Implementação dos algoritmos DES e AES

Carlos Guilherme
Larissa Hortêncio
Leandro Nogueira

Sumário

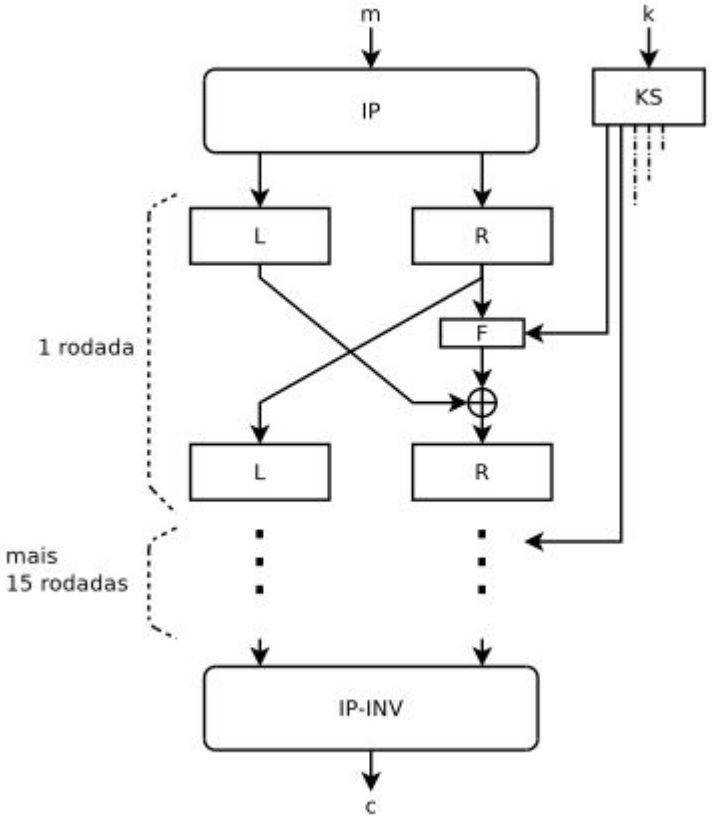
- Introdução
- DES
- AES

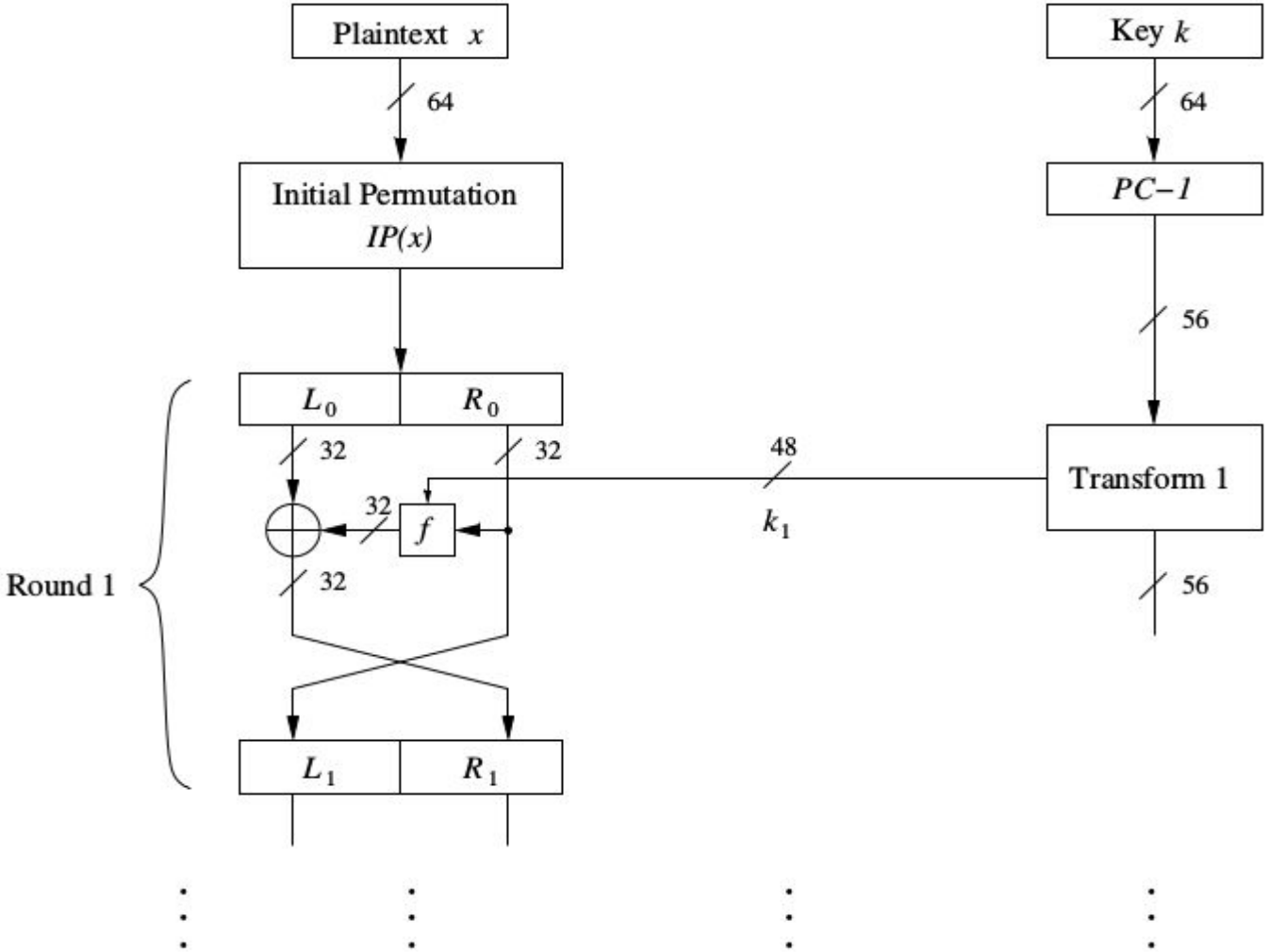
Introdução

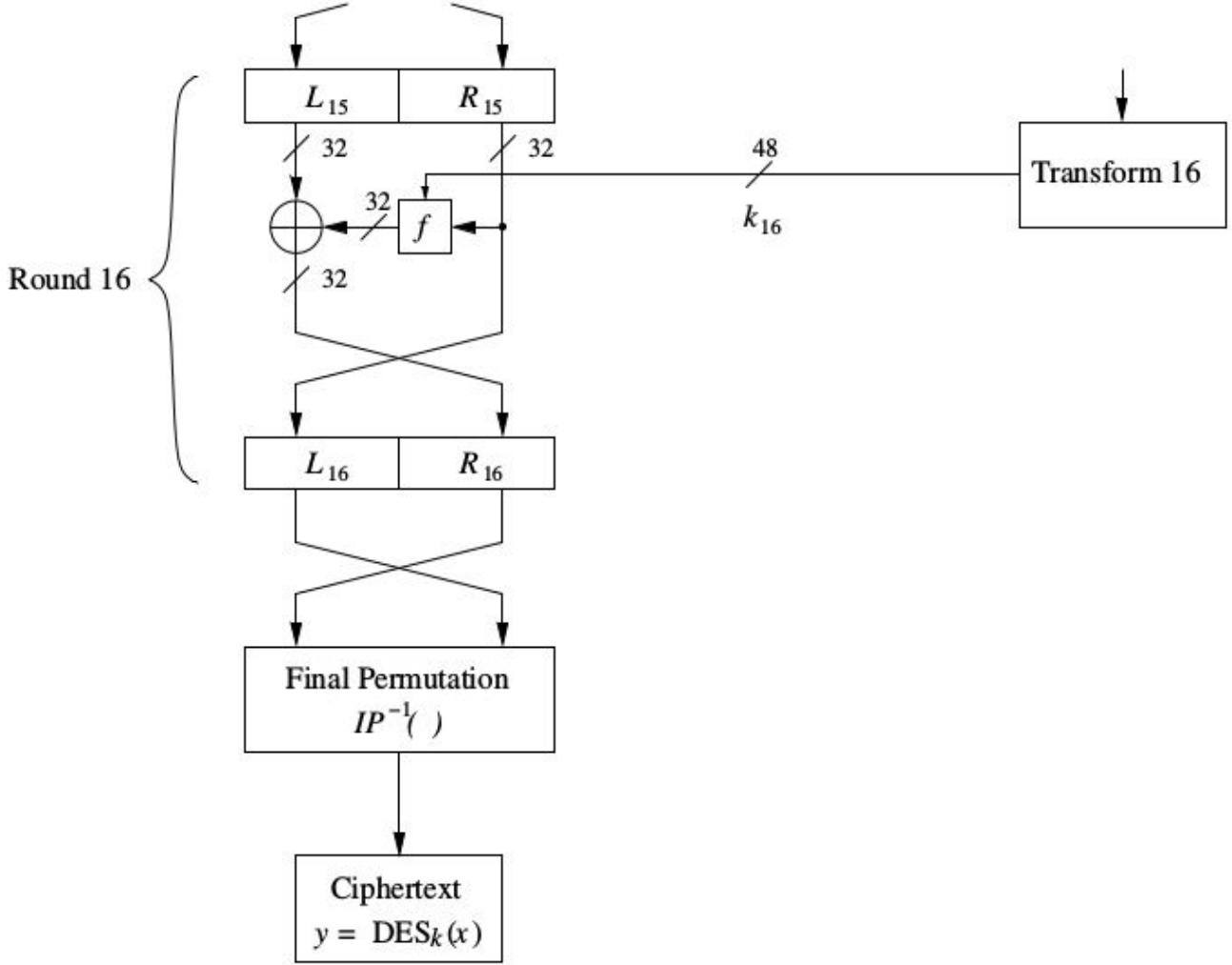
Foi utilizada a linguagem Python 2.7.12 para implementação dos algoritmos, e para os testes foram utilizados arquivos de texto e imagens.

DES

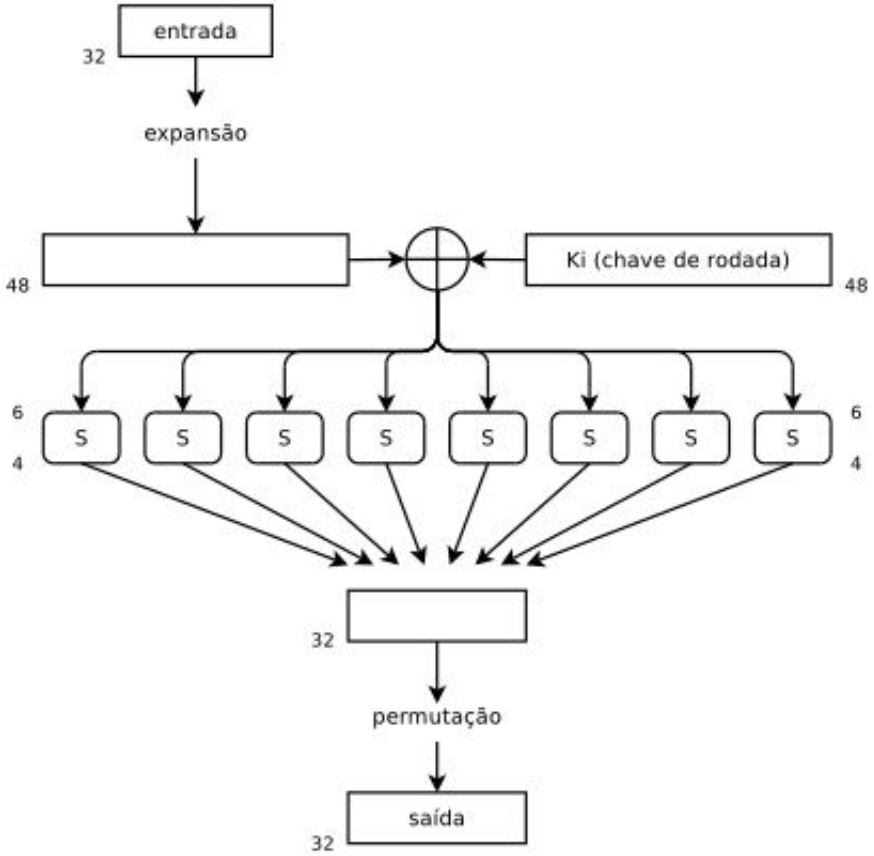
- Até a introdução do Advanced Encription Standard (AES), em 2001, o Data Encription Standard (DES) era o esquema de encriptação mais utilizado.
- O DES é uma rede de Feistel com chave de 56 bits e mensagens de 64 bits, usando 16 rodadas e 8 S-boxes.
- Antes de aplicar a entrada na rede de Feistel, o DES realiza uma permutação inicial na entrada. Esta permutação é revertida na saída da rede.




Plaintext x
64
Initial Permutation
IP(x)
L0
R0
32
32
32
f
32
Round 1
L1
R1
Key k
64
PC–1
56
48
k1
Transform 1
56

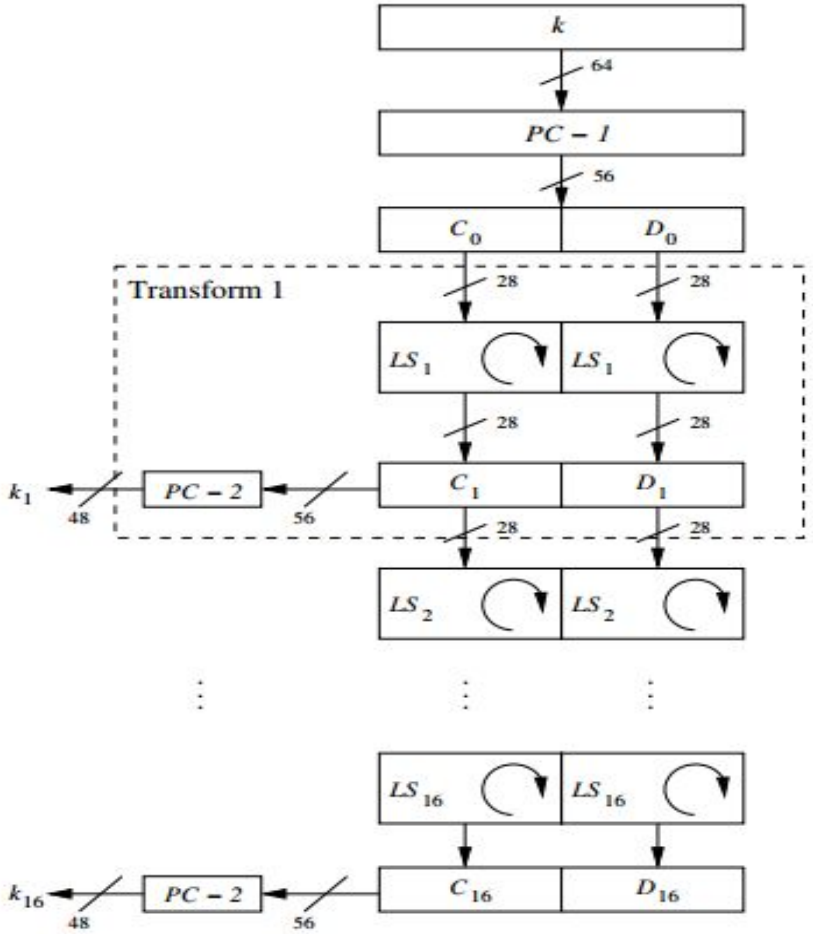

L_{15}
R_{15}
32
32
48
Transform 16
32
f
k_{16}
Round 16
32
L_{16}
R_{16}
Final Permutation
$IP^{-1}(\ )$
Ciphertext
$y = \mathrm{DES}_k(x)$

Função F


entrada
32
expansão
48
Ki (chave de rodada)
48
6
4
S
S
S
S
S
S
S
S
6
4
32
permutação
saída
32

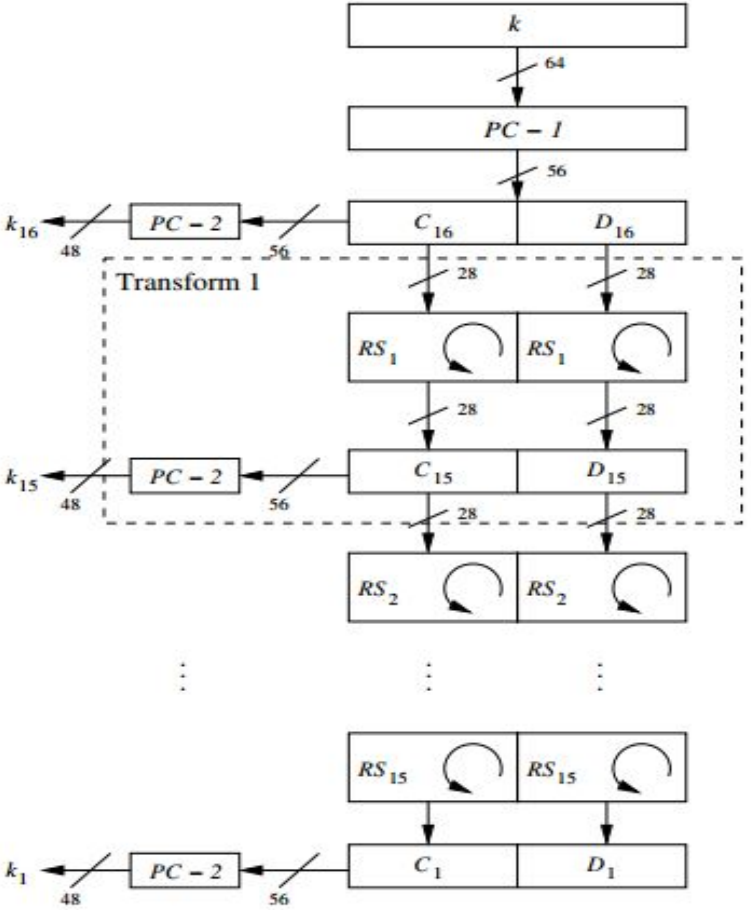
Geração Subchaves - DES

- Criptografia:
  - Recebe uma chave K de 64 bits;
  - O processo de geração das chaves consiste na realização de permutações e rotações nos blocos das chaves.



Geração Subchaves - DES

- Descriptografia:
  - Recebe uma chave K de 64 bits;
  - Realiza o processo inverso na geração das chaves



Entradas para o programa

- Nome do arquivo de entrada
- Nome do arquivo de saída
- Nome do arquivo com chave de Criptografia/Decriptografia"
- Digite o código:
  - 0 - Criptografar
  - 1 - Decriptografar

```
$ python DES.py imagem.jpg imagemCriptografada.jpg chaveDES.txt 0
```

```
$ python DES.py imagemCriptografada.jpg imagemDecriptografada.jpg chaveDES.txt 1
```

AES

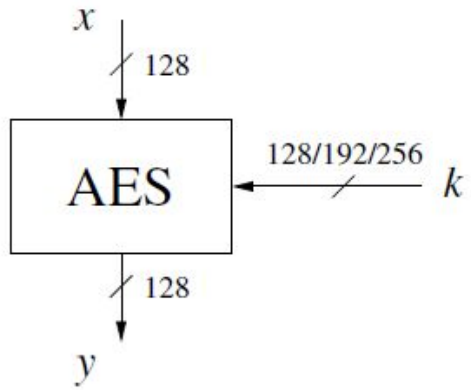
- Em 1997 o NIST anunciou um concurso para definir um novo Padrão Avançado de Criptografia (AES).

- Em 2001, foi declarado o algoritmo de cifra em bloco Rijndael como o AES.

- Combina características:
    - Segurança, desempenho, facilidade de implementação e flexibilidade.

AES

- Características:
  - Cifra de bloco de 128 bits.
  - Chave com tamanho de 128, 192 ou 256 bits.



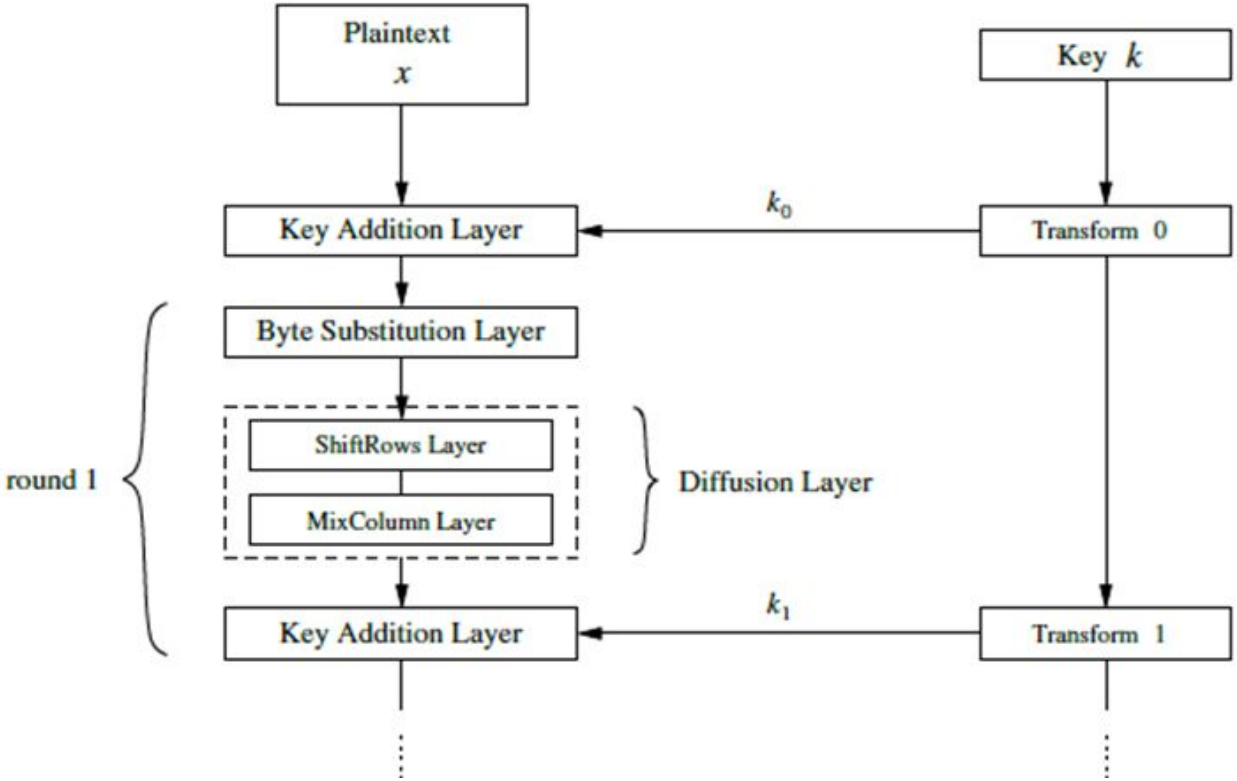
tamanho da chave	no. de rodadas (n_r)
128 bits	10
192 bits	12
256 bits	14

AES

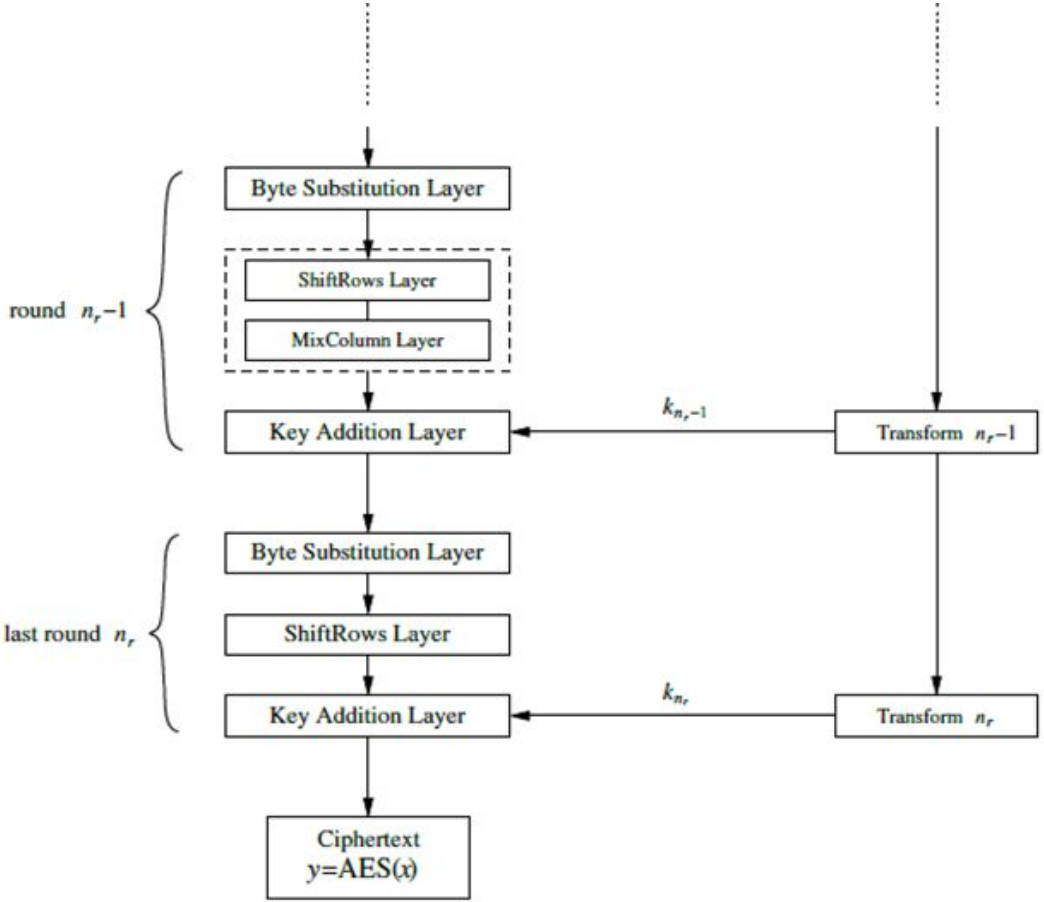
- Para criptografar o AES consiste em quatro estágios:
    - Adição de Chave.
    - Substituição de bytes (S-box).
    - Camada de difusão:
        - Deslocamento de linhas.
        - Mix de colunas.

- Os 128 bits, ou seja, 16 bytes são representado por uma matriz 4X4.

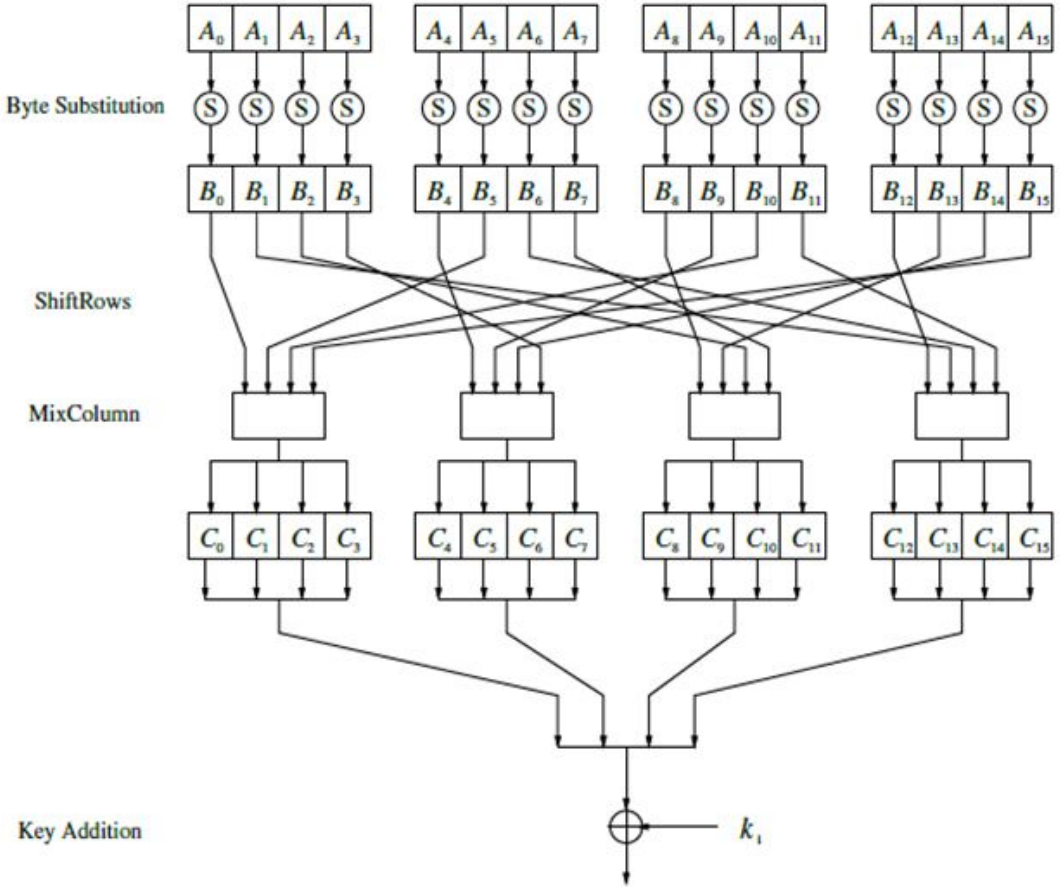
AES


Plaintext
x
Key k
k0
Key Addition Layer
Transform 0
Byte Substitution Layer
round 1
ShiftRows Layer
MixColumn Layer
Diffusion Layer
k1
Key Addition Layer
Transform 1

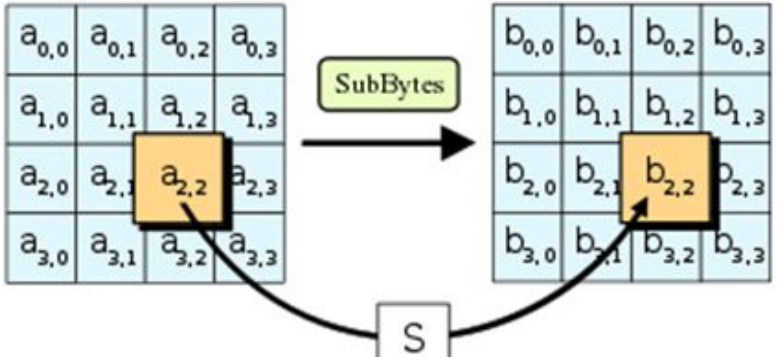
AES


Byte Substitution Layer
ShiftRows Layer
MixColumn Layer
round n_r-1
Key Addition Layer
k_{n_r-1}
Transform n_r-1
Byte Substitution Layer
ShiftRows Layer
last round n_r
Key Addition Layer
k_{n_r}
Transform n_r
Ciphertext
y=AES(x)

AES


Byte Substitution
ShiftRows
MixColumn
Key Addition
A_0 A_1 A_2 A_3 A_4 A_5 A_6 A_7 A_8 A_9 A_{10} A_{11} A_{12} A_{13} A_{14} A_{15}
S
B_0 B_1 B_2 B_3 B_4 B_5 B_6 B_7 B_8 B_9 B_{10} B_{11} B_{12} B_{13} B_{14} B_{15}
C_0 C_1 C_2 C_3 C_4 C_5 C_6 C_7 C_8 C_9 C_{10} C_{11} C_{12} C_{13} C_{14} C_{15}
k_1

Substituição de Bytes




```
def substituicaoBytes (lista):
    for x in xrange(0,16):
        lista[x]= string_to_bit_array(chr(Sbox[bit_to_int(lista[x][0:4])][bit_to_int(lista[x][4:8])]))
```

Deslocamento de linhas

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}

B_0	B_4	B_8	B_{12}
B_5	B_9	B_{13}	B_1
B_{10}	B_{14}	B_2	B_6
B_{15}	B_3	B_7	B_{11}

```
def deslocamentoLinhas (lista):
    lista[4],lista[5],lista[6],lista[7]= lista[5],lista[6],lista[7],lista[4]
    lista[8],lista[9],lista[10],lista[11]= lista[10],lista[11],lista[8],lista[9]
    lista[12],lista[13],lista[14],lista[15]= lista[15],lista[12],lista[13],lista[14]
```

Deslocamento de linhas

$$\begin{pmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{pmatrix} = \begin{pmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{pmatrix} \cdot \begin{pmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{pmatrix}$$

```
def mixColunas (lista):
    c= lista[:]

    c[0] = int_to_bit(g2[(bit_to_int(lista[0]))] ^ g3[(bit_to_int(lista[4]))] ^ bit_to_int(lista[8]) ^ bit_to_int(lista[12]))
    c[4] = int_to_bit(bit_to_int(lista[0]) ^ g2[(bit_to_int(lista[4]))] ^g3[(bit_to_int(lista[8]))] ^ bit_to_int(lista[12]))
    c[8] = int_to_bit(bit_to_int(lista[0]) ^ bit_to_int(lista[4]) ^ g2[(bit_to_int(lista[8]))] ^ g3[(bit_to_int(lista[12]))])
    c[12] = int_to_bit(g3[(bit_to_int(lista[0]))] ^ bit_to_int(lista[4]) ^ bit_to_int(lista[8]) ^ g2[(bit_to_int(lista[12]))])
```

Galois

- Toda aritmética é feita utilizando o campo de Galois.

```
g2 = [0x00,0x02,0x04,0x06,0x08,0x0a,0x0c,0x0e,0x10,0x12,0x14,0x16,0x18,0x1a,0x1c,0x1e,
      0x20,0x22,0x24,0x26,0x28,0x2a,0x2c,0x2e,0x30,0x32,0x34,0x36,0x38,0x3a,0x3c,0x3e,
      0x40,0x42,0x44,0x46,0x48,0x4a,0x4c,0x4e,0x50,0x52,0x54,0x56,0x58,0x5a,0x5c,0x5e,
      0x60,0x62,0x64,0x66,0x68,0x6a,0x6c,0x6e,0x70,0x72,0x74,0x76,0x78,0x7a,0x7c,0x7e,
      0x80,0x82,0x84,0x86,0x88,0x8a,0x8c,0x8e,0x90,0x92,0x94,0x96,0x98,0x9a,0x9c,0x9e,
      0xa0,0xa2,0xa4,0xa6,0xa8,0xaa,0xac,0xae,0xb0,0xb2,0xb4,0xb6,0xb8,0xba,0xbc,0xbe,
      0xc0,0xc2,0xc4,0xc6,0xc8,0xca,0xcc,0xce,0xd0,0xd2,0xd4,0xd6,0xd8,0xda,0xdc,0xde,
      0xe0,0xe2,0xe4,0xe6,0xe8,0xea,0xec,0xee,0xf0,0xf2,0xf4,0xf6,0xf8,0xfa,0xfc,0xfe,
      0x1b,0x19,0x1f,0x1d,0x13,0x11,0x17,0x15,0x0b,0x09,0x0f,0x0d,0x03,0x01,0x07,0x05,
      0x3b,0x39,0x3f,0x3d,0x33,0x31,0x37,0x35,0x2b,0x29,0x2f,0x2d,0x23,0x21,0x27,0x25,
      0x5b,0x59,0x5f,0x5d,0x53,0x51,0x57,0x55,0x4b,0x49,0x4f,0x4d,0x43,0x41,0x47,0x45,
      0x7b,0x79,0x7f,0x7d,0x73,0x71,0x77,0x75,0x6b,0x69,0x6f,0x6d,0x63,0x61,0x67,0x65,
      0x9b,0x99,0x9f,0x9d,0x93,0x91,0x97,0x95,0x8b,0x89,0x8f,0x8d,0x83,0x81,0x87,0x85,
      0xbb,0xb9,0xbf,0xbd,0xb3,0xb1,0xb7,0xb5,0xab,0xa9,0xaf,0xad,0xa3,0xa1,0xa7,0xa5,
      0xdb,0xd9,0xdf,0xdd,0xd3,0xd1,0xd7,0xd5,0xcb,0xc9,0xcf,0xcd,0xc3,0xc1,0xc7,0xc5,
      0xfb,0xf9,0xff,0xfd,0xf3,0xf1,0xf7,0xf5,0xeb,0xe9,0xef,0xed,0xe3,0xe1,0xe7,0xe5]
```

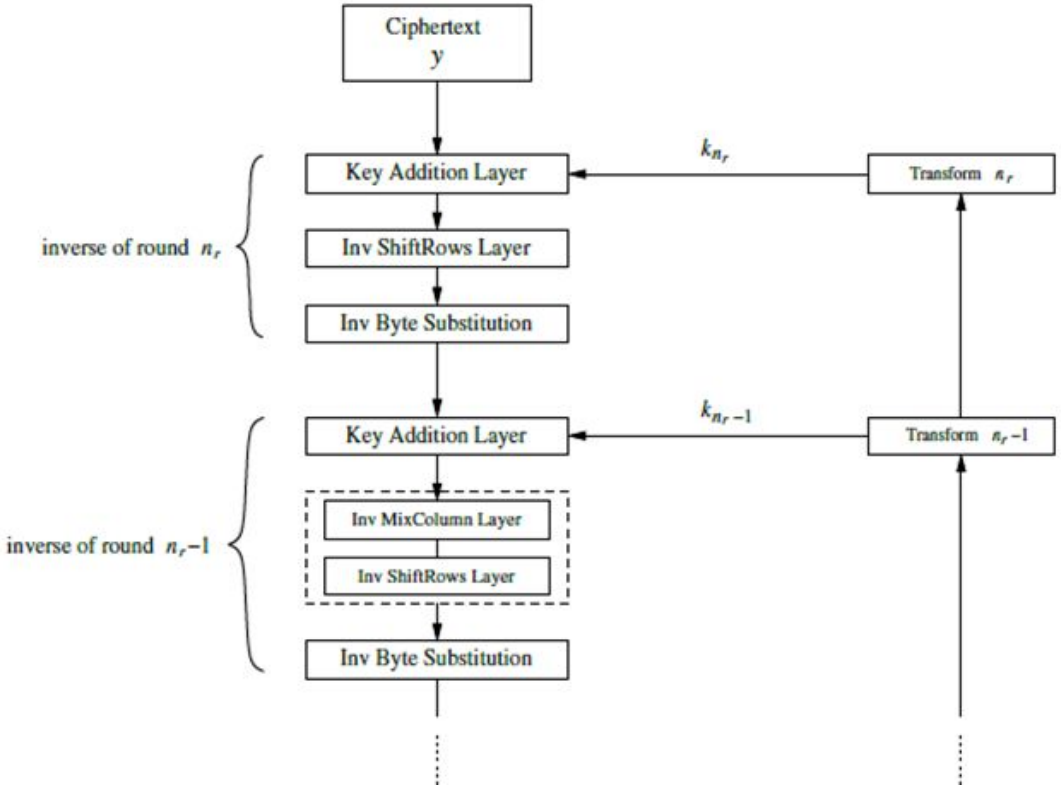
Adição de chave

- Entradas:
  - Matriz C com 16 bytes.
  - Subchave K.

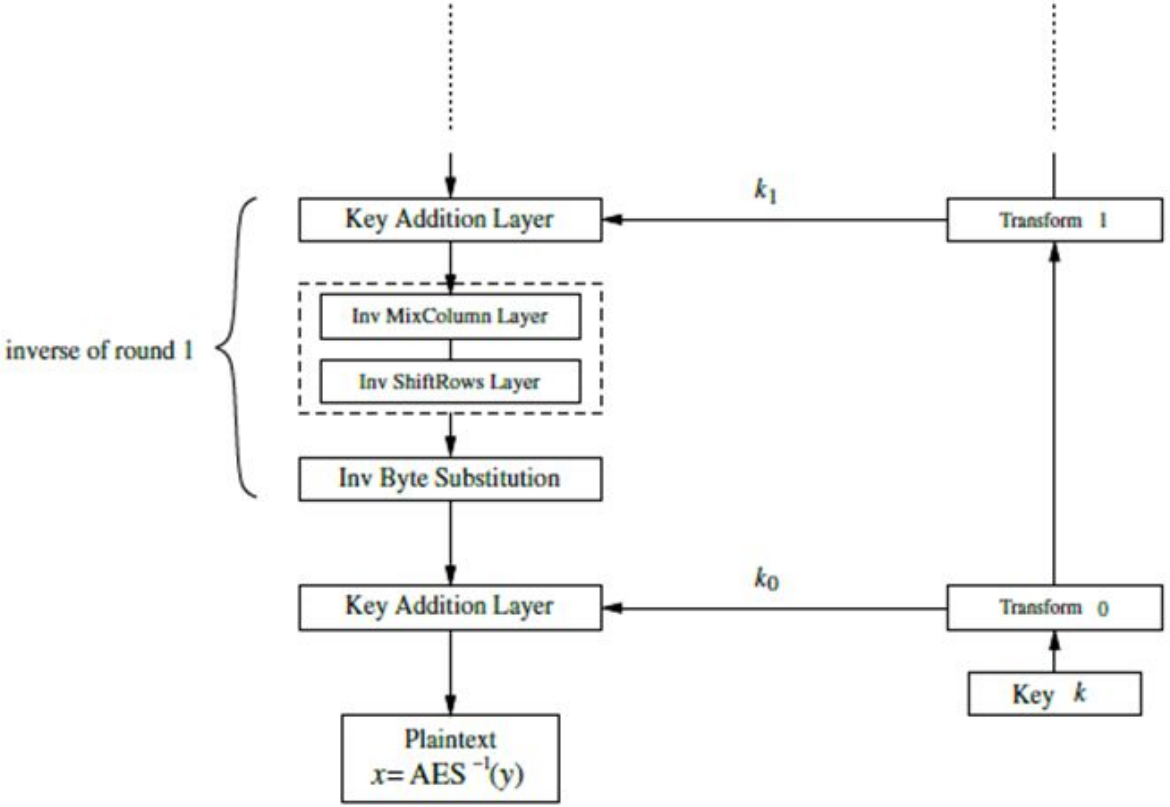
  Saída:
  - C xor K.

```
def adicaoChave (b,c):
    for x in xrange(0,16):
        b[x] =  xor(b[x],c[x])
    return b
```

AES


Ciphertext
y
Key Addition Layer
k_{n_r}
Transform n_r
inverse of round n_r
Inv ShiftRows Layer
Inv Byte Substitution
Key Addition Layer
k_{n_r-1}
Transform n_r-1
inverse of round n_r-1
Inv MixColumn Layer
Inv ShiftRows Layer
Inv Byte Substitution

AES


Key Addition Layer
k_1
Transform 1
inverse of round 1
Inv MixColumn Layer
Inv ShiftRows Layer
Inv Byte Substitution
Key Addition Layer
k_0
Transform 0
Key k
Plaintext
$x = \mathrm{AES}^{-1}(y)$

Geração de Subchave AES

- O processo de geração de chaves no AES utiliza os seguintes parâmetros para a expansão da chave:
  - **K** : chave criptografada secreta pode ter 128, 192, 256 bits.
  - **Nk** : No de colunas de K. Nk = 4 (AES-128), 6 (AES-192), 8 (AES-256). É calculado dividindo tamanho da chave por 32 bits.
  - **Nb** : No de colunas do State. Nb = 4, 6, 8. Similar a Nk.
  - **Nr** : No de rodadas, dado em função de Nb e Nk, Nr=6+max(Nb, Nk)+1

Geração de Subchave AES

- Pseudocódigo KeyExpansion:

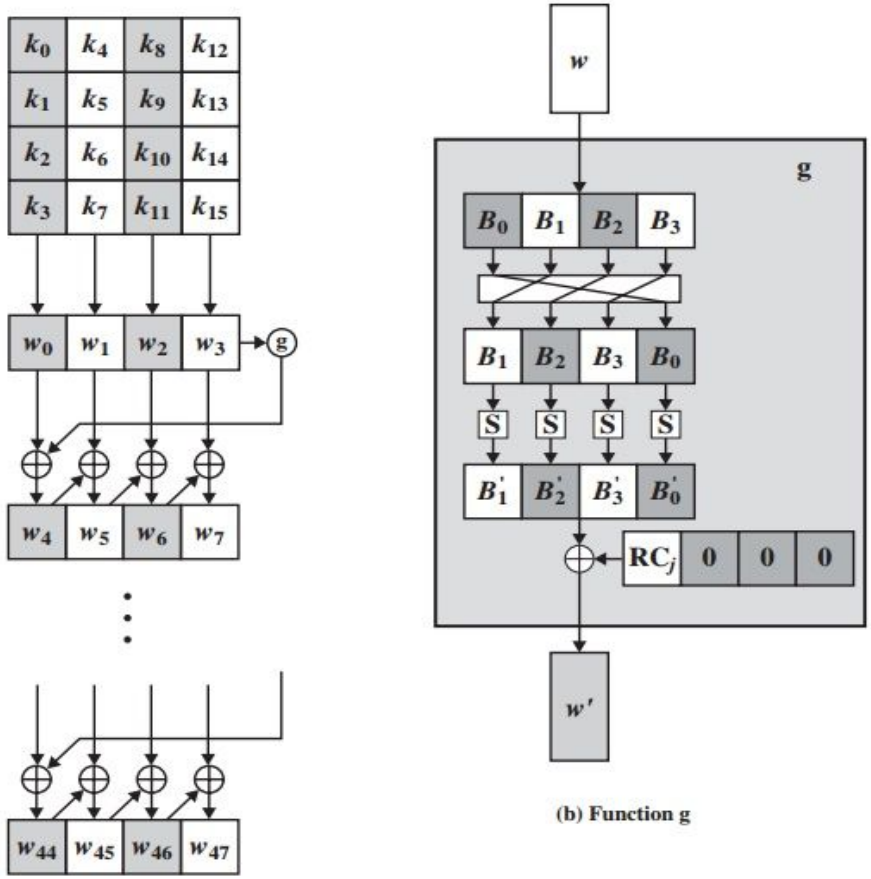
```
KeyExpansion(byte Key[4*Nk] word W[Nb*(Nr+1)])
{
      for(i = 0; i < Nk; i++)
            W[i] = (Key[4*i],Key[4*i+1],Key[4*i+2],Key[4*i+3]);

      for(i = Nk; i < Nb * (Nr + 1); i++)
      {
            temp = W[i - 1];
            if (i % Nk == 0)
                  temp = SubByte(RotByte(temp)) ^ Rcon[i / Nk];
            W[i] = W[i - Nk] ^ temp;
      }
}
```

Geração de Subchave AES

- **RotWord-** executa um deslocamento esquerdo circular de um byte em uma palavra.
- **Rcon-Rcon(j) -** Faz uma operação de XOR entre o resultado dos dois primeiros itens com uma constante diferente a cada rodada (j). Essa constante é dada por Rcon(j)=(RC[j],00,00,00), onde RC[1]=1 e RC[j]=2 · RC[j-1], com a multiplicação sobre GF(28).

```
Rcon = [
    0x00, 0x01, 0x02, 0x04, 0x08, 0x10, 0x20, 0x40,
    0x80, 0x1B, 0x36, 0x6C, 0xD8, 0xAB, 0x4D, 0x9A,
    0x2F, 0x5E, 0xBC, 0x63, 0xC6, 0x97, 0x35, 0x6A,
    0xD4, 0xB3, 0x7D, 0xFA, 0xEF, 0xC5, 0x91, 0x39,
]
```

Geração de Subchave AES



(a) Overall algorithm

(b) Function g

Figure 5.9 AES Key Expansion

Entradas para o programa

- Nome do arquivo de entrada
- Nome do arquivo de saída
- Nome do arquivo com chave de Criptografia/Decriptografia"
- Digite o código:
  - 0 - Criptografar
  - 1 - Decriptografar

```
python AES.py imagem.jpg imagemC.jpg chavesAES.txt 0

python AES.py imagemC.jpg imagemD.jpg chavesAES.txt 1
```