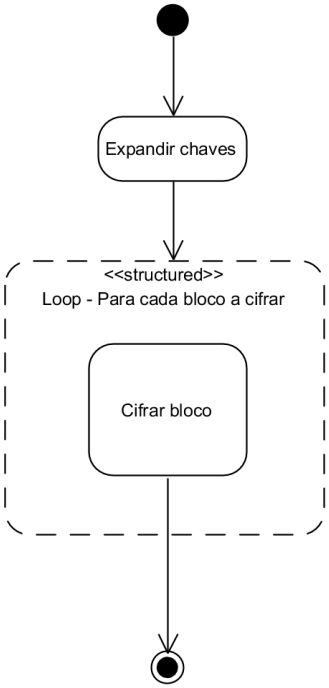
Cifragem utilizando o algoritmo AES

FURB
UNIVERSIDADE DE BLUMENAU

Definições

- Utiliza chave de 128, 192 ou 256 bits.
  - Vamos ver o algoritmo para chaves de 128 bits
- Bloco de 128 bits
- O algoritmo pode ser dividido em duas etapas:
  - Expansão da chave
  - Cifragem de bloco



Matriz de estado

- O algoritmo utiliza matrizes 4 x 4 para efetuar o processamento.
- Esta matriz é chamada de **matriz de estado**:

$$\begin{bmatrix} byte_0 & byte_4 & byte_8 & byte_{12} \\ byte_1 & byte_5 & byte_9 & byte_{13} \\ byte_2 & byte_6 & byte_{10} & byte_{14} \\ byte_3 & byte_7 & byte_{11} & byte_{15} \end{bmatrix}$$

- 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

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

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

$$\downarrow$$

$$\mathrm{w}_0 \quad \mathrm{w}_1 \quad \mathrm{w}_2 \quad \mathrm{w}_3$$

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 seria:

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

Ou, em bytes:

$$\begin{bmatrix} 0x41 & 0x45 & 0x49 & 0x4d \\ 0x42 & 0x46 & 0x4a & 0x4e \\ 0x43 & 0x47 & 0x4b & 0x4f \\ 0x44 & 0x48 & 0x4c & 0x50 \end{bmatrix}$$

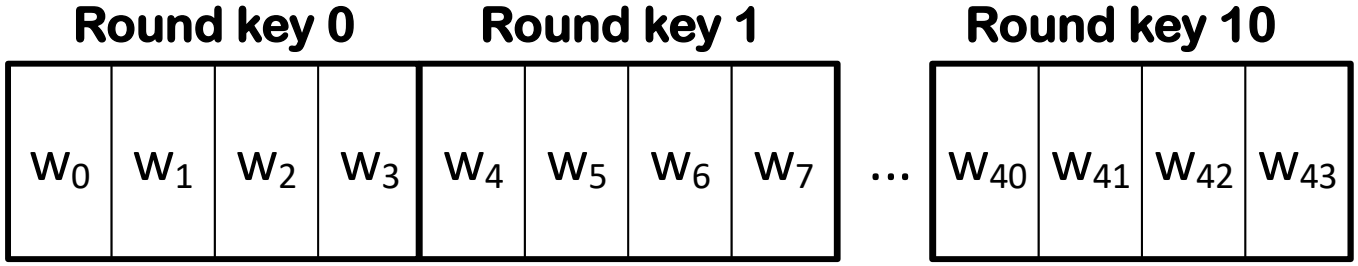
Expansão de chave

- 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)

  - As palavras w_0, w_1, w_2, w_3 constituem a round key 0 (chave original)
  - As palavras w_4, w_5, w_6, w_7 constituem a round key 1
  - As palavras w_8, w_9, w_{10}, w_{11} constituem a round key 2
  - e assim por diante.

Geração de chaves (round key)

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



$$w_5 \leftarrow w_1 \oplus w_4 \qquad w_{41} \leftarrow w_{37} \oplus w_{40}$$
$$w_6 \leftarrow w_2 \oplus w_5 \qquad w_{42} \leftarrow w_{38} \oplus w_{41}$$
$$w_7 \leftarrow w_3 \oplus w_6 \qquad w_{43} \leftarrow w_{39} \oplus w_{42}$$

$\oplus$ Operação XOR

A primeira palavra das Roundkeys 1..10

A primeira palavra das roundKeys de número 1 à 10 requer realizar as seguintes operações:

1) Fazer uma cópia da última palavra da roundkey anterior
2) Rotacionar os bytes desta palavra (RotWord)
3) Substituir os bytes da palavra (SubWord)
4) Gerar uma nova palavra, denominada de RoundConstant
5) Fazer um xor de (3) com (4).
6) Fazer um xor da primeira palavra da roundkey anterior com o resultado de (5)

Geração da primeira palavra da *roundKey*

2 – rotacionar os bytes

- Depois de copiada a última palavra da *round key* imediatamente anterior, deve-se rotacionar os bytes desta palavra

$$\begin{matrix} byte_0 \\ byte_1 \\ byte_2 \\ byte_3 \end{matrix} \Rightarrow \begin{matrix} byte_1 \\ byte_2 \\ byte_3 \\ byte_0 \end{matrix}$$

- Esta etapa também é conhecida como *Rot-word*

Geração da primeira palavra da roundKey

2 – rotacionar os bytes (exemplo)

- 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

- Depois de copiada a palavra w3, rotaciona-se os bytes

0x4E
0x4F
0x50
0x4D

Geração da primeira palavra da *roundKey*

3 – substituição de palavra

- A partir da palavra obtida com o passo (2), deve-se utilizar a tabela S-Box para substituir os bytes da palavra.

	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

- Para cada byte da palavra, considerar que:
  - Os 4 bits mais significativos representam a linha da S-Box
  - Os 4 bits menos significativos representam a coluna da S-Box

Geração da primeira palavra da *roundKey*

3 – substituição de palavra

Exemplo. Supor que o resultado do passo 2 seja:

0x4E
0x4F
0x50
0x4D

O byte “0x4E” será substituído pelo byte que estiver na linha 4/Coluna E

Ou seja: byte: 0x2F

Logo, o novo valor da palavra será:

0x2F
0x84
0x53
0xE3

	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

Geração da primeira palavra da *roundKey*

4 – Geração da RoundConstant

- A **RoundConstant** é uma palavra gerada
  - Seus bytes 1, 2 e 3 são 0
  - 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

- Exemplo: durante o processamento da roundkey 6, a RoundConstant será:

$$0x20$$
$$0$$
$$0$$
$$0$$

Geração da primeira palavra da *roundKey*

5 – XOR com a RoundConstant

- Nesta etapa, é feita uma operação XOR da palavra da etapa 3 (isto é, após aplicada a substituição com a S-Box) com a palavra da etapa 4
- Exemplo:
  - Considerar que o resultado da etapa 3 seja: $\begin{matrix} 0x2F \\ 0x84 \\ 0x53 \\ 0xE3 \end{matrix}$
  - Considerar que o resultado da etapa 4 seja: $\begin{matrix} 0x01 \\ 0 \\ 0 \\ 0 \end{matrix}$

$$\begin{matrix} 0x2F \\ 0x84 \\ 0x53 \\ 0xE3 \end{matrix} \oplus \begin{matrix} 0x01 \\ 0 \\ 0 \\ 0 \end{matrix} = \begin{matrix} 0x2E \\ 0x84 \\ 0x53 \\ 0xE3 \end{matrix}$$

Geração da primeira palavra da *roundKey*

6 – Obtenção da primeira palavra

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

    $0x41$
    $0x42$
    $0x43$
    $0x44$

  - Considerar que a palavra obtida na etapa 5 seja:

    $0x2E$
    $0x84$
    $0x53$
    $0xE3$

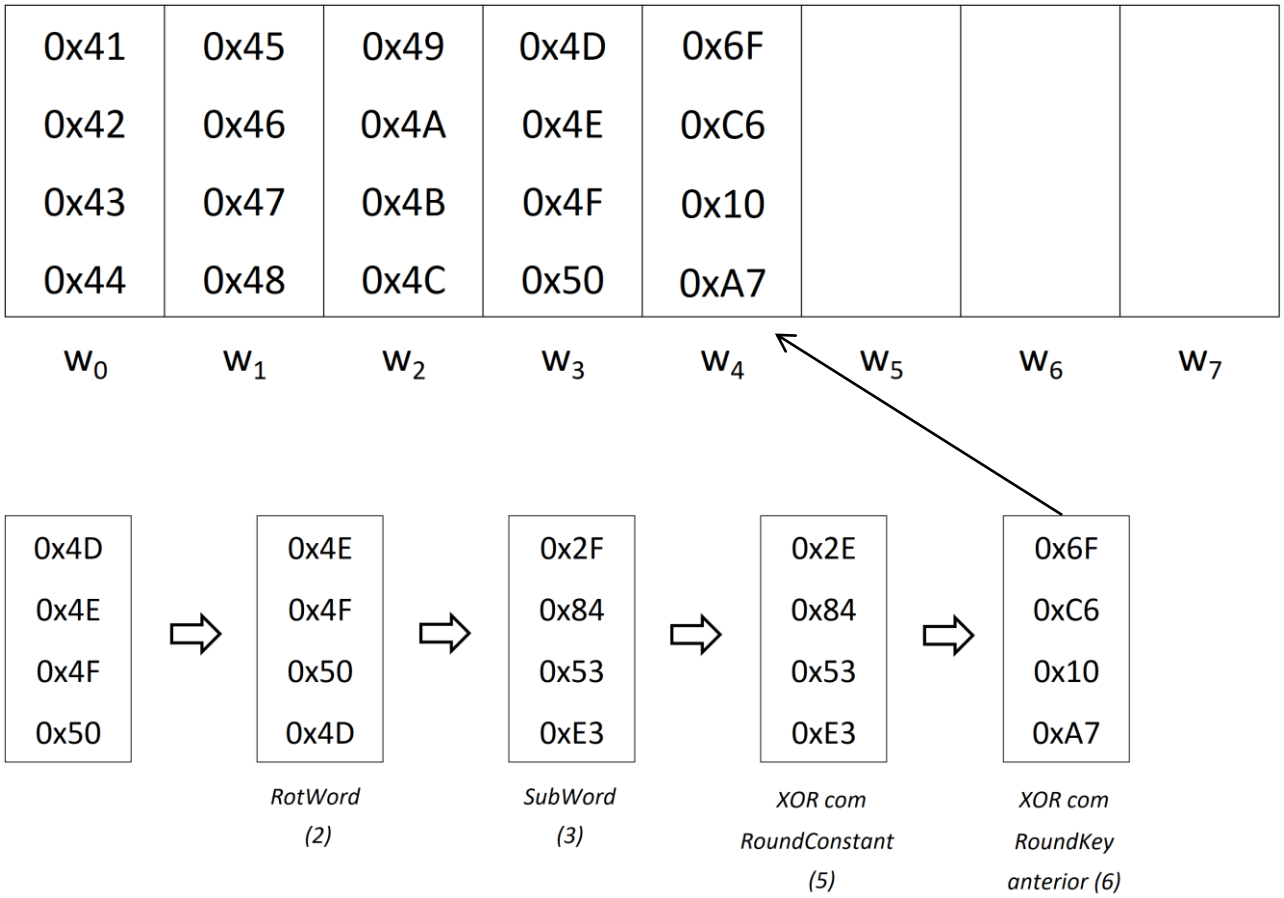
  - Sendo assim:

    $$\begin{matrix} 0x41 \\ 0x42 \\ 0x43 \\ 0x44 \end{matrix} \oplus \begin{matrix} 0x2E \\ 0x84 \\ 0x53 \\ 0xE3 \end{matrix}$$

  - Portanto:

    $0x6F$
    $0xC6$
    $0x10$
    $0xA7$

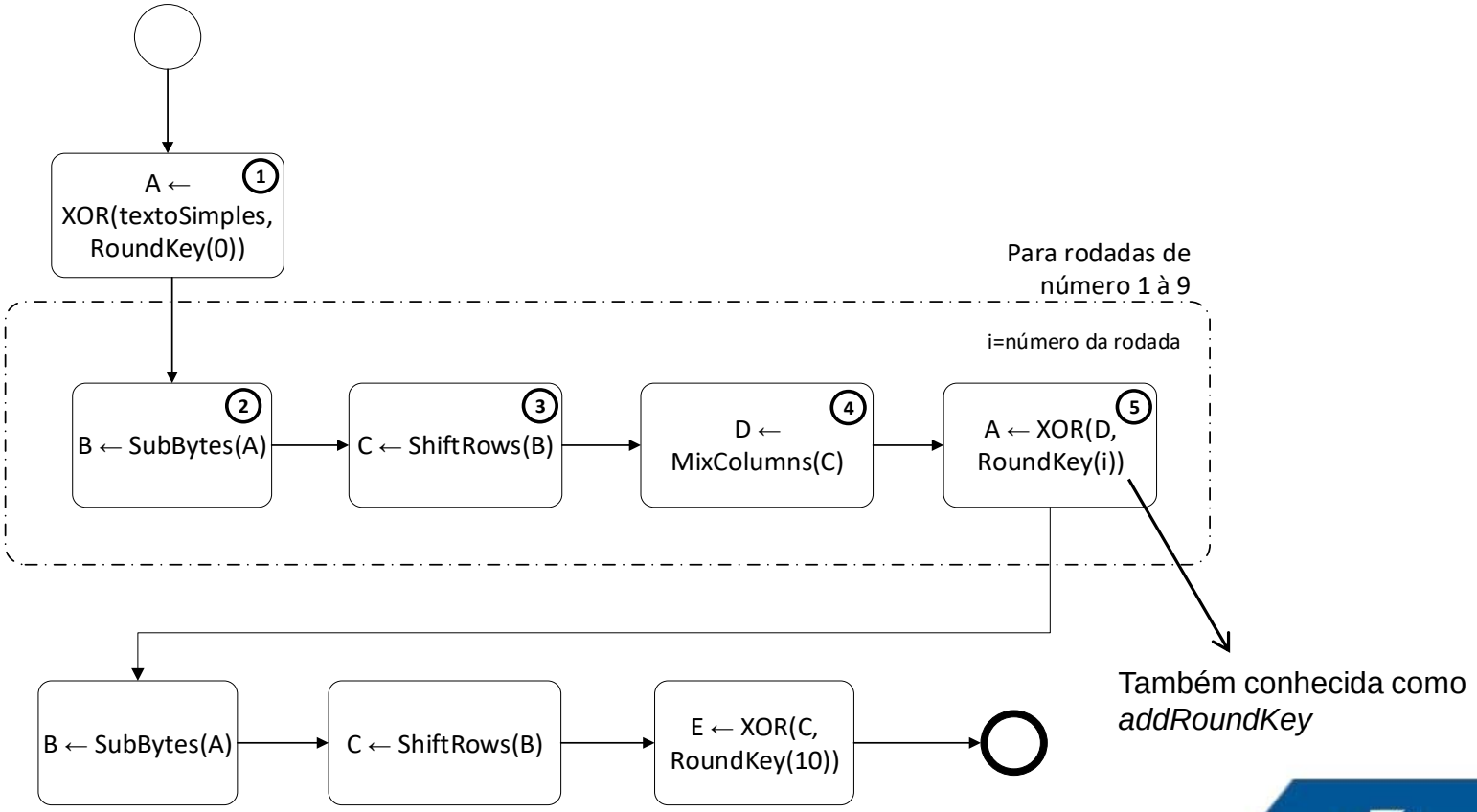
Exemplo


0x41 0x45 0x49 0x4D 0x6F
0x42 0x46 0x4A 0x4E 0xC6
0x43 0x47 0x4B 0x4F 0x10
0x44 0x48 0x4C 0x50 0xA7
w0 w1 w2 w3 w4 w5 w6 w7
0x4D 0x4E 0x4F 0x50
0x4E 0x4F 0x50 0x4D
RotWord (2)
0x2F 0x84 0x53 0xE3
SubWord (3)
0x2E 0x84 0x53 0xE3
XOR com RoundConstant (5)
0x6F 0xC6 0x10 0xA7
XOR com RoundKey anterior (6)

Cifragem de um bloco de 128 bits

Visão geral

- A criptografia consiste na execução de 10 rodadas



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

- 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:

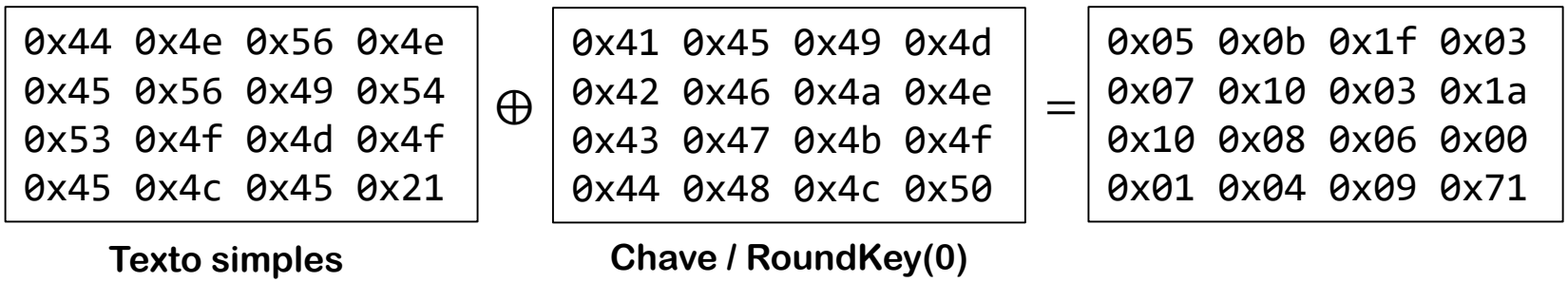
0x44 0x45 0x53 0x45 0x4e 0x56 0x4f 0x4c 0x56 0x49 0x4d 0x45 0x4e 0x54 0x4f 0x21

$$\begin{bmatrix} 0\text{x}44 & 0\text{x}4e & 0\text{x}56 & 0\text{x}4e \\ 0\text{x}45 & 0\text{x}56 & 0\text{x}49 & 0\text{x}54 \\ 0\text{x}53 & 0\text{x}4f & 0\text{x}4d & 0\text{x}4f \\ 0\text{x}45 & 0\text{x}4c & 0\text{x}45 & 0\text{x}21 \end{bmatrix}$$

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

- Uma matriz de estado é construída a partir da aplicação do operador XOR dos elementos da matriz de estado que contém o texto simples e da roundKey inicial (roundKey 0).
  - *RoundKey 0 contém a chave original*

- Exemplo:



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:

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

Matriz resultante da 4ª etapa

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

Matriz de multiplicação

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

Matriz resultante da 3ª etapa (ShiftRows)

- O valor de b_1 é calculado utilizando-se a primeira palavra (horizontal) da matriz de multiplicação, através da seguinte fórmula:

$$b_1 = (r_1 * 2)\ xor\ (r_2 * 3)\ xor\ (r_3 * 1)\ xor\ (r_4 * 1)$$
$$b_2 = (r_1 * 1)\ xor\ (r_2 * 2)\ xor\ (r_3 * 3)\ xor\ (r_4 * 1)$$
$$b_3 = (r_1 * 1)\ xor\ (r_2 * 1)\ xor\ (r_3 * 2)\ xor\ (r_4 * 3)$$
$$b_4 = (r_1 * 3)\ xor\ (r_2 * 1)\ xor\ (r_3 * 1)\ xor\ (r_4 * 2)$$

$$b_5 = (r_5 * 2)\ xor\ (r_6 * 3)\ xor\ (r_7 * 1)\ xor\ (r_8 * 1)$$
$$b_6 = (r_5 * 1)\ xor\ (r_6 * 2)\ xor\ (r_7 * 3)\ xor\ (r_8 * 1)$$

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

	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

Cada termo da multiplicação indica uma coordenada numa tabela denominada de *tabela L*:

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

Etapa 4 - MixColumns

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

- Sendo r_1 = 6B, obtém-se: 0x54
- O mesmo se faz para o segundo termo (02), onde se obtém: 0x19
- Em seguida, somam-se os dois valores. Neste caso: 0x54 + 0x19 = 0x6D

	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

Observação: se o resultado da soma ultrapassar 0xFF, faz-se ajuste:

$resultado\ - 0xFF$

Etapa 4 - MixColumns

- 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

- Existem duas exceções na *multiplicação de Galois* a se considerar:
  - 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

Etapa 5 - AddRoundKey

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