UNIVERSIDADE FEDERAL DE PERNAMBUCO
CENTRO DE TECNOLOGIA E GEOCIÊNCIAS
DEPARTAMENTO DE ELETRÔNICA E SISTEMAS
PROGRAMA DE PÓS-GRADUAÇÃO EM ENGENHARIA ELÉTRICA

JOSÉ ANTONIO PÉREZ DE MORALES ARTILES

TÉCNICAS DE CIFRAGEM E AUTENTICACÃO DE IMAGENS UTILIZANDO SEQUÊNCIAS CAÓTICAS

Recife

2020

JOSÉ ANTONIO PÉREZ DE MORALES ARTILES

TÉCNICAS DE CIFRAGEM E AUTENTICACÃO DE IMAGENS UTILIZANDO SEQUÊNCIAS CAÓTICAS

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, como pré-requisito para obtenção do título de Doutor em Engenharia Elétrica.

Área de Concentração: Comunicações.

Orientador: Prof. Dr. Cecilio José Lins Pimentel.

Coorientador: Prof. Dr. Daniel Pedro Bezerra Chaves.

Recife

2020

Catalogação na fonte:
Bibliotecária Sandra Maria Neri Santiago, CRB-4 / 1267

A791t Artiles, José Antonio Pérez de Morales.
Técnicas de cifragem e autenticação de imagens utilizando sequências caóticas / José Antonio Pérez de Morales Artiles. – 2020.
97 f.: il., figs., tabs.

Orientador: Prof. Dr. Cecilio José Lins Pimentel.
Coorientador: Prof. Dr. Daniel Pedro Bezerra Chaves.
Tese (Doutorado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica, Recife, 2020.
Inclui referências.

1. Engenharia elétrica. 2. Padrão avançado de criptografia. 3. Mapas caóticos. 4. Criptoanálise. 5. Marca d'agua. 6. Código corretor de erro. I. Pimentel, Cecilio José Lins (Orientador). II. Chaves, Daniel Pedro Bezerra (Coorientador). III. Título.

UFPE

621.3 CDD (22. ed.) BCTG/2021-241

JOSÉ ANTONIO PÉREZ DE MORALES ARTILES

TÉCNICAS DE CIFRAGEM E AUTENTICACÃO DE IMAGENS UTILIZANDO SEQUÊNCIAS CAÓTICAS

Tese apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, Centro de Tecnologia e Geociências, como pré-requisito para obtenção do título de Doutor em Engenharia Elétrica.

Área de Concentração: Comunicações.

Aprovado em: 18 / 12 / 2020 .

Prof. Dr. Daniel Pedro Bezerra Chaves (Coorientador e Examinador Interno)
Universidade Federal de Pernambuco

Prof. Dr. Juliano Bandeira Lima (Examinador Interno)
Universidade Federal de Pernambuco

Prof. Dr. Francisco Madeiro Bernardino Júnior (Examinador Externo)
Universidade Federal de Pernambuco

Prof. Dr. Carlos Eduardo Correia de Souza (Examinador Externo)
Universidade Federal de Pernambuco

Prof. Dr. Edson Guedes da Costa (Examinador Externo)
Universidade Federal de Pernambuco

Prof. Dr. Sílvio Barros Melo (Examinador Interno)
Universidade Federal de Pernambuco

A Deus por me dar a força necessária para não desistir nos momentos mais difíceis. Para meus pais por sempre me apoiarem em minhas decisões. A meus orientadores e amigos da UFPE, sem eles não teria sido possível este trabalho.

AGRADECIMENTOS

Agradeço em primeiro lugar a Deus por me dar capacidade e ferramentas para realizar este trabalho, pois sem ELE sei que não seria possível concluí-lo.

À minha família pelo apoio, compreensão, carinho e apoio incondicional nas diversas etapas e dificuldades deste trabalho. Em especial à minha mãe e meu pai, por sempre se preocuparem em oferecer o melhor possível para seus filhos.

Aos meus orientadores, professor Cecilio José Lins Pimentel e o professor Daniel Pedro Bezerra Chaves, pela orientação, paciência, dedicação e esforço em sempre me guiar pelo melhor caminho, porque sem eles a realização deste trabalho não seria possível. Aos meus amigos, pelo apoio nos momentos de dificuldades.

Em especial gostaria de agradecer a Carlos Eduardo pelo apoio e por sempre que possível se dispor a me ajudar nos diversos assuntos teóricos, pelas observações ao longo do desenvolvimento do trabalho, além de sempre me dar força. Aos todos os estudantes do LACRI pelos conselhos e dicas na etapa final deste trabalho.

Aos professores do DES com os quais formei a base do meu aprendizado por meio de diversas disciplinas.

Aos funcionários do DES, em especial à Andréa Tenório, secretária do Programa de Pós Graduação em Engenharia Elétrica (PPGEE) por sua competência, organização e sempre disponibilidade em me ajudar nos diversos momentos.

Por fim à Coordenação do PPGEE e à Fundação de Amparo Ciência e Tecnologia do Estado de Pernambuco (FACEPE) pelo suporte financeiro ao longo do desenvolvimento desta tese.

RESUMO

Esta tese apresenta novas propostas de cifragem e autenticação de imagens aplicando sequências caóticas. Inicialmente, é apresentada uma técnica de cifragem de imagens utilizando a randomização da cifra AES (*Advanced Encryption Standard*) empregando mapas caóticos, mantendo mesmos indicadores de robustez do algoritmo original com redução de até 40% do número de iterações. Também é proposta uma alternativa para a camada de difusão do AES que reduz a complexidade do esquema de cifragem. Os algoritmos propostos são analisados por diferentes métricas de segurança comumente usadas para avaliar tanto a aleatoriedade do texto cifrado como a capacidade de resistência a ataques estatísticos. Em seguida, um algoritmo simplificado do AES é analisado a partir da perspectiva das criptoanálises linear e diferencial. Os resultados para estas criptoanálises mostram que a proposta é mais segura contra esses ataques que o algoritmo original. Esta tese também apresenta um esquema de marca d'água robusto para imagens. Os bits inseridos na imagem são os bits de paridade de um código corretor de erro cuja mensagem é formada pela combinação dos bits de marca d'água com bits caóticos, propiciando robustez a ataques. A robustez do método proposto é testada para vários ataques utilizados em processamentos de sinais e ataques geométricos. Comparações com resultados da literatura revelam que a técnica proposta apresenta desempenho melhor que alguns métodos existentes.

Palavras-chave: padrão avançado de criptografia; mapas caóticos; criptoanálise; marca d'água; código corretor de erro.

ABSTRACT

This thesis presents new proposals for image encryption and authentication using chaotic sequences. Initially, an image encryption technique is presented based on the randomization of the AES (Advanced Encryption Standard) cipher using chaotic maps, maintaining the same robustness indicators as the original algorithm with a reduction in the number of iterations of up to 40%. An alternative to the AES diffusion layer is also proposed which reduces the complexity of the encryption scheme. The proposed algorithms are analyzed by different security metrics commonly used to evaluate both the randomness of ciphertext and the resilience of statistical attacks. Then, a simplified AES algorithm is analyzed from the perspective of linear and differential cryptanalysis. The results for these cryptanalyses show that the proposal is safer against these attacks than the original algorithm. This thesis also proposes a robust image watermark scheme. The watermark bits inserted in the image are the parity bits of an error-correcting code whose message is formed by combining the watermark bits with chaotic bits, providing robustness to attacks. The robustness of the proposed method is tested for various attacks used in signal processing and geometric attacks. Comparisons with literature results reveal that the proposed technique performs better than some existing techniques.

Keywords: advanced encryption standard; chaotic maps; cryptanalysis; watermarking; error correcting code.

LISTA DE FIGURAS

LISTA DE TABELAS

SUMÁRIO

1 INTRODUÇÃO

Um sistema criptográfico é um conjunto de algoritmos necessários para implementar um serviço de segurança específico. Existem várias técnicas complementares para a proteção de dados: criptografia, estenografia, marca d'água (TAO et al., 2014). Este trabalho foca nas áreas de criptografia e marca d'água.

1.1 CRIPTOGRAFIA

Na área de criptografia um dos algoritmos mais utilizados é o Padrão Avançado de Criptografia (AES, *Advanced Encryption Standard*, atual algoritmo de cifra de bloco recomendado pelo Instituto Nacional de Padrões e Tecnologia (NIST, *National Institute of Standards and Technology*), sendo inicialmente indicado para proteção de informação digital não classificada de agentes governamentais, ou para a proteção de informação digital de agentes privados ((FIPS), 2001). A partir de 2003 a Agência de Segurança Nacional dos Estados Unidos (NSA, *National Security Agency*) passou a permitir o emprego do AES para proteção de informação classificada de agentes governamentais para comprimentos de chave de 192 e 256 bits. O algoritmo AES apresenta quatro blocos por iteração: substituição de bytes, deslocamento de linhas, mistura de colunas, adição de chaves. Uma das etapas mais importantes deste algoritmo é a substituição de bytes, que proporciona a confusão do texto cifrado (LI, 2006). Esta etapa é realizada por uma unidade denominada de S-box, que realiza um mapeamento bijetivo determinístico entre um byte de entrada e um byte de saída, definido por uma sequência de operações em GF(2^8). Para cifras de bloco em geral, as S-boxes não são suficientemente seguras contra ataques de criptoanálise diferencial devido à sua arquitetura rígida (BECHTSOUDIS; SKLAVOS, 2010). Portanto, técnicas para melhorar a segurança dessa unidade têm impacto proeminente na segurança da cifra de bloco como um todo.

Nesta tese, o AES é aplicado diretamente como unidade de cifragem, o equivalente a utilizá-lo no modo de operação ECB (*electronic code book*). Existem algumas fraquezas associadas a este modo de operação, visto que a cifragem é realizada de forma determinística (PAAR; PELZL, 2010). Isto significa que blocos de texto claros idênticos são cifrados em blocos de texto cifrado idênticos, quando a mesma chave é utilizada. Essa característica pode ser contornada com o uso de cifragem probabilística ou cifras randomizadas (PAAR; PELZL, 2010); quando cifragens de um mesmo bloco de texto claro geram possivelmente blocos de texto cifrado distintos, para a mesma chave. Uma cifra randomizada pode ser gerada com o emprego de outros modos de operação, tais como: CBC (*cipher block chaining*), CFB (*cipher feedback*), OFB (*output feedback*) e CRT (*counter mode*). Estes modos de operação apresentam uma maior segurança contra ataques de texto claro conhecido e permitem implementação rápida e eficiente (PREISHUBER et al., 2018). A análise de um algoritmo de cifragem de imagem baseado em CBC é realizada em (ZHANG; LI; HOU, 2017). Contudo, técnicas de criptoanálise

com texto claro escolhido mais sofisticadas comprometem a segurança do sistema (BIHAM; SHAMIR, 1993a). Como alternativa, pode-se empregar fontes de entropia descorrelacionadas do texto claro e da chave, *e.g.*, mapas caóticos; estratégia adotada nesta tese.

Recentemente, vários algoritmos para cifragem de imagens baseados em caos foram propostos na literatura (WENHAO; KEHUI; CONGXU, 2016; YANG; LIAO, 2018; WU et al., 2017; GUO; RIYONO; PRASETYO, 2018; WANG; LIU, 2017; TANG et al., 2016; LAN et al., 2018; XU et al., 2016; HUA; ZHOU; HUANG, 2019; HUA; ZHOU; ZHOU, 2019; ALAWIDA et al., 2019; CAO; MAO; ZHOU, 2020; TALHAOUI; WANG; TALHAOUI, 2020). Sistemas caóticos apresentam propriedades adequadas para cifragem de imagem, tais como: aperiodicidade, ergodicidade, sensibilidade à condição inicial (STROGATZ, 2001). Mapas caóticos são usados em projetos de algumas transformações como embaralhamento e permutação. Estes mapas também são usados para gerar números pseudo-aleatórios utilizados nos algoritmos de cifragem. No entanto, explorando a estrutura dos algoritmos, alguns esquemas de cifragem de imagens baseados em caos foram quebrados por ataques de texto claro conhecidos/escolhidos, veja por exemplo (LIU et al., 2016; LI, 2016; ZHU; SUN, 2018; PREISHUBER et al., 2018).

Métricas de segurança (*e.g.* entropia, coeficientes de correlação, sensibilidade à chave) (PREISHUBER et al., 2018; WANG; LIU, 2017; GUO; RIYONO; PRASETYO, 2018) são comumente utilizadas para avaliar a aleatoriedade do texto cifrado e a sua capacidade de resistir a ataques estatísticos. Outras análises devem ser realizadas em algoritmos criptográficos, tais como: criptoanálise linear (CL), criptoanálise diferencial (CD), criptoanálise diferencial impossível, entre outros. Neste trabalho, analisa-se a robustez dos algoritmos propostos sob a perspectiva da CL e da CD.

A CL foi introduzida por Matsui (MATSUI; YAMAGISHI, 1992) em 1992, sendo uma proposta para realizar um ataque à cifra de bloco do FEAL (*Fast Data Encipherment Algorithm*). Em 1993, Matsui (MATSUI, 1994) introduz formalmente a CL, utilizado-a como um ataque para quebrar o algoritmo DES (*Data Encryption Standard*). Matsui utilizou 243 pares de textos claros e textos cifrados para quebrar este algoritmo (MATSUI, 1994). A CL explora relações lineares dos bits de entrada e de saída das S-boxes que apresentam alta probabilidade, uma vez que as S-boxes, geralmente, são os únicos elementos não lineares em uma cifra de bloco. A CL é um ataque de texto claro conhecido, isto é, o adversário conhece um conjunto de pares de texto claro e texto cifrado obtidos com a mesma chave.

A CD é um ataque de texto claro escolhido proposta por Eli Biham e Adi Shamir em 1991 (BIHAM; SHAMIR, 1991a; BIHAM; SHAMIR, 1991b; BIHAM; SHAMIR, 1993b). Esta criptoanálise usa pares de texto claro relacionados por uma diferença constante, e o adversário então calcula as diferenças existentes nos textos cifrados correspondentes com a finalidade de detectar padrões estatísticos. As propriedades estatísticas dependem da natureza das S-boxes utilizadas no algoritmo de cifragem, de modo que o adversário analisa diferenças na saída das S-boxes para diferenças específicas na sua entrada. Se acontecer uma diferença particular no

texto cifrado, a cifra pode ser distinguida de uma cifra aleatória e assim pode-se reduzir o espaço de chaves, permitindo que a chave seja recuperada mais rapidamente que uma busca exaustiva.

1.2 MARCA D'ÁGUA

Observa-se uma ampla disseminação de conteúdos e de tecnologias de multimídia em redes de comunicação digital, o que alça a discussão sobre a proteção contra o uso e a distribuição não autorizados desse conteúdo. Como resposta a essa questão são disponibilizados os serviços de proteção do direito autoral, de rastreamento de cópias, de proteção de cópias, de autenticação de imagem; que devem ser preservados mesmo com a introdução de distorção, seja esta por manipulação não autorizada (*e.g.*, introdução de ruído, filtragem, rotação, translação, etc) ou decorrente do processamento de dados intrínseco a padrões (*e.g.*, compressão com perdas dos padrões JPEG e JPEG 2000). Essas premissas são atendidas pelo emprego de marcas d'água digitais (KATZENBEISSER; PETITCOLAS, 1999).

A marca d'água digital é tipicamente inserida no domínio espacial ou no domínio da frequência, dependendo do grau de robustez e imperceptibilidade desejados (KATZENBEISSER; PETITCOLAS, 1999). No primeiro caso, a marca d'água e inserida diretamente na imagem original. A segunda abordagem é projetada para inserir uma marca d'água no domínio da frequência das imagens originais. A inserção da marca d'água no domínio da frequência, o caso mais comum, é utilizada com a finalidade de difundir os bits de marca d'água por vários píxeis da imagem, o que, em associação com as propriedades da transformada, propicia resistência a diversos ataques (O'RUANAIDH; PUN, 1998; QASIM; MEZIANE; ASPIN, 2018). Diferentes transformadas são empregadas como a transformada discreta de Fourier (DFT) (O'RUANAIDH; PUN, 1998; SU et al., 2019), a transformada fracionária de fourier (FFT) (FARES; AMINE; SALAH, 2020), transformada Mellin-Fourier (SHAO et al., 2016), a transformada wavelet discreta (LEE; SEO; KIM, 2019; KESHAVARZIAN; AGHAGOLZADEH, 2016), a transformada discreta do cosseno (DCT) (QASIM; MEZIANE; ASPIN, 2018; MOOSAZADEH; EKBATANIFARD, 2019; PARAH et al., 2016; KO et al., 2020), a transformada harmônica polar (XU et al., 2019). Algoritmos híbridos de marca d'água no domínio da frequência em conjunto com a decomposição de valor único também foram propostos (KANG et al., 2018; AHMADI; ZHANG; WEI, 2020). A escolha da transformada dependerá de alguns fatores, destacando-se: o esforço computacional, a distorção por manipulação não autorizada ou decorrente de padrões. Entre os atributos pretendidos para a marca d'água ao aplicar-se transformadas estão a robustez a ataques geométricos e a transcodificação.

Para agregar robustez e segurança à marca d'água há duas técnicas comumente aplicadas: a inserção aleatória de bits de marca d'água em posições da imagem ou de sua transformada; e a cifragem da marca d'água. Ambas podem ser incorporadas a um algoritmo de marca d'água com a utilização de mapas caóticos (LOAN et al., 2018; CHEN; LEUNG, 2005; ROY; KUMAR, 2017; GU; GAO, 2013; WU; GUAN, 2007). Esses mapas são caracterizados por

possuírem sensibilidade à condição inicial e comportamento pseudo-aleatório, apesar de serem determinísticos, o que se traduz em trajetórias (ou sinais) assemelhadas a ruído (STROGATZ, 2001). Os mapas caóticos são empregados para embaralhar os bits de marca d'água (LOAN et al., 2018; CHEN; LEUNG, 2005; ROY; KUMAR, 2017) (agregando segurança ao método), bem como para definir as posições na imagem em que serão inseridos estes bits (GU; GAO, 2013; WU; GUAN, 2007) (agregando robustez ao método). Para suportar uma distorção severa imposta à imagem, códigos corretores de erro podem ser aplicados para tornar a marca d'água mais robusta (TAN et al., 2019; LEFEVRE; CARRE; GABORIT, 2019; SARRESHTEDARI; AKHAEE; ABBASFAR, 2018; FAN; CHAO; CHIEU, 2019; ANAND; SINGH, 2020). Neste caso, códigos corretores de erro projetados para canais AWGN são comumente aplicados, o que nem sempre corresponde ao padrão de erros sofrido pela marca d'água (KATZENBEISSER; PETITCOLAS, 1999). Dessa forma, os ganhos obtidos com essa técnica são sensíveis a como o código é projetado e incorporado ao algoritmo de marca d'água.

1.3 MAPAS CAÓTICOS

Denota-se sistema dinâmico aquele cuja saída no instante "k" depende de valores passados da saída, da entrada e do valor presente. Portanto, os sistemas dinâmicos possuem memória e são descritos por equações diferenciais no caso de sistemas contínuos, e por equações de diferença no caso de sistemas discretos. O estado de um sistema dinâmico é um conjunto de variáveis, que podem ou não estar associadas a quantidades físicas, cuja especificação (na ausência de excitações externas) determina completamente a evolução do sistema. Tal conjunto de variáveis, apesar de ter cardinalidade definida, não é unicamente especificado e está associado à memória do sistema dinâmico. Este conjunto recebe a denominação de vetor de estado quando é representado por meio de um espaço de dimensão n, portanto, contém o valor de todas as variáveis dependentes do sistema dinâmico em um dado instante de tempo ou iteração. O conjunto de todos os possíveis vetores de estados é conhecido como *espaço de fase*. Se a regra for aplicada em tempo discreto, este é chamado um sistema dinâmico de tempo discreto, também denominados de *mapa*.

O comportamento de mapas caóticos é observado mediante uma série temporal discreta $\{x_i\}_{i=0}^{\infty}$, obtida pela iteração de uma função $f(x)$, não linear, sobre uma condição inicial x_0, da seguinte forma (STROGATZ, 2001):

$$x_n = f(x_{n-1}), n = 1, 2, 3, \ldots. \tag{1.1}$$

Uma *órbita* de x_0 é um conjunto de pontos $\{x_0, f(x_0), f^2(x_0)\ ...\}$, em que f^k denota a k-ésima composição de $f(x)$ (ALLIGOOD; SAUER; YORKE, 1997). Uma técnica de obtenção de sequências binárias balanceadas utilizando mapas caóticos foi apresentada em (ARTILES et al., 2015). Descarta-se algumas amostras inicias devido ao comportamento transitório. Exemplos de mapas caóticos incluem o Mapa Cúbico (MC) $f(x) = 4x^3 - 3x$ (LAU; TSE, 2010), e o Mapa

Logístico (ML) $f(x) = rx(1-x)$ (LAU; TSE, 2010), em que r é um parâmetro de controle. Novos mapas caóticos unidimensionais foram desenvolvidos pela aplicação de transformações não-lineares na saída de mapas caóticos existentes (chamados de mapas sementes) (HUA; ZHOU; ZHOU, 2019; HUA; ZHOU; HUANG, 2019). Um exemplo destes mapas é o mapa seno-logístico (MSL) que é definido como (HUA; ZHOU; ZHOU, 2019)

$$f(x) = \text{sen}(\pi a x(1-x)) \tag{1.2}$$

em que o parâmetro de controle a pertence ao intervalo $[0, \infty)$. Novos mapas caóticos são obtidos pela aplicação da função cosseno em uma combinação de dois mapas sementes. Por exemplo, o mapa Tenda-Logístico-Cosseno (TLC), definido em (1.3) (HUA; ZHOU; HUANG, 2019) é dado por

$$f(x) = \begin{cases} \cos(\pi(2rx + 4(1-r)x(1-x) - 0,5)), & \text{para } 0 \leq x \leq 0,5; \\ \cos(\pi(2r(1-x) + 4(1-r)x(1-x) - 0,5)), & \text{para } 0,5 < x \leq 1 \end{cases} \tag{1.3}$$

em que o parâmetro de controle r pertence ao intervalo $[0, 1]$.

Existem parâmetros que quantificam a dinâmica caótica, e sabe-se que um sistema para ser caótico deve satisfazer as seguintes propriedades (STROGATZ, 2001):

- **Comportamento aperiódico a longo prazo**: Existem órbitas que não tendem a pontos fixos, órbitas periódicas, ou quase-periódicas.
- **Determinístico**: O comportamento irregular é dado pela não linearidade do sistema, não devido a ruídos externos, ou seja, o sistema deve ser inerentemente determinístico.
- **Sensibilidade às condições iniciais**: Órbitas com condições iniciais próximas separam-se exponencialmente (quantificado pelo expoente de Lyapunov (STROGATZ, 2001)).

Uma sequência unidimensional é obtida a partir de mapas caóticos unidimensionais pela iteração de $f(x)$, sobre x_0, da forma descrita em (1.1). Uma sequência binária balanceada $\{z_n\}$, denotada por sequência binária caótica, é gerada a partir de $\{x_n\}$ particionando o espaço de fase em duas regões $\mathcal{R}_0$ e $\mathcal{R}_1$ satisfazendo $\Pr(x_n \in \mathcal{R}_0) = \Pr(x_n \in \mathcal{R}_1) = 1/2$ (EVANGELISTA et al., 2017). Assim, se $x_n \in \mathcal{R}_0$ então $z_n = 0$, ou se $x_n \in \mathcal{R}_1$ então $z_n = 1$. Métodos de discretização alternativos são propostos na literatura, como por exemplo, em (HUA; ZHOU; ZHOU, 2019).

O valor de x_0 é dado pelos 128 bits da chave original k^0. Esta é dividida em blocos de 16 bytes, $v_1 v_2 \cdots v_{16}$, e calcula-se $v'_0 = \sum_{i=1}^{16} \frac{v_i}{256}$. Então, $x_0 = v'_0 - \lfloor v'_0 \rfloor$, em que $\lfloor \cdot \rfloor$ é a função piso. Gera-se a partir de x_0 uma órbita finita usando (1.1), em que as primeiras amostras são eliminadas devido ao transiente da órbita, e usando-se uma quantização em 2 níveis, gera-se uma sequência binária caótica $\{z_k\}$.

1.4 OBJETIVOS E CONTRIBUIÇÕES DA TESE

Esta tese possui dois objetivos principais:

- Propor algoritmos de cifragem de chaves simétricas com base em uma versão randomizada das S-boxes do algoritmo AES empregando mapas caóticos, a fim de cifrar imagens usando o modo de operação ECB. Estes são menos complexos em relação aos outros modos de operação do AES.
- Propor um novo algoritmo de marca d'água utilizando mapas caóticos e códigos corretores de erro que seja robusto a diferentes ataques.

Nesta tese há cinco contribuições principais:

- Apresenta-se uma nova técnica para adicionar uma fonte de entropia nas S-boxes do AES usando mapas caóticos de uma maneira compatível com o algoritmo AES original.
- Mostra-se que a proposta alcança com menos iterações o mesmo nível de segurança dos modos de operação do AES indicado por uma série de métricas de segurança.
- Apresentam-se técnicas alternativas à camada de difusão do AES que aceleram o processo de cifragem sem comprometer a segurança.
- Investiga-se a robustez dos algoritmos propostos sob a perspectiva da CL e da CD.
- Apresenta-se um novo algoritmo de marca d'água robusto, que utiliza bits caóticos e códigos corretores de erro. Os bits caóticos e os bits de marca d'água formam a mensagem de um código corretor de erro; enquanto os bits de paridade são a parte da palavra código efetivamente inserida na imagem, com a finalidade de obter robustez e segurança. A segurança da marca d'água deve-se ao fato que os bits caóticos só são conhecidos pelos usuários legítimos (por compartilharem uma informação sigilosa). Além disso, como parte da palavra código é conhecida na recepção (no caso, os bits caóticos), a capacidade do código de detecção e correção de erros é empregada exclusivamente na recuperação dos bits de marca d'água; isso aumenta a robustez do método proposto.

1.5 PRODUÇÃO BIBLIOGRÁFICA

Os trabalhos listados a seguir constituem contribuições resultantes do trabalho de pequisa realizado nesta tese.

1.5.1 Artigos Publicados em Periódicos

- J. A. Artiles, D. P. B. Chaves, and C. Pimentel, "Image encryption using block cipher and chaotic sequences", Signal Processing: Image Communication, vol. 79, pp. 24-31, November 2019.
- J. A. Artiles, D. P. B. Chaves, and C. Pimentel, "Linear cryptoanalysis of the simplified AES cipher modified by chaotic sequences", Journal of Communication and Information Systems (JCIS), vol. 34, no. 1,pp. 92-99, April 2019.

1.5.2 Artigos Publicados em Congressos

- J. A. Artiles, D. P. B. Chaves, and C. Pimentel, "Cifragem de imagens usando cifras de bloco e sequências caóticas", in XXXV Simpósio Brasileiro de Telecomunicações (SBrT 2017), Setembro 2017, pp. 1-5.
- J. A. Artiles, D. P. B. Chaves, and C. Pimentel, "Criptoanálise linear do sistema AES simplificado modificado por sequências caóticas", in XXXVI Simpósio Brasileiro de Telecomunicações (SBrT 2018), Setembro 2018, pp. 1-5.
- J. A. Artiles, D. P. B. Chaves, and C. Pimentel, "Sistema de marca d'água em imagens utilizando sequências caóticas", in XXXVII Simpósio Brasileiro de Telecomunicações (SBrT 2019), Setembro 2019, pp. 1-5.

1.6 ORGANIZAÇÃO

Esta tese está organizada em cinco capítulos.

O **Capítulo 1** apresenta a motivação, objetivos e contribuições desta tese.

O **Capítulo 2** apresenta uma introdução ao algoritmo AES, os diferentes modos de operação e modificações propostas com a introdução de uma sequência caótica. Analisam-se os algoritmos propostos mediante diferentes quantificadores de aleatoriedade.

O **Capítulo 3** apresenta um algoritmo AES simplificado, o qual facilita o entendimento das modificações feitas no algoritmo AES original. Analisam-se estes algoritmos com a perspectiva da CL e CD, quantificando o impacto da introdução do caos para estas criptoanálises.

O **Capítulo 4** apresenta um novo algoritmo de geração e inserção de marca d'água que utiliza vários conceitos com a finalidade de obter um algoritmo robusto a diferentes ataques. Compara-se o algoritmo proposto com outros utilizados na literatura.

O **Capítulo 5** apresenta as conclusões e trabalhos futuros.

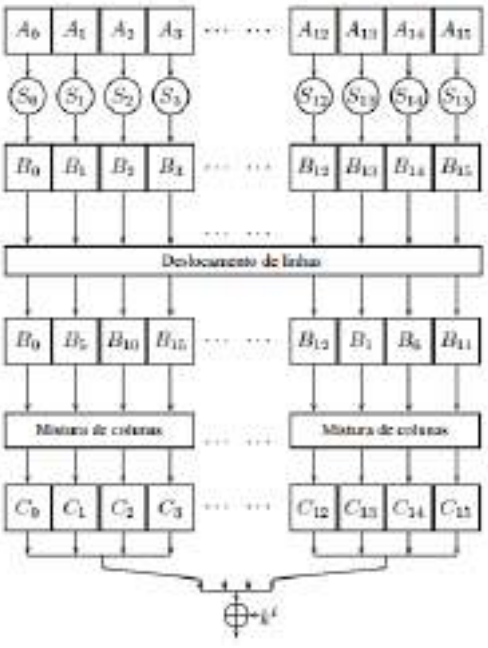
2 CIFRAGEM DE IMAGENS USANDO CIFRAS DE BLOCO E SEQUÊNCIAS CAÓTICAS

Neste capítulo é feita uma modificação no AES com a introdução de uma fonte de entropia em cada S-box (sequência obtida de um mapa caótico). Analisa-se o algoritmo modificado mediante diferentes métricas de segurança utilizadas na literatura e estuda-se a possibilidade da redução da complexidade do algoritmo original com respeito ao algoritmo proposto.

2.1 PRELIMINARES

A estrutura do Algoritmo de Rijndael adotado pelo AES utiliza blocos de informação de 128 bits e chaves de cifragem de 128, 192, 256 bits, variando a quantidade de iterações dependendo do comprimento da chave. Neste trabalho utiliza-se uma chave de 128 bits e 10 iterações (PAAR; PELZL, 2010). O processo de cifragem começa com a conversão do bloco de informação de 128 bits em uma matriz de estado 4×4 em que cada elemento é um byte. De forma similar, a chave também é convertida em uma matriz 4×4. O algoritmo apresenta quatro blocos, como ilustra a Fig. 1. O conjunto de operações realizadas em cada iteração é descrito a seguir.

Figura 1 – Diagrama em blocos do algoritmo AES em uma iteração.



Fonte: O autor (2020).

Substituição

Esta etapa compreende 16 S-boxes idênticas (representadas na Fig. 1 por S_i, $i = 0, \ldots, 15$) que operam de forma paralela, cada uma com 1 byte de entrada e 1 byte de saída.

Em cada S-Box, um byte da matriz de estado é substituído por um byte obtido por operações realizadas no corpo GF(2^8) gerado com o polinômio primitivo $P(x) = x^8 + x^4 + x^3 + x^2 + x + 1$.

Para a i-ésima S-box, a saída B_i está relacionada com a entrada A_i via

$$B_i = S(A_i) = M \cdot A_i^{-1} + N \tag{2.1}$$

em que M é uma matriz de bits 8×8, A_i^{-1} é o inverso multiplicativo de A_i no corpo GF(2^8) e N é um vetor constante não nulo. A operação produto em (2.1) é realizada bit a bit, de modo que A_i^{-1} é empregado como um vetor 8×1.

Deslocamento de linhas

Na i-ésima linha da matriz de estado na saída do bloco de substituição, realiza-se um deslocamento cíclico de i bytes para a direita, $i = 0, 1, 2, 3$. As matrizes de estado antes e depois do deslocamento de linhas são dadas por:

$$\begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_1 & B_5 & B_9 & B_{13} \\ B_2 & B_6 & B_{10} & B_{14} \\ B_3 & B_7 & B_{11} & B_{15} \end{bmatrix}; \quad \begin{bmatrix} B_0 & B_4 & B_8 & B_{12} \\ B_5 & B_9 & B_{13} & B_1 \\ B_{10} & B_{14} & B_2 & B_6 \\ B_{15} & B_3 & B_7 & B_{11} \end{bmatrix}. \tag{2.2}$$

Mistura de colunas

Este processo faz uma transformação linear nas colunas da matriz de estado. Os 4 bytes de cada coluna desta matriz são considerados um vetor que é multiplicado por uma matriz 4×4 que contém elementos inteiros. A multiplicação e adição dos coeficientes é feita em GF(2^8), como é ilustrado a seguir para uma coluna:

$$\begin{bmatrix} C_0 \\ C_1 \\ C_2 \\ C_3 \end{bmatrix} = \begin{bmatrix} 02 & 03 & 01 & 01 \\ 01 & 02 & 03 & 01 \\ 01 & 01 & 02 & 03 \\ 03 & 01 & 01 & 02 \end{bmatrix} \begin{bmatrix} B_0 \\ B_5 \\ B_{10} \\ B_{15} \end{bmatrix}. \tag{2.3}$$

Adição de sub-chave

Nesta operação, a matriz de estado após a mistura de colunas é somada (XOR) com a matriz de sub-chave, formando uma nova matriz de estado usada na próxima iteração. A sub-chave k^i, $i = 1, 2, \ldots, 10$, usada em cada iteração é obtida de um processo realizado a partir da chave k^0. Neste processo, realizam-se várias operações, nas quais encontra-se a utilização de quatro S-boxes.

Em cada iteração realiza-se o mesmo conjunto de operações, exceto na primeira em que a chave k^0 é somada com os dados de informação antes do bloco de substituição, e na décima

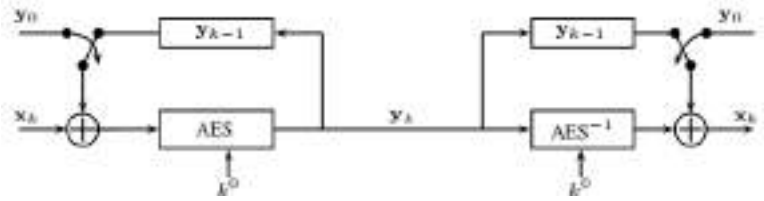
em que o bloco mistura de colunas não é utilizado. Este modo de cifragem é conhecido como modo de operação ECB (*electronic code book*). Com a finalidade de randomizar o processo de cifragem e evitar que blocos de entrada idênticos produzam saídas idênticas, outros modos de operação são apresentados na próxima subseção.

2.1.1 Modos de Operação

Cipher block chaining mode (CBC)

Neste modo, o texto cifrado não depende apenas do bloco de entrada atual, mas depende também dos blocos cifrados anteriores. Sejam $\mathbf{x}_k$ e $\mathbf{y}_k$ o k-ésimo bloco de informação e o bloco cifrado (ambos de 128 bits), respectivamente, para $k = 1, 2, \ldots, N$. O k-ésimo bloco de entrada do AES é $\mathbf{x}_k + \mathbf{y}_{k-1}$ que gera uma saída $\mathbf{y}_k$, em que $\mathbf{y}_0$ é um bloco de inicialização aleatório. Este esquema é mostrado na Fig. 2. O bloco de inicialização deve ser mudado após N blocos.

Figura 2 – Modo de operação CBC.



Fonte: O autor (2020).

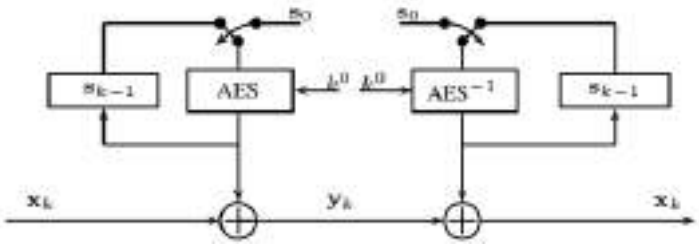
Output feedback mode (OFB)

Este modo utiliza um cifra de blocos para criar uma cifra de fluxo. O bloco cifrado é dado por $\mathbf{y}_k = \mathbf{x}_k + \mathbf{s}_k$, em que $\mathbf{s}_k$ é o bloco de saída do AES quando o bloco de entrada é $\mathbf{s}_{k-1}$, sendo $\mathbf{s}_0$ um bloco de inicialização aleatório, como ilustra a Fig. 3.

Cipher feedback mode (CFB)

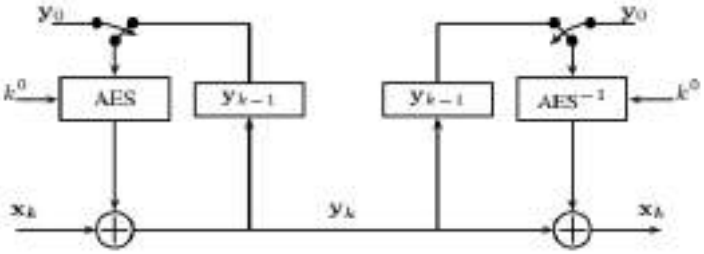
Este modo também utiliza uma cifra de bloco para a construção de uma cifra de fluxo. Seu funcionamento é similar ao modo OFB, mas o bloco de entrada do AES é $\mathbf{y}_{k-1}$, sendo $\mathbf{y}_0$ um bloco de inicialização aleatório, como mostra a Fig. 4.

Figura 3 – Modo de operação OFB.



Fonte: O autor (2020).

Figura 4 – Modo de operação CFB.

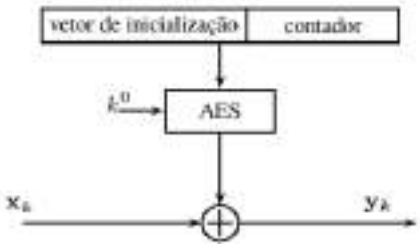


Fonte: O autor (2020).

Counter mode (CTR)

Como no modo anterior, utiliza-se uma cifra de bloco para a construção de uma cifra de fluxo. Um vetor de inicialização de comprimento menor do que o bloco de entrada (por exemplo, de comprimento 96 bits) é concatenado a um contador (CTR) que neste caso tem comprimento 32 bits. Sempre que um bloco é cifrado, o contador é incrementado, mas o vetor de inicialização permanece constante. O número de blocos cifrados até a mudança deste vetor é 2^{32}. O esquema é mostrado na Fig. 5.

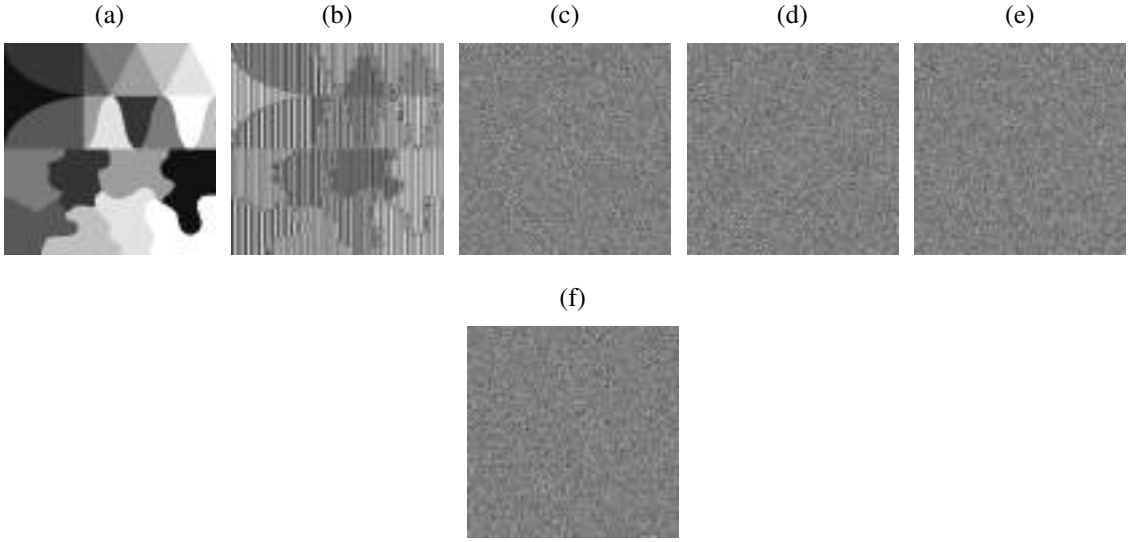
Figura 5 – Modo de operação CRT.



Fonte: O autor (2020).

Os modos de operação descritos nesta subseção (CBC, OFB, CFB, CTR) serão referenciados neste trabalho como modos de operação randomizados. A Fig. 6 mostra os resultados da cifragem de uma imagem em escala de cinza pelos modos de operação com 10 iterações. Com exceção do modo ECB, que mantém padrões da imagem original, os modos randomizados produzem imagens cifradas com aparência ruidosa. A aleatoriedade da imagem cifrada será analisada na próxima seção.

Figura 6 – Imagem original (a). Modo de operação com 10 iterações: ECB (b), CBC (c), OFB (d), CFB (e), CRT (f).



Fonte: O autor (2020).

2.2 MÉTRICAS DE SEGURANÇA

Nesta seção são descritas as métricas de segurança comumente usadas na literatura para avaliar a capacidade de um esquema de cifragem de imagem resistir ataques (TANG et al., 2016; LAN et al., 2018; XU et al., 2016; WANG et al., 2016; ZHENJUN et al., 2016). Neste trabalho, consideram-se imagens monocromáticas de tamanho 512×512 píxeis com $L = 256$ níveis de cinza.

Entropia (H)

Uma medida de aleatoriedade é dada pela entropia da fonte, definida por

$$H = -\sum_{i=1}^{L} \Pr(m_i) \log \Pr(m_i) \tag{2.4}$$

em que $\Pr(m_i)$ é a probabilidade de ocorrer um nível de cinza m_i. Para $L = 256$, o valor máximo da entropia da fonte é 8 bits.

Entropia Local (H)

A entropia em (2.4) é calculada sobre a imagem inteira. Uma métrica alternativa, chamada entropia local, considera a aleatoriedade de blocos da imagem e é definida como (WU et al., 2013)

$$H_{(g,T_B)} = \sum_{i=1}^{g} \frac{H_i}{g} \tag{2.5}$$

em que H_i, $i = 1, \ldots, g$, é a entropia de blocos não sobrepostos com T_B píxeis, selecionados aleatoriamente da imagem. Valores típicos dos parâmetros usados em (WU et al., 2013) são

$g = 30$ e $T_B = 1936$. É fornecido em (WU et al., 2013) um critério estatístico, baseado em teste de hipótese com nível de significância α, para decidir se um algoritmo de cifragem é aprovado neste teste. Isso ocorre se $H_{(30;1936)}$ estiver no intervalo $[7{,}9019; 7{,}9030]$ para $\alpha = 0{,}05$, $[7{,}9017; 7{,}9032]$ para $\alpha = 0{,}01$, $[7{,}9015; 7{,}9034]$ para $\alpha = 0{,}001$.

Ataque diferencial (A-D)

A robustez de um algoritmo de cifragem de imagem em relação a ataques diferenciais é comumente avaliada pelo número de taxa de alteração de píxel (NPCR, *n*umber of changing pixel rate) e pela intensidade de alteração média unificada (UACI, *u*nified average changing intensity).

Sejam C_1 e C_2 duas imagens cifradas de dimensão $W \times H$, tais que suas imagens originais diferem em um píxel. Os valores dos píxeis (variando entre 0 e 255) no ponto (i, j) de C_1 e de C_2 são denotados por $C_1(i, j)$ e $C_2(i, j)$, respetivamente. O NPCR e o UACI entre as imagens C_1 e C_2 são definidos por

$$\text{NPCR} = \frac{\sum_{ij} D(i,j)}{WH} \times 100 \tag{2.6}$$

$$\text{UACI} = \frac{1}{WH} \sum_{ij} \frac{|C_1(i,j) - C_2(i,j)|}{255} \times 100 \tag{2.7}$$

em que $D(i, j)$ é igual a 1 se $C_1(i, j) = C_2(i, j)$, ou igual a 0 se $C_1(i, j) \neq C_2(i, j)$. O valor máximo do NPCR é 100. Neste trabalho, 10.000 testes são realizados com pares de imagens C_1 e C_2 e os valores médios de NPCR e UACI são tabulados. Com base no critério estatístico fornecido em (WU; NOONAN; AGAIAN, 2011), um algoritmo de cifragem é aprovado neste teste (com nível de significância α) se o valor de NPCR for maior que um valor crítico $N^{\star}_{\alpha}$. Para imagens de tamanho 512×512, esse limiar para três valores de α são: $N^{\star}_{0,05} = 99{,}5893$, $N^{\star}_{0,01} = 99{,}5810$, $N^{\star}_{0,001} = 99{,}5717$. Um critério estatístico também é fornecido em (WU; NOONAN; AGAIAN, 2011) para passar no teste UACI. Nesse caso, essa métrica deve estar nos seguintes intervalos: $[33{,}3730; 33{,}5541]$ para $\alpha = 0{,}05$, $[33{,}3445; 33{,}5826]$ para $\alpha = 0{,}01$, $[33{,}3115; 33{,}6156]$ por $\alpha = 0{,}001$.

Sensibilidade à chave (S-C)

Uma boa cifra deve ter alta sensibilidade com respeito a mudanças na chave original k^0. Inverter um único bit desta chave deve produzir a uma imagem cifrada amplamente diferente. O NPCR pode ser usado para validar essa propriedade. Nesse caso, uma imagem original é cifrada com duas chaves que diferem em um bit em uma posição aleatória, gerando duas imagens cifradas C_1 e C_2. Este teste é denotado por k-NPCR. Este teste é realizado para 10.000 valores distintos de k^0.

NIST

A bateria de teste NIST (versão 800-22) ((FIPS), 2001) é usada para testar se uma sequência é adequada para aplicações criptográficas. Esta inclui 15 testes estatísticos com foco em tipos

distintos de não aleatoriedade que podem existir em uma sequência. Os testes são usados para determinar a aceitação ou rejeição da hipótese de aleatoriedade ideal com nível de significância α. Adotamos neste trabalho $\alpha = 0{,}01$, um valor comumente usado em criptografia ((FIPS), 2001). Nas simulações, uma sequência binária representando uma imagem cifrada é a entrada da bateria de teste NIST.

Análise da correlação (A-C)

O coeficiente de correlação ρ mede a correlação entre dois píxeis adjacentes nas direções horizontal, vertical e diagonal. Seleciona-se aleatoriamente J pares de píxeis adjacentes (x_i, y_i) em uma determinada direção (horizontal (Hrt), vertical (Vrt), diagonal (Dgn)) e calcula-se ρ como

$$\rho = \frac{\text{cov}(x, y)}{\sqrt{\text{Var(x)}}\,\sqrt{\text{Var(y)}}}$$

em que

$$\text{cov}(x, y) = \frac{1}{J}\sum_{i=1}^{J}(x_i - \eta_x)\,(y_i - \eta_y)$$

$$\text{Var}(x) = \frac{1}{J}\sum_{i=1}^{J}(x_i - \eta_x)^2$$

e

$$\eta_x = \frac{1}{J}\sum_{i=1}^{J} x_i.$$

Em uma imagem altamente correlacionada, os valores de ρ estão próximos de 1, enquanto o valor ideal desse parâmetro para uma imagem cifrada é 0. Considera-se neste trabalho $J = 10.000$.

Utilizam-se 141 imagens originais do banco de dados de imagens USC-SIPI (http://sipi.usc.edu/database). Esse banco de dados contém imagens coloridas e em escala de cinza de três tamanhos distintos (256×256 píxeis, 512×512 píxeis, 1024×1024 píxeis). Algumas imagens são redimensionadas para criar um novo banco de dados que contenha imagens de mesmo tamanho 512×512 píxeis. Além disso, algumas imagens coloridas são convertidas em imagens em escala de cinza com 256 níveis de cinza. Em resumo, todas as simulações usam um banco de dados com 141 imagens em escala de cinza de tamanho 512×512 píxeis no formato bmp (cada imagem de tamanho 768,1 k bytes).

A Tabela 1 mostra os valores mínimos e máximos de cada métrica de segurança para todas as imagens no banco de dados para cada modo de operação AES com 10 iterações, importante ressaltar que no análise da correlação mostra-se o valor do módulo do coeficiente de correlação. O valor mínimo da entropia para o modo ECB não alcança um valor próximo de 8 bits, mesmo na décima iteração e uma correlação significativa entre os píxeis adjacentes também é observada. Para os modos de operação randomizados, há uma melhoria significativa nos valores dessas

Tabela 1 – Valores mínimos e máximos das métricas de segurança para os modos de operação do algoritmo AES.

Modos de Operação	Entropia		Análise da Correlação					
			Hrt		Vrt		Dgn	
	Min	Max	Min	Max	Min	Max	Min	Max
ECB	6,3872	7,9999	0,0212	0,3146	0,0189	0,2973	0,0218	0,3208
CBC	7,9982	7,9999	0,0109	0,0214	0,0093	0,0228	0,0115	0,0191
OFB	7,9958	7,9999	0,0116	0,0249	0,0113	0,0218	0,0111	0,0276
CFB	7,9979	7,9999	0,0109	0,01498	0,0111	0,01839	0,0097	0,0204
CTR	7,9988	7,9999	0,0087	0,0107	0,0084	0,0119	0,0087	0,0138

Modos de Operação	NPCR		UACI		k-NPCR		Entropia Local	
	Min	Max	Min	Max	Min	Max	Min	Max
ECB	0,003	0,006	0,0022	0,0035	99,578	99,591	6,8860	7,9025
CBC	99,599	99,617	33,395	33,503	99,598	99,606	7,9020	7,9029
OFB	99,589	99,611	33,382	33,445	99,589	99,606	7,9020	7,9030
CFB	99,590	99,609	33,383	33,443	99,589	99,604	7,9020	7,9030
CTR	99,601	99,621	33,397	33,509	99,601	99,611	7,9020	7,9030

Fonte: O autor (2020).

métricas e todas as imagens passam no teste NIST. Além disso, os valores mínimos de NPCR, UACI, k-NPCR e entropia local para esses modos passam nos testes correspondentes de acordo com o critério estatístico descrito nesta seção. Os melhores resultados são obtidos com o modo CTR.

2.3 NOVOS ALGORITMOS PROPOSTOS UTILIZANDO MAPAS CAÓTICOS

Os modos de operação randomizados utilizam um vetor de inicialização como fonte de aleatoriedade para quebrar o comportamento determinístico do modo de operação ECB. Nesta seção a fonte de aleatoriedade é obtida a partir da iteração de mapas caóticos com a introdução dos bits caóticos em diferentes etapas do algoritmo AES, mantendo-se o modo de operação ECB.

2.3.1 AES Modificado com o Mapa Caótico

Nesta seção é apresentado o algoritmo AES1, que é idêntico ao AES exceto pela soma na saída das S-boxes de um byte gerado a partir de um mapa caótico. Apresenta-se esta proposta com 3 bits caóticos por S-box, podendo ser empregados até 8 bits (uma melhoria nos indicadores de aleatoriedade e robustez contra criptoanálise pode ser obtida com o aumento deste número de bits).

Para cada bloco de informação, a sequência binária caótica $\{z_k\}$ é seccionada em 60 bits, em que os primeiros 48 bits são usados nas 16 S-boxes da etapa de substituição, enquanto os 12 bits restantes são usados nas 4 S-boxes da unidade de geração de sub-chaves. Para cada

S-box, utilizam-se três bits caóticos (z_j, z_{j+1}, z_{j+2}), com representação polinomial dada por $c(x) = z_j x^2 + z_{j+1} x + z_{j+2}$.

Como as operações de cada S-box são definidas em GF(2^8), multiplica-se $c(x)$ por um polinômio primitivo $p(x)$ em GF(2^5), obtendo-se assim o polinômio $h(x) = c(x)p(x) \mathrm{mod} P(x)$, em que $P(x)$ é apresentado na Seção 2.1. Os coeficientes deste polinômio formam um byte denominado de **h**. A escolha de $p(x)$ é feita para que **h** seja o mais balanceado possível entre zeros e uns. Neste trabalho, o polinômio $p(x) = x^5 + x^4 + x^3 + x + 1$ foi escolhido via teste exaustivo. Este byte é somado (XOR bit a bit) com o byte da saída da S-box na etapa de substituição de bytes. Na segunda iteração, os 48 bits caóticos obtidos na primeira iteração são utilizados novamente fazendo-se um deslocamento cíclico à direita dado pelo valor na base dez dos últimos três bits utilizados na última S-box. Por exemplo, se o polinômio da última S-box é $c(x) = x^2 + x + 1$, faz-se um deslocamento de 7 bits na sequência de 48 bits obtida do mapa caótico. Este procedimento é repetido em cada iteração. O deslocamento é feito para evitar a rigidez das S-boxes, que levam a um comportamento determinístico, mantendo os mesmos 48 bits caóticos por bloco de informação. Na obtenção das sub-chaves, um procedimento similar é realizado nas S-boxes usadas para obtenção das sub-chaves k^i, $i = 1, \ldots, 10$.

O desempenho do algoritmo AES1 (valores mínimos e máximos das métricas de segurança para todas as imagens no banco de dados) é mostrado na Tabela 2 para cada iteração. Os valores mínimos de NPCR, UACI e k-NPCR após a segunda iteração são maiores que os valores críticos e isso ocorre para a entropia local após a terceira iteração. Então, o AES1 passa nesses testes após a terceira iteração. Esses resultados indicam que o algoritmo AES1 é altamente sensível a pequenas alterações na imagem original ou na chave original. Observa-se que após a sétima iteração os valores de entropia e coeficiente de correlação alcançam os valores correspondentes alcançados pelo modo de operação CTR na Tabela 1. Esta iteração é destacada em negrito nesta tabela. Esta é uma indicação de que o algoritmo proposto é robusto contra ataques estatísticos. Além disso, todas as imagens são aprovadas na bateria de teste NIST após a primeira iteração.

Concluímos que as modificações propostas nas S-boxes com a inclusão de bits caóticos (tanto no caminho de dados quanto na geração de sub-chaves) permitem alcançar bons indicadores de aleatoriedade e robustez contra criptoanálise com poucas iterações. A Fig. 7 mostra o histograma e o coeficiente de correlação em cada direção de uma imagem original e da imagem cifrada pelo algoritmo AES1 com sete iterações. Uma inspeção visual revela que o histograma da imagem cifrada é uniformemente distribuído e o AES1 quebra a alta correlação entre os píxeis adjacentes da imagem original.

Tabela 2 – Valores mínimos e máximos das métricas de segurança para o AES1 por iteração com mapa cúbico.

Iterações	Entropia		Análise da Correlação					
			Hrt		Vrt		Dgn	
	Min	Max	Min	Max	Min	Max	Min	Max
1	7,9538	7,9876	0,1012	0,1424	0,0861	0,1652	0,0871	0,1714
2	7,9826	7,9968	0,0547	0,1048	0,0343	0,0741	0,0311	0,0891
3	7,9867	7,9985	0,0223	0,0424	0,0237	0,0391	0,0196	0,0351
4	7,9868	7,9989	0,0165	0,0283	0,0163	0,0275	0,0147	0,0287
5	7,9954	7,9995	0,0128	0,0211	0,0117	0,0197	0,0109	0,0203
6	7,9977	7,9999	0,0101	0,0184	0,0097	0,0174	0,0093	0,1056
7	**7,9988**	**7,9999**	**0,0095**	**0,0112**	**0,0092**	**0,0101**	**0,0088**	**0,0098**
8	7,9989	7,9999	0,0077	0,0094	0,0085	0,0091	0,0082	0,0083
9	7,9999	7,9999	0,0071	0,0088	0,0075	0,0078	0,0075	0,0076
10	7,9999	7,9999	0,0053	0,0081	0,0064	0,0072	0,0058	0,0069

Iterações	NPCR		UACI		k-NPCR		Entropia Local	
	Min	Max	Min	Max	Min	Max	Min	Max
1	99,393	99,598	33,212	33,482	99,589	99,607	7,8697	7,9010
2	99,592	99,618	33,421	33,498	99,598	99,612	7,9017	7,9024
3	99,601	99,621	33,424	33,510	99,599	99,612	7,9019	7,9025
4	99,609	99,621	33,431	33,510	99,604	99,613	7,9020	7,9026
5	99,611	99,622	33,431	33,511	99,605	99,616	7,9020	7,9029
6	99,611	99,621	33,432	33,512	99,605	99,618	7,9020	7,9029
7	**99,612**	**99,622**	**33,434**	**33,514**	**99,607**	**99,619**	**7,9020**	**7,9030**
8	99,612	99,622	33,435	33,517	99,608	99,619	7,9020	7,9030
9	99,612	99,622	33,435	33,518	99,608	99,620	7,9020	7,9030
10	99,612	99,623	33,437	33,521	99,609	99,622	7,9020	7,9030

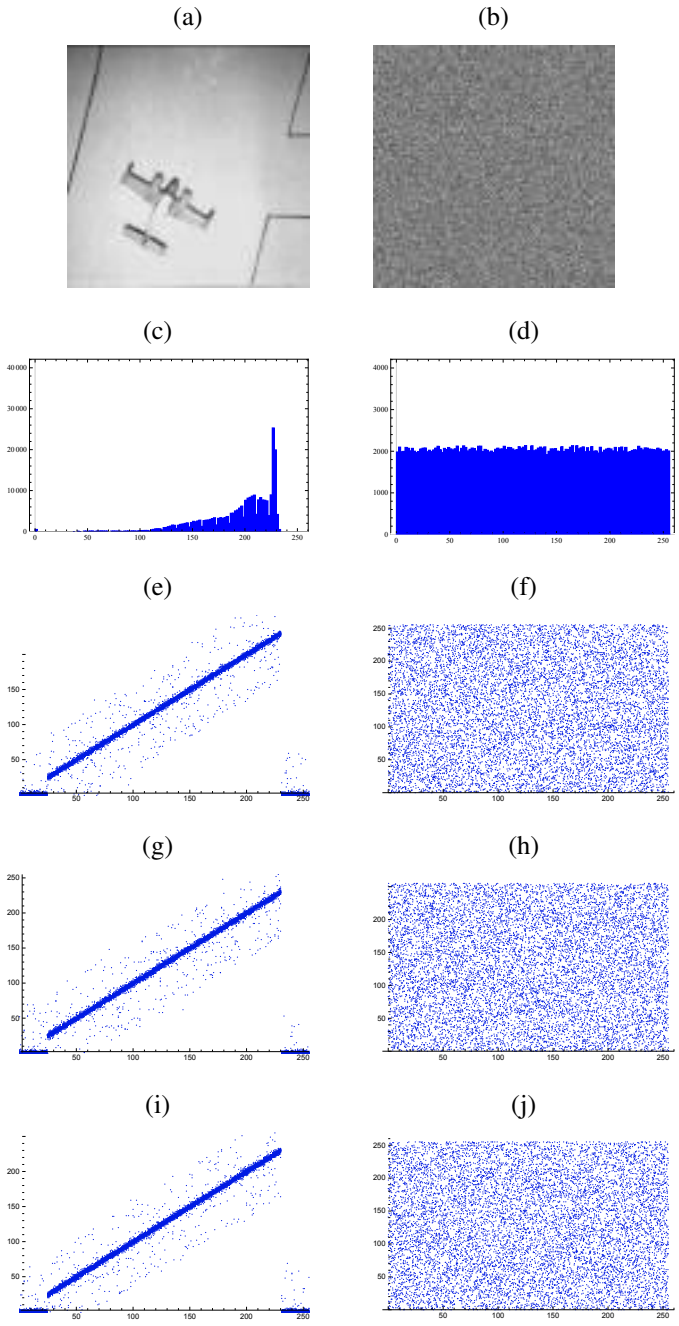
Fonte: O autor (2020).

2.3.2 Algoritmo AES2

A introdução de incerteza no sistema pelos bits caóticos permite simplificar o algoritmo AES, visando reduzir sua complexidade. Nesta subseção, a qualidade da imagem cifrada é analisada quando o bloco mistura de colunas é eliminado. Esse bloco é responsável pela difusão de bits e exige um número elevado de operações de soma módulo dois. Denominaremos por AES2 esta variação do AES1.

O número de operações deste bloco pode ser contabilizado a partir de (2.3) ((FIPS), 2001). A multiplicação por 01, produz na saída o mesmo valor da entrada. A multiplicação por 02 corresponde a fazer um deslocamento na sequência de entrada e se o bit mais significativo de B_i é 1, faz-se uma soma (XOR) com a sequência 0001 1011. A multiplicação por 03 é equivalente a fazer uma soma (XOR) dos valores obtidos nas duas multiplicações anteriores. Na obtenção de cada C_i, $i = 1 \ldots 4$, são feitas 160 somas (XOR) em média. Em uma iteração existem quatro operações de mistura de colunas, então existem 640 somas (XOR) para a implementação deste

Figura 7 – Imagem original (avião) (a), imagem cifrada (b), histograma da imagem original (c), histograma da imagem cifrada (d), correlação horizontal da imagem original (e), correlação horizontal da imagem cifrada (f), correlação vertical da imagem original (g), correlação vertical da imagem cifrada (h), correlação diagonal da imagem original (i), correlação diagonal da imagem original (j).



Fonte: O autor (2020).

bloco por iteração.

A Tabela 3 apresenta os resultados dos quantificadores para o algoritmo AES2 em função do número de iterações. Observa-se que com 9 iterações os quantificadores para o algoritmo AES2 apresentam valores similares ao do algoritmo AES1 com 9 iterações, indicando a viabilidade deste algoritmo.

Tabela 3 – Valores mínimos e máximos das métricas de segurança para o AES2 por iteração com mapa cúbico.

Iterações	Entropia		Análise da Correlação					
			Hrt		Vrt		Dgn	
	Min	Max	Min	Max	Min	Max	Min	Max
1	7,0087	7,4015	0,1972	0,2287	0,1937	0,2144	0,2004	0,2291
2	7,6270	7,9517	0,1007	0,1274	0,1074	0,1300	0,1028	0,1270
3	7,8023	7,9985	0,0913	0,1170	0,0897	0,1089	0,0901	0,1102
4	7,9270	7,9989	0,0891	0,1008	0,0795	0,1091	0,0853	0,1013
5	7,9495	7,9992	0,0771	0,0999	0,0801	0,0998	0,0792	0,0962
6	7,9723	7,9994	0,0428	0,0671	0,03924	0,0700	0,0408	0,0599
7	7,9848	7,9995	0,0107	0,0204	0,0110	0,0237	0,0106	0,0226
8	7,9901	7,9999	0,0100	0,0187	0,0101	0,0199	0,0099	0,01907
9	**7,9987**	**7,9999**	**0,0096**	**0,0102**	**0,0084**	**0,0099**	**0,0080**	**0,0100**
10	7,9998	7,9999	0,0059	0,0093	0,0064	0,0088	0,0060	0,0089

Iterações	NPCR		UACI		k-NPCR		Entropia Local	
	Min	Max	Min	Max	Min	Max	Min	Max
1	99,287	99,381	33,108	33,291	99,582	99,595	7,7290	7,9009
2	99,432	99,468	33,183	33,302	99,588	99,599	7,8471	7,9018
3	99,501	99,588	33,297	33,336	99,590	99,603	7,9011	7,9019
4	99,591	99,619	33,380	33,385	99,597	99,610	7,9015	7,9025
5	99,600	99,620	33,399	33,480	99,601	99,616	7,9019	7,9026
6	99,604	99,620	33,411	33,596	99,604	99,615	7,9020	7,9028
7	99,610	99,621	33,432	33,503	99,605	99,618	7,9020	7,9028
8	99,611	99,621	33,433	33,509	99,607	99,619	7,9020	7,9029
9	**99,611**	**99,622**	**33,438**	**33,515**	**99,608**	**99,622**	**7,9021**	**7,9030**
10	99,612	99,622	33,438	33,520	99,608	99,622	7,9021	7,9030

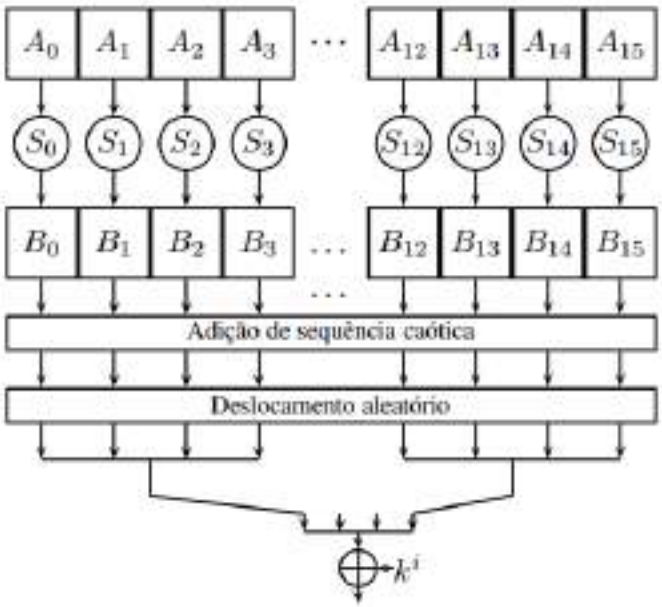
Fonte: O autor (2020).

2.3.3 Algoritmo AES3

O bloco de mistura de colunas do algoritmo AES realiza uma difusão dos dados, fazendo uma mistura dos bits de saída de quatro S-boxes. A eliminação deste bloco implica no aumento do número de iterações do algoritmo AES2 com respeito ao algoritmo AES1 devido à redução da capacidade de difusão de bits provenientes de S-boxes distintas. Introduz-se nesta seção outra variação do AES1, denominada AES3, mostrada na Fig. 8. Nesse algoritmo, os blocos deslocamento de linhas e mistura de colunas são substituídas por um novo bloco, denominado de mistura aleatória. Em cada iteração, este bloco realiza um deslocamento cíclico à esquerda nos 128 bits obtidos após a soma (XOR) com bits caóticos em m posições ($m = 0, \ldots, 127$), em que m é o valor na base dez dos primeiros 7 bits caóticos dos 48 bits utilizados no caminho de dados. A Tabela 4 apresenta os resultados das métricas de segurança para o algoritmo AES3 como uma função do número de iterações. O AES3 com 6 iterações apresenta métricas com valores semelhantes aos obtidos pelo algoritmo AES1 com 7 iterações, indicando a viabilidade

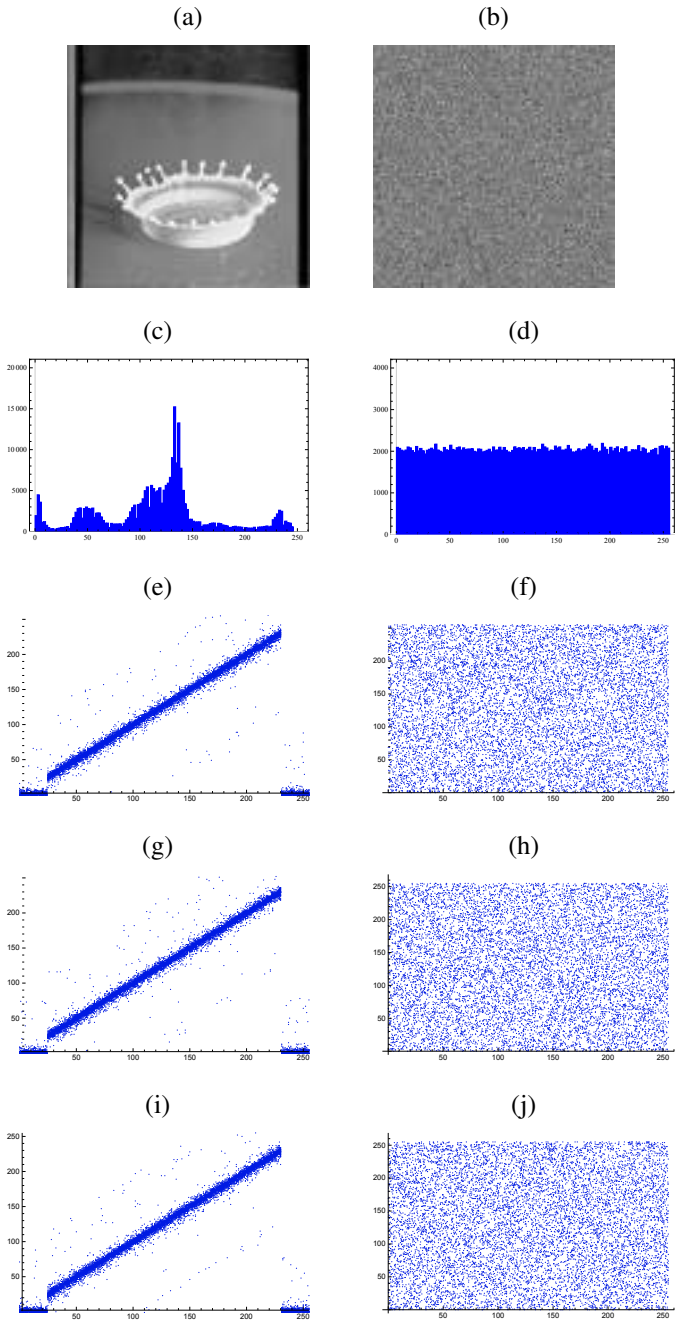
do deste algoritmo. Uma comparação entre o histograma e os coeficientes de correlação para as imagens originais e cifradas é fornecida na Fig. 9 para o AES3 com 6 iterações.

Figura 8 – Diagrama em blocos do algoritmo AES3 por iteração.



Fonte: O autor (2020).

Figura 9 – Imagem original (pingo) (a), imagem cifrada (b), histograma da imagem original (c), histograma da imagem cifrada (d), correlação horizontal da imagem original (e), correlação horizontal da imagem cifrada (f), correlação vertical da imagem original (g), correlação vertical da imagem cifrada (h), correlação diagonal da imagem original (i), correlação diagonal da imagem original (j).



Fonte: O autor (2020).

2.3.4 Tempo de Simulação

Comparamos nesta subseção o tempo necessário para cifrar imagens de tamanho 512×512 em escala de cinza para os algoritmos considerados neste trabalho. Estes são implementados com o programa Mathematica v.10 no sistema operacional Linux, executado em um computador pessoal com

Tabela 4 – Valores mínimos e máximos das métricas de segurança para o algoritmo AES3 por iteração com mapa cúbico.

Iterações	Entropia		Análise da Correlação					
			Hrt		Vrt		Dgn	
	Min	Max	Min	Max	Min	Max	Min	Max
1	7,9534	7,9164	0,0527	0,0976	0,0485	0,0956	0,0556	0,0824
2	7,9746	7,9868	0,0256	0,0424	0,0244	0,0417	0,0234	0,0437
3	7,9899	7,9938	0,0152	0,0182	0,0122	0,0173	0,0142	0,0168
4	7,9965	7,9989	0,0105	0,0118	0,0098	0,0122	0,0108	0,0113
5	7,9982	7,9993	0,0076	0,0099	0,0087	0,0101	0,0099	0,0109
6	**7,9986**	**7,9999**	**0,0079**	**0,0092**	**0,0084**	**0,0098**	**0,0074**	**0,0095**
7	7,9997	7,9999	0,0068	0,0085	0,0068	0,0093	0,0067	0,0086
8	7,9999	7,9999	0,0057	0,0077	0,0058	0,0079	0,0059	0,0081
9	7,9999	7,9999	0,0054	0,0073	0,0056	0,0074	0,0058	0,0069
10	7,9999	7,9999	0,0050	0,0058	0,0051	0,0064	0,0056	0,0059
Iterações	NPCR		UACI		k-NPCR		Entropia Local	
	Min	Max	Min	Max	Min	Max	Min	Max
1	99,585	99,609	33,384	33,443	99,602	99,611	7,8790	7,9019
2	99,607	99,617	33,388	33,456	99,611	99,617	7,9019	7,9021
3	99,610	99,621	33,418	33,481	99,612	99,617	7,9020	7,9026
4	99,613	99,622	33,433	33,450	99,612	99,617	7,9020	7,9028
5	99,614	99,625	33,442	33,451	99,613	99,619	7,9020	7,9029
6	**99,614**	**99,628**	**33,446**	**33,453**	**99,614**	**99,620**	**7,9020**	**7,9030**
7	99,615	99,628	33,449	33,455	99,614	99,624	7,9020	7,9030
8	99,616	99,629	33,449	33,455	99,615	99,630	7,9021	7,9030
9	99,616	99,630	33,449	33,458	99,615	99,631	7,9022	7,9030
10	99,618	99,634	33,503	33,458	99,617	99,633	7,9022	7,9030

Fonte: O autor (2020).

- Processador: 8 X Intel(R) Xeon(R) CPU E5-1620 v2 3.70 GHz.
- Memória RAM: 64GB.
- Placa de Video: NVIDIA GK-107GL.

As imagens de teste são todas as 141 imagens no banco de dados descritas em Subseção 2.2. A implementação é sequencial e nenhum processamento paralelo é executado. Uma tabela armazena todos os valores possíveis de entrada e saída de uma S-box, realizando um mapeamento entre a entrada A_i e a saída correspondente B_i. A sequência binária caótica usada nos algoritmos AES1, AES2 e AES3 é gerada e armazenada em um vetor.

A Tabela 5 mostra o tempo médio de cifragem (em segundos), bem como a taxa (em MB/s) de cada algoritmo. Os algoritmos AES1, AES2 e AES3 são implementados com 7, 9 e 6 iterações, respectivamente. Os modos de operação do AES têm 10 iterações, conforme especificado pelo padrão. O algoritmo AES3, que usa o bloco de mistura aleatória com um

Tabela 5 – Comparação do tempo de cifragem dos diferentes algoritmos.

Algoritmo	Tempo (s)	MB/s
AES1	10,42	0,0738
AES2	7,21	0,1067
AES3	4,02	0,1419
CBC	15,41	0,0499
OFB	14,08	0,0546
CBC	14,01	0,0549
CTR	12,52	0,0614

Fonte: O autor (2020).

número reduzido de iterações no modo ECB, é o algoritmo mais rápido. Observa-se que o algoritmo mais simples (AES2), que não usa o bloco mistura de colunas, apresenta um tempo de simulação menor quando comparado com o AES1, mesmo tendo o AES2 um número maior de iterações.

2.3.5 Análise da Sensibilidade

A redução do número de iterações implica uma diminuição da difusão e confusão de bits, fazendo o sistema menos sensível a variações dos bits de entrada e dos bits do algoritmo de geração de chave. É desejável que os algoritmos de cifra de bloco apresentem um comportamento pseudo-aleatório. Este comportamento é evidente se, quando uma entrada for alterada ligeiramente (por exemplo, mudando-se um único bit do texto claro), a saída muda significativamente (por exemplo, cada bit de saída apresenta a mesma probabilidade de ser 0 ou 1). No caminho de dados, este comportamento pode ser observado na entrada e saída das S-boxes (único elemento não linear). No algoritmo de geração de sub-chaves, pode-se analisar o efeito da modificação de um bit na chave original.

Sensibilidade da geração de sub-chaves

É importante analisar a sensibilidade do processo de geração de sub-chaves a uma pequena variação da chave original. Dada uma chave original k^0 e as sub-chaves correspondentes $k^1, \ldots, k^{10}$, seja $\hat{k}^0$ uma chave que difere de k^0 por um único bit e as sub-chaves correspondentes são $\hat{k}^1, \cdots, \hat{k}^{10}$. Seleciona-se aleatoriamente 10.000 chaves originais e calcula-se quantos bits são alterados entre k^i e $\hat{k}^i$ na $i-$ésima iteração do algoritmo. O valor médio é mostrado na Tabela 6. Para uma boa difusão, este valor deve ficar próximo a 50% em cada iteração. Para o algoritmo AES, esse número converge para $49,99\%$ na sétima iteração, enquanto o algoritmo proposto alcança esse nível de difusão na segunda iteração, devido à sequência binária caótica usada na unidade de geração de sub-chaves. Convém destacar que o processo de geração de sub-chaves é o mesmo para os algoritmos propostos (AES1, AES2, AES3). Conclui-se então que a redução do número de iterações dos algoritmos propostos não acarreta uma perda na sensibilidade de geração de sub-chaves em virtude da utilização de bits caóticos no processo de

Tabela 6 – Sensibilidade no caminho de geração de sub-chaves.

Iteração	AES	AES1
1	8,8927	49,9378
2	24,3097	49,9921
3	28,6164	49,9938
4	36,7991	49,9958
5	39,9431	49,9967
6	49,9385	49,9981
7	49,9917	49,9984
8	49,9924	49,9985
9	49,9956	49,9991
10	49,9962	49,9992

Fonte: O autor (2020).

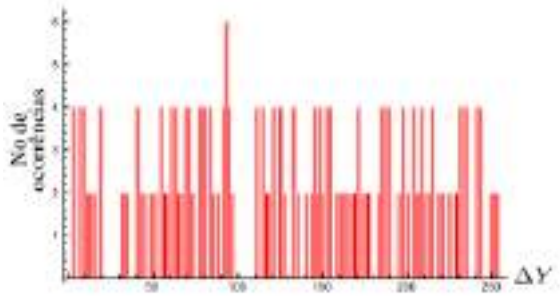
obtenção das sub-chaves.

Sensibilidade no caminho de dados

A ideia básica utilizada é similar à criptoanálise diferencial, que é tabular diferenças específicas na entrada de uma S-box que levam a diferenças específicas na sua saída com probabilidade maior do que seria esperado para uma permutação aleatória, em que ocorrem todas as possíveis saídas com a mesma probabilidade. Define-se a diferença ΔX entre duas entradas de uma S-Box A_i e A'_i (sequências de 8 bits), como $\Delta X = A_i \oplus A'_i$, em que $\oplus$ denota adição em GF(2). Com o uso da chave k^0, obtém-se uma diferença na saída $\Delta Y = S(A_i) \oplus S(A'_i)$, em que $S(\cdot)$ é a função que define a permutação de entrada e saída da S-Box, com uma probabilidade $P(\Delta Y)$. Para uma cifra de bloco fraca, podem haver diferenças nas saídas que ocorrem com alta probabilidade enquanto algumas diferenças podem não ocorrer. Este comportamento pode ser explorado para diminuir o espaço de chaves do sistema (PAAR; PELZL, 2010).

Fixando um valor de A_i e variando todos os possíveis 256 valores de A'_i calculam-se os possíveis ΔY, como mostra a Fig. 10. Observa-se que só ocorrem aproximadamente a metade dos possíveis valores de ΔY. Este comportamento é similar para qualquer valor de A_i fixado. Para um valor de entrada A_i fixado, a partir das probabilidades $P(\Delta Y)$ obtidas a partir da Fig. 10 usa-se (2.4) e obtém-se uma entropia igual a 6,86 bits (valor similar para qualquer A_i fixado).

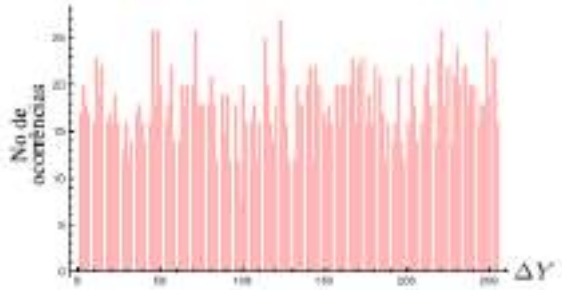
Figura 10 – Sensibilidade à mudança de um bits na entrada da S-box do AES.



Fonte: O autor (2020).

No algoritmo AES1, introduz-se o mapa caótico nas S-boxes. Nesse caso, para um determinado par de entrada A_i e A'_i, existem oito possíveis valores de ΔY, com uma probabilidade de ocorrência similar, devido aos oito possíveis polinômios $h(x)$. Agora, para cada valor de A_i, deve-se calcular todos os possíveis valores de ΔY para todos os valores de A'_i e $h(x)$. A Fig. 11 mostra que todos os possíveis valores de ΔY ocorrem e têm probabilidades mais equânimes do que o mostrado na Fig. 10. Para um valor de entrada A_i, a partir das probabilidades $P(\Delta Y)$, obtém-se uma entropia igual a 7,86 bits. O emprego do mapa caótico leva a um aumento da entropia, visto que para cada entrada das S-boxes existem oito possíveis saídas, aumentando o número de possíveis diferenças ΔY e menor diferença nos valores de $P(\Delta Y)$.

Figura 11 – Sensibilidade à mudança de um bits na entrada da S-box do AES1.

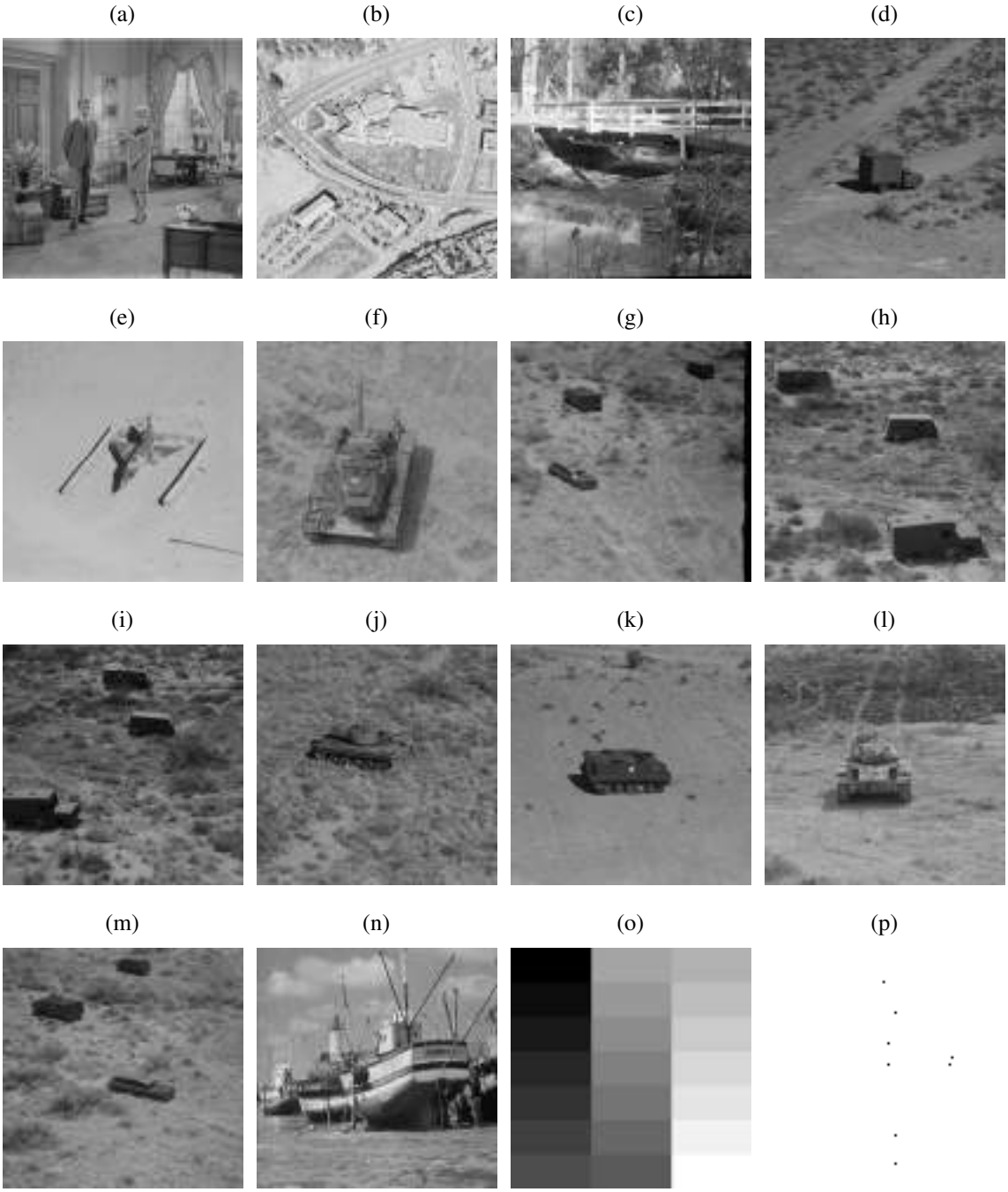


Fonte: O autor (2020).

2.4 COMPARAÇÃO DE RESULTADOS

Para uma comparação com alguns algoritmos de cifragem existentes, consideram-se 16 imagens em escala cinza do banco de dados USC-SIPI, utilizando o grupo *Miscellaneous*, em que cada imagem tem tamanho 512×512 píxeis, mostradas na Fig. 12. As Tabelas 7 e 8 mostram os valores de NPCR e UACI obtidos para o algoritmo AES1 com 7 iterações para cada imagem usando três mapas caóticos definidos na Seção **??** (CM, MSL com $\alpha = 4$, TLC com $r = 0{,}5$) bem como os resultados dos algoritmos propostos em (HUA; ZHOU; HUANG, 2019) e (WANG et al., 2018) para as mesmas imagens. Pode-se observar que o AES1 passa nos testes NPCR e UACI para os três mapas e para as 16 imagens (os limiares de NPCR e UACI são calculados com nível de significância α igual a $0{,}05$, significando que NPCR $\geq 99{,}5893$, UACI $\in [33{,}3730; 33{,}5541]$). Para algumas imagens, o algoritmo AES1 apresenta valores desta métrica superiores que os obtidos pelas cifras utilizadas nestas comparações. Utilizando os mesmos parâmetros pode-se observar o comportamento da entropia local na Tabela 9. Observa-se que o algoritmo proposto em (ALAWIDA et al., 2019) não passa neste teste para algumas imagens (destacado em negrito nesta tabela).

Figura 12 – 5.2.08 (a), 5.2.09 (b), 5.2.10 (c), 7.1.01 (d), 7.1.02 (e), 7.1.03 (f), 7.1.04 (g), 7.1.05 (h), 7.1.06 (i), 7.1.07 (j), 7.1.08 (k), 7.1.09 (l), 7.1.10 (m), Boat.512 (n), Gray21.512 (o), Ruler.512 (p).



Fonte: O autor (2020).

Tabela 7 – Valores de NPCR do grupo USC-SIPI “Miscellaneous”.

Imagem	CM	MSL	TLC	(HUA; ZHOU; HUANG, 2019)	(WANG et al., 2018)
5.2.08	99,6212	99,6107	99,6201	99,6250	99,5934
5.2.09	99,6119	99,6203	99,6134	99,6292	99,6002
5.2.10	99,6189	99,6205	99,6244	99,6212	99,6365
7.1.01	99,6244	99,6193	99,6127	99,6208	99,6147
7.1.02	99,6220	99,6117	99,6270	99,6025	99,6075
7.1.03	99,6142	99,6255	99,6119	99,6078	99,6048
7.1.04	99,6121	99,6140	99,6281	99,6082	99,6162
7.1.05	99,6091	99,6239	99,6153	99,6014	99,6326
7.1.06	99,6257	99,6131	99,6188	99,6063	99,6086
7.1.07	99,6150	99,6014	99,6248	99,5964	99,6185
7.1.08	99,6202	99,6240	99,6077	99,5953	99,6140
7.1.09	99,6106	99,6287	99,6212	99,6094	99,6002
7.1.10	99,6162	99,6077	99,6127	99,6078	99,6174
Boat.512	99,6113	99,6015	99,6099	99,6181	99,6151
Gray.512	99,6185	99,6107	99,6129	99,6029	99,5998
Ruler.512	99,6188	99,6177	99,6150	99,6033	99,6181

Fonte: O autor (2020).

Tabela 8 – Valores de UACI do grupo USC-SIPI “Miscellaneous”.

Imagem	CM	MSL	TLC	(HUA; ZHOU; HUANG, 2019)	(WANG et al., 2018)
5.2.08	33,4410	33,4025	33,4528	33,4973	33,4642
5.2.09	33,4873	33,4806	33,5103	33,4778	33,5235
5.2.10	33,5014	33,5439	33,4621	34,4327	33,4638
7.1.01	33,3974	33,3927	33,4981	33,4154	33,4310
7.1.02	33,4715	33,4208	33,4015	33,4694	33,5172
7.1.03	33,4823	33,5119	33,4717	33,4632	33,4884
7.1.04	33,5319	33,4890	33,5021	33,4996	33,4156
7.1.05	33,4748	33,3914	33,4691	33,4647	33,4123
7.1.06	33,4509	33,4821	33,4770	33,4416	33,5102
7.1.07	33,3918	33,3872	33,4938	33,3906	33,4759
7.1.08	33,4782	33,4753	33,5092	33,4029	33,4500
7.1.09	33,4431	33,4392	33,4429	33,4686	33,4552
7.1.10	33,5211	33,4827	33,4632	33,4434	33,4374
Boat.512	33,4928	33,4718	33,4498	33,4472	33,4068
Gray.512	33,4701	33,4792	33,4972	33,4781	33,4598
Ruler.512	33,5017	33,5452	33,5289	33,3883	33,5389

Fonte: O autor (2020).

Tabela 9 – Valores da entropia local do grupo USC-SIPI "Miscellaneous". Os números em negrito não passam no teste de entropia local para $\alpha = 0,05$, ou seja, eles não estão dentro do intervalo [7,9019; 9,9030].

File Name	MC	MSL	TLC	(ALAWIDA et al., 2019)	(WANG et al., 2018)
5.2.08	7,9027	7,9024	7,9025	7,9025	7,9020
5.2.09	7,9026	7,9024	7,9028	7,9027	7,9023
5.2.10	7,9030	7,9028	7,9022	7,9022	7,9030
7.1.01	7,9027	7,9030	7,9028	7,9027	7,9022
7.1.02	7,9023	7,9024	7,9028	**7,8935**	7,9028
7.1.03	7,9029	7,9026	7,9027	**7,9007**	7,9020
7.1.04	7,9024	7,9028	7,9027	7,9022	7,9022
7.1.05	7,9027	7,9022	7,9026	7,9022	7,9025
7.1.06	7,9029	7,9024	7,9030	7,9030	7,9023
7.1.07	7,9030	7,9028	7,9027	7,9028	7,9028
7.1.08	7,9026	7,9028	7,9029	7,9024	7,9024
7.1.09	7,9020	7,9026	7,9022	7,9027	7,9030
7.1.10	7,9026	7,9023	7,9024	7,9027	7,9028
Boat.512	7,9029	7,9025	7,9022	7,9025	7,9024
Gray.512	7,9023	7,9029	7,9025	**7,8871**	7,9029
Ruler.512	7,9028	7,9022	7,9029	**7,8987**	7,9028

Fonte: O autor (2020).

3 CRIPTOANÁLISE LINEAR E DIFERENCIAL DO SISTEMA AES SIMPLIFICADO MODIFICADO POR SEQUÊNCIAS CAÓTICAS

Neste capítulo faz-se um análise dos algoritmos propostos no Capítulo 2 sob a perspectiva da criptoanálise linear (CL) e criptoanálise diferencial (CD). Para um melhor entendimento, estuda-se um algoritmo simplificado para mostrar o impacto da inserção dos bits caóticos no algoritmo original.

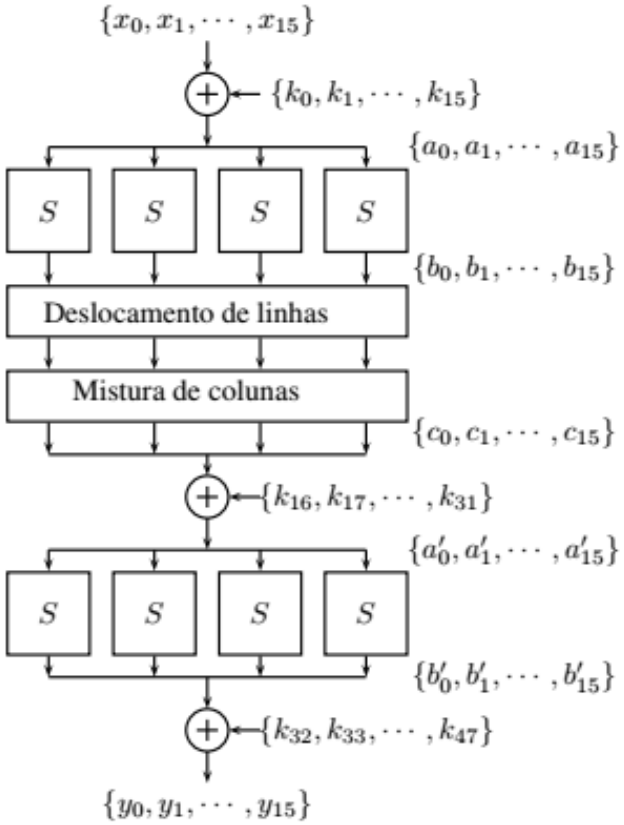
3.1 ESQUEMA AES SIMPLIFICADO

Em (MOHAMMAD; SCHAEFER; WEDIG, 2003) foi proposto o algoritmo AES simplificado (SAES) que apresenta uma estrutura de dados de entrada de saída mais simples que o AES, sem contudo perder a essência do algoritmo original. Isso significa que, ao entender o algoritmo SAES e expandir seus conceitos, pode-se entender o comportamento da criptoanálise do AES de forma mais simples. No SAES, cada bloco de entrada (texto claro) tem 16 bits $\{x_0, \ldots, x_{15}\}$ e a chave original também tem 16 bits $\{k_0, \ldots, k_{15}\}$. Considera-se duas iterações, sendo necessário expandir os 16 bits da chave criando a primeira sub-chave $\{k_{16}, \ldots, k_{31}\}$ e a segunda sub-chave $\{k_{32}, \ldots, k_{47}\}$, totalizando 48 bits.

3.1.1 Caminho de Dados

Na primeira iteração, a chave original é somada bit a bit (módulo 2) com o texto claro e realiza-se as mesmas etapas do algoritmo AES original (substituição realizada por S-Boxes, deslocamento de linhas e mistura de colunas). Na segunda iteração só se realiza a etapa de substituição e somas de sub-chaves, como ilustra a Fig. 13. A saída do SAES é o texto cifrado $\{y_0, \ldots, y_{15}\}$. O conjunto de operações realizadas em cada iteração é descrito a seguir.

Figura 13 – Diagrama em blocos do algoritmo SAES com duas iterações.



Fonte: O autor (2020).

Substituição

Os bits de entrada da etapa de substituição são dados por $a_i = x_i \oplus k_i$, para $i = 0, 1, \ldots, 15$. Esta etapa compreende 4 S-Boxes idênticas (denotado por S na Fig. 13) que operam de forma paralela, tendo cada S-box 4 bits (*nybble*) de entrada e 4 bits de saída da forma:

$$\begin{aligned} S(a_0, a_1, a_2, a_3) &= b_0 b_1 b_2 b_3 \\ S(a_4, a_5, a_6, a_7) &= b_4 b_5 b_6 b_7 \\ S(a_8, a_9, a_{10}, a_{11}) &= b_8 b_9 b_{10} b_{11} \\ S(a_{12}, a_{13}, a_{14}, a_{15}) &= b_{12} b_{13} b_{14} b_{15}. \end{aligned} \tag{3.1}$$

Os bits de saída são obtidos por um mapeamento não linear e reversível por meio de operações definidas em GF(2^4), geradas pelo polinômio primitivo $P(x) = x^4 + x + 1$. Seja a_0, a_1, a_2, a_3 a entrada de uma S-Box. Inicialmente, esta sequência é invertida em GF(2^4) (0000 não é invertível, então, nesta etapa, sua saída é 0000). A sequência de entrada invertida $a_0^-, a_1^-, a_2^-, a_3^-$ é utilizada

Tabela 10 – Entrada-saída de uma S-box no algoritmo SAES.

Entrada	Saída	Entrada	Saída
$\{0,0,0,0\} \rightarrow$	$\{1,0,0,1\}$	$\{1,0,0,0\} \rightarrow$	$\{0,1,1,0\}$
$\{0,0,0,1\} \rightarrow$	$\{0,1,0,0\}$	$\{1,0,0,1\} \rightarrow$	$\{0,0,1,0\}$
$\{0,0,1,0\} \rightarrow$	$\{1,0,1,0\}$	$\{1,0,1,0\} \rightarrow$	$\{0,0,0,0\}$
$\{0,0,1,1\} \rightarrow$	$\{1,0,1,1\}$	$\{1,0,1,1\} \rightarrow$	$\{0,0,1,1\}$
$\{0,1,0,0\} \rightarrow$	$\{1,1,0,1\}$	$\{1,1,0,0\} \rightarrow$	$\{1,1,0,0\}$
$\{0,1,0,1\} \rightarrow$	$\{0,0,0,1\}$	$\{1,1,0,1\} \rightarrow$	$\{1,1,1,0\}$
$\{0,1,1,0\} \rightarrow$	$\{1,0,0,0\}$	$\{1,1,1,0\} \rightarrow$	$\{1,1,1,1\}$
$\{0,1,1,1\} \rightarrow$	$\{0,1,0,1\}$	$\{1,1,1,1\} \rightarrow$	$\{0,1,1,1\}$

Fonte: O autor (2020).

para obter a saída da S-box da seguinte forma:

$$\begin{bmatrix} b_0 \\ b_1 \\ b_2 \\ b_3 \end{bmatrix} = \begin{bmatrix} 1 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 1 & 1 & 1 & 0 \\ 0 & 1 & 1 & 1 \end{bmatrix} \begin{bmatrix} a_0^- \\ a_1^- \\ a_2^- \\ a_3^- \end{bmatrix} + \begin{bmatrix} 1 \\ 0 \\ 0 \\ 1 \end{bmatrix}. \tag{3.2}$$

O mapeamento entre entrada e saída de uma S-box é mostrado na Tabela 10.

Deslocamento de linhas

Este bloco realiza o seguinte mapeamento:

$$\{b_0, b_1, b_2, b_3, b_4, b_5, b_6, b_7, b_8, b_9, b_{10}, b_{11}, b_{12}, b_{13}, b_{14}, b_{15}\} \rightarrow \{b_0, b_1, b_2, b_3, b_{12}, b_{13}, b_{14}, b_{15}, b_8, b_9, b_{10}, b_{11}, b_4, b_5, b_6, b_7\}$$

sendo b_i os bits da saída da etapa de substituição.

Mistura de Colunas

A etapa de mistura de colunas realiza uma mistura nos bits de saída da etapa de deslocamento de linhas, com a finalidade de misturar saídas de S-box distintas. Este é a principal etapa de difusão no SAES. Os bits de saída desta etapa são:

$$\begin{aligned}
c_0 &= b_0 \oplus b_{14} & c_8 &= b_6 \oplus b_8 \\
c_1 &= b_1 \oplus b_{12} \oplus b_{15} & c_9 &= b_4 \oplus b_7 \oplus b_9 \\
c_2 &= b_2 \oplus b_{12} \oplus b_{13} & c_{10} &= b_4 \oplus b_5 \oplus b_{10} \\
c_3 &= b_3 \oplus b_{13} & c_{11} &= b_5 \oplus b_{11} \\
c_4 &= b_2 \oplus b_{12} & c_{12} &= b_4 \oplus b_{10} \\
c_5 &= b_0 \oplus b_3 \oplus b_{13} & c_{13} &= b_5 \oplus b_8 \oplus b_{11} \\
c_6 &= b_0 \oplus b_1 \oplus b_{14} & c_{14} &= b_6 \oplus b_8 \oplus b_9 \\
c_7 &= b_1 \oplus b_{15} & c_{15} &= b_7 \oplus b_9.
\end{aligned}$$

Adição de Sub-chaves

Nesta etapa soma-se bit a bit (módulo 2) os 16 bits de saída do bloco mistura de colunas com os bits da sub-chave $\{k_{16}, \ldots, k_{31}\}$, finalizando a primeira iteração. Os bits resultantes desta soma são a entrada do bloco de substituição da segunda iteração. Os bits de saída deste bloco $\{b'_0, b'_1, \ldots, b'_{15}\}$ são somados com os bits da sub-chave $\{k_{32}, \ldots, k_{47}\}$ para gerar o texto cifrado

$$y_i = b'_i + k_{i+32}. \tag{3.3}$$

3.1.2 Obtenção de Sub-chaves

No algoritmo de obtenção de sub-chaves, 4 S-boxes são usadas para se obter os 48 bits da chave, a partir da chave original k^0. Seja

$$\begin{aligned} k^0 &= k_0^0\, k_1^0 = (k_0, k_1, \ldots, k_7)(k_8, k_9, \ldots, k_{15}) \\ k^1 &= k_0^1\, k_1^1 = (k_{16}, k_{17}, \ldots, k_{23})(k_{24}, k_{25}, \ldots, k_{31}) \\ k^2 &= k_0^2\, k_1^2 = (k_{32}, k_{33}, \ldots, k_{39})(k_{40}, k_{41}, \ldots, k_{47}). \end{aligned}$$

As entradas e saídas $(l_0, l_1, \ldots l_{15})$ das S-Boxes são relacionadas da seguinte forma:

$$\begin{aligned} S(k_{12}, k_{13}, k_{14}, k_{15}) &= l_0 l_1 l_2 l_3 \\ S(k_8, k_9, k_{10}, k_{11}) &= l_4 l_5 l_6 l_7 \\ S(k_{28}, k_{29}, k_{30}, k_{31}) &= l_8 l_9 l_{10} l_{11} \\ S(k_{24}, k_{25}, k_{26}, k_{27}) &= l_{12} l_{13} l_{14} l_{15}. \end{aligned} \tag{3.4}$$

As sub-chaves são obtidas por

$$k_0^1 = S(k_8, k_9, k_{10}, k_{11}) S(k_{12}, k_{13}, k_{14}, k_{15}) \oplus \{1, 0, 0, 0, 0, 0, 0, 0\} \oplus k_0^0$$

e $k_1^1 = k_0^1 \oplus k_1^0$. De forma similar

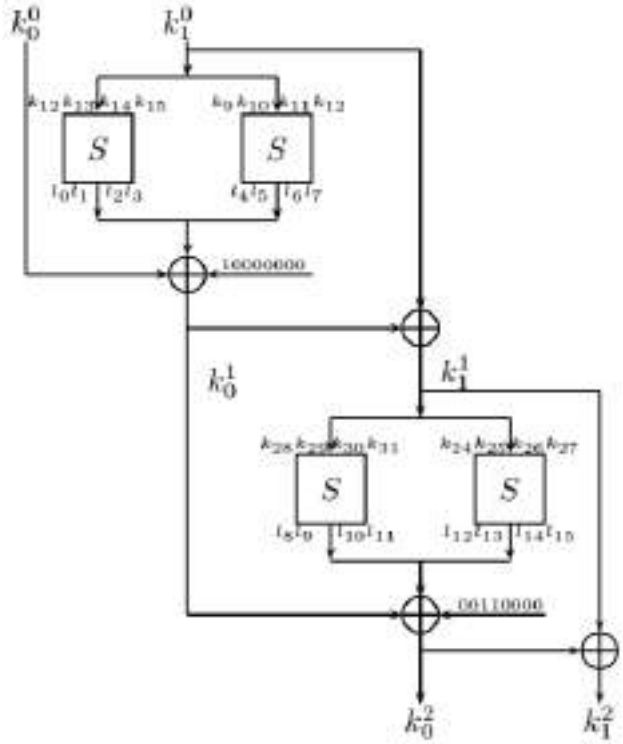
$$k_0^2 = S(k_{24}, k_{25}, k_{26}, k_{27}) S(k_{28}, k_{29}, k_{30}, k_{31}) \oplus \{0, 1, 1, 0, 0, 0, 0, 0\} \oplus k_0^1$$

e $k_1^2 = k_0^2 \oplus k_1^1$. A Fig. 14 ilustra o algoritmo de obtenção de sub-chaves. Cada bit obtido com este algoritmo é dado por:

$$
\begin{array}{llll}
k_{16} = k_0 \oplus l_0 \oplus 1 & k_{24} = k_0 \oplus k_8 \oplus l_0 \oplus 1 & k_{32} = k_0 \oplus l_0 \oplus l_8 \oplus 1 & k_{40} = k_8 \oplus l_8 \\
k_{17} = k_1 \oplus l_1 & k_{25} = k_1 \oplus k_9 \oplus l_1 & k_{33} = k_1 \oplus l_1 \oplus l_9 & k_{41} = k_9 \oplus l_9 \\
k_{18} = k_2 \oplus l_2 & k_{26} = k_2 \oplus k_{10} \oplus l_2 & k_{34} = k_2 \oplus l_2 \oplus l_{10} \oplus 1 & k_{42} = k_{10} \oplus l_{10} \oplus 1 \\
k_{19} = k_3 \oplus l_3 & k_{27} = k_3 \oplus k_{11} \oplus l_3 & k_{35} = k_3 \oplus l_3 \oplus l_{11} \oplus 1 & k_{43} = k_{11} \oplus l_{11} \oplus 1 \\
k_{20} = k_4 \oplus l_4 & k_{28} = k_4 \oplus k_{12} \oplus l_4 & k_{36} = k_4 \oplus l_4 \oplus l_{12} & k_{44} = k_{12} \oplus l_{12} \\
k_{21} = k_5 \oplus l_5 & k_{29} = k_5 \oplus k_{13} \oplus l_5 & k_{37} = k_5 \oplus l_5 \oplus l_{13} & k_{45} = k_{13} \oplus l_{13} \\
k_{22} = k_6 \oplus l_6 & k_{30} = k_6 \oplus k_{14} \oplus l_6 & k_{38} = k_6 \oplus l_6 \oplus l_{14} & k_{46} = k_{14} \oplus l_{14} \\
k_{23} = k_7 \oplus l_7 & k_{31} = k_7 \oplus k_{15} \oplus l_7 & k_{39} = k_7 \oplus l_7 \oplus l_{15} & k_{47} = k_{15} \oplus l_{15}.
\end{array}
$$

Existem 16 bits que são obtidos de equações lineares (k_1^1 e k_1^2). Portanto, é necessário conhecer 32 bits (k_0^0, k_1^0, k_0^1 e k_0^2) para descobrir os 48 bits totais. Este tópico será abordado na próxima seção.

Figura 14 – Algoritmo de obtenção de sub-chaves do SAES.



Fonte: O autor (2020).

Tabela 11 – Quantidade de equações para todas as possíveis probabilidades.

$p_\ell = 0,5$	$p_\ell = 0,5625$	$p_\ell = 0,6252$	$p_\ell = 0,75$
152	32	60	12

Fonte: O autor (2020).

3.2 CRIPTOANÁLISE LINEAR

A ideia da CL é encontrar equações da forma

$$\sum_{k \in S_1} x_k \oplus \sum_{l \in S_2} y_l = \left(\sum_{m \in S_3} k_m \right) \oplus t' \tag{3.5}$$

que apresente probabilidade maior que $0{,}5$, sendo t' um bit com valor 0 ou 1, x_k denota o k-ésimo bit de texto claro, y_l denota o l-ésimo bit de texto cifrado, k_m representa o m-ésimo bit da chave e cada S_i é um subconjunto de $\{0, 1, \ldots, 15\}$. Para cada equação, o adversário avalia o lado esquerdo de (3.5) para cada par de texto claro e texto cifrado e estima a probabilidade do lado direito estar correto. Define-se a probabilidade que a ℓ-ésima equação esteja correta por p_ℓ, em que o bit t' é escolhido para o qual $0,5 \leq p_\ell \leq 1$. Se um algoritmo de cifragem mostra uma tendência que (3.5) seja satisfeita com probabilidade $1/2$, isto é uma evidência de um algoritmo criptográfico robusto para esta criptoanálise. Quanto mais afastada a probabilidade p_ℓ é de $1/2$, mais eficaz é a CL, ou seja, precisa-se conhecer menos pares de textos claros e textos cifrados para realizar o ataque.

3.2.1 Criptoanálise Linear do SAES

Esta seção analisa a CL do esquema SAES descrito na Seção 3.1. A ideia central consiste em encontrar equações lineares correspondentes aos bits de entrada e de saída das S-boxes que possuem probabilidade maior que $0{,}5$. Seja, por exemplo, a primeira S-box, em que a relação de entrada e saída é dada por $S(a_0a_1a_2a_3) = b_0b_1b_2b_3$. Existem 256 equações para todas as possíveis combinações de entrada e de saída, sendo possível extrair 12 equações com probabilidade $0{,}75$ entre as entradas e saídas desta S-box:

$$\begin{array}{rcrcr}
a_0 \oplus b_2 = & 0 & a_1 \oplus b_0 \oplus b_3 & =0 & a_1 \oplus b_1 = \quad 0 \\
a_2 \oplus a_3 \oplus b_3 = & 1 & a_0 \oplus a_1 \oplus b_1 \oplus b_2 \oplus b_3 & =1 & a_0 \oplus a_1 \oplus a_2 \oplus a_3 \oplus b_1 = \quad 0 \\
a_2 \oplus b_2 \oplus b_3 = & 1 & a_3 \oplus b_0 & =1 & a_1 \oplus a_2 \oplus b_0 \oplus b_1 = \quad 1 \\
a_2 \oplus b_1 \oplus b_3 = & 1 & a_0 \oplus a_1 \oplus b_0 & =1 & a_0 \oplus b_0 \oplus b_1 = \quad 1.
\end{array} \tag{3.6}$$

O número de equações para cada probabilidade possível é mostrada na Tabela 11. No algoritmo SAES existem 4 S-boxes por iteração no caminho de dados, então para cada S-box obtém-se 12 equações similares a (3.6), totalizando 48 equações com probabilidade $0{,}75$. Após a primeira etapa de substituição obtém-se 48 equações com probabilidade 0,75 que dependem dos bits de texto claro $(x_0, \ldots, x_{15})$, dos bits da chave original $(k_0, \ldots, k_{15})$ e dos bits de saída das 4 S-boxes da primeira iteração $(b_0, \ldots, b_{15})$:

$$
\begin{aligned}
x_0 \oplus b_2 \oplus k_0 &= 0 \\
x_4 \oplus b_6 \oplus k_4 &= 0 \\
x_8 \oplus b_{10} \oplus k_8 &= 0 \\
x_{12} \oplus b_{14} \oplus k_{12} &= 0 \\
x_2 \oplus b_2 \oplus b_3 \oplus k_2 &= 1 \\
x_6 \oplus b_6 \oplus b_7 \oplus k_6 &= 1 \\
x_{10} \oplus b_{10} \oplus b_{11} \oplus k_{10} &= 1 \\
x_{14} \oplus b_{14} \oplus b_{15} \oplus k_{14} &= 1
\end{aligned}
\tag{3.7}
$$

$$
\begin{aligned}
x_2 \oplus x_3 \oplus b_3 \oplus k_2 \oplus k_3 &= 1 \\
x_6 \oplus x_7 \oplus b_7 \oplus k_6 \oplus k_7 &= 1 \\
x_{10} \oplus x_{11} \oplus b_{11} \oplus k_{10} \oplus k_{11} &= 1 \\
x_{14} \oplus x_{15} \oplus b_{15} \oplus k_{14} \oplus k_{15} &= 1 \\
x_2 \oplus b_1 \oplus b_3 \oplus k_2 &= 1 \\
x_6 \oplus b_5 \oplus b_7 \oplus k_6 &= 1 \\
x_{10} \oplus b_9 \oplus b_{11} \oplus k_{10} &= 1 \\
x_{14} \oplus b_{13} \oplus b_{15} \oplus k_{14} &= 1 \\
x_3 \oplus b_0 \oplus k_3 &= 1 \\
x_7 \oplus b_4 \oplus k_7 &= 1 \\
x_{11} \oplus b_8 \oplus k_{11} &= 1 \\
x_{15} \oplus b_{12} \oplus k_{15} &= 1 \\
x_1 \oplus b_1 \oplus k_1 &= 0 \\
x_5 \oplus b_5 \oplus k_5 &= 0 \\
x_9 \oplus b_9 \oplus k_9 &= 0 \\
x_{13} \oplus b_{13} \oplus k_{13} &= 0 \\
x_0 \oplus x_1 \oplus b_0 \oplus k_0 \oplus k_1 &= 1 \\
x_4 \oplus x_5 \oplus b_4 \oplus k_4 \oplus k_5 &= 1 \\
x_8 \oplus x_9 \oplus b_8 \oplus k_8 \oplus k_9 &= 1 \\
x_{12} \oplus x_{13} \oplus b_{12} \oplus k_{12} \oplus k_{13} &= 1
\end{aligned}
$$

$$
\begin{aligned}
x_1 \oplus b_0 \oplus b_3 \oplus k_1 &= 0 \\
x_5 \oplus b_4 \oplus b_7 \oplus k_5 &= 0 \\
x_9 \oplus b_8 \oplus b_{11} \oplus k_9 &= 0 \\
x_{13} \oplus b_{12} \oplus b_{15} \oplus k_{13} &= 0 \\
x_1 \oplus x_2 \oplus b_0 \oplus b_1 \oplus k_1 \oplus k_2 &= 1 \\
x_5 \oplus x_6 \oplus b_4 \oplus b_5 \oplus k_5 \oplus k_6 &= 1 \\
x_9 \oplus x_{10} \oplus b_8 \oplus b_9 \oplus k_9 \oplus k_{10} &= 1 \\
x_{13} \oplus x_{14} \oplus b_{12} \oplus b_{11} \oplus k_{13} \oplus k_{14} &= 1 \\
x_0 \oplus x_1 \oplus b_1 \oplus b_2 \oplus b_3 \oplus k_0 \oplus k_1 &= 0 \\
x_4 \oplus x_5 \oplus b_5 \oplus b_6 \oplus b_7 \oplus k_4 \oplus k_5 &= 0 \\
x_8 \oplus x_9 \oplus b_9 \oplus b_{10} \oplus b_{11} \oplus k_8 \oplus k_9 &= 0 \\
x_{12} \oplus x_{13} \oplus b_{13} \oplus b_{14} \oplus b_{15} \oplus k_{12} \oplus k_{13} &= 0 \\
x_0 \oplus b_0 \oplus b_1 \oplus k_0 &= 1 \\
x_4 \oplus b_4 \oplus b_5 \oplus k_4 &= 1 \\
x_8 \oplus b_8 \oplus b_9 \oplus k_8 &= 1 \\
x_{12} \oplus b_{12} \oplus b_{13} \oplus k_{12} &= 1.
\end{aligned}
$$

Realizando o mesmo procedimento na segunda iteração, obtém-se 48 novas equações com probabilidade $0{,}75$ que dependem dos bits de saída das S-boxes da primeira iteração $(b_0, \ldots, b_{15})$, dos 48 bits da chave k^0, k^1, k^2, e dos bits de texto cifrado $(y_0, \ldots, y_{15})$:

$$
\begin{aligned}
b_0 \oplus b_{14} \oplus y_2 \oplus k_{16} \oplus k_{34} &= 1 \\
b_2 \oplus b_{12} \oplus y_6 \oplus k_{20} \oplus k_{37} &= 0 \\
b_6 \oplus b_8 \oplus y_{10} \oplus k_{24} \oplus k_{41} &= 1 \\
b_4 \oplus b_{10} \oplus y_{14} \oplus k_{28} \oplus k_{45} &= 0 \\
b_2 \oplus b_3 \oplus b_{12} \oplus y_3 \oplus k_{18} \oplus k_{19} \oplus k_{35} &= 1 \\
b_0 \oplus b_{14} \oplus b_{15} \oplus y_5 \oplus k_{22} \oplus k_{23} \oplus k_{39} &= 1 \\
b_4 \oplus b_{10} \oplus b_{10} \oplus y_9 \oplus k_{26} \oplus k_{27} \oplus k_{43} &= 1 \\
b_6 \oplus b_7 \oplus b_8 \oplus y_{13} \oplus k_{30} \oplus k_{31} \oplus k_{47} &= 1 \\
b_2 \oplus b_{12} \oplus b_{13} \oplus y_2 \oplus y_3 \oplus k_{18} \oplus k_{34} \oplus k_{35} &= 1 \\
b_0 \oplus b_1 \oplus b_{14} \oplus y_6 \oplus y_7 \oplus k_{22} \oplus k_{38} \oplus k_{39} &= 1 \\
b_4 \oplus b_5 \oplus b_{10} \oplus y_{10} \oplus y_{11} \oplus k_{26} \oplus k_{42} \oplus k_{43} &= 1 \\
b_6 \oplus b_8 \oplus b_9 \oplus y_{14} \oplus y_{15} \oplus k_{30} \oplus k_{46} \oplus k_{47} &= 1 \\
b_2 \oplus b_{12} \oplus b_{13} \oplus y_1 \oplus y_3 \oplus k_{18} \oplus k_{33} \oplus k_{35} &= 1 \\
b_0 \oplus b_1 \oplus b_{14} \oplus y_5 \oplus y_7 \oplus k_{22} \oplus k_{37} \oplus k_{39} &= 1 \\
b_4 \oplus b_5 \oplus b_{10} \oplus y_9 \oplus y_{11} \oplus k_{26} \oplus k_{41} \oplus k_{43} &= 1 \\
b_6 \oplus b_8 \oplus b_9 \oplus y_{13} \oplus y_{15} \oplus k_{30} \oplus k_{45} \oplus k_{47} &= 1 \\
b_1 \oplus b_{12} \oplus b_{15} \oplus y_0 \oplus y_3 \oplus k_{17} \oplus k_{32} \oplus k_{35} &= 0 \\
b_0 \oplus b_3 \oplus b_{13} \oplus y_4 \oplus y_7 \oplus k_{21} \oplus k_{36} \oplus k_{39} &= 0 \\
b_4 \oplus b_7 \oplus b_9 \oplus y_8 \oplus y_{11} \oplus k_{25} \oplus k_{40} \oplus k_{43} &= 1 \\
b_5 \oplus b_8 \oplus b_{11} \oplus y_{12} \oplus y_{15} \oplus k_{29} \oplus k_{44} \oplus k_{47} &= 1
\end{aligned}
\tag{3.8}
$$

$$
\begin{aligned}
b_0 \oplus b_1 \oplus b_{12} \oplus b_{14} \oplus b_{15} \oplus y_1 \oplus y_2 \oplus y_3 \oplus k_{16} \oplus k_{17} \oplus k_{33} \oplus k_{34} \oplus k_{35} &= 1 \\
b_0 \oplus b_2 \oplus b_3 \oplus b_{12} \oplus b_{13} \oplus y_5 \oplus y_6 \oplus y_7 \oplus k_{20} \oplus k_{21} \oplus k_{37} \oplus k_{38} \oplus k_{39} &= 1 \\
b_4 \oplus b_6 \oplus b_7 \oplus b_8 \oplus b_9 \oplus y_9 \oplus y_{10} \oplus y_{11} \oplus k_{24} \oplus k_{25} \oplus k_{41} \oplus k_{42} \oplus k_{43} &= 1 \\
b_4 \oplus b_5 \oplus b_8 \oplus b_{10} \oplus b_{11} \oplus y_{13} \oplus y_{14} \oplus y_{15} \oplus k_{28} \oplus k_{29} \oplus k_{45} \oplus k_{46} \oplus k_{47} &= 1 \\
b_3 \oplus b_{13} \oplus y_0 \oplus k_{19} \oplus k_{32} &= 1 \\
b_1 \oplus b_{15} \oplus y_4 \oplus k_{23} \oplus k_{36} &= 1 \\
b_5 \oplus b_{11} \oplus y_8 \oplus k_{27} \oplus k_{40} &= 1 \\
b_7 \oplus b_9 \oplus y_{12} \oplus k_{31} \oplus k_{44} &= 1 \\
b_0 \oplus b_1 \oplus b_{12} \oplus b_{14} \oplus b_{15} \oplus y_0 \oplus k_{16} \oplus k_{17} \oplus k_{32} &= 1 \\
b_0 \oplus b_2 \oplus b_3 \oplus b_{12} \oplus b_{13} \oplus y_4 \oplus k_{20} \oplus k_{21} \oplus k_{36} &= 1 \\
b_4 \oplus b_6 \oplus b_7 \oplus b_8 \oplus b_9 \oplus y_8 \oplus k_{24} \oplus k_{25} \oplus k_{40} &= 1 \\
b_4 \oplus b_5 \oplus b_8 \oplus b_{10} \oplus b_{11} \oplus y_{12} \oplus k_{28} \oplus k_{29} \oplus k_{44} &= 1 \\
b_1 \oplus b_{12} \oplus b_{15} \oplus y_1 \oplus k_{17} \oplus k_{33} &= 0 \\
b_0 \oplus b_3 \oplus b_{13} \oplus y_5 \oplus k_{21} \oplus k_{37} &= 0 \\
b_4 \oplus b_7 \oplus b_9 \oplus y_9 \oplus k_{25} \oplus k_{41} &= 0 \\
b_5 \oplus b_8 \oplus b_{11} \oplus y_{13} \oplus k_{29} \oplus k_{45} &= 0 \\
b_0 \oplus b_1 \oplus b_2 \oplus b_3 \oplus b_{14} \oplus b_{15} \oplus y_1 \oplus k_{16} \oplus k_{17} \oplus k_{18} \oplus k_{19} \oplus k_{33} &= 0 \\
b_2 \oplus b_3 \oplus b_{12} \oplus b_{13} \oplus b_{14} \oplus b_{15} \oplus y_5 \oplus k_{20} \oplus k_{21} \oplus k_{22} \oplus k_{23} \oplus k_{37} &= 0 \\
b_6 \oplus b_7 \oplus b_8 \oplus b_9 \oplus b_{10} \oplus b_1 1 \oplus y_9 \oplus k_{24} \oplus k_{25} \oplus k_{26} \oplus k_{27} \oplus k_{41} &= 0 \\
b_4 \oplus b_5 \oplus b_6 \oplus b_7 \oplus b_{10} \oplus b_{11} \oplus y_{13} \oplus k_{28} \oplus k_{29} \oplus k_{30} \oplus k_{31} \oplus k_{45} &= 0
\end{aligned}
$$

$$
\begin{aligned}
b_1 \oplus b_2 \oplus b_{13} \oplus b_{15} \oplus y_0 \oplus y_1 \oplus k_{18} \oplus k_{32} \oplus k_{33} &= 1 \\
b_1 \oplus b_3 \oplus b_{13} \oplus b_{14} \oplus y_4 \oplus y_5 \oplus k_{22} \oplus k_{36} \oplus k_{37} &= 1 \\
b_5 \oplus b_7 \oplus b_9 \oplus b_{13} \oplus y_8 \oplus y_9 \oplus k_{26} \oplus k_{40} \oplus k_{41} &= 1 \\
b_5 \oplus b_6 \oplus b_7 \oplus b_{11} \oplus y_{12} \oplus y_{13} \oplus k_{30} \oplus k_{44} \oplus k_{45} &= 1 \\
b_0 \oplus b_{14} \oplus y_0 \oplus y_1 \oplus k_{18} \oplus k_{32} \oplus k_{33} &= 1 \\
b_2 \oplus b_{12} \oplus y_4 \oplus y_5 \oplus k_{22} \oplus k_{36} \oplus k_{37} &= 1 \\
b_6 \oplus b_8 \oplus y_8 \oplus y_9 \oplus k_{26} \oplus k_{40} \oplus k_{41} &= 1 \\
b_4 \oplus b_{10} \oplus y_{12} \oplus y_{13} \oplus k_{30} \oplus k_{44} \oplus k_{45} &= 1.
\end{aligned}
$$

Como o criptoanalista tem conhecimento de pares de texto claro ($x_0, \ldots x_{15}$) e de texto cifrado ($y_0, \ldots y_{15}$) é preciso combinar (somar módulo 2) equações de iterações distintas afim de obter equações na forma mostrada em (3.5), que não dependem dos bits de saída das S-boxes da primeira iteração $(b_0, \ldots, b_{15})$. Como cada equação em cada iteração é satisfeita com um certa probabilidade, a probabilidade de uma equação obtida a partir da combinação de equações será discutida na próxima subseção.

Combinação de Equações de Iterações Distintas

Como é apresentado em (MANSOORI; KHALEGHI, 2006), cada expressão do lado esquerdo da igualdade em (3.7) e (3.8) é considerada uma variável aleatória binária. Sejam X e Y variáveis aleatórias de Bernoulli independentes associadas a equações lineares obtidas de S-boxes distintas do SAES (na mesma iteração ou em iterações distintas). O evento $X = 1$ significa que uma equação é satisfeita. A mesma definição vale para Y. Seja $p_1 \triangleq \Pr(X = 1)$ e $p_2 \triangleq \Pr(Y = 1)$, em que $0{,}5 < p_1; p_2 < 1$. Seja V uma variável aleatória de Bernoulli tal que $V = 1$ significa que a combinação linear de equações associadas a X e Y é satisfeita, então

$$\begin{aligned} P(V=1) &= P(X=1,Y=1) + P(X=0,Y=0) \\ &= P(X=1)P(Y=1) + P(X=0)P(Y=0) \\ &= p_1p_2 + (1-p_1)(1-p_2). \end{aligned} \tag{3.9}$$

Quando $p_1 = p_2 = p$ (caso mostrado em (3.7) e (3.8)), obtém-se $q \triangleq P(V = 1) = 2p^2 - 2p + 1$. No caso em que $p = 0{,}75$, então:

$$q = 2(0{,}75)^2 - 2(0{,}75) + 1 = 0{,}625. \tag{3.10}$$

A combinação de equações com probabilidade p resulta em novas equações com probabilidade da forma $q = 2p^2 - 2p + 1$. É importante frisar que $0{,}5 \leq q \leq p$ quando p pertence ao intervalo $0{,}5 \leq p \leq 1$, uma vez que

$$q - p = 2p^2 - 3p + 1 = (2p - 1)(p - 1).$$

Para o intervalo válido para p, o termo $(p - 1)$ tem valores negativos e o termo $(2p - 1)$ valores positivos, obtendo $q - p \leq 0$. Seguindo um raciocínio análogo, pode ser mostrado que a combinação de equações com probabilidades distintas é limitado superiormente por:

$$q = P(X \oplus Y = 1) = p_1p_2 + (1-p_1)(1-p_2) \leq \min(p_1, p_2). \tag{3.11}$$

No intervalo válido de p_1 e p_2, existem duas regiões de mínimos para q, em $p_1 = 0{,}5$ ou $p_2 = 0{,}5$, e um valor de máximo quando $p_1 = p_2 = 1$. Portanto, a probabilidade da combinação de equações com probabilidades diferentes é limitada pelo menor delas, atingindo o menor valor igual a $0{,}5$ quando uma das probabilidades é $0{,}5$. Este método é adequado para determinar um

limitante superior para a probabilidade de uma equação ser satisfeita, quando é gerada a partir da combinação de equações de S-boxes distintas na mesma iteração ou de iterações distintas. Os bits de texto claro e bits da chave que formam essas equações são variáveis aleatórias binárias independentes.

Utilizando (3.7), escrevem-se os bits de saída da etapa de substituição da primeira iteração ($b_0, \ldots, b_{15}$) em função dos bits de entrada ($x_0, \ldots, x_{15}$) e dos bits da chave original ($k_0, \ldots, k_{15}$) e combina-se a equação resultante com (3.8). Por exemplo, considere a seguinte equação de (3.8) $b_5 \oplus b_8 \oplus b_{11} \oplus y_{12} \oplus y_{15} \oplus k_{29} \oplus k_{44} \oplus k_{47} = 1$ (válida com probabilidade 0,75) e combinando esta com duas equações de (3.7) $b_5 = x_5 \oplus k_5$ e $b_8 \oplus b_{11} = x_9 \oplus k_9$ (cada uma com probabilidade 0,75), resulta em uma equação na forma mostrada em (3.5) $x_5 \oplus x_9 \oplus y_{12} \oplus y_{15} = k_5 \oplus k_9 \oplus k_{29} \oplus k_{44} \oplus k_{47} \oplus 1$. A soma de duas equações obtidas de (3.7), resulta em uma equação com probabilidade:

$$\begin{aligned} q &= 2p^2 - 2p + 1 \\ &= 2(0{,}75)^2 - 2(0{,}75) + 1 \\ &= 0{,}625 \end{aligned}$$

que somada com uma equação de (3.8) obtém-se uma equação com probabilidade:

$$\begin{aligned} q_1 &= p_1 p_2 + (1 - p_1)(1 - p_2) \\ &= (0{,}75)(0{,}625) + (1 - 0{,}75)(1 - 0{,}625) \\ &= 0{,}5625. \end{aligned}$$

Usando um procedimento similar, obtém-se as seguintes 32 equações linearmente independentes

com probabilidade $0{,}5625$:

$$
\begin{aligned}
x_3 \oplus x_{12} \oplus y_2 &= k_3 \oplus k_{12} \oplus k_{16} \oplus k_{34} \oplus 1 \\
x_0 \oplus x_{15} \oplus y_6 &= k_0 \oplus k_{15} \oplus k_{20} \oplus k_{38} \oplus 1 \\
x_4 \oplus x_{11} \oplus y_{10} &= k_4 \oplus k_{11} \oplus k_{24} \oplus k_{42} \oplus 1 \\
x_7 \oplus x_8 \oplus y_{14} &= k_7 \oplus k_8 \oplus k_{28} \oplus k_{46} \oplus 1 \\
x_0 \oplus x_3 \oplus x_{15} \oplus y_3 &= k_0 \oplus k_3 \oplus k_{15} \oplus k_{18} \oplus k_{19} \oplus k_{35} \oplus 1 \\
x_3 \oplus x_{12} \oplus x_{13} \oplus x_{15} \oplus y_7 &= k_3 \oplus k_{12} \oplus k_{13} \oplus k_{15} \oplus k_{22} \oplus k_{23} \oplus k_{39} \oplus 1 \\
x_7 \oplus x_8 \oplus x_9 \oplus x_{11} \oplus y_{11} &= k_7 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{26} \oplus k_{27} \oplus k_{43} \oplus 1 \\
x_4 \oplus x_5 \oplus x_7 \oplus x_{11} \oplus y_{15} &= k_4 \oplus k_6 \oplus k_7 \oplus k_{11} \oplus k_{30} \oplus k_{31} \oplus k_{47} \oplus 1 \\
x_0 \oplus x_{13} \oplus x_{15} \oplus y_2 \oplus y_3 &= k_0 \oplus k_{13} \oplus k_{15} \oplus k_{18} \oplus k_{34} \oplus k_{35} \\
x_1 \oplus x_3 \oplus x_{12} \oplus y_6 \oplus y_7 &= k_1 \oplus k_3 \oplus k_{12} \oplus k_{22} \oplus k_{38} \oplus k_{39} \\
x_5 \oplus x_7 \oplus x_8 \oplus y_{10} \oplus y_{11} &= k_5 \oplus k_7 \oplus k_8 \oplus k_{26} \oplus k_{42} \oplus k_{43} \\
x_4 \oplus x_9 \oplus x_{11} \oplus y_{14} \oplus y_{15} &= k_4 \oplus k_9 \oplus k_{11} \oplus k_{30} \oplus k_{46} \oplus k_{47} \\
x_1 \oplus x_{13} \oplus y_0 \oplus y_3 &= k_1 \oplus k_{13} \oplus k_{17} \oplus k_{32} \oplus k_{35} \\
x_1 \oplus x_{13} \oplus y_4 \oplus y_7 &= k_1 \oplus k_{13} \oplus k_{21} \oplus k_{36} \oplus k_{39} \\
x_5 \oplus x_9 \oplus y_8 \oplus y_{11} &= k_5 \oplus k_9 \oplus k_{25} \oplus k_{40} \oplus k_{43} \\
x_5 \oplus x_9 \oplus y_{12} \oplus y_{15} &= k_5 \oplus k_9 \oplus k_{29} \oplus k_{44} \oplus k_{47} \\
x_1 \oplus x_3 \oplus x_{12} \oplus x_{13} \oplus y_0 &= k_1 \oplus k_3 \oplus k_{12} \oplus k_{13} \oplus k_{16} \oplus k_{17} \oplus k_{32} \\
x_0 \oplus x_1 \oplus x_{13} \oplus x_{15} \oplus y_4 &= k_0 \oplus k_1 \oplus k_{13} \oplus k_{15} \oplus k_{20} \oplus k_{21} \oplus k_{36} \\
x_4 \oplus x_5 \oplus x_9 \oplus x_{11} \oplus y_8 &= k_4 \oplus k_5 \oplus k_9 \oplus k_{11} \oplus k_{24} \oplus k_{25} \oplus k_{40} \\
x_5 \oplus x_7 \oplus x_8 \oplus x_9 \oplus y_{12} &= k_5 \oplus k_7 \oplus k_8 \oplus k_9 \oplus k_{28} \oplus k_{29} \oplus k_{44}
\end{aligned}
\tag{3.12}
$$

$$
\begin{aligned}
x_0 \oplus x_{12} \oplus x_{13} \oplus x_{15} \oplus y_1 &= k_0 \oplus k_{12} \oplus k_{13} \oplus k_{15} \oplus k_{16} \oplus k_{17} \oplus k_{18} \oplus k_{19} \oplus k_{33} \\
x_0 \oplus x_1 \oplus x_3 \oplus x_{12} \oplus y_5 &= k_0 \oplus k_1 \oplus k_3 \oplus k_{12} \oplus k_{20} \oplus k_{21} \oplus k_{22} \oplus k_{23} \oplus k_{37} \\
x_4 \oplus x_5 \oplus x_7 \oplus x_8 \oplus y_9 &= k_4 \oplus k_5 \oplus k_7 \oplus k_8 \oplus k_{24} \oplus k_{25} \oplus k_{26} \oplus k_{27} \oplus k_{41} \\
x_4 \oplus x_8 \oplus x_9 \oplus x_{11} \oplus y_{13} &= k_4 \oplus k_8 \oplus k_9 \oplus k_{11} \oplus k_{28} \oplus k_{29} \oplus k_{30} \oplus k_{31} \oplus k_{45} \\
x_0 \oplus x_1 \oplus x_{15} \oplus y_0 \oplus y_1 &= k_0 \oplus k_1 \oplus k_{15} \oplus k_{17} \oplus k_{18} \oplus k_{32} \oplus k_{33} \oplus 1 \\
x_2 \oplus x_{12} \oplus x_{13} \oplus y_4 \oplus y_5 &= k_2 \oplus k_{12} \oplus k_{13} \oplus k_{21} \oplus k_{22} \oplus k_{36} \oplus k_{35} \oplus 1 \\
x_7 \oplus x_8 \oplus x_9 \oplus y_8 \oplus y_9 &= k_7 \oplus k_8 \oplus k_9 \oplus k_{25} \oplus k_{26} \oplus k_{40} \oplus k_{41} \oplus 1 \\
x_4 \oplus x_5 \oplus x_{11} \oplus y_{12} \oplus y_{13} &= k_4 \oplus k_5 \oplus k_{11} \oplus k_{29} \oplus k_{30} \oplus k_{44} \oplus k_{45} \oplus 1 \\
x_3 \oplus x_{12} \oplus y_0 \oplus y_1 &= k_3 \oplus k_{12} \oplus k_{16} \oplus k_{32} \oplus k_{33} \\
x_0 \oplus x_{15} \oplus y_4 \oplus y_5 &= k_0 \oplus k_{15} \oplus k_{20} \oplus k_{36} \oplus k_{37} \\
x_4 \oplus x_{11} \oplus y_8 \oplus y_9 &= k_4 \oplus k_{11} \oplus k_{24} \oplus k_{40} \oplus k_{41} \\
x_7 \oplus x_8 \oplus y_{12} \oplus y_{13} &= k_7 \oplus k_8 \oplus k_{28} \oplus k_{44} \oplus k_{45}.
\end{aligned}
$$

A questão principal é, quantos pares de texto claro e texto cifrado n_p são necessários para o adversário quebrar a cifra (com uma certa confiabilidade) usando as 32 equações. Define-se que o adversário quer uma confiabilidade de 95% que os bits da chave obtidos sejam corretos.

Seja W uma variável aleatória definida como o número de pares de texto claro e texto cifrado para o qual o lado direito de cada equação em (3.12) é o valor correto (0 ou 1) do lado esquerdo dividido por n_p, para uma dada chave. Cada par de texto claro e texto cifrado é uma realização do experimento e a probabilidade que o lado direito da equação seja correto é q. Cada realização é independente e podemos descrever este processo como uma distribuição binomial normalizada pela quantidade de pares de texto claro texto cifrado n_p. Então, a média de W é q e sua variância é:

$$
\sigma^2 = \frac{q\,(1-q)}{n_p}. \tag{3.13}
$$

Para a CL é desejado que $P(W \geq 0{,}5)$. Usando a confiabilidade estabelecida, então $P(W \geq 0{,}5) = \sqrt[32]{0{,}95} = 0{,}9984$. Para um valor suficientemente grande de n_p, a função densidade de probabilidade de W tende a uma função densidade de probabilidade de uma variável aleatória normal. Definindo uma variável aleatória normal $Z = (W - q)/\sigma$ de média zero e variância

unitária, temos que:

$$\begin{aligned} P(W \geq 0{,}5) &= P\left(Z \geq \frac{0{,}5-q}{\sigma}\right) \\ &= P\left(Z \geq \frac{\sqrt{n_p}(0{,}5-q)}{\sqrt{q(1-q)}}\right) \\ &= P\left(Z \geq -\frac{\sqrt{n_p}(q-0{,}5)}{\sqrt{q(1-q)}}\right) \\ &= 1 - Q\left(\frac{\sqrt{n_p}(q-0{,}5)}{\sqrt{q(1-q)}}\right) \\ &= 0{,}9984 \end{aligned}$$

em que $Q(x) \triangleq 1/\sqrt{2\pi}\int_x^\infty \exp\{-t^2/2\}dt$. Portanto

$$Q\left(\frac{\sqrt{n_p}(q-0{,}5)}{\sqrt{q(1-q)}}\right) = 0{,}0016. \tag{3.14}$$

Para o caso em que $q = 0{,}5625$, obtém-se de (3.14):

$$Q(0,126\sqrt{n_p}) = 0{,}0016. \tag{3.15}$$

O argumento da função $Q(x)$ que satisfaz (3.15) é $2{,}9$, sendo $0{,}121\sqrt{n_p} = 2{,}9$, obtendo-se $n_p = 575{,}24$. Desta forma, necessita-se 576 pares de texto claro e texto cifrado para achar os bits da chave com uma confiabilidade de 95 % que estes estejam corretos. Assim, a CL é atraente comparado a um ataque de força bruta para o SAES com duas iterações. Na próxima subseção, uma análise similar é realizada para algoritmos SAES modificados por sequências caóticas.

3.2.2 Criptoanálise Linear do SAES Modificado com Sequência Caótica

Nesta subseção analisa-se a CL para três algoritmos baseados no SAES com as S-Boxes modificadas por uma sequência caótica. Estes são denominados de SAES1, SAES2, SAES3 e seguem as mesmas ideias dos esquemas AES1, AES2, AES3 introduzidas no Capítulo 3. Realiza-se uma análise de como os bits caóticos impactam na probabilidade que as equações sejam satisfeitas e no número de pares de texto claro e texto cifrado n_p.

SAES1

Neste algoritmo, os 4 bits de saída de cada S-Box são somados (XOR) bit a bit com uma sequência **h** gerada a partir de um mapa caótico. Para cada S-box no caminho de dados, utilizam-se dois bits caóticos (z_0, z_1), com representação polinomial dada por $c(x) = z_0x + z_1$. Como as operações de cada S-box são definidas em GF(2^4), multiplica-se $c(x)$ por um polinômio primitivo $p(x) = x^3 + x + 1$ em GF(2^3), obtendo assim o polinômio $h(x) = c(x)p(x) \mathrm{mod} P(x)$.

Os coeficientes deste polinômio formam uma sequência $\mathbf{h} = (h_0, h_1, h_2, h_3)$. Este mapeamento é dado por:

$$\begin{aligned}(z_0, z_1) &\rightarrow (h_0, h_1, h_2, h_3) \\ (0,0) &\rightarrow (0,0,0,0) \\ (0,1) &\rightarrow (1,0,1,1) \\ (1,0) &\rightarrow (0,1,0,1) \\ (1,1) &\rightarrow (1,1,1,0).\end{aligned} \tag{3.16}$$

Dois bits caóticos (z_2, z_3) são usados na obtenção das sub-chaves de forma similar, totalizando quatro bits caóticos para cifrar um bloco de 16 bits de texto claro. Para simplificar a análise, na segunda iteração utiliza-se a mesma sequência $\mathbf{h}$ usada na iteração anterior. A utilização dos bits caóticos tem um impacto direto nos bits de saída da etapa de substituição. Observa-se em (3.16) que h_0 e h_2 são iguais a z_1, h_1 é igual a z_0, e h_3 é igual a $z_0 \oplus z_1$. No caminho de dados, os bits de saída das S-boxes na primeira iteração do SAES1 se relacionam com os bits de saída do SAES da seguinte forma:

$$[\hat{b}_i, \hat{b}_{i+1}, \hat{b}_{i+2}, \hat{b}_{i+3}] = [b_i, b_{i+1}, b_{i+2}, b_{i+3}] \oplus [z_1, z_0, z_1, z_0 \oplus z_1] \tag{3.17}$$

sendo $i = \{0, 4, 8, 12\}$. De forma similar são obtidos os bits na saída da segunda iteração $(\hat{b}'_0, \ldots, \hat{b}'_{15})$.

As sub-chaves do SAES1 $(\hat{k}_{16}, \ldots, \hat{k}_{47})$ se relacionam com as sub-chaves do SAES da seguinte forma:

$$[\hat{k}_i, \hat{k}_{i+1}, \hat{k}_{i+2}, \hat{k}_{i+3}] = [k_i, k_{i+1}, k_{i+2}, k_{i+3}] \oplus [z_3, z_2, z_3, z_2 \oplus z_3] \tag{3.18}$$

sendo $i = \{16, 20, 24, 28, 32, 36, 40, 44\}$. O texto cifrado é expresso como

$$[\hat{y}_i, \hat{y}_{i+1}, \hat{y}_{i+2}, \hat{y}_{i+3}] = [y_i, y_{i+1}, y_{i+2}, y_{i+3}] \oplus [z_1, z_0, z_1, z_0 \oplus z_1] \tag{3.19}$$

para $i = \{0, 4, 8, 12\}$. O algoritmo SAES1 tem a mesma estrutura que o SAES, portanto as 48 equações de (3.8) com probabilidade 0,75 são válidas para o SAES1, bastando substituir b_i, y_i e k_i por $\hat{b}_i$, $\hat{y}_i$ e $\hat{k}_i$, respectivamente. Usando (3.17), (3.18), (3.19) obtemos equações que dependem dos bits caóticos. Por exemplo, realizando estes procedimentos na seguinte equação de (3.8):

$$\hat{b}_5 \oplus x_5 = k_5, \quad \text{ou} \quad b_5 \oplus z_0 \oplus x_5 = k_5. \tag{3.20}$$

Analogamente,

$$\hat{b}_8 \oplus \hat{b}_{11} \oplus x_9 = k_9, \quad \text{ou} \quad b_8 \oplus z_1 \oplus b_{11} \oplus z_0 \oplus z_1 \oplus x_9 = k_9$$

ou

$$b_8 \oplus b_{11} \oplus z_0 \oplus x_9 = k_9. \tag{3.21}$$

A equação da segunda iteração $b_5 \oplus b_8 \oplus b_{11} \oplus y_{12} \oplus y_{15} = k_{29} \oplus k_{44} \oplus k_{47} \oplus 1$ do algoritmo SAES torna-se

$$\hat{b}_5 \oplus \hat{b}_8 \oplus \hat{b}_{11} \oplus \hat{y}_{12} \oplus \hat{y}_{15} = \hat{k}_{29} \oplus \hat{k}_{44} \oplus \hat{k}_{47} \oplus 1$$

ou

$$b_5 \oplus z_0 \oplus b_8 \oplus z_1 \oplus b_{11} \oplus z_0 \oplus z_1 \oplus y_{12} \oplus z_1 \oplus y_{15} \oplus z_0 \oplus z_1 = \\ k_{29} \oplus z_2 \oplus k_{44} \oplus z_3 \oplus k_{47} \oplus z_2 \oplus z_3 \oplus 1$$

ou

$$b_5 \oplus b_8 \oplus b_{11} \oplus y_{12} \oplus y_{15} = k_{29} \oplus k_{44} \oplus k_{47} \oplus z_0 \oplus 1. \tag{3.22}$$

Combinando (3.20), (3.21), e (3.22), obtêm-se

$$\frac{\begin{array}{c} b_5 \oplus z_0 \oplus x_5 = k_5 \\ b_8 \oplus b_{11} \oplus z_0 \oplus x_9 = k_9 \\ b_5 \oplus b_8 \oplus b_{11} \oplus y_{12} \oplus y_{15} = k_{29} \oplus k_{44} \oplus k_{47} \oplus z_0 \oplus 1 \end{array}}{x_5 \oplus x_9 \oplus y_{12} \oplus y_{15} = k_5 \oplus k_9 \oplus k_{29} \oplus k_{44} \oplus k_{47} \oplus z_0 \oplus 1}. \tag{3.23}$$

Em geral, a equação (3.5) é modificada para

$$\sum_{k \in S_1} x_k \oplus \sum_{l \in S_2} y_l = \left(\sum_{m \in S_3} k_m \right) \oplus \left(\sum_{r \in \Gamma} z_r \right) \oplus t' \tag{3.24}$$

em que Γ é um subconjunto de $\{0, 1, 2, 3\}$. A partir das combinações de equações de cada iteração, obtemos 48 equações que são divididas em 9 grupos, dependendo da combinação de bits caóticos em cada equação. O número de equações em cada grupo é mostrado na Tabela 12. Por exemplo, as 8 equações do grupo z_0 são

$$
\begin{aligned}
x_1 \oplus x_{13} \oplus y_0 \oplus y_3 &= k_1 \oplus k_{13} \oplus k_{17} \oplus k_{32} \oplus \\
&\quad k_{35} \oplus z_0 \\
x_1 \oplus x_{13} \oplus y_4 \oplus y_7 &= k_1 \oplus k_{13} \oplus k_{21} \oplus k_{36} \oplus \\
&\quad k_{39} \oplus z_0 \\
x_5 \oplus x_9 \oplus y_8 \oplus y_{11} &= k_5 \oplus k_9 \oplus k_{25} \oplus k_{40} \oplus \\
&\quad k_{43} \oplus z_0 \\
x_5 \oplus x_9 \oplus y_{12} \oplus y_{15} &= k_5 \oplus k_9 \oplus k_{29} \oplus k_{44} \oplus \\
&\quad k_{47} \oplus z_0 \\
x_1 \oplus x_{13} \oplus y_1 &= k_1 \oplus k_{13} \oplus k_{17} \oplus k_{33} \oplus z_0 \\
x_1 \oplus x_{13} \oplus y_5 &= k_1 \oplus k_{13} \oplus k_{21} \oplus k_{37} \oplus z_0 \\
x_5 \oplus x_9 \oplus y_9 &= k_5 \oplus k_9 \oplus k_{25} \oplus k_{41} \oplus z_0 \\
x_5 \oplus x_9 \oplus y_{13} &= k_5 \oplus k_9 \oplus k_{29} \oplus k_{45} \oplus z_0.
\end{aligned}
$$

As 4 equações do grupo $z_1 \oplus z_2$ são

$$
\begin{aligned}
x_1 \oplus x_3 \oplus x_{12} \oplus x_{13} \oplus y_0 &= k_1 \oplus k_3 \oplus k_{12} \oplus k_{13} \oplus k_{16} \oplus \\
&\quad k_{17} \oplus k_{32} \oplus z_1 \oplus z_2 \\
x_0 \oplus x_1 \oplus x_{13} \oplus x_{15} \oplus y_4 &= k_0 \oplus k_1 \oplus k_{13} \oplus k_{15} \oplus k_{20} \oplus \\
&\quad k_{21} \oplus k_{36} \oplus z_1 \oplus z_2 \\
x_4 \oplus x_5 \oplus x_9 \oplus x_{11} \oplus y_8 &= k_4 \oplus k_5 \oplus k_9 \oplus k_{11} \oplus k_{24} \oplus \\
&\quad k_{25} \oplus k_{40} \oplus z_1 \oplus z_2 \\
x_5 \oplus x_7 \oplus x_8 \oplus x_9 \oplus y_{12} &= k_5 \oplus k_7 \oplus k_8 \oplus k_9 \oplus k_{28} \oplus \\
&\quad k_{29} \oplus k_{44} \oplus z_1 \oplus z_2
\end{aligned}
$$

e as equações do grupo $z_0 \oplus z_1 \oplus z_2$ são

$$\begin{aligned}
x_3 \oplus x_{12} \oplus y_0 \oplus y_1 &= k_3 \oplus k_{12} \oplus k_{16} \oplus k_{32} \oplus k_{33} \oplus \\
&\quad z_0 \oplus z_1 \oplus z_2 \\
x_0 \oplus x_{15} \oplus y_4 \oplus y_5 &= k_0 \oplus k_{15} \oplus k_{20} \oplus k_{36} \oplus k_{37} \oplus \\
&\quad z_0 \oplus z_1 \oplus z_2 \\
x_4 \oplus x_{11} \oplus y_8 \oplus y_9 &= k_4 \oplus k_{11} \oplus k_{24} \oplus k_{40} \oplus k_{41} \oplus \\
&\quad z_0 \oplus z_1 \oplus z_2 \\
x_7 \oplus x_8 \oplus y_{12} \oplus y_{13} &= k_7 \oplus k_8 \oplus k_{28} \oplus k_{44} \oplus k_{45} \oplus \\
&\quad z_0 \oplus z_1 \oplus z_2.
\end{aligned}$$

Tabela 12 – Grupos de equações que dependendo dos bits caóticos para SAES1.

Bits caótiocs	Número de equações
z_0	8
z_1	8
z_2	4
z_3	4
$z_0 \oplus z_1$	4
$z_1 \oplus z_2$	4
$z_2 \oplus z_3$	8
$z_0 \oplus z_1 \oplus z_2$	4
$z_0 \oplus z_2 \oplus z_3$	4

Fonte: O autor (2020).

Considerando que os bits caóticos são variáveis aleatórias independentes e identicamente distribuídas, a probabilidade de que uma equação da forma (3.24) seja satisfeita é $0{,}5$, portanto a CL não pode ser aplicada neste caso. No entanto, há combinações de equações de grupos distintos (listadas na Tabela 12) que permitem obter 32 equações linearmente independentes sem bits caóticos. A seguir, mostra-se as combinações necessárias e calculam-se as probabilidades correspondentes das equações resultantes sem considerar os bits caóticos (esse cálculo é realizado da mesma forma que no algoritmo SAES), já que o objetivo é calcular (depois de todas as combinações) a probabilidade de equações que não envolvam bits caóticos. Por exemplo, do algoritmo SAES, cada equação na Tabela 12 é satisfeita com probabilidade $0{,}5625$. Assim, a adição do módulo 2 de equações dos grupos z_0 e $z_1 \oplus z_2$, produz 32 equações com probabilidade

$$\begin{aligned}
q_1 &\triangleq 2q^2 - 2q + 1 \\
&= 2(0{,}5625)^2 - 2(0{,}5625) + 1 = 0{,}5078
\end{aligned}$$

e adicionando essas equações com as do grupo $z_0 \oplus z_1 \oplus z_2$, obtemos um número suficiente de equações linearmente independentes que não dependem dos bits caóticos, cada uma com

probabilidade

$$\begin{aligned} q_2 &= (0{,}5625)(0{,}5078) + (1 - 0{,}5625)(1 - 0{,}5078) \\ &= 0{,}5009. \end{aligned} \tag{3.25}$$

Usando essa probabilidade, descobrimos que o adversário precisa de $n_p = 2.667.777$ pares de texto claro e texto cifrado (para encontrar esse valor de n_p, substitui-se $q = 0{,}5009$ em (3.14) e prossegue-se de maneira similar ao descrito no parágrafo após (3.15)). Outras combinações de equações podem ser obtidas, mas as probabilidades resultantes são mais próximas de $0{,}5$, o que aumenta o valor de n_p. Em resumo, os bits caóticos selecionam os grupos de equações a serem combinadas, enquanto as probabilidades das equações após as combinações são calculadas da mesma maneira que no SAES. Aplica-se em seguida esta metodologia para o SAES2.

SAES2

O algoritmo SAES2 é uma versão simplificada do SAES1 em que o bloco de mistura de colunas é eliminado. O impacto desta modificação na CL é analisado nesta subseção. Devido à eliminação deste bloco, as equações obtidas na etapa de substituição da segunda iteração são modificadas da seguinte forma (cada equação com probabilidade 0,75):

$$\begin{aligned} \hat{b}_0 \oplus \hat{y}_2 &= \hat{k}_{16} \oplus \hat{k}_{34} \\ \hat{b}_{12} \oplus \hat{y}_6 &= \hat{k}_{20} \oplus \hat{k}_{38} \\ \hat{b}_8 \oplus \hat{y}_{10} &= \hat{k}_{24} \oplus \hat{k}_{42} \\ \hat{b}_4 \oplus \hat{y}_{14} &= \hat{k}_{28} \oplus \hat{k}_{46} \\ \hat{b}_2 \oplus \hat{b}_3 \oplus \hat{y}_3 &= \hat{k}_{18} \oplus \hat{k}_{19} \oplus \hat{k}_{35} \\ \hat{b}_{14} \oplus \hat{b}_{15} \oplus \hat{y}_5 &= \hat{k}_{22} \oplus \hat{k}_{23} \oplus \hat{k}_{39} \\ \hat{b}_{10} \oplus \hat{b}_{11} \oplus \hat{y}_9 &= \hat{k}_{26} \oplus \hat{k}_{27} \oplus \hat{k}_{43} \\ \hat{b}_6 \oplus \hat{b}_7 \oplus \hat{y}_{13} &= \hat{k}_{30} \oplus \hat{k}_{31} \oplus \hat{k}_{47} \end{aligned}$$

$$
\begin{aligned}
\hat{b}_2 \oplus \hat{y}_2 \oplus \hat{y}_3 &= \hat{k}_{18} \oplus \hat{k}_{34} \oplus \hat{k}_{35} \\
\hat{b}_{14} \oplus \hat{y}_6 \oplus \hat{y}_7 &= \hat{k}_{22} \oplus \hat{k}_{38} \oplus \hat{k}_{39} \\
\hat{b}_{10} \oplus \hat{y}_{10} \oplus \hat{y}_{11} &= \hat{k}_{26} \oplus \hat{k}_{42} \oplus \hat{k}_{43} \\
\hat{b}_6 \oplus \hat{y}_{14} \oplus \hat{y}_{15} &= \hat{k}_{30} \oplus \hat{k}_{46} \oplus \hat{k}_{47} \\
\hat{b}_2 \oplus \hat{y}_1 \oplus \hat{y}_3 \oplus &= \hat{k}_{18} \oplus \hat{k}_{33} \oplus \hat{k}_{35} \\
\hat{b}_{14} \oplus \hat{y}_5 \oplus \hat{y}_7 &= \hat{k}_{22} \oplus \hat{k}_{37} \oplus \hat{k}_{39} \\
\hat{b}_{10} \oplus \hat{y}_9 \oplus \hat{y}_{11} &= \hat{k}_{26} \oplus \hat{k}_{41} \oplus \hat{k}_{43} \\
\hat{b}_6 \oplus \hat{y}_{13} \oplus \hat{y}_{15} &= \hat{k}_{30} \oplus \hat{k}_{45} \oplus \hat{k}_{47} \\
\hat{b}_1 \oplus \hat{y}_0 \oplus \hat{y}_3 &= \hat{k}_{17} \oplus \hat{k}_{32} \oplus \hat{k}_{35} \\
\hat{b}_{13} \oplus \hat{y}_4 \oplus \hat{y}_7 &= \hat{k}_{21} \oplus \hat{k}_{36} \oplus \hat{k}_{39} \\
\hat{b}_9 \oplus \hat{y}_8 \oplus \hat{y}_{11} &= \hat{k}_{25} \oplus \hat{k}_{40} \oplus \hat{k}_{43} \\
\hat{b}_5 \oplus \hat{y}_{12} \oplus \hat{y}_{15} &= \hat{k}_{29} \oplus \hat{k}_{44} \oplus \hat{k}_{47} \\
\hat{b}_0 \oplus \hat{b}_1 \oplus \hat{y}_1 \oplus \hat{y}_2 \oplus \hat{y}_3 &= \hat{k}_{16} \oplus \hat{k}_{17} \oplus \hat{k}_{33} \oplus \hat{k}_{34} \oplus \hat{k}_{35} \\
\hat{b}_{12} \oplus \hat{b}_{13} \oplus \hat{y}_5 \oplus \hat{y}_6 \oplus \hat{y}_7 &= \hat{k}_{20} \oplus \hat{k}_{21} \oplus \hat{k}_{37} \oplus \hat{k}_{38} \oplus \hat{k}_{39} \\
\hat{b}_8 \oplus \hat{b}_9 \oplus \hat{y}_9 \oplus \hat{y}_{10} \oplus \hat{y}_{11} &= \hat{k}_{24} \oplus \hat{k}_{25} \oplus \hat{k}_{41} \oplus \hat{k}_{42} \oplus \hat{k}_{43} \\
\hat{b}_4 \oplus \hat{b}_5 \oplus \hat{y}_{13} \oplus \hat{y}_{14} \oplus \hat{y}_{15} &= \hat{k}_{28} \oplus \hat{k}_{29} \oplus \hat{k}_{45} \oplus \hat{k}_{46} \oplus \hat{k}_{47} \\
\hat{b}_3 \oplus \hat{y}_0 &= \hat{k}_{19} \oplus \hat{k}_{32} \\
\hat{b}_{15} \oplus \hat{y}_4 &= \hat{k}_{23} \oplus \hat{k}_{36} \\
\hat{b}_{11} \oplus \hat{y}_8 &= \hat{k}_{27} \oplus \hat{k}_{40} \\
\hat{b}_7 \oplus \hat{y}_{12} &= \hat{k}_{31} \oplus \hat{k}_{44}
\end{aligned}
$$

$$
\begin{aligned}
\hat{b}_0 \oplus \hat{b}_1 \oplus \hat{y}_0 &= \hat{k}_{16} \oplus \hat{k}_{17} \oplus \hat{k}_{32} \\
\hat{b}_{12} \oplus \hat{b}_{13} \oplus \hat{y}_4 &= \hat{k}_{20} \oplus \hat{k}_{21} \oplus \hat{k}_{36} \\
\hat{b}_8 \oplus \hat{b}_9 \oplus \hat{y}_8 &= \hat{k}_{24} \oplus \hat{k}_{25} \oplus \hat{k}_{40} \\
\hat{b}_4 \oplus \hat{b}_5 \oplus \hat{y}_1 2 &= \hat{k}_{28} \oplus \hat{k}_{29} \oplus \hat{k}_{44} \\
\hat{b}_1 \oplus \hat{y}_1 &= \hat{k}_{17} \oplus \hat{k}_{33} \\
\hat{b}_{13} \oplus \hat{y}_5 &= \hat{k}_{21} \oplus \hat{k}_{37} \\
\hat{b}_9 \oplus \hat{y}_9 &= \hat{k}_{25} \oplus \hat{k}_{41} \\
\hat{b}_5 \oplus \hat{y}_{13} &= \hat{k}_{29} \oplus \hat{k}_{45} \\
\hat{b}_0 \oplus \hat{b}_1 \oplus \hat{b}_2 \oplus \hat{b}_3 \oplus \hat{y}_1 &= \hat{k}_{16} \oplus \hat{k}_{17} \oplus \hat{k}_{18} \oplus \hat{k}_{19} \oplus \hat{k}_{33} \\
\hat{b}_{12} \oplus \hat{b}_{13} \oplus \hat{b}_{14} \oplus \hat{b}_{15} \oplus \hat{y}_5 &= \hat{k}_{20} \oplus \hat{k}_{21} \oplus \hat{k}_{22} \oplus \hat{k}_{23} \oplus \hat{k}_{37} \\
\hat{b}_8 \oplus \hat{b}_9 \oplus \hat{b}_{10} \oplus \hat{b}_{11} \oplus \hat{y}_9 &= \hat{k}_{24} \oplus \hat{k}_{25} \oplus \hat{k}_{26} \oplus \hat{k}_{27} \oplus \hat{k}_{41} \\
\hat{b}_4 \oplus \hat{b}_5 \oplus \hat{b}_6 \oplus \hat{b}_7 \oplus \hat{y}_{13} &= \hat{k}_{28} \oplus \hat{k}_{29} \oplus \hat{k}_{30} \oplus \hat{k}_{31} \oplus \hat{k}_{45} \\
\hat{b}_1 \oplus \hat{b}_2 \oplus \hat{y}_0 \oplus \hat{y}_1 &= \hat{k}_{18} \oplus \hat{k}_{32} \oplus \hat{k}_{33} \\
\hat{b}_{13} \oplus \hat{b}_{14} \oplus \hat{y}_4 \oplus \hat{y}_5 &= \hat{k}_{22} \oplus \hat{k}_{36} \oplus \hat{k}_{37} \\
\hat{b}_8 \oplus \hat{b}_9 \oplus \hat{y}_8 \oplus \hat{y}_9 &= \hat{k}_{26} \oplus \hat{k}_{40} \oplus \hat{k}_{41} \\
\hat{b}_4 \oplus \hat{b}_5 \oplus \hat{y}_{12} \oplus \hat{y}_{13} &= \hat{k}_{30} \oplus \hat{k}_{44} \oplus \hat{k}_{45} \\
\hat{b}_0 \oplus \hat{y}_0 \oplus \hat{y}_1 &= \hat{k}_{16} \oplus \hat{k}_{32} \oplus \hat{k}_{33} \\
\hat{b}_{12} \oplus \hat{y}_4 \oplus \hat{y}_5 &= \hat{k}_{20} \oplus \hat{k}_{36} \oplus \hat{k}_{37} \\
\hat{b}_8 \oplus \hat{y}_8 \oplus \hat{y}_9 &= \hat{k}_{24} \oplus \hat{k}_{40} \oplus \hat{k}_{41} \\
\hat{b}_4 \oplus \hat{y}_{12} \oplus \hat{y}_{13} &= \hat{k}_{28} \oplus \hat{k}_{44} \oplus \hat{k}_{45}.
\end{aligned}
$$

Realizando as mesmas operações que conduziram a (3.17) e que combinadas com equações em (3.7) produzem 48 equações divididas em 4 grupos (cada equação com probabilidade $0{,}625$) dependendo da presença de bits caóticos, como mostra a Tabela 13. Uma combinação (soma módulo 2) dos grupos $\{z_0 \oplus z_2\}$ e $\{z_1 \oplus z_3\}$ conduz a 96 equações do grupo $z_0 \oplus z_1 \oplus z_2 \oplus z_3$ com probabilidade $0{,}53125$, o que combinadas com equações do grupo $z_0 \oplus z_1 \oplus z_2 \oplus z_3$, obtém-se equações que não dependem dos bits caóticos, com probabilidade $0{,}5078$, precisando o adversário de $n_p = 35.518$ pares de texto claro e texto cifrado.

Tabela 13 – Grupos de equações dependendo dos bits caóticos para o SAES2.

Bits caóticos	**Quantidade de equações**
Não	20
$z_0 \oplus z_2$	12
$z_1 \oplus z_3$	8
$z_0 \oplus z_1 \oplus z_2 \oplus z_3$	8

Fonte: O autor (2020).

SAES3

A eliminação do bloco mistura de colunas no caminho de dados do SAES2 acarreta uma perda da difusão de bits de S-boxes distintas. No algoritmo AES3, os blocos deslocamento de linhas e mistura de colunas são substituídos pelo bloco mistura aleatória. Realiza-se um deslocamento cíclico à direita aleatório nos bits de saída das 4 S-boxes $(b_0, \ldots, b_{15})$ do caminho de dados dependendo dos bits caóticos z_0z_1, podendo-se realizar quatro deslocamentos possíveis, $z_0z_1 = 00$ sem deslocamento, $z_0z_1 = 01$ realiza-se o deslocamento de um bit, $z_0z_1 = 10$ deslocamento de dois bits e 11 deslocamento de três bits. Cada deslocamento ocorre com a mesma probabilidade, $1/4$, e para cada um deles existem 48 equações possíveis. A Tabela 14 mostra a quantidade de equações que dependem dos bits caóticos para cada deslocamento, em que equações têm probabilidades $p = 0{,}625$ ou $p = 0{,}5625$.

Tabela 14 – Grupos de equações dependendo dos bits caóticos para o SAES3.

Deslocamento	Bits caóticos	Números de equações	Probabilidade
00	Não	20	0,625
	$z_0 \oplus z_2$	12	0,625
	$z_1 \oplus z_3$	8	0,625
	$z_0 \oplus z_1 \oplus z_2 \oplus z_3$	8	0,625
01	z_0	4	0,5625
	z_1	4	0,5625
	z_2	4	0,5625
	$z_0 \oplus z_1$	8	0,5625
	$z_1 \oplus z_2$	4	0,5625
	$z_1 \oplus z_3$	4	0,625
	$z_2 \oplus z_3$	4	0,5625
	$z_1 \oplus z_2 \oplus z_3$	4	0,625
	$z_0 \oplus z_2 \oplus z_3$	4	0,625
	$z_0 \oplus z_1 \oplus z_3$	8	0,5625
10	z_1	8	0,625
	z_3	8	0,625
	$z_0 \oplus z_2$	4	0,5625
	$z_0 \oplus z_3$	8	0,5625
	$z_1 \oplus z_2$	8	0,5625
	$z_0 \oplus z_1 \oplus z_2$	8	0,625
	$z_0 \oplus z_1 \oplus z_2 \oplus z_3$	4	0,5625
11	z_0	8	0,5625
	z_2	8	0,5625
	$z_0 \oplus z_1$	4	0,5625
	$z_2 \oplus z_3$	4	0,5625
	$z_0 \oplus z_3$	4	0,5625
	$z_0 \oplus z_1 \oplus z_3$	8	0,5625
	$z_0 \oplus z_2 \oplus z_3$	4	0,5625
	$z_1 \oplus z_2 \oplus z_3$	8	0,625

Fonte: O autor (2020).

Um conjunto de 32 equações linearmente independentes pode ser obtido a partir da combinação de equações que são afetadas por bits caóticos. Por exemplo, uma combinação dos grupos z_0 e z_1 na Tabela 14, com deslocamento 01 resulta em 16 equações que dependem de $z_1 \oplus z_0$ com probabilidade $0{,}5078$. Combinando estas 16 equações com 8 equações que dependem de $z_0 \oplus z_1$ (para o mesmo deslocamento) resulta em 128 equações, sendo possível extrair 32 equações linearmente independentes com probabilidade $0{,}5009$. Este procedimento também pode ser executado para os deslocamentos 10 e 11 com a determinação de 32 equações linearmente independentes com probabilidade $0{,}5009$ que não dependem dos bits caóticos. Os procedimentos realizados com o deslocamento 00 são semelhantes aos do algoritmo SAES2, e as 32 equações apresentam probabilidade $0{,}5078$. Assim, o valor médio da probabilidade de obter 32 equações linearmente independentes que não dependem dos bits caóticos é

$$q = \frac{1}{4}(0{,}5078) + \frac{3}{4}(0{,}5009) = 0{,}5026. \tag{3.26}$$

Seguindo os passo descritos, o adversário precisa de $n_p = 319.660$ pares de texto claro e texto cifrado para achar bits da chave com certa confiabilidade. A Tabela 15 apresenta uma comparação dos valores da probabilidade de equações linearmente independentes que são satisfeitas corretamente e do número de pares texto claro e texto cifrado que é necessário para que a CL seja bem sucedida com uma confiabilidade de 95 %. A introdução de bits caóticos leva a um aumento considerável na quantidade de pares de texto claro e texto cifrado necessários em comparação com o algoritmo SAES. O algoritmo SAES1 apresenta o melhor desempenho para esta análise, mas também é o algoritmo mais complexo, realizando um maior número de operações. O algoritmo SAES3 apresenta boa robustez contra a CL sendo o que realiza o menor número de operações.

Tabela 15 – Pares de texto claro e texto cifrado para quebrar os algoritmos propostos.

Algoritmo	Probabilidade	n_p
SAES	0,5625	545
SAES1	0,5009	2.667.777
SAES2	0,5078	35.518
SAES3	0,5026	319.660

Fonte: O autor (2020).

3.3 CRIPTOANÁLISE DIFERENCIAL DO SAES

A CD é um ataque de texto claro escolhido, em que uma chave é utilizada para cifrar dois textos claros. É assumido que os textos cifrados também estão disponíveis para o adversário. O adversário geralmente escolhe um par de textos claros que diferem em bits específicos e observa o comportamento do algoritmo para achar bits da chave, reduzindo o espaço de chaves.

Para a análise da CD no sistema SAES, escolhe-se um par de texto claro $\{x_0, x_1, \ldots, x_{15}\}$ e $\{x_0^*, x_1^*, \ldots, x_{15}^*\}$ em que $x_0 \neq x_0^*$ e $x_i = x_i^*$, para $1 \leq i \leq 15$. Cada par de texto claro é cifrado com a mesma chave $\{k_0, \ldots, k_{15}\}$ resultando em um par de texto cifrado $\{y_0, \ldots, y_{15}\}$ e $\{y_0^*, \ldots, y_{15}^*\}$. Na saída do primeiro bloco de substituição, $b_i = b_i^*$ para $4 \leq i \leq 15$. Sabendo que $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1000$, existem oito possíveis valores de $x_0x_1x_2x_3$ que satisfazem a esta condição, sendo que dois pares fornecem uma soma XOR na saída da primeira S-Box igual a 1111 ($b_0b_1b_2b_3 \oplus b_0^*b_1^*b_2^*b_3^* = 1111$) enquanto os outros seis pares fornecem uma soma XOR na saída com valores distintos. Este comportamento é mostrado na Tabela 16. Desta forma, $b_0b_1b_2b_3 \oplus b_0^*b_1^*b_2^*b_3^* = 1111$ ocorre com probabilidade $1/4$.

Tabela 16 – Diferenças na saída de uma S-Box para pares de entrada que satisfazem $x_0x_1x_2x_3 \oplus x_0^*x_1^*x_2^*x_3^* = 1000$.

$x_0x_1x_2x_3$ $x_0^*x_1^*x_2^*x_3^*$	$b_0b_1b_2b_3$ $b_0^*b_1^*b_2^*b_3^*$	$b_0b_1b_2b_3 \oplus b_0^*b_1^*b_2^*b_3^*$	$x_0x_1x_2x_3$ $x_0^*x_1^*x_2^*x_3^*$	$b_0b_1b_2b_3$ $b_0^*b_1^*b_2^*b_3^*$	$b_0b_1b_2b_3 \oplus b_0^*b_1^*b_2^*b_3^*$
1000 0000	0110 1001	**1111**	1100 0100	1100 1101	0001
1001 0001	0010 0100	0110	1101 0101	1110 0001	**1111**
1010 0010	0000 1010	1010	1110 0110	1111 1000	0111
1011 0011	0011 1011	1000	1111 0111	0111 0101	0010

Fonte: O autor (2020).

Considerando que a etapa de deslocamento de linhas não afeta a saída desta S-Box, e utilizando a mistura de coluna descrita na Seção 3.1 e considerando que a mesma chave é usada em cada par, obtém-se na saída da primeira iteração.

$$\begin{aligned}(a_0', a_1', a_2', a_3') &= (b_0 \oplus b_{14} \oplus k_{16}, b_1 \oplus b_{12} \oplus b_{15} \oplus k_{17}, b_2 \oplus b_{12} \oplus b_{13} \oplus k_{18}, \\ &\quad b_3 \oplus b_{13} \oplus k_{19}) \qquad (3.27)\\ (a_0'^*, a_1'^*, a_2'^*, a_3'^*) &= (b_0^* \oplus b_{14}^* \oplus k_{16}, b_1^* \oplus b_{12}^* \oplus b_{15}^* \oplus k_{17}, b_2^* \oplus b_{12}^* \oplus b_{13}^* \oplus k_{18}, \\ &\quad b_3^* \oplus b_{13}^* \oplus k_{19}) \qquad (3.28)\end{aligned}$$

em que $b_i = b_i^*$ para $i = 4, \ldots, 15$. Somando ambas equações, obtém-se $(a_0', a_1', a_2', a_3') \oplus (a_0'^*, a_1'^*, a_2'^*, a_3'^*) = (b_0 \oplus b_0^*, b_1 \oplus b_1^*, b_2 \oplus b_2^*, b_3 \oplus b_3^*)$. Então, a diferença na entrada da primeira S-Box na segunda etapa de substituição é $(a_0', a_1', a_2', a_3') \oplus (a_0'^*, a_1'^*, a_2'^*, a_3'^*) = 1111$, e ocorre

com probabilidade igual a $1/4$. Assumindo que:

$$\begin{aligned}(a'_0, a'_1, a'_2, a'_3) \oplus (a'^*_0, a'^*_1, a'^*_2, a'^*_3) &= 1111 \\ (a'^*_0, a'^*_1, a'^*_2, a'^*_3) &= (a'_0, a'_1, a'_2, a'_3) \oplus 1111\end{aligned} \tag{3.29}$$

então

$$S(a'_0, a'_1, a'_2, a'_3) \oplus S((a'_0, a'_1, a'_2, a'_3) \oplus 1111) = (y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3) \tag{3.30}$$

ocorre com probabilidade $1/4$. Dado que o texto cifrado é conhecido, existem 16 possíveis valores de $(y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3)$. Por exemplo, para um valor fixo de $(y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3)$ igual a 1010, observa-se na Tabela 17 que para os 16 possíveis pares (a'_0, a'_1, a'_2, a'_3) e $(a'^*_0, a'^*_1, a'^*_2, a'^*_3)$ que produzem na saída $(y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3) = 1010$, 2 pares satisfazem (3.29). Repetindo esta análise os outros valores de $(y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3)$, observa-se que em 12 casos ($3/4$ das vezes) 2 de pares de entrada satisfazem (3.29) enquanto para outros 4 casos ($1/4$ das vezes) 4 pares satisfazem (3.29).

Tabela 17 – Pares de entrada de uma S-Box que satisfazem $(y_0, y_1, y_2, y_3) \oplus (y^*_0, y^*_1, y^*_2, y^*_3) = 1010$.

(y_0, y_1, y_2, y_3)	$(y^*_0, y^*_1, y^*_2, y^*_3)$	(a'_0, a'_1, a'_2, a'_3)	$a'^*_0, a'^*_1, a'^*_2, a'^*_3)$	$(a'_0, a'_1, a'_2, a'_3) \oplus (a'^*_0, a'^*_1, a'^*_2, a'^*_3)$
0000	1010	1010	0010	1000
0001	1011	0101	0011	0110
0010	1000	1001	0110	**1111**
0011	1001	1011	0000	1011
0100	1110	0001	1101	1100
0101	1111	0111	1110	1001
0110	1100	1000	1100	0100
0111	1101	1111	0100	1011
1000	0010	0110	1001	**1111**
1001	0011	0000	1011	1011
1010	0000	0010	1010	1000
1011	0001	0011	0101	0110
1100	0110	1100	1000	0100
1101	0111	0100	1111	1011
1110	0100	1101	0001	1100
1111	0101	1110	0111	1001

Fonte: O autor (2020).

Portanto em $3/4$ das vezes, existem duas possibilidades de satisfazer (3.29), sendo uma delas a solução correta. Assim, este resultado ocorre com probabilidade $(1/4)(3/4)(1/2) = 3/32$. Em $1/4$ das vezes, existem quatro possibilidades de satisfazer (3.29), sendo uma delas a solução correta e esse caso ocorre com probabilidade $(1/4)(1/4)(1/4) = 1/64$. Portanto, entre todos os conjuntos de candidatos dos bits da chave, espera-se que a escolha correta ocorra com probabilidade $P_c = 1/64 + 3/32 = 7/64$.

3.3.1 Análise dos Algoritmos Propostos

Nos algoritmos propostos neste capítulo, introduz-se bits caóticos nas saídas das S-boxes. Para analisar o efeito da CD com a presença destes bits caóticos, utilizam-se dois bits caóticos distintos para cada S-box no caminho de dados do algoritmo, enquanto para simplificar a análise, estes não são utilizados na geração de sub-chaves. No caminho de dados, na primeira iteração, existem 4 S-boxes, sendo utilizados 8 bits caóticos, $z_0, z_1, z_2, z_3, z_4, z_5, z_6, z_7$. Um aspecto relevante é que o adversário não tem a capacidade de reinicializar o sistema para que os bits caóticos utilizados na obtenção de $y_0, y_1, \ldots, y_{15}$ e de $y_0^*, y_1^*, \ldots, y_{15}^*$ sejam iguais.

Considerando uma mudança de um bit no texto claro, $x_0 \neq x_0^*$ e $x_i = x_i^*$ para $1 \leq i \leq 15$, na saída da primeira etapa de substituição, todos os bits podem ser mudados devido à influência dos bits caóticos. Conforme é mostrado em (3.16), as saída das S-boxes são $\{b_0 \oplus z_1, b_1 \oplus z_0, b_2 \oplus z_1, b_3 \oplus z_0 \oplus z_1, \ldots, b_{12} \oplus z_7, b_{13} \oplus z_6, b_{14} \oplus z_7, b_{15} \oplus z_6 \oplus z_7\}$. Então, repetindo os procedimentos que conduziram a (3.27) e (3.28), obtém-se

$$
\begin{aligned}
(a'_0, \ldots, a'_{15}) \oplus (a'^*_0, \ldots, a'^*_{15}) &= (c_0 \oplus k_{16}, c_1 \oplus k_{17}, \ldots, c_{15} \\
&\quad \oplus k_{31}) \oplus (c_0^* \oplus k_{16}, c_1^* \oplus k_{17}, \ldots, c_{15}^* \oplus k_{31}) \\
&= (b_0 \oplus b_{14} \oplus k_{16} \oplus v'_0, b_1 \oplus b_{12} \oplus b_{15} \oplus k_{17} \oplus v'_1, \\
&\quad b_2 \oplus b_{12} \oplus b_{13} \oplus k_{18} \oplus v'_2, b_3 \oplus b_{13} \oplus k_{19} \oplus v'_3, \ldots, \\
&\quad b_7 \oplus b_9 \oplus k_{31} \oplus v'_{15}) \oplus (b_0^* \oplus b_{14}^* \oplus k_{16} \oplus v_0^*, b_1^* \\
&\quad \oplus b_{12}^* \oplus b_{15}^* \oplus k_{17} \oplus v_1^*, \\
&\quad b_2^* \oplus b_{12}^* \oplus b_{13}^* \oplus k_{18} \oplus v_2^*, b_3^* \oplus b_{13}^* \oplus k_{19} \oplus v_3^*, \\
&\quad \ldots, b_7^* \oplus b_9^* \oplus k_{31} \oplus v_{15}^*) \\
&= (b_0 \oplus b_0^*, b_1 \oplus b_1^*, b_2 \oplus b_2^*, b_3 \oplus b_3^*, 0, 0, \ldots, 0) \oplus \\
&\quad (v_0, v_1, v_2, v_3, v_4, \ldots, v_{15})
\end{aligned}
\tag{3.31}
$$

sendo v_i, para $i = 0, \ldots, 15$, definidos por:

$$
\begin{aligned}
v_0 &= v'_0 \oplus v^*_0 = z_1 \oplus z_7 \oplus z^*_1 \oplus z^*_7 \\
v_1 &= v'_1 \oplus v^*_1 = z_0 \oplus z_6 \oplus z^*_0 \oplus z^*_6 \\
v_2 &= v'_2 \oplus v^*_2 = z_1 \oplus z_6 \oplus z_7 \oplus z^*_1 \oplus z^*_6 \oplus z^*_7 \\
v_3 &= v'_3 \oplus v^*_3 = z_0 \oplus z_1 \oplus z_6 \oplus z^*_0 \oplus z^*_1 \oplus z^*_6 \\
v_4 &= v'_4 \oplus v^*_4 = z_1 \oplus z_7 \oplus z^*_1 \oplus z^*_7 \\
v_5 &= v'_5 \oplus v^*_5 = z_0 \oplus z_6 \oplus z^*_0 \oplus z^*_6 \\
v_6 &= v'_6 \oplus v^*_6 = z_0 \oplus z_1 \oplus z_7 \oplus z^*_0 \oplus z^*_1 \oplus z^*_7 \\
v_7 &= v'_7 \oplus v^*_7 = z_0 \oplus z_6 \oplus z_7 \oplus z^*_0 \oplus z^*_6 \oplus z^*_7 \\
v_8 &= v'_8 \oplus v^*_8 = z_3 \oplus z_5 \oplus z^*_3 \oplus z^*_5 \\
v_9 &= v'_9 \oplus v^*_9 = z_2 \oplus z_4 \oplus z^*_2 \oplus z^*_4 \\
v_{10} &= v'_{10} \oplus v^*_{10} = z_2 \oplus z_3 \oplus z_5 \oplus z^*_2 \oplus z^*_3 \oplus z^*_5 \\
v_{11} &= v'_{11} \oplus v^*_{11} = z_2 \oplus z_4 \oplus z_5 \oplus z^*_2 \oplus z^*_4 \oplus z^*_5 \\
v_{12} &= v'_{12} \oplus v^*_{12} = z_3 \oplus z_5 \oplus z^*_3 \oplus z^*_5 \\
v_{13} &= v'_{13} \oplus v^*_{13} = z_2 \oplus z_4 \oplus z^*_2 \oplus z^*_4 \\
v_{14} &= v'_{14} \oplus v^*_{14} = z_3 \oplus z_4 \oplus z_5 \oplus z^*_3 \oplus z^*_4 \oplus z^*_5 \\
v_{15} &= v'_{15} \oplus v^*_{15} = z_2 \oplus z_3 \oplus z_4 \oplus z^*_2 \oplus z^*_3 \oplus z^*_4.
\end{aligned}
$$

Na saída da primeira iteração, dado que

$$(x_0, x_1, x_2, \ldots, x_{15}) \oplus (x^*_0, x^*_1, x^*_2, \ldots, x^*_{15}) = (1, 0, 0, \ldots, 0)$$

exitem 2^{12} valores de $(a'_0, \ldots, a'_{15}) \oplus (a'^*_0, \ldots, a'^*_{15})$ em (3.31), uma vez que $v_0 = v_4$, $v_1 = v_5$, $v_8 = v_{12}$, $v_9 = v_{13}$. A modificação de um bit do texto claro ($x_0 \neq x^*_0$) pode modificar os 16 bits na saída da primeira iteração, devido aos bits caóticos.

Fazendo uma análise similar à realizada para o algoritmo SAES, quando as saídas das S-boxes são somadas com as possíveis sequências **h**, como é mostrado em (3.16), obtém-se 256 possíveis valores de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3 \oplus b^*_0, \ldots, b^*_3 \oplus h^*_0, \ldots, h^*_3$, na saída da primeira S-box. A Tabela 18 ilustra os quatro possíveis valores de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3$ e $b^*_0, \ldots, b^*_3 \oplus h^*_0, \ldots, h^*_3$, sendo estes valores equiprováveis, para o caso $x_0, x_1, x_2, x_3 = 0000$ e $x^*_0, x^*_1, x^*_2, x^*_3 = 1000$ ($x_0 \neq x^*_0$).

Tabela 18 – Possíveis valores de saída da S-box dado $x_0 \neq x_0^*$.

$x_0, \ldots, x_3$	$b_0, \ldots, b_3$	$h_0, \ldots, h_3$	$b_0, \ldots, b_3 \oplus h_0, \ldots, h_3$
0000	1001	0000	1001
		1011	0010
		0101	1100
		1110	0111
$x_0^*, \ldots, x_3^*$	$b_0^*, \ldots, b_3^*$	$h_0^*, \ldots, h_3^*$	$b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$
1000	0110	0000	0110
		1011	1101
		0101	0011
		1110	1000

Fonte: O autor (2020).

A Tabela 19 mostra as possíveis combinações de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3$ com $b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$ mostradas na Tabela 18, formando os valores de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3 \oplus b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$. Observa-se nesta tabela quatro valores distintos entre os 16 possíveis valores, para entrada considerada, todos equiprováveis. Este comportamento é observado para todos os possíveis $x_0, \ldots, x_3$ e $x_0^*, \ldots, x_3^*$ que diferem no primeiro bit. Então ocorrem 16 possíveis valores distintos de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3 \oplus b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$, todos equiprováveis com probabilidade $16/256$.

O valor de $b_4, \ldots, b_7 \oplus h_4, \ldots, h_7 \oplus b_4^*, \ldots, b_7^* \oplus h_4^*, \ldots, h_4^*$ da segunda S-box, é igual a 0000 com probabilidade $1/4$ (isto ocorre quando $h_0, \ldots, h_3 \oplus h_0^*, \ldots, h_3^* = 0000$). Uma análise similar pode ser feita para a terceira e quarta S-box. Portanto, na saída da primeira iteração $(a'_0, \ldots, a'_{15}) \oplus (a'^*_0, \ldots, a'^*_{15}) = 1111000000000000$ ocorre com probabilidade $\frac{1}{16}\frac{1}{4}\frac{1}{4}\frac{1}{4} = \frac{1}{1024}$. Usando a Tabela 16, este valor é obtido no algoritmo SAES com probabilidade $1/4$.

Tabela 19 – Valores de $b_0, \ldots, b_3 \oplus h_0, \ldots, h_3 \oplus b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$ para uma entrada com $x_0 \neq x_0^*$.

$b_0, \ldots, b_3 \oplus h_0, \ldots, h_3$	$b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$	$b_0, \ldots, b_3 \oplus h_0, \ldots, h_3 \oplus b_0^*, \ldots, b_3^* \oplus h_0^*, \ldots, h_3^*$
1001	0110	1111
	1101	0100
	0011	1010
	1000	0001
0010	0110	0100
	1101	1111
	0011	0001
	1000	1010
1100	0110	1010
	1101	0001
	0011	1111
	1000	0100
0111	0110	0001
	1101	1010
	0011	0100
	1000	1111

Fonte: O autor (2020).

Na entrada da primeira S-box na segunda iteração:

$$(a'_0,\ldots,a'_3)\oplus(a'^*_0,\ldots,a'^*_3) \;=\; (b_0\oplus b^*_0\oplus v'_0, b_1\oplus b^*_1\oplus v'_1, b_2\oplus b^*_2\oplus v'_2, b_3\oplus b^*_3\oplus v'_3)$$

obtendo na saída da segunda iteração:

$$\begin{aligned}(y_0,y_1,y_2,y_3)\oplus(y^*_0,y^*_1,y^*_2,y^*_3) &= S(a'_0,a'_1,a'_2,a'_3)\oplus S(a'_0,a'_1,a'_2,a'_3\oplus 1111)\\ &\quad\oplus(h_0,\ldots,h_3\oplus h^*_0,\ldots,h^*_3)\end{aligned} \tag{3.32}$$

ou

$$\begin{aligned}S(a'_0,a'_1,a'_2,a'_3)\oplus S(a'_0,a'_1,a'_2,a'_3\oplus 1111) &= (h_0,\ldots,h_3\oplus h^*_0,\ldots,h^*_3)\\ &\quad\oplus(y_0,y_1,y_2,y_3)\oplus(y^*_0,y^*_1,y^*_2,y^*_3).\end{aligned} \tag{3.33}$$

Existem 64 possíveis $(y_0,y_1,y_2,y_3)\oplus(y^*_0,y^*_1,y^*_2,y^*_3)\oplus(h_0,\ldots,h_3\oplus h^*_0,\ldots,h^*_3)$ combinações entre $(h_0,\ldots,h_3\oplus h^*_0,\ldots,h^*_3)$ e $(y_0,y_1,y_2,y_3)\oplus(y^*_0,y^*_1,y^*_2,y^*_3)$, sendo cada um dos 16 possíveis valores distintos acontecem com probabilidade $4/64$. Para cada um desses 16 valores distintos de existe um comportamento similar ao do algoritmo SAES, existindo em $3/4$ das vezes duas possibilidades de satisfazer $a'^*_0,a'^*_1,a'^*_2,a'^*_3 = a'_0,a'_1,a'_2,a'_3\oplus 1111$ e (3.32). Em $1/4$ das vezes, quatro valores satisfazem (3.32). Para a segunda S-box, obtém-se $S(a'_4,\ldots,a'_7)\oplus S(a'^*_4,\ldots,a'^*_7)=0000$, sendo (3.32) reescrita da seguinte forma:

$$\begin{aligned}S(a'_4,a'_5,a'_6,a'_7)\oplus S(a'^*_4,a'^*_5,a'^*_6,a'^*_7) &= (h_3,\ldots,h_7)\oplus(h^*_3,\ldots,h^*_7)\\ &\quad\oplus(y_4,y_5,y_6,y_7)\oplus(y^*_4,y^*_5,y^*_6,y^*_7)\end{aligned}$$

ou

$$(y_4,y_5,y_6,y_7)\oplus(y^*_4,y^*_5,y^*_6,y^*_7) \;=\; (h_3,\ldots,h_7)\oplus(h^*_3,\ldots,h^*_7).$$

As quatro possíveis sequências $(z_3\oplus z^*_3, z_2\oplus z^*_2, z_3\oplus z^*_3, z_2\oplus z^*_2\oplus z_3\oplus z^*_3)$ ocorrem com probabilidade $4/16$. Portanto, entre os conjuntos dos bits da chave, espera-se que os bits corretos ocorram com probabilidade $P_c = \frac{1}{16}\frac{3}{4}\frac{1}{2}\frac{1}{4}\frac{1}{4}\frac{1}{4} + \frac{1}{16}\frac{1}{4}\frac{1}{4}\frac{1}{4}\frac{1}{4}\frac{1}{4} = \frac{7}{16384}$.

Convém ressaltar que a etapa mistura de colunas não afeta este resultado, portanto sob a perspectiva da CD, os algoritmos SAES2 e SAES3 apresentam o mesmo resultado obtido pelo algoritmo SAES1. A introdução de bits caóticos acarreta uma diminuição considerável na probabilidade de uma escolha correta de bits da chave, sendo para o algoritmo SAES esta probabilidade é $7/64$, enquanto os algoritmos propostos é $7/16384$.

4 SISTEMA DE MARCA D'ÁGUA EM IMAGENS UTILIZANDO SEQUÊNCIAS CAÓTICAS

Propõe-se neste capítulo um novo algoritmo para inserção e extração de marca d'água em imagens em escala de cinza e compara-se as métricas de imperceptibilidade e robustez do algoritmo proposto com as de alguns algoritmos existentes na literatura para vários ataques.

4.1 MARCA D'ÁGUA

Os sistemas de marcas d'água digitais em imagens incluem dois componentes principais: esquema de inserção e de extração de marca d'água. Cada sistema de marca d'água deve ter propriedades particulares para uma dada aplicação. Portanto, não há um conjunto exclusivo de propriedades que todos os sistemas de marca d'água tenham que satisfazer. Em geral, existem várias questões importantes que são geralmente consideradas em aplicações práticas (KATZENBEISSER; PETITCOLAS, 1999).

4.1.1 Visibilidade

Marca d'água visível Consiste na sobreposição de uma imagem de marca d'água visível à imagem original, de modo que pode-se executar a proteção de direitos autorais (KATZENBEISSER; PETITCOLAS, 1999).

Marca d'água invisível Consiste em inserir uma imagem de marca d'água digital de modo que exista uma semelhança visual entre a versão com marca d'água e a imagem original. Dois tipos de marca d'água invisível são: marca d'água frágil e marca d'água robusta.

- Frágil: Um esquema de marca d'água frágil deve ser capaz de detectar qualquer alteração na imagem com marca d'água, identificar onde ocorreu e recuperar a imagem original. Serve para atestar a autenticidade de uma imagem.
- Robusta: Imagem com marca d'água devem ser resistente a vários ataques e operações comuns em processamento de sinais. Uma vez que uma imagem de marca d'água é inserida em uma imagem original, esta deve ser recuperada em presença de distorções que inevitavelmente ocorrem quando a imagem é processada, comprimida e transmitida através de um canal ruidoso. Marcas d'água em imagens, geralmente, devem ser resistentes à adição de ruído, filtragem, transformações geométricas (por exemplo, escalonamento, rotação, translação) e também à compressão JPEG.

4.1.2 Processo de Extração

Marca d'água não cega Estes algoritmos exigem o conhecimento de dados originais no processo de extração, como a imagem original ou a imagem de marca d'água.

Marca d'água cega O processo de extração não exige o conhecimento de dados originais. Espera-se que o uso deste tipo de esquema seja menos robusto que algoritmos não cegos, uma vez que não requer acesso a dados originais.

4.1.3 Capacidade

A capacidade é definida como a quantidade de informação que as marcas d'água inseridas são capazes de ocultar, podendo ser extraídas com credibilidade. Para imagens, a capacidade refere-se à quantidade de bits incorporados por pixel. Em linhas gerais, quanto menor a capacidade, maior imperceptibilidade e menor robustez.

4.1.4 Código BCH

Códigos corretores de erro são utilizados em sistemas de marca d'água em imagens (TAN et al., 2019; LEFEVRE; CARRE; GABORIT, 2019; SARRESHTEDARI; AKHAEE; ABBASFAR, 2018; FAN; CHAO; CHIEU, 2019), dando robustez ao algoritmo de marca d'água. A introdução do código permite corrigir erros introduzidos por ruídos em diversos ataques bem como por um canal de comunicação ruidoso. Utiliza-se neste capítulo o código BCH sobre GF(q) com palavras código de comprimento $n = q^m - 1$ (sendo m um inteiro), k bits de informação, $n - k$ bits de paridade e capacidade de correção t bits, denominado de BCH(n, k, t). Este código é completamente especificado por seu polinômio gerador $g(x) = 1 + g_1 x + \cdots + g_{n-k-1}x^{n-k-1} + x^{n-k}$, em que $g_i \in$ GF(q). O grau de $g(x)$ é igual ao número de bits de paridade do código. Uma representação polinomial $c(x)$ de uma palavra de código $\mathbf{c} = (c_0, \ldots, c_{n-1})$ é da forma $c(x) = c_0 + c_1 x + \cdots + c_{n-1}x^{n-1}$. A operação de codificação pode ser expressa na forma polinomial $c(x) = g(x)\, u(x)$, em que $u(x)$ é a mensagem a ser codificada e as operações com polinômios seguem as regras da operação definidas sobre o corpo. O algoritmo Berlekamp-Massey (BM) é um algoritmo de decodificação algébrico para o código BCH.

Neste trabalho utiliza-se $q = 2$ e $m = 4$, sendo o comprimento da palavra código $n = 15$, $k = 11$, tendo 4 bits de paridade e $t = 1$. Seja α um elemento primitivo de GF(2^4), e seja $m_i(x)$ o polinômio minimal de α^i em GF(2^4). O polinômio gerador $g(x)$ é obtido com o mínimo múltiplo comum dos polinômios minimais $g(x) = \text{mmc}(m_1(x), m_2(x), \ldots, m_{d-1}(x))$, sendo d a distância mínima do código. Para capacidade de correção de um bit, o polinômio gerador é dado por:

$$\begin{aligned} g(x) &= \text{mmc}(m_1(x), m_2(x), m_3(x), m_4(x)) \\ &= (x^4 + x + 1) \end{aligned} \tag{4.1}$$

obtendo um código BCH$(15, 11, 1)$.

4.2 ALGORITMO PROPOSTO

Apresenta-se nesta seção um novo esquema de inserção e extração de marca d'água. O algoritmo de inserção $E(\cdot)$ tem como entrada uma imagem original C_O, uma chave K e uma imagem de marca d'água W, obtendo na saída a imagem com marca d'água C_W, que é descrita da seguinte forma:

$$C_W = E(C_O, W, K).$$

Utiliza-se um esquema de extração cega, em que a entrada do algoritmo de extração $E^{-}(\cdot)$ é a imagem com marca d'água possivelmente corrompida devido a ataques, denominada de C'_W e a chave K. A saída é a imagem de marca d'água extraída W' que não é necessariamente igual a W, isto é

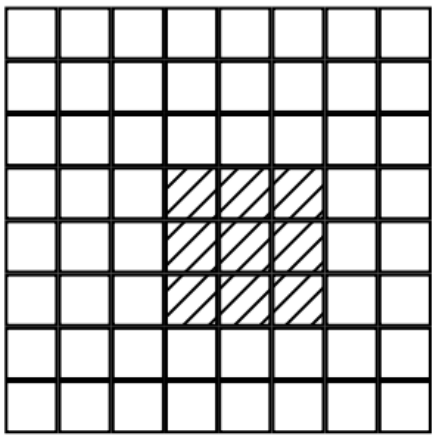
$$W' = E^{-}(C'_W, K).$$

A imagem original C_O em escala de cinza de tamanho 512×512 píxeis é dividida em blocos não sobrepostos de 8×8 píxeis (o número de blocos de uma imagem é 4096) e a DCT bidimensional é executada em cada bloco. Considerando $C_O^i(x, y)$ o valor do pixel (x, y) do i-ésimo bloco na imagem original, a versão bidimensional da DCT deste bloco é da forma (KATZENBEISSER; PETITCOLAS, 1999):

$$X_O^i(u, v) = \alpha(u)\alpha(v) \sum_{x=0}^{N_b-1} \sum_{y=0}^{M_b-1} C_O^i(x, y) \cos\left(\frac{\pi u(2x+1)}{2N_b}\right) \cos\left(\frac{\pi v(2y+1)}{2M_b}\right) \quad (4.2)$$

sendo $\alpha(u) = 1/\sqrt{N_b}$ para $u = 0$, caso contrário $\alpha(u) = \sqrt{2/N_b}$, e de forma semelhante $\alpha(v) = 1/\sqrt{M_b}$ para $v = 0$, caso contrário $\alpha(v) = \sqrt{2/M_b}$, $0 \leq u \leq N_b - 1$ e $0 \leq v \leq M_b - 1$. Para o tamanho de bloco considerado, $N_b = M_b = 8$. Como as altas frequências têm um grande impacto visual e as baixas frequências apresentam fraquezas a diferentes ataques, a inserção da marca d'água é feita nas frequências médias (KATZENBEISSER; PETITCOLAS, 1999; ERNAWAN; KABIR, 2018), mostradas na área hachurada da Fig. 15 (as posições de um bloco da DCT em que a marca d'água é inserida estão hachuradas, sendo denominadas de sub-bloco). Denota-se por γ_i e θ_i a mediana dos coeficientes da DCT do i-ésimo bloco e a média dos coeficientes da DCT do sub-bloco correspondente, respectivamente, para $i = 1, \ldots, 4096$.

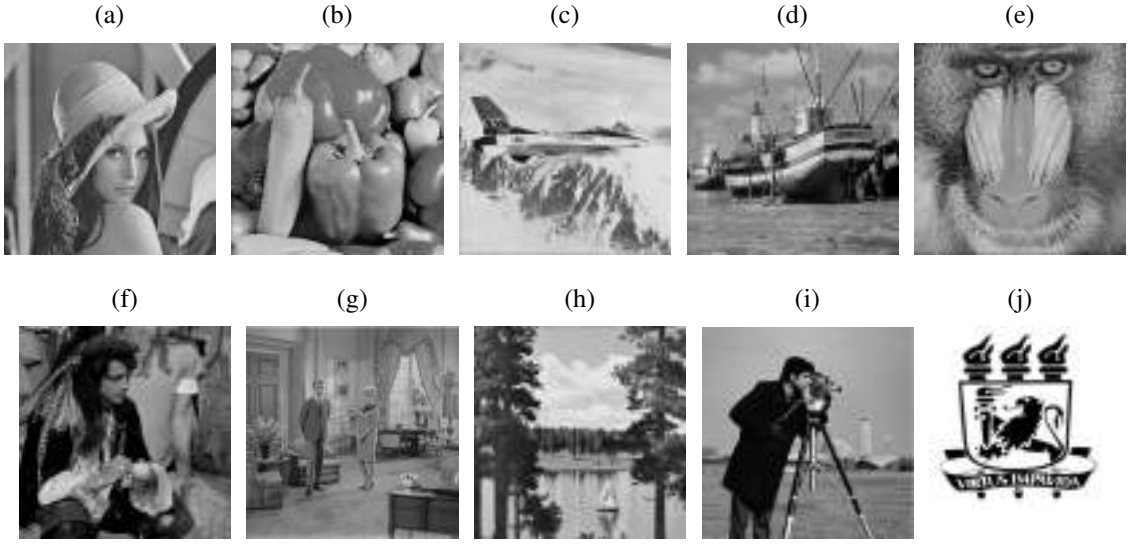
Figura 15 – Um bloco de tamanho 8×8 e um sub-bloco de tamanho 3×3 (área hachurada).



Fonte: O autor (2020).

Neste capítulo utiliza-se como imagem original as 141 imagens do banco de dados USC-SIPI (http://sipi.usc.edu/database), modificadas para tamanho 512×512 píxeis com 256 níveis de cinza. A Fig. 16 ilustra nove imagens originais e uma imagem binária de marca d'água W.

Figura 16 – Lena (a), Pimenta (b), Avião (c), Bote (d), Mandril (e), Pirata (f), Sala (g), Lago (h), Fotógrafo (i), imagem de marca d'água (j).



Fonte: O autor (2020).

4.2.1 Algoritmo de Inserção de Marca d'água

Para cada bloco, bits de marca d'água são inseridos na imagem original de acordo com os seguintes passos:

- A imagem C_O é divida em blocos 8×8 não sobrepostos. Calcula-se a DCT de cada bloco, bem como os valores de θ_i e γ_i.

- Realiza-se a geração de uma sequência binária a partir de um mapa caótico, sendo a condição inicial determinada pela chave K do algoritmo de marca d'água.

- Os 11 bits da sequência de informação do código são obtidos concatenando-se k_1 bits da imagem de marca d'água com k_2 bits obtidos com o mapa caótico, sendo $k_1 + k_2 = 11$. Após a codificação, obtém-se uma palavra código de 15 bits, dos quais 4 são bits de paridade. Cada bit de paridade é inserido em um bloco da imagem C_O. Para uma imagem de marca d'água de tamanho $M' \times N'$, o número total de bits de paridade é $4M'N'/k_1$ e este valor é igual a 4096 (número de blocos). Então

$$k_1 = \frac{4M'N'}{4096}. \tag{4.3}$$

  A sequência de paridade de comprimento 4096 é embaralhada, formando uma nova sequência de paridade $p_i, i = 1, \ldots, 4096$. Este embaralhamento é realizado considerando-se a sequência de paridade como uma imagem de tamanho 64×64 e embaralha-se essa imagem com o mapa de Arnold (MA). O objetivo do embaralhamento é incorporar bits de paridade não consecutivos em blocos consecutivos para melhorar a robustez contra alguns ataques geométricos. A sequência de paridade embaralhada é inserida na imagem C_0 de acordo com o procedimento a seguir.

- Cada bit de paridade p_i é inserido no i-ésimo bloco da imagem de acordo com as seguintes regras. O bit de paridade $p_i = 0$ indica que para este bloco $|\theta_i| - |\gamma_i| \geq df$, em que df é uma constante positiva fixa para todos os blocos. Se esta condição não é satisfeita, a cada coeficiente da DCT do sub-bloco é adicionado um valor ε_i de tal forma que a média do sub-bloco modificado θ_i' satisfaça $|\theta_i'| - |\gamma_i| > df$. O valor de ε_i é calculado para cada bloco. De forma análoga, o bit de paridade $p_i = 1$ indica que $|\theta_i| - |\gamma_i| < df$. Se isto não é satisfeito, ε_i é subtraído dos coeficientes de cada sub-bloco até satisfazer $|\theta_i'| - |\gamma_i| < df$. O valor de df estabelece um compromisso entre imperceptibilidade e robustez e sua escolha será descrita na Seção 4.4.

- Após a inserção de todos os bits de paridade é aplicada a DCT inversa de cada bloco, obtendo-se a imagem com marca d'água C_W.

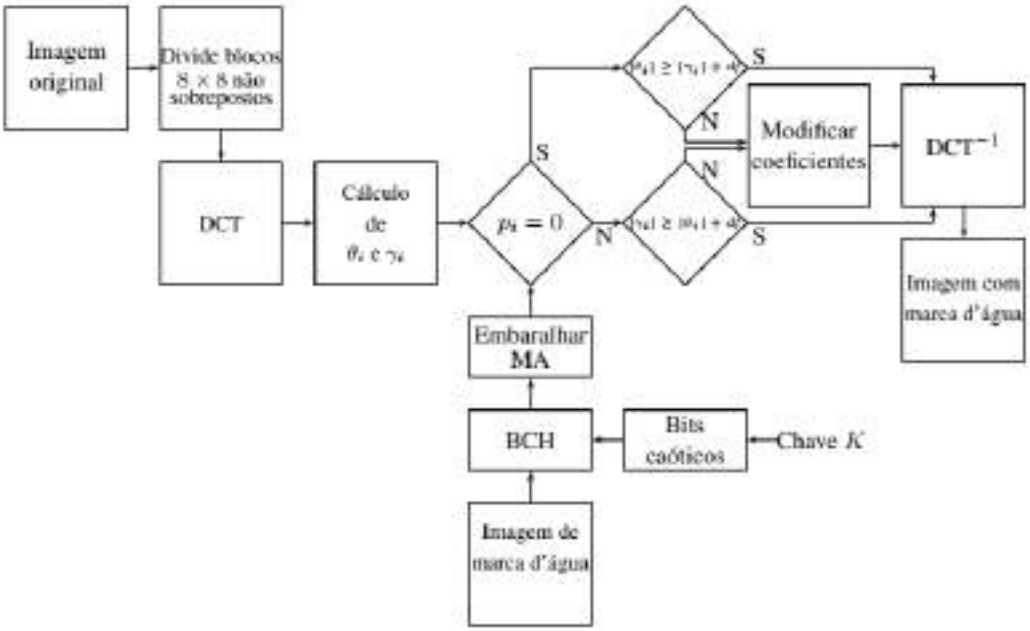
A Fig. 17 mostra o diagrama em blocos do algoritmo de inserção proposto.

4.2.2 Algoritmo de Recuperação da Marca d'água

Dada uma imagem com marca d'água C_W', podendo ser diferente de C_W devido a ataques, e a chave K, o algoritmo de recuperação da marca d'água segue os seguintes passos:

- A imagem C_W' é divida em blocos 8×8 não sobrepostos e executa-se a DCT para cada bloco.

Figura 17 – Diagrama em blocos do algoritmo de inserção da imagem de marca d'água proposto.



Fonte: O autor (2020).

- Calcula-se γ_i para o i-ésimo bloco e θ_i para o sub-bloco correspondente. Estima-se o bit de paridade $\hat{p}_i$ do i-ésimo bloco da seguinte forma: se $|\theta_i| > |\gamma_i|$, então $\hat{p}_i = 0$, caso contrário $\hat{p}_i = 1$. A sequência de bits de paridade estimada para a imagem $(\hat{p}_i, i = 1, \ldots, 4096)$ é desembaralhada e dividida em subsequências de 4 bits cada, $\hat{\mathbf{p}}_j = \hat{p}_{j1} \cdots \hat{p}_{j4}$, para $j = 1, \ldots, 1024$.

- Utiliza-se a chave K para replicar a sequência binária gerada a partir do mapa caótico.

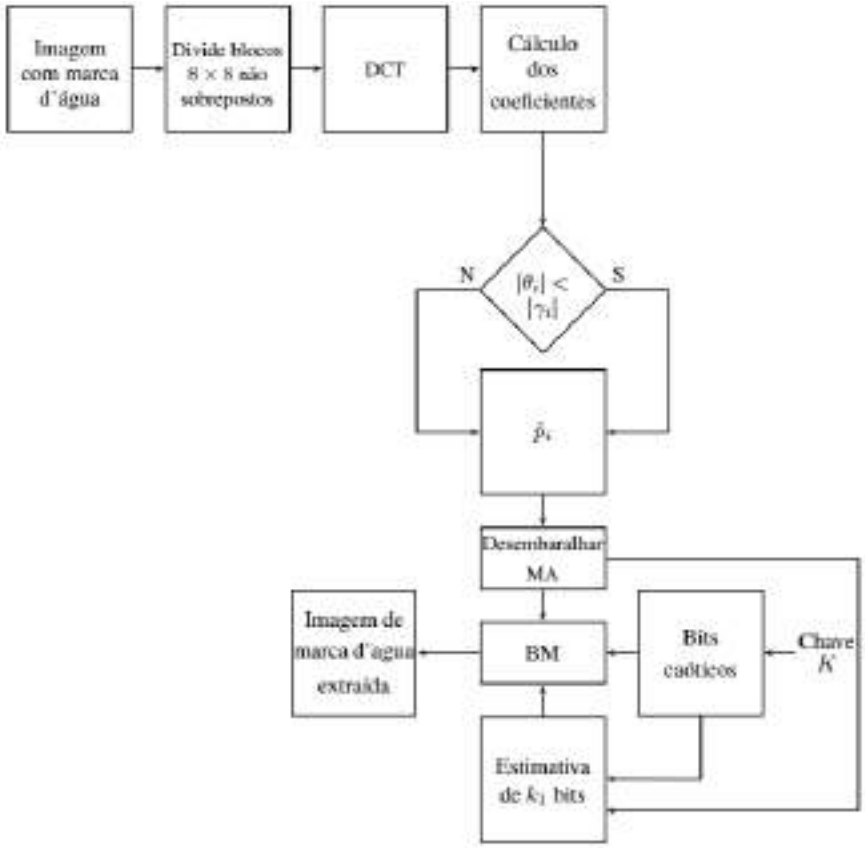
- Para cada $\hat{\mathbf{p}}_j$, o algoritmo de extração conhece k_2 bits caóticos de uma sequência de informação de 11 bits. Fixando estes k_2 bits, existem 2^{k_1} possíveis sequências de paridade, dependendo dos k_1 bits da imagem de marca d'água. Uma estimativa destes bits é obtida pela menor distância Hamming entre $\hat{\mathbf{p}}_j$ e estas possíveis sequências de paridade. Considere, por exemplo, $\hat{\mathbf{p}}_j = 0000$, $k_1 = 2$, $k_2 = 9$, e a sequência de bits caóticos 001011010. As 4 palavras código são obtidas concatenando as sequências 00, 01, 10, 11 com os bits caóticos produzindo as sequências de paridade 1101, 0010, 1010, 1010, respectivamente. A menor distância de Hamming é alcançada pela sequência de paridade 0010, significando que os k_1 bits estimados são 01.

- A palavra de 15 bits é formada a partir da concatenação dos k_1 bits estimados, dos k_2 bits caóticos e dos 4 bits de paridade com a menor distância de Hamming. No exemplo anterior, essa palavra é 010010110100010. A palavra de 15 bits é decodificada usando o algoritmo BM, fornecendo uma nova estimativa dos k_1 bits da imagem da marca d'água.

- Repete-se este procedimento para cada $j = 1, \ldots, 1024$.

A Fig. 18 mostra o diagrama em blocos do algoritmo de extração proposto.

Figura 18 – Diagrama em blocos do algoritmo de extração da imagem com marca d'água.



Fonte: O autor (2020).

4.3 MÉTRICAS DE IMPERCEPTIBILIDADE E ROBUSTEZ

Esta seção inicia com a descrição de métricas comumente usadas para avaliar a imperceptibilidade e robustez da marca d'água em imagens.

4.3.1 Métricas de Imperceptibilidade

A relação sinal-ruído de pico (PSNR, *peak signal to noise ratio*) é uma medida da degradação na imagem causada pela inserção da marca d'água. Esta é uma medida da imperceptibilidade da marca d'água, sendo expressa em unidade de decibéis (dB) e para imagens em escala de cinza com valor máximo igual a 255 é definida por:

$$\text{PSNR} = 10\log_{10}\left(\frac{255^2}{\text{MSE}}\right) \text{ (dB)} \tag{4.4}$$

em que

$$\text{MSE} = \frac{1}{M \times N}\sum_{i=1}^{M}\sum_{j=1}^{N}(C_O(i,j) - C_W(i,j))^2 \tag{4.5}$$

é o erro quadrático médio entre a imagem original e a imagem com marca d'água. Em (4.5), $C_O(i,j)$ e $C_W(i,j)$ são os valores dos píxeis na posição (i,j) na imagem original C_O e na imagem com marca d'água C_W, respectivamente, para imagens de tamanho $M \times N$. Outro parâmetro que mede a degradação da imagem com marca d'água com respeito à imagem original é o Índice de Similaridade Estrutural (SSIM, *Structural Similarity Index*), definido por (ERNAWAN; KABIR, 2018):

$$\text{SSIM} = \frac{(2\mu_O\mu_W + \alpha)(2\rho_{OW} + \beta)}{(\mu_O^2\mu_W^2 + \alpha)(\sigma_O^2\sigma_W^2 + \beta)} \tag{4.6}$$

sendo μ_O e μ_W a média da imagem original e da imagem com marca d'água, respectivamente, σ_O^2 e σ_W^2 as variâncias das duas imagens, ρ_{OW} a covariância entre C_O e C_W, α e β são constantes, sendo $\alpha = 2{,}55$ e $\beta = 7{,}65$ (ERNAWAN; KABIR, 2018).

4.3.2 Ataques a Imagem com Marca d'água

Para testar a robustez da marca d'água utilizam-se diferentes ataques. **Compressão com perdas** O JPEG (*Joint Photographic Experts Group*) introduziu um formato gráfico de imagem estática em 1992 como ITU-T Recommendation T.81, conhecido como JPEG (MITCHELL, 1992), que se tornou o formato dominante para a compressão de imagem. Geralmente, o JPEG é um algoritmo de compressão com perdas, o que significa que algumas informações são removidas da imagem durante sua compactação. O grau de perda pode ser ajustado pela taxa de compressão, chamada de ajuste de qualidade ou fator q. Neste trabalho, utilizam-se os fatores de compressão $q = 40, 50, 60, 70$. **Ruído Gaussiano** Adiciona-se ruído Gaussiano branco com média zero e variância σ^2 na imagem com marca d'água C_W. A Fig. 19 ilustra como o ruído Gaussiano modifica a imagem Lena para três valores de variância $\sigma^2 = \{0{,}001; 0{,}003; 0{,}005\}$.

Figura 19 – Imagem Lena com ruído Gaussiano: imagem original (a), ruído com $\sigma^2 = 0{,}001$ (b), ruído com $\sigma^2 = 0{,}003$ (c), ruído com $\sigma^2 = 0{,}005$ (d).

Fonte: O autor (2020).

Ruído Sal e Pimenta

O ruído sal e pimenta consiste em píxeis brancos e pretos que substituem os píxeis da imagem original em posições aleatórias. Estes assumem valores altos (sal) na faixa $(200, \ldots, 255)$ e valores baixos (pimenta) na faixa $(0, \ldots, 5)$. O efeito do ruído sal e pimenta na imagem Pimenta é ilustrado na Fig. 20, para três valores de densidade de ruído $d = \{0{,}001; 0{,}002; 0{,}01\}$.

Figura 20 – Imagem Pimenta com ruído Sal e Pimenta: imagem original (a), ruído com $d = 0{,}001$ (b), ruído com $d = 0{,}002$ (c), ruído com $d = 0{,}01$ (d).

(a) (b) (c) (d)

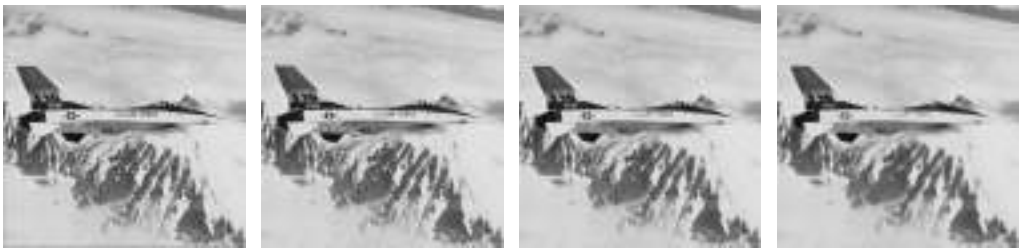
Fonte: O autor (2020).

Filtro de Mediana

É uma técnica de filtragem digital não linear, geralmente usada para reduzir o efeito do ruído em uma imagem, que pode destruir ou modificar a marca d'água. Este filtro utiliza uma janela que desliza sobre toda a imagem. A Fig. 21 mostra a imagem Avião com filtros de janelas de tamanho 3×3, 5×5 e 7×7.

Figura 21 – Imagem Avião com Filtro de Mediana com janela: imagem original (a), 3×3 (b), 5×5 (c), 7×7 (d).

(a) (b) (c) (d)



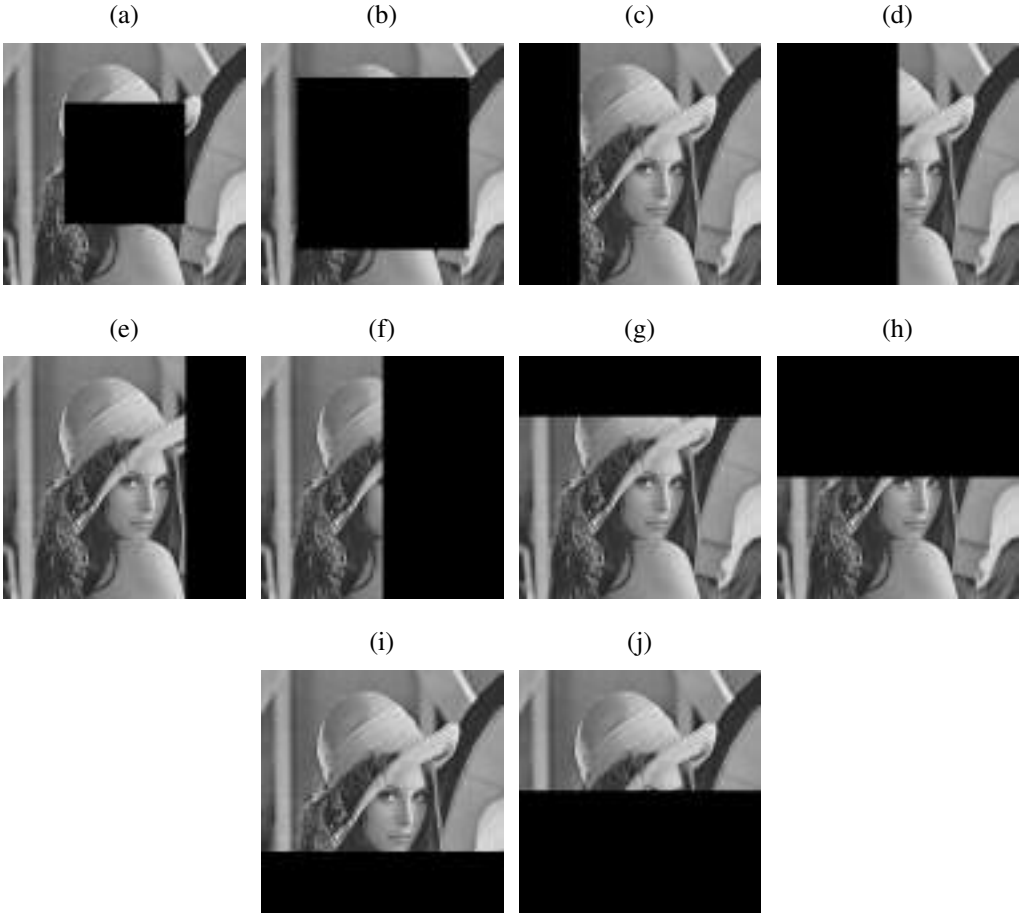
Fonte: O autor (2020).

Existem outros tipos de ataques que realizam uma transformação geométrica na imagem C_W, realizando um mapeamento da posição de um pixel na imagem C_W para uma nova posição em uma imagem C'_W. Os ataques geométricos incluem manipulações de processamento de imagem, como redimensionar, cortar para extrair uma região de interesse, rotacionar, entre outros (LICKS; JORDAN, 2005). Cada ataque geométrico é definido por um conjunto de parâmetros que determina a operação executada na imagem com marca d'água (por exemplo, ângulo de rotação, deslocamento da imagem, etc).

Corte

Extrai-se uma determinada região de interesse da imagem com marca d'água enquanto descarta-se as outras partes que são substituídas por píxeis preto ou branco. Este ataque incorre em uma perda de informação permanente sobre os píxeis descartados. A Fig. 22 mostra diferentes tipos de cortes utilizados neste trabalho.

Figura 22 – Diferentes cortes na imagem Lena: centro 25% (a), centro 50% (b), coluna esquerda 25% (c), coluna esquerda 50% (d), coluna direita 25% (e), coluna direita 50% (f), fila superior 25% (g), fila superior 50% (h), fila inferior 25% (i), fila inferior 50% (j).



Fonte: O autor (2020).

Translação

Uma translação desloca a imagem com marca d'água por um número especificado de píxeis na direção x ou y, ou em ambos, como é mostrado na Fig. 23.

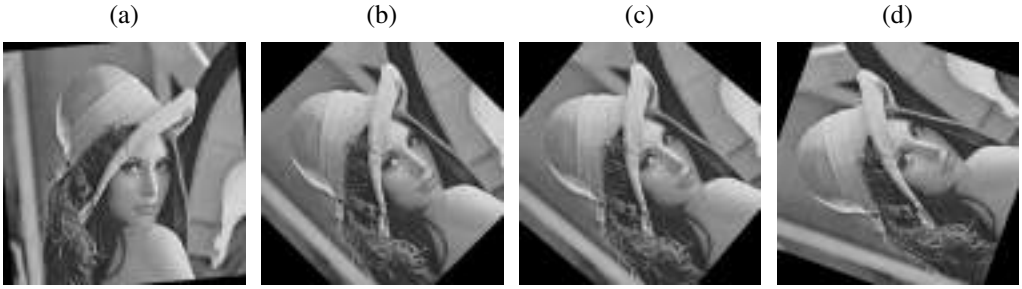
Figura 23 – Diferentes valores de deslocamento na imagem Lena: 5×5 (a), 10×10 (b), 20×20 (c).

Fonte: O autor (2020).

Rotação

Este ataque consiste em rotacionar uma imagem no sentido horário ou anti-horário por um determinado ângulo (LICKS; JORDAN, 2005). A Fig. 24 mostra as imagens com rotação ($\theta = 5°, 45°, 50°, 70°$) em sentido anti-horário.

Figura 24 – Diferentes valores de rotação na imagem Lena: $5°$ (a), $45°$ (b), $50°$ (c), $70°$ (d).



Fonte: O autor (2020).

Escalonamento

O escalonamento realiza uma transformação de compactação ou expansão de uma imagem. O fator de escala utilizado é o mesmo na direção horizontal e vertical. A Fig. 25 mostra as imagens com valores de escalonamento $0{,}25; 0{,}5; 0{,}8$.

Figura 25 – Diferentes valores de escalonamento na imagem Lena: 0,8 (a), 0,5 (b), 0,25 (c).

Fonte: O autor (2020).

4.3.3 Correlação Normalizada

A qualidade da imagem de marca d'água extraída é comumente avaliada por um parâmetro denominado correlação normalizada (NC). Em (ERNAWAN; KABIR, 2018) este é definido da seguinte forma:

$$\mathrm{NC}_1 = \frac{\sum_{i=1}^{M'} \sum_{j=1}^{N'} W(i,j) \times W'(i,j)}{\sqrt{\sum_{i=1}^{M'} \sum_{j=1}^{N'} W(i,j)^2 \sum_{i=1}^{M'} \sum_{j=1}^{N'} W'(i,j)^2}} \tag{4.7}$$

em que W' é a imagem de marca d'água extraída e W a imagem de marca d'água original, ambas de dimensão $M' \times N'$. O valor máximo igual a 1 indica que $W' = W$. A correlação normalizada

é uma medida de robustez da extração da marca d'água. Uma definição similar deste parâmetro é usada em (MAKBOL; KHOO; RASSEM, 2017):

$$\text{NC}_2 = \frac{\sum_{i=1}^{M'} \sum_{j=1}^{N'} W(i,j) \times W'(i,j)}{M' \times N'}. \tag{4.8}$$

Em (LOAN et al., 2018) a NC é definida como:

$$\text{NC}_3 = \frac{\sum_{i=1}^{M'} \sum_{j=1}^{N'} W(i,j) \times W'(i,j)}{\sum_{i=1}^{M'} \sum_{j=1}^{N'} W(i,j)^2}. \tag{4.9}$$

4.4 RESULTADOS

Nesta seção utiliza-se a imagem de marca d'água da Fig. 16 com dois tamanhos distintos (32×32 e 64×64) para avaliar as métricas PSNR, SSIM e NC do algoritmo proposto. Como é descrito na Subseção 4.2.1, os valores de k_1 e k_2 da sequência de informação do código BCH depende do tamanho da imagem de marca d'água. A utilização da imagem de marca d'água de tamanho 32×32 define $k_1 = 1$ e $k_2 = 10$ (1 bit da imagem de marca d'água é concatenado com 10 bits caóticos) e a imagem de marca d'água de tamanho 64×64 define $k_1 = 4$ e $k_2 = 7$.

O valor de df utilizado no algoritmo de inserção modifica a degradação da imagem original. A Tabela 20 mostra os valores mínimos e máximos da PSNR e SSIM para vários valores de df, para as 141 imagens da base de dados. Verifica-se que a redução do valor de df melhora a imperceptibilidade da imagem com marca d'água. Adotaremos $df = 0{,}40$, o que implica em uma PSNR mínima maior que 46 dB e um SSIM mínimo maior que 0,997 (valores mínimos considerando os dois tamanhos da imagem de marca d'água).

Tabela 20 – Valores mínimos e máximos da PSNR e SSIM com imagem de marca d'água 32×32 e 64×64.

df	PSNR				SSIM			
	32×32		64×64		32×32		64×64	
	Min	Max	Min	Max	Min	Max	Min	Max
0,30	50,47	53,50	49,67	52,07	0,9982	0,9992	0,9980	0,9991
0,35	48,92	51,55	48,72	50,72	0,9975	0,9989	0,9976	0,9988
0,40	47,71	49,82	46,01	48,92	0,9880	0,9984	0,9971	0,9985
0,45	41,84	45,66	40,27	44,60	0,9822	0,9867	0,9829	0,9869
0,50	39,27	41,88	36,20	40,15	0,9572	0,9801	0,9591	0,9796

Fonte: O autor (2020).

As Tabelas 21 e 22 mostram os resultados da NC definida em (4.7) para vários valores de df, para vários ataques, para as 141 imagens do banco de dados. Observa-se que o aumento do valor de df aumenta a robustez do algoritmo contra os ataques considerados.

Realiza-se a seguir comparações com algumas técnicas existentes na literatura assumindo-se a mesma capacidade, ou seja, imagens originais de tamanho 512×512 píxeis e imagem de

Tabela 21 – Valores mínimos e máximos de NC_1 para vários ataques com imagem de marca d'água 32×32.

Ataque	Fator	$df = 0{,}3$		$df = 0{,}35$		$df = 0{,}40$		$df = 0{,}45$	
		Min	Max	Min	Max	Min	Max	Min	Max
Ruído Gaussiano	$\sigma^2 = 0,001$	0,9897	0,9993	0,9900	0,9999	0,9923	1	0,9990	1
	$\sigma^2 = 0,005$	0,9766	0,9963	0,9893	0,9973	0,9910	1	0,9980	1
Ruído Sal e Pimenta	$d = 0,001$	0,9983	1	0,9993	1	1	1	1	1
	$d = 0,01$	0,9801	0,9992	0,9873	0,9997	0,9920	1	0,9982	1
Compressão JPEG	$q = 70$	1	1	1	1	1	1	1	1
	$q = 60$	1	1	1	1	1	1	1	1
	$q = 50$	0,9931	1	0,9978	1	0,9996	1	1	1
	$q = 40$	0,9820	0,9982	0,9952	0,9996	0,9911	1	0,9992	1
Filtro Mediana	3×3	0,9998	1	1	1	1	1	1	1
	5×5	0,9901	1	0,9953	1	0,9931	1	1	1
	7×7	0,9814	0,9905	0,9895	0,9975	0,9902	1	0,9928	1
Corte	25% Centro	0,9872	1	0,9895	1	0,9972	1	0,9984	1
	25% Coluna	0,9888	0,9989	0,9903	1	0,9965	1	0,9980	1
	25% Fila	0,9883	0,9996	0,9902	1	0,9985	1	0,9932	1
	50% Centro	0,9790	0,9982	0,9823	0,9995	0,9882	1	0,9920	1
	50% Coluna	0,9820	0,9973	0,9832	1	0,9868	1	0,9938	1
	50% Fila	0,9811	0,9985	0,9819	0,9998	0,9879	1	0,9932	1
Translação	(5,5)	0,9820	1	0,9911	1	0,9985	1	0,9998	1
	(10,10)	0,9752	0,9914	0,9768	0,9988	0,9823	0,9998	0,9901	1
	(10,20)	0,9434	0,9685	0,9552	0,9851	0,9693	0,9986	0,9923	0,9994
Rotação	5^o	0,9410	0,9714	0,9486	0,9820	0,9511	0,9910	0,9508	0,9750
	45^o	0,9130	0,9520	0,9249	0,9608	0,9401	0,9743	0,9508	0,9750
	50^o	0,9101	0,9508	0,9238	0,9611	0,9293	0,9741	0,9493	0,9738
	70^o	0,9008	0,9317	0,9132	0,9524	0,9193	0,9534	0,9408	0,9689
Escalonamento	0,25	0,9250	0,9590	0,9380	0,9632	0,9402	0,9810	0,9673	0,9931
	0,5	0,9375	0,9678	0,9407	0,9788	0,9453	0,9853	0,9762	0,9989
	0,8	0,9634	0,9827	0,9689	0,9910	0,9711	0,9962	0,9877	1

Fonte: O autor (2020).

marca d'água W de tamanho 32×32 bits (MAKBOL; KHOO; RASSEM, 2017; ERNAWAN; KABIR, 2018; KANG et al., 2018) e de tamanho 64×64 bits (LOAN et al., 2018; KO et al., 2020; DAS et al., 2014). Utiliza-se nas comparações as imagens mostradas na Fig. 16, uma vez que os resultados para estas imagens são fornecidos em (MAKBOL; KHOO; RASSEM, 2017; ERNAWAN; KABIR, 2018; KANG et al., 2018; LOAN et al., 2018). Os valores de PSNR e SSIM para o método proposto (com $df = 0{,}40$) e para os métodos apresentados em (ERNAWAN; KABIR, 2018; MAKBOL; KHOO; RASSEM, 2017; KANG et al., 2018; LOAN et al., 2018; KO et al., 2020; DAS et al., 2014) são comparados nas Tabelas 23 e 24. Observa-se que o método proposto apresenta resultados melhores.

As Tabelas 25, 26, 27, 28 e 29 apresentam comparações dos valores da NC obtidas com o algoritmo proposto e algoritmos propostos em (ERNAWAN; KABIR, 2018; MAKBOL; KHOO; RASSEM, 2017; KANG et al., 2018) e (THANKI; KOTHARI; TRIVEDI, 2019; LOAN et al., 2018), respectivamente, considerando diferentes ataques. Os resultados mostram que o algoritmo proposto apresenta robustez aos ataques analisados, apresentando na maioria do casos valores

Tabela 22 – Valores mínimos e máximos de NC_1 para vários ataques com imagem de marca d'água 64×64.

Ataque	Fator	$df = 0{,}3$		$df = 0{,}35$		$df = 0{,}4$		$df = 0{,}45$	
		Min	Max	Min	Max	Min	Max	Min	Max
Ruído Gaussiano	$\sigma^2 = 0,001$	0,9153	0,9480	0,9393	0,9712	0,9402	0,9872	0,9732	1
	$\sigma^2 = 0,005$	0,8731	0,9252	0,9105	0,9490	0,9285	0,9619	0,9426	0,9705
Ruído Sal e Pimenta	$d = 0,001$	0,9625	0,9903	0,9852	1	0,9561	1	0,9972	1
	$d = 0,01$	0,8862	0,9330	0,9138	0,9504	0,9315	0,9611	0,9510	0,9909
Compressão JPEG	$q = 70$	0,9923	1	1	1	1	1	1	1
	$q = 60$	0,9878	1	0,9910	1	0,9992	1	1	1
	$q = 50$	0,9675	0,9943	0,9863	1	0,9632	0,9990	0,9872	1
	$q = 40$	0,9448	0,9824	0,9588	0,9901	0,9470	0,9912	0,9790	1
Filtro Mediana	3×3	0,9583	0,9985	0,9751	1	0,9843	1	0,9924	1
	5×5	0,9109	0,9548	0,9234	0,9823	0,9342	0,9898	0,9638	0,9974
	7×7	0,8893	0,9372	0,9081	0,9652	0,9152	0,9734	0,9352	0,9736
Corte	25% Centro	0,9087	0,9387	0,9208	0,9509	0,9332	0,9593	0,9486	0,9710
	25% Coluna	0,9108	0,9298	0,9234	0,9500	0,9404	0,9622	0,9452	0,9705
	25% Fila	0,9014	0,9321	0,9280	0,9534	0,9374	0,9610	0,9401	0,9725
	50% Centro	0,8641	0,9086	0,8811	0,9380	0,9024	0,9557	0,9215	0,9592
	50% Coluna	0,8559	0,9019	0,8779	0,9290	0,8891	0,9492	0,9193	0,9606
	50% Fila	0,8591	0,8925	0,8803	0,9342	0,9015	0,9525	0,9208	0,9609
Translação	(5,5)	0,8953	0,9350	0,9148	0,9508	0,9322	0,9671	0,9402	0,9731
	(10,10)	0,8513	0,9187	0,8935	0,9294	0,9057	0,9428	0,9115	0,9580
	(10,20)	0,8508	0,9003	0,8762	0,9073	0,8881	0,9197	0,9008	0,9351
Rotação	5^o	0,8812	0,9119	0,8917	0,9336	0,9011	0,9483	0,9135	0,9572
	45^o	0,8078	0,8532	0,8371	0,9076	0,8611	0,9288	0,8952	0,9433
	50^o	0,7909	0,8502	0,8276	0,8890	0,8592	0,9242	0,8672	0,9417
	70^o	0,7881	0,8331	0,8094	0,8564	0,8268	0,8951	0,8515	0,9154
Escalonamento	0,25	0,8732	0,9205	0,8915	0,9376	0,9089	0,9711	0,9204	0,9793
	0,5	0,8641	0,9042	0,8836	0,9297	0,9025	0,9560	0,9371	0,9682
	0,8	0,8608	0,8891	0,8782	0,9008	0,8992	0,9432	0,9411	0.9569

Fonte: O autor (2020).

Tabela 23 – Comparação de PSNR para as nove imagens originais sem ataques com imagem de marca d'água 32×32 e 64×64.

Imagens	32×32				64×64			
	Proposto	(ERNAWAN; KABIR, 2018)	(MAKBOL; KHOO; RASSEM, 2017)	(KANG et al., 2018)	Proposto	(LOAN et al., 2018)	(KO et al., 2020)	(DAS et al., 2014)
Lena	48,78	45,69	41,39	40,07	48,08	42,65	41,44	41,78
Pimenta	49,21	45,53	43,07	42,25	48,77	42,65	41,66	42,13
Avião	48,05	45,93	42,75	39,10	47,24	42,54	41,15	40,79
Bote	47,74	45,50	42,26	38,71	47,03	42,08	40,82	40,08
Mandril	48,02	44,55	42,22	37,14	47,32	42,54	41,02	40,24
Sala	47,72	45,47	42,94	36,48	46,74	42,18	42,08	40,20
Pirata	47,80	45,30	42,88	39,38	47,38	42,25	42,46	41,83
Fotógrafo	48,82	46,01	42,12	42,51	48,27	43,05	41,69	42,63
Lago	47,72	45,74	43,01	35,90	46,71	41,71	41,73	40,07

Fonte: O autor (2020).

da NC melhores que os obtidos com as técnicas existentes. Os valores da NC maiores em cada comparação estão marcadas em negrito. A robustez do algoritmo é devido ao código corretor de erro, uma vez que no processo de recuperação da imagem de marca d'água, existem bits que são obtidos a partir do mapa caótico e são reproduzidos no processo de extração sem erros. Após estes procedimentos o algoritmo Berlekamp-Massey é capaz de corrigir um erro. No caso da existência de mais de um erro, estes nem sempre estão localizados nas posições da mensagem referentes aos bits da imagem de marca d'água.

Tabela 24 – Comparação de SSIM para as nove imagens originais sem ataques com imagem de marca d'água 32×32 e 64×64.

Imagens	32×32				64×64			
	Proposto	(ERNAWAN; KABIR, 2018)	(MAKBOL; KHOO; RASSEM, 2017)	(KANG et al., 2018)	Proposto	(LOAN et al., 2018)	(KO et al., 2020)	(DAS et al., 2014)
Lena	0,998	0,994	0,994	0,976	0,998	0,993	0,989	0,990
Pimenta	0,998	0,994	0,995	0,975	0,998	0,992	0,991	0,992
Avião	0,998	0,993	0,994	0,961	0,998	0,993	0,990	0,987
Bote	0,997	0,995	0,992	0,971	0,997	0,993	0,988	0,986
Mandril	0,998	0,995	0,995	0,980	0,998	0,994	0,992	0,987
Sala	0,997	0,996	0,994	0,959	0,998	0,993	0,987	0,985
Pirata	0,998	0,995	0,995	0,957	0,997	0,994	0,988	0,991
Fotógrafo	0,998	0,994	0,994	0,980	0,997	0,992	0,991	0,989
Lago	0,996	0,994	0,993	0,955	0,997	0,991	0,989	0,988

Fonte: O autor (2020).

Tabela 25 – Comparação da NC_1 para as imagens de Lena, Fotógrafo e Avião para vários ataques com imagem de marca d'água 32×32. Os valores da NC_1 maiores em cada comparação estão marcadas em negrito.

Ataques	Fator	Lena		Fotógrafo		Avião	
		Proposto	(ERNAWAN; KABIR, 2018)	Proposto	(ERNAWAN; KABIR, 2018)	Proposto	(ERNAWAN; KABIR, 2018)
Ruído Gaussiano	$\sigma^2 = 0,001$	**0,9984**	0,9393	**0,9977**	0,9344	**0,9983**	0,9322
Ruído Sal e Pimenta	$d = 0,001$	**0,9990**	0,9843	**1**	0,9804	**1**	0,9804
	$d = 0,01$	**0,9981**	0,8314	**0,9935**	0,8768	**0,9967**	0,8793
Compressão JPEG	$q = 70$	1	1	1	1	1	1
	$q = 60$	1	1	1	1	1	1
	$q = 50$	**1**	0,9990	**1**	0,9951	**0,9998**	0,9971
	$q = 40$	**1**	0,8733	**0,9993**	0,8929	**0,9992**	0,8669
Filtro Mediana	3×3	**1**	0,9854	**1**	0,9942	**1**	0,9912
Corte	25% Centro	1	1	**0,9990**	0,9648	**1**	0,9904
	25% Coluna	**1**	0,9168	**0,9985**	0,8485	**0,9991**	0,8839
	25% Flia	**1**	0,8564	**1**	0,9561	**0,9989**	0,9971
	50% Centro	**0,9994**	0,9773	**0,9934**	0,8413	**0,9951**	0,9639
	50% Coluna	**0,9990**	0,8908	**0,9892**	0,7193	**0,9927**	0,8606
	50% Fila	**0,9991**	0,7680	**0,9903**	0,8593	**0,9918**	0,9352
Translação	(5,5)	**0,9989**	0,5197	**0,9986**	0,5178	**0,9988**	0,5201
Rotação	5^o	**0,9704**	0,4942	**0,9628**	0,5015	**0,9681**	0,4936
Escalonamento	0,8	0,9954	**1**	0,9851	**1**	0,9833	**1**
	0,5	**0,9721**	0,7056	**0,9670**	0,7255	**0,9705**	0,7112

Fonte: O autor (2020).

Tabela 26 – Comparação da NC_2 para as imagens Lena e Pimenta para vários ataques com imagem de marca d'água 32×32. Os valores da NC_2 maiores em cada comparação estão marcadas em negrito.

Ataque	Fator	Lena		Pimenta	
		Proposto	(MAKBOL; KHOO; RASSEM, 2017)	Proposto	(MAKBOL; KHOO; RASSEM, 2017)
Ruído Gaussiano	$\sigma^2 = 0,001$	**0,9942**	0,9092	**0,9938**	0,7910
	$\sigma^2 = 0,005$	**0,9934**	0,9033	**0,9941**	0,7910
Ruído Sal e Pimenta	$d = 0,001$	**1**	0,9990	**1**	0,9980
	$d = 0,01$	**0,9948**	0,9883	**0,9932**	0,9541
Compressão JPEG	$q = 70$	1	1	**1**	0,9746
	$q = 50$	**1**	0,9980	**0,9999**	0,9238
Filtro Mediana	3×3	1	1	**1**	0,9385
	5×5	**0,9970**	0,7842	**0,9954**	0,6875
	7×7	**0,9908**	0,7842	**0,9901**	0,4238
Corte	50% Centro	**0,9990**	0,9648	**0,9997**	0,8643
	50% Coluna	**0,9989**	0,7305	**1**	0,5547
	50% Fila	**0,9992**	0,5703	**0,9997**	0,6582
Translação	(10,10)	**0,9935**	0,3984	**0,9942**	0,4922
	(10,20)	**0,9907**	0,4395	**0,9937**	0,4697
Rotação	$45°$	**0,9651**	0,4912	**0,9762**	0,4922
	$50°$	**0,9683**	0,5000	**0,9698**	0,4932
	$70°$	**0,9588**	0,4785	**0,9437**	0,5068
Escalonamento	0,25	**0,9980**	0,9951	**0,9982**	0,8584
	0,5	0,9990	**1**	**0,9976**	0,9756

Fonte: O autor (2020).

Tabela 27 – Comparação da NC_3 para as imagens Lena e Avião para vários ataques com imagem de marca d'água 32×32. Os valores da NC_3 maiores em cada comparação estão marcadas em negrito.

Ataque	Fator	Lena		Avião	
		Proposto	(KANG et al., 2018)	Proposto	(KANG et al., 2018)
Ruído Gaussiano	$\sigma^2 = 0,005$	**0,9962**	0,8470	**0,9938**	0,8344
Ruído Sal e Pimenta	$d = 0,01$	**0,9971**	0,8817	**0,9982**	0,8889
Compressão JPEG	$q = 70$	**1**	0,9859	**1**	0,9697
	$q = 50$	**1**	0,9449	**0,9997**	0,9395
Filtro Mediana	3×3	**1**	0,9967	**1**	0,9987
	5×5	**0,9988**	0,9486	**0,9990**	0,9455
Corte	25% Fila	**0,9995**	0,9574	**0,9992**	0,9554
	50% Fila	**0,9940**	0,9105	**0,9951**	0,9061

Fonte: O autor (2020).

Tabela 28 – Comparação da NC_3 para a imagem Lena para vários ataques com imagem de marca d'água 64×64. Os valores da NC_3 maiores em cada comparação estão marcadas em negrito.

Ataques	Fator	Lena	
		Proposto	(LOAN et al., 2018)
Ruído Gaussiano	$\sigma^2 = 0,001$	**0,9595**	0,9179
	$\sigma^2 = 0,01$	**0,9301**	0,8612
Ruído Sal e Pimenta	$d = 0,01$	**0,9588**	0,8548
Compressão JPEG	$q = 70$	1	1
	$q = 60$	1	1
	$q = 50$	0,9804	**1**
	$q = 40$	**0,9619**	0,9501
Filtro Mediana	3×3	**0,9951**	0,9258
Corte	25% Coluna	**0,9479**	0,9000
	25% Fila	**0,9402**	0,9027
Rotação	5^o	0,8930	**0,9731**
	10^o	0,8911	**0,9531**
Escalonamento	0,8	0,9301	**0,9979**

Fonte: O autor (2020).

Tabela 29 – Comparação da NC_3 para a imagem Pimenta para vários ataques com imagem de marca d'água 64×64. Os valores da NC_3 maiores em cada comparação estão marcadas em negrito.

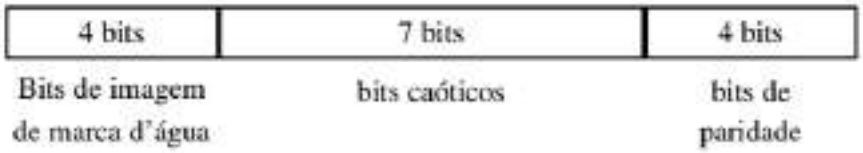
Ataque	Fator	Pimenta	
		Proposed	(THANKI; KOTHARI; TRIVEDI, 2019)
Ruído Gaussiano	$\sigma^2 = 0,001$	0,9607	**0,9837**
	$\sigma^2 = 0,01$	0,9299	**0,9922**
Ruído Sal e Pimenta	$d = 0,001$	**1**	0,9831
	$d = 0,005$	0,9708	**0,9934**
	$d = 0,01$	0,9497	**0,9843**
Compressão JPEG	$q = 70$	**1**	0,9837
	$q = 60$	**0,9997**	0,9741
	$q = 50$	**0,9798**	0,9639
	$q = 40$	**0,9601**	0,9597
Filtro Mediana	3×3	**0,9937**	0,4876
	5×5	**0,9584**	0,4656
	7×7	**0,9253**	0,3630
Corte	25% Centro	0,9535	**0,9795**
Rotação	20^o	**0,8875**	0,4798
	45^o	**0,8593**	0,4798
Escalonamento	0,25	**0,9008**	0,5051

Fonte: O autor (2020).

Robustez do Código

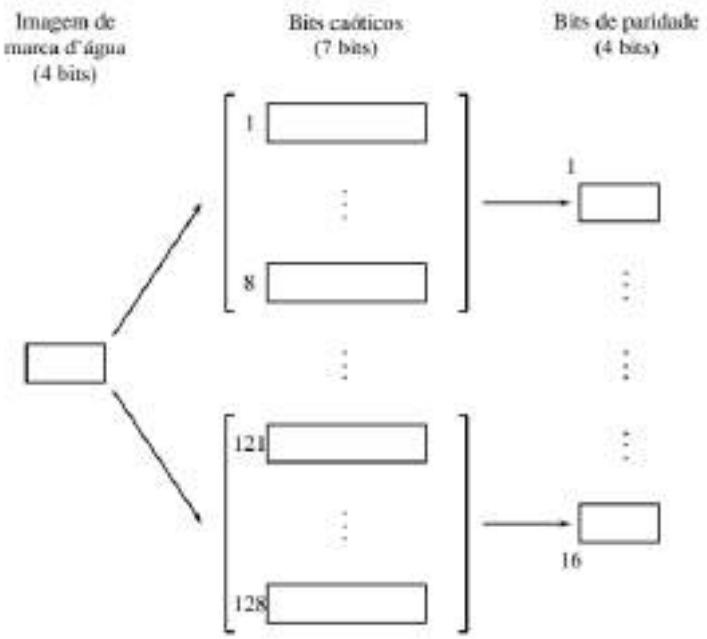
Esta subseção analisa a influência do código BCH (15,11,1) na robustez do algoritmo de extração. Cada palavra código é constituída de três partes, como mostra a Fig. 26. As 2^{11} palavras código possuem a seguinte característica: Para cada 4 bits da imagem de marca d'água, existem só 8 sequências caóticas distintas de 7 bits que mapeiam em cada uma das 2^4 sequências de paridade, como mostra a Fig. 27.

Figura 26 – Estrutura da palavra código.



Fonte: O autor (2020).

Figura 27 – Estrutura do código.



Fonte: O autor (2020).

Esta propriedade do código pode ser explorada por um adversário que quer descobrir os bits da chave. Utilizando como premissas que o adversário conhece a imagem de marca d'água W, não conhece a chave K e consegue extrair os 4 bits de paridade ($\hat{\mathbf{p}}_j$) sem erro. Desta forma o adversário conhece 8 bits de cada palavra código de comprimento 15, sendo que 8 sequências caóticas de 7 bits são possíveis. Para achar os bits da chave deve-se gerar todas as possíveis $2^{|K|}$ condições iniciais (para uma chave K de comprimento $|K|$ bits) e verifica-se se a chave gera uma das 8 possíveis sequências caóticas obtidas no primeira palavra código, reduzindo em média o espaço de chaves por um oitavo. Na segunda palavra código, realiza-se uma análise similar com o novo espaço de chaves (que agora tem uma cardinalidade média de $2^{|K|-3}$), verificando-se

a ocorrência das 8 sequências caóticas possíveis, reduzindo em média novamente o espaço de chaves em um oitavo. Realizando esta análise sucessivamente, a incerteza sobre a chave diminui em média 3 bits por palavra código, tendo que realizar esta análise em $\left\lceil \frac{|K|}{3} \right\rceil$ palavras códigos (em média) para descobrir a chave. A complexidade espacial de um ataque por um espaço-chave com cardinalidade $|K|$ é limitado inferiormente por

$$2^{|K|} + 2^{|K|-3} + \ldots + 2^{|K|-3s} = 2^{|K|} \times \left(1 + 2^{-3} + \ldots + 2^{-3s}\right) = 2^{|K|} \times \frac{1 - 2^{-3(s+1)}}{1 - 2^{-3}} \tag{4.10}$$

em que $s + 1 = \left\lceil \frac{|K|}{3} \right\rceil$. Como a razão em (4.10) é maior que um, o espaço médio de busca é maior que $2^{|K|}$, o qual é proibitivo para uma chave de extensão prática, e.g. $|K| = 128$ bits.

5 CONCLUSÕES E TRABALHOS FUTUROS

Este capítulo sumariza as principais contribuições e resultados obtidos nesta tese e comenta sobre os possíveis trabalhos futuros.

Nesta tese, diferentes formas de utilizar fontes de aleatoriedade baseadas em caos são propostas para modificar ou projetar algoritmos de cifragem e de marca d'água. Primeiramente sequências obtidas a partir de mapas caóticos são utilizadas como fonte de aleatoriedade para modificar o algoritmo AES, na etapa de substituição, conduzindo a um novo algoritmo AES1. A introdução de fonte de aleatoriedade tem um impacto direto na confusão e difusão da informação, como também impacta na geração de sub-chaves, característica que permite uma diminuição do número de iterações do algoritmo, obtendo dois novos algoritmos AES2 e AES3. Utiliza-se para avaliar cada algoritmo várias métricas de segurança utilizadas na literatura e conclui-se que estes apresentam um comportamento similar aos dos modos de operação conhecidos na literatura (CBC, OFB, CFB e CRT) com tempo de simulação menor.

Os algoritmos apresentados AES1, AES2 e AES3 foram modificados com estrutura similar para uma análise mais simples sob a perspectiva das criptoanálises linear e diferencial. A criptoanálise linear indica que estes algoritmos de cifra de bloco modificados (com a introdução de bits caóticos na etapa de substituição de bytes e na geração de sub-chaves) aumentam o número de pares de texto claro e texto cifrado necessários para achar bits da chave com certa confiabilidade. Uma análise da criptoanálise diferencial implica em uma diminuição considerável da probabilidade de uma escolha correta de bits de chave.

Também foi apresentada uma nova técnica de inserção e extração de marca d'água robusta em imagens em escala de cinza. Os bits da marca d'água (bits de paridade de um código corretor de erro) são incorporados nos coeficientes da DCT dependendo do valor de um limiar (df) no qual existe um compromisso entre imperceptibilidade e robustez. Inicialmente, estuda-se esse limiar usando o banco de dados de imagens USC-SIPI. Na sequência, para um valor fixo de $df = 0{,}4$, realiza-se comparações que mostram que o esquema proposto apresenta, em vários casos, maior imperceptibilidade e melhor robustez contra ataques em comparação com as técnicas existentes em termos de PSNR, SIMM, NC. A aplicação conjunta de bits caóticos e códigos BCH não apenas fornece segurança, mas também contribui para a robustez do esquema proposto.

5.1 TRABALHOS FUTUROS

Algumas ideias para o prosseguimento deste trabalho estão listadas a seguir.

- Realizar um estudo detalhado da criptoanálise diferencial impossível para o algoritmo simplificado do AES, quantificando-se o impacto da introdução dos bits caóticos no algoritmo AES em relação a esta criptoanálise.

- Realizar um estudo do algoritmo de marca d'água para outros códigos corretores de erro. Em particular, a classe de códigos com proteção desigual de erros pode ser promissora uma vez que parte da sequência de informação (bits caóticos) é conhecida pelo decodificador (ZHANG et al., 2011; CHUAN; QIAN; XINPENG, 2009).
- Realizar uma análise do algoritmo de marca d'água proposto em imagens coloridas.
- Propor uma análise de mapas caóticos em sistemas criptográficos de chave pública. Estes sistemas podem ser utilizados em redes de sensores para autenticação e autorização. Recentemente para fornecer segurança e privacidade nestas redes, foi introduzido o conceito de cadeia de blocos (BC, *Blockchain*) (MISTRY et al., 2019; KHAN; SALAH, 2017). No entanto, a BC é complexa computacionalmente, característica que pode ser melhorada com a utilização de mapas caóticos.
- Realizar um estudo sobre a utilização de mapas caóticos em marcas d'água no contexto da Internet das coisas (HAMEED et al., 2017; MISTRY et al., 2019) e redes 5G (AYESHA; MASILAMANI, 2018; MISTRY et al., 2019).

REFERÊNCIAS

AHMADI, S. B. B.; ZHANG, G.; WEI, S. Robust and hybrid SVD-based image watermarking schemes: A survey. **Multimedia Tools and Applications**, v. 79, p. 1075–1117, 2020. 16

ALAWIDA, M. et al. A new hybrid digital chaotic system with application in image encryption. **Signal Processing**, v. 160, p. 45–58, 2019. 15, 38, 41

ALLIGOOD, K.; SAUER, T.; YORKE, J. **Chaos An Introduction to Dynamical Systems**. [S.l.]: New York, NY, 1997. ISBN 9780387946771. 17

ANAND, A.; SINGH, K. An improved DWT-SVD domain watermarking for medical information security. **Computer Communications**, v. 152, p. 72–80, 2020. 17

ARTILES, J. et al. Uma metodologia para geração de sequências aleatórias usando mapas caóticos. In: **Simpósio Brasileiro de Telecomunicações (SBrT 2015), Juiz de Fora**. [S.l.: s.n.], 2015. p. 1–5. 17

AYESHA, S.; MASILAMANI, V. A novel digital watermarking scheme for data authentication and copyright protection in 5G networks. **Computers and Electrical Engineering**, v. 72, p. 589–605, 2018. 92

BECHTSOUDIS, A.; SKLAVOS, N. Side channel attacks cryptanalysis against block ciphers based on FPGA devices. In: **2010 IEEE Computer Society Annual Symposium on VLSI**. [S.l.: s.n.], 2010. p. 460–461. ISSN 2159-3469. 14

BIHAM, E.; SHAMIR, A. Difierential cryptanalysis of DES like cryptosystems. **Advances in Cryptology CRYPTO'90**, v. 537, p. 2–21, 1991. 15

BIHAM, E.; SHAMIR, A. Difierential cryptanalysis of DESlike cryptosystems. **Journal of Cryptology**, v. 4, n. 1, p. 3–72, 1991. 15

BIHAM, E.; SHAMIR, A. **Differential Cryptanalysis of the Data Encryption Standard**. [S.l.]: Springer-Verlag, 1993. ISBN 9780387979304. 15

BIHAM, E.; SHAMIR, A. Difierential cryptanalysis of the full 16 round DES. **Advances in Cryptology CRYPTO'92**, v. 740, p. 487–496, 1993. 15

CAO, W.; MAO, Y.; ZHOU, Y. Designing a 2D infinite collapse map for image encryption. **Siganl Processing**, v. 171, p. 1–13, 2020. 15

CHEN, S.; LEUNG, H. Ergodic chaotic parameter modulation with application to digital image watermarking. **IEEE Transactions on Image Processing**, v. 14, n. 19, p. 1590 – 1602, 2005. 16, 17

CHUAN, Q.; QIAN, M.; XINPENG, Z. Image watermarking with unequal error protection capability. In: **2009 International Conference on Multimedia Information Networking and Security**. [S.l.: s.n.], 2009. p. 1–5. 92

DAS, C. et al. A novel blind robust image watermarking in DCT domain using inter-block coefficient correlation. **AEU - International Journal of Electronics and Communications**, v. 68, p. 244–253, 2014. 84, 85, 86

ERNAWAN, F.; KABIR, M. A robust image watermarking technique with an optimal DCT psychovisual threshold. **IEEE Access**, v. 6, p. 20464–20480, 2018. 74, 79, 82, 84, 85, 86

EVANGELISTA, J. V. et al. Emitter-coupled pair chaotic generator circuit. **AEU - International Journal of Electronics and Communications**, v. 77, p. 112–117, 2017. 18

FAN, T.; CHAO, H.; CHIEU, B. Lossless medical image watermarking method based on significant difference of cellular automata transform coefficient. **Signal Processing: Image Communication**, v. 70, p. 174–183, 2019. 17, 73

FARES, K.; AMINE, K.; SALAH, E. A robust blind color image watermarking based on fourier transform domain. **Optik**, v. 208, p. 1–9, 2020. 16

(FIPS), F. I. P. S. **Advanced Encryption Standard (AES)**. [S.l.], 2001. 14, 26, 27, 30

GU, Q.; GAO, G. A novel reversible robust watermarking algorithm based on chaotic system. **Digital Signal Processing**, v. 23, p. 213–217, 2013. 16, 17

GUO, J.; RIYONO, D.; PRASETYO, H. Improved beta chaotic image encryption for multiple secret sharing. **IEEE Access**, v. 6, p. 46297–46321, 2018. 15

HAMEED, K. et al. Towards a formally verified zero watermarking scheme for data integrity in the internet of things based-wireless sensor networks. **Future Generation Computer Systems**, v. 82, p. 274–289, 2017. 92

HUA, Z.; ZHOU, B.; ZHOU, Y. Sine chaotification model for enhancing chaos and its hardware implementation. **IEEE Transactions on Industrial Electronics**, v. 66, n. 2, p. 1273–1284, 2019. 15, 18

HUA, Z.; ZHOU, Y.; HUANG, H. Cosine-transformation-based chaotic system for image encryption. **Information Science**, v. 409, p. 403–419, 2019. 15, 18, 38, 40

KANG, X. et al. A novel hybrid of DCT and SVD in DWT domain for robust and invisible blind image watermarking with optimal embedding strength. **Multimedia Tools and Applications**, v. 77, p. 13197–13224, 2018. 16, 84, 85, 86, 87

KATZENBEISSER, S.; PETITCOLAS, F. **Information Hiding Techniques for Steganography and Digital Watermarking**. [S.l.]: Artech House Print on Demand, 1999. 16, 17, 72, 74

KESHAVARZIAN, R.; AGHAGOLZADEH, A. ROI based robust and secure image watermarking using DWT and Arnold map. **AEU - International Journal of Electronics and Communications**, v. 70, n. 3, p. 278–288, 2016. 16

KHAN, M.; SALAH, K. IoT security: Review, blockchain solutions, and open challenges. **Future Generation Computer Systems**, v. 82, p. 395–411, 2017. 92

KO, H. J. et al. Robust and blind image watermarking in DCT domain using inter-block coefficient correlation. **Information Sciences**, v. 517, p. 128–147, 2020. 16, 84, 85, 86

LAN, R. et al. Integrated chaotic systems for image encryption. **Signal Processing**, v. 147, p. 133–145, 2018. 15, 25

LAU, F.; TSE, C. **Chaos Based Digital Communication Systems**. [S.l.]: Springer, 2010. (Engineering online library). ISBN 9783540006022. 17, 18

LEE, Y.; SEO, Y.; KIM, D. Digital blind watermarking based on depth variation prediction map and DWT for DIBR free-viewpoint image. **Signal Processing: Image Communication**, v. 70, p. 104–113, 2019. 16

LEFEVRE, P.; CARRE, P.; GABORIT, P. Application of rank metric codes in digital image watermarking. **Signal Processing: Image Communication**, v. 74, p. 119–128, 2019. 17, 73

LI, C. Cracking a hierarchical chaotic image encryption algorithm based on permutation. **Signal Processing**, v. 118, p. 203–210, 2016. 15

LI, H. Efficient and flexible architecture for AES. **IEE Proceedings - Circuits, Devices and Systems**, v. 153, n. 6, p. 533–538, 2006. ISSN 1350-2409. 14

LICKS, V.; JORDAN, R. Geometric attacks on image watermarking systems. **IEEE MultiMedia**, v. 12, n. 3, p. 68 – 78, 2005. 80, 82

LIU, Y. et al. Chosen-plaintext attack of an image encryption scheme based on modified permutation- diffusion structure. **Nonlinear Dynamics**, v. 84, n. 4, p. 2241–2250, 2016. 15

LOAN, N. et al. Secure and robust digital image watermarking using coefficient differencing and chaotic encryption. **IEEE Access**, v. 6, p. 19876–19897, 2018. 16, 17, 83, 84, 85, 86, 87

MAKBOL, N.; KHOO, B.; RASSEM, T. Block-based discrete wavelet transform-singular value decomposition image watermarking scheme using human visual system characteristics. **IET Image Processing**, v. 10, n. 1, p. 34 – 52, 2017. 83, 84, 85, 86

MANSOORI, D.; KHALEGHI, H. Linear Cryptanalysis on Second Round Simplified AES. In: **The 8th International Conference on Advanced Communication Technology**. [S.l.: s.n.], 2006. p. 1210–1214. 52

MATSUI, M. Linear cryptanalysis method for DES cipher. **Advances in Cryptology EUROCRYPT'93**, v. 765, p. 386–397, 1994. 15

MATSUI, M.; YAMAGISHI, A. A new method for known plaintext attack of FEAL cipher. **Workshop on the Theory and Application of of Cryptographic Techniques**, v. 658, p. 81–91, 1992. 15

MISTRY, I. et al. Blockchain for 5G-enabled IoT for industrial automation: A systematic review, solutions, and challenges. **Mechanical Systems and Signal Processing**, v. 135, p. 1–21, 2019. 92

MITCHELL, J. Information technology - digital compression and coding of continuous-tone still images - requirements and guidelines. **International Communication Union**, 1992. Disponível em: <http://www.itu.int/rec/T-REC-T.81-199209-I/en>. 79

MOHAMMAD, M.; SCHAEFER, E.; WEDIG, S. A simplified Rijndael algorithm and its linear and differential cryptanalysis. **Cryptologia**, v. 27, n. 2, p. 148–177, 2003. 42

MOOSAZADEH, M.; EKBATANIFARD, G. A new DCT based robust image watermarking method using teaching-learning-based optimization. **Journal of Information Security and Applications**, v. 47, p. 28–38, 2019. 16

O'RUANAIDH, J.; PUN, T. Rotation, scale and translation invariant spread spectrum digital image watermarking. **Signal Processing**, v. 66, n. 3, p. 303–317, 1998. 16

PAAR, C.; PELZL, J. **Understanding Cryptography, A Textbook for Students and Practitioners**. [S.l.]: Springer, 2010. ISBN 978-3-642-04101-3. 14, 21, 37

PARAH, S. A. et al. Robust and blind watermarking technique in DCT domain using inter-block coefficient differencing. **Digital Signal Processing**, v. 53, p. 11–24, 2016. 16

PREISHUBER, M. et al. Depreciating motivation and empirical security analysis of chaos-based image nd video encryption. **IEEE Transactions on Information Forensics and Security**, v. 13, n. 9, p. 2137–2150, 2018. 14, 15

QASIM, A.; MEZIANE, F.; ASPIN, R. Digital watermarking applicability for developing trust in medical imaging workflows state of the art review. **Computer Science Review**, v. 27, p. 45–60, 2018. 16

ROY, S.; KUMAR, A. A blind DCT based color watermarking algorithm for embedding multiple watermarks. **AEU International Journal of Electronics and Communications**, v. 72, p. 149–161, 2017. 16, 17

SARRESHTEDARI, S.; AKHAEE, M.; ABBASFAR, A. Source-channel coding-based watermarking for self-embedding of JPEG images. **Signal Processing: Image Communication**, v. 62, p. 106–116, 2018. 17, 73

SHAO, Z. et al. Robust watermarking using orthogonal Fourier-Mellin moments and chaotic map for double images. **Signal Processing**, v. 120, p. 522–531, 2016. 16

STROGATZ, S. **Nonlinear Dynamics and Chaos with Applications to Physics, Biology, Chemistry, and Engineering**. [S.l.]: Westview Press, 2001. (Studies in Nonlinearity Series). ISBN 9780813349114. 15, 17, 18

SU, Q. et al. New rapid and robust color image watermarking technique in spatial domain. **IEEE Access**, v. 7, p. 30398–30409, 2019. 16

TALHAOUI, M.; WANG, X.; TALHAOUI, A. A new one-dimensional chaotic map and its application in a novel permutation-less image encryption scheme. **The Visual Computer**, 2020. 15

TAN, Y. et al. A robust watermarking scheme in YCbCr color space based on channel coding. **IEEE Access**, v. 7, p. 25026 – 25036, 2019. 17, 73

TANG, Z. et al. Multiple image encryption with bit plane decomposition and chaotic maps. **Optics and Lasers in Engineering**, v. 80, p. 1–11, 2016. 15, 25

TAO, H. et al. Robust image watermarking theories and techniques: A review. **Journal of applied research and technology**, UNAM, Centro de Ciencias Aplicadas y Desarrollo Tecnológico, v. 12, n. 1, p. 122–138, 2014. 14

THANKI, R.; KOTHARI, A.; TRIVEDI, D. Hybrid and blind watermarking scheme in DCuT-RDWT domain. **Journal of Information Security and Applications**, v. 46, p. 231–249, 2019. 84, 88

WANG, B. et al. Evaluating the permutation and diffusion operations used in image encryption based on chaotic maps. **Optik**, v. 127, n. 7, p. 3541–3545, 2016. 25

WANG, M. et al. A novel chaotic encryption scheme based on image segmentation and multiple diffusion models. **Optics and Laser Technology**, v. 108, p. 558–573, 2018. 38, 40, 41

WANG, X.; LIU, C. A novel and effective image encryption algorithm based on chaos and DNA encoding. **Multimedia Tools and Applications**, v. 76, n. 5, p. 6229–6245, 2017. 15

WENHAO, L.; KEHUI, S.; CONGXU, Z. A fast image encryption algorithm based on chaotic map. **Optics and Lasers in Engineering**, v. 84, p. 26–36, 2016. 15

WU, X.; GUAN, Z. A novel digital watermark algorithm based on chaotic maps. **Physics Letters A**, v. 365, p. 403–406, 2007. 16, 17

WU, X. et al. A novel color image encryption scheme using rectangular transform-enhanced chaotic tent maps. **IEEE Access**, v. 5, p. 6429–6436, 2017. 15

WU, Y.; NOONAN, J.; AGAIAN, S. NPCR and UACI randomness tests for image encryption. **Cyber Journals: Multidisciplinary Journals in Science and Technology, Journal of Selected Areas in Telecommunications (JSAT)**, p. 31–38, 2011. 26

WU, Y. et al. Local shannon entropy measure with statistical tests for image randomness. **Information Sciences**, v. 22, p. 323–342, 2013. 25, 26

XU, H. et al. Rotation and scale invariant image watermarking based on polar harmonic transforms. **Optik**, v. 183, p. 401–414, 2019. 16

XU, L. et al. A novel bit level image encryption algorithm based on chaotic maps. **Optical and Laser in Engineering**, v. 78, p. 17–25, 2016. 15, 25

YANG, B.; LIAO, X. Some properties of the logistic map over the finite field and its application. **Signal Processing**, v. 153, p. 231–242, 2018. 15

ZHANG, W. et al. Unequal error protection of JPEG2000 images using short block length turbo codes. **IEEE Communications Letters**, v. 15, n. 6, p. 659 – 661, 2011. 92

ZHANG, Y.; LI, X.; HOU, W. A fast image encryption scheme based on AES. In: **2017 2nd International Conference on Image, Vision and Computing (ICIVC)**. [S.l.: s.n.], 2017. p. 2624–2628. 14

ZHENJUN, Z. et al. Multiple image encryption with bit plane decomposition and chaotic maps. **Optics and Lasers in Engineering**, v. 80, p. 1–11, 2016. 25

ZHU, C.; SUN, K. Cryptanalyzing and improving a novel color image encryption algorithm using RT-enhanced chaotic tent maps. **IEEE Access**, v. 6, p. 18759–18770, 2018. 15