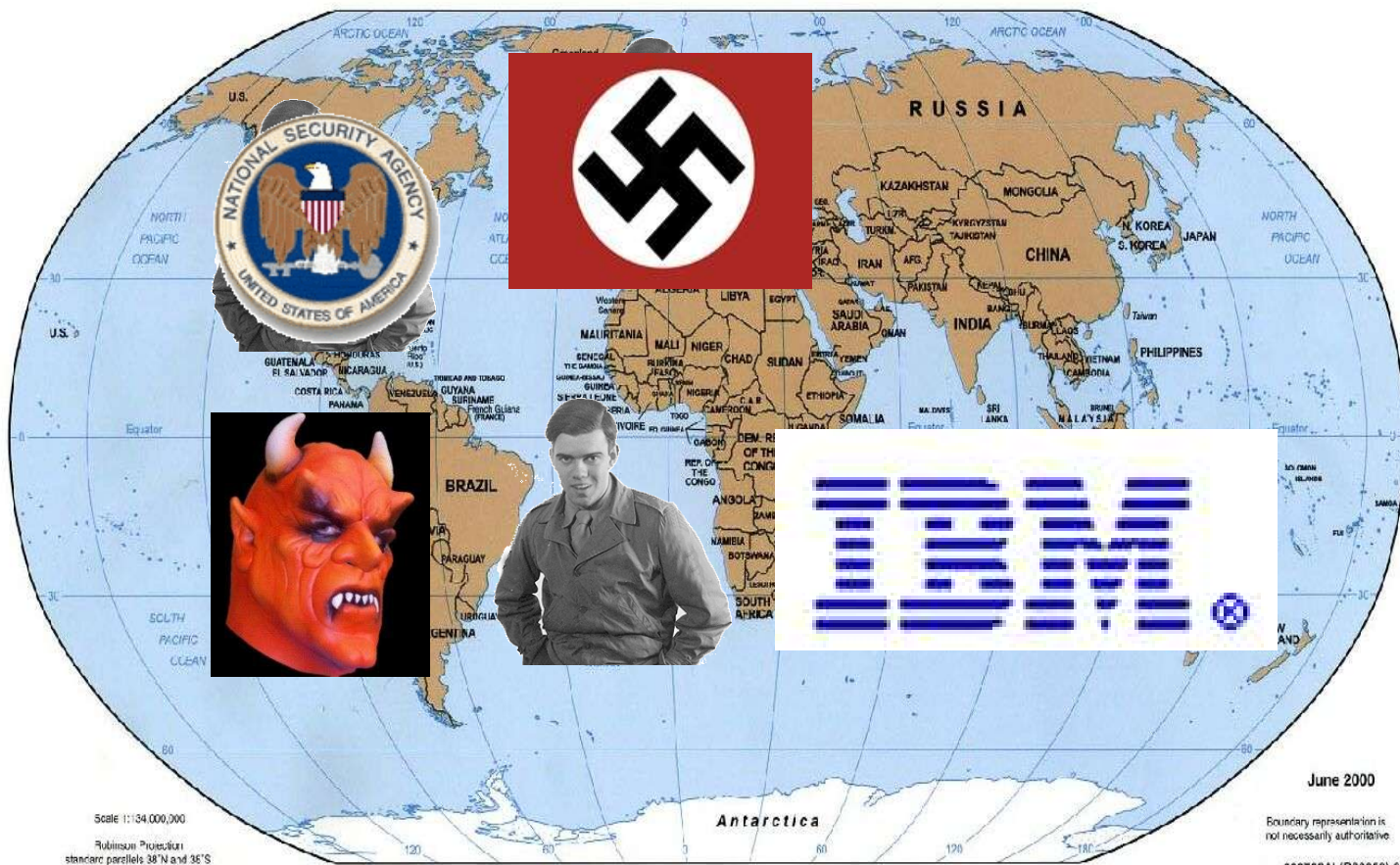
Um Pouco de História:

Década de 70



Um Pouco de História:

Década de 70



June 2000

Boundary representation is
not necessarily authoritative.

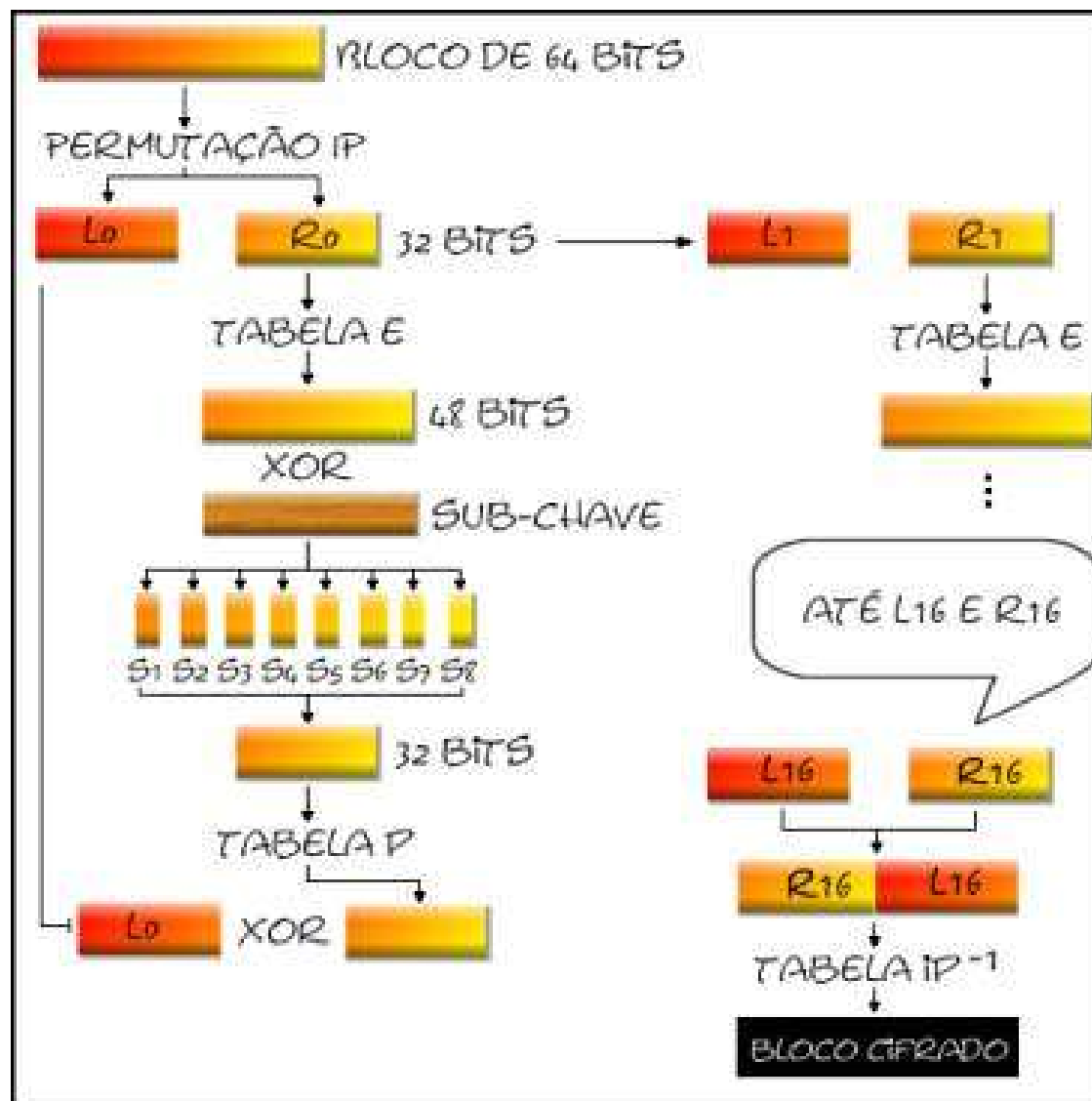
8027C3AI (R00352) 6-00

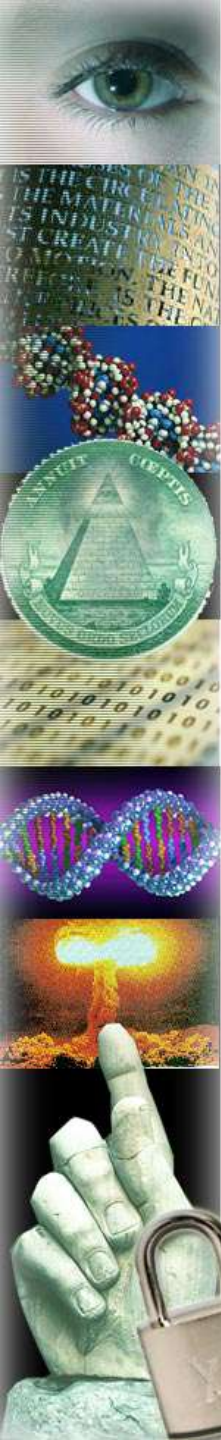
Um Pouco de História:

Década de 70



O Funcionamento do DES:





Passo a passo – DES:

Entrada 1: Blocos de Texto pleno em Hexadecimal com 64 bits:

M = 0123456789ABCDEF

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001
1010 1011 1100 1101 1110 1111

É feita uma permutação inicial **IP** dos 64 bits de dados da mensagem **M**

Tabela 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



Texto Pleno:

M = 0000 0001 0010 0011 0100 0101 0110 0111 1000 1001
1010 1011 1100 1101 1110 1111

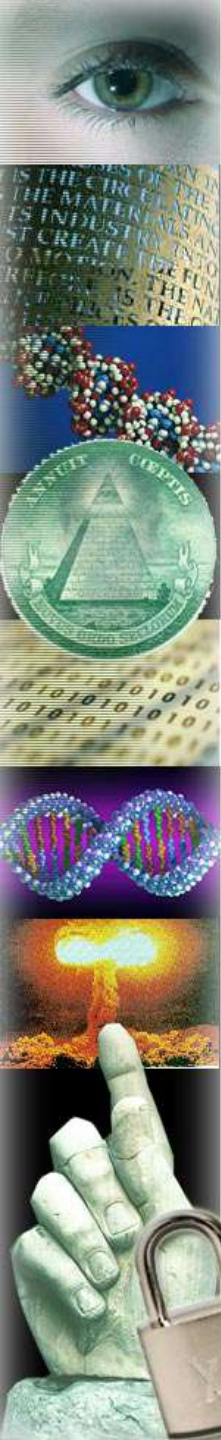
Texto Permutado via tabela IP:

M_{ip} = 1100 1100 0000 0000 1100 1100 1111 1111 1111 0000
1010 1010 1111 0000 1010 1010

A seguir divide-se o texto Permutado em duas partes:

L_o = 1100 1100 0000 0000 1100 1100 1111 1111

R_o = 1111 0000 1010 1010 1111 0000 1010 1010



Entrada 2: Chave em Hexadecimal com 64 bits

K = 133457799BBCDFF1

K = 00010011 00110100 01010111 01111001 10011011 10111100
11011111 11110001

Na verdade é utilizado uma chave de 56 bits, os 8 bits restantes são apenas bits de paridade(bits 8, 16, 24, 32, 40, 48, 56 e 64):

K = 00010011 00110100 01010111 01111001 10011011 10111100
11011111 11110001

A seguir é feita uma permutação, utilizando-se da tabela PC-1

Tabela 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



Da chave original de 64 bits

K = 00010011 00110100 01010111 01111001 10011011 10111100
11011111 11110001

Obtém-se a permutação PC-1 de 56 bits

K+ = 1111000 0110011 0010101 0101111 0101010 1011001 1001111
0001111

A seguir, divide-se esta chave em duas metades, esquerda C_0 e direita D_0 , cada metade com 28 bits.

C_0 = 1111000 0110011 0010101 0101111

D_0 = 0101010 1011001 1001111 0001111



Agora monta-se as chaves K_n , para $1 \leq n \leq 16$, aplicando-se a tabela de permutação abaixo em cada um dos pares concatenados $C_n D_n$. Cada par possui 56 bits, porém **PC-2** usa apenas 48 bits.

Tabela PC-2

14	17	11	21	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



Para a primeira chave tem-se:

$$\mathbf{C}_1 \mathbf{D}_1 = 1110000 \ 1100110 \ 0101010 \ 1011111 \ 1010101 \ 0110011 \\ 0011110 \ 0011110$$

a qual, após aplicarmos a permutação **PC-2** transforma-se em

$$\mathbf{K}_1 = 000110 \ 110000 \ 001011 \ 101111 \ 111111 \ 000111 \ 000001 \\ 110010$$

Para as outras chaves:

$$\mathbf{K}_2 = 011110 \ 011010 \ 111011 \ 011001 \ 110110 \ 111100 \ 100111 \\ 100101 \quad \mathbf{K}_3 = 010101 \ 011111 \ 110010 \ 001010 \ 010000 \ 101100 \\ 111110 \ 011001 \quad \mathbf{K}_4 = 011100 \ 101010 \ 110111 \ 010110 \ 110110 \\ 110011 \ 010100 \ 011101 \quad \mathbf{K}_5 = 011111 \ 001110 \ 110000 \ 000111 \\ 111010 \ 110101 \ 001110 \ 101000$$

...

$$\mathbf{K}_{16} = 110010 \ 110011 \ 110110 \ 001011 \ 000011 \ 100001 \ 011111 \\ 110101$$



Lembrando-se que do texto pleno permutado tem-se:

$$L_o = 1100\ 1100\ 0000\ 0000\ 1100\ 1100\ 1111\ 1111$$

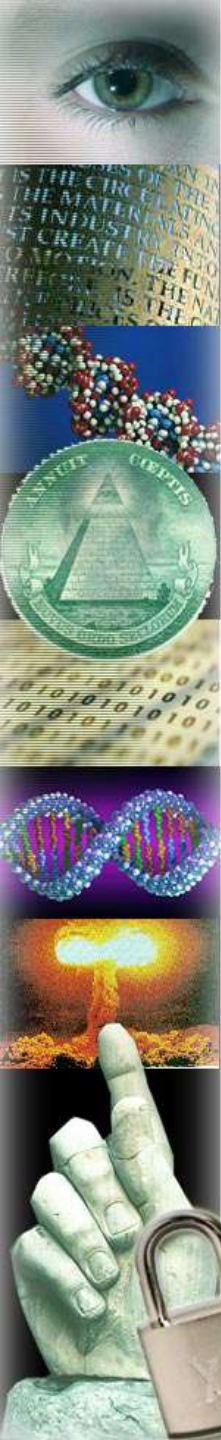
$$R_o = 1111\ 0000\ 1010\ 1010\ 1111\ 0000\ 1010\ 1010$$

Então utilizamos as **Transformadas de Feistel**:

$$L_n = R_{n-1}$$

$$R_n = L_{n-1} + f(R_{n-1}, K_n)$$

Isso é feito para n variando de 1 até 16.



Para $n = 1$ temos

$$\mathbf{K}_1 = 000110 \ 110000 \ 001011 \ 101111 \ 111111 \ 000111 \ 000001 \ 110010$$

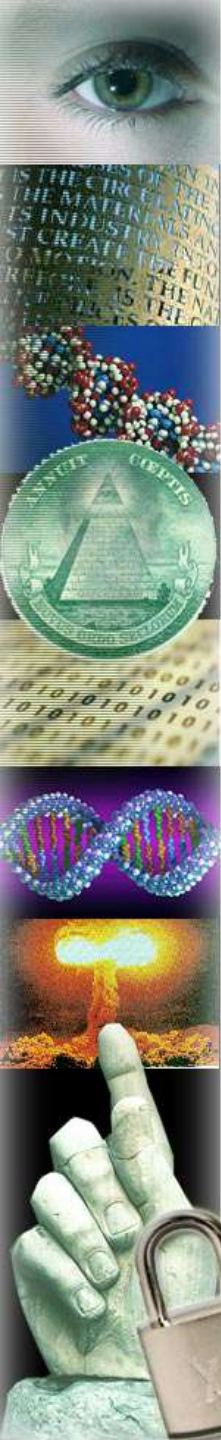
$$\mathbf{L}_1 = \mathbf{R}_0 = 1111 \ 0000 \ 1010 \ 1010 \ 1111 \ 0000 \ 1010 \ 1010$$

$$\mathbf{R}_1 = \mathbf{L}_0 + \mathbf{f}(\mathbf{R}_0, \mathbf{K}_1) = \text{1100 1100 0000 0000 1100 1100 1111 1111} + \mathbf{f}(\text{1111 0000 1010 1010 1111 0000 1010 1010}, \mathbf{K}_1)$$

A função $\mathbf{f}$ transforma o $\mathbf{R}_{n-1}$ que tem 32 bits em 48 bits e em seguida faz um Xor com $\mathbf{K}_n$. A transformação se dá com o uso da tabela E de seleção de bits.

Tabela E

3	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



Exemplo de uma transformação:

$$R_0 = 1111\ 0000\ 1010\ 1010\ 1111\ 0000\ 1010\ 1010$$

$$E(R_0) = \mathbf{011110\ 100001\ 010101\ 010101\ 011110\ 100001\ 010101\ 010101}$$

Xor

$$K_1 = \quad 000110\ 110000\ 001011\ 101111\ 111111\ 000111\ 000001\ 110010$$

=

$$f(R_0, K_1) = \mathbf{011000\ 010001\ 011110\ 111010\ 100001\ 100110\ 010100\ 100111}$$

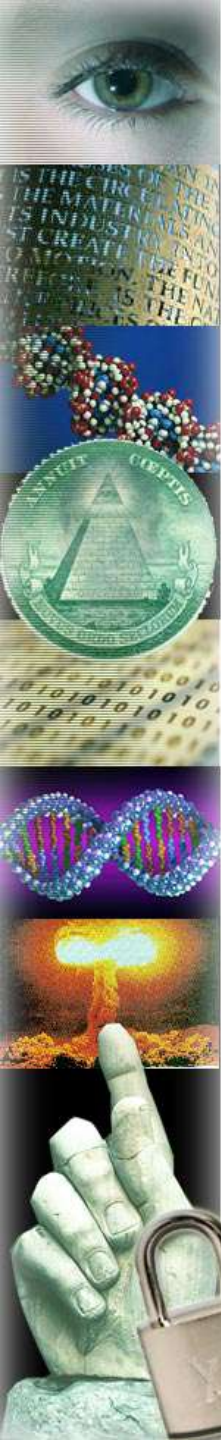
Logo $f(R_0, K_1)$ tem 48 bits, porém teremos que fazer a soma bit a bit com R_0 , que possui apenas 32 bits.

A solução é usar tabelas denominadas de **S-Boxes**.

S-Boxes

		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_1	0 0	14	4	13	1	2	15	11	8	3	10	6	12	5	9	0	7
	0 1	0	15	7	4	14	2	13	1	10	6	12	11	9	5	3	8
	1 0	4	1	14	8	13	6	2	11	15	12	9	7	3	10	5	0
	1 1	15	12	8	2	4	9	1	7	5	11	3	14	10	0	6	13
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_2	0 0	15	1	8	14	6	11	3	4	9	7	2	13	12	0	5	10
	0 1	3	13	4	7	15	2	8	14	12	0	1	10	6	9	11	5
	1 0	0	14	7	11	10	4	13	1	5	8	12	6	9	3	2	15
	1 1	13	8	10	1	3	15	4	2	11	6	7	12	0	5	14	9
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_3	0 0	10	0	9	14	6	3	15	5	1	13	12	7	11	4	2	8
	0 1	13	7	0	9	3	4	6	10	2	8	5	14	12	11	15	1
	1 0	13	6	4	9	8	15	3	0	11	1	2	12	5	10	14	7
	1 1	1	10	13	0	6	9	8	7	4	15	14	3	11	5	2	12
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
S_4	0 0	7	13	14	3	0	6	9	10	1	2	8	5	11	12	4	15
	0 1	13	8	11	5	6	15	0	3	4	7	2	12	1	10	14	9
	1 0	10	6	9	0	12	11	7	13	15	1	3	14	5	2	8	4
	1 1	3	15	0	6	10	1	13	8	9	4	5	11	12	7	2	14

S-Boxes



S_5 :

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

S_6 :

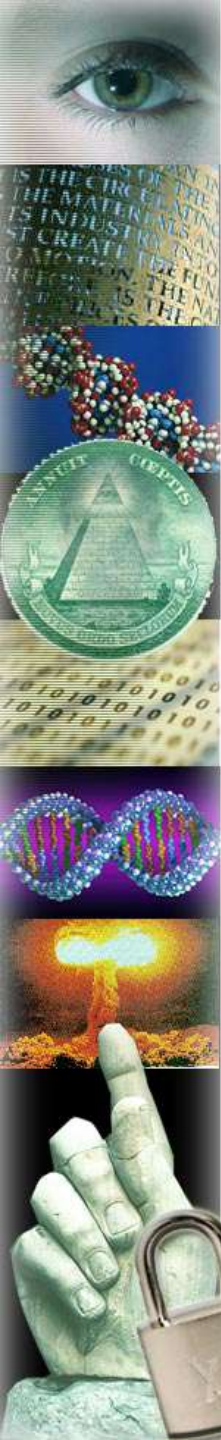
		0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	0	12	1	10	15	9	2	6	8	0	13	3	4	14	7	5	11
0	1	10	15	4	2	7	12	9	5	6	1	13	14	0	11	3	8
1	0	9	14	15	5	2	8	12	3	7	0	4	10	1	13	11	6
1	1	4	3	2	12	9	5	15	10	11	14	1	7	6	0	8	13

S_7 :

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

S_8 :

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



Funcionamento das X-Boxes:

$$F'(R_0, K_1) = 011000 \ 010001 \ 011110 \ 111010 \ 100001 \ 100110 \ 010100 \\ 100111$$

$$F'(R_0, K_1) = S1(B1) \ S2 \ (B1) \ S3 \ (B1) \ S4 \ (B1) \ S5 \ (B1) \ S6 \ (B1) \ S7 \ (B1) \ S8 \ (B1)$$

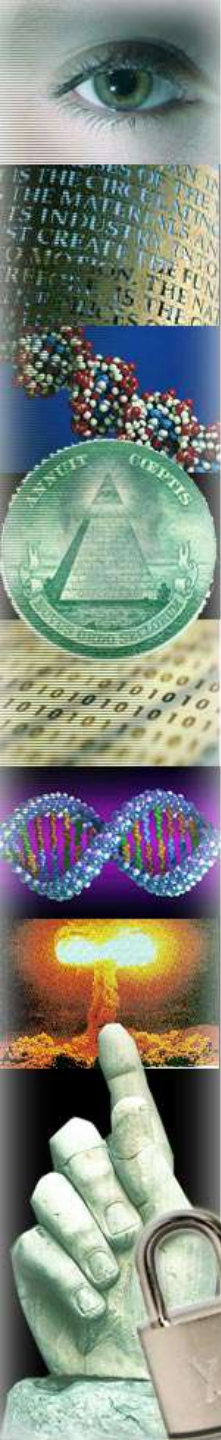
$$F'(R_0, K_1) = 0101 \ 1100 \ 1000 \ 0010 \ 1011 \ 0101 \ 1001 \ 0111$$

Por último é necessário fazer uma permutação com a tabela P:

$$F(R_0, K_1) = 0010 \ 0011 \ 0100 \ 1010 \ 1010 \ 1001 \ 1011 \ 1011$$

Tabela 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	3	9
22	11	4	25



Agora possuímos todos os elementos necessários para calcular R_1 , ou seja, $R_1 = L_0 + f(R_0, K_1)$

$$L_0 = 1100 \ 1100 \ 0000 \ 0000 \ 1100 \ 1100 \ 1111 \ 1111$$

$$f(R_0, K_1) = 0010 \ 0011 \ 0100 \ 1010 \ 1010 \ 1001 \ 1011 \ 1011$$

$$R_1 = L_0 + f(R_0, K_1) = 1110 \ 1111 \ 0100 \ 1010 \ 0110 \ 0101 \ 0100 \ 0100$$

Todos os passos acima são repetidos até a 16ª rodada. Quando estivermos em posse de L_{16} e R_{16} , concatenamos as duas entradas de forma invertida $R_{16} L_{16}$. E em seguida permutamos utilizando-se da tabela IP^{-1} .

Tabela 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



$$\mathbf{R}_{16}\mathbf{L}_{16} = \begin{matrix} 00001010 & 01001100 & 11011001 & 10010101 & 01000011 \\ 01000010 & 00110010 & 00110100 & & \end{matrix}$$

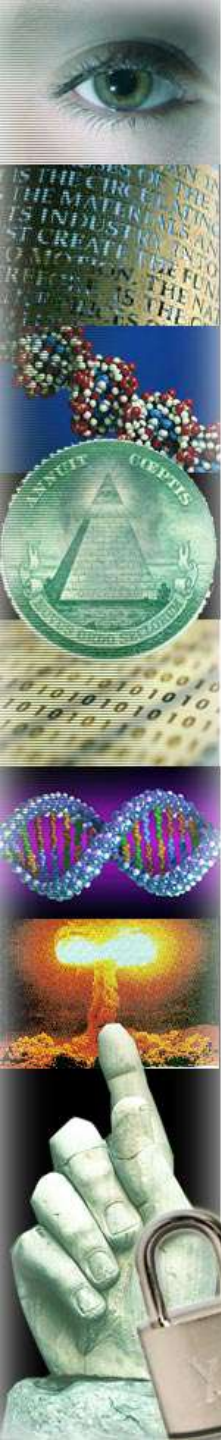
$$\mathbf{IP}^{-1} = \begin{matrix} 10000101 & 11101000 & 00010011 & 01010100 & 00001111 \\ 00001010 & 10110100 & 00000101 & & \end{matrix}$$

o que, em formato hexadecimal, é 85E813540F0AB405.

Portanto, a forma cifrada de $\mathbf{M} = \mathbf{0123456789ABCDEF}$ é

$$\mathbf{C} = \mathbf{85E813540F0AB405}.$$

Decifrar é simplesmente o inverso de cifrar, seguindo os mesmos passos acima descritos porém invertendo a ordem das sub-chaves aplicadas.



Crescimento do Tempo para Quebra de Chaves Simétricas*

Bits	1% do espaço de chave	50% do espaço de chave
56	1 segundo	1 minuto
57	2 segundos	2 minutos
58	4 segundos	4 minutos
64	4,2 minutos	4,2 horas
72	17,9 horas	44,8 dias
80	190,0 dias	31,4 anos
90	535 anos	321 séculos
108	140.000 milênios	8 milhões de milênios
128	147 bilhões de milênios	8 trilhões de milênios

*Fonte: Criptografia e Segurança – Guia Oficial RSA



Em 1998 sob a direção de John Gilmore do EFF, uma equipe gastou US\$ 200.000 e construiu uma máquina que pode analisar todo o espaço de chaves de 56 bits do DES precisando em média 4.5 dias para completar a tarefa. Em 17 de Julho de 1998 eles anunciaram que haviam quebrado uma chave de 56 bits em 46 horas. O computador, chamado de Deep Crack, usa 27 placas, cada uma com 64 chips, e é capaz de testar 90 bilhões de chaves por segundo.

Com o processamento atual, o custo é orçado segundo a tabela abaixo:

Key Search Machine Cost	Expected Search Time
\$10,000	2.5 days
\$100,000	6 hours
\$1,000,000	35 minutes
\$10,000,000	3.5 minutes



Advanced Encryption Standard (AES)

- Em fevereiro de 2001 um novo padrão foi escolhido para substituir o DES, ele é baseado numa cifra de bloco com tamanho de 128 e chaves de tamanhos variáveis de 128, 192 ou 256 bits.

Triple DES

- Triple-DES 1 é apenas o DES efetuado três vezes com duas chaves usadas numa determinada ordem.
- Dada uma mensagem em texto claro, a primeira chave é usada para cifrar a mensagem em DES. A segunda chave é usada para decifrar o DES da mensagem cifrada. Como a segunda chave não é a correta para decifrar, esta decifragem apenas embaralha ainda mais os dados. A mensagem duplamente embaralhada é, então, cifrada novamente com a primeira chave para se obter o criptograma final. Este procedimento em três etapas é chamado de triple-DES 2.

Modos de Operação de Cifras de blocos

•Electronic Code Book:

Seja $E: \{0, 1\}^k \times \{0, 1\}^l \rightarrow \{0, 1\}^l$ uma cifra de bloco. (Cada bloco é cifrado/decifrado individualmente).

Algorithm $\mathcal{E}_K(M)$

If $|M| < l$ then return $\perp$

If $|M| \bmod l \neq 0$ then return $\perp$

Parse M as $M[1] \dots M[n]$

For $i = 1, \dots, n$ do

$C[i] \leftarrow E_K(M[i])$

EndFor

$C \leftarrow C[1] \dots C[n]$

Return C

Algorithm $\mathcal{D}_K(C)$

If $|C| < l$ then return $\perp$

If $|C| \bmod l \neq 0$ then return $\perp$

Parse C as $C = C[1] \dots C[n]$

For $i = 1, \dots, n$ do

$M[i] \leftarrow E_K^{-1}(C[i])$

EndFor

$M \leftarrow M[1] \dots M[n]$

Return M



Modos de Operação de Cifras de blocos

•Cipher-block chaining mode(CBC)

Cada bloco cifrado depende de todos os outros blocos de mensagem anteriores através de uma operação inicial de XOR com um vetor previamente escolhido.

Algorithm $\mathcal{E}_K(M)$

If $|M| < l$ then return $\perp$

If $|M| \bmod l \neq 0$ then return $\perp$

Parse M as $M[1] \dots M[n]$

$C[0] \xleftarrow{R} \{0, 1\}^l$

For $i = 1, \dots, n$ do

$C[i] \leftarrow F_K(C[i-1] \oplus M[i])$

EndFor

$C \leftarrow C[0]C[1] \dots C[n]$

Return C

Algorithm $\mathcal{D}_K(C)$

If $|C| < 2l$ then return $\perp$

If $|C| \bmod l \neq 0$ then return $\perp$

Parse C as $C[0]C[1] \dots C[n]$

For $i = 1, \dots, n$ do

$M[i] \leftarrow E_K^{-1}(C[i]) \oplus C[i-1]$

EndFor

$M \leftarrow M[1] \dots M[n]$

Return M



Modos de Operação de Cifras de blocos

•Counter Mode (R-CTR)

Um valor inicial para o contador é escolhido aleatoriamente a cada mensagem.

Algorithm $\mathcal{E}_K(M)$

```
If  $|M| < L$  then return  $\perp$ 
If  $|M| \bmod L \neq 0$  then return  $\perp$ 
Parse  $M$  as  $M[1] \dots M[n]$ 
 $R \xleftarrow{R} \{0, 1, \dots, 2^l - 1\}$ 
For  $i = 1, \dots, n$  do
     $C[i] \leftarrow F_K(\text{NtS}_l(R + i)) \oplus M[i]$ 
EndFor
 $C[0] \leftarrow \text{NtS}_l(R)$ 
 $C \leftarrow C[0]C[1] \dots C[n]$ 
Return  $C$ 
```

Algorithm $\mathcal{D}_K(C)$

```
If  $|C| < l + L$  then return  $\perp$ 
If  $(|C| - l) \bmod L \neq 0$  then return  $\perp$ 
Let  $C[0]$  be the first  $l$  bits of  $C$ 
Parse the rest of  $C$  as  $C[1] \dots C[n]$ 
 $R \leftarrow \text{StN}(C[0])$ 
For  $i = 1, \dots, n$  do
     $M[i] \leftarrow F_K(\text{NtS}_l(R + i)) \oplus C[i]$ 
EndFor
 $M \leftarrow M[1] \dots M[n]$ 
Return  $M$ 
```

Modos de Operação de Cifras de blocos

•Counter Mode (C-CTR)

Uma variável global (contador) é mantido entre as duas partes, cada vez que o protocolo executa o contador é atualizado.

Algorithm $\mathcal{E}_K(M)$

```
If  $|M| < L$  then return  $\perp$ 
If  $|M| \bmod L \neq 0$  then return  $\perp$ 
Parse  $M$  as  $M[1] \dots M[n]$ 
 $R \xleftarrow{R} \{0, 1, \dots, 2^l - 1\}$ 
For  $i = 1, \dots, n$  do
     $C[i] \leftarrow F_K(\text{NtS}_l(R + i)) \oplus M[i]$ 
EndFor
 $C[0] \leftarrow \text{NtS}_l(R)$ 
 $C \leftarrow C[0]C[1] \dots C[n]$ 
Return  $C$ 
```

Algorithm $\mathcal{D}_K(C)$

```
If  $|C| < l + L$  then return  $\perp$ 
If  $(|C| - l) \bmod L \neq 0$  then return  $\perp$ 
Let  $C[0]$  be the first  $l$  bits of  $C$ 
Parse the rest of  $C$  as  $C[1] \dots C[n]$ 
 $R \leftarrow \text{StN}(C[0])$ 
For  $i = 1, \dots, n$  do
     $M[i] \leftarrow F_K(\text{NtS}_l(R + i)) \oplus C[i]$ 
EndFor
 $M \leftarrow M[1] \dots M[n]$ 
Return  $M$ 
```



•Questões sobre Segurança

- Duas partes compartilham uma chave K gerada aleatoriamente;
- O adversário inicialmente não conhece K ;
- Geralmente se assume que o adversário pode capturar qualquer texto cifrado que segue pelo canal entre as duas partes;
- Queremos explorar as questões sobre qual segurança (privacidade) o esquema pode oferecer.
- A pergunta principal é: “Que tarefas, realizadas pelo adversário, acarretaria num esquema inseguro?”
- É muito mais fácil pensar em insegurança do que em segurança.
- Ex: Se o adversário puder a partir de poucos textos cifrados, derivar a chave K , então o esquema pode ser declarado inseguro.
- A recíproca não é verdadeira !!!



•Questões sobre Segurança

- Possíveis tentativa de descrever um esquema seguro:
 - “Dado C , o adversário não tem idéia do que seja M ”.
 - “Dado C , o adversário não pode recuperar facilmente o texto pleno M ”
- Não podemos fazer suposições sobre o formato dos dados (mensagens) no desenvolvimento de protocolos seguros.
- O esquema “ideal” pode ser visto como
 - “Um anjo recebeu a mensagem M e entregou ao receptor de uma forma mágica.”
- A meta é ter um esquema que tenha propriedades próximas do modelo ideal, ou seja, nenhuma informação parcial pode ser descoberta.



•Questões sobre Segurança

•Exemplo: Considere o esquema ECB.

- Dado o texto cifrado, o adversário pode descobrir a mensagem?
- Considere o caso de uma mensagem de único bloco. Suponha que existam apenas duas mensagens, 01 para "comprar" e 11 para "vender". Uma parte fica enviando dados, mas sempre algum desses dois. O que acontece?
- Em um esquema de encriptação seguro não deve ser possível co-relacionar textos cifrados de diferentes mensagens.
- Esta afirmação tem uma implicação forte. “A encriptação deve ser probabilística ou dependente de uma informação de estado”.
- A noção histórica de encriptação era pensado como um código ou um mapeamento fixo de texto pleno para texto cifrado.
- Um único texto pleno terá muitos possíveis textos cifrados, dependendo das escolhas aleatórias ou do estado do algoritmo. Mas ainda assim, deve ser possível a sua deciptação.



•Definições de Segurança

•Segurança na teoria da informação:

- O adversário inicialmente não conhece K .
- Assume-se que o adversário tem a capacidade de capturar qualquer mensagem de texto cifrado que segue pelo canal entre as duas partes.
- Gostaríamos de dizer que de posse de C o adversário não tem nenhuma idéia do que trata M .
- Entretanto o adversário pode “advinhar” a mensagem ou parte dela.
- Segurança perfeita existe se "a melhor advinhação da mensagem pelo adversário depois de ter visto o texto cifrado é a mesma que ele faria antes de ver o texto cifrado".
- Em outras palavras, o texto cifrado não ajuda na descoberta de novas informações sobre a mensagem.



•Definições de Segurança

•Segurança contra o Ataque de Texto Pleno Escolhido

- Segurança perfeita só pode ser conseguida em esquemas onde o tamanho da chave seja tão grande quanto o tamanho do texto pleno.
- Queremos achar uma noção de segurança que mesmo não sendo “perfeita”, seja boa na prática.
- Intuitivamente, a diferença principal é levar em conta que o adversário tem o poder computacional limitado.
- Pode existir informação no texto cifrado, mas se você não pode computá-las, o texto cifrado na realidade não acrescenta informação.



•Definições de Segurança

•Segurança contra o Ataque de Texto Pleno Escolhido

- O adversário pode escolher duas mensagens do mesmo tamanho. Em seguida, uma é encriptada e dada ao adversário. O esquema é seguro se o adversário precisar de muito tempo para dizer qual mensagem foi encriptada.

- Por fim, consideremos uma sequência de mensagens. Teremos uma sequência de pares de mensagens do mesmo tamanho

$$(M_{1,0}, M_{1,1}), \dots, (M_{q,0}, M_{q,1})$$

- Em seguida, um bit “de desafio” b é escolhido aleatoriamente e a sequência $C_1, \dots, C_q$ é produzida, onde $C_i \leftarrow \mathcal{E}_K(M_{i,b})$.

- O adversário recebe a sequência de textos cifrados e deve adivinhar b para vencer. Em outras palavras, o adversário está tentando descobrir se o emissor enviou $M_{1,0}, \dots, M_{q,0}$ or $M_{1,1}, \dots, M_{q,1}$.



•Definições de Segurança

•Segurança contra o Ataque de Texto Pleno Escolhido

•Formalizando:

- Considere o adversário como um programa A , com acesso a um oráculo para o qual ele fornece um par de entradas (M_0, M_1) .
- O oráculo pode computar o texto encriptado de duas maneiras diferentes, correspondentes aos dois “mundos” onde o adversário “vive”.
- Definimos o Oráculo de encriptação direita ou esquerda como segue:

Oracle $\mathcal{E}_K(\text{LR}(M_0, M_1, b)) \quad // \quad b \in \{0, 1\} \text{ and } M_0, M_1 \in \{0, 1\}^*$
 $C \leftarrow \mathcal{E}_K(M_b)$
Return C

- **Definições de Segurança**

- **Segurança contra o Ataque de Texto Pleno Escolhido**

Mundo 0

O Oráculo fornecido ao adversário é:

$$\mathcal{E}_K(\text{LR}(\cdot, \cdot, 0))$$

Sempre que o adversário fizer uma consulta (M_0, M_1) , receberá:

$$C \xleftarrow{R} \mathcal{E}_K(M_0)$$

Mundo 1

O Oráculo fornecido ao adversário é:

$$\mathcal{E}_K(\text{LR}(\cdot, \cdot, 1))$$

Sempre que o adversário fizer uma consulta (M_0, M_1) , receberá:

$$C \xleftarrow{R} \mathcal{E}_K(M_1)$$



•Definições de Segurança

•Segurança contra o Ataque de Texto Pleno Escolhido

- A tarefa do adversário é dizer em que “mundo” ele “vive”, após um certo tempo “conversando” com o oráculo.
- Observação:** A escolha do “mundo” é feita uma vez, antes do início da execução do adversário.
- Considera-se um esquema seguro contra o ataque de texto pleno escolhido se um adversário “razoável” não puder obter vantagem significativa em distinguir os casos em que $b = 0$ e $b = 1$.
- Esta noção é chamada de indisguibilidade sobre o ataque de texto pleno escolhido IND-CPA.

- **Definições de Segurança**

- **Segurança contra o Ataque de Texto Pleno Escolhido**

- Considere o experimento:

Experiment $\text{Exp}_{\mathcal{SE}, A}^{\text{ind-cpa-}b}$

$$K \xleftarrow{R} \mathcal{K}$$
$$d \leftarrow A^{\mathcal{E}_K(\text{LR}(\cdot, \cdot, b))}$$

Return d

- A *ind-cpa-advantage* de A é definida como:

$$\text{Adv}_{\mathcal{SE}, A}^{\text{ind-cpa}} = \mathbf{P} \left[\text{Exp}_{\mathcal{SE}, A}^{\text{ind-cpa-1}} = 1 \right] - \mathbf{P} \left[\text{Exp}_{\mathcal{SE}, A}^{\text{ind-cpa-0}} = 1 \right]$$

$$\text{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(t, q, \mu) = \max_A \left\{ \text{Adv}_{\mathcal{SE}, A}^{\text{ind-cpa}} \right\}$$

- Onde o máximo é sobre todos A tendo complexidade de tempo t , fazendo no máximo q consultas ao oráculo, de soma no máximo μ bits.

- Definições de Segurança

- Segurança contra o Ataque de Texto Pleno Escolhido

- Ataque contra o modo ECB

Adversary $A^{\mathcal{E}_K(\text{LR}(\cdot, \cdot, b))}$

$M_1 \leftarrow 0^{2^l}$; $M_0 \leftarrow 0^l \| 1^l$

$C[1]C[2] \leftarrow \mathcal{E}_K(\text{LR}(M_0, M_1, b))$

If $C[1] = C[2]$ then return 1 else return 0

Daí, temos:

$$\mathbf{P} \left[\text{Exp}_{\mathcal{SE}, A}^{\text{ind-cpa-1}} = 1 \right] = 1$$

$$\mathbf{P} \left[\text{Exp}_{\mathcal{SE}, A}^{\text{ind-cpa-0}} = 1 \right] = 0 .$$

Logo, com apenas 1 consulta: $\text{Adv}_{\mathcal{SE}, A}^{\text{ind-cpa}} = 1 - 0 = 1$.

Conclusão: ECB é inseguro mesmo utilizando uma cifra de bloco E segura.

Pode-se provar de forma semelhante que qualquer esquema determinístico e sem estado é inseguro.

A vertical collage of seven images. From top to bottom: a close-up of a human eye; a DNA double helix structure; a green pyramid with the word 'EUREKA' on the left and 'CORPUS' on the right, with a banner at the base; a background of binary code (0s and 1s); two DNA double helix structures; a nuclear mushroom cloud; and a hand holding a padlock.

A vertical collage of seven images. From top to bottom: 1. A close-up of a green eye. 2. A DNA double helix structure. 3. A green pyramid with the word 'EUREKA' on the left and 'CORPUS' on the right, with a banner at the base. 4. A background of binary code (0s and 1s). 5. Two DNA double helix structures. 6. A nuclear explosion with a mushroom cloud. 7. A hand holding a padlock.

-
- A vertical collage of seven images. From top to bottom: 1. A close-up of a green eye. 2. A DNA double helix structure. 3. A green pyramid with the word 'EUREKA' on the left and 'CORPUS' on the right, with a banner at the base. 4. A background of binary code (0s and 1s). 5. Two DNA double helix structures. 6. A nuclear explosion with a mushroom cloud. 7. A hand holding a padlock.



•Definições de Segurança

•Segurança contra o Ataque de Recuperação de Texto Pleno

- O resultado a seguir é o que se deseja chegar com esta abordagem:
“Provar a segurança de uma construção criptográfica de alto nível baseado em suposições de alguma primitiva”.

•Segurança do Modo CTR contra o Ataque de Texto Pleno Escolhido

$$\text{Adv}_{\mathcal{SE}}^{\text{ind-cpa}}(t, q, \mu) \leq 2 \cdot \text{Adv}_F^{\text{prf}}(t, q', lq'),$$



•Definições de Segurança

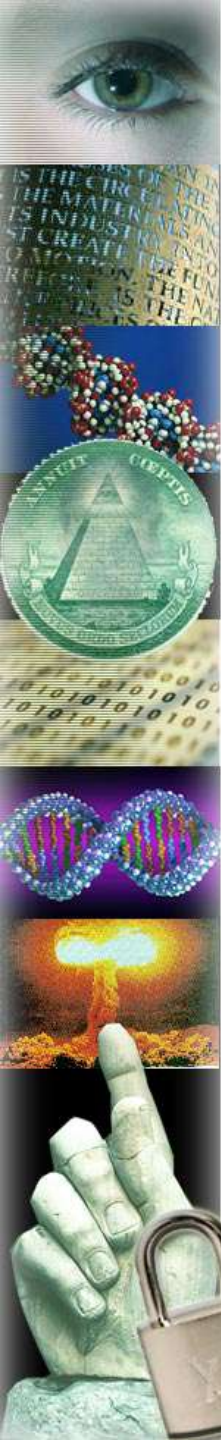
•Segurança contra o Ataque de Texto Cifrado Escolhido

- É um tipo de ataque mais forte que o CPA.
- Neste tipo de ataque o adversário tem acesso a um oráculo de deciptação. Ele pode fornecer um texto cifrado ao oráculo e recuperar o texto pleno.
- Não é permitido consultar C ao oráculo de deciptação, tendo C sido retornada pelo oráculo LR .
- Esta restrição ainda deixa o adversário com muito poder. Tipicamente, um ataque CCA de sucesso ocorre da seguinte forma:
“O adversário pega o texto cifrado C retornado pelo oráculo LR, modifica para C' , e chama o oráculo de deciptação com C' .”
- Ataques de CCA são poderosos suficiente para quebrar todos os modos de operação padrões.



Referências:

- Criptografia e Segurança – Guia Oficial RSA – Burnett, Paine
- Lecture Notes on Cryptography – Godwasser, Bellare
- Site - <http://www.numaboa.com.br>



**“Quousque tandem abutere, Naor,
patientia nostra? quam diu etiam furor
iste tuus nos eludet? quem ad finem sese
effrenata iactabit audacia?”**

Cícero

**“Até quando, ó Naor, abusarás de nossa paciência? Por quanto
tempo ainda há-de zombar de nós essa tua loucura? A que
extremos se há-de precipitar a tua audácia sem freio?”**

