Cifragem utilizando o algoritmo AES

FURB
UNIVERSIDADE DE BLUMENAU

Exemplo de cifragem

```
KeyGenerator keyGenerator;
keyGenerator = KeyGenerator.getInstance("AES");
keyGenerator.init(128);
SecretKey chave = keyGenerator.generateKey();

Path arquivoTextoSimples = Paths.get("etc/exemplo.pdf");
byte[] textoSimples = Files.readAllBytes(arquivoTextoSimples);

Cipher cifrador = Cipher.getInstance("AES/ECB/PKCS5Padding");
cifrador.init(Cipher.ENCRYPT_MODE, chave);
byte[] dadosCifrados = cifrador.doFinal(textoSimples);

Path arquivoTextoCifrado = Paths.get("etc/saida.bin");
Files.write(arquivoTextoCifrado, dadosCifrados);
```

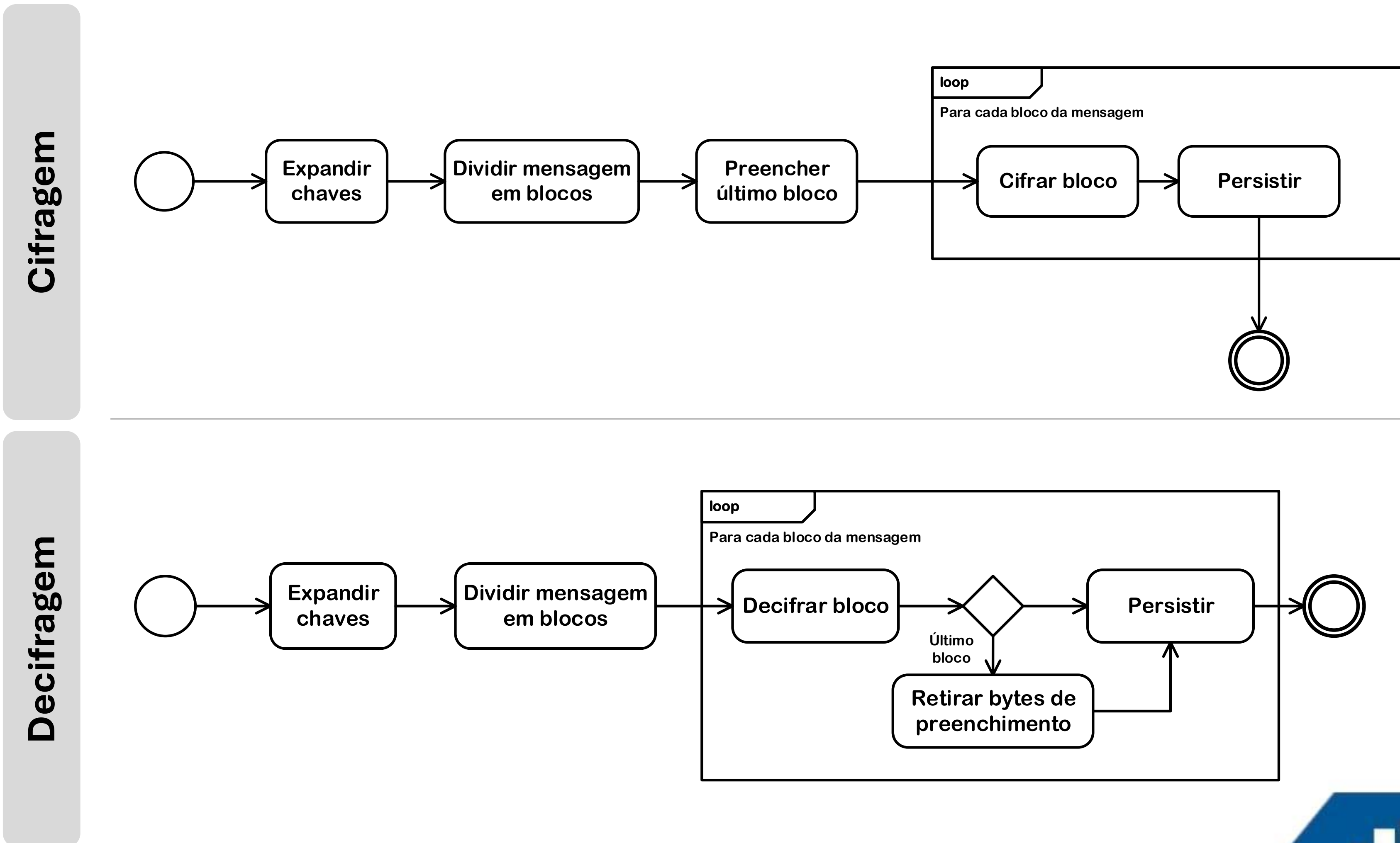
Definições

- Cifragem de bloco

- Utiliza chave de 128, 192 ou 256 bits.
  - Vamos estudar o algoritmo para chaves de 128 bits

- Utiliza bloco de 128 bits

Processos de cifragem e decifragem com AES



Matriz de estado

- O algoritmo utiliza matrizes 4 x 4 para efetuar o processamento.

- Esta matriz é chamada de **matriz de estado**:

$$\begin{pmatrix} b_0 & b_4 & b_8 & b_{12} \\ b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \end{pmatrix}$$

- O Algoritmo AES utiliza a notação “palavra” (*word*) que consiste em 4 bytes. Assim, cada coluna é uma palavra, bem como cada linha.

Expansão de chave

Expansão de chave

Conceitos

Os 16 bytes da chave de criptografia são representados numa matriz de estado. Os quatro primeiros bytes ocupam a primeira coluna. Os próximos 4 bytes ocupam a segunda coluna, e assim por diante.

$$\begin{pmatrix} k_0 & k_4 & k_8 & k_{12} \\ k_1 & k_5 & k_9 & k_{13} \\ k_2 & k_6 & k_{10} & k_{14} \\ k_3 & k_7 & k_{11} & k_{15} \end{pmatrix}$$

$$\Downarrow$$

$$w_0 \quad w_1 \quad w_2 \quad w_3$$

A chave forma 4 palavras denominadas de w_0, w_1, w_2 e w_3.

Expansão de chave

Exemplo

Exemplo: supor que a chave seja "**ABCDEFGHIJKLMNOP**".

Sua representação na matriz de estado seria:

$$\begin{pmatrix} A & E & I & M \\ B & F & J & N \\ C & G & K & O \\ D & H & L & P \end{pmatrix}$$

Ou, em bytes:

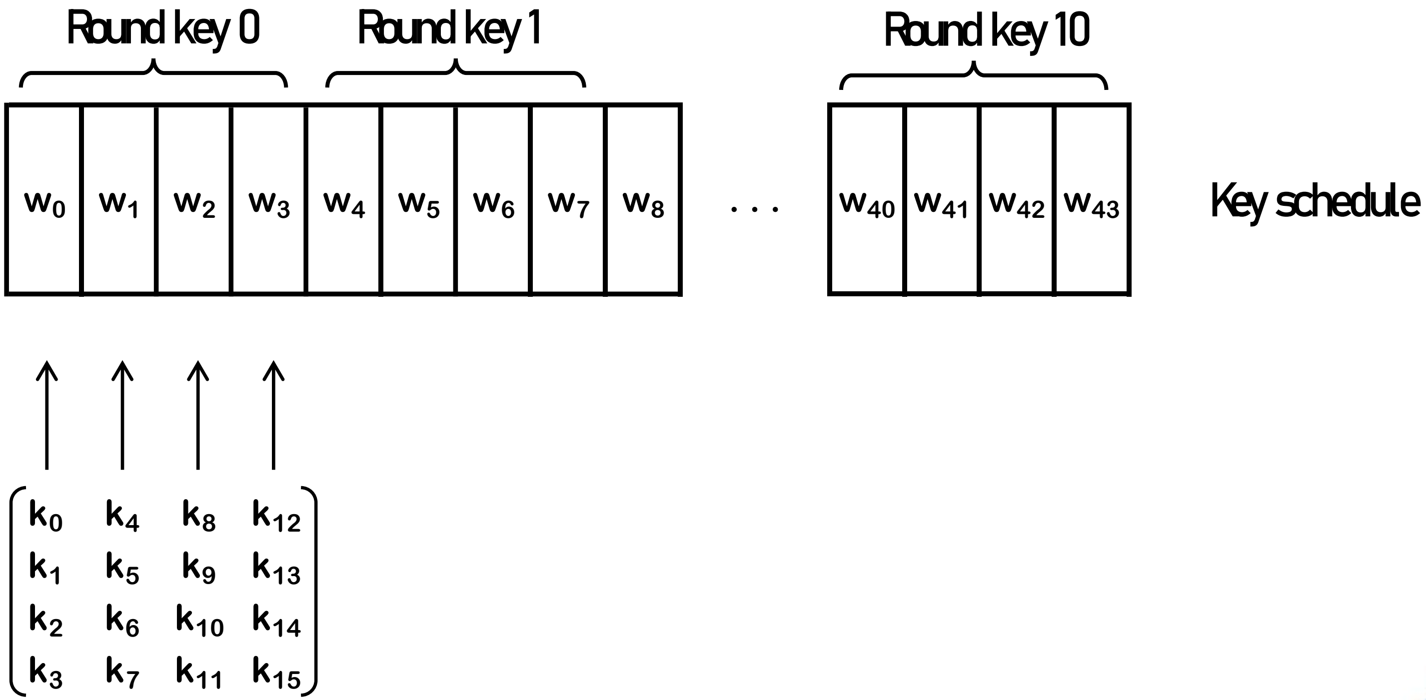
$$\begin{pmatrix} 65 & 69 & 73 & 77 \\ 66 & 70 & 74 & 78 \\ 67 & 71 & 75 & 79 \\ 68 & 72 & 76 & 80 \end{pmatrix} \quad \text{ou} \quad \begin{pmatrix} 0x41 & 0x45 & 0x49 & 0x4D \\ 0x42 & 0x46 & 0x4A & 0x4E \\ 0x43 & 0x47 & 0x4B & 0x4F \\ 0x44 & 0x48 & 0x4C & 0x50 \end{pmatrix}$$

formato decimal

formato hexadecimal

Expansão de chave

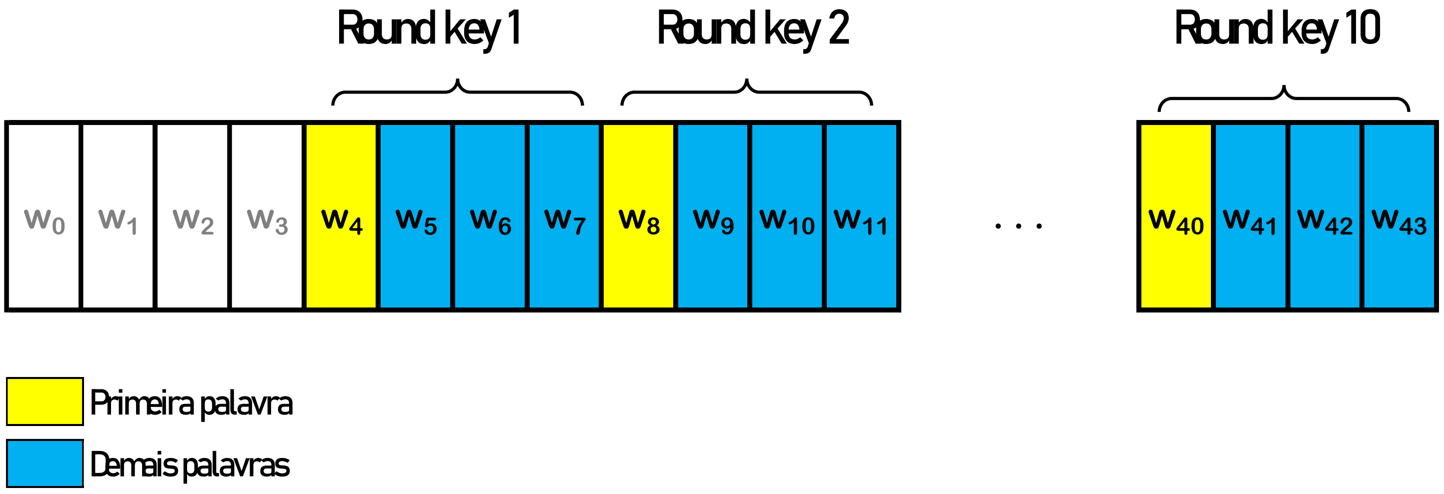
- O algoritmo expande as palavras w_0, w_1, w_2 e $w_{3,}$ gerando 10 novas chaves. Cada chave é chamada de **round key.**
- As round keys são distribuídas numa tabela denominada de **key schedule.** Esta tabela contém 11 chaves (a chave original mais as 10 chaves derivadas)



Expansão de chave

Existem dois algoritmos para geração das palavras da round key:

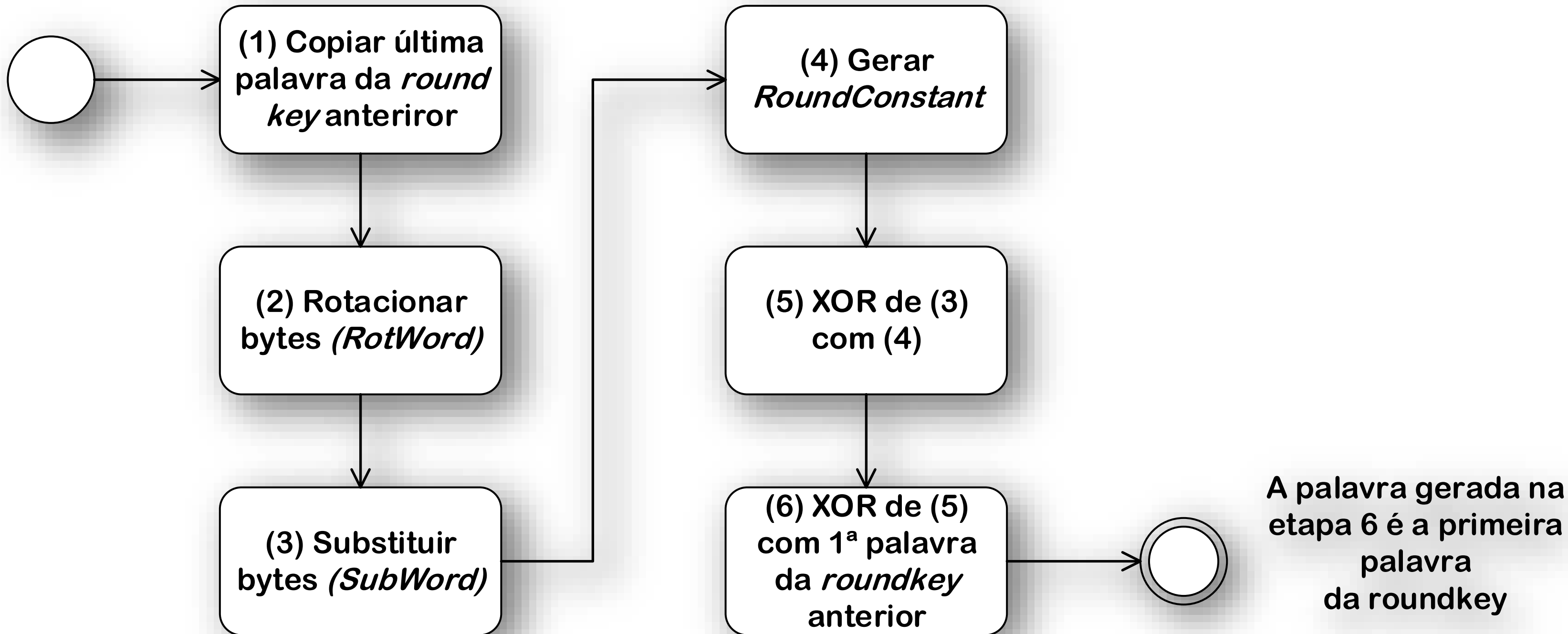
- Algoritmo para a primeira palavra
- Algoritmo para as demais palavras



Expansão de chave

Geração da primeira palavra da round key

A geração da primeira palavra de uma *round key* requer realizar as seguintes operações:



FURB
UNIVERSIDADE DE BLUMENAU

(2) Rotacionar bytes

Geração da primeira palavra da round key

- Deve-se rotacionar os bytes da primeira palavra conforme visto abaixo

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} \Rightarrow \begin{bmatrix} b_1 \\ b_2 \\ b_3 \\ b_0 \end{bmatrix}$$

- Esta etapa também é conhecida como *RotWord*

(2) Rotacionar bytes - Exemplo

Geração da primeira palavra da round key

- Exemplo:

0x41	0x45	0x49	0x4D					
0x42	0x46	0x4A	0x4E					
0x43	0x47	0x4B	0x4F					
0x44	0x48	0x4C	0x50					
w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	w_8

Key schedule

- Depois de copiada a palavra w_3, rotaciona-se os bytes

0x4D	⇨	0x4E
0x4E		0x4F
0x4F		0x50
0x50		0x4D

3 – Substituição de palavra

Geração da primeira palavra da *round key*

- Os bytes da palavra são substituídos por outros valores, que são obtidos de uma tabela denominada de S-BOX:

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
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

S-Box

Utiliza-se o próprio valor para obter as coordenadas da S-BOX. Considerar que:

- Os 4 bits mais significativos representam a linha
- Os 4 bits menos significativos representam a coluna

3 – Substituição de palavra

Geração da primeira palavra da *round key*

- Exemplo:

$$\begin{pmatrix} 0x4e \\ 0x4f \\ 0x50 \\ 0x4d \end{pmatrix}$$

Resultado da etapa 2 (RotWord)

0x4e

78

0100 1110

	0	1	2	3	4	5	6	7	8	9	a	b	c	d	e	f
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

$$\begin{pmatrix} 0x2f \\ 0x84 \\ 0x53 \\ 0xe3 \end{pmatrix}$$

Resultado da etapa 3 (SubWord)

4 – Geração da RoundConstant

Geração da primeira palavra da roundKey

- A ***RoundConstant*** é uma palavra que possui a seguinte composição:
  - O primeiro byte é relativo ao número da *roundKey*:

    Sendo `i` o número da *roundKey,* o valor do 1º byte será:

i	1	2	3	4	5	6	7	8	9	10
valor	0x01	0x02	0x04	0x08	0x10	0x20	0x40	0x80	0x1B	0x36

  - Os demais bytes da palavra são 0 (zero)

- Exemplo:
  - A *RoundConstant* da 6ª *RoundKey* é:

$$\begin{pmatrix} \mathbf{0x20} \\ \mathbf{0x00} \\ \mathbf{0x00} \\ \mathbf{0x00} \end{pmatrix}$$

5 – XOR das etapas (3) e (4)

Geração da primeira palavra da round Key

- Nesta etapa, é feita uma operação XOR da palavra da etapa 3 (isto é, após aplicada a substituição com a S-Box) com a *Round Constante* (isto é, palavra da etapa 4)

- Exemplo:

$$\underbrace{\begin{pmatrix} \text{0x2f} \\ \text{0x84} \\ \text{0x53} \\ \text{0xe3} \end{pmatrix}}_{\substack{\text{Resultado} \\ \text{da etapa 3} \\ (\textit{SubWord})}} \oplus \underbrace{\begin{pmatrix} \text{0x01} \\ \text{0x00} \\ \text{0x00} \\ \text{0x00} \end{pmatrix}}_{\substack{\text{Resultado} \\ \text{da etapa 4} \\ (\textit{RoundConstant})}} = \begin{pmatrix} \text{0x2e} \\ \text{0x84} \\ \text{0x53} \\ \text{0xe3} \end{pmatrix}$$

6 – XOR de (5) com a 1ª palavra da roundkey anterior

Geração da primeira palavra da *roundKey*

- Nesta última etapa, faz-se um XOR da primeira palavra da *roundKey* anterior com a palavra obtida na etapa 5

Roundkey 0				Roundkey 1				
0x41	0x45	0x49	0x4D					. . .
0x42	0x46	0x4A	0x4E					
0x43	0x47	0x4B	0x4F					
0x44	0x48	0x4C	0x50					
w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	

$$\underbrace{\begin{pmatrix} 0x41 \\ 0x42 \\ 0x43 \\ 0x44 \end{pmatrix}}_{w_0} \oplus \underbrace{\begin{pmatrix} 0x2e \\ 0x84 \\ 0x53 \\ 0xe3 \end{pmatrix}}_{\text{Resultado da etapa 5}} = \begin{pmatrix} 0x6f \\ 0xc6 \\ 0x10 \\ 0xa7 \end{pmatrix}$$

Exemplo

w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7
0x41	0x45	0x49	0x4D	0x6F			
0x42	0x46	0x4A	0x4E	0xC6			
0x43	0x47	0x4B	0x4F	0x10			
0x44	0x48	0x4C	0x50	0xA7			

	⇨	*RotWord (2)*	⇨	*SubWord (3)*	⇨	*XOR com RoundConstant (5)*	⇨	*XOR com RoundKey anterior (6)*
0x4D		0x4E		0x2F		0x2E		0x6F
0x4E		0x4F		0x84		0x84		0xC6
0x4F		0x50		0x53		0x53		0x10
0x50		0x4D		0xE3		0xE3		0xA7

Expansão de chaves (round key)

Demais palavras

A geração das demais palavras da *round key* consiste essencialmente em operações XOR com a palavra imediatamente anterior e a palavra de posição equivalente na *round key* anterior.

Round key 0				**Round key 1**					**Round key 10**			
w_0	w_1	w_2	w_3	w_4	w_5	w_6	w_7	...	w_{40}	w_{41}	w_{42}	w_{43}

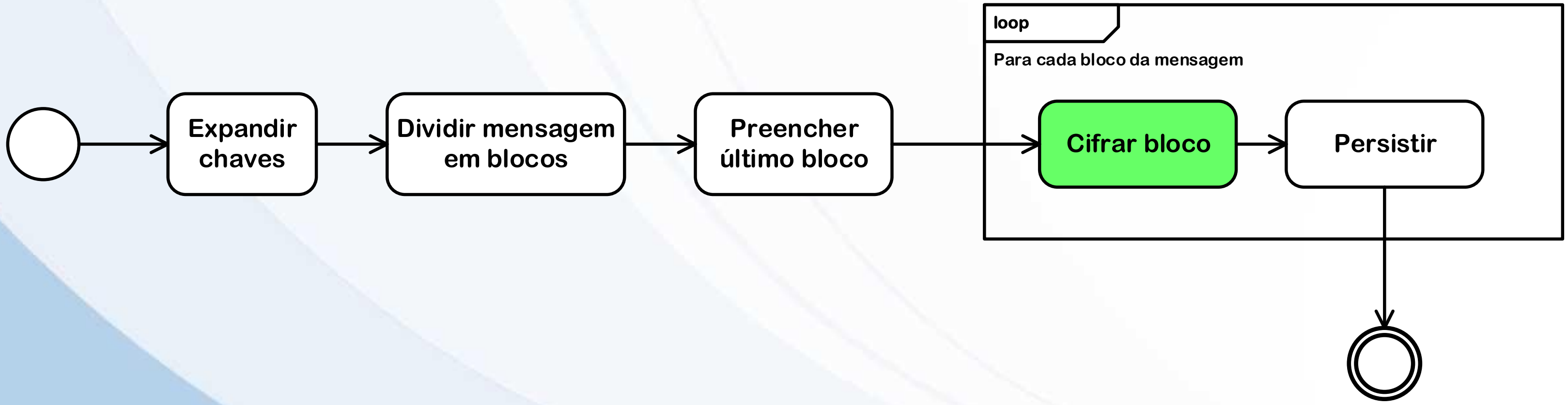
$$w_5 \leftarrow w_1 \oplus w_4 \qquad w_{41} \leftarrow w_{37} \oplus w_{40}$$

$$w_6 \leftarrow w_2 \oplus w_5 \qquad w_{42} \leftarrow w_{38} \oplus w_{41}$$

$$w_7 \leftarrow w_3 \oplus w_6 \qquad w_{43} \leftarrow w_{39} \oplus w_{42}$$

$\oplus$ Operação XOR

Cifragem de um bloco de 128 bits



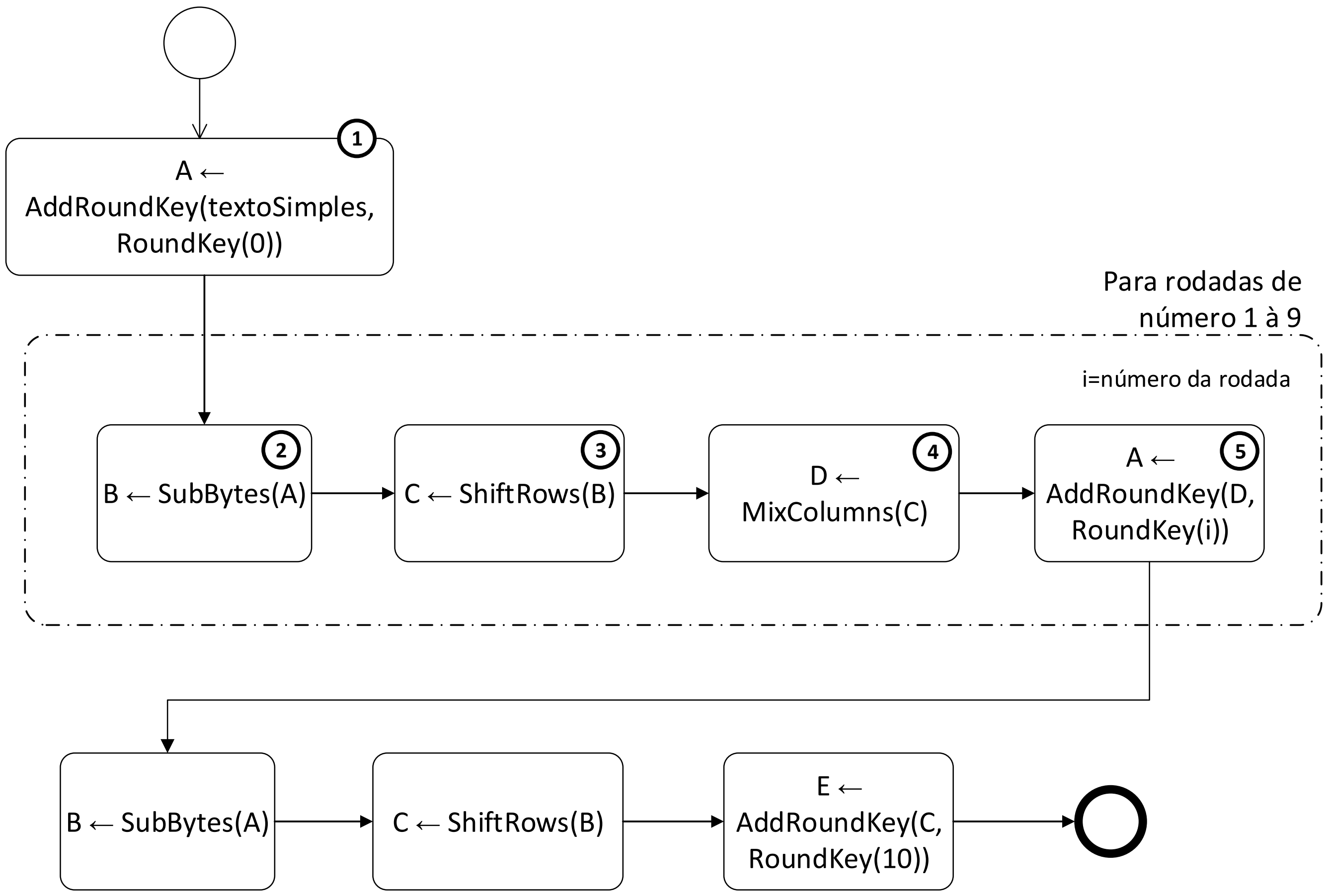
Visão geral

- O algoritmo é organizado em rodadas, onde o texto simples é subordinado à múltiplas rodadas de processamento.
  - Para cada rodada há uma **matriz de estado de entrada** e se produz uma **matriz de estado de saída**

- A matriz de estado de saída produzida na última rodada é organizada num bloco de saída (cifrado) de 128 bits.

Visão geral

- A criptografia consiste na execução de 10 rodadas



Visão geral

- Os dados do texto simples são armazenados numa matriz de estado. Os primeiros quatro bytes ocupam a primeira coluna. Os próximos 4 bytes ocupam a segunda coluna, e assim por diante:

- Exemplo: Texto simples: “DESENVOLVIMENTO!”

- Representação em formato hexadecimal:

0x44 0x45 0x53 0x45 0x4e 0x56 0x4f 0x4c 0x56 0x49 0x4d 0x45 0x4e 0x54 0x4f 0x21

$$\begin{bmatrix} 0x44 & 0x4e & 0x56 & 0x4e \\ 0x45 & 0x56 & 0x49 & 0x54 \\ 0x53 & 0x4f & 0x4d & 0x4f \\ 0x45 & 0x4c & 0x45 & 0x21 \end{bmatrix}$$

Etapa 1 – AddRoundKey(textoSimples,RoundKey(0))

- A função AddRoundKey() é uma função que realiza a operação XOR entre duas matrizes de estado.
- Nesta etapa, as matrizes são o texto simples e a roundKey inicial (roundKey 0).
  - *RoundKey 0 contém a chave original*

- Exemplo:

$$\underbrace{\begin{bmatrix} \texttt{0x44} & \texttt{0x4e} & \texttt{0x56} & \texttt{0x4e} \\ \texttt{0x45} & \texttt{0x56} & \texttt{0x49} & \texttt{0x54} \\ \texttt{0x53} & \texttt{0x4f} & \texttt{0x4d} & \texttt{0x4f} \\ \texttt{0x45} & \texttt{0x4c} & \texttt{0x45} & \texttt{0x21} \end{bmatrix}}_{\textbf{Texto simples}} \oplus \underbrace{\begin{bmatrix} \texttt{0x41} & \texttt{0x45} & \texttt{0x49} & \texttt{0x4d} \\ \texttt{0x42} & \texttt{0x46} & \texttt{0x4a} & \texttt{0x4e} \\ \texttt{0x43} & \texttt{0x47} & \texttt{0x4b} & \texttt{0x4f} \\ \texttt{0x44} & \texttt{0x48} & \texttt{0x4c} & \texttt{0x50} \end{bmatrix}}_{\textbf{Chave / RoundKey(0)}} = \begin{bmatrix} \texttt{0x05} & \texttt{0x0b} & \texttt{0x1f} & \texttt{0x03} \\ \texttt{0x07} & \texttt{0x10} & \texttt{0x03} & \texttt{0x1a} \\ \texttt{0x10} & \texttt{0x08} & \texttt{0x06} & \texttt{0x00} \\ \texttt{0x01} & \texttt{0x04} & \texttt{0x09} & \texttt{0x71} \end{bmatrix}$$

Etapa 2 - SubBytes

- Nesta etapa, uma nova matriz de estado é construída. Seu conteúdo é originado do resultado da etapa 1 e utiliza-se a S-Box para substituir cada byte desta matriz.

- Exemplo:

0x05	0x0b	0x1f	0x03
0x07	0x10	0x03	0x1a
0x10	0x08	0x06	0x00
0x01	0x04	0x09	0x71

Resultado da etapa 1

⇨

0x6b	0x2b	0xc0	0x7b
0xc5	0xca	0x7b	0xa2
0xca	0x30	0x6f	0x63
0x7c	0xf2	0x01	0xa3

Matriz de estado resultante

Etapa 3 - ShiftRows

- Uma matriz de estado é construída partindo do resultado da etapa 2 mas embaralhando os bytes da seguinte forma:

$$\begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix} \Rightarrow \begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,1} & b_{1,2} & b_{1,3} & b_{1,0} \\ b_{2,2} & b_{2,3} & b_{2,0} & b_{2,1} \\ b_{3,3} & b_{3,0} & b_{3,1} & b_{3,2} \end{bmatrix}$$

- Exemplo:

```
0x6b 0x2b 0xc0 0x7b
0xc5 0xca 0x7b 0xa2
0xca 0x30 0x6f 0x63
0x7c 0xf2 0x01 0xa3
```

⇨

```
0x6b 0x2b 0xc0 0x7b
0xca 0x7b 0xa2 0xc5
0x6f 0x63 0xca 0x30
0xa3 0x7c 0xf2 0x01
```

Etapa 4 - MixColumns

- Uma nova matriz de estado é construída:

$$\begin{bmatrix} b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \\ b_4 & b_8 & b_{12} & b_{16} \end{bmatrix}$$

- Seu novo conteúdo depende de uma *matriz de multiplicação*, cujo conteúdo é:

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Etapa 4 - MixColumns

- Temos três matrizes:

$$\begin{bmatrix} b_1 & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \\ b_4 & \boxed{b_8} & b_{12} & b_{16} \end{bmatrix}$$

Matriz resultante da 4ª etapa

Coluna 2, Linha 4

$$\begin{bmatrix} r_1 & r_5 & r_9 & r_{13} \\ r_2 & r_6 & r_{10} & r_{14} \\ r_3 & r_7 & r_{11} & r_{15} \\ r_4 & r_8 & r_{12} & r_{16} \end{bmatrix}$$

Matriz resultante da 3ª etapa (ShiftRows)

$$\begin{bmatrix} 2 & 3 & 1 & 1 \\ 1 & 2 & 3 & 1 \\ 1 & 1 & 2 & 3 \\ 3 & 1 & 1 & 2 \end{bmatrix}$$

Matriz de multiplicação

- O valor de b_n, que está na linha x e coluna y, da matriz resultante é calculado usando-se:
  - Uma palavra (da vertical) da matriz resultante da 3ª etapa (ShiftRows)
  - Uma palavra (da horizontal) da matriz de multiplicação
- Através da seguinte fórmula (exemplo):

$$b_8 = (r_5 * 3)\ xor\ (r_6 * 1)\ xor\ (r_7 * 1)\ xor\ (r_8 * 2)$$

Etapa 4 - MixColumns

- A operação de multiplicação na etapa MixColumns é uma multiplicação *no Campo de Galois*. Não é uma operação de multiplicação tradicional.

$$b_1 = (r_1 * 2)\ xor\ (r_2 * 3)\ xor\ (r_3 * 1)\ xor\ (r_4 * 1)$$

Multiplicação no *campo de Galois* com os termos r_1 e 2

- Se um dos termos for 0, o resultado da multiplicação é 0.
- Se um dos termos for 1, o resultado da multiplicação é igual ao outro termo
- Se os termos não forem 0 e nem 1, deve-se recorrer à **tabela L** e à **tabela E**

Etapa 4 – MixColumns

Multiplicação de Galois

Quando nenhum dos termos da *multiplicação de Galois* for 0 ou 1 :

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	19	01	32	02	1a	c6	4b	c7	1b	68	33	ee	df	03
1	64	04	e0	0e	34	8d	81	ef	4c	71	08	c8	f8	69	1c	c1
2	7d	c2	1d	b5	f9	b9	27	6a	4d	e4	a6	72	9a	c9	09	78
3	65	2f	8a	05	21	0f	e1	24	12	f0	82	45	35	93	da	8e
4	96	8f	db	bd	36	d0	ce	94	13	5c	d2	f1	40	46	83	38
5	66	dd	fd	30	bf	06	8b	62	b3	25	e2	98	22	88	91	10
6	7e	6e	48	c3	a3	b6	1e	42	3a	6b	28	54	fa	85	3d	ba
7	2b	79	0a	15	9b	9f	5e	ca	4e	d4	ac	e5	f3	73	a7	57
8	af	58	a8	50	f4	ea	d6	74	4f	ae	e9	d5	e7	e6	ad	e8
9	2c	d7	75	7a	eb	16	0b	f5	59	cb	5f	b0	9c	a9	51	a0
A	7f	0c	f6	6f	17	c4	49	ec	d8	43	1f	2d	a4	76	7b	b7
B	cc	bb	3e	5a	fb	60	b1	86	3b	52	a1	6c	aa	55	29	9d
C	97	b2	87	90	61	be	dc	fc	bc	95	cf	cd	37	3f	5b	d1
D	53	39	84	3c	41	a2	6d	47	14	2a	9e	5d	56	f2	d3	ab
E	44	11	92	d9	23	20	2e	89	b4	7c	b8	26	77	99	e3	a5
F	67	4a	ed	de	c5	31	fe	18	0d	63	8c	80	c0	f7	70	07

Tabela L

Precisa-se obter dois números da Tabela L (obtém-se um número para cada termo da multiplicação).

Dado um termo da multiplicação os seus bits indicam as coordenadas da Tabela L, como abaixo:

- Os 4 bits mais significativos representam a linha desta tabela
- Os 4 bits menos significativos representam a coluna desta tabela

Etapa 4 – MixColumns

Multiplicação de Galois

Exemplo: $\boldsymbol{b_1} = (r_1 * 2)\ xor\ (r_2 * 3)\ xor\ (r_3 * 1)\ xor\ (r_4 * 1)$

Sendo $r_1 = 0x6\mathrm{B}$, obtém-se $\boldsymbol{0x54}$

O mesmo se faz para o segundo termo $(0x02)$, onde se obtém $\boldsymbol{0x19}$

Em seguida, **somam-se os dois valores**. Neste caso:
$0x54 + 0x19 = 0x6\mathrm{D}$

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	00	00	19	01	32	02	1a	c6	4b	c7	1b	68	33	ee	df	03
1	64	04	e0	0e	34	8d	81	ef	4c	71	08	c8	f8	69	1c	c1
2	7d	c2	1d	b5	f9	b9	27	6a	4d	e4	a6	72	9a	c9	09	78
3	65	2f	8a	05	21	0f	e1	24	12	f0	82	45	35	93	da	8e
4	96	8f	db	bd	36	d0	ce	94	13	5c	d2	f1	40	46	83	38
5	66	dd	fd	30	bf	06	8b	62	b3	25	e2	98	22	88	91	10
6	7e	6e	48	c3	a3	b6	1e	42	3a	6b	28	54	fa	85	3d	ba
7	2b	79	0a	15	9b	9f	5e	ca	4e	d4	ac	e5	f3	73	a7	57
8	af	58	a8	50	f4	ea	d6	74	4f	ae	e9	d5	e7	e6	ad	e8
9	2c	d7	75	7a	eb	16	0b	f5	59	cb	5f	b0	9c	a9	51	a0
A	7f	0c	f6	6f	17	c4	49	ec	d8	43	1f	2d	a4	76	7b	b7
B	cc	bb	3e	5a	fb	60	b1	86	3b	52	a1	6c	aa	55	29	9d
C	97	b2	87	90	61	be	dc	fc	bc	95	cf	cd	37	3f	5b	d1
D	53	39	84	3c	41	a2	6d	47	14	2a	9e	5d	56	f2	d3	ab
E	44	11	92	d9	23	20	2e	89	b4	7c	b8	26	77	99	e3	a5
F	67	4a	ed	de	c5	31	fe	18	0d	63	8c	80	c0	f7	70	07

Tabela L

Observação: se o resultado da soma ultrapassar $0x\mathrm{FF}$, faz-se ajuste, subtraindo o valor de $0x\mathrm{FF}$: $resultado$ - $0x\mathrm{FF}$

Etapa 4 – MixColumns

Multiplicação de Galois

- O valor resultante do cálculo anterior permite obter o valor a partir de uma outra tabela (tabela E)
  - Os 4 bits mais significativos representam a linha desta tabela
  - Os 4 bits menos significativos representam a coluna desta tabela

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	01	03	05	0f	11	33	55	ff	1a	2e	72	96	a1	f8	13	35
1	5f	e1	38	48	d8	73	95	a4	f7	02	06	0a	1e	22	66	aa
2	e5	34	5c	e4	37	59	eb	26	6a	be	d9	70	90	ab	e6	31
3	53	f5	04	0c	14	3c	44	cc	4f	d1	68	b8	d3	6e	b2	cd
4	4c	d4	67	a9	e0	3b	4d	d7	62	a6	f1	08	18	28	78	88
5	83	9e	b9	d0	6b	bd	dc	7f	81	98	b3	ce	49	db	76	9a
6	b5	c4	57	f9	10	30	50	f0	0b	1d	27	69	bb	d6	61	a3
7	fe	19	2b	7d	87	92	ad	ec	2f	71	93	ae	e9	20	60	a0
8	fb	16	3a	4e	d2	6d	b7	c2	5d	e7	32	56	fa	15	3f	41
9	c3	5e	e2	3d	47	c9	40	c0	5b	ed	2c	74	9c	bf	da	75
A	9f	ba	d5	64	ac	ef	2a	7e	82	9d	bc	df	7a	8e	89	80
B	9b	b6	c1	58	e8	23	65	af	ea	25	6f	b1	c8	43	c5	54
C	fc	1f	21	63	a5	f4	07	09	1b	2d	77	99	b0	cb	46	ca
D	45	cf	4a	de	79	8b	86	91	a8	e3	3e	42	c6	51	f3	0e
E	12	36	5a	ee	29	7b	8d	8c	8f	8a	85	94	a7	f2	0d	17
F	39	4b	dd	7c	84	97	a2	fd	1c	24	6c	b4	c7	52	f6	01

Tabela E

Exemplo: para o valor 0x6D, mapeia-se: 0xD6. **Este é o valor da multiplicação no campo de *Galois*.**

Etapa 4 – MixColumns

Multiplicação de Galois

$$b_1 = (r_1 * 2)\ xor\ (r_2 * 3)\ xor\ (r_3 * 1)\ xor\ (r_4 * 1)$$

$$r_1 = 0x6\mathrm{B}$$

$$(0x6\mathrm{B} * 0x02) \xrightarrow{\text{Substituição tabela L}} (0x54 + 0x19)$$

$$(0x54 + 0x19) \longrightarrow 0x6\mathrm{D}$$

Eventual ajuste

$$0x\mathrm{D}6 \xleftarrow{\text{Substituição tabela E}} 0x6\mathrm{D}$$

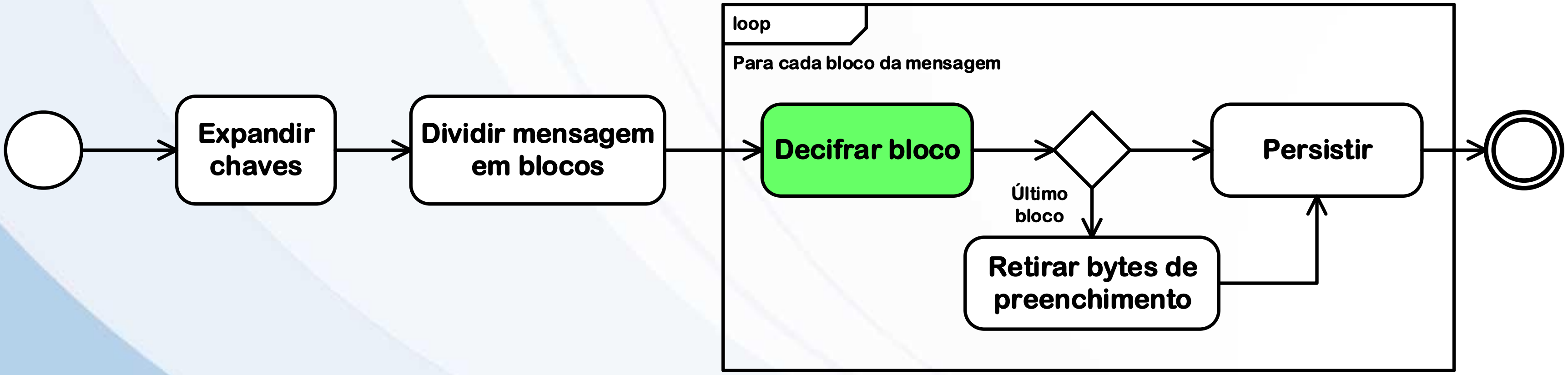
$$b_1 = (D6)\ xor\ (r_2 * 3)\ xor\ (r_3 * 1)\ xor\ (r_4 * 1)$$

Valor de b_1 $\longrightarrow$ $\begin{bmatrix} 0x5f & b_5 & b_9 & b_{13} \\ b_2 & b_6 & b_{10} & b_{14} \\ b_3 & b_7 & b_{11} & b_{15} \\ b_4 & b_8 & b_{12} & b_{16} \end{bmatrix}$

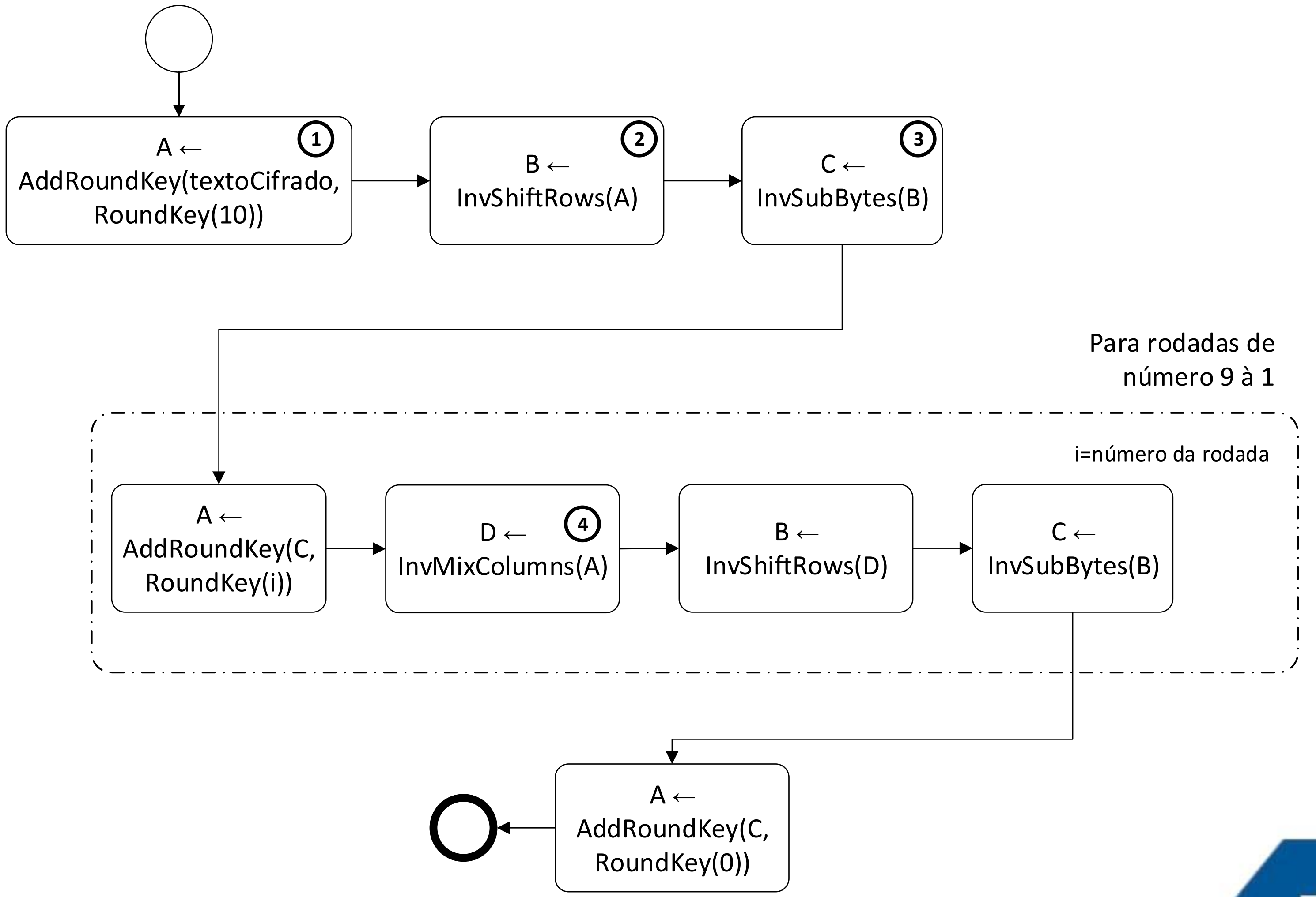
Etapa 5 - AddRoundKey

- Nesta etapa, o resultado da etapa 4 (*mixColumns*) é combinado através do operador XOR com a RoundKey da rodada corrente.

Decifragem de um bloco de 128 bits



Decifragem


A ← AddRoundKey(textoCifrado, RoundKey(10))
1
B ← InvShiftRows(A)
2
C ← InvSubBytes(B)
3
Para rodadas de número 9 à 1
i=número da rodada
A ← AddRoundKey(C, RoundKey(i))
D ← InvMixColumns(A)
4
B ← InvShiftRows(D)
C ← InvSubBytes(B)
A ← AddRoundKey(C, RoundKey(0))

Etapa 2 – Inverter Shift Rows

- Uma matriz de estado é criada partindo do resultado da etapa 1 e deslocando os bytes para a direita, da seguinte forma:

$$\begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,0} & b_{1,1} & b_{1,2} & b_{1,3} \\ b_{2,0} & b_{2,1} & b_{2,2} & b_{2,3} \\ b_{3,0} & b_{3,1} & b_{3,2} & b_{3,3} \end{bmatrix} \Rightarrow \begin{bmatrix} b_{0,0} & b_{0,1} & b_{0,2} & b_{0,3} \\ b_{1,3} & b_{1,0} & b_{1,1} & b_{1,2} \\ b_{2,2} & b_{2,3} & b_{2,0} & b_{2,1} \\ b_{3,1} & b_{3,2} & b_{3,3} & b_{3,0} \end{bmatrix}$$

- Exemplo:

$$\begin{pmatrix} \text{0xe9} & \text{0xcb} & \text{0x3d} & \text{0xaf} \\ \text{0x31} & \text{0x32} & \text{0x2e} & \text{0x09} \\ \text{0x7d} & \text{0x2c} & \text{0x89} & \text{0x07} \\ \text{0xb5} & \text{0x72} & \text{0x5f} & \text{0x94} \end{pmatrix} \Rightarrow \begin{pmatrix} \text{0xe9} & \text{0xcb} & \text{0x3d} & \text{0xaf} \\ \text{0x09} & \text{0x31} & \text{0x32} & \text{0x2e} \\ \text{0x89} & \text{0x07} & \text{0x7d} & \text{0x2c} \\ \text{0x72} & \text{0x5f} & \text{0x94} & \text{0xb5} \end{pmatrix}$$

Etapa 3 – Inverter Sub Bytes

- Cada byte da matriz de estado é substituído utilizando a *S-Box inversa*.

$$\begin{pmatrix} \text{0xe9} & \text{0xcb} & \text{0x3d} & \text{0xaf} \\ \text{0x09} & \text{0x31} & \text{0x32} & \text{0x2e} \\ \text{0x89} & \text{0x07} & \text{0x7d} & \text{0x2c} \\ \text{0x72} & \text{0x5f} & \text{0x94} & \text{0xb5} \end{pmatrix}$$

$$\Downarrow$$

$$\begin{pmatrix} \text{0xeb} & \text{0x59} & \text{0x8b} & \text{0x1b} \\ \text{0x40} & \text{0x2e} & \text{0xa1} & \text{0xc3} \\ \text{0xf2} & \text{0x38} & \text{0x13} & \text{0x42} \\ \text{0x1e} & \text{0x84} & \text{0xe7} & \text{0xd2} \end{pmatrix}$$

	0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0	52	09	6a	d5	30	36	a5	38	bf	40	a3	9e	81	f3	d7	fb
1	7c	e3	39	82	9b	2f	ff	87	34	8e	43	44	c4	de	e9	cb
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	6b
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	6e
A	47	f1	1a	71	1d	29	c5	89	6f	b7	62	0e	aa	18	be	1b
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	5f
D	60	51	7f	a9	19	b5	4a	0d	2d	e5	7a	9f	93	c9	9c	ef
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	7d

Tabela S-Box inversa

Etapa 4 – Inverter MixColumns

- Mesmo procedimento da cifragem, porém considerar a seguinte matriz de multiplicação

$$\begin{pmatrix} \text{0x0e} & \text{0x0b} & \text{0x0d} & \text{0x09} \\ \text{0x09} & \text{0x0e} & \text{0x0b} & \text{0x0d} \\ \text{0x0d} & \text{0x09} & \text{0x0e} & \text{0x0b} \\ \text{0x0b} & \text{0x0d} & \text{0x09} & \text{0x0e} \end{pmatrix}$$