Análise dos Algoritmos de Perfil II do Projeto eSTREAM para Criptografia de Imagens

Alternative Title: Analysis of eSTREAM Profile II Ciphers for Image Encryption

João Paulo F. C. César
joaopaulofcc@gmail.com

Wilian Soares Lacerda
lacerda@dcc.ufla.br

Bruno de Abreu Silva
bruno.abreu@dcc.ufla.br

Universidade Federal de Lavras - Departamento de Ciência da Computação
Laboratório de Sistemas Inteligentes e Embarcados - LABSINE
Caixa Postal 3037 - CEP 37200-000
Lavras - MG - Brasil

RESUMO

Segurança e privacidade sempre foram alvo de pesquisas, mas atualmente com a popularização dos meios de comunicação em massa, como a Internet, esse assunto se torna ainda mais fundamental. A comunicação nos dias de hoje não é realizada apenas pela troca de arquivos de texto ou áudio, mas também pela troca de imagens digitais. Sistemas de criptografia vêm sendo constantemente aprimorados e padronizados para prover segurança e privacidade, inclusive para algoritmos especialistas em cifra de imagens. Apesar de serem traduzidas em dados binários assim como os textos, as imagens possuem características particulares que impedem o uso de sistemas de criptografia populares, como o RSA, DES e AES. Sistemas de criptografia por fluxo são compactos e de simples implementação. Para promover seu desenvolvimento o ECRYPT *(European Network of Excellence for Cryptology)* organizou o projeto eSTREAM, resultando em um portfólio de cifras de fluxo validadas para implementações em *software* e *hardware*. O presente artigo apresenta a análise das cifras de Perfil II desse projeto quanto a sua qualidade quando aplicadas em imagens digitais.

Palavras-Chave

Criptografia, Criptografia de Imagens, eSTREAM.

ABSTRACT

Security and privacy have always been the subject of research, but currently with the popularization of mass media, such as the Internet, this subject becomes even more fundamental. Communication today is not only done by exchanging text or audio files, but also by exchanging digital images files. Encryption systems have been constantly improved and standardized to provide security and privacy, including algorithms specialized in image encryption. Although they are translated into binary data as well as texts, the images have particular characteristics that prevent the use of popular cryptographic systems such as RSA, DES and AES. Stream cipher systems are compact and simple to implement. To promote their development ECRYPT (European Network of Excellence for Cryptology) has organized the eSTREAM project, resulting in a portfolio of validated stream ciphers for software and hardware implementations. This article presents the analysis of eSTREAM Profile II ciphers regarding their quality when applied in digital images.

SBSI 2017 June 5^{th} – 8^{th}, 2017, Lavras, Minas Gerais, Brazil

CCS Concepts

•**Security and privacy** → **Hardware-based security protocols;**

Keywords

Encryption, Image Encryption, eSTREAM.

1. INTRODUÇÃO

A criptografia é usada como técnica de transformação de dados, segundo um código ou algoritmo, para que eles se tornem ininteligíveis para quem não possua a chave do código [18]. Ao longo do tempo foram propostas diversas técnicas e algoritmos para a transferência segura de dados, bem como órgãos governamentais e instituições para controle e normatização de padrões [17]. Com a popularização da Internet, tornou-se comum enviar arquivos para outros usuários dessa rede, seja por meio de correio eletrônico, redes sociais ou mensageiros instantâneos. Entre os tipos de arquivos compartilhados estão as imagens, que podem conter informações privadas, sigilosas e sensíveis. Para que essas imagens possam trafegar por um meio não seguro, como é a Internet, de forma segura, deve-se utilizar dos conceitos da criptografia.

Diversos esquemas de criptografia foram propostos com o objetivo de transmitir dados de maneira segura, porém para a criptografia de imagens não é recomendada a utilização de algoritmos clássicos, como o RSA e AES. Essa restrição deve-se às características intrínsecas de uma imagem como por exemplo sua alta capacidade de armazenamento de dados e correlação entre *pixels*. Esse problema é ressaltado ao tratar de imagens em meios de comunicação online [4]. Uma das propostas atuais para criptografia de imagens é a utilização de sistemas caóticos e o aprimoramento de geradores

para cifragem em fluxo, sendo utilizado inclusive combinação dessas duas técnicas, como em [15] [16].

O projeto eSTREAM foi anunciado em 2004 pelo *European Network of Excellence for Cryptology (ECRYPT)* com objetivo de identificar novos algoritmos de criptografia de fluxo para futura utilização [14]. O projeto é dividido em dois perfis de implementação, o Perfil I com algoritmos para *software* capazes de suprir demandas de alta vazão de dados, e o Perfil II aplicado em *hardware* com limitação de recursos. No ano de 2008 foram escolhidos os finalistas de cada perfil, que passaram então a compor o portfólio do projeto. O objetivo desse trabalho é analisar os algoritmos de Perfil II do projeto eSTREAM, quanto a qualidade desses no processo de criptografia de imagens. Nenhum trabalho publicado até o momento realizou a análise da qualidade desses algoritmos quando aplicados em imagens digitais.

Esse trabalho inova ao realizar essa análise, proporcionando à comunidade científica o embasamento estatístico da qualidade desses algoritmos. O restante desse artigo é organizado da seguinte forma: na Seção 2 são discutidos conceitos básicos de criptografia, a Seção 3 aborda os algoritmos de Perfil II do projeto eSTREAM. Já na Seção 4 é discutido a metodologia utilizada na execução dos experimentos, cujos resultados obtidos são exibidos na Seção 5. Na Seção 6 são apresentadas as conclusões

2. CRIPTOGRAFIA

Os algoritmos de criptografia podem ser classificados pelo número de chaves utilizadas, como simétricos ou assimétricos. Também podem ser classificados quanto ao tratamento que é realizado nas informações que serão processadas, como por blocos ou por fluxo [17].

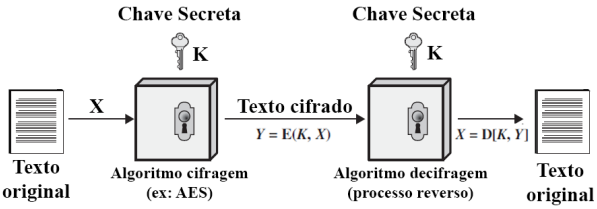

Figura 1: Esquema de criptografia simétrica. Adaptado [17].

O conceito de criptografia simétrica é a unicidade da chave de cifragem e decifragem, tanto o emissor quanto o destinatário compartilham a mesma chave para comunicação. Esses, por sua vez, combinam previamente, através de um canal seguro, qual chave será usada, assim, o remetente cifra sua mensagem com essa chave e envia para o destinatário, que utiliza a mesma chave para decifrar a mensagem recebida e ler seu conteúdo. A Figura 1 mostra um esquema para o modelo simétrico. São exemplos de algoritmos simétricos: DES, 3DES, AES, RC4, e Blowfish. Como forma de solucionar a deficiência da distribuição de chaves no modelo simétrico, foi proposto o modelo assimétrico de criptografia. Neste modelo, não é usada apenas uma chave, mas sim um par de chaves. O princípio básico desse modelo é a existência de uma chave pública, ou seja, que deve ser divulgada a todos, e uma chave privada, que deve ser guardada em segredo. O remetente utiliza a chave pública do destinatário para cifrar sua mensagem, enquanto o destinatário utiliza sua chave privada para decifrar tal mensagem. São exemplos de algoritmos assimétricos: ElGamal, RSA e ECC.

A criptografia por blocos é aquela onde a informação original é dividida em blocos de tamanho n, que são tratados para produzir blocos cifrados também de tamanho n, onde n geralmente equivale a 64 ou 128 *bits*. Remetente e destinatário compartilham a mesma chave de criptografia, que é utilizada tanto no processo de cifragem quanto na decifragem. Por sua vez, na criptografia por fluxo os *bits* ou *bytes* da informação original são cifrados individualmente. Esse processo pode ser realizado por meio da aplicação da operação lógica ou-exclusivo (XOR - $\bigoplus$) entre os bits da informação original e os bits de uma cadeia binária gerada, por exemplo, por uma função pseudoaleatória. Ao projetar um criptossistema baseado em fluxo, deve-se atentar a qualidade do gerador de sequência pseudoaleatória, garantindo assim a dificuldade em quebrar a mensagem cifrada. Além disso deve-se atentar para a necessidade da implementação desse gerador em ambos os lados comunicantes, já que dado uma semente, a mesma sequência deverá ser gerada tanto para o processo de cifragem quanto para a decifragem. Na Figura 2 é mostrado um esquema do processo de cifragem e decifragem baseado em fluxo.

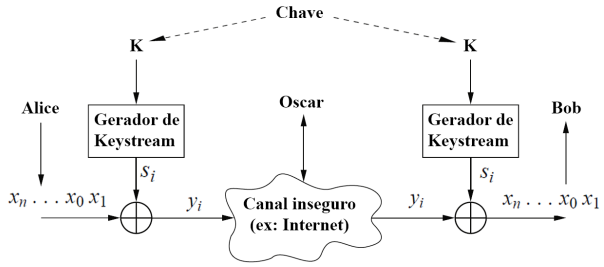

Figura 2: Esquema de criptografia por fluxo. Adaptado [12].

3. PROJETO ESTREAM

A fase final do projeto foi divulgada em 2008 e era composta por um portfólio de quatro algoritmos de Perfil I e outros quatro de Perfil II. Após a descoberta de fraquezas no algoritmo de F-FCSR-H (Perfil II), a lista foi revisada, removendo tal algoritmo [14]. Até o momento, o projeto é constituído de sete sistemas de cifra por fluxo, são elas, Perfil I: HC-128, Rabbit, SALSA20/12 e SOSEMANUK; Perfil II: Grain V1, MICKEY v2 e Trivium [14]. Em 2012, quatro anos após a divulgação do portfólio final, o ECRYPT divulgou outro documento [5], onde afirma que após quatro anos de testes na segurança e tentativas de criptoanálise nos algoritmos, nenhum desses havia tido sua segurança comprometida, e reafirmou que o projeto continua sendo mantido e que qualquer modificação no portfólio será anunciada no site oficial do projeto [6]. Nas próximas seções serão descritos os algoritmos de Perfil II, que são objetos de estudo deste trabalho

3.1 Trivium

Desenvolvido por Christophe De Canniere e Bart Preneel [2][3], Trivium foi concebido com o propósito de explorar a simplicidade sem sacrificar a segurança, velocidade ou flexibilidade. Seu projeto é baseado na combinação de três registradores de deslocamento acrescido de componentes não lineares usados na saída de cada um desses registradores [12].

O sistema foi projetado para gerar até 2^{64} *bits* de valores *(keystream)* utilizando uma chave de 80 *bits* e vetor de inicialização *(Initializion Vector - IV)* de 80 *bits*. Assim como a maioria das cifras de fluxo, o funcionamento do Trivium é dividido em duas fases, a primeira consiste na inicialização dos estados internos do sistema usando o valor da chave e do IV, em seguida o sistema é repetidamente atualizado a fim de gerar cada um dos *bits* do *keystream* [3]. A estrutura do sistema, mostrada na Figura 3, é composta por três registradores de deslocamento, A, B e C, de tamanhos 93, 84 e 111 respectivamente. O fluxo de *bits* de saída é gerado por meio de uma função XOR dos valores de saída de cada registrador de deslocamento. Como a saída de cada registrador é conectada à entrada do outro, o sistema pode ser compreendido de maneira circular, ou seja, um único registrador de deslocamento de tamanho $93 + 84 + 111 = 288$ [12].

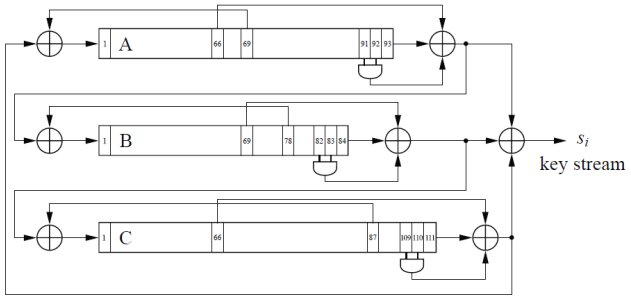

Figura 3: Estrutura interna do Trivium [12].

3.2 Grain

O sistema de criptografia Grain [8] tem como objetivo primário estabelecer um sistema de cifra por fluxo que agregue simplicidade, segurança e velocidade de processamento. Como exemplo de simplicidade os autores citam a importância de uma cifra capaz de ser acoplada no *hardware* de uma *tag* RFID, provendo assim segurança no tráfego de informações, por exemplo em sistemas de pagamentos que utilizam tal tecnologia. Sistemas clássicos como é o caso, por exemplo, do AES não são adequados para essa finalidade, porém, com o Grain essa aplicação seria viável e segura. A primeira versão funcional do algoritmo, denominada de Grain v1 [8], é baseado em cifra por fluxo síncrona, ou seja, o fluxo de *bits* aleatórios *(keystream)* é independente dos *bits* originais.

A construção do Grain consiste em dois registradores de deslocamento, sendo um linear i.e. *Linear Feedback Shift Register (LFSR)* e outro não linear i.e. *Nonlinear Feedback Shift Register (NFSR)*. Ambos registradores têm comprimento igual a 80 *bits*, o sistema utiliza chave de 80 *bits* e IV de 64 *bits*. O Grain foi criado para ser resistente a qualquer tipo de ataque exceto o de força bruta, que necessitaria de uma complexidade computacional não menor do que 2^{80} para quebrar a cifra. Um esquemático da construção do sistema é mostrado na Figura 4, de maneira sucinta o sistema é composto pela combinação de três funções, são elas:

- $f(x)$**:** polinômio primitivo para construção do LFSR.
- $g(x)$**:** polinômio para construção do NFSR.
- $h(x)$**:** função que atua como um filtro não linear entre saídas intermediárias tanto do LFSR quanto do NFSR.

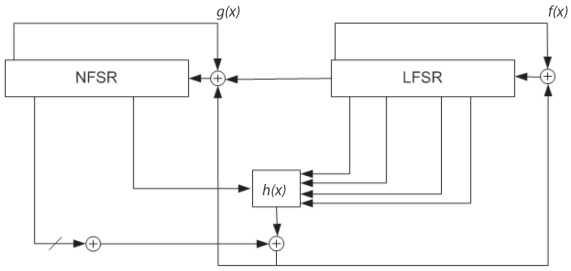

Figura 4: Estrutura interna do Grain [8].

3.3 Mickey

Desenvolvido por Steve Babbage e Matthew Dodd, o sistema de criptografia por fluxo MICKEY *(**M**utual **I**rregular **C**locking **KEY**stream generator)* foi projetado para estruturas de *hardware* com restrições de recursos, tendo assim baixa complexidade de construção, além de prover alto nível de segurança. Atualmente o sistema recomendado no portfólio do projeto eSTREAM é a sua segunda versão, MICKEY 2.0 [1]. A primeira versão foi retirada do portfólio após serem encontradas fraquezas em sua segurança [7].

O sistema utiliza uma chave secreta (K) de 80 *bits* além de um IV também de 80 *bits*, sendo capaz de gerar uma cadeia com tamanho igual a 2^{40} *bits* com um único par (K, IV), podendo fixar o valor de K e alterar o valor do IV. De modo geral o sistema é composto por dois registradores de deslocamento, cada um com comprimento igual a 100 *bits*. O primeiro desses, chamado de **R** atua como registrador linear, já o segundo, chamado de **S** atua como registrador não linear. O MICKEY é dito ter *clock* irregular pois o sinal de *clock* do registrador R é influenciado pelo estado do registrador S, e vice-versa, o que contribui para a segurança do sistema contra variados tipos de ataque além de prover garantias no período dos registradores e a qualidade pseudo-aleatória das sequências geradas [1]. A estrutura do sistema é mostrada na Figura 5.

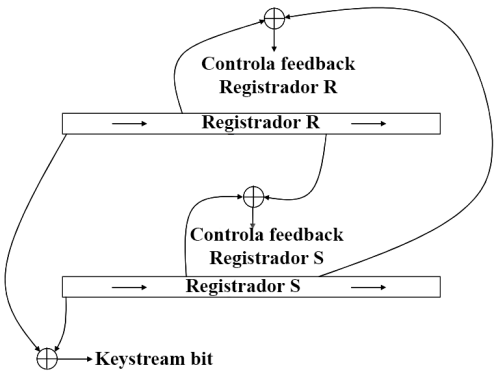

Figura 5: Estrutura interna do MICKEY 2.0. Adaptado [1].

4. METODOLOGIA

Para análise da qualidade dos algoritmos de criptografia torna-se necessário a reprodutibilidade dos experimentos. Uma forma de garantir isso é utilizar um conjunto de dados de entrada padronizados comumente chamado de *dataset*. Nesse trabalho foi escolhido o *dataset* ***USC-SIPI image database, miscellaneous volume*** da University of

Southern California [19]. Esse é composto por um total de 44 imagens, sendo 16 coloridas e 28 em escala de cinza, 14 imagens possuem dimensão igual a $256*256$ *pixels*, 26 iguais a $512*512$ e 4 iguais a $1024*1024$. A Figura 6 exibe a miniatura de todas as imagens desse *dataset*.

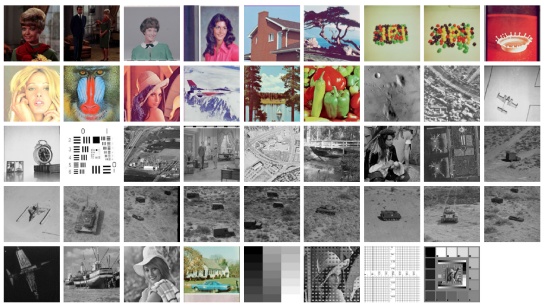

Figura 6: Conjunto de imagens *miscellaneous*.

Após definição do *dataset* foi escolhido um conjunto de métricas utilizadas na literatura para definir se os algoritmos são ou não adequados para o processo de criptografia de imagens. Foram escolhidas como forma de avaliação, a análise visual das imagens processadas, análise de histograma, de entropia, de correlação, de sensibilidade e por fim, a análise diferencial. Para tornar o processo de análise automatizado foram criados *scripts* com auxílio da ferramenta Matlab R2015a [10]. Esses são capazes de realizar o processo de cifragem/decifragem além de calcular as métricas estabelecidas. Após cada execução os resultados numéricos são salvos em arquivos ".csv" para posterior síntese estatística. Para cada execução são salvos também as imagens após cada etapa de processamento e gráficos (histogramas, correlação e sensibilidade). Um esquemático do sistema de análise é mostrado na Figura 7. A implementação dos sistemas de criptografia analisados foi realizada em *software* a partir de adaptações nos códigos ".c" disponibilizados pelos seus respectivos autores [6]. Os leitores interessados podem acessar os *scripts*, códigos dos geradores e dados de análise em https://goo.gl/SNMwgd.

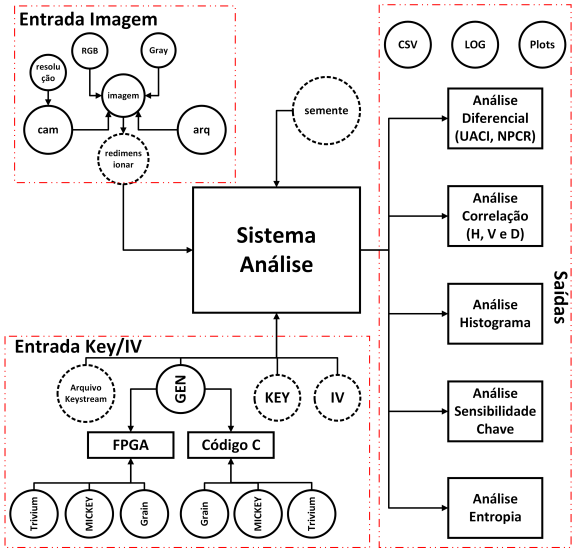

Figura 7: Esquemático do sistema de análise automatizada.

Para reprodutibilidade dos experimentos, foram salvos 50 geradores aleatórios no Matlab, assim, cada uma das 44 imagens foi executada 50 vezes, cada i-ésima execução utilizando o i-ésimo gerador, sem repetir nenhum. Esse processo foi realizado para cada um dos três algoritmos analisados, totalizando $44*50*3 = 6600$ execuções. Os resultados foram sintetizados por meio do cálculo da média e desvio padrão.

5. RESULTADOS

Essa seção explora os resultados obtidos para cada métrica considerada na análise dos algoritmos aplicados às imagens do *dataset*. No artigo é exibido detalhadamente apenas quatro imagens desse conjunto, duas delas coloridas (RGB): "`4.2.04`" e "`4.1.03`", além de duas em tons de cinza: "`boat.512`" e "`7.1.02`". Apesar de escolhidas apenas essas quatro imagens, os resultados para elas apresentados refletem os resultados encontrados para o *dataset* como um todo.

5.1 Análise Visual

Após cifragem de qualquer imagem espera-se de um algoritmo de criptografia duas coisas. A primeira é que a imagem cifrada não contenha nenhum traço de informação da imagem original, isso é, se pareça o máximo possível com um ruído aleatório. A segunda característica é que após o processo de decifragem com a chave correspondente, a imagem decifrada seja exatamente igual à original. Dessa maneira, na Figura 8 é mostrada a eficiência dos algoritmos quanto à análise visual quando aplicados sobre imagens coloridas (RGB), enquanto na Figura 9 essa análise é mostrada para imagens em tons de cinza.

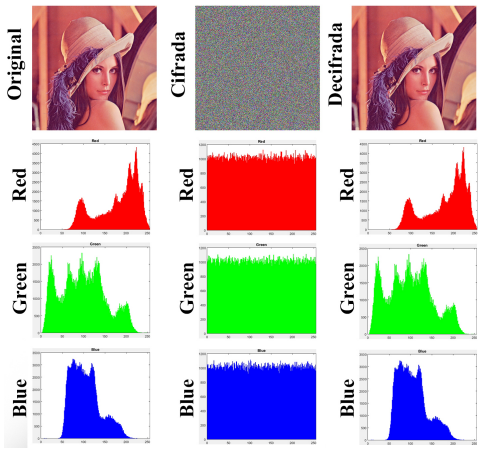

Figura 8: Imagem "4.2.04" - algoritmo Grain.

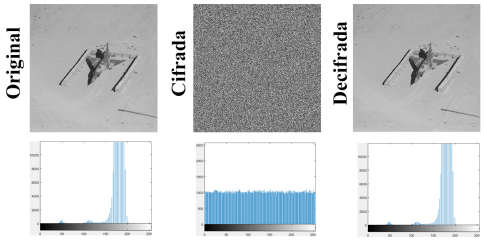

Figura 9: Imagem "7.1.02" - algoritmo Trivium.

5.2 Análise de Histograma

Um histograma descreve de forma gráfica a distribuição estatística da intensidade das cores de uma imagem. Esquemas de criptografia seguros são caracterizados por um histograma uniforme em cada canal de cor. As Figuras 8 e 9 exibem os histogramas das imagens originais, cifradas e decifradas. Além da análise gráfica, calcula-se o valor médio de intensidade dos *pixels* de cada imagem a fim de comprovar o comportamento uniforme dessa. Assim, é esperado em uma imagem cifrada com valores de *pixel* variando de $0 \ldots 255$, um valor médio o mais próximo possível de $\frac{a+b}{2} = \frac{0+255}{2} = \frac{255}{2} = 127.5$. A Tabela 1 exibe a intensidade média para as imagens em tons de cinza, enquanto a Tabela 2 exibe para coloridas (RGB). Ambas tabelas comprovam a uniformidade das imagens cifradas pelos três algoritmos.

Tabela 1: Valor médio dos *pixels* - *grayscale*.

Arquivo	Original	Trivium	Grain	Mickey
boat.512	129.707966	127.452536 ± 0.1469	127.509175 ± 0.1220	127.525035 ± 0.1304
7.1.02	175.334938	127.454014 ± 0.1480	127.498130 ± 0.1226	127.522193 ± 0.1343

Fonte: Do autor (2017)

Tabela 2: Valor médio dos *pixels* - RGB.

	Original		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	137.602783	139.957764	144.017792
4.2.04	180.223660	99.051216	105.410252

	Trivium		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	127.539200 ± 0.2486	127.506234 ± 0.3091	127.427942 ± 0.2857
4.2.04	127.489926 ± 0.1379	127.488449 ± 0.1551	127.518575 ± 0.1544

	Grain		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	127.514123 ± 0.2774	127.392363 ± 0.3005	127.529581 ± 0.3293
4.2.04	127.470704 ± 0.1194	127.525226 ± 0.1500	127.502625 ± 0.1174

	Mickey		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	127.517062 ± 0.2472	127.513598 ± 0.2873	127.521242 ± 0.2621
4.2.04	127.500020 ± 0.1462	127.503870 ± 0.1376	127.519766 ± 0.1567

Fonte: Do autor (2017)

5.3 Entropia

É a medida da previsibilidade de uma fonte aleatória. Para evitar que uma imagem cifrada possa sofrer algum vazamento de informações contidas nela, é necessário que tal imagem seja parecida como um ruído aleatório. Para uma fonte binária S que produz 2^8 símbolos cada um com iguais probabilidades, a entropia pode ser definida como na Equação 1, onde $P(S_i)$ é a probabilidade do símbolo S_i [11]. Em uma fonte realmente aleatória capaz de produzir 2^N símbolos diferentes, o valor da entropia é igual a N [13]. As Tabelas 3 e 4 exibem os valores de entropia para as imagens originais e cifradas, tanto das imagens coloridas quanto em tons de cinza. Os valores encontrados nas imagens cifradas são extremamente próximos do valor ótimo $E = 8.0$.

$$Entropy = -\sum_{i=1}^{2^8} P(S_i) \log 2P(S_i) \quad (1)$$

Tabela 3: Entropia das imagens *grayscale*.

Arquivo	Original	Trivium	Grain	Mickey
boat.512	7.191370	7.999305 ± 0.0001	7.999311 ± 0.0001	7.999308 ± 0.0001
7.1.02	4.004499	7.999293 ± 0.0000	7.999302 ± 0.0001	7.999307 ± 0.0001

Fonte: Do autor (2017)

Tabela 4: Entropia das imagens RGB.

	Original		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	5.715009	5.373845	5.711657
4.2.04	7.253102	7.594038	6.968427

	Trivium		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	7.997215 ± 0.0002	7.997191 ± 0.0003	7.997202 ± 0.0003
4.2.04	7.999291 ± 0.0001	7.999294 ± 0.0001	7.999304 ± 0.0001

	Grain		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	7.997244 ± 0.0002	7.997206 ± 0.0003	7.997205 ± 0.0003
4.2.04	7.999295 ± 0.0001	7.999287 ± 0.0001	7.999313 ± 0.0001

	Mickey		
Arquivo	**Red**	**Green**	**Blue**
4.1.03	7.997201 ± 0.0002	7.997209 ± 0.0002	7.997145 ± 0.0003
4.2.04	7.999291 ± 0.0001	7.999295 ± 0.0001	7.999304 ± 0.0001

Fonte: Do autor (2017)

5.4 Correlação

Figura 10: Alta correlação na imagem original 4.2.04 (detalhe ampliado na imagem à esquerda).

Uma baixa correlação entre *pixels* adjacentes indica um sistema de criptografia robusto. A correlação entre dois *pixels* adjacentes x e y é dada pela Equação 2, onde $cov(x, y)$ representa a covariância entre os *pixels*, $D(x)$ e $D(y)$ representam respectivamente a variância de x e de y [11]. Nesse trabalho foram analisadas as correlações para *pixels* adjacentes horizontalmente, verticalmente e diagonalmente, para um total de 5000 *pixels*. A Figura 10 demonstra a alta correlação em uma imagem sem criptografia, ou seja, *pixels* adjacentes (figura esquerda) possuem valores muito próximos. As Tabelas 5 e 6 mostram que ambos os algoritmos possuem níveis de correlação próximos de 0 após processo de cifragem. A Figura 11 exibe de forma numérica e gráfica a correlação antes e depois da cifragem da imagem "boat.512".

$$\gamma_{xy} = \frac{cov(x, y)}{\sqrt{D(x)}\sqrt{D(y)}} \tag{2}$$

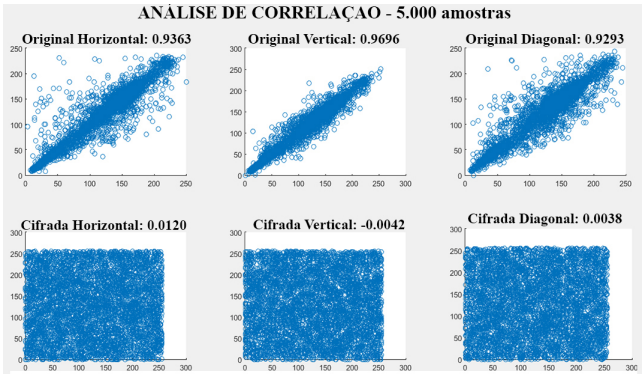

Figura 11: Análise de correlação da imagem "boat.512".

Tabela 5: Correlação das imagens RGB.

Arquivo	Original Horz	Original Vert	Original Diag
4.1.03	0.975691	0.916755	0.902670
4.2.04	0.961044	0.976239	0.946753

Arquivo	Trivium Horz	Trivium Vert	Trivium Diag
4.1.03	-0.001031 ± 0.0094	-0.000418 ± 0.0092	0.000418 ± 0.0088
4.2.04	0.002112 ± 0.0090	-0.000955 ± 0.0092	-0.000955 ± 0.0092

Arquivo	Grain Horz	Grain Vert	Grain Diag
4.1.03	0.000208 ± 0.0082	0.002067 ± 0.0079	-0.000948 ± 0.0086
4.2.04	-0.001726 ± 0.0077	0.000492 ± 0.0078	0.000492 ± 0.0078

Arquivo	Mickey Horz	Mickey Vert	Mickey Diag
4.1.03	0.001078 ± 0.0085	0.002020 ± 0.0093	-0.000864 ± 0.0091
4.2.04	0.000217 ± 0.0089	-0.000785 ± 0.0098	-0.000785 ± 0.0098

Fonte: Do autor (2017)

Tabela 6: Correlação das imagens *grayscale*.

Arquivo	Original Horz	Original Vert	Original Diag
boat.512	0.937101	0.971190	0.922101
7.1.02	0.943854	0.945249	0.898709

Arquivo	Trivium Horz	Trivium Vert	Trivium Diag
boat.512	-0.000527 ± 0.0135	-0.001804 ± 0.0143	-0.001804 ± 0.0143
7.1.02	-0.000756 ± 0.0158	-0.002667 ± 0.0134	0.000298 ± 0.0155

Arquivo	Grain Horz	Grain Vert	Grain Diag
boat.512	-0.000458 ± 0.0120	-0.000953 ± 0.0145	-0.000953 ± 0.0145
7.1.02	-0.000433 ± 0.0123	0.000193 ± 0.0145	-0.002695 ± 0.0152

Arquivo	Mickey Horz	Mickey Vert	Mickey Diag
boat.512	0.003101 ± 0.0138	-0.001335 ± 0.0144	-0.001335 ± 0.0144
7.1.02	-0.000637 ± 0.0169	-0.001005 ± 0.0138	0.001643 ± 0.0140

Fonte: Do autor (2017)

5.5 Análise de Sensibilidade

Para que um algoritmo de criptografia possa ser seguro é desejável que ao realizar uma pequena mudança na chave de cifragem (alterar apenas um *bit*) a correlação entre a imagem cifrada com a chave original (K1) e a imagem cifrada com a chave alterada (K2), seja próximo de 0. O mesmo processo vale para o processo de decifragem. De modo geral pode-se dizer que é desejável que uma pequena mudança na chave gere uma enorme mudança na imagem cifrada.

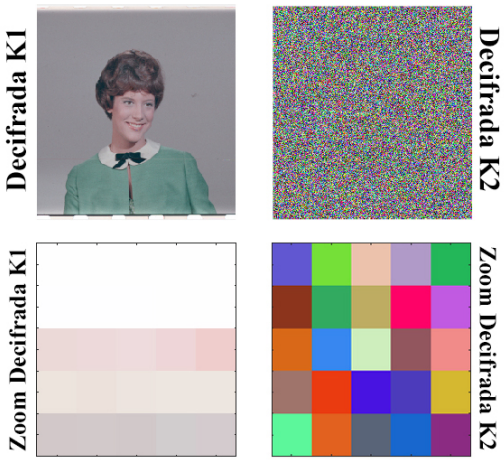

Figura 12: Alta sensibilidade a variação de chave.

As Tabelas 7 e 8 mostram que todos os três algoritmos possuem correlação próxima de 0 considerando K1 e K2, reforçando a qualidade desses nessa aplicação. A Figura 12 demonstra a alta sensibilidade encontrada no processo de decifragem utilizando o algoritmo Trivium. Na imagem de título "Decifrada K1" a amostra "4.1.03" do *dataset* é deci-

frada com a chave original (K1), em detalhe na figura "Zoom Decifrada K1" é mostrada essa imagem ampliada. Já a imagem "Decifrada K2" exibe a mesma amostra "4.1.03", porém, decifrada com a chave alterada em um *bit* (K2). Comparando as imagens resultantes dos processos de decifragem observa-se que elas não são de modo algum semelhantes, corroborando assim para a conclusão tomada na observação das Tabelas 7 e 8.

Tabela 7: Correlação entre imagem cifrada (K1) e imagem cifrada (K2).

Arquivo	Trivium	Grain	Mickey
4.1.03	-0.000217 ± 0.0020	-0.000129 ± 0.0021	0.000279 ± 0.0025
4.2.04	0.000134 ± 0.0012	-0.000190 ± 0.0011	-0.000217 ± 0.0009
boat.512	-0.000018 ± 0.0020	0.000075 ± 0.0018	0.000210 ± 0.0022
7.1.02	0.000058 ± 0.0018	-0.000401 ± 0.0021	-0.000183 ± 0.0024

Fonte: Do autor (2017)

Tabela 8: Correlação entre imagem decifrada (K1) e imagem decifrada (K2).

Arquivo	Trivium	Grain	Mickey
4.1.03	-0.000333 ± 0.0027	-0.000126 ± 0.0028	-0.000275 ± 0.0031
4.2.04	0.000064 ± 0.0013	-0.000243 ± 0.0015	0.000033 ± 0.0014
boat.512	-0.000204 ± 0.0018	-0.000123 ± 0.0017	0.000148 ± 0.0018
7.1.02	0.000056 ± 0.0021	0.000094 ± 0.0019	-0.000276 ± 0.0018

Fonte: Do autor (2017)

5.6 Análise Diferencial

Pequenas alterações na imagem original devem resultar em alterações significativas na imagem cifrada para que sejam evitadas quaisquer relações estatísticas entre a entrada e a saída do sistema de criptografia. A sensibilidade de um sistema quanto aos dados de entrada pode ser calculado por duas métricas, são elas [9]:

- ***Number of Pixels Change Rate (NPCR)*:** representa a taxa de *pixels* modificados na imagem cifrada enquanto apenas um *pixel* é alterado na imagem original. Assumindo $C1$ e $C2$ como imagens cifradas provenientes de duas imagens com apenas um *pixel* de diferença respectivamente, o cálculo pode ser realizado por meio da Equação 3, onde M e N correspondem a largura e altura da imagem respectivamente.

$$D(i,j) = \begin{cases} 0, & C_1(i,j) = C_2(i,j) \\ 1, & C_1(i,j) \neq C_2(i,j) \end{cases} \tag{3a}$$

$$NPCR = \frac{1}{MxN} \sum_{i=1}^{M} \sum_{j=1}^{N} D(i,j) \tag{3b}$$

- ***Unified Average Changing Intensity (UACI)*:** diferença de intensidade média entre duas imagens cifradas, cujas imagens originais são diferentes por apenas um *pixel*, o calculo é realizado por meio da Equação 4.

$$UACI = \frac{1}{MxN} \sum_{i=1}^{M} \sum_{j=1}^{N} \left(\frac{|C_1(i,j) - C_2(i,j)|}{255} \right) x100 \tag{4}$$

As Tabelas 9 e 11 exibem os valores calculados para a métrica NPCR, enquanto que as Tabelas 10 e 12 exibem os valores para a métrica UACI. Os valores encontrados para NPCR das imagens são maiores que os limiares críticos $N^*_{0.05}$, $N^*_{0.01}$ e $N^*_{0.001}$, enquanto que os valores para UACI também encontram-se contidos nos mesmos intervalos de confiança [20]. Essa análise demonstra que ambos os algoritmos são extremamente sensíveis a pequenas mudanças na imagem original, garantindo assim a segurança.

Tabela 9: NPCR cifradas - RGB.

	Trivium		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.996131 ± 0.0003	0.996056 ± 0.0002	0.996068 ± 0.0002
4.2.04 (512)	0.996065 ± 0.0001	0.996095 ± 0.0001	0.996092 ± 0.0002

	Grain		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.996123 ± 0.0003	0.996117 ± 0.0003	0.996062 ± 0.0003
4.2.04 (512)	0.996114 ± 0.0001	0.996082 ± 0.0001	0.996065 ± 0.0001

	Mickey		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.996108 ± 0.0003	0.996086 ± 0.0002	0.996149 ± 0.0003
4.2.04 (512)	0.996108 ± 0.0001	0.996084 ± 0.0001	0.996098 ± 0.0001

Fonte: Do autor (2017)

Tabela 10: UACI cifradas - RGB.

	Trivium		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.334795 ± 0.0010	0.334582 ± 0.0010	0.334680 ± 0.0008
4.2.04 (512)	0.334652 ± 0.0004	0.334658 ± 0.0004	0.334669 ± 0.0005

	Grain		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.334646 ± 0.0009	0.334440 ± 0.0008	0.334673 ± 0.0011
4.2.04 (512)	0.334680 ± 0.0005	0.334550 ± 0.0005	0.334629 ± 0.0005

	Mickey		
Arquivo	**Red**	**Green**	**Blue**
4.1.03 (256)	0.334463 ± 0.0009	0.334399 ± 0.0011	0.334919 ± 0.0009
4.2.04 (512)	0.334650 ± 0.0005	0.334639 ± 0.0004	0.334692 ± 0.0005

Fonte: Do autor (2017)

Tabela 11: NPCR cifradas - *grayscale*.

Arquivo	Trivium	Grain	Mickey
boat.512	0.996065	0.996114	0.996108
(512)	± 0.0001	± 0.0001	± 0.0001
7.1.02	0.996065	0.996114	0.996108
(512)	± 0.0001	± 0.0001	± 0.0001

Fonte: Do autor (2017)

Tabela 12: UACI cifradas - *grayscale*.

Arquivo	Trivium	Grain	Mickey
boat.512	0.334730	0.334695	0.334666
(512)	± 0.0004	± 0.0004	± 0.0005
7.1.02	0.334744	0.334681	0.334657
(512)	± 0.0005	± 0.0005	± 0.0004

Fonte: Do autor (2017)

6. CONCLUSÕES

Após o processamento do *dataset* e análise dos três algoritmos por meio das métricas citadas na literatura, conclui-se que os algoritmos realizaram de maneira segura e confiável o processo de criptografia de imagens. Essa análise é de grande valia para o desenvolvimento futuro de sistemas de segurança em *hardware*, em especial aqueles com limitação de recursos. Como ambos os sistemas obtiveram resultados muito próximos, a escolha entre qual sistema de criptografia deve ser utilizado está condicionada a uma posterior análise de desempenho e recursos nas implementações em *hardware*. A aplicação de sistemas seguros para transmissão de imagens, seja de maneira estática (foto) ou dinâmica (vídeo), em tempo real ou não, é hoje área de grande interesse tanto no meio acadêmico quanto cada vez mais também para o público em geral. Esse artigo contribui para a comunidade acadêmica fornecendo uma base empírica e estocástica para futuros trabalhos utilizando os algoritmos de Perfil II do projeto eSTREAM ao mostrar a qualidade dos mesmos.

Agradecimentos

Os autores agradecem a Universidade Federal de Lavras, em especial ao Departamento de Ciência da Computação pelo apoio e estrutura física, à FAPEMIG que, por meio do suporte financeiro, viabilizou a realização desse trabalho, e aos colegas do LABSINE pelo companheirismo.

7. REFERÊNCIAS

[1] S. Babbage and M. Dodd. The stream cipher mickey 2.0. *ECRYPT Stream Cipher*, 2006.

[2] C. D. Cannière. Trivium: A stream cipher construction inspired by block cipher design principles. In *Lecture Notes in Computer Science*, pages 171–186. Springer Science Business Media, 2006.

[3] C. D. Cannière and B. Preneel. Trivium specifications. techreport, ECRYPT, 2007.

[4] G. Chen, Y. Mao, and C. K. Chui. A symmetric image encryption scheme based on 3D chaotic cat maps. *Chaos, Solitons & Fractals*, 21(3):749–761, jul 2004.

[5] C. Cid and M. Robshaw. *The eSTREAM Portfolio in 2012: ECRYPT II report D.SYM.10*. ECRYPT II, 1 2012.

[6] ECRYPT II. estream: the ecrypt stream cipher project, 2012.

[7] B. Gierlichs, L. Batina, C. Clavier, T. Eisenbarth, A. Gouget, H. Handschuh, T. Kasper, K. Lemke-Rust, S. Mangard, A. Moradi, et al. Susceptibility of estream candidates towards side channel analysis. *Proceedings of SASC*, pages 123–150, 2008.

[8] M. Hell, T. Johansson, and W. Meier. Grain: a stream cipher for constrained environments. *International Journal of Wireless and Mobile Computing*, 2(1):86, 2007.

[9] A. S. Mansingka, M. L. Barakat, A. G. Radwan, and K. N. Salama. Hardware stream cipher with controllable chaos generator for colour image encryption. *IET Image Processing*, 8(1):33–43, jan 2014.

[10] Mathworks. R2015a release highlights, 2015.

[11] L. Merah, A. Ali-Pacha, and N. Hadj-Said. Real-time cryptosystem based on synchronized chaotic systems. *Nonlinear Dynamics*, 82(1-2):877–890, oct 2015.

[12] C. Paar and J. Pelzl. *Understanding Cryptography*. Springer Science Business Media, 2010.

[13] M. T. Ramirez-Torres, J. S. Murguia, and M. Mejia-Carlos. FPGA implementation of a reconfigurable image encryption system. In *2014 International Conference on ReConFigurable Computing and FPGAs (ReConFig14)*, pages 1–4. IEEE, dec 2014.

[14] V. Rijmen. Stream ciphers and the estream project. *The ISC Int'l Journal of Information Security*, 2(1):3–11, jan 2010.

[15] S. Rohith, K. N. H. Bhat, and A. N. Sharma. Image encryption and decryption using chaotic key sequence generated by sequence of logistic map and sequence of states of linear feedback shift register. In *2014 International Conference on Advances in Electronics Computers and Communications*. Institute of Electrical & Electronics Engineers (IEEE), oct 2014.

[16] K. M. Shruthi, S. Sheela, and S. V. Sathyanarayana. Image encryption scheme with key sequences based on chaotic functions. In *2014 International Conference on Contemporary Computing and Informatics (IC3I)*. Institute of Electrical & Electronics Engineers (IEEE), nov 2014.

[17] W. Stallings. *Cryptography and Network Security: Principles and Practice*. Prentice Hall Press, Upper Saddle River, NJ, USA, 5th edition, 2010.

[18] R. Terada. *Segurança de dados: criptografia em redes de computador*. Blucher, São Paulo, 1st edition, 2000.

[19] University of Southern California. Sipi image database, 2017.

[20] Y. Wu, J. P. Noonan, and S. Agaian. Npcr and uaci randomness tests for image encryption. *Cyber journals: multidisciplinary journals in science and technology, Journal of Selected Areas in Telecommunications (JSAT)*, pages 31–38, 2011.