**UNIVERSIDADE FEDERAL DE SERGIPE**

Iuri Rodrigo Ferreira Alves da silva
Gregory Medeiros Melgaço Pereira
Raul Rodrigo Silva de Andrade
Rafael Castro Nunes
Ruan Robert Bispo dos Santos
Vítor do Bomfim Almeida Carvalho

# Encriptação AES

São Cristóvão,SE
20 de abril de 2017

Iuri Rodrigo Ferreira Alves da silva
Gregory Medeiros Melgaço Pereira
Raul Rodrigo Silva de Andrade
Rafael Castro Nunes
Ruan Robert Bispo dos Santos
Vítor do Bomfim Almeida Carvalho

# Encriptação AES

Relatório em conformidade com as normas ABNT

Universidade Federal De Sergipe
Faculdade de Engenharia Eletrônica
Redes e Comunicações

São Cristóvão,SE
20 de abril de 2017

# Resumo

A busca por uma maior segurança e privacidade em diversas áreas da comunicação humana foi um grande ponto de partida para o surgimento da Criptografia, onde inicialmente surgiu a Criptografia Simétrica e após um tempo a Criptografia Assimétrica. Surgiram vários tipos de algoritmos para esses dois tipos de criptografia, como por exemplo o Algoritmo AES (Criptografia Simétrica) e Algoritmo RSA (Criptografia Assimétrica).

Nesse trabalho será feito uma explicação sobre o que é a criptografia, os seus tipos e sobre o algoritmo padrão de parâmetro AES utilizado nesse projeto para a criptografia de Audios, videos e imagens. Após isso, esse algoritmo será implementado no software computacional MATLAB. Com essa implementação será possível analisar o desempenho desse algoritmo.

**Palavras-chave** : Segurança, Privacidade, Criptografia, Algoritmo AES, MATLAB

# Lista de ilustrações

# Lista de tabelas

# Sumário

# 1 Introdução

No contexto capitalista e competitivo atual é cada vez mais prescindível que os dados enviados e recebidos, principalmente online, sejam protegidos, em que apenas quem envia e quem recebe tenha acesso ao seu conteúdo, garantindo assim o direito de privacidade. Essa ideia de segurança de dados se aplica diretamente à diversas áreas como : troca de mensagens entre usuários de aplicativos, compras e processos financeiros online e troca de informações entre países ou entre organizações de um único país, já que muitos conteúdos são confidenciais e apenas autoridades do governo podem ter acesso. Os fundamentos de segurança (REFERENCIAR) são definidos por disponibilidade, integridade, controle de acesso, autenticidade, não-repudiação e privacidade. Foi pensando-se nisso que a criptografia foi criada. A palavra criptografia que provém dos radicais gregos kriptos (oculto) e grafo (escrita), é o nome dado à ciência de codificar mensagens utilizando algoritmos que serão usados novamente para decodificar essa mensagem. A criptografia apresenta dois tipos básicos: Simétrica (chave fechada) e Assimétrica (chave aberta).

A criptografia assimétrica foi criada na década de 1970. Nesse modelo, cada dispositivo envolvido na comunicação possui dois tipos de chaves diferentes, uma particular e uma pública. Essas chaves são processos digitais complexos que podem eventualmente estar associados a senhas. A chave pública é conhecida por qualquer usuário e é utilizada quando se quer se comunicar com outro usuário de modo seguro. Já a chave particular apenas cada dispositivo conhece e tem a sua. É com essa chave particular que o destinatário pode descriptografar a mensagem que foi criptografada com sua respectiva chave pública. A mensagem pode ser entendido com um bem precioso, a chave pública o cadeado que protege esse bem e a chave particular é chave física capaz de abrir esse cadeado.

A vantagem desse método é a segurança, já que não é necessário o compartilhamento das chaves particulares e elas se encontram em poder do destinatário e da fonte, não há risco de intercepção por terceiros para saber essa chave particular, eles apenas podem conhecer a chave pública do destinatário. É importante ressaltar que para um dispositivo enviar uma mensagem a outro, ele já tem que conhecer a chave pública do destino. A desvantagem é que com esse método o tempo de processamento de mensagens fica muito maior que a criptografia simétrica. Vários algoritmos para a criptografia assimétrica já existem, como o RSA e o Elgamal. O algoritmo RSA é o algoritmo de chave pública mais amplamente utilizado, além de ser uma das mais poderosas formas de criptografia de chave pública conhecidas até o momento. O RSA utiliza números primos. A premissa por trás do RSA consiste na facilidade de multiplicar dois números primos para obter um terceiro número, mas muito difícil de recuperar os dois primos a partir daquele terceiro número.

A criptografia simétrica é o modelo mais antigo de criptografia. Nesse modelo, a chave que dá acesso ao conteúdo da mensagem trocada entre dois dispositivos deve permanecer em segredo. Geralmente essa chave é representada por uma senha que é usada pelo remetente para codificar a mensagem e usada pelo destinatário para decodificar a mensagem. A vantagem desse modelo é a sua simplicidade. Caso a chave seja complexa o algoritmo não necessariamente precisa também ser muito complexo, o que é bom, já que quanto mais simples o algoritmo, maior é a sua velocidade de processamento e facilidade de implementação. A principal desvantagem deste modelo é que como é utilizada apenas uma chave para ciframento e desciframento, conhecendo a chave se tem acesso aos dois processos, o que pode ocorrer interceptando o canal utilizado. Com isso, é muito importante a comunicação por um canal seguro evitando assim a ação de intrusos que podem ter acesso a mensagem.

Outros problemas desse método é que como cada par necessita de uma chave, em uma rede com 'n' usuários, serão necessárias $n^2$ chaves, o que dificulta o gerenciamento. Além disso, não é fácil armazenar essas chaves de forma segura. Com isso, esse método não garante os princípios de autenticidade e não-repudiação. Vários algoritmos para a criptografia simétrica já existem, como o AES e o DES. O algoritmo AES é o mais utilizado e é o adotado como padrão pelo governo dos EUA. Esse algoritmo possui um bloco fixo em 128 bits e uma chave com tamanho de 128, 192 ou 256 bits, é relativamente fácil de executar e requer pouca memória. Esse algoritmo será o utilizado neste projeto.

# 2 Revisão bibliográfica

Com o avanço tecnológico alcançado pelo homem, aumentou-se a necessidade por segurança em todos os aspectos. As trocas de informações se tornaram mais intensas e se viu necessária a presença de ferramentas para a proteção de arquivos armazenados em bancos de dados. A criptografia é usada como um desses métodos para assegurar sigilo de qualquer tipo de informação virtual, tornando-as códigos a serem decriptadas apenas pelo receptor da mensagem.

Em (LU; TSENG, 2002) é proposto um método para integrar a criptografia e descriptografia AES em uma ferramenta funcional. Para implementar esse algoritmo é mostrado que são necessárias 5 operações principais: AddRoundKey, SubBytes, ShiftRows, MixColumns e keyExpansion, em que são detalhadas cada uma dessas operações para implementação do algoritmo. Ao término do trabalho, conclui-se que a criptografia AES mostra-se bastante eficiente e de baixa complexidade, proporcionando altas taxas de processamento em hardwares tanto no processo de encriptação como na decriptação.

Em (OLIVEIRA, 2012) é ressaltado que as técnicas computacionais tornaram-se fundamentais para que os requisitos da proteção a informação sejam atendidos, apresentando neste cenário dois tipos básicos para criptografia: simétrica e assimétrica. Na criptografia simétrica, que se encontra o AES, existe a presença de uma chave secreta, usada tanto pelo remetente quanto o destinatário e a ideia é que se use um algoritmo de deciframento junto com esta chave para cifrar e decifrar a mensagem. As principais vantagens observadas são a simplicidade e rapidez para executar os processos criptográficos.

Na criptografia assimétrica, diferentemente da simétrica, existem duas chaves, uma secreta e outra pública. A chave pública pode ficar disponível para qualquer pessoa que queira se comunicar de modo seguro, mas a secreta é de posse apenas de cada titular e é com ela que o destinatário poderá decodificar uma mensagem que foi criptografada para ele com sua respectiva chave pública. A vantagem desse sistema é permitir a qualquer um enviar uma mensagem secreta, apenas utilizando a chave pública de quem irá recebê-la, a desvantagem é a complexidade empregada no desenvolvimento dos algoritmos que devem ser capazes de reconhecer a dupla de chaves existentes.

já no artigo (RIBEIRO, 2001) é feito uma comparação entre DES(Data encryption Standard) e AES(Advanced Encryption Standard), onde é indagado, no começo, quais as melhorias trazidas pelo AES, já que este substituiu o DES como sistema de criptografia. Sendo que as principais diferenças entre o DES e o AES são os algoritmos concorrentes, onde no DES só tinha 1, no AES tiveram vários.

Para análise de algoritmos, ambos foram testados na mesma máquina para obtenção

dos resultados. No experimento foi obtido que entre o DES e o Rijndael o tempo do DES é menor, junto ao tamanho do arquivo criado.

# 3 Objetivos

A criptografia está relacionada a segurança de dados e este é o seu principal objetivo, garantir a transmissão de dados encriptados, não permitindo a interpretação e entendimento destes dados por pessoas não atutorizadas.

O algoritmo de Rijndael foi criado com objetivo de atender os parâmetros AES, substituindo os parâmetros utilizados anteriormente, DES, criando um novo padrão de criptografia.

Este trabalho teve como objetivo principal a implementação, utilizando o software MatLab, do algoritmo de Rijndael, que segue os parâmetros AES de encriptação. Através da implementação, realizar testes e verificar o funcionamento do algoritmo, assim como avaliar a eficiência e eficácia da implementação feita. Utilizar o algoritmo para decriptografar e comparar com a mensagem original antes da criptografia.

# 4 Fundamentação Teórica

## 4.1 Criptografia

A palavra criptografia tem origem no grego cryptos, que significa segredo, oculto, e isto já nos dá uma boa ideia do que significa criptografar. Com o intuito de que uma mensagem transmitida, seja ela um texto, imagem, áudio ou qualquer outro de informação seja entendida apenas pelo destinatário desejado, podemos criptografá-la, ou seja, tornar ilegível a qualquer um que não contenha a chave, necessária que a mensagem seja decriptografada. A criptografia pode ser então entendida como uma ferramenta de segurança, garantindo que apenas quem ou o que possuir autorização possa interpretar a mensagem enviada.

Conforme a tecnologia avança se torna mais fácil e rápido o processamento de dados, permitindo a evolução dos algoritmos de criptografia, porém evolui também os métodos e algoritmos para quebrar e decifrar mensagens criptografadas, exigindo maiores níveis de segurança de informação.

### 4.1.1 Criptografia Simétrica

Criptografia simétrica é o tipo de criptografia que utiliza a mesma chave, tanto para o processo de criptografia quanto para a decriptografia.

### 4.1.2 Criptografia Assimétrica

A criptografia assimétrica utiliza chaves diferentes para o processo de criptografia e decriptografia

## 4.2 Bit x Byte x Word

Um bit não é nada além de um binário, ou seja pode assumir o valor 0 ou 1. Um byte é um conjunto de 8 bits. Uma word representa um conjunto de bytes, não apresentando tamanho fixo, mas uma multiplicação de um fato inteiro pelo número de bytes.

## 4.3 Operação XOR

Esta operação nada mais é do que uma comparação. Comparando 2 bits, se estes forem iguais a operação retorna o bit 0 se estes forem diferentes a operação retorna o bit 1.

## 4.4 AES (Advanced Encryption Standard)

É uma cifra, para criptografia simétrica, que utiliza tamanho de bloco de 128 bits e o tamanho de chave pode variar entre 128,192 ou 256 bits. Isto significa que a entrada, mensagem, será um bloco de 128 bits, o tamanho da chave será definido entre as opções ditas acima e o algoritmo responsável pela criptografia, retornará um bloco de 128 bits, chamado cifra. Utilizando a chave correta, e como entrada a cifra de 128 bits, retornaremos a mensagem original.

Tabela 1 – Parâmtros do AES

| **Tamanho da Chave (words/bytes/bits)** | 4/16/128 | 6/24/192 | 8/32/256 |
|---|---|---|---|
| **Tamanho do Bloco de entrada (words/bytes/bits)** | 4/16/128 | 4/16/128 | 4/16/128 |
| **Número de Rodadas** | 10 | 12 | 14 |
| **Tamanho da Chave da Rodada (words/bytes/bits)** | 4/16/128 | 4/16/128 | 4/16/128 |
| **Tamanho da Chave Expandida (words/bytes)** | 44/176 | 52/208 | 60/240 |

### 4.4.1 Algoritmo de Rijndael

O algoritmo apresenta um número limitado de rodadas para concluir o processo de criptografia, este número depende do tamanho de chave escolhido, sendo assim:

- 10 se o bloco e a chave forem de 128 bits;
- 12 se o bloco ou a chave forem de 192 bits, e nenhum deles for maior que isso;
- 14 se o bloco ou a chave forem de 256 bits.

Observando que para entender os requisitos do AES o bloco utilizado será sempre de 128 bits. Este bloco é representado por uma matriz quadrada de bytes e copiado para um vetor State, este vetor é alterado em cada etapa, seja na criptografia ou decriptografia.

Para cada rodada então teremos quatro estágios diferentes, sendo um de permutação e três de substituição. Os quatro estágios e suas funções são os seguintes:

**SubBytes:** Utiliza uma matriz, caixa-S, para realizar a substituição byte a byte do bloco

**ShiftRows:** Uma permutação simples

**MixColumns:** Uma combinação linear

**AddRoundKey:** Um XOR bit a bit

Tanto para a criptografia, quanto para a decriptografia, a cifra inicia no estágio AddRoundKey, seguido de (n-1) rodadas com os quatro estágios, sendo n igual a o número

total de rodadas, e uma última rodada apenas com três estágios. É importante notar que apenas o estágio AddRoundKey utiliza a chave, sendo o único que agrega segurança. Os demais estágios são importantes para adicionar confusão, difusão e não-linearidade.

#### 4.4.1.1 Estágio SubBytes

Este estágio consiste numa substituição direta de bytes. O AES definiu uma matriz, chamada Caixa-S, com 16x16 elementos, de valores de bytes. Esta matriz contém todos os valores possíveis de 8 bits permutados.

Tabela 2 – Caixa-S

| | **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 | C | 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 | 44 | 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 |

Cada byte do vetor State é então atualizado para um valor correspondente contido na Caixa-S. Os quatro primeiros bits de um byte serão correspondentes a linha e os quatro últimos bits correspondem a coluna, com os quatro bits transformados em hexadecimal temos uma posição na Caixa-S, o elemento substituirá o atual no vetor State. Para descriptografia é feito o mesmo processo, porém com a Caixa-S inversa.

#### 4.4.1.2 Estágio ShiftRows

Neste estágio ocorre um simples deslocamento nas linhas. Na primeira linha não ocorre deslocamento, na segunda ocorre o deslocamento circular de 1 byte à esquerda, na terceira ocorre o deslocamento circular de 2 bytes à esquerda e na quarta ocorre o deslocamento circular de 3 bytes à esquerda. Como mostrado no exemplo abaixo:

Para a decriptografia o deslocamento é feito para a direita, nas três últimas linhas.

Tabela 3 – Exemplo de ShiftRows

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| 87 | F2 | 4D | 97 | → | 87 | F2 | 4D | 97 |
| EC | 6E | 4C | 90 | → | 6E | 4C | 90 | EC |
| 4A | C3 | 46 | E7 | → | 46 | E7 | 4A | C3 |
| 8C | D8 | 95 | A6 | → | A6 | 8C | D8 | 95 |

#### 4.4.1.3 Estágio MixColumms

Neste estágio ocorre atualização na matriz State, através de uma operação que relaciona cada coluna separadamente. Podemos expressar como a seguinte multiplicação de matrizes:

$$\begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} S_{0,0} & S_{0,1} & S_{0,2} & S_{0,3} \\ S_{1,0} & S_{1,1} & S_{1,2} & S_{1,3} \\ S_{2,0} & S_{2,1} & S_{2,2} & S_{2,3} \\ S_{3,0} & S_{3,1} & S_{3,2} & S_{3,3} \end{bmatrix} = \begin{bmatrix} S'_{0,0} & S'_{0,1} & S'_{0,2} & S'_{0,3} \\ S'_{1,0} & S'_{1,1} & S'_{1,2} & S'_{1,3} \\ S'_{2,0} & S'_{2,1} & S'_{2,2} & S'_{2,3} \\ S'_{3,0} & S'_{3,1} & S'_{3,2} & S'_{3,3} \end{bmatrix}$$

Os novos elementos para cada coluna serão então definidos pelas equações:

$$\begin{aligned} S'_{0,j} &= (2 \cdot S_{0,j}) \oplus (3 \cdot S_{1,j}) \oplus S_{2,j} \oplus S_{3,j} \\ S'_{1,j} &= S_{0,j} \oplus (2 \cdot S_{1,j}) \oplus (3 \cdot S_{2,j}) \oplus S_{3,j} \\ S'_{2,j} &= S_{0,j} \oplus S_{1,j} \oplus (2 \cdot S_{2,j}) \oplus (3 \cdot S_{3,j}) \\ S'_{3,j} &= (3 \cdot S_{0,j}) \oplus \cdot S_{1,j} \oplus S_{2,j} \oplus (2 \cdot S_{3,j}) \end{aligned}$$

#### 4.4.1.4 Estágio AddRoundKey

Nesse estágio e que incluímos a presença da chave no algoritmo. Cada bit da matriz State passa por um XOR bit a bit com a chave da rodada. O resultado desta operação é o que atualiza a matriz State. Para decriptografar esse estágio é igual, já que a operação XOR é a própria inversa.

#### 4.4.1.5 Algoritmo de Expansão de Chave

Este algoritmo é utilizado para garantir que cada rodada tenha uma nova chave. Ele utiliza como entrada uma chave de 4 words e produz um vetor de 44 words. Garantindo uma chave de 4 words para cada rodada.

# 5 Formulação do Problema

## 5.1 Etapas

### 5.1.1 Cipher e decipher

A função Cipher é responsável por criptografar cada pacote de 16bytes

```
function ciphertext = cipher (plaintext, w, s_box, poly_mat,ind_matleft)
 plaintext=reshape(plaintext,4,4); % organiza a entrada em matriz
 plaintext = abs(plaintext); % converte para decimal
 state = plaintext; % estado recebe texto leg vel
 round_key = (w(1:4, :))'; % chave da rodada

%%%%% Passo inicial (soma) %%%%%
 state = bitxor(round_key,state);

%%%%% Passos intermedi rios ( 9x ) %%%%%
 for i_round = 1 : 9
    %substitui   o
    state = s_box (state + 1);

    %rota   o
    state = state(ind_matleft);

    %multiplica   o
    state = mix_columns (state, poly_mat);

    %soma de colunas
    round_key = (w((1:4) + 4*i_round, :))';
    state = bitxor (state, round_key);
 end

%%%%% Passos finais %%%%%
 %substitui   o
 state = s_box (state+1);

 %rota   o
 state = state(ind_matleft);

 %soma de colunas
 round_key = (w(41:44, :))';
 ciphertext = bitxor (state, round_key);

 %organiza   o
 ciphertext = reshape(ciphertext,1,16); % Organiza arquivo criptografado
%  ciphertext = char(ciphertext); % Transforma para formato original
```

A função decipher realiza o processo contrário, realizando a decriptação de um pacote de 16bytes.

```
function deciphertext = decipher (plaintext, w, inv_s_box, inv_poly_mat,inv_matright)
% organiza a entrada em matriz
 plaintext=reshape(plaintext,4,4);
```

```
 plaintext = abs(plaintext); % converte para decimal
 state = plaintext; % estado recebe texto leg vel
 round_key = (w(41:44, :))'; % chave da rodada

%%%%% Passo inicial (soma) %%%%%
 state = bitxor(round_key,state);

%%%%% Passos intermedi rios ( 9x ) %%%%%
 for i_round = 9:-1:1
    %rota   o
    state = state(inv_matright);

    %substitui   o
    state = inv_s_box (state + 1);

    %soma de colunas
    round_key = (w((1:4) + 4*i_round, :))';
    state = bitxor (state, round_key);

    %multiplica   o
    state = mix_columns (state, inv_poly_mat);

 end

%%%%% Passos finais %%%%%
 %rota   o
 state = state(inv_matright);

 %substitui   o
 state = inv_s_box (state+1);

 %soma de colunas
 round_key = (w(1:4, :))';
 deciphertext = bitxor (state, round_key);

 %organiza   o
 % Organiza arquivo criptografado
 deciphertext = reshape(deciphertext,1,16);
 % Transforma para formato original
 deciphertext = char(deciphertext);
```

### 5.1.2 mix columns

A função a seguir (mix colums) realiza a multiplicação entre matrizes:

```
function state_out = mix_columns (state_in, poly_mat)
 mod_pol = bin2dec ('100011011');
 for i_col_state = 1 : 4
 for i_row_state = 1 : 4
 temp_state = 0;
 for i_inner = 1 : 4
 temp_prod = poly_mult (...
 poly_mat(i_row_state, i_inner), ...
 state_in(i_inner, i_col_state), ...
 mod_pol);
 temp_state = bitxor (temp_state, temp_prod);
end
 state_out(i_row_state, i_col_state) = temp_state;
```

```
end
end
```

### 5.1.3 Key expansion

Vai expandir a matriz da chave, expandindo de 4x4 para 4x44

```
function w = key_expansion (chave, s_box, rcon)
 chave = abs(chave); % converte
 w = (reshape (chave, 4, 4))';
 for i = 5 : 44
 temp = w(i - 1, :);
 if mod (i, 4) == 1
 temp =  temp([2 3 4 1]); %rota   o
 temp = s_box(temp+1);    %substitui   o
 r = rcon ((i - 1)/4, :);
 temp = bitxor (temp, r); %soma pt1
end
 w(i, :) = bitxor (w(i - 4, :), temp); %soma pt2
end
```

## 5.2 Vídeo

### 5.2.1 Encriptar

Programa principal (crip video) para encriptar vídeo.

```
clc
clear all
ssi; % Carrega matrizes pre definidas

%% Trabalho de Redes de comunica   es
%%%%%%%%  CRIPTOGRAFIA AES %%%%%%%%
%
% Entrada:                          Video
% Saida:          Video criptografado

%--------------- Chave ----------------%
chave = ['1234567812345678'];
%expan   o da chave
subchaves = key_expansion(chave, s_box, rcon);

%--------------- Entrada ----------------%
% Arquivo a ser carregado
fid = fopen('ex-quilo.mp4','r');
entrada = fread(fid,'uint8');
fclose(fid);

%---------- Criptografia ----------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);

         %arquivo criptografado
```

```
        ciphertext = cipher (parcial, subchaves, s_box,
        poly_mat,ind_matleft);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Criptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];

        %arquivo criptografado
        ciphertext = cipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Criptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow
    fid = fopen('Video criptografado.mp4','w');
    fwrite(fid,a);
    fclose(fid);
```

### 5.2.2 Decriptar

Programa (decrip video) para decriptografar o video criptografado obtido a partir do programa anterior:

```
clc
clear all
ssi; % Carrega matrizes pre definidas

%% Trabalho de Redes de comunica  es
%%%%%%%%  CRIPTOGRAFIA AES %%%%%%%%
%
% Entrada:        Video criptografado
% Saida:                          Video

%--------------- Chave ----------------%
chave = ['1234567812345678'];
%expans o da chave
subchaves = key_expansion(chave, s_box, rcon);

%--------------- Entrada ----------------%
% Arquivo a ser carregado
fid = fopen('Video criptografado.mp4','r');
entrada = fread(fid,'uint8');
fclose(fid);
```

```
%---------- Descriptografia ----------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);

        % retorna arquivo criptografado
        ciphertext = decipher(parcial, subchaves, inv_s_box, inv_poly_mat,ind_matright);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Descriptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];

        % retorna arquivo criptografado
        ciphertext = decipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Descriptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow

display('Video Recuperado')
fid = fopen('Video descriptografado.mp4','w');
fwrite(fid,a);
fclose(fid);
```

## 5.3 Imagem

### 5.3.1 Encriptar

Programa principal para encriptar imagem.

```
clc
clear all
ssi; % Carrega matrizes pre definidas

%% Trabalho de Redes de comunica  es
%%%%%%%%% CRIPTOGRAFIA AES %%%%%%%%
%
% Entrada:                    Imagem
% Saida:        Imagem criptografada

%---------- Chave ----------%
chave = ['1234567812345678'];
```

```
%expans o da chave
subchaves = key_expansion(chave, s_box, rcon);

%------------ Entrada ---------------%
% Arquivo a ser carregado
fid = fopen('meg.png','r');
entrada = fread(fid,'uint8');
fclose(fid);

%---------- Criptografia ------------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);
        % retorna arquivo criptografado
        ciphertext = cipher (parcial, subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Criptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];
        % retorna arquivo criptografado
        ciphertext = cipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Criptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow
    fid = fopen('imagem criptografada.png','w');
    fwrite(fid,a);
    fclose(fid);
```

### 5.3.2 Decriptar

Programa principal para decriptar a imagem.

```
clc
clear all
ssi; % Carrega matrizes pre definidas

%% Trabalho de Redes de comunica  es
%%%%%%%%% CRIPTOGRAFIA AES %%%%%%%%
%
% Entrada:       Imagem criptografada
% Saida:                        Imagem

%------------- Chave ---------------%
```

```
chave = ['1234567812345678'];
%expan   o da chave
subchaves = key_expansion(chave, s_box, rcon);

%------------ Entrada ---------------%
% Arquivo a ser carregado
fid = fopen('imagem criptografada.png','r');
entrada = fread(fid,'uint8');
fclose(fid);

%--------- Descriptografia ----------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);
        % retorna arquivo criptografado
        ciphertext = decipher(parcial, subchaves, inv_s_box, inv_poly_mat,ind_matright);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Descriptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];
        % retorna arquivo criptografado
        ciphertext = decipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Descriptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow

display('Imagem Recuperada')
fid = fopen('Imagem descriptografada.png','w');
fwrite(fid,a);
fclose(fid);
```

## 5.4 Audio

### 5.4.1 Encriptar

```
clc
clear all
ssi; % Carrega matrizes pre definidas

%% Trabalho de Redes de comunica  es
%%%%%%%%  CRIPTOGRAFIA AES %%%%%%%%
%
```

```
% Entrada:                          Audio
% Saida:          Audio criptografado

%-------------- Chave ----------------%
chave = ['1234567812345678'];
%expans o da chave
subchaves = key_expansion(chave, s_box, rcon);

%-------------- Entrada ----------------%
% Arquivo a ser carregado
fid = fopen('Weee (Angry Birds).mp3','r');
entrada = fread(fid,'uint8');
fclose(fid);

%---------- Criptografia ----------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);

        % retorna arquivo criptografado
        ciphertext = cipher (parcial, subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Criptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];
        % retorna arquivo criptografado
        ciphertext = cipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Criptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow
    sound(a)
    fid = fopen('audio criptografado.mp3','w');
    fwrite(fid,a);
    fclose(fid);
```

### 5.4.2 Decriptar

Programa principal para decriptografar a imagem

```
clc
clear all
ssi; % Carrega matrizes pre definidas
```

```
%% Trabalho de Redes de comunica  es
%%%%%%%%  CRIPTOGRAFIA AES %%%%%%%%
%
% Entrada:        Imagem criptografada
% Saida:                          Imagem

%--------------- Chave ----------------%
chave = ['1234567812345678'];
%expan  o da chave
subchaves = key_expansion(chave, s_box, rcon);

%--------------- Entrada ----------------%
% Arquivo a ser carregado
fid = fopen('imagem criptografada.png','r');
entrada = fread(fid,'uint8');
fclose(fid);

%---------- Descriptografia ----------%
a=[];
    for i_round=0:1:(fix(length(entrada)/16)-1)
        parcial = entrada((1:16)+ 16*i_round,1);
        % retorna arquivo criptografado
        ciphertext = decipher(parcial, subchaves, inv_s_box, inv_poly_mat,ind_matright);
        a = [a ciphertext];
        if (rem(length(a),1000) == 0)
            hold off
            barh(length(a),'b')
            title('Descriptografando...')
            xlim([0 length(entrada)])
            ylim([-3 5])
            drawnow
        end
    end
    rest = rem(length(entrada),16);
    if rest ~=0
        fim = [entrada(length(entrada)-rest:length(entrada)); zeros((15-rest),1)];
        % retorna arquivo criptografado
        ciphertext = decipher (fim', subchaves, s_box, poly_mat,ind_matleft);
        a = [a ciphertext(1:rest)];
    end
    % Plot final
    hold off
    barh(length(a),'b')
    title('Descriptografia concluida')
    xlim([0 length(entrada)])
    ylim([-3 5])
    drawnow

display('Imagem Recuperada')
fid = fopen('Imagem descriptografada.png','w');
fwrite(fid,a);
fclose(fid);
```

# 6 Resultados Obtidos

Após a implementação do algoritmo, os códigos foram testados utilizando diferentes formatos de arquivo de entrada, permitindo analisar o funcionamento do algoritmo para tais entradas.

Para todas as entradas de áudio, imagem ou vídeo testadas, o algoritmo conseguiu criptografar e após decriptografar foi possível observar e visualizar a entrada original, sem alteração.

Foram testados arquivos de áudio dos formatos mp3, wav, m4a,sendo que em todos os formatos como o algoritmo criptografa o cabeçalho de reconhecimentos dos players comuns de mídia ao executar nesses players o arquivo tem comportamento similar ao de arquivo corrompido. Isso pode ser visto como uma vantagem já que um possível invasor não conseguira interpretar o arquivo, podendo concluir qu pacotes foram perdidos durante a transmissão, pois a criptografia apresentará um resultado semelhante a perda de dados. O algoritmo implementado permite escutar o ruído proveniente do áudio encriptado, através do MatLab.

Os arquivos de imagem encriptados, antes da decriptografia, só podem ser visualizados através do MatLab e quando abertos mostram uma imagem similar a uma linha colorida, escondendo completamente o conteúdo do arquivo.

# 7 Conclusão

Conclui-se que, com esse trabalho, foi possível reproduzir um algoritmo de criptografia capaz de realizar a encriptação de diversos formatos de mídia. Uma vez que o algoritmo AES trabalha com pacotes de 16 bytes e chave simétrica é possível, com um sub-algoritmo, quebrar arquivos maiores que 16 bytes em vários formatos, seja vídeo, áudio, imagem ou texto. Vale ressaltar que todos os formatos de mídia testados foram encriptados e decriptado com sucesso. Embora não fosse possível reproduzir os arquivos encriptados.

# Referências


BURNETT, S.; PAINE, S. *Criptografia e segurança: o guia oficial RSA*. [S.l.]: Gulf Professional Publishing, 2002. Nenhuma citação no texto.

COUTINHO, S. C. *Números inteiros e criptografia RSA*. [S.l.]: IMPA, 2009. Nenhuma citação no texto.

HAMALAINEN, P. et al. Design and implementation of low-area and low-power aes encryption hardware core. In: IEEE. *Digital System Design: Architectures, Methods and Tools, 2006. DSD 2006. 9th EUROMICRO Conference on*. [S.l.], 2006. p. 577–583. Nenhuma citação no texto.

HARRISON, O.; WALDRON, J. Aes encryption implementation and analysis on commodity graphics processing units. In: SPRINGER. *International Workshop on Cryptographic Hardware and Embedded Systems*. [S.l.], 2007. p. 209–226. Nenhuma citação no texto.

LU, C.-C.; TSENG, S.-Y. Integrated design of aes (advanced encryption standard) encrypter and decrypter. In: IEEE. *Application-Specific Systems, Architectures and Processors, 2002. Proceedings. The IEEE International Conference on*. [S.l.], 2002. p. 277–285. Citado na página 9.

OLIVEIRA, R. R. Criptografia simétrica e assimétrica-os principais algoritmos de cifragem. *Segurança Digital [Revista online]*, v. 31, p. 11–15, 2012. Citado na página 9.

RIBEIRO, V. G. Um estudo comparativo entre algoritmos de criptografia des–lucifer (1977) e aes–rijndael (2000). In: *VII Congreso Argentino de Ciencias de la Computación*. [S.l.: s.n.], 2001. Citado na página 9.

SILVA, M. d. L. G. da; OLIVEIRA, C. C. Criptografia assimétrica em documentos de áudio: uma experiência inicial com o algoritmo rsa. Nenhuma citação no texto.

STALLINGS, W.; VIEIRA, D. *Criptografia e segurança de redes: princípios e práticas*. [S.l.]: Pearson Prentice Hall, 2008. Nenhuma citação no texto.