FM-223: Exame

Vinícius Freitas de Almeida[1]

Resumo
Este relatório apresenta uma análise do artigo *Cracking a hierarchical chaotic image encryption algorithm based on permutation*

Palavras-chave
Caos — Criptografia — Permutação

[1] *Aluno do quarto ano da graduação de Engenharia Eletrônica, no Instituto Tecnológico de Aeronáutica, São José dos Campos, Brasil*
E-mail correspondente: [2] vinicius.almeida@ga.ita.br

1 Introdução

Observação: na notação aqui utilizada, os índices de matrizes, vetores e sequências começam por 0. Além disso, denota-se matrizes e vetores em negrito, sendo as matrizes denotadas por letras maiúsculas e vetores por letras minúsculas.

1.1 Representação vetorial de uma imagem

Toda imagem em escala de cinza (as únicas nas quais estamos interessados nesta prática) pode ser representada matematicamente por uma matriz de números reais $\mathbf{X} \in \mathbb{R}^{m\times n}$. No entanto, uma representação mais versátil para demonstrar resultados com álgebra linear (e coerente com o *layout* de memória de um computador (6)) consiste em organizar as matrizes $\mathbb{R}^{m\times n}$ como vetores-coluna $\mathbb{R}^{mn}$, isto é, em ordem linear. O mapeamento de índices da matriz $\mathbf{X} \in \mathbb{R}^{m\times n}$ para um vetor-coluna $\mathbf{x} \in \mathbb{R}^{mn}$ foi escolhido arbitrariamente para ser em *row-major order* (padrão para linguagem C e derivados) e é dado por:

$$k = i \cdot n + j \tag{1}$$

Inversamente:

$$\begin{cases} i = \left\lfloor \frac{k}{n} \right\rfloor \\ j = k \mod n \end{cases} \tag{2}$$

Para uma matriz $\mathbf{X} \in \mathbb{R}^{4\times 5}$, por exemplo, esse é o mapeamento dos índices para o vetor $\mathbf{x} \in \mathbb{R}^{20}$:

$$\mathbf{X} = \begin{bmatrix} x_0 & x_1 & x_2 & x_3 & x_4 \\ x_5 & x_6 & x_7 & x_8 & x_9 \\ x_{10} & x_{11} & x_{12} & x_{13} & x_{14} \\ x_{15} & x_{16} & x_{17} & x_{18} & x_{19} \end{bmatrix} \iff \mathbf{x} = [x_0, x_1, x_2, x_3, \ldots, x_{17}, x_{18}, x_{19}]^\top \tag{3}$$

Utilizando a representação vetorial para imagens $\mathbf{X} \in \mathbb{R}^{m\times n}$, $\mathbf{Y} \in \mathbb{R}^{m\times n}$ (i.e. $\mathbf{X} \to \mathbf{x}$ e $\mathbf{Y} \to \mathbf{y}$), qualquer mapa linear $X \mapsto Y$ pode ser definido como um operador linear $A : \mathbb{R}^{mn} \to \mathbb{R}^{mn}$, isto é, uma matriz $\mathbf{A} \in \mathbb{R}^{mn\times mn}$.

1.2 Propriedades das matrizes de permutação

Uma matriz de permutação $\mathbf{P}$ é uma matriz quadrada cujas linhas e colunas são permutações de uma matriz identidade $\mathbf{I}$, isto é, uma matriz com apenas um elemento 1 em cada linha e coluna e todos os demais elementos nulos. Por exemplo:

$$\mathbf{P} = \begin{bmatrix} 0 & 1 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 0 & 0 & 1 & 0 \end{bmatrix} \tag{4}$$

Em geral, uma matriz de permutação $\mathbf{P} \in \mathbb{R}^{n\times n}$ tem as seguintes propriedades:

- **Inversível**, uma vez que $\mathbf{P}^{-1} = \mathbf{P}^\top$.
- **Ortogonal**, uma vez que $\mathbf{P}^{-1} = \mathbf{P}^\top$.
- **Esparsa**, uma vez que cada linha e coluna de $\mathbf{P}$ tem apenas um elemento não-nulo.

Como abordado na seção anterior, uma matriz de permutação $\mathbf{P} \in \mathbb{R}^{mn\times mn}$ pode ser vista como um operador linear $P : \mathbb{R}^{mn} \mapsto \mathbb{R}^{mn}$, capaz de permutar os elementos de uma imagem representada pelo vetor-coluna $\mathbf{x} \in \mathbb{R}^{mn}$ por meio de sua aplicação $\mathbf{y} = \mathbf{P}\mathbf{x}$.

Assim, fica claro que uma matriz de permutação $\mathbf{P}$ pode ser utilizada para embaralhar os elementos de uma imagem, o que é uma das operações básicas do algoritmo de criptografia aqui abordado.

1.3 O mapa logístico

O mapa logístico é um mapa caótico que pode ser utilizado para gerar números pseudoaleatórios. Sua fórmula é:

$$x_{n+1} = \mu x_n (1 - x_n) \tag{5}$$

em que x_n é o n-ésimo elemento da sequência, μ é o parâmetro de controle e x_0 é a condição inicial.

A motivação para usá-lo como PRNG é que, para certos valores de μ, a sequência gerada é caótica, ou seja, é imprevisível e tem alta sensibilidade às condições iniciais.

Na prática, a fórmula recursiva é implementada com aritmética de ponto flutuante (e, portanto, com precisão finita de L bits). Com os valores x_n gerados com precisão finita, a

sequência de bits aleatórios (b_n) pode ser obtida extraindo-se os bits da representação binária de x_n. O algoritmo da Figura 1 mostra como gerar a sequência (b_n) a partir do mapa logístico. A implementação em particular extrai os 8 bits após o ponto decimal de cada x_n, assim como no artigo (3).

LOGISTICMAPPRNG(x_0, μ, L_b)
1 $x = x_0$
2 **for** $i = 0$ **to** $\lceil L_b/8 \rceil - 1$
3 $\quad x = \mu x(1-x)$
4 $\quad x^* = x - \lfloor x \rfloor$
5 $\quad$ **for** $j = 0$ **to** 7
6 $\quad\quad x^* = 2x^*$
7 $\quad\quad b_{8i+j} = \lfloor x^* \rfloor$
8 $\quad\quad x^* = x^* - \lfloor x^* \rfloor$
9 **return** $(b_0, b_1, \ldots, b_{8\lceil L_b/8 \rceil - 1})$

Figura 1: *A função* `LogisticMapPRNG`.

1.4 Visão geral do HCIE

A criptografia de chave pública é um método de criptografia que utiliza um par de chaves: uma pública e outra privada. A chave pública é conhecida por todos e é utilizada para criptografar mensagens, enquanto a chave privada é conhecida apenas pelo destinatário e é utilizada para descriptografar as mensagens.

No caso do algoritmo *Hierarchical Chaotic Image Encryption*(HCIE), a chave privada é o par (x_0, μ), em que x_0 é a condição inicial do mapa logístico e μ é o parâmetro de controle. A chave pública, por sua vez, é composta pelos parâmetros $S_M, S_N, \alpha, \beta, \gamma, no$. S_M, S_N são as dimensões dos blocos, α, β, γ são inteiros e no é o número de iterações da função `Sub_HCIE`. M, N são as dimensões da imagem (que, evidentemente, são públicas). Além disso:

- $M \equiv 0 \mod S_M$, $\sqrt{M} \leq S_M \leq M$
- $N \equiv 0 \mod S_N$, $\sqrt{N} \leq S_N \leq N$

O algoritmo pode ser descrito, em alto nível, pelas seguintes etapas:

1. **Inicialização:** gerar a sequência de bits aleatórios (b_n) a partir do mapa logístico com os parâmetros x_0 e μ.
2. **Embaralhamento hierárquico:**
   (a) **Embaralhamento em blocos:** dividir a imagem em $M/S_M \times N/S_N$ blocos de dimensão $S_M \times S_N$ e permutar os blocos de acordo com a função `Sub_HCIE`.
   (b) **Embaralhamento dentro dos blocos:** permutar os pixels de cada bloco de acordo com a função `Sub_HCIE`.

SUBHCIE$(f_{sub}, \{b(i)\}, no, S_M, S_N)$
1 **for** $ite = 0$ **to** $no - 1$
2 $\quad q = i_0 + (3S_M + 3S_N - 2) \cdot ite$
3 $\quad p = \alpha + \beta \cdot b(q) + \gamma \cdot b(q+1)$
4 $\quad$ **for** $i = 0$ **to** $S_M - 1$
5 $\quad\quad f'_{sub} = \text{ROLR}^{i,p}_{b(i+q)}(f_{sub})$
6 $\quad$ **for** $j = 0$ **to** $S_N - 1$
7 $\quad\quad f'_{sub} = \text{ROUD}^{i,p}_{b(j+q+S_M)}(f'_{sub})$
8 $\quad$ **for** $k = 0$ **to** $S_M + S_N - 2$
9 $\quad\quad f'_{sub} = \text{ROUR}^{k,p}_{b(k+q+S_M+S_N)}(f'_{sub})$
10 $\quad$ **for** $l = 1 - S_N$ **to** $S_M - 1$
11 $\quad\quad f'_{sub} = \text{ROUL}^{l,p}_{b(l+q+2S_M+3S_N-2)}(f'_{sub})$
12 $io = io + (3S_M + 3S_N - 2) \cdot no$
13 **return** (f'_{sub}, io)

Figura 2: *A função* `Sub_HCIE` *(3).*

Definições das funções auxiliares:

- $\text{ROLR}^{i,p}_{b(i+q)}$: Rotaciona a i-ésima linha de f por p pixels para a esquerda (quando $b = 0$) ou direita (quando $b = 1$).
- $\text{ROUD}^{i,p}_{b(j+q+S_M)}$: Rotaciona a j-ésima coluna de f por p pixels para cima (quando $b = 0$) ou para baixo (quando $b = 1$).
- $\text{ROUR}^{k,p}_{b(k+q+S_M+S_N)}$: Rotaciona todos os pixels que satisfazem $i + j = k$ por p pixels para a parte inferior-esquerda (quando $b = 0$) ou superior-direita (quando $b = 1$).
- $\text{ROUL}^{l,p}_{b(l+q+2S_M+3S_N-2)}$: Rotaciona todos os pixels que satisfazem $i - j = l$ por p pixels para a parte superior-esquerda (quando $b = 0$) ou inferior-direita (quando $b = 1$).

2 Criptanálise

2.1 Ataque de texto cifrado conhecido

O *ciphertext-only attack*, ou ataque de texto cifrado conhecido, é um tipo de ataque criptoanalítico em que o atacante tem acesso apenas ao texto cifrado, sem acesso ao texto plano ou à chave de criptografia. No caso do HCIE, o atacante tem acesso à imagem cifrada e aos parâmetros públicos do algoritmo.

No artigo que introduziu o HCIE (8), os autores afirmam que o algoritmo é robusto contra ataques de texto cifrado conhecido, dizendo que a complexidade de ataques de força bruta é de $O(2^{L_b})$, uma vez que existem $L_b = (1 + M/S_M \cdot N/S_N) \cdot (3S_M + 3S_N - 2) \cdot no$ bits de (b_n), que são desconhecidos. No entanto, o artigo (3) argumenta que, como a sequência (b_n) é gerada a partir de um mapa logístico(i.e. um processo exato e determinístico), é possível recuperar a sequência (b_n) apenas com o conhecimento de $\{x_0, \mu\}$, que contém $2L$ bits, em que L é a precisão da aritmética em ponto flutuante usada.

Por fim, cabe ressaltar que o artigo (3) apresenta limites superiores ainda mais estreitos, uma vez que a escolha do parâmetro μ é limitada a valores próximos de 4, devido à necessidade de que o mapa logístico seja caótico. Assim, é possível atingir um limite superior de complexidade de $O(2^{2L}(L_b + MN))$, que é muito menor do que a estimativa original de $O(2^{L_b/8})$ para valores grandes de L_b.

2.2 Ataque de texto claro conhecido

O *known-plaintext attack*, ou ataque de texto claro conhecido, é um tipo de ataque criptoanalítico em que o atacante tem acesso a pares de texto claro e texto cifrado. No caso do HCIE, o atacante tem acesso a pares de imagens, uma original e outra cifrada, e aos parâmetros públicos do algoritmo.

Como o HCIE é um algoritmo que envolve exclusivamente uma permutação dos pixels da imagem, pode-se aplicar a teoria de matrizes de permutação para analisar a segurança do algoritmo. Em particular, no *known-plaintext attack*, temos tanto a imagem $\mathbf{x}$ quanto a imagem cifrada $\mathbf{y}$, e queremos encontrar a matriz de permutação $\mathbf{W}$ que satisfaz $\mathbf{y} = \mathbf{W}\mathbf{x}$. Não só isso, mas a estrutura hierárquica do HCIE garante uma maior facilidade para encontrar $\mathbf{W}$, uma vez que a permutação é feita em blocos de dimensão $S_M \times S_N$. O artigo (3) propõe um algoritmo para encontrar $\mathbf{W}$ que aproveita dessa estrutura hierárquica, que será analisado logo a seguir. No entanto, primeiro iremos analisar o algoritmo `Get_Permutation_Matrix` proposto no artigo (5), e melhorado no artigo (4).

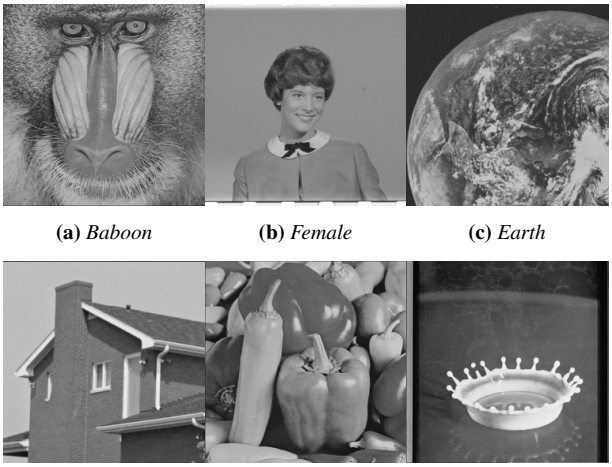
Como explicado na seção 3.1 do artigo (5), o algoritmo `Get_Permutation_Matrix` é baseado na ideia de que, num ataque de texto claro conhecido, o atacante tem acesso aos p pares $\{(\mathbf{X}_0, \mathbf{Y}_0), (\mathbf{X}_1, \mathbf{Y}_1), \ldots, (\mathbf{X}_{p-1}, \mathbf{Y}_{p-1})\}$, em que $\mathbf{X}_i$ é a imagem original e $\mathbf{Y}_i$ é a imagem cifrada correspondente. Aqui, denota-se por $v \in \mathbb{Z}_+^*$ a quantidade de valores possíveis para cada pixel, por $\mathbb{M}$ o conjunto $\{0, 1, 2, \ldots, M-1\}$ e por $\mathbb{N}$ o conjunto $\{0, 1, \ldots, N-1\}$. Nas imagens trabalhadas neste relatório, temos $v = 256$. O passo a passo de algoritmo é o seguinte:

- **Passo 1:** Comparar os valores de pixels dentro das p imagens cifradas $\mathbf{Y}_i$ e encontrar $v \cdot p$ conjuntos de posições $\{\Lambda_0(0), \ldots, \Lambda_0(v-1), \ldots, \Lambda_{p-1}(0), \ldots, \Lambda_{p-1}(v-1)\}$, em que $\Lambda_m(l) \subseteq \mathbb{M} \times \mathbb{N}$ é o conjunto que contém as posições de todos os pixels em $\mathbf{Y}_m$ que possuem valor $l \in \{0, 1, \ldots, v-1\}$. Ou seja, $\Lambda_m(l) = \{(i, j) \in \mathbb{M} \times \mathbb{N} \mid \mathbf{Y}_m(i, j) = l\}$.
- **Passo 2:** Encontrar uma matriz de permutação de valores múltiplos, $\widetilde{\mathbf{W}}^*$, tal que $\widetilde{\mathbf{W}}^*(i, j) = \bigcap_{m=0}^{p-1} \Lambda_m(\mathbf{X}_m(i, j))$ para todo $(i, j) \in \mathbb{M} \times \mathbb{N}$.
- **Passo 3:** Determinar uma matriz de permutação $\widetilde{\mathbf{W}}$ de valores únicos, tal que $\widetilde{\mathbf{W}}(i, j) \in \widetilde{\mathbf{W}}^*(i, j)$ para todo $(i, j) \in \mathbb{M} \times \mathbb{N}$. Além disso, $\widetilde{\mathbf{W}}$ deve ter valores únicos, isto é, $\widetilde{\mathbf{W}}(i, j) \neq \widetilde{\mathbf{W}}(i', j')$ para todo $(i, j), (i', j') \in \mathbb{M} \times \mathbb{N}$, $(i, j) \neq (i', j')$.
- **Passo 4:** Retornar $\widetilde{\mathbf{W}}$ como estimativa para a matriz de permutação $\mathbf{W}$.

2.3 Ataque de texto claro escolhido

O *chosen-plaintext attack*, ou ataque de texto claro escolhido, é um tipo de ataque criptoanalítico em que o atacante tem acesso a pares de texto claro e texto cifrado, e pode escolher o texto claro. No caso do HCIE, o atacante tem acesso a pares de imagens, uma original (de sua escolha) e sua cifra correspondente, e aos parâmetros públicos do algoritmo.

3 Resultados e discussão



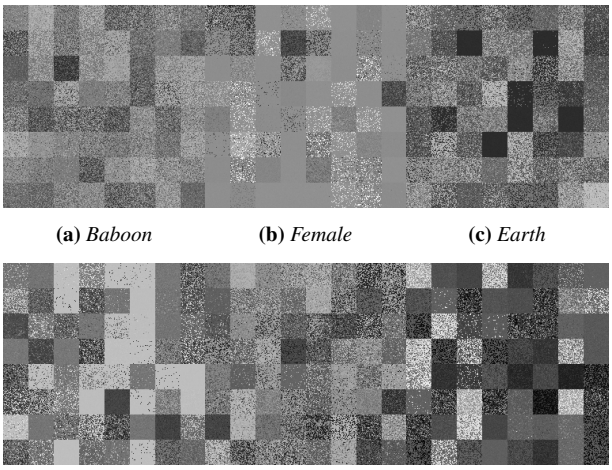
(a) *Baboon* **(b)** *Female* **(c)** *Earth*

(d) *House* **(e)** *Peppers* **(f)** *Splash*

Figura 3: *Imagens originais*

As imagens foram selecionadas para ter a máxima semelhança com o artigo original (3). Todas foram recolhidas da base de dados da USC-SIPI (7) e redimensionadas para 256×256 pixels. Algumas imagens não foram reproduzidas por não serem encontradas na base de dados, devido a motivos éticos, como a imagem *Lena*, ou por outros motivos.

Além disso, é importante notar que o autor forneceu os parâmetros públicos $\alpha = 4, \beta = 2, \gamma = 1$, porém não forneceu os parâmetros privados x_0 e μ. Ademais, a precisão da aritmética em ponto flutuante não foi descrita. Sendo assim, é extremamente difícil reproduzir os resultados com exatidão. Levando em conta tal consideração, os parâmetros privados foram escolhidos de forma a obter resultados satisfatórios, com $x_0 = 0.1, \mu = 3.9999$. Todos os algoritmos foram implementados como uma biblioteca na linguagem de programação *Rust* (1), e o mapa logístico foi implementado com precisão de 64 bits, isto é, usando o *type* `f64`, que implementa o padrão IEEE 754-2008 (2).



(a) *Baboon* **(b)** *Female* **(c)** *Earth*

(d) *House* **(e)** *Peppers* **(f)** *Splash*

Figura 4: *Imagens criptografadas com* $S_M = S_N = 32$

4 Conclusão

Com base na análise do artigo, que se concentrou na avaliação do desempenho de um algoritmo de criptografia de permutação única, HCIE, em face de ataques de criptoanálise de texto cifrado e ataques de texto claro conhecido/escolhido, destacam-se conclusões importantes. Verificou-se uma sobreestimação da capacidade do HCIE contra ataques de criptoanálise de texto cifrado, e algoritmos de criptografia de imagem de permutação única hierárquicos, como o HCIE, foram identificados como menos seguros em comparação com suas contrapartes de permutação única sem estruturas de criptografia hierárquicas. A análise efetivamente ressalta a importância de maximizar o tamanho do domínio real de permutação em algoritmos dessa natureza para otimizar o desempenho. Além disso, enfatiza-se que, dado que a operação de permutação isoladamente não oferece um nível suficientemente elevado de segurança, é recomendável sua combinação com outras funções de substituição de valores para fortalecer a robustez do sistema de criptografia. Essas conclusões fornecem uma visão mais crítica quanto o desenvolvimento e avaliação de algoritmos de criptografia baseado em mapas caóticos, destacando a necessidade de considerações abrangentes em relação à segurança e desempenho.

Referências

1 ALMEIDA, Vinícius Freitas de. **HCIE-rs: Hierarchical Chaotic Image Encryption in Rust**. [S.l.]: GitHub, 2023. https://github.com/vini-fda/hcie_rs.

2 ELECTRICAL, Institute of; ENGINEERS, Electronics. **IEEE Standard for Floating-Point Arithmetic**. [S.l.]: IEEE, 2008. DOI: 10.1109/IEEESTD.2008.4610935.

3 LI, Chengqing. Cracking a hierarchical chaotic image encryption algorithm based on permutation. **Signal Processing**, Elsevier BV, v. 118, p. 203–210, jan. 2016. ISSN 0165-1684. DOI: 10.1016/j.sigpro.2015.07.008. Disponível em: <http://dx.doi.org/10.1016/j.sigpro.2015.07.008>.

4 LI, Chengqing; LO, Kwok-Tung. Optimal quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. **Signal Processing**, Elsevier BV, v. 91, n. 4, p. 949–954, abr. 2011. ISSN 0165-1684. DOI: 10.1016/j.sigpro.2010.09.014. Disponível em: <http://dx.doi.org/10.1016/j.sigpro.2010.09.014>.

5 LI, Shujun et al. A general quantitative cryptanalysis of permutation-only multimedia ciphers against plaintext attacks. **Signal Processing: Image Communication**, v. 23, n. 3, p. 212–223, 2008. ISSN 0923-5965. DOI: https://doi.org/10.1016/j.image.2008.01.003. Disponível em: <https://www.sciencedirect.com/science/article/pii/S0923596508000064>.

6 PATTERSON, David A; HENNESSY, John L. **Computer organization and Design**. [S.l.]: Morgan Kaufmann, 1994.

7 USC-SIPI Image Database. [S.l.]: University of Southern California, Signal e Image Processing Institute. Disponível em: <https://sipi.usc.edu/database/>.

8 YEN, J-C; GUO, J-I. Efficient hierarchical chaotic image encryption algorithm and its VLSI realisation. **IEE Proceedings-vision, image and signal processing**, IET, v. 147, n. 2, p. 167–175, 2000.