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

DAVI CARVALHO MORENO DE ALMEIDA

TÉCNICAS PARA GERAÇÃO DE NÚMEROS PSEUDOALEATÓRIOS E AUTENTICAÇÃO EM CAMADA FÍSICA UTILIZANDO SEQUÊNCIAS CAÓTICAS

Recife

2023

DAVI CARVALHO MORENO DE ALMEIDA

TÉCNICAS PARA GERAÇÃO DE NÚMEROS PSEUDOALEATÓRIOS E AUTENTICAÇÃO EM CAMADA FÍSICA UTILIZANDO SEQUÊNCIAS CAÓTICAS

Dissertação submetida ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco como requisito parcial para obtenção do título de Mestre em Engenharia Elétrica.

Área de Concentração: Comunicações.

Orientador: Prof. Dr. Daniel Pedro Bezerra Chaves

Coorientador: Prof. Dr. Cecilio José Lins Pimentel

Recife

2023

Catalogação na fonte
Bibliotecário Gabriel Luz, CRB-4 / 2222

A447t Almeida, Davi Carvalho Moreno de.
Técnicas para geração de números pseudoaleatórios e autenticação em camada física utilizando sequencias caóticas / Davi Carvalho Moreno de Almeida, 2023.
69 f.: il.

Orientador: Prof. Dr. Daniel Pedro Bezerra Chaves.
Coorientador: Prof. Dr. Cecilio José Lins Pimentel.
Dissertação (Mestrado) – Universidade Federal de Pernambuco. CTG. Programa de Pós-Graduação em Engenharia Elétrica. Recife, 2023.
Inclui referências.

1. Engenharia elétrica. 2. Geradores de números pseudoaleatórios. 3. Autenticação em camada física. 4. Segurança da informação. 5. Mapa de Arnold. 6. q-Análogos. 7. Sequências caóticas. I. Chaves, Daniel Pedro Bezerra (Orientador). II. Pimentel, Cecilio José Lins (Coorientador). III. Título.

UFPE

621.3 CDD (22. ed.) BCTG / 2023 - 182

DAVI CARVALHO MORENO DE ALMEIDA

TÉCNICAS PARA GERAÇÃO DE NÚMEROS PSEUDOALEATÓRIOS E AUTENTICAÇÃO EM CAMADA FÍSICA UTILIZANDO SEQUÊNCIAS CAÓTICAS

Dissertação apresentada ao Programa de Pós-Graduação em Engenharia Elétrica da Universidade Federal de Pernambuco, como requisito parcial para a obtenção do título de Mestre em Engenharia Elétrica, na área de concentração em Comunicações.

Aprovada em: 28/07/2023

BANCA EXAMINADORA

_____

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

_____

Prof. Dr. Ricardo Menezes Campello de Souza
(Examinador Interno)
Universidade Federal de Pernambuco

_____

Prof. Dr. João Victor de Carvalho Evangelista
(Examinador Externo)
ERICSSON CANADÁ

Para Ana e Jonas.

AGRADECIMENTOS

Primeiramente, agradeço aos meus pais, Ana e Jonas, cuja dedicação à minha educação e constante apoio formaram a base essencial para esta conquista. Um agradecimento especial se estende aos meus orientadores, Prof. Dr. Daniel Chaves e Prof. Dr. Cecilio Pimentel, cujo comprometimento desde o início da pesquisa se traduziu em grandes oportunidades e colaborações em diversos projetos. Suas discussões construtivas e contribuições na formulação de ideias e na escrita de artigos foram de extrema importância.

Também agradeço ao Dr. Carlos pelas enriquecedoras trocas de ideias e discussões que desempenharam um papel fundamental no desenvolvimento deste trabalho. Meus amigos e familiares merecem meu reconhecimento pelo suporte diário, que desempenhou um papel vital na manutenção do meu bem-estar físico e mental ao longo dessa jornada.

À Universidade Federal de Pernambuco, estendo meu agradecimento pelo ensino de qualidade que recebi durante o período de pesquisa, que contribuiu significativamente para o desenvolvimento deste trabalho. Por fim, expresso minha gratidão ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq) pelo apoio financeiro crucial concedido durante todo o processo de desenvolvimento desta pesquisa. Suas contribuições foram essenciais para a realização deste trabalho e serão lembradas com profunda gratidão.

RESUMO

Com o avanço das tecnologias de comunicação e o aumento do volume de dados transmitidos, a segurança da informação é uma preocupação fundamental em diversos setores. Esta dissertação apresenta contribuições em duas áreas de segurança da informação: geradores de números pseudoaleatórios (PRNGs, *Pseudorandom Number Generators*) e autenticação em camada física (PLA, *Physical Layer Authentication*). Inicialmente, duas novas técnicas de PRNGs são propostas, a primeira é baseada no mapa de Arnold aplicado a anéis de inteiros, enquanto a segunda é um PRNG baseado em q-Análogos sobre corpos finitos. Ambos são comparados com algoritmos existentes na literatura, utilizando métricas de análise estatística e de implementação em hardware (com FPGAs), e apresentam ganhos acima de 270% na taxa de geração de bits. O PRNG baseado no mapa de Arnold apresenta valores de ocupação de hardware comparáveis aos menores valores encontrados na literatura, enquanto o baseado em q-análogos apresenta uma redução no número de *look up tables* e registradores da ordem de 60% e 62%, respectivamente. Além disso, é proposto um novo algoritmo de geração de códigos de autenticação, chamados de *tags*, para sistemas de PLA, baseado em sequências caóticas discretizadas, e é realizada uma comparação com métodos existentes na literatura, utilizando métricas baseadas em teoria da informação. O algoritmo proposto apresenta uma segurança incondicional, independente do ruído, positiva, o que não é válido para o algoritmo clássico de geração da *tag* encontrado na literatura. A dissertação também analisa a informação incondicional que um usuário malicioso tem da chave secreta utilizada no sistema de PLA ao interceptar múltiplos pares legítimos de mensagem e *tag*, considerando o algoritmo proposto para geração da *tag*, e duas métricas associadas a essa medida de informação são propostas.

Palavras-chave: geradores de números pseudoaleatórios; autenticação em camada física; segurança da informação; mapa de arnold; q-análogos; sequências caóticas.

ABSTRACT

With the advancement of communication technologies and the increase in transmitted data volume, information security is a fundamental concern in various sectors. This dissertation presents contributions in two areas of information security: Pseudorandom Number Generators (PRNGs) and Physical Layer Authentication (PLA). Initially, two new PRNG techniques are proposed. The first is based on the Arnold map applied to rings of integers, while the second is a PRNG based on q-Analogs over finite fields. Both techniques are compared with existing algorithms in the literature, using metrics of statistical analysis and hardware implementation (on FPGAs), and they exhibit gains above 270% in the bit generation rate. The PRNG based on the Arnold map shows hardware occupancy values comparable to the lowest values found in the literature, while the q-Analog-based PRNG reduces the number of look up tables and registers by approximately 60% and 62%, respectively. Additionally, a new authentication code, called "tag," generation algorithm is proposed for PLA systems, based on discretized chaotic sequences. A comparison is made with existing methods in the literature, using metrics based on information theory. The proposed algorithm provides unconditional security, independent of noise, which is not valid for the classical algorithm for tag generation found in the literature. The dissertation also analyzes the unconditional information that a malicious user has about the secret key used in the PLA system when intercepting multiple legitimate pairs of message and tag, considering the proposed algorithm for tag generation, and two metrics associated with this measure of information are proposed.

Keywords: pseudorandom number generators; physical layer authentication; information security; arnold map; q-analogs; chaotic sequences.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE ABREVIATURAS E SIGLAS

DFT	Transformada Discreta de Fourier
DSP	*Digital Signal Processor*
FF	*Flip-Flops*
FPGA	*Field-Programmable Gate Arrays*
LFSR	Registradores de Deslocamento com Retroalimentação Linear
LSB	Bit Menos Significativo
LUT	*Look Up Table*
MMM	Multiplicação Modular de Montgomery
MSB	Bit Mais Significativo
NIST	*National Institute of Standards and Technology*
PLA	Autenticação em Camada Física
PRNG	Gerador de Números Pseudoaleatórios
RTL	*Register Transfer Level*
TNR	Relação Sinal-Ruído da *Tag*

SUMÁRIO

1 INTRODUÇÃO

Com o avanço constante das tecnologias de comunicação e o crescente volume de dados transmitidos, a segurança da informação tem se tornado uma preocupação fundamental em diversos setores. Empresas financeiras, provedores de serviços de telecomunicações, setor governamental e instituições de saúde são apenas alguns exemplos dos segmentos que dependem da garantia de confidencialidade e integridade das informações. Considerando a importância crítica da segurança da informação nesses setores, dois relevantes campos de pesquisa na área envolvem os algoritmos Geradores de Números Pseudoaleatórios (PRNGs, *Pseudorandom Number Generators*) e a Autenticação em Camada Física (PLA, *Physical Layer Authentication*). Nesse contexto, esta dissertação aborda os desafios na elaboração de PRNGs confiáveis e na geração da *tag* para sistemas de PLA, propondo dois PRNGs e um novo método de geração da *tag* para contribuir nesses campos específicos.

1.1 MOTIVAÇÃO

A geração de números pseudoaleatórios desempenha um papel fundamental em diversas aplicações de ciência da computação e engenharia, sendo amplamente utilizada em diversos protocolos criptográficos, sistemas de criptografia simétrica e assimétrica, além de aplicações estatísticas e científicas, sendo crucial para a segurança dos sistemas. A qualidade dos números gerados influencia diretamente a segurança dos sistemas, pois qualquer padrão ou viés nas sequências pseudoaleatórias pode ser explorado por atacantes maliciosos. Portanto, é fundamental desenvolver algoritmos eficientes e robustos capazes de produzir sequências pseudoaleatórias de alta qualidade. Como estes valores precisam ser de alguma forma imprevisíveis, os PRNGs apresentam uma maneira conveniente de gerá-los. Desta forma, os PRNGs são utilizados, por exemplo, para gerar chaves secretas, vetores de inicialização, e *nonces* (número usado uma única vez) requeridos por muitos algoritmos criptográficos. No geral, o uso de PRNGs é essencial em aplicações relacionadas a segurança, ajudando a manter a confidencialidade, integridade e disponibilidade de informações confidenciais.

Estruturas algébricas discretas, como os anéis de inteiros e os corpos finitos, são estruturas definidas sobre conjuntos discretos que têm ampla aplicação em diversas áreas da engenharia de comunicações, como no desenvolvimento de PRNGs. Essas estruturas oferecem vantagens na implementação de hardware, uma vez que todas as operações são realizadas em aritmética de inteiros, o que se traduz em benefícios quando comparadas às operações no campo real. Adicionalmente, mapas iterativos em estruturas discretas têm sido explorados na literatura para aplicações de criptografia com exigências de hardware competitivas ou reduzidas (SOUZA; CHAVES; PIMENTEL, 2021). Os dois PRNGs apresentados nesta dissertação se utilizam de estruturas algébricas discretas para gerar sequências pseudoaleatórias competitivas com os

algoritmos existentes na literatura.

Também no contexto da segurança, a autenticação é um problema de extrema importância. Com o crescente uso de meios digitais para comunicação, assegurar a integridade e a procedência das mensagens transmitidas é essencial. A autenticação desempenha um papel crucial nesse sentido, permitindo estabelecer a autoria e a origem de uma mensagem. Imagine um cenário em que dois usuários legítimos compartilham uma chave secreta e se comunicam entre si, enquanto um adversário ativo tenta interromper a conversa se passando por um usuário legítimo. O objetivo do sistema de autenticação é garantir, com base apenas no conhecimento compartilhado da chave secreta, que um usuário seja capaz de identificar a verdadeira origem da mensagem, mantendo assim a confiança e a segurança na comunicação. Usualmente, a autenticação é feita nas camadas superiores de uma rede (TANENBAUM; WETHERALL, 2011), em que os usuários legítimos geram *tags* de autenticação, que são sequências binárias geradas com base na chave secreta e em uma função criptográfica. Grande parte dessas funções criptográficas são baseadas na dificuldade de solução de algum problema matemático, como o logaritmo discreto ou a fatoração de inteiros, por exemplo, o que garante a segurança computacional do sistema. No entanto, o desenvolvimento de novos algoritmos e de hardwares mais poderosos pode comprometer essa segurança. Para adicionar uma camada extra de segurança ao sistema, o uso de algoritmos de PLA é uma alternativa que explora as características estocásticas do canal de comunicação (MASSEY, 1998). Existem dois tipos de abordagens para PLA: ativa e passiva. Na PLA ativa, as mensagens transmitidas são incorporadas a uma *tag* legítima, geralmente gerada a partir de uma chave secreta (XIE; CHEN; MING, 2021; PERAZZONE et al., 2018; YU; BARAS; SADLER, 2008; YU; VERMA; SADLER, 2015). Já na PLA passiva, a singularidade da resposta ao impulso do canal é explorada em canais de desvanecimento de caminhos múltiplos entre os usuários legítimos (LIU; WANG, 2016; ZHANG et al., 2020; SENIGAGLIESI; BALDI; GAMBI, 2021). No que diz respeito à PLA, o foco deste trabalho está nos sistemas de PLA ativos, mais especificamente nos métodos de geração da *tag* utilizados. A maioria desses sistemas utiliza *tags* geradas a partir de funções de *hash* que dependem da chave secreta e da mensagem transmitida (XIE; CHEN; MING, 2021; YU; BARAS; SADLER, 2008; PERAZZONE et al., 2018; YU; VERMA; SADLER, 2015; YU; SADLER, 2011; QAISI; ALTHUNIBAT; QARAQE, 2022; XIE et al., 2022; MAENG et al., 2022; XIE et al., 2023; GU et al., 2020), garantindo uma segurança condicionada ao ruído associado ao canal de comunicação. Nesta dissertação, é desenvolvido um algoritmo de geração da *tag* que proporciona uma segurança incondicional, independente do ruído, que é obtida com o uso de sequências caóticas discretizadas.

Os mapas caóticos são sistemas dinâmicos que exibem um comportamento complexo e imprevisível. Eles são caracterizados por propriedades como sensibilidade às condições iniciais, espectro de banda larga, comportamento recursivo e não periódico (STROGATZ, 2015). Essas propriedades conferem aos mapas caóticos um comportamento semelhante ao ruído, mesmo sendo determinísticos. Essa natureza caótica pode ser aproveitada de forma vantajosa no projeto de sistemas criptográficos, pois proporcionam um método simples para gerar sequências com

estatísticas semelhantes ao ruído, ao mesmo tempo em que exigem baixos requisitos de hardware. Esses atributos fazem com que os mapas caóticos sejam utilizados tanto em algoritmos de PRNG (PREISHUBER et al., 2018; LAN et al., 2019; ZHU et al., 2019; ALAWIDA et al., 2019; MACHICAO et al., 2021; BAO et al., 2021) como em sistemas de PLA (EVANGELISTA, 2016). Além de utilizar sequências caóticas no algoritmo desenvolvido para geração da *tag* para sistemas de PLA, um dos PRNGs apresentados nesta dissertação também utiliza mapas caóticos.

1.2 OBJETIVOS E CONTRIBUIÇÕES

Os objetivos deste trabalho são:

- Investigar técnicas de geração de números pseudoaleatórios sobre estruturas algébricas discretas.
- Explorar o uso de mapas caóticos na geração das *tags* em sistemas de PLA.
- Investigar a informação que um usuário malicioso possui da chave secreta utilizada no sistema de PLA quando ele intercepta várias comunicações sem ruído.

As contribuições deste trabalho são:

- Propor um novo PRNG baseado no mapa de Arnold sobre anéis de inteiros do tipo $\mathbb{Z}_{2^m}$.
- Propor um novo PRNG baseado em uma definição de q-análogos sobre corpos finitos.
- Desenvolver circuitos síncronos digitais que implementam os PRNGs propostos.
- Mostrar a qualidade das sequências pseudoaleatórias geradas pelos PRNGs propostos, empregando métricas de análise estatística.
- Mostrar as vantagens, em termos de implementação em hardware, que os PRNGs propostos possuem em relação a outros PRNGs da literatura.
- Propor um novo método de geração da *tag* para sistemas de PLA, baseado em sequências caóticas discretizadas.
- Mostrar que o método proposto garante segurança incondicional, independente do ruído, algo que não é presente no algoritmo clássico de geração da *tag* presente na literatura.
- Comparar, através de métricas baseadas em teoria da informação, o sistema proposto de geração da *tag* com dois algoritmos presentes da literatura.
- Propor duas métricas para analisar a informação que um usuário malicioso tem da chave ao interceptar múltiplos pares legítimos de mensagem e *tag*, considerando que o algoritmo proposto para geração da *tag* está sendo utilizado. As métricas são analisadas para dois tipos de mapeamentos distintos associados ao algoritmo proposto.

1.3 ORGANIZAÇÃO DA DISSERTAÇÃO

No Capítulo 1, é realizada uma discussão preliminar sobre a motivação que envolve o problema da geração de números pseudoaleatórios e PLA, acompanhada de uma breve revisão bibliográfica. Também são listados os principais objetivos e contribuições do trabalho.

No Capítulo 2, discutem-se os principais conceitos e métricas que envolvem os algoritmos estudados nos capítulos posteriores, que abordam a geração de números pseudoaleatórios e a geração da *tag* para sistemas de PLA.

No Capítulo 3, são apresentados dois PRNGs, e são utilizadas métricas de análise estatística e de implementação em hardware para verificar a qualidade dos mesmos.

No Capítulo 4, é desenvolvido o algoritmo proposto para a geração da *tag* em sistemas de PLA, utilizando sequências caóticas discretizadas. Além disso, é feita uma comparação desse algoritmo com outros algoritmos de geração da *tag* presentes na literatura. Também é exibida uma análise, em termos da informação sobre a chave secreta, de como o algoritmo proposto se comporta quando um usuário malicioso tem acesso a múltiplas observações legítimas.

No Capítulo 5, são resumidas todas as conclusões derivadas ao longo da dissertação e as diretrizes para trabalhos futuros são apresentadas.

1.4 PUBLICAÇÕES

O conteúdo trabalhado nesta dissertação contribuiu com algumas publicações em periódicos e simpósios, listadas a seguir.

- Publicações em Periódicos:
  - J. V. C. Evangelista, D. Moreno, D. P. B. Chaves and C. Pimentel, “Tag Generation using Chaotic Sequences for Physical-Layer Authentication,” in *IEEE Access*, 2023.
  - C. E. C. Souza, D. Moreno, R. B. D. Figueiredo, D. P. B. Chaves and C. Pimentel, “q-Analogs Over Finite Fields: Definition, Algebraic Properties, and Application in Pseudo-Random Number Generators,” in *IEEE Transactions on Circuits and Systems II: Express Briefs*, 2023.
  - C. E. C. Souza, D. Moreno, D. P. B. Chaves and C. Pimentel, “Pseudo-Chaotic Sequences Generated by the Discrete Arnold’s Map Over $\mathbb{Z}_{2^m}$: Period Analysis and FPGA Implementation,” in *IEEE Transactions on Instrumentation and Measurement*, vol. 71, pp. 1-10, 2022.
- Publicações em Simpósios:

- D. Moreno, D. P. B. Chaves, C. Pimentel, “Geração de *Tags* para Autenticação em Camada Física Utilizando Sequências Caóticas Discretizadas,” in XLI Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2023.
- C. E. C. Souza, D. Moreno, D. P. B. Chaves, C. Pimentel, “Geradores de Números Pseudoaleatórios de Alta Taxa Baseados no Mapa de Arnold Discreto,” in XL Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2022.
- D. C. M. de Almeida, C. E. C. Souza, D. P. B. Chaves, C. Pimentel, “Implementação em FPGA de Geradores de Números Pseudoaleatórios sobre Anéis de Inteiros,” in XXXIX Simpósio Brasileiro de Telecomunicações e Processamento de Sinais, 2021.

2 CONCEITOS FUNDAMENTAIS

O propósito deste capítulo é fornecer ao leitor uma compreensão dos conceitos básicos que envolvem PRNGs e PLA, além de apresentar as métricas utilizadas na análise dos algoritmos que serão descritos nos capítulos posteriores. Também introduz-se o conceito de mapas caóticos, que está presente em um dos PRNGs e no método de geração da *tag* para sistemas de PLA, ambos propostos nesta dissertação.

A Seção 2.1 apresenta conhecimentos básicos de PRNGs, juntamente com as métricas que serão utilizadas na análise dos mesmos. Na Seção 2.2, uma introdução aos sistemas de PLA é realizada, e as métricas para analisar os métodos de geração da *tag* também são descritas. Por fim, as Seções 2.3 e 2.4 exibem os fundamentos básicos de mapas caóticos e uma introdução a q-análogos e corpos finitos, respectivamente.

2.1 GERADORES DE NÚMEROS PSEUDOALEATÓRIOS

Os PRNGs são algoritmos determinísticos que geram sequências de números que aparentam serem aleatórios, pois possuem propriedades estatísticas semelhantes a sequências verdadeiramente aleatórias. Apesar deste comportamento, as sequências obtidas de um PRNG são completamente determinadas por uma semente escolhida para ser utilizada com o algoritmo.

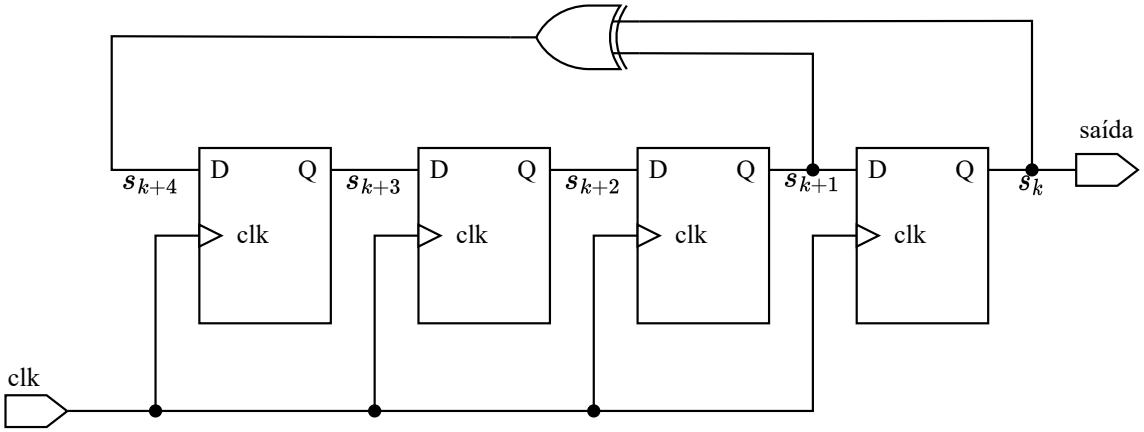
Um tipo de PRNG amplamente estudado em criptografia são os chamados de Registradores de Deslocamento com Retroalimentação Linear (LFSRs, *Linear Feedback Shift Registers*), que recebem este nome por serem registradores de deslocamento que possuem uma realimentação por meio de uma função linear. Os LFSRs são conhecidos pela sua simplicidade de implementação e possibilidade de gerar sequências de período longo. Apesar destas características, existem ataques de criptoanálise eficientes (PAAR; PELZL, 2009, Seção 2.3.2) que permitem que um atacante descubra a semente associada a um LFSR a partir de uma sequência suficientemente grande gerada pelo mesmo, e, portanto, consiga reproduzir toda a sequência pseudoaleatória. Por esse motivo, em aplicações em que a semente do PRNG é um parâmetro secreto, os LFSRs podem ser usados como blocos de construção dos PRNGs, mas nunca sozinhos.

Para ilustrar o comportamento e as características de um PRNG, considere o LFSR da Fig. 1. O LFSR gera um novo bit na saída a cada ciclo de *clock*, e pode ser representado pela seguinte equação de recorrência:

$$s_{k+4} = s_{k+1} \oplus s_k, \quad k \geq 0, \tag{2.1}$$

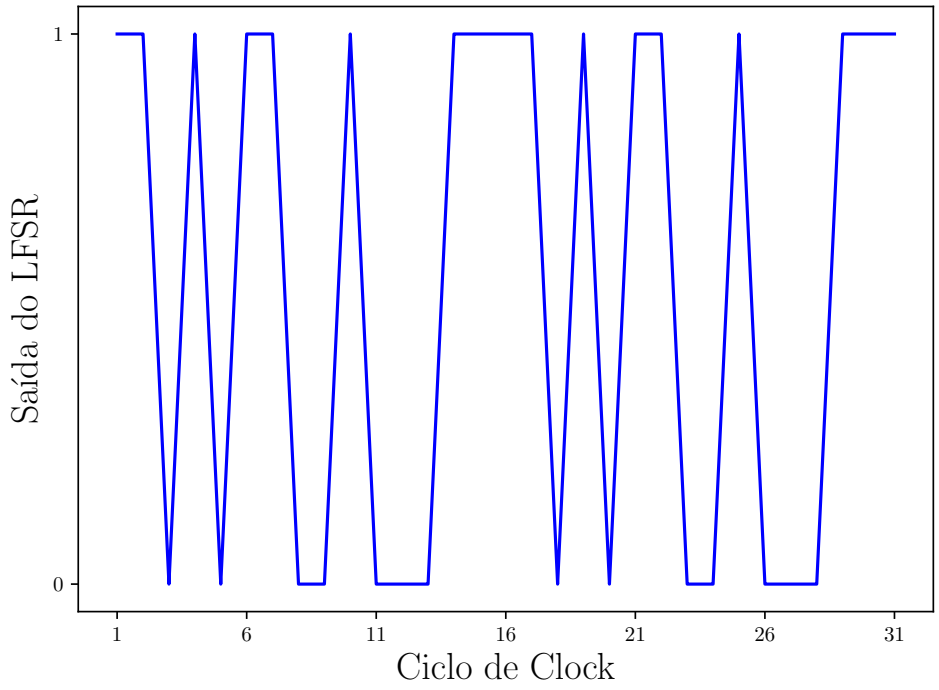
em que $\oplus$ representa a operação de adição módulo 2. Evidentemente, para gerar uma sequência a partir do LFSR analisado, é necessário saber os estados iniciais de cada um dos quatro registradores, ou seja, os valores de s_0, s_1, s_2 e s_3. Como a estrutura do LFSR está definida,

Figura 1 – Exemplo de um LFSR.



Fonte: O autor (2023)

Figura 2 – Exemplo de sequência de saída para o LFSR analisado.



Fonte: O autor (2023)

estes estados iniciais podem ser interpretados como a semente do algoritmo (a própria estrutura do LFSR também poderia fazer parte da semente). Ao escolher os valores $s_0 = 1$, $s_1 = 1$, $s_2 = 0$ e $s_3 = 1$ como condições iniciais, a sequência de saída do LFSR é como a mostrada na Fig. 2. Imediatamente pode-se perceber que a saída do LFSR se repete após o 15º pulso de *clock*, o que caracteriza o algoritmo como tendo uma sequência de saída de período 15. Este é um período pequeno quando comparado com um caso prático, no qual deseja-se um PRNG com um período muito maior que este. A função `random` presente na biblioteca `random` da linguagem de programação *Python* fornece um PRNG com período $2^{19937} - 1$, por exemplo.

A análise do LFSR exemplificado permite que sejam identificados alguns parâmetros que devem ser levados em consideração no desenvolvimento de um PRNG. A taxa de bits de saída por pulso de *clock* e o período da sequência de saída são dois destes fatores. A complexidade da implementação do algoritmo, que neste caso são quatro registradores e uma porta XOR (vide Fig. 1), também deve ser considerada.

As métricas de performance que são utilizadas para analisar os PRNGs propostos no Capítulo 3 são apresentadas nas próximas duas subseções. Na Subseção 2.1.1, as métricas estatísticas que um PRNG deve obedecer são discutidas. As métricas associadas à implementação em hardware são abordadas na Subseção 2.1.2.

2.1.1 Métricas de análise estatística

Os testes estatísticos são essenciais para garantir a qualidade das sequencias pseudoaleatórias geradas pelos PRNGs. Uma bateria de testes amplamente utilizada é o conjunto de testes estatísticos do NIST (*National Institute of Standards and Technology*). A versão aqui utilizada desta bateria de testes é a SP800-22 (BASSHAM III et al., 2010), que agrupa quinze diferentes testes para analisar a performance estatística de uma sequência pseudoaleatória. Estes testes são:

1. Teste de Frequência (Monobit): o foco deste teste é a proporção de zeros e uns na sequência fornecida. Este teste avalia o quanto a fração de uns se aproxima de $1/2$, ou seja, o número de zeros e uns devem ser aproximadamente os mesmos.

2. Teste de Frequência em Blocos: este teste tem o objetivo de testar a proporção de zeros e uns em blocos de M bits na sequência fornecida. Ele testa se o número de uns dentro do bloco é próximo de $M/2$.

3. Teste de Corridas: o foco deste teste é o número total de corridas na sequência, em que uma corrida é uma sequência ininterrupta de bits idênticos. O objetivo do teste de corridas é determinar se o número de corridas de uns e zeros de vários comprimentos é o esperado para uma sequência aleatória.

4. Teste de Corridas mais Longas de Uns em um Bloco: o foco do teste é a corrida de uns mais longa dentro de blocos de M bits. O objetivo deste teste é determinar se o comprimento da corrida mais longa de uns dentro da sequência testada é consistente com o comprimento da corrida mais longa de uns que seria esperado em uma sequência aleatória.

5. Teste de Posto de Matriz Binária: o foco do teste é o posto de submatrizes disjuntas de toda a sequência. O objetivo deste teste é verificar a dependência linear entre subsequências de comprimento fixo da sequência original.

6. Teste de Transformada Discreta de Fourier (DFT, *Discrete Fourier Transform*): o foco deste teste são os picos na Transformada Discreta de Fourier da sequência. O objetivo deste

teste é detectar características periódicas na sequência testada que indicaria um desvio da suposição de aleatoriedade.

7. Teste de Correspondência de Modelo Não-Sobreposto: o foco deste teste é o número de ocorrências de sequências de bits pré-definidas. O objetivo deste teste é detectar geradores que produzem muitas ocorrências de um determinado padrão de bits não periódico. As sequências encontradas aqui pertencem a janelas de bits não sobrepostas.

8. Teste de Correspondência de Modelo Sobreposto: este teste possui mesmo foco e objetivo do teste anterior, mas as sequências encontradas aqui podem pertencer a janelas de bits que se sobrepõem.

9. Teste Estatístico Universal de Maurer: o foco deste teste é o número de bits entre padrões correspondentes (uma medida relacionada ao comprimento de uma sequência compactada). O objetivo do teste é detectar se a sequência pode ou não ser significativamente comprimida sem perda de informação. Uma sequência significativamente compressível é considerada não aleatória.

10. Teste de Complexidade Linear: o foco deste teste é o comprimento de um LFSR que poderia gerar a sequência. O objetivo deste teste é determinar se a sequência é ou não complexa o suficiente para ser considerada aleatória. Sequências aleatórias são caracterizadas por LFSRs mais longos. Um LFSR muito curto implica não aleatoriedade.

11. Teste Serial: o foco deste teste é a frequência de todos os possíveis padrões de m-bits sobrepostos em toda a sequência. O objetivo deste teste é determinar se o número de ocorrências dos 2^m padrões de m-bits é aproximadamente o mesmo que seria esperado para uma sequência aleatória. As sequências aleatórias têm uniformidade, ou seja, todo padrão de m-bits tem a mesma chance de ocorrer.

12. Teste de Entropia Aproximada: assim como o teste anterior, o foco deste teste é a frequência de todos os possíveis padrões de m-bits sobrepostos em toda a sequência. O objetivo do teste é comparar a frequência de blocos sobrepostos de dois comprimentos consecutivos/adjacentes (m e $m + 1$) com o resultado esperado para uma sequência aleatória.

13. Teste de Soma Cumulativa: o foco deste teste é a excursão máxima (a partir de zero) do passeio aleatório definido pela soma cumulativa dos dígitos ajustados (-1, +1) na sequência. O objetivo do teste é determinar se a soma cumulativa das sequências parciais que ocorrem na sequência testada é muito grande ou muito pequena em relação ao comportamento esperado dessa soma cumulativa para sequências aleatórias, que seria algo próximo de zero.

14. Teste de Excursões Aleatórias: o foco deste teste é o número de ciclos com exatamente K visitas em um passeio aleatório de soma cumulativa. O passeio aleatório de soma

cumulativa é derivado das somas parciais depois que a sequência (0,1) é transformada na sequência (-1, +1) apropriada. Um ciclo de um passeio aleatório consiste em uma sequência de passos de comprimento unitário realizados aleatoriamente que começam e retornam à origem. O objetivo deste teste é determinar se o número de visitas a um estado específico dentro de um ciclo desvia do que se esperaria para uma sequência aleatória.

15. Teste de Variações em Excursões Aleatórias: o foco deste teste é o número total de vezes que um determinado estado é visitado em um passeio aleatório de soma cumulativa. O objetivo deste teste é detectar desvios do número esperado de visitas a vários estados no passeio aleatório.

Esta bateria de testes é amplamente reconhecida na indústria, sendo bem documentada e servindo como um indicador da qualidade da sequência gerada por um PRNG. De uma maneira geral, os testes estatísticos do NIST fornecem medidas rigorosas e efetivas de medir a performance estatística de um PRNG, ajudando a garantir se os mesmos atendem requisitos de segurança e aleatoriedade desejados.

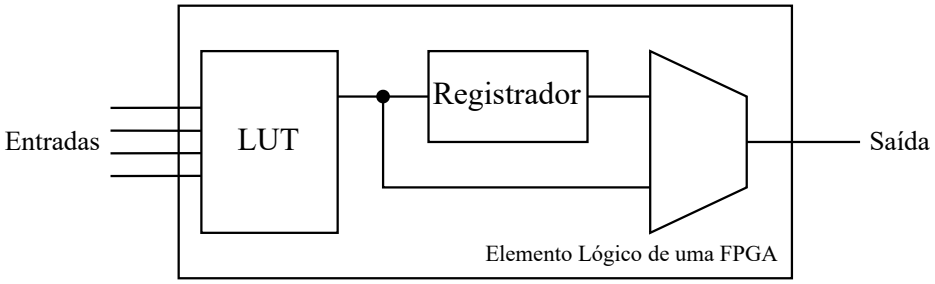
2.1.2 Métricas de implementação em hardware

Tendo em vista que um PRNG passa pelas métricas estatísticas desejadas, é preciso mensurar a eficiência da implementação do mesmo. A princípio, pode-se realizar a implementação em termos de um circuito digital, visualizada por meio de um *Register Transfer Level* (RTL), no qual descreve-se o comportamento do circuito em termos do fluxo de sinais entre registradores com o uso de operações lógicas, como mostrado na Fig. 1. O RTL além de dar uma ideia da quantidade de hardware que uma implementação do algoritmo exige, também ajuda no entendimento do funcionamento do mesmo no mais baixo nível.

Apesar da simplicidade, o RTL não vai fornecer informações como a frequência máxima suportada ou a potência exigida para o funcionamento do circuito. Para ter acesso a essas informações, a implementação do RTL tem de ser feita, e o uso de *Field-Programmable Gate Arrays* (FPGAs) é extensamente aplicado na literatura para essa finalidade (KOPPARTHI et al., 2022; KOYUNCU et al., 2020; KALANADHABHATTA et al., 2020; GARCIA-BOSQUE et al., 2019; PANDA; RAY, 2020; GUPTA; CHAUHAN, 2021).

Ao implementar um PRNG utilizando um software de implementação em FPGA como o Quartus Prime Software, da Intel®, um relatório com os resultados da implementação é fornecido. Ao invés de mostrar a ocupação em hardware do algoritmo em termos de números de registradores e portas lógicas exigidas, o relatório irá informar o número de registradores e de elementos lógicos (ou blocos lógicos). A composição de um elemento lógico pode variar entre as diferentes famílias de FPGAs que existem, mas no geral ele contém uma *Look Up Table* (LUT), um registrador (que pode estar sendo usado ou não) e algumas portas lógicas. Um exemplo

Figura 3 – Exemplo simplificado em um elemento lógico de uma FPGA.



Fonte: O autor (2023)

simplificado de um elemento lógico está ilustrado na Fig. 3. O resultado da implementação e o relatório fornecido pelo software nos permite definir algumas métricas:

- Número de LUTs utilizadas.
- Número de registradores (FFs, *Flip-Flops*) utilizados.
- Número de unidades de *Digital Signal Processors* (DSPs) utilizadas. (Os PRNGs apresentados nesta tese não empregam DSPs, mas alguns dos presentes na literatura utilizam.)
- Máxima frequência admitida pela implementação, medida em mega-hertz (MHz).
- Número de bits pseudoaleatórios gerados por ciclo de *clock*.
- A frequência de bits pseudoaleatórios gerados por segundo, chamada de *Throughput*, medida em megabits por segundo (Mbps). Este valor é a multiplicação da máxima frequência pelo número de bits por ciclo de *clock*.
- A potência dissipada pela implementação operando na máxima frequência, medida em miliwatts (mW).
- A potência do item acima dividida pelo *Throughput*, medida em mW/Mbps. Este valor permite ter uma ideia da potência necessária para gerar um bit pseudoaleatório do algoritmo.

As métricas apresentadas são utilizadas no Capítulo 3 para comparar os PRNGs propostos com outros da literatura.

2.2 AUTENTICAÇÃO EM CAMADA FÍSICA

PLA é uma técnica de autenticação de usuários em um sistema de comunicação que utiliza as características físicas do canal para tal. Dessa maneira, pode-se empregar PLA para identificar a identidade de um usuário ou dispositivo, e para isso, utiliza-se características do canal, que podem incluir o ruído, a resposta ao impulso e a intensidade do sinal.

Uma das abordagens mais comuns de PLA é a autenticação baseada em *tags*, em que cada mensagem transmitida é acompanhada de uma *tag* utilizada para autenticar o emissor da mensagem. A *tag* pode ser gerada de diversas maneiras, podendo incluir informações do canal de comunicação e uma chave secreta. Este trabalho aborda os sistemas de PLA que utilizam uma chave secreta no processo de geração da *tag*. Como a chave secreta é de conhecimento apenas dos usuários legítimos, apenas eles devem poder enviar *tags* autênticas.

O foco desta dissertação, no que diz respeito a PLA, está relacionado ao método de geração da *tag* utilizado no sistema. No Capítulo 4, é proposto um métodos de geração da *tag* que utiliza mapas caóticos. Em geral, o desempenho associado a um método de geração da *tag* é obtido por meio de métricas associadas à teoria da informação (MAURER, 2000), que são utilizadas neste trabalho. Estas métricas medem o nível de informação que um atacante com acesso a *tags* pode ter sobre a chave secreta, além das probabilidades de sucesso do mesmo em alguns tipos de ataques.

A Subseção 2.2.1 introduz o leitor a um sistema geral de PLA com *tag*, e as Subseções 2.2.2, 2.2.3 e 2.2.4 apresentam as métricas de desempenho baseadas em teoria da informação, utilizadas para avaliar os sistemas de geração das *tags*.

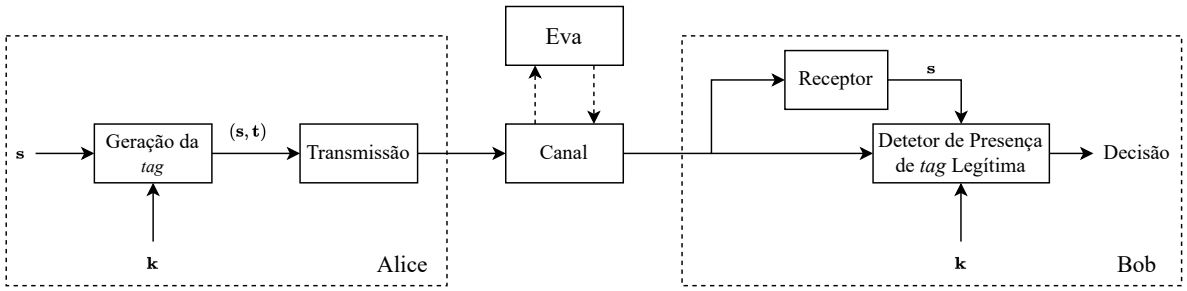
2.2.1 Descrição do sistema

Considera-se o cenário clássico em que três usuários compartilham o mesmo canal inseguro. Alice e Bob são os usuários legítimos, ou seja, eles utilizam um protocolo de PLA e compartilham uma chave secreta $\mathbf{k}$. Alice envia a Bob uma mensagem $\mathbf{s}$ juntamente com uma *tag* $\mathbf{t}$, que normalmente é gerada usando $\mathbf{s}$ e $\mathbf{k}$. Supõe-se que Bob decodifica $\mathbf{s}$ sem erros. Já que Bob também tem conhecimento da chave secreta $\mathbf{k}$, ele pode obter localmente, a partir de $\mathbf{s}$ e $\mathbf{k}$, a *tag* legítima. Bob então realiza um teste de detecção para verificar se a *tag* legítima está presente no sinal recebido. De uma maneira geral, esta detecção é feita por meio de um teste de hipóteses (YU; BARAS; SADLER, 2008). Se Bob conclui que a *tag* legítima está presente no sinal recebido, ele aceita a mensagem; caso contrário, a mensagem é rejeitada. Eva é um usuário malicioso que tem conhecimento de todos os detalhes do protocolo de autenticação, exceto a chave secreta. Considera-se que ela é um adversário ativo, sendo capaz de escutar as mensagens enviadas por Alice e enviar pacotes maliciosos para Bob. Esse sistema é ilustrado na Fig. 4.

Tendo em vista que o foco deste trabalho é o procedimento de geração das *tags*, não serão abordados os detalhes associados à forma de transmitir o par $(\mathbf{s}, \mathbf{t})$. As técnicas de geração da *tag* que serão abordadas no Capítulo 4 podem ser aplicadas a diferentes esquemas de transmissão, incluindo o amplamente utilizado esquema de PLA com superposição da *tag* (YU; BARAS; SADLER, 2008; YU; SADLER, 2011).

É feita em (MAURER, 2000) uma análise do problema de autenticação empregando técnicas de teoria da informação. Esta análise define algumas métricas como a equivocação da chave e os limites inferiores em algumas probabilidades de ataques. Nas Subseções 2.2.2, 2.2.3 e

Figura 4 – Sistema genérico de PLA utilizando *tags*.



Fonte: O autor (2023)

2.2.4, estas métricas são apresentadas.

2.2.2 Equivocação da chave

Para definir a equivocação da chave, utiliza-se a entropia $H(\cdot)$, definida pela primeira vez em (SHANNON, 1948). A função $H(X)$ mede o valor $\mathbb{E}[-\log_2 \Pr(X)]$, em que $\mathbb{E}[\cdot]$ é o operador valor esperado, X é uma variável aleatória e a função $\Pr(\cdot)$ indica a probabilidade associada aos valores de X. Ao definir a entropia utilizando o logaritmo em base 2, a medida da entropia é feita em bits.

Seja $(\mathbf{s}, \mathbf{t})$ um par interceptado/observado pela atacante Eva. A equivocação da chave $\mathbf{k}$ é a medida de incerteza que Eva tem sobre a chave legítima utilizada por Alice e Bob, dado que ela interceptou o par $(\mathbf{s}, \mathbf{t})$. Essa medida é representada por $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$. Os limites para a equivocação da chave são dados por

$$0 \leq H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) \leq H(\mathbf{k}). \tag{2.2}$$

O limite inferior $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = 0$ indica que Eva não tem nenhuma incerteza sobre a chave, ou seja, ela consegue determinar o valor da chave legítima. Já o limite superior $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = H(\mathbf{k})$ indica que o conhecimento do par $(\mathbf{s}, \mathbf{t})$ não fornece nenhuma informação sobre a chave, e a incerteza que Eva tem sobre a chave é, portanto, a mesma que ela tinha sem o par observado, dada por $H(\mathbf{k})$. Já que existe ruído associado ao canal de comunicação, Eva pode não ter certeza sobre a *tag* $\mathbf{t}$ exata transmitida, e conseguir apenas fazer uma estimativa $\hat{\mathbf{t}}$ da mesma. Neste caso, a equivocação da chave quando se considera o ruído do canal é representada por $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}})$. A equivocação para o caso sem ruído $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$ é referida neste trabalho como equivocação incondicional da chave.

2.2.3 Probabilidade de sucesso em um ataque de substituição

Neste tipo de ataque, Eva tenta adivinhar a chave para gerar um par $(\mathbf{s}', \mathbf{t}')$ legítimo, dado que ela interceptou um par $(\mathbf{s}, \mathbf{t})$ legítimo. A probabilidade de sucesso em um ataque de substituição P_S consiste em medir a probabilidade do par gerado por Eva ser legítimo.

Conforme mostrado em (MAURER, 2000, Seção VI), este ataque está limitado inferiormente pela probabilidade de Eva adivinhar a chave legítima $\mathbf{k}$ utilizada a partir do par observado, que é representada por

$$P_S \geq 2^{-H(\mathbf{k}|\mathbf{s},\mathbf{t})}. \tag{2.3}$$

Quando se considera o ruído associado ao canal tem-se que a probabilidade de sucesso em um ataque de substituição obedece a

$$P_S \geq 2^{-H(\mathbf{k}|\mathbf{s},\hat{\mathbf{t}})}. \tag{2.4}$$

2.2.4 Probabilidade de sucesso em um ataque de personificação

Em um ataque de personificação (ou *Impersonation Attack*), a atacante Eva intercepta um par $(\mathbf{s}, \mathbf{t})$ legítimo, e usa esse para criar um par $(\mathbf{s}', \mathbf{t})$, que ela espera que seja aceito como legítimo. A probabilidade de sucesso em um ataque de personificação P_I mede então a chance de Eva conseguir encontrar uma outra mensagem $\mathbf{s}$ que gera a mesma *tag*. A partir dos resultados em (MAURER, 2000), pode-se encontrar que um limitante inferior para P_I é

$$P_I \geq 2^{-I(\mathbf{t};\mathbf{k}|\mathbf{s})}, \tag{2.5}$$

em que $I(\mathbf{t}; \mathbf{k} \mid \mathbf{s})$ é a informação mútua condicional entre a *tag* e a chave secreta dada a mensagem. A expressão da informação mútua é reescrita como

$$\begin{aligned} I(\mathbf{t}; \mathbf{k} \mid \mathbf{s}) &= H(\mathbf{k} \mid \mathbf{s}) - H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) \\ &= H(\mathbf{k}) - H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}), \end{aligned} \tag{2.6}$$

em que $H(\mathbf{k} \mid \mathbf{s}) = H(\mathbf{k})$ vem do fato de que a chave é escolhida de forma aleatória, sem nenhuma dependência da mensagem. Um limitante inferior no valor de P_I é então dado por

$$P_I \geq 2^{-H(\mathbf{k})+H(\mathbf{k}|\mathbf{s},\mathbf{t})}. \tag{2.7}$$

Ao considerar o ruído associado ao canal, a probabilidade de sucesso em um ataque de personificação obedece a

$$P_I \geq 2^{-H(\mathbf{k})+H(\mathbf{k}|\mathbf{s},\hat{\mathbf{t}})}. \tag{2.8}$$

2.3 MAPAS CAÓTICOS

Os mapas caóticos são modelos matemáticos que apresentam comportamento complexo e imprevisível, geralmente referido como caos. Apesar de serem modelos determinísticos, ou seja, completamente determinados por suas condições iniciais, a imprevisibilidade está no fato de que esses mapas são extremamente sensíveis a mudanças em suas condições iniciais, de forma que uma pequena mudança nessas condições faz as órbitas associadas a um mapa divergirem com o tempo. Consequentemente, se as condições iniciais utilizadas não forem conhecidas, a tarefa de determiná-las é complexa. O conceito de mapas caóticos possui aplicações em diversos campos

da ciência. Nesta dissertação, estudam-se algumas aplicações desses mapas em algoritmos de segurança, focando em PRNGs e PLA.

Neste trabalho, são considerados apenas os sistemas caóticos de tempo discreto, em vez de contínuo. Nesses sistemas, a iteração sobre o mapa pode ser representada por uma simples relação de recorrência. Uma breve descrição de mapas caóticos unidimensionais é feita na Subseção 2.3.1. Na Subseção 2.3.2, o comportamento caótico de um mapa é descrito em termos do coeficiente de Lyapunov. Por fim, a Subseção 2.3.3 apresenta um tipo de mapa caótico que é empregado no desenvolvimento dos algoritmos estudados no Capítulo 4.

2.3.1 Definição de mapa caótico unidimensional

Um mapa caótico unidimensional é caracterizado por um sistema dinâmico com comportamento caótico obtido pela iteração sobre uma função não inversível e não linear adequada $f : A \to A$, de forma que

$$x[n] = f(x[n-1]), \quad n = 1, 2, \ldots \tag{2.9}$$

O valor $x[0]$ é a condição inicial do mapa. Chamamos de $\mathcal{O}$ a órbita de $x[0]$ sobre $f(\cdot)$, representada pela série temporal $x[n]_{n=0}^{\infty}$:

$$\begin{aligned} \mathcal{O} &= \Big[x[0] \quad f(x[0]) \quad f^2(x[0]) \cdots\Big] \\ &= \Big[x[0] \quad x[1] \quad x[2] \cdots\Big], \end{aligned} \tag{2.10}$$

em que $f^n(x) = f^{n-1}(f(x))$. Um exemplo de um mapa caótico unidimensional é o mapa da tenda (STROGATZ, 2015). O mapa da tenda com domínio $[-1, +1]$, mostrado na Fig. 5, é definido pela seguinte relação iterativa:

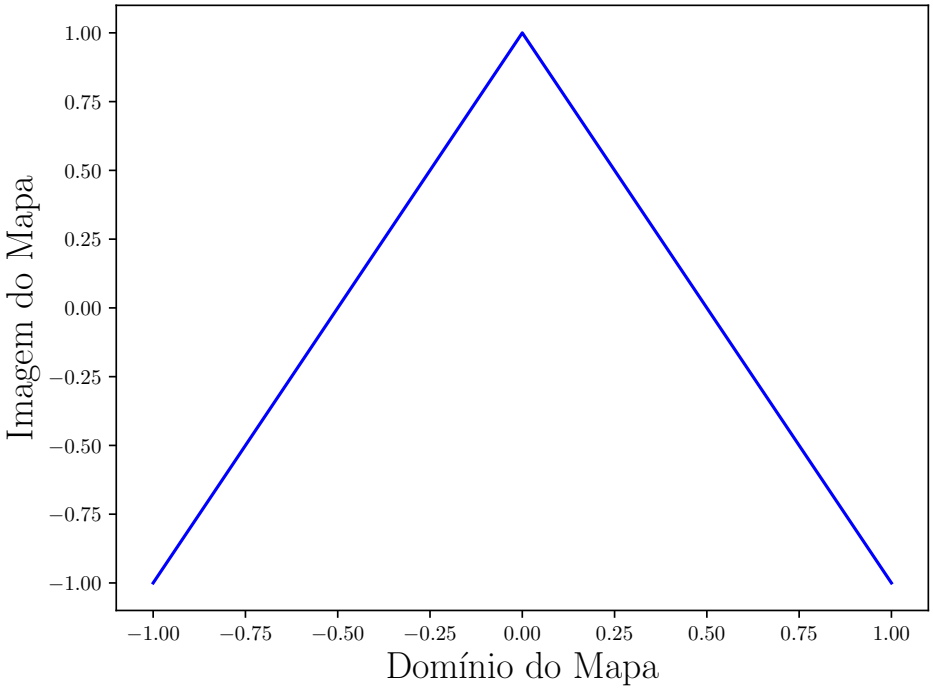
$$f(x) = \begin{cases} 2x + 1, & \text{for } x \in [-1, 0) \\ 1 - 2x, & \text{for } x \in [0, 1]. \end{cases} \tag{2.11}$$

A sensibilidade às condições iniciais é ilustrada na Fig. 6, em que as órbitas de duas condições iniciais separadas por um valor da ordem de 10^{-3} são mostradas. Perceba que a princípio as órbitas assumem valores próximos, mas com o passar das iterações elas vão assumindo formas distintas.

2.3.2 Expoente de Lyapunov e comportamento caótico

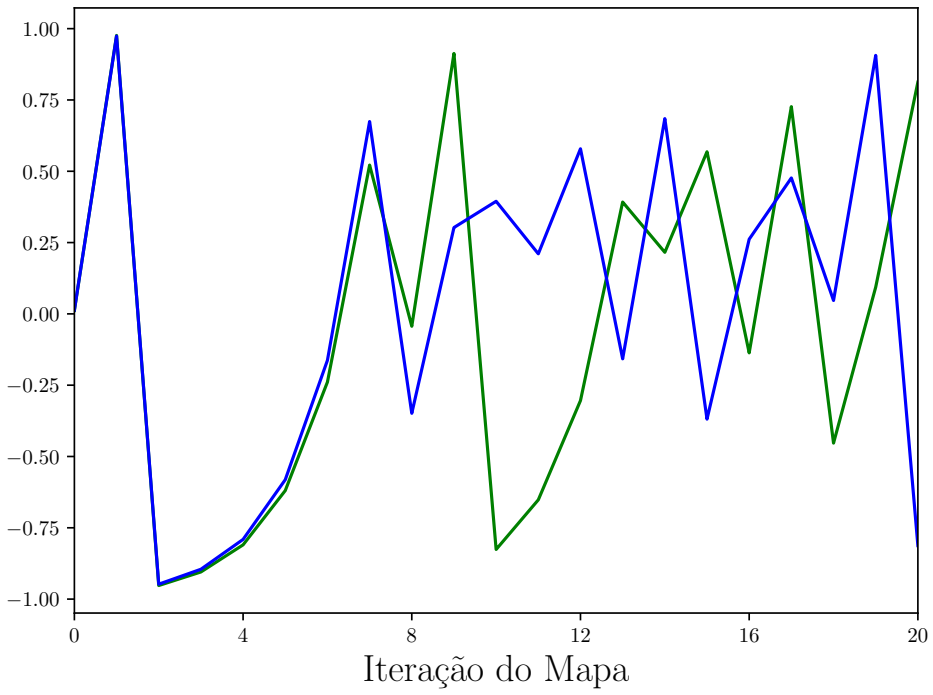
O coeficiente de Lyapunov, também conhecido como expoente de Lyapunov, é uma medida matemática que indica a sensibilidade às condições iniciais de um sistema dinâmico. Ele quantifica a taxa de divergência ou convergência exponencial de trajetórias próximas no espaço de fase do sistema, ou seja, indica o quão sensível o sistema é a pequenas perturbações em suas condições iniciais. Dessa forma, o coeficiente de Lyapunov pode ser usado para determinar se

Figura 5 – O mapa da tenda.



Fonte: O autor (2023)

Figura 6 – Órbitas de duas condições inicias do mapa da tenda separadas por valor da ordem de 10^{-3}.



Fonte: O autor (2023)

um sistema possui um comportamento caótico. Uma análise sobre o expoente de Lyapunov baseada em (STROGATZ, 2015) é apresentada no decorrer desta subseção.

Considere um sistema dinâmico unidimensional como o definido em (2.9). Dada uma condição inicial $x[0]$, considere uma ponto próximo $x[0] + \delta_0$, em que o valor de δ_0 é pequeno (tende a zero). Seja δ_n a separação entre as órbitas destes dois pontos após n iterações. O valor λ é chamado de expoente de Lyapunov se

$$|\delta_n| \approx |\delta_0| e^{n\lambda}. \tag{2.12}$$

Considerando que $\delta_0 \to 0$, pode-se reescrever a última expressão como

$$\begin{aligned} \lambda &\approx \lim_{\delta_0 \to 0} \frac{1}{n} \ln \left| \frac{\delta_n}{\delta_0} \right| \\ &= \lim_{\delta_0 \to 0} \frac{1}{n} \ln \left| \frac{f^n(x[0] + \delta_0) - f^n(x[0])}{\delta_0} \right| \\ &= \frac{1}{n} \ln \left| \frac{\mathrm{d} f^n(x[0])}{\mathrm{d}x} \right|. \end{aligned} \tag{2.13}$$

Pode-se expandir a derivada que aparece dentro o logaritmo no último passo de (2.13) usando a regra da cadeia, obtendo-se

$$\begin{aligned} \frac{\mathrm{d} f^n(x[0])}{\mathrm{d}x} &= (f^n)'(x[0]) \\ &= \prod_{i=0}^{n-1} f'(x[i]). \end{aligned} \tag{2.14}$$

Reescreve-se então (2.13) como

$$\begin{aligned} \lambda &\approx \frac{1}{n} \ln \left| \prod_{i=0}^{n-1} f'(x[i]) \right| \\ &= \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x[i])|. \end{aligned} \tag{2.15}$$

Quando toma-se o limite $n \to \infty$ em (2.15), tem-se o valor do expoente de Lyapunov para uma órbita associada a condição inicial $x[0]$,

$$\lambda = \lim_{n \to \infty} \frac{1}{n} \sum_{i=0}^{n-1} \ln |f'(x[i])|. \tag{2.16}$$

O comportamento caótico é caracterizado por um $\lambda > 0$, o que garante que as órbitas associadas a duas condições iniciais próximas se separam seguindo uma relação exponencial, ou seja, o mapa é sensível a pequenas mudanças nas condições iniciais. Em geral, o cálculo de λ é feito computacionalmente, mas para o caso específico do mapa da tenda (2.11) (um mapa linear por

partes), o valor de λ é obtido analiticamente como

$$\begin{aligned}\lambda &= \lim_{n\to\infty} \frac{1}{n}\sum_{i=0}^{n-1} \ln|f'(x[i])| \\ &= \lim_{n\to\infty} \frac{1}{n}\sum_{i=0}^{n-1} \ln 2 \\ &= \lim_{n\to\infty} \frac{n\ln 2}{n} \\ &= \ln 2\end{aligned} \tag{2.17}$$

e, como $\ln 2 > 0$, o mapa possui um comportamento caótico.

2.3.3 Mapas com pré-imagens binárias constantes

Uma classe de mapas caóticos de interesse neste trabalho são os mapas com pré-imagens binárias constantes. Seja $f(\cdot)$ a função que representa um mapa caótico unidimensional. Define-se a i-ésima pré-imagem do ponto y como o conjunto $\mathcal{S}_i(y) = \{x \mid f^i(x) = y\}$. Os mapas com pré-imagens binárias constantes são aqueles que obedecem a $|\mathcal{S}_i(y)| = 2^i, i \geq 0$, para todos os valores de y pertencentes à imagem de $f(\cdot)$, exceto por possivelmente um conjunto finito de pontos. Existem vários mapas que possuem essa propriedade, como o mapa logístico (STROGATZ, 2015), o mapa da tangente hiperbólica (CHAVES; SOUZA; PIMENTEL, 2016) e o mapa da tenda (STROGATZ, 2015).

Para exemplificar a propriedade das pré-imagens binárias constantes para o mapa da tenda, observa-se a partir de (2.11) que, dado um valor y pertencente ao intervalo $[-1, +1]$, tem-se que

$$\begin{aligned}\mathcal{S}_1(y) &= \left\{\pm\frac{y-1}{2}\right\} \\ &= \left\{\frac{y-1}{2}, \frac{1-y}{2}\right\},\end{aligned} \tag{2.18}$$

e, portanto, $|\mathcal{S}_1(y)| = 2^1 = 2$. É possível obter $\mathcal{S}_2(y)$ a partir de $\mathcal{S}_1(y)$ da seguinte forma:

$$\begin{aligned}\mathcal{S}_2(y) &= \left\{\mathcal{S}_1\left(\frac{y-1}{2}\right), \mathcal{S}_1\left(\frac{1-y}{2}\right)\right\} \\ &= \left\{\pm\frac{\left(\frac{y-1}{2}\right)-1}{2}, \pm\frac{1-\left(\frac{1-y}{2}\right)}{2}\right\} \\ &= \left\{\frac{\left(\frac{y-1}{2}\right)-1}{2}, \frac{1-\left(\frac{y-1}{2}\right)}{2}, \frac{\left(\frac{1-y}{2}\right)-1}{2}, \frac{1-\left(\frac{1-y}{2}\right)}{2}\right\}.\end{aligned} \tag{2.19}$$

Logo, $|\mathcal{S}_2(y)| = 2^2 = 4$. De forma iterativa, pode-se obter $\mathcal{S}_i(y)$ para o mapa da tenda da seguinte maneira:

$$\mathcal{S}_{i+1}(y) = \left\{\mathcal{S}_i\left(\frac{y-1}{2}\right), \mathcal{S}_i\left(\frac{1-y}{2}\right)\right\}, i \geq 0, \tag{2.20}$$

e, consequentemente, em cada iteração inversa sobre o mapa, o conjunto de pré-imagens dobra de tamanho, resultando em $|\mathcal{S}_i(y)| = 2^i$, em que $i \geq 0$. Considera-se que, quando $i = 0$, tem-se o caso trivial $\mathcal{S}_0(y) = \{y\}$.

Para exemplificar este resultado, considere o mapa da tenda e o ponto $y = 0$. Os conjuntos $\mathcal{S}_1(0), \mathcal{S}_2(0)$ e $\mathcal{S}_3(0)$ podem ser obtidos de forma iterativa a partir de (2.20) como segue,

$$\begin{aligned}
\mathcal{S}_0(0) &= \{0\}, \\
\mathcal{S}_1(0) &= \left\{-\frac{1}{2}, +\frac{1}{2}\right\}, \\
\mathcal{S}_2(0) &= \left\{-\frac{3}{4}, -\frac{1}{4}, +\frac{1}{4}, +\frac{3}{4}\right\}, \\
\mathcal{S}_3(0) &= \left\{-\frac{7}{8}, -\frac{5}{8}, -\frac{3}{8}, -\frac{1}{8}, +\frac{1}{8}, +\frac{3}{8}, +\frac{5}{8}, +\frac{7}{8}\right\}.
\end{aligned} \tag{2.21}$$

Na prática, o conjunto $\mathcal{S}_i(0)$ é obtido dividindo-se o intervalo $[-1, +1]$ igualmente em 2^i partes, e, posteriormente, selecionando-se os elementos centrais a cada parte.

A maneira como os elementos dos conjuntos em (2.21) iteram sobre o mapa até colidirem no ponto $y = 0$ pode ser descrita como uma árvore binária. A Fig. 7 ilustra esse comportamento, em que cada nó tem dois filhos que, ao iterarem sobre o mapa, colidem no valor do nó pai.

O conceito de mapas com pré-imagens binárias constantes é fundamental para o desenvolvimento do algoritmo de geração da *tag* para PLA proposto no Capítulo 4.

2.4 INTRODUÇÃO A q-ANÁLOGOS E CORPOS FINITOS

O conhecimento sobre q-Análogos e Corpos Finitos é necessário para o entendimento do PRNG descrito na Seção 3.2. Uma breve introdução a esses assuntos é feita nas Subseções 2.4.1 e 2.4.2, respectivamente.

2.4.1 q-Análogos

Na teoria clássica de q-análogos, considere um número inteiro não negativo n e um número real q. O q-análogo de n é denotado por $[n]_q$ e é definido por meio da expressão

$$[n]_q \triangleq \frac{1 - q^n}{1 - q}. \tag{2.22}$$

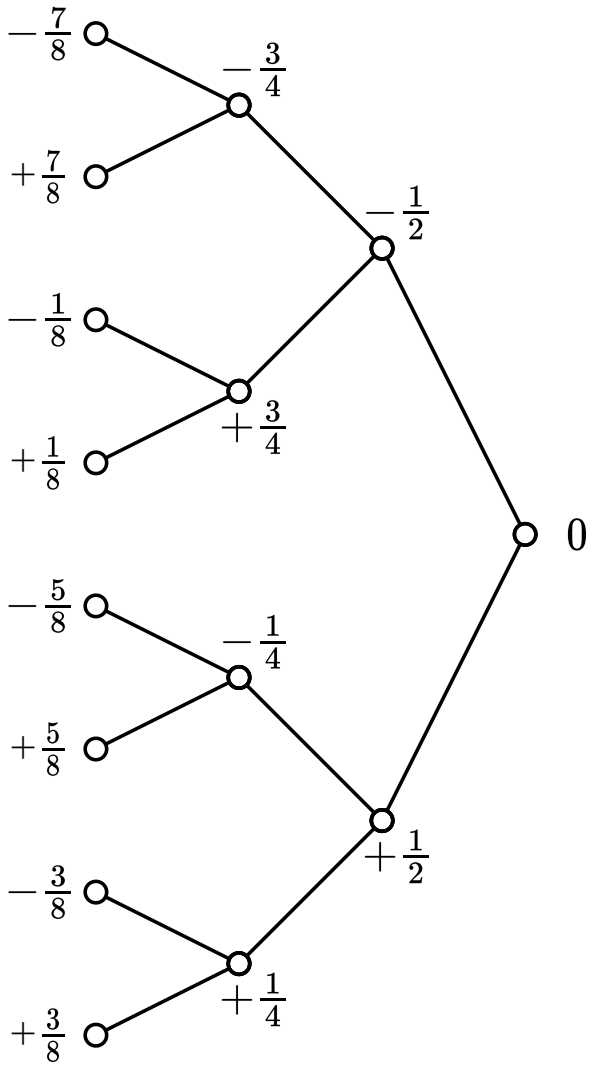
De forma equivalente, pode-se representar $[n]_q$ como uma série de potências em relação a q:

$$[n]_q = 1 + q + q^2 + \cdots + q^{n-1}. \tag{2.23}$$

Define-se que o q-análogo de zero seja igual a zero, ou seja, $[0]_q = 0$. A partir de (2.22), é evidente que $[1]_q = 1$. Quando q se aproxima de 1, tem-se $\lim_{q\to 1}[n]_q \triangleq [n]_{q\to 1} = n$, conforme indicado por (2.23). Os q-análogos também podem ser definidos por meio da relação de recorrência

$$[n + 1]_q = q[n]_q + 1, \tag{2.24}$$

Figura 7 – Árvore binária dos elementos presentes em $\mathcal{S}_1(0), \mathcal{S}_2(0)$ e $\mathcal{S}_3(0)$ para o mapa da tenda.



Fonte: O autor (2023)

que é obtida diretamente de (2.23).

O PRNG desenvolvido na Seção 3.2 se utiliza de uma definição de q-análogos sobre corpos finitos.

2.4.2 Corpos finitos

Formalmente, um corpo finito é definido a partir de um conjunto finito W com duas operações binárias em W, chamadas de adição $(+)$ e multiplicação $(\times)$, que mapeiam pares de elementos de W em algum outro elemento de W. O corpo finito formado por W e as duas operações é usualmente representado por $(W, +, \times)$. Dados quaisquer $a, b, c \in W$, essas duas operações precisam satisfazer os seguintes axiomas:

- Associatividade:

$$\begin{aligned} a + (b + c) &= (a + b) + c \\ a \times (b \times c) &= (a \times b) \times c. \end{aligned} \tag{2.25}$$

- Comutatividade:

$$\begin{aligned} a + b &= b + a \\ a \times b &= b \times a. \end{aligned} \tag{2.26}$$

- Identidades aditivas e multiplicativas: existem em W os elementos 0 e 1, chamados de identidade aditiva e multiplicativa, respectivamente, de forma que

$$\begin{aligned} a + 0 &= a \\ a \times 1 &= a. \end{aligned} \tag{2.27}$$

- Inversos aditivos: para todo elemento $a \in W$, vai existir em W o inverso aditivo de a, chamado de $-a$, tal que $a + (-a) = 0$.

- Inversos multiplicativos: para todo elemento $a \in W$ ($a \neq 0$), vai existir em W o inverso multiplicativo de a, chamado de a^{-1}, tal que $a \times a^{-1} = 1$.

- Distributividade:

$$a \times (b + c) = (a \times b) + (a \times c). \tag{2.28}$$

Se define a ordem multiplicativa de um elemento a em um corpo finito como o menor número inteiro positivo n que, quando esse elemento é multiplicado repetidamente n vezes por si mesmo, resulta na identidade multiplicativa do corpo.

Considere um número primo p e o conjunto $W = \{0, 1, 2, \ldots, p - 1\}$, que representa o conjunto de inteiros módulo p, $\mathbb{Z}_p$, também conhecido como o anel de inteiros módulo p. Utilizando as operações módulo p de adição ($+_p$) e multiplicação ($\times_p$), esse conjunto forma o corpo finito $(W, +_p, \times_p)$ de característica p, também conhecido como GF(p).

Os corpos finitos considerados na definição de q-análogos, utilizada no desenvolvimento do PRNG da Seção 3.2, são os do tipo GF(p).

3 TÉCNICAS PARA GERAÇÃO DE NÚMEROS PSEUDOALEATÓRIOS

Sendo importantes em diversas aplicações criptográficas, os PRNGs fornecem uma forma rápida e eficiente de gerar sequências de números com um comportamento similar ao de sequências aleatórias. Essas características tornam o uso de PRNGs adequados para a geração de chaves criptográficas, simulação computacional de eventos de comportamento estocástico, simulação de resultados imprevisíveis em jogos eletrônicos de azar, dentre outras. Este capítulo aborda dois algoritmos para geração de números pseudoaleatórios, mostrando a qualidade dos mesmos através de testes estatísticos e implementação em hardware utilizando FPGAs.

Na Seção 3.1, um PRNG baseado no Mapa de Arnold é proposto. Já na Seção 3.2, se propõe um PRNG com base numa definição de q-Análogos sobre corpos finitos. Por fim, a Seção 3.3 compara as implementações em FPGA desses algoritmos com PRNGs encontrados na literatura.

3.1 PRNG BASEADO NO MAPA DE ARNOLD

O Mapa de Arnold Discreto $\Lambda : \mathbb{Z}_v \times \mathbb{Z}_v \to \mathbb{Z}_v \times \mathbb{Z}_v$ é um mapa caótico bidimensional definido como

$$\Lambda(x, y) = (2x + y, x + y) (\text{mod } v), \tag{3.1}$$

em que $v = p^m$ é uma potência de um número primo p, e $\mathbb{Z}_v$ é o anel de inteiros módulo v. A aplicação iterativa da função $\Lambda(\cdot)$ sobre duas condições iniciais $(x_0, y_0) \in \mathbb{Z}_v \times \mathbb{Z}_v$ gera uma sequência bidimensional $\{(x_0, y_0), (x_1, y_1), (x_2, y_2), \ldots\}$, da qual (x_n, y_n) é definido recursivamente como

$$\begin{bmatrix} x_n \\ y_n \end{bmatrix} = \mathbf{B}^{2n} \begin{bmatrix} x_0 \\ y_0 \end{bmatrix} (\text{mod } v), \tag{3.2}$$

em que $\mathbf{B}$ é dada por

$$\mathbf{B} = \begin{bmatrix} 1 & 1 \\ 1 & 0 \end{bmatrix}. \tag{3.3}$$

As sequências z, representadas por $\{z_n\} = \{z_0, z_1, z_2, \ldots\}$, $z_n \in \mathbb{Z}_v, \forall\, n \geq 0$, são definidas em (SOUZA; CHAVES; PIMENTEL, 2021) como

$$z_n = x_n y_n (\text{mod } v), \tag{3.4}$$

em que $(x_n, y_n) \in \mathbb{Z}_v \times \mathbb{Z}_v$ é obtida de (3.2). Como apresentado em (SOUZA et al., 2022) para o caso $v = 2^m$, o período máximo para as sequências z é dado por $3 \times 2^{m-3}$, e é obtido para qualquer sequência que possui condições iniciais (x_0, y_0) não nulas e não divisíveis por 2. Um PRNG para $\mathbb{Z}_{2^m}$ poderia então ser desenvolvido diretamente das sequências z correspondentes. A desvantagem de um PRNG definido dessa forma está no fato de que o mesmo não passa pelos

testes estatísticos discutidos na Subseção 2.1.1. Para passar nos testes, é necessário considerar a representação binária em $\lceil \log_2 v \rceil$ bits associada a um z_i, dada por B_{z_i}, em que é necessário eliminar alguns bits associados a cada z_i, o que não é desejado, visto que esta operação diminui a taxa de geração de bits do PRNG. Para superar esse problema, define-se uma pós-operação aos bits em B_{z_i} que é realizada com baixo custo computacional, pois envolve apenas operações $\oplus$. Essa operação quebra a correlação que pode existir em alguns bits associados à sequência. O i-ésimo bloco B_i de saída associado ao PRNG sobre $\mathbb{Z}_{2^m}$ é então dada por

$$B_i = B_{z_{i-1}} \oplus \tilde{B}_{x_i} \oplus \tilde{B}_{y_i}, \tag{3.5}$$

na qual $\tilde{B}_{x_i}$ e $\tilde{B}_{y_i}$ representam os blocos de bits associados a x_i e y_i, respectivamente, mas com seus bits invertidos. Um outro PRNG baseado no mapa de Arnold é apresentado em (SOUZA; CHAVES; PIMENTEL, 2021), no qual se consideram as sequências z sobre $\mathbb{Z}_{3^m}$, em que cada bloco de saída B_i é obtido eliminando 8 bits presentes em B_{z_i}, sendo esses os cinco bits mais significativos (MSBs) e os três bits menos significativos (LSBs). Tanto o PRNG apresentado para $\mathbb{Z}_{2^m}$ como o PRNG sobre $\mathbb{Z}_{3^m}$ passam pelos testes estatísticos da bateria de testes SP800-22 (BASSHAM III et al., 2010), em que se utilizam 100 subsequências de saída de cada PRNG, cada uma com comprimento 10^6 bits. Como recomendado em (BASSHAM III et al., 2010), se utiliza um nível de confiança para os testes de $\alpha = 0{,}01$. A proporção mínima para aprovação em cada teste individual para os parâmetros considerados é $0{,}96$. Os resultados associados aos testes estão mostrados na Tabela 1, na qual se considera os anéis de inteiros $\mathbb{Z}_{2^{32}}$ e $\mathbb{Z}_{3^{25}}$, pois ambos geram 32 bits pseudoaleatórios por bloco de saída.

Nas Subseções 3.1.1 e 3.1.2, a arquitetura dos PRNGs sobre $\mathbb{Z}_{2^m}$ e $\mathbb{Z}_{3^m}$ são apresentadas, respectivamente.

3.1.1 Arquitetura do PRNG baseado no mapa de Arnold sobre $\mathbb{Z}_{2^m}$

A Fig. 8 mostra o RTL da arquitetura do PRNG apresentado para $\mathbb{Z}_{2^m}$ com $m = 32$, na qual é observada sua facilidade de dimensionamento para qualquer anel de inteiros $\mathbb{Z}_{2^m}$. Conforme destacado na Fig. 8, o circuito é dividido em seis níveis. O primeiro nível calcula os valores de x_{n+1} e y_{n+1} a partir de x_n e y_n, respectivamente, implementando essencialmente (3.1). O segundo nível, composto por dois multiplexadores MUX 2×1, seleciona se as operações são realizadas em x_0 e y_0 (condições iniciais) ou em valores subsequentes. No terceiro nível, formado por um par de *flip-flops* do tipo D, os valores de x_n e y_n são armazenados durante um período de *clock*, durante o qual os valores de x_{n+1} e y_{n+1} calculados no primeiro nível e o valor de z_n são calculados no quarto nível. Finalmente, na próxima borda de subida do *clock*, a saída do registrador do quinto nível é atualizada para z_n, e a saída dos *flip-flops* do terceiro nível também é atualizada para x_{n+1} e y_{n+1}. Por fim, o sexto nível corresponde à operação de pós-processamento que foi descrita em (3.5).

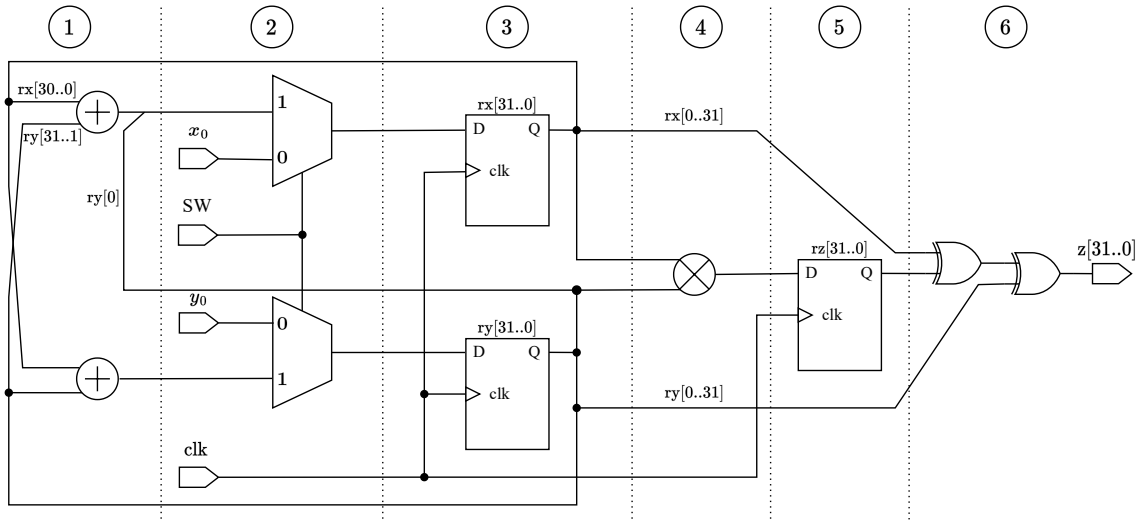
O circuito calcula um elemento z_i da sequência z_n a cada período de *clock*. Devido à unidade de pós-processamento, todos os $m = 32$ bits de z_i são extraídos, formando assim uma

Tabela 1 – Proporções do teste NIST para os anéis de inteiros $\mathbb{Z}_{2^{32}}$ e $\mathbb{Z}_{3^{25}}$. Nos testes múltiplos (indicados por *), é mostrada apenas a proporção mínima.

Teste Estatístico	$\mathbb{Z}_{2^{32}}$	$\mathbb{Z}_{3^{25}}$
Frequência	1	0,99
Frequência em Blocos	1	0,98
Soma Cumulativa*	0,99	0,99
Corridas	1	0,99
Corridas mais Longas	1	0,97
Posto de Matriz	0,99	1
FFT	1	0,99
Corresp. Mod. Não-Sobreposto*	0,96	0,96
Corresp. Mod. Sobreposto	0,99	0,99
Universal	0,98	1
Entropia Aproximada	0,99	1
Excur. Aleatórias*	0,97	0,97
Var. em Excur. Aleatórias*	0,97	0,96
Serial*	0,98	1
Complexidade Linear	0,99	0,99

Fonte: (SOUZA et al., 2022)

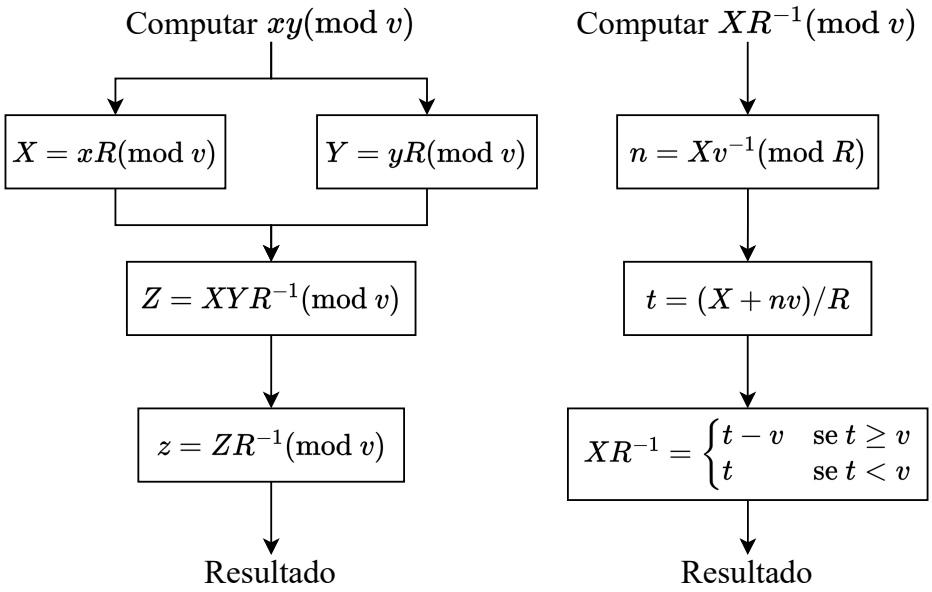
Figura 8 – RTL do PRNG baseado no mapa de Arnold sobre $\mathbb{Z}_{2^{32}}$.



Fonte: (SOUZA et al., 2022)

sequência binária. A complexidade do circuito está concentrada principalmente no quarto nível, onde a operação de produto é usada para calcular z_n. Como os valores são representados na base dois e reduzidos módulo uma potência de dois, a operação $2x_n + y_n$ em (3.1) para o cálculo de x_{n+1} do mapa é realizada concatenando $y[0]$ ao LSB da operação $x[30..0] + y[31..1]$, seguida pela redução módulo 2^{32} do resultado, o que não adiciona complexidade à arquitetura proposta.

Figura 9 – Diagrama de blocos do algoritmo MMM.



Fonte: (SOUZA et al., 2022)

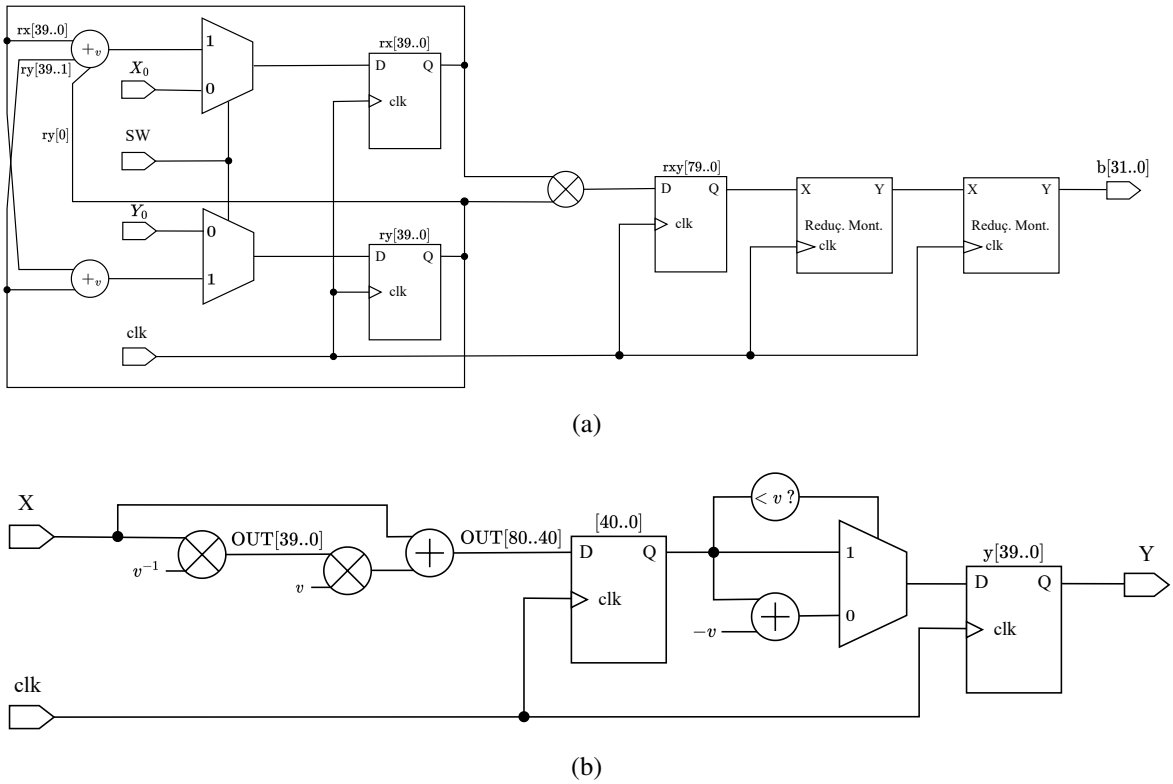
3.1.2 Arquitetura do PRNG baseado no mapa de Arnold sobre $\mathbb{Z}_{3^m}$

A implementação do PRNG sobre $\mathbb{Z}_{3^m}$ é baseada no algoritmo de Multiplicação Modular de Montgomery (MMM), já que o cálculo da sequência z envolve o produto de dois elementos em $\mathbb{Z}_{3^m}$. Em seguida, descreve-se brevemente o algoritmo MMM, que está detalhado em (BLAHUT, 2010), e o RTL para o PRNG correspondente.

Os passos do algoritmo MMM são representados pelo diagrama de blocos na Fig. 9. O primeiro passo é calcular as formas de Montgomery de x e y, chamadas de X e Y, respectivamente, como mostrado no diagrama à esquerda. A forma de Montgomery do produto $xy(\mathrm{mod}\ v)$ é então calculada usando $Z = XYR^{-1}(\mathrm{mod}\ v)$. O diagrama à direita é usado para realizar essa redução, na qual a redução módulo v é substituída pela redução módulo R. Tipicamente, utiliza-se R na forma 2^ℓ, $\ell \geq 2$, e $R > v$; portanto, a redução módulo R de um inteiro $a_0 + a_1 \times 2^1 + a_2 \times 2^2 + \cdots + a_t \times 2^t$ consiste em descartar as posições dos bits a_j tal que $j \geq \ell$. Essa redução modular simplificada também é usada na conversão de Z para z.

O RTL do PRNG sobre $\mathbb{Z}_{3^{25}}$ é apresentado na Fig. 10a. É semelhante à mostrada na Fig. 8, mas com diferenças nas unidades aritméticas e a inclusão dos blocos de redução de Montgomery. A implementação do diagrama na Fig. 9 começa na unidade de multiplicação e avança até a saída da Fig. 10a. As entradas do PRNG são X_0 e Y_0, as formas de Montgomery das condições iniciais x_0 e y_0, respectivamente. Elas são selecionadas definindo SW como zero no primeiro ciclo de *clock*. Durante os ciclos de *clock* subsequentes, as trilhas de realimentação formam as entradas para as unidades de soma do módulo v quando SW vale um. A unidade superior calcula $\mathsf{rx}[39..0] + \mathsf{ry}[39..1]$ e concatena $\mathsf{ry}[0]$ ao LSB do resultado, após subtrair v desse resultado no máximo duas vezes, a saída é equivalente a calcular X_n, a forma de Montgomery de x_n em (3.1).

Figura 10 – RTL utilizado para gerar as sequências binárias sobre $\mathbb{Z}_{3^{25}}$. (a) RTL para o PRNG apresentado, (b) RTL da subunidade de redução de Montgomery.



Fonte: (SOUZA et al., 2022)

A saída da unidade inferior é $\mathsf{rx}[39..0] + \mathsf{ry}[39..0]$ com no máximo uma subtração de v, o que é equivalente ao cálculo de Y_n, a forma de Montgomery de y_n em (3.1). Os valores X_n e Y_n, disponíveis nas saídas dos *flip-flops* mais à esquerda, são as entradas de 40 bits do multiplicador, que produz uma saída com 80 bits. Os próximos passos são as reduções de Montgomery indicadas no diagrama à direita na Fig. 9. A saída do último bloco são os bits extraídos da representação binária de z_n, conforme especificado em (SOUZA; CHAVES; PIMENTEL, 2021).

A RTL na Fig. 10b corresponde à etapa de redução de Montgomery, que são os dois últimos blocos na Fig. 10a. Seguindo o diagrama à direita na Fig. 9, e usando $R = 2^{40}$, o cálculo de $m = Xv^{-1} (\text{mod } R)$ é implementado pelo primeiro multiplicador na Fig. 10b. A redução por uma potência de dois torna o processo muito simples. O segundo bloco do diagrama é implementado pelo segundo multiplicador e pela unidade de soma, e corresponde ao quociente da divisão $(X + mv)/R$. O resultado é a saída do primeiro *flip-flop* no próximo ciclo de *clock*, que passa pela implementação do terceiro bloco do diagrama, um módulo de redução v. O módulo de redução emite o sinal na entrada 0 do MUX se o quociente for maior ou igual a v; caso contrário, o sinal na entrada 1 é selecionado. A saída é atualizada para $XR^{-1} (\text{mod } v)$ no próximo ciclo de *clock*.

3.2 PRNG BASEADO EM q-ANÁLOGOS SOBRE CORPOS FINITOS

Uma definição para q-análogos sobre corpos finitos é apresentada em (SOUZA et al., 2023), em que, para um número primo p e um $q \in \text{GF}(p), q \neq 0$, o q-análogo de $n \in \text{GF}(p), n \neq 0$, é definido por

$$[n]_q \triangleq \frac{q^n - 1}{q - 1} (\text{mod } p), \tag{3.6}$$

e o q-análogo de $n = 0$ é definido por

$$\begin{cases} [0]_1 & \triangleq 0 \\ [0]_q & \triangleq (1-q)^{-1} (\text{mod } p),\ q \neq 1. \end{cases} \tag{3.7}$$

De maneira similar ao q-análogo clássico sobre inteiros, apresentado na Subseção 2.4.1, o q-análogo sobre corpos finitos também é escrito como uma série de potências em q,

$$[n]_q = (1 + q + q^2 + \cdots + q^{n-1}) (\text{mod } p). \tag{3.8}$$

Também pode-se utilizar a série de potências em (3.8) para encontrar, para um $n \geq 1$, a relação recursiva

$$[n+1]_q = (q[n]_q + 1) (\text{mod } p), \tag{3.9}$$

na qual $[0]_q$ é definida particularmente por (3.7). O PRNG baseado na definição aqui apresentada de q-análogos sobre corpos finitos se baseia na relação de recorrência (3.9). Inicialmente é preciso definir valores para p e q, além de um valor inicial $[n]_q$, que é a semente do algoritmo. A sequência pseudoaleatória é então obtida diretamente da aplicação recursiva de (3.9). Perceba que, ao assumir que $[n]_q = [n+1]_q$ em (3.9), se encontra que $[n]_q = [0]_q = (1-q)^{-1} (\text{mod } p)$, ou seja, a aplicação recursiva de $[0]_q$ em (3.9) gera uma sequência constante. Para um $[n]_q \neq [0]_q$, tem-se que a sequência gerada vai possuir um período máximo de $p - 1$, já que ela no máximo vai passar pelos q-análogos de todos os números módulo p, exceto o 0. Em (SOUZA et al., 2023), é visto que o período dessa sequência pseudoaleatória será o mesmo que a ordem multiplicativa do elemento q.

As métricas estatísticas associadas à bateria de testes SP800-22 (BASSHAM III et al., 2010) são utilizadas para analisar a qualidade da sequência pseudoaleatória gerada pelo PRNG. Se utiliza um nível de confiança $\alpha = 0{,}01$ e 100 subsequências de comprimento 10^6 bits são utilizadas em cada teste. Os parâmetros escolhidos para o PRNG foram $p = 2^{31} - 1$, $q = 1025$, além de uma semente $[n]_q \neq [0]_q$. Essa escolha de parâmetros será justificada na Subseção 3.2.1. Os resultados relativos aos testes estatísticos estão mostrados na Tabela 2. O valor mínimo considerado para uma aprovação em todos os testes foi $0{,}96$.

3.2.1 Arquitetura do PRNG baseado em q-análogos

A Fig. 11 apresenta o esquema RTL da arquitetura proposta para um número primo do tipo $p = 2^e - 1$ (um número primo de Mersenne). As vantagens da escolha de um primo de

Tabela 2 – Proporções do teste NIST para as sequências binárias geradas pelo PRNG baseado em q-análogos para $p = 2^{31} - 1$ e $q = 1025$. Nos testes múltiplos (indicados por *), é mostrada apenas a proporção mínima.

Teste Estatístico	$q = 1025$
Frequência	0,990
Frequência em Blocos	0,980
Soma Cumulativa*	0,990
Corridas	0,995
Corridas mais Longas	0,990
Posto de Matriz	0,995
FFT	0,980
Corresp. Mod. Não-Sobreposto*	0,970
Corresp. Mod. Sobreposto	0,975
Universal	0,985
Entropia Aproximada	0,985
Excur. Aleatórias*	0,969
Var. em Excur. Aleatórias*	0,977
Serial*	0,995
Complexidade Linear	0,995

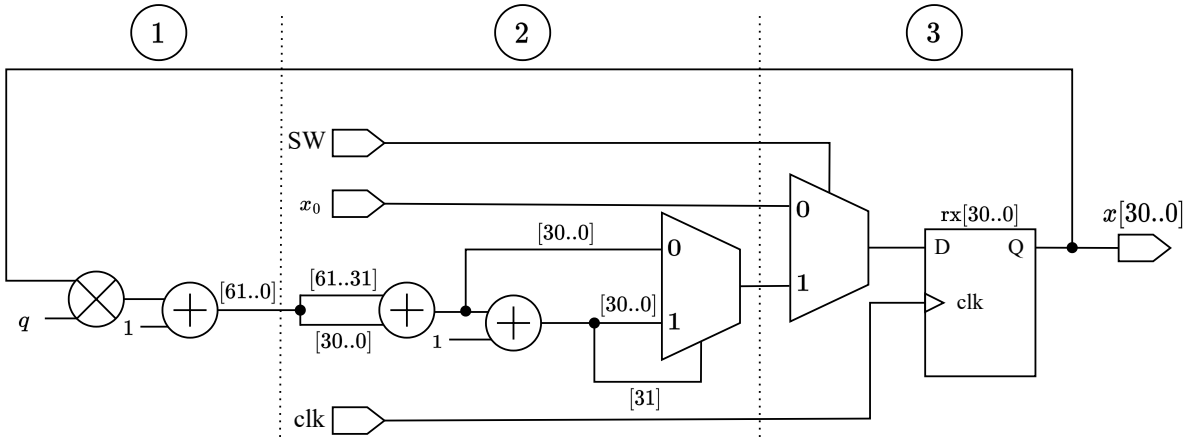
Fonte: (SOUZA et al., 2023)

Mersenne são destacadas nos próximos parágrafos. Especificamente, a implementação utiliza $e = 31$, mas é facilmente adaptada para outros números primos de Mersenne, bastando escolher um novo valor de q e ajustar os comprimentos dos barramentos e registradores para se adequar ao primo desejado, como mostrado na Fig. 11. Como destacado na Fig. 11, o circuito é dividido em três níveis, que juntos implementam (3.9). O primeiro nível calcula a relação de recorrência $q[n]_q + 1$, onde $[n]_q = x$ é a saída atual do sistema, ou seja, o q-análogo calculado no último ciclo de *clock*. O segundo nível calcula a redução módulo p do resultado, conforme explicado nos parágrafos seguintes. A terceira parte utiliza um MUX 2×1 para selecionar (com o sinal SW) se as operações são realizadas em x_0 (a condição inicial que precisa ser diferente de $[0]_q$) ou nos valores subsequentes, e também utiliza *flip-flops* do tipo D para armazenar a saída atual do sistema. Como os *flip-flops* são atualizados a cada ciclo de *clock*, e $p = 2^e - 1$, uma amostra pseudoaleatória de e bits é gerada a cada ciclo de *clock*.

A redução módulo p no segundo nível da Fig. 11 pode ser realizada por um algoritmo genérico, como o algoritmo de redução de Montgomery. No entanto, como $p = 2^e - 1$ é um número primo de Mersenne, um primo amigável de Montgomery, a redução é feita usando o Algoritmo 1(BAJARD; DUQUESNE, 2021). Dado que a saída da última etapa é $q[n]_q + 1 \leq (p-1)^2 + 1$, e $(p-1)^2 + 1 < p^2 + p = (p+1)p = 2^e p$, o requisito para a entrada do algoritmo é satisfeito.

Com esse algoritmo, a redução é simplificada para uma adição de e bits e uma incremen-

Figura 11 – RTL do PRNG baseado em q-análogos para um $p = 2^{31} - 1$.



Fonte: (SOUZA et al., 2023)

tação. A soma é calculada diretamente entre os e bits menos significativos da saída da etapa anterior e os e bits mais significativos, resultando em um número menor que $2p$. Agora, esse resultado precisa ser subtraído de p se for maior que p. Em vez de comparar diretamente o resultado com p, o que não é trivial, incrementamos o resultado em um e o comparamos com $p + 1 = 2^e$. Essa incrementação resulta em um número menor que $2p + 1 = 2^{e+1} - 1$, portanto, essa comparação é feita verificando apenas o $(e+1)$-ésimo bit do resultado. Esse bit faz com que o MUX 2×1 selecione os primeiros e bits do resultado incrementado (se for 1) ou o resultado original (se for 0). Observe que quando o $(e + 1)$-ésimo bit do resultado incrementado é um, e o ignoramos na saída do MUX, isso é equivalente a subtrair 2^e, portanto, essas operações (incrementar em um e ignorar o $(e + 1)$-ésimo bit) são equivalentes a subtrair $p = 2^e - 1$ de um resultado que é maior que p.

No exemplo considerado na Tabela 2, considerou-se o parâmetro $q = 2^{10} + 1 = 1025$. Essa escolha tem dois motivos principais. O primeiro é que esse valor de q é um gerador do grupo multiplicativo de GF(p), ou seja, garante o período máximo de $p - 1$ da sequência de saída do PRNG. O segundo motivo leva em conta a estrutura do valor de q, que pode ser escrito como uma potência de 2 incrementada de 1, o que faz com que a multiplicação por q, presente no primeiro nível da Fig. 11, seja reduzida a um deslocamento em dez bits à esquerda e uma soma. Os valores $p = 2^{31} - 1$ e $q = 1025$ foram os considerados na implementação do PRNG em FPGA feita na Seção 3.3.

3.3 COMPARAÇÕES DAS IMPLEMENTAÇÕES EM FPGA DOS PRNGS

Os RTLs dos PRNGs apresentados nas Seções 3.1 e 3.2 são aqui implementados na FPGA Xilinx Virtex-5 (V5) XC5VLX50T-1FFG1136 (ML505). Os resultados das implementações são mostrados em termos das métricas de implementação em hardware descritas na Subseção 2.1.2, e estão dispostos na Tabela 3. Os resultados associados a PRNGs encontrados na literatura

Algoritmo 1 Redução módulo um primo de Mersenne.

Requer: Um primo de Mersenne $p = 2^e - 1$
Entrada: $0 \leq a < 2^e p$
Saída: $r = a \pmod p$, em que $0 \leq r < p$
  Escreva $a = a_1 2^e + a_0$
  $r \leftarrow a_0 + a_1$
  $r' \leftarrow r + 1$
  se $r' \geq 2^e$ **então**
    $r \leftarrow r' \pmod{2^e}$
  retorna r

Tabela 3 – Comparações das implementações em FPGA dos PRNGs. Os PRNGs encontrados na literatura são enumerados de 1 a 6, correspondendo, respectivamente, aos PRNGs presentes em (KOPPARTHI et al., 2022; KOYUNCU et al., 2020; KALANADHABHATTA et al., 2020; GARCIA-BOSQUE et al., 2019; PANDA; RAY, 2020; GUPTA; CHAUHAN, 2021).

Método	1	2	3	4	5	6	Arnold $\mathbb{Z}_{3^{25}}$	Arnold $\mathbb{Z}_{2^{32}}$	q-Análogos
FPGA Utilizada	Zynq 7000	Virtex-6	Virtex-7		Virtex-5				
LUTs	4215	1355	631	510	480	340	5021	614	137
FFs	578	1318	662	120	128	81	329	99	31
DSPs	0	0	0	13	0	0	0	0	0
Max. Freq. (MHz)	81	464,688	52	132	219,35	341,384	46,75	112,64	122,54
Bits Aleat. por Ciclo	16	1	16	1	1	1	32	32	31
Throughput (Mbps)	1296	464,688	832	132	219,35	341,384	1496	3604,42	3798,74
Potência na Max. Freq. (mW)	-	-	-	-	42,36	2519,41	533	600	493
Potência/*Throughput* (mW/Mbps)	-	-	-	-	0,193	7,38	0,356	0,166	0,130

Fonte: O autor (2023)

também estão exibidos e enumerados de 1 a 6, correspondendo, respectivamente, aos métodos de geração de sequências pseudoaleatórias presentes em (KOPPARTHI et al., 2022; KOYUNCU et al., 2020; KALANADHABHATTA et al., 2020; GARCIA-BOSQUE et al., 2019; PANDA; RAY, 2020; GUPTA; CHAUHAN, 2021). Perceba que o PRNG baseado no mapa de Arnold para $\mathbb{Z}_{3^{25}}$ não apresenta bons resultados de ocupação (LUTs e FFs) quando comparados com os outros algoritmos, principalmente quando se trata de LUTs, o que é justificado pela complexidade da operação de redução módulo 3^{25}. Já os PRNGs sobre $\mathbb{Z}_{2^{32}}$ e o baseado em q-análogos apresentam resultados bastante competitivos em todos os aspectos, principalmente quando se trata da *Throughput*. O PRNG baseado em q-análogos é o que apresenta os melhores valores de ocupação quando comparados aos menores valores encontrados na literatura, apresentando uma redução no número de LUTs e FFs da ordem de 60% e 62%, respectivamente, além de também possuir a menor relação de Potência/*Throughput*. A baixa complexidade do PRNG baseado em q-análogos é uma consequência da escolha inteligente dos parâmetros p e q do mesmo, como detalhado na Subseção 3.2.1.

4 TÉCNICAS DE GERAÇÃO DA *TAG* PARA SISTEMAS DE PLA

O algoritmo de geração da *tag* é uma parte fundamental de um sistema de PLA, como ilustrado na Fig. 4. A *tag* garante a autenticidade do emissor da mensagem, além de proteger o sistema de ataques maliciosos que ameaçam a comunicação segura entre os usuários legítimos. Os sistema de geração da *tag* aqui discutidos consideram que a mensagem $\mathbf{s}$ e a chave $\mathbf{k}$ são vetores binários de comprimento K, e com valores pertencentes ao conjunto $\{-1, +1\}$. A *tag* $\mathbf{t}$ é um vetor de comprimento L. O sistema clássico de geração da *tag* é apresentado na Seção 4.1, e considera que $\mathbf{t}$ possui valores binários pertencentes ao conjunto $\{-1, +1\}$. A Seção 4.2 apresenta o sistema proposto de geração da *tag* a partir de sequências caóticas discretizadas, que também utiliza $\mathbf{t}$ com valores binários pertencentes ao conjunto $\{-1, +1\}$. Sempre que forem realizadas operações binárias entre vetores binários considere que o -1 assume o papel do 0 na operação. O contexto deixará claro se o tratamento dos vetores binários está considerando a aritmética binária usual com os valores $\{0, 1\}$. Uma comparação entre os diferentes sistemas de geração da *tag* é realizada na Seção 4.3, em que se consideram as métricas baseadas em teoria da informação discutidas na Seção 2.2. Por fim, a Seção 4.4 apresenta uma análise da informação que um usuário malicioso possui da chave secreta utilizada no sistema de PLA, utilizando o algoritmo proposto, quando ele intercepta várias comunicações sem ruído.

4.1 SISTEMA CLÁSSICO DE GERAÇÃO DA *TAG*

Como discutido na Seção 2.2.1, toda vez que Alice deseja enviar uma mensagem $\mathbf{s}$ para Bob, ela envia em conjunto uma *tag* $\mathbf{t}$, que é gerada a partir de $\mathbf{s}$ e de uma chave secreta $\mathbf{k}$ compartilhada por Alice e Bob. O sistema descrito nesta seção é o que será chamado de clássico, e está em diversos artigos da literatura (XIE; CHEN; MING, 2021; YU; BARAS; SADLER, 2008; PERAZZONE et al., 2018; YU; VERMA; SADLER, 2015; YU; SADLER, 2011; QAISI; ALTHUNIBAT; QARAQE, 2022; XIE et al., 2022; MAENG et al., 2022; XIE et al., 2023; GU et al., 2020). No sistema clássico, a *tag* é gerada utilizando uma função $g(\cdot)$, referida muitas vezes como uma função *hash*, unidirecional *difícil* de inverter,

$$g(\mathbf{s}, \mathbf{k}) = \mathbf{t}. \tag{4.1}$$

Os conjuntos de chaves, mensagens e *tags* possíveis são denotados por $\mathcal{K}$, $\mathcal{S}$, e $\mathcal{T}$, respectivamente. Quando $|\mathcal{K}| \leq |\mathcal{T}|$, assume-se que, fixada a mensagem $\mathbf{s}$, a função $g(\cdot)$ mapeia cada chave em uma *tag* distinta. Desta maneira, como assume-se que a atacante Eva recupera $\mathbf{s}$ sem erros e também tem acesso ao conjunto $\mathcal{K}$, ela pode enumerar todas as *tags* possíveis:

$$\mathbf{t}_i = g(\mathbf{s}, \mathbf{k}_i), \quad \forall\, \mathbf{k}_i \in \mathcal{K}, \tag{4.2}$$

e, consequentemente, se Eva consegue obter a *tag* transmitida sem ruído, um ataque de força bruta sobre a chave tem garantia de sucesso, assumindo que a atacante tem um poder computacional

ilimitado. Portanto, no caso sem ruído, tem-se $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = 0$. A segurança deste esquema está unicamente baseada no ruído sobre a *tag* que é inserida pelo canal durante a transmissão.

4.2 GERAÇÃO DA *TAG* UTILIZANDO SEQUÊNCIAS CAÓTICAS DISCRETIZADAS

O método de geração da *tag* proposto nesta seção envolve o uso de sequências caóticas discretizadas. Inicialmente, as características do mapa e o processo de discretização de suas órbitas são abordadas na Subseção 4.2.1. O método de geração da *tag* é detalhado na Subseção 4.2.2. Uma análise da equivocação incondicional da chave para este método é realizada na Subseção 4.2.3.

4.2.1 Discretização das órbitas do mapa caótico

Considere um mapa caótico com pré-imagens binárias constantes representado pela função $f(\cdot)$, como o definido na Subseção 2.3.3. Para um y pertencente ao domínio A do mapa, considere os conjuntos $\mathcal{S}_1(y), \mathcal{S}_2(y), \ldots, \mathcal{S}_K(y)$, em que $\mathcal{S}_i(y)$ contém as i-ésimas pré-imagens de y no mapa, com $|S_i(y)| = 2^i$. A forma como estes subconjuntos iteram sobre o mapa pode ser representada como uma árvore binária, pois vai existir sempre um par de elementos presentes no conjunto $\mathcal{S}_{i+1}$ que mapeiam em um mesmo elemento do conjunto $\mathcal{S}_i$, semelhante ao exemplo ilustrado na Fig. 7.

Para um dado y, define-se o conjunto $\mathcal{X}_i$[1] como $\mathcal{X}_i = \mathcal{S}_{K-i}(y)$ $(0 \leq i < K)$, e, portanto, como $|\mathcal{X}_i| = 2^{K-i}$, pode-se reescrever $\mathcal{X}_i$ enumerando seus elementos,

$$\mathcal{X}_i = \{x_i^0, x_i^1, \ldots, x_i^{2^{K-i}-1}\}, \quad 0 \leq i < K. \tag{4.3}$$

O conjunto $\mathcal{X}_0$ é chamado de conjunto de condições iniciais do mapa. Uma condição inicial $x[0] \in \mathcal{X}_0$ possui uma órbita $\mathcal{O}$ sobre o mapa definida em (2.10). Existem $|\mathcal{X}_0| = 2^K$ condições iniciais do mapa, e como a iteração sobre as mesmas pode ser vista no formato de uma árvore binária, sabe-se que as órbitas associadas a estas condições iniciais distintas encontram-se em algum momento até a iteração K do mapa. A órbita discretizada $\mathcal{O}_d$ de $x[0]$ é obtida pela discretização dos elementos de $\mathcal{O}$ por uma função de discretização $d(\cdot)$, de forma que

$$\mathcal{O}_d = \begin{bmatrix} d(x[0]) & d(x[1]) & \cdots & d(x[K-1]) \end{bmatrix}. \tag{4.4}$$

A função de discretização $d : A \to \{-1, +1\}$ mapeia pontos do domínio do mapa nos valores -1 ou $+1$. Assume-se também que $d(\cdot)$ é uma função com a propriedade

$$d(x_i^m) \neq d(x_i^n) \text{ se } f(x_i^m) = f(x_i^n), \tag{4.5}$$

ou seja, se dois elementos x_i^m e x_i^n são mapeados no mesmo ponto pelo mapa, seus valores de discretização são distintos. A função de discretização garante que órbitas discretizadas com

[1] A análise realizada sobre os conjuntos $\mathcal{X}_i$ é válida para qualquer valor de y que garanta pré-imagens binárias constantes no mapa.

condições iniciais distintas são distintas, pois na iteração anterior ao encontro das órbitas elas possuem valores diferentes de discretização, como descrito em (4.5). Consequentemente, cada condição inicial gera uma órbita discretizada distinta $\mathcal{O}_d$ pertencente no conjunto $\{-1,+1\}^K$.

Para ilustrar o processo de discretização, considere o mapa da tenda definido em (2.11). Quando duas condições distintas x_i^m e x_i^n são mapeadas no mesmo ponto pelo mapa, tem-se

$$\begin{aligned} 2x_i^m + 1 &= 1 - 2x_i^n \qquad (4.6)\\ x_i^m &= -x_i^n. \qquad (4.7)\end{aligned}$$

Uma possível função de discretização que satisfaz a propriedade (4.5) é a função sinal, e, portanto, $d(\cdot)$ pode ser definida como

$$d(x) = \begin{cases} +1, & \text{se } x \geq 0 \\ -1, & \text{se } x < 0. \end{cases} \qquad (4.8)$$

Considere as três pré-imagens do ponto $y = 0$ no mapa da tenda, como calculado em (2.21). Têm-se que $\mathcal{X}_0 = \mathcal{S}_3(0)$, $\mathcal{X}_1 = \mathcal{S}_2(0)$ e $\mathcal{X}_2 = \mathcal{S}_1(0)$. A Fig. 7 ilustra como os elementos desses conjuntos podem ser organizados como uma árvore binária quando iterados pelo mapa da tenda. A aplicação da função de discretização (4.8) nos nós desta árvore é ilustrada na Fig. 12. Cada nó da árvore possui dois filhos, cada um com um valor de discretização distinto, como esperado por (4.5). As órbitas e as órbitas discretizadas associadas a cada condição inicial deste exemplo são

$$\begin{aligned}
-7/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} -7/8 & -3/4 & -1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} -1 & -1 & -1 \end{bmatrix},\\
+7/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} +7/8 & -3/4 & -1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} +1 & -1 & -1 \end{bmatrix},\\
-1/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} -1/8 & +3/4 & -1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} -1 & +1 & -1 \end{bmatrix},\\
+1/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} +1/8 & +3/4 & -1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} +1 & +1 & -1 \end{bmatrix},\\
-5/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} -5/8 & -1/4 & +1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} -1 & -1 & +1 \end{bmatrix},\\
+5/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} +5/8 & -1/4 & +1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} +1 & -1 & +1 \end{bmatrix},\\
-3/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} -3/8 & +1/4 & +1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} -1 & +1 & +1 \end{bmatrix},\\
+3/8 &\xrightarrow{\mathcal{O}} \begin{bmatrix} +3/8 & +1/4 & +1/2 \end{bmatrix} \xrightarrow{d} \begin{bmatrix} +1 & +1 & +1 \end{bmatrix}.
\end{aligned} \qquad (4.9)$$

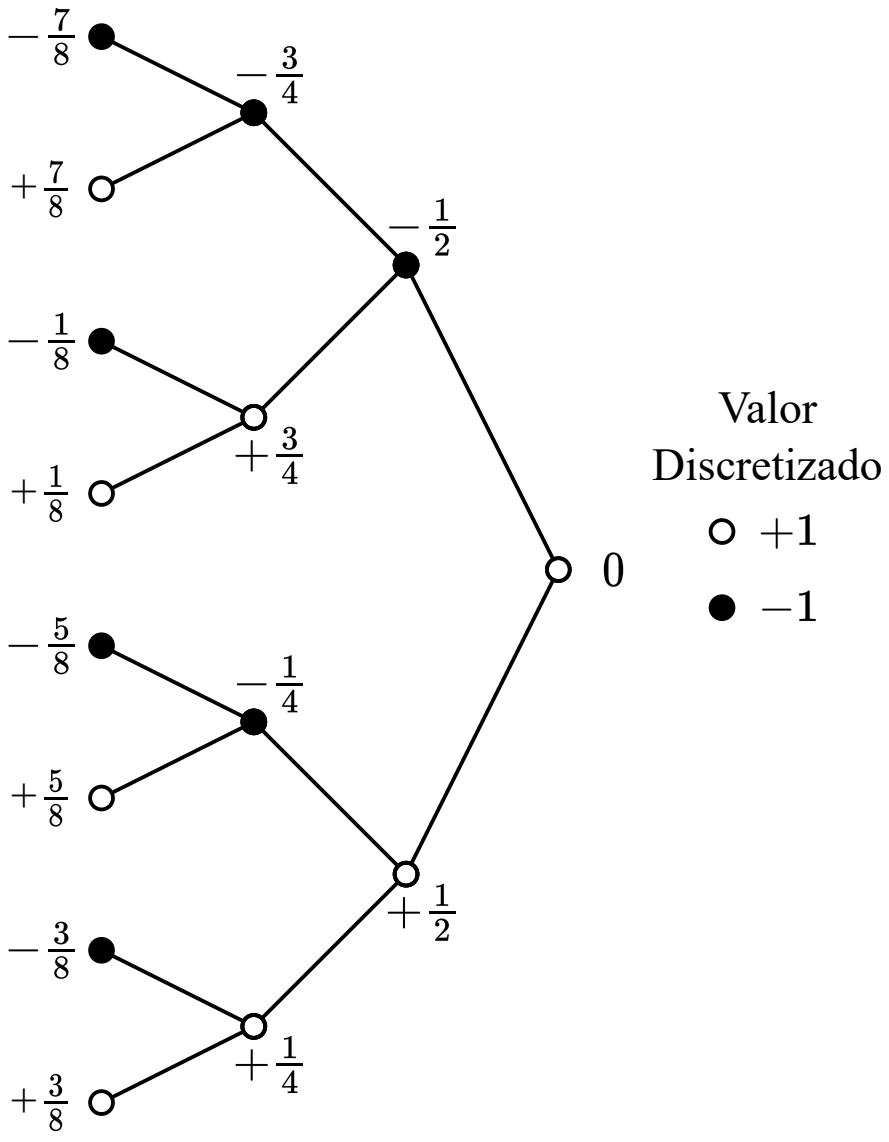
Cada condição inicial possui uma órbita discretizada distinta pertencente ao conjunto $\{-1,+1\}^3$.

4.2.2 Geração da *tag* a partir das órbitas discretizadas

A geração da *tag* depende da mensagem $\mathbf{s}$ transmitida e da chave $\mathbf{k}$ compartilhada entre os usuários legítimos. Os valores $\mathbf{s}$ e $\mathbf{k}$ são mapeados nas condições iniciais $\mathcal{X}_0$ do mapa utilizando

$$M(\mathbf{k} \oplus \mathbf{s}) = x[0], \quad x[0] \in \mathcal{X}_0, \qquad (4.10)$$

Figura 12 – Árvore binária dos elementos presentes em $\mathcal{S}_1(0), \mathcal{S}_2(0)$ e $\mathcal{S}_3(0)$, para o mapa da tenda, com seus respectivos valores discretizados.



Fonte: O autor (2023).

em que $M(\cdot)$ é uma função bijetiva e $\oplus$ é a operação XOR entre os vetores binários $\mathbf{s}$ e $\mathbf{k}$. A *tag* $\mathbf{t}$ é obtida desconsiderando os σ primeiros valores da órbita discretizada $\mathcal{O}_d$ associada a $x[0]$, e considerando os L valores seguintes, sendo definida como

$$\mathbf{t} = \begin{bmatrix} d(x[\sigma]) & d(x[\sigma+1]) & \cdots & d(x[\sigma+L-1]) \end{bmatrix}. \tag{4.11}$$

A *tag* é então uma fatia de comprimento L da órbita discretizada $\mathcal{O}_d$. Como cada uma das 2^K condições iniciais possui uma órbita discretizada distinta $\mathcal{O}_d \in \{-1,+1\}^K$, existem 2^{K-L} condições iniciais que geram a mesma *tag*. O conjunto de *tags* possíveis é $\mathcal{T} = \{-1,+1\}^L$. Como são feitas $\sigma + L - 1$ iterações, e o máximo número de iterações antes de todas as órbitas convergirem é $K-1$, existe a restrição

$$\sigma + L \leq K. \tag{4.12}$$

O procedimento de gerar uma *tag* a partir de uma condição inicial é descrito em termos da função $T : \mathcal{X}_0 \to \mathcal{T}$, de forma que

$$T(x[0]) = \begin{bmatrix} d(x[\sigma]) & d(x[\sigma+1]) & \cdots & d(x[\sigma+L-1]) \end{bmatrix}, \quad x[0] \in \mathcal{X}_0. \tag{4.13}$$

A obtenção de uma *tag* a partir dos valores de $\mathbf{s}$ e $\mathbf{k}$ é representada em termos da composição $F = T \circ M$, e, portanto, o processo de geração da *tag* associada a um par $\mathbf{s}$ e $\mathbf{k}$ é descrita como

$$F(\mathbf{k} \oplus \mathbf{s}) = T(M(\mathbf{k} \oplus \mathbf{s})) = \mathbf{t}. \tag{4.14}$$

Para exemplificar como funciona o processo de discretização, considere as condições inicias em (4.15) e a função de discretização em (4.8). Escolhe-se $\sigma = 1$ e $L = 2$, de forma que a *tag* associada a uma condição inicial representa os dois últimos valores da sua órbita discretizada. No exemplo considerado, as *tags* associadas a cada condição inicial são

$$\begin{aligned}
-7/8 &\xrightarrow{T} \begin{bmatrix} -1 & -1 \end{bmatrix}, \\
+7/8 &\xrightarrow{T} \begin{bmatrix} -1 & -1 \end{bmatrix}, \\
-1/8 &\xrightarrow{T} \begin{bmatrix} +1 & -1 \end{bmatrix}, \\
+1/8 &\xrightarrow{T} \begin{bmatrix} +1 & -1 \end{bmatrix}, \\
-5/8 &\xrightarrow{T} \begin{bmatrix} -1 & +1 \end{bmatrix}, \\
+5/8 &\xrightarrow{T} \begin{bmatrix} -1 & +1 \end{bmatrix}, \\
-3/8 &\xrightarrow{T} \begin{bmatrix} +1 & +1 \end{bmatrix}, \\
+3/8 &\xrightarrow{T} \begin{bmatrix} +1 & +1 \end{bmatrix}.
\end{aligned} \tag{4.15}$$

Note que uma mesma *tag* é gerada por $2^{K-L} = 2^{3-2} = 2$ condições iniciais distintas.

4.2.3 Equivocação incondicional da chave para o sistema com *tags* caóticas discretizadas

Como visto na Subseção 2.2.2, a equivocação da chave $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$ mede a quantidade de informação que a atacante Eva tem sobre a chave secreta $\mathbf{k}$, dado que ela intercepta um par legítimo $(\mathbf{s}, \mathbf{t})$. Para o cálculo deste valor, considere inicialmente $H(x[0], \mathbf{k} \mid \mathbf{s}, \mathbf{t})$ e a regra da cadeia para a entropia,

$$\begin{aligned} H(x[0], \mathbf{k} \mid \mathbf{s}, \mathbf{t}) &= H(x[0] \mid \mathbf{k}, \mathbf{s}, \mathbf{t}) + H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) && (4.16) \\ &= H(\mathbf{k} \mid x[0], \mathbf{s}, \mathbf{t}) + H(x[0] \mid \mathbf{s}, \mathbf{t}). && (4.17) \end{aligned}$$

Como $M(\cdot)$ em (4.10) é injetiva, então $H(x[0] \mid \mathbf{k}, \mathbf{s}, \mathbf{t}) = 0$ e $H(\mathbf{k} \mid x[0], \mathbf{s}, \mathbf{t}) = 0$, e portanto

$$H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = H(x[0] \mid \mathbf{s}, \mathbf{t}). \tag{4.18}$$

Analogamente, considera-se agora $H(x[0], \mathbf{s} \mid \mathbf{t})$,

$$\begin{aligned} H(x[0], \mathbf{s} \mid \mathbf{t}) &= H(x[0] \mid \mathbf{s}, \mathbf{t}) + H(\mathbf{s} \mid \mathbf{t}) && (4.19) \\ &= H(\mathbf{s} \mid x[0], \mathbf{t}) + H(x[0] \mid \mathbf{t}). && (4.20) \end{aligned}$$

Como $\mathbf{s}$ and $\mathbf{k}$ são sequências binárias de mesmo comprimento e se considera a distribuição de probabilidade de $\mathbf{k}$ uniforme, então $H(\mathbf{k}) \geq H(\mathbf{s})$. Tendo em vista (4.10), isto implica que para qualquer par $x[0]$ e $\mathbf{s}$ existe uma chave $\mathbf{k}'$, tal que, $x[0] = M(\mathbf{s} \oplus \mathbf{k}')$. Consequentemente, se $\mathbf{k}$ é desconhecida, o conhecimento de $x[0]$ não dá nenhuma informação sobre $\mathbf{s}$. Como $\mathbf{s}$ é independente de $x[0]$ e de $\mathbf{t}$, têm-se que $H(\mathbf{s} \mid \mathbf{t}) = H(\mathbf{s})$ e $H(\mathbf{s} \mid x[0], \mathbf{t}) = H(\mathbf{s})$. Então, $H(x[0] \mid \mathbf{s}, \mathbf{t}) = H(x[0] \mid \mathbf{t})$, que em conjunto com (4.18) permite obter a igualdade

$$H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = H(x[0] \mid \mathbf{t}). \tag{4.21}$$

Para calcular a equivocação de $x[0]$ em relação a $\mathbf{t}$, considere a partição do espaço de condições iniciais $\mathcal{X}_0$ em 2^L conjuntos $\mathcal{X}_0^i$, de forma que

$$\mathcal{X}_0 = \mathcal{X}_0^0 \cup \mathcal{X}_0^1 \cup \cdots \cup \mathcal{X}_0^{2^L-1}, \quad \mathcal{X}_0^i \cap \mathcal{X}_0^j = \emptyset, \, \forall \, i \neq j \tag{4.22}$$

e

$$\mathcal{X}_0^i = \{x[0] \mid T(x[0]) = \mathbf{t}_i, x[0] \in \mathcal{X}_0\}, \quad \mathbf{t}_i \in \mathcal{T}, \, 0 \leq i \leq 2^L - 1, \tag{4.23}$$

ou seja, $\mathcal{X}_0^i$ é formado por todas as condições inicias que geram a *tag* $\mathbf{t}_i$, logo, $|\mathcal{X}_0^i| = 2^{K-L}$. Em virtude da distribuição de $\mathbf{k}$ ser uniforme, (4.10) garante que a distribuição de $x[0]$ também será uniforme. Como o número de condições iniciais que gera cada *tag* também é constante, a distribuição de $\mathbf{t}$ também é uniforme. A probabilidade condicional de uma condição inicial $x[0]$ qualquer gerar uma *tag* $\mathbf{t}_i$ é dada por

$$P(x[0] \mid \mathbf{t}_i) = \begin{cases} \frac{1}{2^{K-L}} & , \quad \text{se } x[0] \in \mathcal{X}_0^i \\ 0 & , \quad \text{c.c.} \end{cases} \tag{4.24}$$

A equivocação $H(x[0] \mid \mathbf{t})$ é dada por

$$\begin{aligned} H(x[0] \mid \mathbf{t}) &= \mathbb{E}[-\log_2 \Pr(x[0] \mid \mathbf{t})] && (4.25) \\ &= -\sum_{i=0}^{2^L-1} \Pr(\mathbf{t}_i) \sum_{j=0}^{2^K-1} \Pr(x_0^j|\mathbf{t}_i) \log_2 \Pr(x_0^j|\mathbf{t}_i) && (4.26) \\ &= K - L, && (4.27) \end{aligned}$$

Conclui-se a partir de (4.21) que a equivocação da chave para o método de geração da *tag*, utilizando sequências caóticas discretizadas, é dada por

$$H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = K - L, \tag{4.28}$$

e assim, uma segurança incondicional positiva é garantida por este método de geração da *tag*. Já que L é positivo e obedece (4.12), pode-se escolher L no intervalo

$$1 \leq L \leq K - \sigma. \tag{4.29}$$

4.3 COMPARAÇÃO ENTRE OS SISTEMAS DE GERAÇÃO DA *TAG*

Em uma transmissão real de um par $(\mathbf{s}, \mathbf{t})$, possíveis ruídos associados ao canal de comunicação afetam o conhecimento exato da *tag*. Dessa forma, a atacante Eva precisa utilizar a *tag* ruidosa recebida para criar uma estimativa $\hat{\mathbf{t}}$ da *tag* transmitida. Quando se trabalha com *tags* com dígitos binários, Eva pode calcular a distância de Hamming $h(\hat{\mathbf{t}}, \mathbf{t}_j)$ para cada *tag* $\mathbf{t}_j$ possível. A probabilidade de uma *tag* $\mathbf{t}_j$ ter sido utilizada na transmissão, dado $\hat{\mathbf{t}}$ (considerando que o erro entre cada componente de $\mathbf{t}_j$ e $\hat{\mathbf{t}}$ são variáveis independentes e identicamente distribuídas com probabilidade p_e), é então modelada por uma distribuição binomial (YU; SADLER, 2011)

$$\Pr(\mathbf{t}_j \mid \hat{\mathbf{t}}) = p_e^{h(\hat{\mathbf{t}}, \mathbf{t}_j)} (1 - p_e)^{L - h(\hat{\mathbf{t}}, \mathbf{t}_j)}. \tag{4.30}$$

A probabilidade de erro p_e depende do canal de comunicação considerado. Aqui considera-se um canal de comunicação com ruído aditivo Gaussiano de média nula e variância Γ. Para este canal, a probabilidade de erro é dada por $p_e = Q(\sqrt{\text{TNR}})$, no qual $Q(\cdot)$ é a probabilidade da cauda de uma distribuição Gaussiana e TNR é a relação sinal-ruído da *tag*. Como cada componente da *tag* pertence a $\{-1, +1\}$, a TNR é definida por

$$\text{TNR} = \frac{1}{\Gamma}. \tag{4.31}$$

A probabilidade em (4.30) é usada nos cálculos das métricas associadas à teoria da informação analisadas nas Subseções 4.3.1 e 4.3.2.

4.3.1 Equivocação da chave para os sistemas de geração da *tag*

O conhecimento que a atacante Eva tem sobre a chave $\mathbf{k}$ dada uma observação considerando ruídos pode ser medido pela equivocação $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}})$. Para obter uma expressão para essa equivocação, considera-se inicialmente a equivocação $H(\mathbf{k}, \mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}})$, que é escrita como

$$\begin{aligned} H(\mathbf{k}, \mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}}) &= H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}}, \mathbf{t}) + H(\mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}}) && (4.32) \\ &= H(\mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}}, \mathbf{k}) + H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}}), && (4.33) \end{aligned}$$

em que pode-se escrever $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}}, \mathbf{t}) = H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$, pois $\hat{\mathbf{t}}$ não dá nenhuma informação nova quando se tem $\mathbf{t}$. Observe que $H(\mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}}) = H(\mathbf{t} \mid \hat{\mathbf{t}})$, já que o conhecimento de $\mathbf{s}$ sem $\mathbf{k}$ não ajuda na identificação de $\mathbf{t}$, e $H(\mathbf{t} \mid \mathbf{s}, \hat{\mathbf{t}}, \mathbf{k}) = 0$, uma vez que o conhecimento de $\mathbf{k}$ e $\mathbf{s}$ define completamente o valor de $\mathbf{t}$. Usando a igualdade entre (4.32) e (4.33) com essas simplificações, reescreve-se $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}})$ como

$$H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}}) = H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) + H(\mathbf{t} \mid \hat{\mathbf{t}}), \tag{4.34}$$

na qual $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$ representa a parcela incondicional da equivocação (independente do ruído), e $H(\mathbf{t} \mid \hat{\mathbf{t}})$ representa a parcela da equivocação associada ao ruído, dada por

$$\begin{aligned} H(\mathbf{t} \mid \hat{\mathbf{t}}) &= \mathbb{E}[-\log_2 \Pr(\mathbf{t}_j \mid \hat{\mathbf{t}})] && (4.35) \\ &= -\sum_j \Pr(\mathbf{t}_j \mid \hat{\mathbf{t}}) \log_2 \Pr(\mathbf{t}_j \mid \hat{\mathbf{t}}), && (4.36) \end{aligned}$$

em que $\Pr(\mathbf{t}_j \mid \hat{\mathbf{t}})$ é dada por (4.30). Na Seção 4.2, foi observado que no sistema que emprega sequências caóticas discretizadas, o conjunto de *tags* possíveis em uma transmissão, representado por $\{\mathbf{t}_j\}$, é idêntico ao espaço total de *tags*, denominado $\mathcal{T}$. Já no sistema clássico, em que existe uma correspondência única entre uma chave e uma *tag*, dada uma mensagem $\mathbf{s}$, só tem-se que $\{\mathbf{t}_j\} = \mathcal{T}$ quando $|\mathcal{K}| = |\mathcal{T}|$. Considerando o caso em que $\{\mathbf{t}_j\} = \mathcal{T}$, no qual $\mathcal{T}$ representa todas as *tags* binárias de comprimento L, $\mathcal{T} = \{-1, +1\}^L$, pode-se agrupar as *tags* $\mathbf{t}_j$ que possuem mesmo valor de distância de Hamming $h(\hat{\mathbf{t}}, \mathbf{t}_j)$ em (4.36). As probabilidades em (4.30) são representadas pela função $\Psi(\cdot)$ definida por

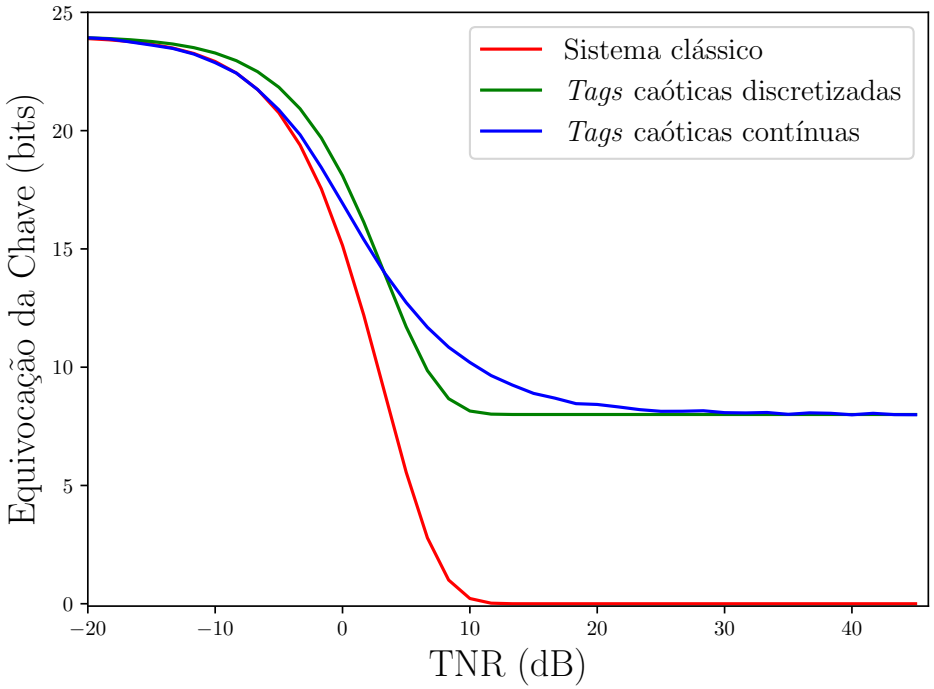
$$\Psi(p_e, L, n) = p_e^n (1 - p_e)^{L-n}, \tag{4.37}$$

e a equivocação $H(\mathbf{t} \mid \hat{\mathbf{t}})$ é reescrita da seguinte forma:

$$H(\mathbf{t} \mid \hat{\mathbf{t}}) = -\sum_{n=0}^{L} \binom{L}{n} \Psi(p_e, L, n) \log_2 \Psi(p_e, L, n). \tag{4.38}$$

Para exemplificar o comportamento das curvas da equivocação da chave em diferentes esquemas PLA que usam *tags*, considera-se o sistema que usa sequências caóticas discretizadas com $K = 24$ e $L = 16$, além do sistema clássico, apresentado na Seção 4.1, com $K = L = 24$, pois existe aqui a restrição de que $|\mathcal{T}| \geq |\mathcal{K}|$, então é mantido o valor de $K = 24$ e se usa o

Figura 13 – Comparação da equivocação da chave entre o sistema clássico, com $K = L = 24$, e os sistemas de geração da *tag* que utilizam sequências caóticas, com $K = 24$ e $L = 16$.



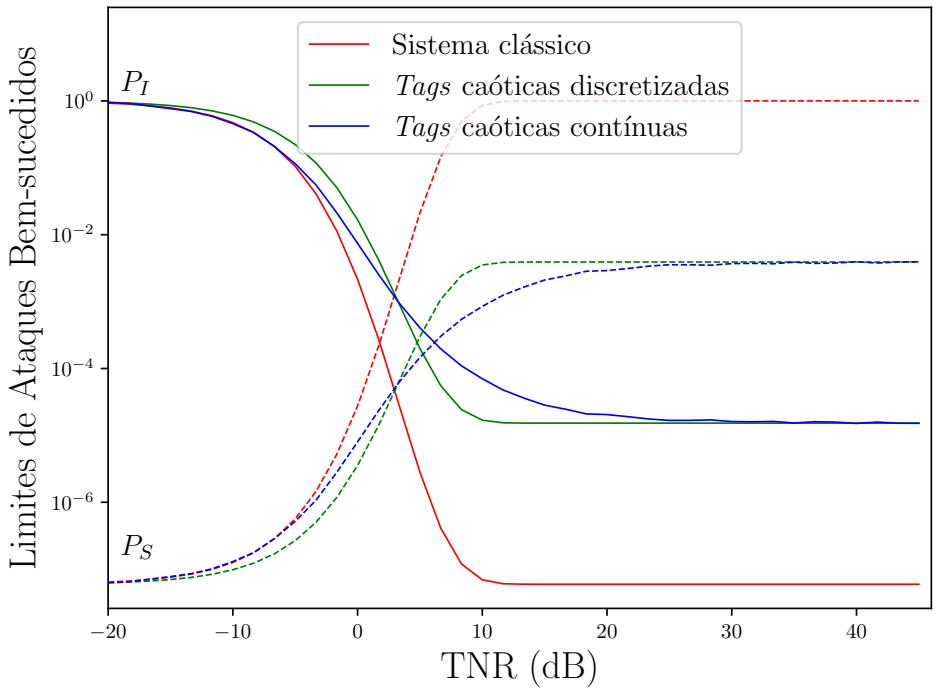
Fonte: O autor (2023)

menor valor possível de L. O cálculo da equivocação em ambos os sistemas segue (4.34), em que $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$ é a equivocação incondicional de cada sistema e $H(\mathbf{t} \mid \hat{\mathbf{t}})$ é dado em (4.38). Para o sistema clássico, a equivocação incondicional é $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t}) = 0$. Já para o sistema que gera *tags* utilizando sequências caóticas discretizadas, a equivocação incondicional é obtida em (4.28) como $K - L$. Também calcula-se a equivocação para um sistema de geração da *tag* a partir de sequências caóticas contínuas apresentado em (EVANGELISTA, 2016), com os parâmetros $K = 24$ e $L = 16$. As curvas de equivocação em função da TNR para cada um desses sistemas estão mostradas na Fig. 13. Enquanto o sistema clássico não possui limiar inferior positivo, e se aproxima de zero para TNR alta, os dois sistemas que utilizam sequências caóticas possuem um limiar inferior em $K - L = 8$. Existe um *tradeoff* entre os dois sistemas que utilizam sequências caóticas. Para TNR $\lesssim 3{,}21$ dB, o sistema discretizado apresenta comportamento melhor. Já para valores de TNR maiores que $3{,}21$ dB, o sistema com *tag* caóticas contínuas possui melhor comportamento, até que as curvas dos dois sistemas se encontram em TNR ≈ 25 dB.

4.3.2 Probabilidades de ataques para os sistemas de geração da *tag*

Os limiares das probabilidades de ataques de substituição (P_S) e personificação (P_I) são obtidos diretamente dos resultados de equivocação apresentados na Fig. 13, bastando apenas aplicar os valores obtidos para $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}})$ em (2.4) e (2.8). Esses limiares estão ilustrados na Fig. 14. É importante notar que os limiares de P_S e P_I se cruzam quando $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}}) = H(\mathbf{k})/2$, ou seja, quando $P_S = P_I = 2^{-H(\mathbf{k})/2} = 2^{-12}$, pois $H(\mathbf{k}) = K = 24$, dado que as chaves são

Figura 14 – Comparação dos limites das probabilidades de ataques bem-sucedidos entre o sistema clássico, com $K = L = 24$, e os sistemas de geração da *tag* que utilizam sequências caóticas, com $K = 24$ e $L = 16$. Os limiares associados a um ataque de personificação (P_I) e substituição (P_S) são representados em linhas cheias e pontilhadas, respectivamente.



Fonte: O autor (2023)

equiprováveis. Esse limiar é verificado ao igualar (2.4) e (2.8). Além disso, o efeito de um limiar inferior positivo para a equivocação nos sistemas que envolvem sequências caóticas implica em um limiar superior para P_S, dado por $P_S \approx 2^{-(K-L)} = 2^{-8}$, e um limiar inferior para P_I, dado por $P_I \approx 2^{-K+(K-L)} = 2^{-16}$. Ambos os limiares são derivados de (2.4) e (2.8) quando se considera uma alta TNR e pode-se aproximar o valor de $H(\mathbf{k} \mid \mathbf{s}, \hat{\mathbf{t}})$ por $H(\mathbf{k} \mid \mathbf{s}, \mathbf{t})$.

4.4 O PROBLEMA DAS MÚLTIPLAS OBSERVAÇÕES

A equivocação incondicional da chave analisada até então considera que a atacante Eva consegue obter um par legítimo de mensagem $\mathbf{s}$ e *tag* $\mathbf{t}$, medindo a quantidade de informação que este par fornece sobre a chave $\mathbf{k}$. Quando se considera que a atacante tem acesso a N pares legítimos $(\mathbf{s}^i, \mathbf{t}^i)$, a análise da informação que o conhecimento destes pares fornece sobre $\mathbf{k}$ é denominada nesta seção de problema das múltiplas observações. Nesta seção, o problema das múltiplas observações é analisado para a técnica de geração da *tag* utilizando sequências caóticas discretizadas, que foi apresentada na Seção 4.2. Toda a análise realizada considera que a atacante tem acesso a *tags* legítimas, isto é, se considera que ela não sofre com o ruído presente no canal de comunicação. Esta abordagem permite que o foco esteja somente em como o algoritmo de geração da *tag* proposto influencia o problema das múltiplas observações.

4.4.1 Descrição do problema das múltiplas observações

Suponha que a chave legítima $\mathbf{k}_{\text{leg}}$ é utilizada para transmitir N mensagens $\mathbf{s}^1, \mathbf{s}^2, \ldots, \mathbf{s}^N$. Cada $\mathbf{s}^i$ pode assumir algum valor de $\mathcal{S}$ com igual probabilidade. A cada mensagem $\mathbf{s}^i$ transmitida, a *tag* $\mathbf{t}^i = F(\mathbf{k}_{\text{leg}} \oplus \mathbf{s}^i)$ é associada a essa mensagem. Já que existem 2^{K-L} valores distintos $\mathbf{k} \oplus \mathbf{s}^i$ que são mapeados pela função $F(\cdot)$ em $\mathbf{t}^i$, então existem 2^{K-L} chaves distintas que podem ter sido utilizadas para transmitir um par $(\mathbf{s}^i, \mathbf{t}^i)$. Define-se o conjunto de chaves que podem ter sido utilizadas para transmitir o par $(\mathbf{s}^i, \mathbf{t}^i)$ como

$$\mathcal{K}^i = \{\mathbf{k}_j \mid F(\mathbf{k}_j \oplus \mathbf{s}^i) = \mathbf{t}^i, \mathbf{k}_j \in \mathcal{K}\}, \tag{4.39}$$

em que $|\mathcal{K}^i| = 2^{K-L}$. Para cada uma das N mensagens transmitidas, pode-se construir um subconjunto como este a partir do par $(\mathbf{s}^i, \mathbf{t}^i)$ transmitido. Têm-se então os subconjuntos $\mathcal{K}^1, \mathcal{K}^2, \ldots, \mathcal{K}^N$, cada um associado a um dos N pares enviados. Como a chave $\mathbf{k}_{\text{leg}}$ foi a chave utilizada para transmitir todos os N pares, então obrigatoriamente tem-se que $\mathbf{k}_{\text{leg}} \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N$. O problema das múltiplas observações consiste em calcular o conhecimento que a atacante Eva tem sobre a chave $\mathbf{k}_{\text{leg}}$, supondo que ela tem acesso aos N pares transmitidos, todos utilizando a mesma chave $\mathbf{k}_{\text{leg}}$. Quando se considera apenas uma observação, a atacante pode construir $\mathcal{K}^1$, e a chave legítima $\mathbf{k}_{\text{leg}}$ é uma dentre as $|\mathcal{K}^1| = 2^{K-L}$ chaves deste conjunto. Para N observações, a atacante pode construir os N subconjuntos $\mathcal{K}^1, \mathcal{K}^2, \ldots, \mathcal{K}^N$, e $\mathbf{k}_{\text{leg}}$ será uma dentre as $|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|$ chaves possíveis. Uma forma de estimar a informação que a atacante tem sobre a chave legítima é então calculando

$$\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]. \tag{4.40}$$

Uma outra forma de medir a informação da atacante sobre a chave legítima é através da equivocação em relação à uma chave $\mathbf{k}$, dado um conjunto de N observações de pares $(\mathbf{s}^i, \mathbf{t}^i)$. Pode-se estender o raciocínio de (2.2) para expressar a equivocação no caso de múltiplas observações como $H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$. Utilizando a definição da função $H(\cdot)$, escreve-se

$$H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N) = \mathbb{E}[-\log_2 \Pr(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)], \tag{4.41}$$

em que $\Pr(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$ representa a probabilidade de alguma chave $\mathbf{k}$ ser a legítima, dados os N pares observados. Para cada par $(\mathbf{s}^i, \mathbf{t}^i)$, obtém-se a partir de (4.39) o conjunto $\mathcal{K}^i$ de chaves possíveis associadas a i-ésima observação. Pode-se então reescrever a probabilidade $\Pr(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$ como $\Pr(\mathbf{k} \mid \mathcal{K}^1, \mathcal{K}^2, \ldots, \mathcal{K}^N)$. De acordo com o que foi visto anteriormente, a chave legítima $\mathbf{k}_{\text{leg}}$ pertence ao subconjunto $\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N$, e já que supõe-se que todas as chaves em $\mathcal{K}$ possuem a mesma probabilidade de serem escolhidas, cada chave no subconjunto também possui mesma probabilidade de ser escolhida. A construção do subconjunto $\mathcal{K}^i$ depende do valor específico que o par $(\mathbf{s}^i, \mathbf{t}^i)$ assume. Para alguma escolha dos N pares, digamos $(\mathbf{s}^{i'}, \mathbf{t}^{i'})$, suponha que os subconjuntos $\mathcal{K}^i$ assumem os valores $\mathcal{K}^i = \mathcal{K}^{i'}$. A probabilidade de uma chave $\mathbf{k}_j$ ser a legítima, dados os N pares, é então representada por $\Pr(\mathbf{k} =$

$\mathbf{k}_j \mid \mathcal{K}^1 = \mathcal{K}^{1'}, \ldots, \mathcal{K}^N = \mathcal{K}^{N'})$, que será aqui simplificada como $\Pr(\mathbf{k}_j \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'})$, e assume os valores

$$\Pr(\mathbf{k}_j \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) = \begin{cases} \frac{1}{|\mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'}|}, & \text{se } \mathbf{k}_j \in \mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'} \\ 0, & \text{c.c.} \end{cases} \tag{4.42}$$

A equivocação da chave para essa transmissão específica é então calculada como sendo

$$\begin{aligned} H(\mathbf{k} \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) &= \mathbb{E}[-\log_2 \Pr(\mathbf{k} \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'})] \\ &= -\sum_{\mathbf{k}_j \in \mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'}} \Pr(\mathbf{k}_j \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) \log_2 \Pr(\mathbf{k}_j \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) \\ &= \log_2 |\mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'}|. \end{aligned} \tag{4.43}$$

A equivocação $H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$ é então reescrita como

$$\begin{aligned} H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N) &= H(\mathbf{k} \mid \mathcal{K}^1, \ldots, \mathcal{K}^N) \\ &= \sum_{\mathcal{K}^{1'}} \cdots \sum_{\mathcal{K}^{N'}} \Pr(\mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) H(\mathbf{k} \mid \mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) \\ &= \sum_{\mathcal{K}^{1'}} \cdots \sum_{\mathcal{K}^{N'}} \Pr(\mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}) \log_2 |\mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'}|. \end{aligned} \tag{4.44}$$

Pode-se agrupar em (4.44) os conjuntos $\mathcal{K}^{1'}, \ldots, \mathcal{K}^{N'}$ que possuem o mesmo valor de $|\mathcal{K}^{1'} \cap \cdots \cap \mathcal{K}^{N'}|$. Sabe-se que, de uma forma geral, a variável aleatória $|\mathcal{K}^1 \cap \cdots \cap \mathcal{K}^N|$ varia de 1 (a interseção contém apenas a chave legítima) até 2^{K-L} (os subconjuntos são todos iguais), e assim, reescreve-se a equivocação para múltiplas observações como

$$H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N) = \sum_{m=1}^{2^{K-L}} \Pr(|\mathcal{K}^1 \cap \cdots \cap \mathcal{K}^N| = m) \log_2 m. \tag{4.45}$$

Note que, ao utilizar (4.45) para o caso de apenas uma observação, obtém-se que o único valor possível para m é 2^{K-L}, pois $|\mathcal{K}^1| = 2^{K-L}$. Consequentemente,

$$H(\mathbf{k} \mid \mathbf{s}^1, \mathbf{t}^1) = \sum_{m=1}^{2^{K-L}} \Pr(|\mathcal{K}^1| = m) \log_2 m \tag{4.46}$$

$$= \log_2 2^{K-L} \tag{4.47}$$

$$= K - L, \tag{4.48}$$

o que está de acordo com o resultado obtido na Subseção 4.2.3.

Assim como em (4.45), a estimativa do valor da equivocação para múltiplas observações requer o conhecimento de como o valor de $|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|$ se comporta durante a transmissão. Conforme observado em (4.39), a construção dos conjuntos $\mathcal{K}^i$ depende de como o mapeamento $F(\cdot)$ é definido. Portanto, alguns mapeamentos são considerados a seguir e as métricas definidas em (4.40) e (4.45) são utilizadas para analisá-los.

4.4.2 Mapeamento aleatório

Inicialmente considera-se o caso em que $F(\cdot)$ representa algum mapeamento aleatório. Assume-se que os pares $(\mathbf{s}^i, \mathbf{t}^i)$ são transmitidos utilizando a chave legítima $\mathbf{k}_{\text{leg}}$. As mensagens $\mathbf{s}^i$ são escolhidas de maneira aleatória sobre o conjunto $\mathcal{S}$, e, consequentemente, as *tags* $\mathbf{t}^i$ também são aleatoriamente escolhidas sobre $\mathcal{T}$. A probabilidade de uma chave $\mathbf{k}$ ter sido utilizada na transmissão do par $(\mathbf{s}^1, \mathbf{t}^1)$ é a probabilidade da mesma pertencer ao conjunto $\mathcal{K}^1$,

$$\begin{aligned} \Pr(\mathbf{k} \in \mathcal{K}^1) &= \Pr(F(\mathbf{k} \oplus \mathbf{s}^1) = F(\mathbf{k}_{\text{leg}} \oplus \mathbf{s}^1)) && (4.49) \\ &= \Pr(F(\mathbf{k} \oplus \mathbf{s}^1) = \mathbf{t}^1). && (4.50) \end{aligned}$$

Quando $\mathbf{k} = \mathbf{k}_{\text{leg}}$, a probabilidade é igual a 1. Como $F(\cdot)$ representa uma permutação aleatória, quando $\mathbf{k} \neq \mathbf{k}_{\text{leg}}$ a probabilidade é calculada como

$$\begin{aligned} \Pr(\mathbf{k} \in \mathcal{K}^1) &= \frac{|\{\mathbf{k} \mid F(\mathbf{k} \oplus \mathbf{s}^1) = \mathbf{t}^1, \mathbf{k} \in \mathcal{K} - \{\mathbf{k}_{\text{leg}}\}\}|}{|\mathcal{K} - \{\mathbf{k}_{\text{leg}}\}|} && (4.51) \\ &= \frac{2^{K-L} - 1}{2^K - 1}, \quad \forall\, \mathbf{k} \neq \mathbf{k}_{\text{leg}}. && (4.52) \end{aligned}$$

Tem-se então que

$$\Pr(\mathbf{k} \in \mathcal{K}^1) = \begin{cases} 1, & \text{se } \mathbf{k} = \mathbf{k}_{\text{leg}} \\ \frac{2^{K-L}-1}{2^K-1}, & \text{se } \mathbf{k} \neq \mathbf{k}_{\text{leg}}. \end{cases} \tag{4.53}$$

Ao considerar a transmissão de um outro par $(\mathbf{s}^2, \mathbf{t}^2)$, tem-se a probabilidade de uma chave $\mathbf{k}_j$ pertencer à interseção $\mathcal{K}^1 \cap \mathcal{K}^2$. Como as mensagens $\mathbf{s}^1$ e $\mathbf{s}^2$ são variáveis independentes e uniformemente distribuídas sobre $\mathcal{S}$, é possível escrever que $\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2) = \Pr(\mathbf{k} \in \mathcal{K}^1) \Pr(\mathbf{k} \in \mathcal{K}^2)$. De forma semelhante, pode-se estender o raciocínio para a transmissão de N pares, de forma que $\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N) = \Pr(\mathbf{k} \in \mathcal{K}^1) \Pr(\mathbf{k} \in \mathcal{K}^2) \cdots \Pr(\mathbf{k} \in \mathcal{K}^N)$, e assim

$$\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N) = \begin{cases} 1, & \text{se } \mathbf{k} = \mathbf{k}_{\text{leg}} \\ \left(\frac{2^{K-L}-1}{2^K-1}\right)^N, & \text{se } \mathbf{k} \neq \mathbf{k}_{\text{leg}}. \end{cases} \tag{4.54}$$

Define-se a variável aleatória A_j como sendo

$$A_j = \begin{cases} 1, & \text{se } \mathbf{k}_j \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N, \\ 0, & \text{caso contrário.} \end{cases} \tag{4.55}$$

Consequentemente, reescreve-se $|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|$ como

$$\begin{aligned} |\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N| &= \sum_{j=0}^{2^K-1} A_j && (4.56) \\ &= 1 + \sum_{\substack{j=0 \\ \mathbf{k}_j \neq \mathbf{k}_{\text{leg}}}}^{2^K-1} A_j, && (4.57) \end{aligned}$$

em que o fator 1 em (4.57) é devido à chave legítima, que é garantida de pertencer à interseção dos subconjuntos. Cada A_j é uma variável aleatória de Bernoulli com probabilidade de sucesso $\left(\frac{2^{K-L}-1}{2^K-1}\right)^N$. O cálculo de (4.40) pode então ser feito por meio de

$$\begin{aligned}\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|] &= \mathbb{E}\left[\sum_{j=0}^{2^K-1} A_j\right] && (4.58)\\ &= 1 + \mathbb{E}\left[\sum_{\substack{j=0\\ \mathbf{k}_j \neq \mathbf{k}_{\text{leg}}}}^{2^K-1} A_j\right] && (4.59)\\ &= 1 + (2^K-1)\left(\frac{2^{K-L}-1}{2^K-1}\right)^N. && (4.60)\end{aligned}$$

Se forem escolhidos valores de K e L tais que $2^{K-L} >> 1$, aproxima-se (4.60) por

$$\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|] \approx 1 + 2^{K-NL}. \tag{4.61}$$

Conclui-se então que para um mapeamento $F(\cdot)$ aleatório, $\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]$ decai rapidamente por um fator aproximado de 2^L a cada nova transmissão.

Para exemplificar como o resultado obtido se aplica a um mapeamento específico, considere a função $F(\cdot) = T \circ M$ definida em (4.14). Considere o particionamento do conjunto de condições iniciais $\mathcal{X}_0$ em 2^L subconjuntos $\mathcal{X}_0^i$ de tamanho 2^{K-L} tais que

$$\begin{aligned}\mathcal{X}_0^i &= \{x_0^{i2^{K-L}}, x_0^{i2^{K-L}+1}, \ldots, x_0^{i2^{K-L}+2^{K-L}-1}\} && (4.62)\\ &= \{x_0^{i2^{K-L}}, x_0^{i2^{K-L}+1}, \ldots, x_0^{(i+1)2^{K-L}-1}\}, \quad 0 \leq i < 2^L. && (4.63)\end{aligned}$$

Considera-se que o mapeamento $T(\cdot)$ é tal que

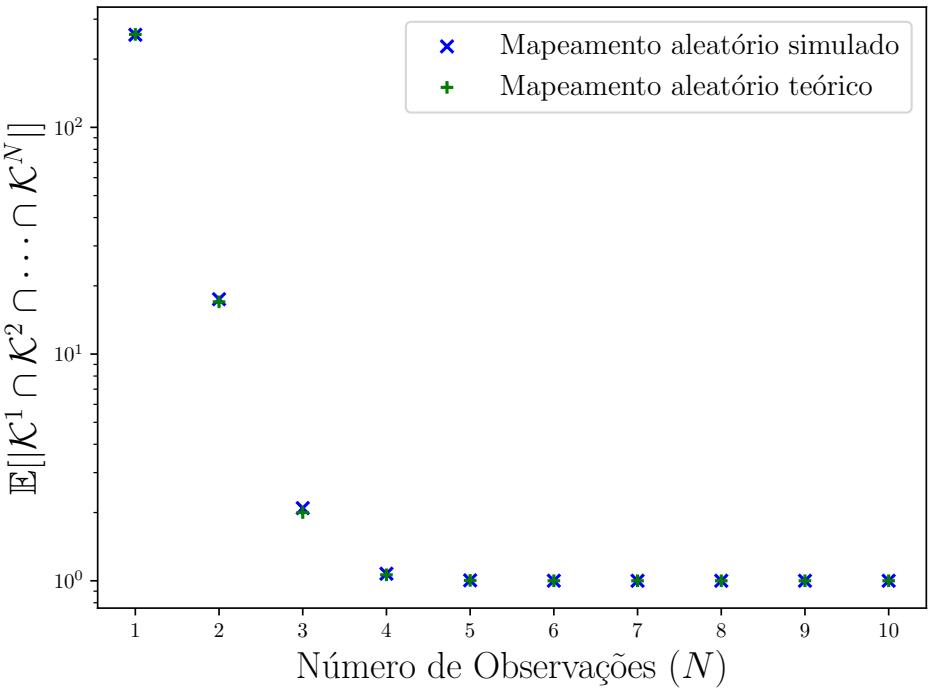
$$T(x[0]) = \mathbf{t}_i, \quad \forall\, x[0] \in \mathcal{X}_0^i,\, 0 \leq i < 2^L. \tag{4.64}$$

O mapeamento $M(\cdot)$ tem no seu domínio os 2^K valores possíveis de $\mathbf{k} \oplus \mathbf{s}$. Como $\mathbf{k} \oplus \mathbf{s}$ é um vetor binário, pode-se representar o valor associado a este vetor como um número inteiro em $\{0, 1, \ldots, 2^K - 1\}$. Já a imagem de $M(\cdot)$ são as 2^K condições iniciais pertencentes a $\mathcal{X}_0$, cada uma enumerada por um índice diferente pertencente a $\{0, 1, \ldots, 2^K - 1\}$. A função $M(\cdot)$ aqui considerada mapeia o valor decimal associado a $\mathbf{k} \oplus \mathbf{s}$ no índice associado a alguma condição inicial x_0^i, de forma que

$$M(\mathbf{k} \oplus \mathbf{s}) = x_0^{P(\mathbf{k} \oplus \mathbf{s})}, \tag{4.65}$$

em que $P(\cdot)$ representa alguma permutação dos inteiros no intervalo $\{0, 1, \ldots, 2^K - 1\}$. A Fig. 15 ilustra os valores de $\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]$ dado por (4.61) para N variando entre 1 e 10, considerando $K = 12$, $L = 4$. Uma simulação de Monte Carlo para este valor esperado usando o mapeamento $P(x) = (4x^2 + x + 1)(\text{mod } 2^K)$, que é um polinômio de permutação módulo

Figura 15 – Valores simulados e teóricos de $\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]$ versus N para um mapeamento aleatório com $K = 12$ e $L = 4$.



Fonte: O autor (2023)

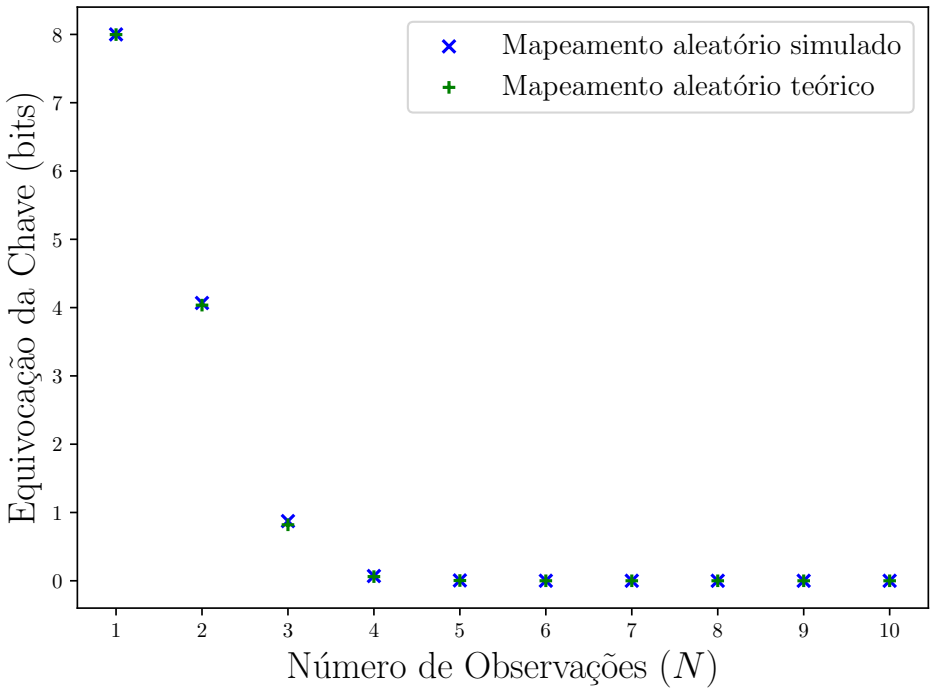
2^K (RIVEST, 2001), também é realizada, considerando que 10000 pares distintos $(\mathbf{s}^i, \mathbf{t}^i)$ são gerados para cada valor de N. Os valores teóricos e simulados possuem excelente concordância.

O valor da equivocação da chave para múltiplas observações, dada por (4.45), é calculada teoricamente para o tipo de mapeamento considerado ao aproximar em (4.57) que as variáveis de Bernoulli A_j presentes são independentes, de forma que se escreve $|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|$ como uma variável aleatória descrita como 1 somada com uma variável Binomial. O valor teórico da equivocação pode então ser computado numericamente a partir de

$$\begin{aligned} H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N) &= \sum_{m=1}^{2^{K-L}} \Pr(|\mathcal{K}^1 \cap \cdots \cap \mathcal{K}^N| = m) \log_2 m \\ &\approx \sum_{n=0}^{2^K-1} \binom{2^K-1}{n} \Psi\left(\left(\frac{2^{K-L}-1}{2^K-1} \right)^N, 2^K - 1, n \right) \log_2 n, \end{aligned} \tag{4.66}$$

em que $\Psi(\cdot)$ foi definida em (4.37). Uma simulação de Monte Carlo também é realizada com 10000 pares distintos de $(\mathbf{s}^1, \mathbf{t}^1), \ldots, (\mathbf{s}^{10}, \mathbf{t}^{10})$. Os valores teóricos e simulados da equivocação da chave para múltiplas observações estão ilustrados na Fig. 16. Observa-se que para uma observação o valor da equivocação apresenta o valor de $K - L = 8$, como esperado pelo obtido na Subseção 4.2.3. Após a primeira observação, a equivocação cai por um fator aproximado de L a cada nova observação, até ir se aproximando de zero, o que ocorre, aproximadamente, para N maior que 3. Os valores teóricos e simulados obtidos apresentam boa concordância.

Figura 16 – Valores simulados e teóricos de $H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$ versus N para um mapeamento aleatório com $K = 12$ e $L = 4$.



Fonte: O autor (2023)

4.4.3 Mapeamento entre subconjuntos

O tipo de mapeamento definido nesta subseção apresenta uma forma de controlar o decaimento dos valores em (4.40) e (4.45) de uma maneira mais suave com o aumento de N. Para isso, considera-se o mapeamento $F(\cdot)$ como uma composição dos mapeamentos $T(\cdot)$ e $M(\cdot)$, como definido em (4.14). Cada um desses mapeamentos tem suas propriedades descritas em separado nesta subseção. Assim como visto no final da Subseção 4.4.2, se considera a partição no espaço de condições iniciais $\mathcal{X}_0$,

$$\mathcal{X}_0 = \mathcal{X}_0^0 \cup \mathcal{X}_0^1 \cup \cdots \cup \mathcal{X}_0^{2^L-1}, \quad \mathcal{X}_0^i \cap \mathcal{X}_0^j = \emptyset, \, \forall \, i \neq j, \tag{4.67}$$

em que cada $\mathcal{X}_0^i$ é definido como em (4.63). O mapeamento $T(\cdot)$ aqui considerado também segue (4.64). Para entrar em detalhes sobre as características do mapeamento $M(\cdot)$, define-se inicialmente a função $\text{Ind}(\cdot)$, que mapeia subconjuntos de $\mathcal{X}_0$ nos índices associados às condições iniciais pertencentes a este subconjunto,

$$\begin{aligned} \text{Ind}(\mathcal{X}_0^i) &= \text{Ind}(\{x_0^{i2^{K-L}}, x_0^{i2^{K-L}+1}, \ldots, x_0^{(i+1)2^{K-L}-1}\}) && (4.68) \\ &= \{i2^{K-L}, i2^{K-L}+1, \ldots, (i+1)2^{K-L}-1\}. && (4.69) \end{aligned}$$

Considere também o agrupamento de subconjuntos $\mathcal{X}_0^i$ em pares da seguinte forma:

$$\mathcal{C}^i = \mathcal{X}_0^{2i} \cup \mathcal{X}_0^{2i+1}, \quad 0 \leq i < 2^{L-1}, \tag{4.70}$$

de forma que $|C^i| = 2^{K-L+1}$. Ao aplicar a função Ind$(\cdot)$ nos conjuntos C^i tem-se

$$\begin{aligned}
\text{Ind}(\mathcal{C}^i) &= \text{Ind}(\mathcal{X}_0^{2i} \cup \mathcal{X}_0^{2i+1}) && (4.71)\\
&= \text{Ind}(\mathcal{X}_0^{2i}) \cup \text{Ind}(\mathcal{X}_0^{2i+1}) && (4.72)\\
&= \{i2^{K-L+1}, i2^{K-L+1}+1, \ldots, i2^{K-L+1}+2^{K-L+1}-1\} && (4.73)\\
&= \{i2^{K-L+1}, i2^{K-L+1}+1, \ldots, (i+1)2^{K-L+1}-1\}. && (4.74)
\end{aligned}$$

Ao interpretar os valores em Ind$(\mathcal{C}^i)$ como números binários de comprimento K, os $L-1$ bits mais significativos são sempre iguais e a diferença entre estes está nos $K-L+1$ bits menos significativos. Desta forma, ao aplicar a operação $\oplus$ de um valor binário qualquer aos elementos de Ind$(\mathcal{C}^i)$ na sua representação binária, o subconjunto resultante também tem a característica de conter elementos com os $L-1$ bits mais significativos iguais, com a diferença estando nos $K-L+1$ bits menos significativos. Consequentemente, aplicar a operação $\oplus$ de algum valor binário aos elementos de Ind$(\mathcal{C}^i)$ vai gerar um outro subconjunto Ind$(\mathcal{C}^j)$. Esta propriedade é descrita como

$$v_1 \in \text{Ind}(C^i), v_1 \oplus v_2 \in \text{Ind}(C^j) \implies v_2 \oplus \text{Ind}(C^i) = \text{Ind}(C^j). \tag{4.75}$$

Para exemplificar a propriedade descrita em (4.75), considere os valores $K = 4$, $L = 2$. Os subconjuntos $\mathcal{X}_0^i$ são definidos em (4.63) como

$$\begin{aligned}
\mathcal{X}_0^0 &= \{x_0^0, x_0^1, x_0^2, x_0^3\}\\
\mathcal{X}_0^1 &= \{x_0^4, x_0^5, x_0^6, x_0^7\}\\
\mathcal{X}_0^2 &= \{x_0^8, x_0^9, x_0^{10}, x_0^{11}\}\\
\mathcal{X}_0^3 &= \{x_0^{12}, x_0^{13}, x_0^{14}, x_0^{15}\},
\end{aligned} \tag{4.76}$$

e, consequentemente,

$$\begin{aligned}
\text{Ind}(\mathcal{X}_0^0) &= \{0,1,2,3\}\\
&= \{0000_2, 0001_2, 0010_2, 0011_2\}\\
\text{Ind}(\mathcal{X}_0^1) &= \{4,5,6,7\}\\
&= \{0100_2, 0101_2, 0110_2, 0111_2\}\\
\text{Ind}(\mathcal{X}_0^2) &= \{8,9,10,11\}\\
&= \{1000_2, 1001_2, 1010_2, 1011_2\}\\
\text{Ind}(\mathcal{X}_0^3) &= \{12,13,14,15\}\\
&= \{1100_2, 1101_2, 1110_2, 1111_2\},
\end{aligned} \tag{4.77}$$

em que os elementos de cada conjunto Ind$(\mathcal{X}_0^i)$ também são mostrados na sua representação binária com K bits. De maneira semelhante, os conjuntos $\mathcal{C}^i$ são construídos seguindo (4.70),

$$\begin{aligned}
\mathcal{C}^0 &= \{x_0^0, x_0^1, x_0^2, x_0^3, x_0^4, x_0^5, x_0^6, x_0^7\}\\
\mathcal{C}^1 &= \{x_0^8, x_0^9, x_0^{10}, x_0^{11}, x_0^{12}, x_0^{13}, x_0^{14}, x_0^{15}\},
\end{aligned} \tag{4.78}$$

e, consequentemente,

$$\begin{aligned}\text{Ind}(\mathcal{C}^0) &= \{0,1,2,3,4,5,6,7\} \\ &= \{0000_2, 0001_2, 0010_2, 0011_2, 0100_2, 0101_2, 0110_2, 0111_2\} \\ \text{Ind}(\mathcal{C}^1) &= \{8,9,10,11,12,13,14,15\} \\ &= \{1000_2, 1001_2, 1010_2, 1011_2, 1100_2, 1101_2, 1110_2, 1111_2\}.\end{aligned} \tag{4.79}$$

Considere $v_1 = 1 = 0001_2$ e $v_2 = 13 = 1101_2$, de forma que $v_1 \in \text{Ind}(\mathcal{C}^0)$. Como $v_1 \oplus v_2 = 12 = 1100_2$, logo, $v_1 \oplus v_2 \in \text{Ind}(\mathcal{C}^1)$. Note que, como esperado por (4.75), tem-se que $v_2 \oplus \text{Ind}(\mathcal{C}^0) = \text{Ind}(\mathcal{C}^1)$, isto é, se a operação $\oplus$ pelo valor v_2 é feita elemento a elemento no subconjunto $\text{Ind}(\mathcal{C}^0)$ o resultado é o subconjunto $\text{Ind}(\mathcal{C}^1)$.

Com a definição do mapeamento $T(\cdot)$ seguindo (4.64), dos subconjuntos $\mathcal{C}^i$, e da função $M(\cdot)$ seguindo (4.65), define-se um comportamento desejado para a função de permutação $P(\cdot)$ associada ao mapeamento $M(\cdot)$ como

$$\mathbf{k} \oplus \mathbf{s} \in \text{Ind}(\mathcal{C}^i) \iff P(\mathbf{k} \oplus \mathbf{s}) \in \text{Ind}(\mathcal{C}^i), \quad \text{em que } M(\mathbf{k} \oplus \mathbf{s}) = x_0^{P(\mathbf{k} \oplus \mathbf{s})}, \tag{4.80}$$

ou seja, $M(\cdot)$ é definido de maneira semelhante a (4.65), mas com a restrição de que a função de permutação $P(\cdot)$ realiza permutações apenas entre os elementos de cada subconjunto $\text{Ind}(\mathcal{C}^i)$. Com as propriedades dos mapeamentos $T(\cdot)$ e $M(\cdot)$ bem definidas, o mapeamento $F(\cdot)$ está definido. Para calcular os valores de (4.40) e (4.45), considera-se primeiramente a probabilidade

$$\begin{aligned}\Pr(\mathbf{k} \in \mathcal{K}^1) &= \Pr(F(\mathbf{k} \oplus \mathbf{s}^1) = F(\mathbf{k}_{\text{leg}} \oplus \mathbf{s}^1)) &(4.81)\\ &= \Pr(F(\mathbf{k} \oplus \mathbf{s}^1) = \mathbf{t}^1). &(4.82)\end{aligned}$$

Quando $\mathbf{k} = \mathbf{k}_{\text{leg}}$ (a chave legítima), a probabilidade $\Pr(\mathbf{k}_{\text{leg}} \in \mathcal{K}^1) = 1$. O valor $M(\mathbf{k}_{\text{leg}} \oplus \mathbf{s}^1)$ pertence a algum conjunto $\mathcal{C}^i$, de forma que $\mathbf{k}_{\text{leg}} \oplus \mathbf{s}^1$ pertence a $\text{Ind}(C^i)$. Em consequência, a *tag* $\mathbf{t}^1$ é gerada pelas condições iniciais associadas a algum dos subconjuntos que formam $\mathcal{C}^i$ ($\mathcal{X}_0^{2i}$ ou $\mathcal{X}_0^{2i+1}$). Dessa forma, a propriedade (4.80) indica que se uma chave $\mathbf{k}$ é utilizada em conjunto com $\mathbf{s}^1$ para gerar a *tag* $\mathbf{t}^1$, então uma condição necessária é que $\mathbf{k} \oplus \mathbf{s}^1 \in \text{Ind}(\mathcal{C}^i)$, ou seja, $\mathbf{k} \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i)$. Consequentemente, todas as chaves em $\mathcal{K}^1$ estão presentes em $\mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i)$, e então $\mathcal{K}^1 \subset \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i)$. Para algum $\mathbf{k} \oplus \mathbf{s}^1 \notin \text{Ind}(\mathcal{C}^i)$, $M(\mathbf{k} \oplus \mathbf{s}^1)$ vai gerar alguma condição inicial que não gera $\mathbf{t}^1$, logo, $\Pr(\mathbf{k} \in \mathcal{K}^1) = 0$. Já no caso em que $\mathbf{k} \oplus \mathbf{s}^1 \in \text{Ind}(\mathcal{C}^i)$, a probabilidade de $\mathbf{k} \in \mathcal{K}^1$ é

$$\begin{aligned}\Pr(\mathbf{k} \in \mathcal{K}^1) &= \frac{|\{\mathbf{k} \mid F(\mathbf{k} \oplus \mathbf{s}^1) = \mathbf{t}^1, \mathbf{k} \in \mathcal{C}^i - \{\mathbf{k}_{\text{leg}}\}\}|}{|\mathcal{C}^i - \{\mathbf{k}_{\text{leg}}\}|} &(4.83)\\ &= \frac{2^{K-L}-1}{2^{K-L+1}-1}, \quad \forall\, M(\mathbf{k} \oplus \mathbf{s}^1) \in \mathcal{C}^i - \{\mathbf{k}_{\text{leg}}\}. &(4.84)\end{aligned}$$

Tem-se então

$$\Pr(\mathbf{k} \in \mathcal{K}^1) = \begin{cases} 1, & \text{se } \mathbf{k} = \mathbf{k}_{\text{leg}}, \text{ em que } \mathbf{k}_{\text{leg}} \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) \\ \frac{2^{K-L}-1}{2^{K-L+1}-1}, & \text{se } \mathbf{k} \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) - \{\mathbf{k}_{\text{leg}}\} \\ 0, & \text{se } \mathbf{k} \notin \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i). \end{cases} \tag{4.85}$$

Como visto, para o par $(\mathbf{s}^1, \mathbf{t}^1)$, tem-se que $\mathcal{K}^1 \subset \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i)$. Para um outro par $(\mathbf{s}^2, \mathbf{t}^2)$, tem-se que $\mathcal{K}^2 \subset \mathbf{s}^2 \oplus \text{Ind}(\mathcal{C}^j)$. A propriedade (4.75) indica que $\mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i)$ resulta em algum subconjunto $\text{Ind}(\mathcal{C}^{i'})$, e da mesma maneira $\mathbf{s}^2 \oplus \text{Ind}(\mathcal{C}^j)$ resultará em algum subconjunto $\text{Ind}(\mathcal{C}^{j'})$. Como os subconjuntos $\text{Ind}(C^i)$ são disjuntos, e tem-se que $\mathbf{k}_{\text{leg}} \in \text{Ind}(\mathcal{C}^{i'})$ e $\mathbf{k}_{\text{leg}} \in \text{Ind}(\mathcal{C}^{j'})$, então $\text{Ind}(\mathcal{C}^{i'}) = \text{Ind}(\mathcal{C}^{j'})$, isto é, $\mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) = \mathbf{s}^2 \oplus \text{Ind}(\mathcal{C}^j)$. Já que as mensagens $\mathbf{s}^1$ e $\mathbf{s}^2$ são variáveis independentes e uniformemente distribuídas sobre $\mathcal{S}$, e também tem-se que $\mathcal{K}^1$ e $\mathcal{K}^2$ são subconjuntos de $\text{Ind}(\mathcal{C}^{i'}) = \text{Ind}(\mathcal{C}^{j'})$, pode-se aproximar que a construção dos conjuntos $\mathcal{K}^1$ e $\mathcal{K}^2$ são independentes, exceto pelo fato de que a chave legítima $\mathbf{k}_{\text{leg}}$ pertence a ambos. Pode-se escrever então que $\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2) = \Pr(\mathbf{k} \in \mathcal{K}^1) \Pr(\mathbf{k} \in \mathcal{K}^2)$. De uma maneira geral, este raciocínio é estendido para N observações, obtendo-se que $\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N) = \Pr(\mathbf{k} \in \mathcal{K}^1) \Pr(\mathbf{k} \in \mathcal{K}^2) \cdots \Pr(\mathbf{k} \in \mathcal{K}^N)$, e assim,

$$\Pr(\mathbf{k} \in \mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N) = \begin{cases} 1, & \text{se } \mathbf{k} = \mathbf{k}_{\text{leg}}, \text{ em que } \mathbf{k}_{\text{leg}} \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i), \\ \left(\frac{2^{K-L}-1}{2^{K-L+1}-1}\right)^N, & \text{se } \mathbf{k} \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) - \{\mathbf{k}_{\text{leg}}\}, \\ 0, & \text{se } \mathbf{k} \notin \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i). \end{cases} \tag{4.86}$$

De maneira semelhante ao feito na Subseção 4.4.2, define-se a variável aleatória A_j como em (4.55), e o valor de $|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|$ é reescrito como sendo

$$\begin{aligned} |\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N| &= \sum_{j=0}^{2^K-1} A_j & (4.87) \\ &= 1 + \sum_{\substack{\mathbf{k}_j \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) \\ \mathbf{k}_j \neq \mathbf{k}_{\text{leg}}}} A_j, & (4.88) \end{aligned}$$

O cálculo do valor esperado em (4.40) pode então ser feito por meio de

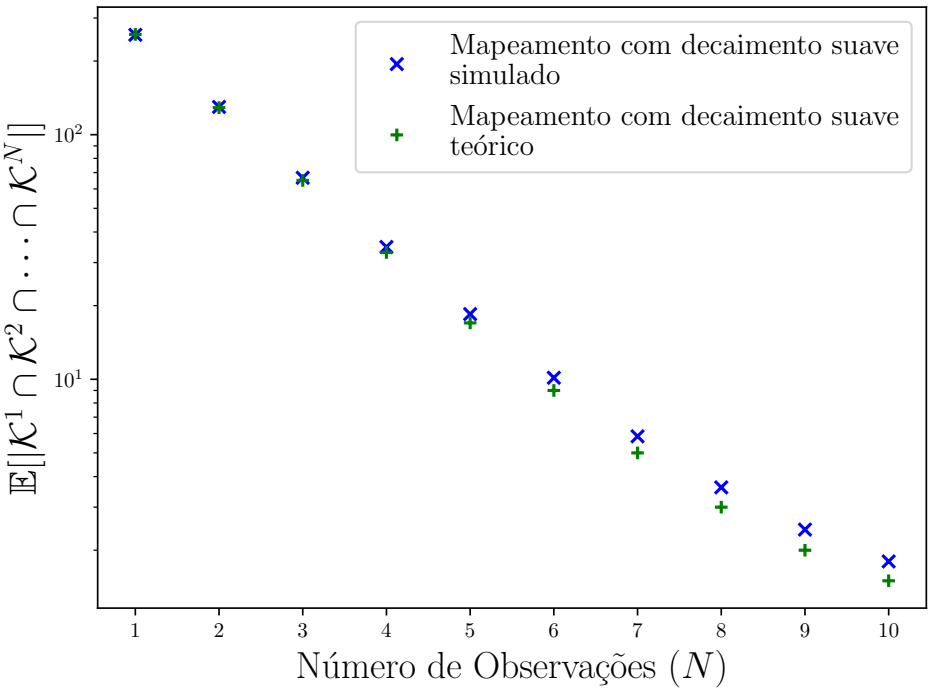
$$\begin{aligned} \mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|] &= \mathbb{E}\left[\sum_{j=0}^{2^K-1} A_j\right] & (4.89) \\ &= 1 + \mathbb{E}\left[\sum_{\substack{\mathbf{k}_j \in \mathbf{s}^1 \oplus \text{Ind}(\mathcal{C}^i) \\ \mathbf{k}_j \neq \mathbf{k}_{\text{leg}}}} A_j\right] & (4.90) \\ &= 1 + (2^{K-L+1} - 1)\left(\frac{2^{K-L}-1}{2^{K-L+1}-1}\right)^N. & (4.91) \end{aligned}$$

Se forem escolhidos valores de K e L tais que $2^{K-L} >> 1$, aproxima-se a última expressão por

$$\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|] \approx 1 + 2^{K-L-(N-1)}. \tag{4.92}$$

Conclui-se então que, para um mapeamento $F(\cdot)$ utilizando permutações entre elementos dos subconjuntos $\mathcal{C}^i$, a queda do número de chaves possíveis decai mais lentamente que o caso aleatório, e por um fator aproximado de 2 a cada nova observação.

Figura 17 – Valores simulados e teóricos de $\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]$ versus N para um mapeamento com decaimento suave com $K = 12$ e $L = 4$.



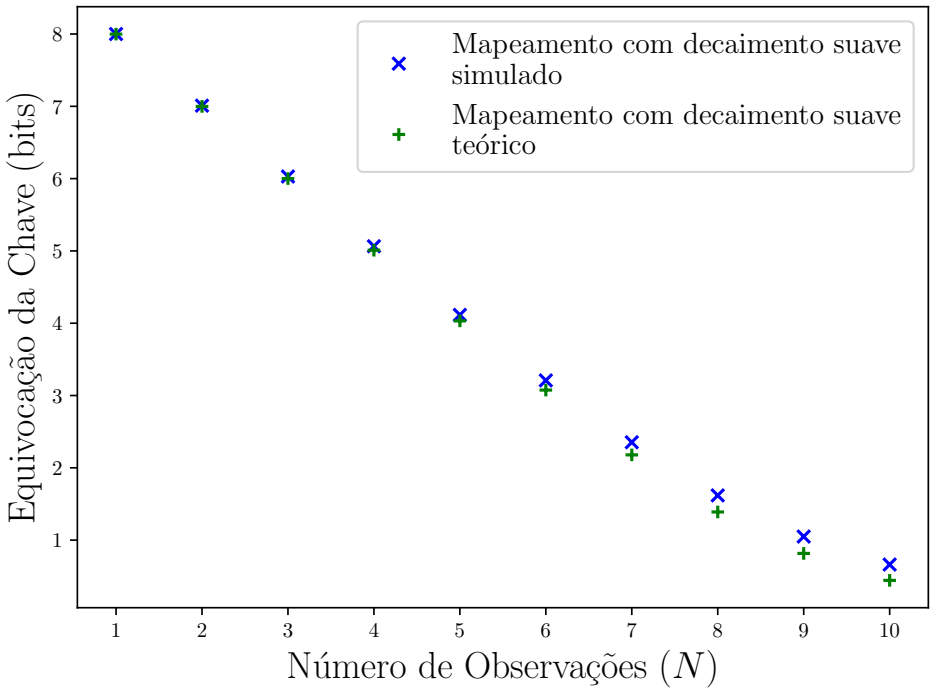
Fonte: O autor (2023)

Para exemplificar os resultados obtidos nesta subseção considere a escolha de parâmetros $K = 12$ e $L = 4$, além de uma função $F = T \circ M$ com $T(\cdot)$ e $M(\cdot)$ definidos como em (4.64) e (4.65), respectivamente. A diferença aqui está na permutação $P(\cdot)$ escolhida, que neste caso é

$$P(x) = (6x^2 + x + 1)(\text{mod } 2^{K-L+1}) + \left\lfloor \frac{x}{2^{K-L+1}} \right\rfloor 2^{K-L+1}. \tag{4.93}$$

A particularidade desta permutação $P(\cdot)$ se dá no fato de que ela divide o domínio $\{0, 1, \ldots, 2^K - 1\}$ em intervalos de tamanho 2^{K-L+1} e permuta apenas os elementos dentro de cada intervalo, como esperado pela propriedade (4.80). A Fig. 17 ilustra os valores de $\mathbb{E}[|\mathcal{K}^1 \cap \mathcal{K}^2 \cap \cdots \cap \mathcal{K}^N|]$ dados por (4.92) e os obtidos por simulação com a permutação considerada, para N variando entre 1 e 10, $K = 12$ e $L = 4$. De maneira semelhante ao feito na Subseção 4.4.2, o valor da equivocação da chave para múltiplas observações, dada por (4.45), também é calculada numericamente a partir de (4.88), na qual se considera que as variáveis de Bernoulli A_j são independentes. Os valores teóricos e simulados da equivocação da chave para múltiplas observações estão ilustrados na Fig. 18. Para uma observação, o valor da equivocação apresenta o valor de $K - L = 8$, como esperado pelo obtido na Subseção (4.2.3). Após a primeira observação, a equivocação cai por um fator aproximado de 1 bit a cada nova observação, até ir se aproximando de zero. Os valores teóricos e simulados mostrados nas Figs. 17 e 18 apresentam boa concordância.

Figura 18 – Valores simulados e teóricos de $H(\mathbf{k} \mid \mathbf{s}^1, \ldots, \mathbf{s}^N, \mathbf{t}^1, \ldots, \mathbf{t}^N)$ versus N para um mapeamento com decaimento suave com $K = 12$ e $L = 4$.



Fonte: O autor (2023)

5 CONCLUSÕES

Este trabalho aborda os desafios e as contribuições relacionados a duas áreas de segurança da informação: os PRNGs e PLA. Os objetivos estabelecidos são alcançados por meio da proposição de técnicas de geração de números pseudoaleatórios em estruturas algébricas discretas, da exploração do uso de mapas caóticos na geração das *tags* em sistemas de PLA e da investigação da informação obtida por um usuário malicioso sobre a chave secreta utilizada no sistema de PLA, quando o mesmo intercepta múltiplas comunicações legítimas.

No que diz respeito aos PRNGs, dois algoritmos são propostos. O primeiro PRNG baseia-se no mapa de Arnold sobre anéis de inteiros do tipo $\mathbb{Z}_{2^m}$, enquanto o segundo é obtido a partir de uma definição de q-análogos sobre corpos finitos. Ambos os algoritmos são descritos por meio de RTLs e são submetidos a análises estatísticas que comprovaram a qualidade das sequências pseudoaleatórias geradas. Além disso, as implementações em hardware desses PRNGs são comparadas com outros algoritmos da literatura, demonstrando suas vantagens em termos de implementação em FPGA. Dentre essas vantagens, os PRNGs propostos apresentam ganhos acima de 270% na taxa de geração de bits. O PRNG baseado no mapa de Arnold apresenta valores de ocupação de hardware comparáveis aos menores valores encontrados na literatura, enquanto o baseado em q-análogos apresenta uma redução no número de LUTs e FFs da ordem de 60% e 62%, respectivamente.

No contexto de PLA, é proposto um novo método de geração das *tags* baseado em sequências caóticas discretizadas. Esse método apresenta a vantagem de garantir segurança incondicional, independente do ruído presente no canal de comunicação. Em contraste, o algoritmo clássico de geração da *tag* não oferece essa garantia de segurança incondicional. Comparando-se o método proposto com outros algoritmos presentes na literatura, métricas baseadas em teoria da informação são utilizadas para avaliar a eficácia do sistema proposto. Os resultados mostram que o sistema proposto supera o algoritmo clássico em termos de equivocação da chave, enquanto apresenta um desempenho comparável a outro sistema baseado em sequências caóticas, apresentando resultados superiores para baixos valores de TNR.

Na Dissertação também se analisa a segurança incondicional do algoritmo proposto quando um usuário malicioso tem acesso a múltiplos pares legítimos de mensagem e *tag*. Duas métricas são propostas para medir essa segurança, sendo calculadas e simuladas para diferentes mapeamentos associados ao método proposto de geração das *tags*. Os resultados indicam que o sistema proposto pode oferecer segurança incondicional mesmo diante de múltiplas observações. Esse tipo de análise não foi encontrado na literatura.

Em resumo, a utilização de PRNGs baseados em estruturas algébricas discretas, como anéis de inteiros e corpos finitos, demonstra ser uma alternativa viável e eficiente para a geração

de sequências pseudoaleatórias. Os algoritmos propostos neste trabalho mostram resultados promissores em termos de qualidade das sequências geradas, além de apresentarem uma boa escalabilidade em implementações em hardware. Além disso, o método de geração da *tag* baseado em sequências caóticas discretizadas oferece segurança incondicional e vantagens em relação a outros algoritmos presentes na literatura. A análise da segurança incondicional diante de múltiplas observações também representa um avanço importante na avaliação e no aprimoramento desses sistemas.

5.1 TRABALHOS FUTUROS

Para trabalhos futuros, há várias direções que podem ser exploradas para aprimorar e expandir os conhecimentos desenvolvidos neste estudo. Primeiramente, seria interessante realizar o estudo de PRNGs sobre outras estruturas algébricas, como corpos de extensão. Visto que os PRNGs apresentados envolvem anéis de inteiros do tipo $\mathbb{Z}_{p^m}$ e corpos finitos do tipo GF(p), a investigação do uso de corpos de extensão (GF(p^m)) permitiria explorar novas possibilidades de geração de sequências pseudoaleatórias e expandir ainda mais o escopo das aplicações práticas desses algoritmos. Uma abordagem promissora seria o estudo de uma definição de q-análogos sobre corpos de extensão, visando obter um novo PRNG.

Outra linha de pesquisa promissora é a extensão da análise do sistema de geração da *tag* para outros tipos de canais (além do canal com ruído aditivo Gaussiano de média nula). Também é promissora a implementação do algoritmo proposto de geração da *tag* em um sistema prático. Esse tipo de implementação poderia ser realizada em sistemas de rádio definido por software, por exemplo. Ao utilizar simulações e métricas associadas, seria possível avaliar o desempenho e a segurança do algoritmo em um cenário prático. Essa abordagem permitiria testar e validar o método proposto em condições reais, levando em consideração fatores como ruído do canal de comunicação, interferência e outros desafios presentes em ambientes reais. Além disso, a implementação em um sistema prático proporcionaria a oportunidade de realizar experimentos e coletar dados empíricos, o que contribuiria para a compreensão mais aprofundada do desempenho e das características do algoritmo proposto.

SOBRE O AUTOR

O autor nasceu em Recife, Brasil, no ano de 1998. Em 2021, formou-se em Engenharia Eletrônica pela Universidade Federal de Pernambuco. Ele tem interesse em diversas áreas relacionadas à computação e tecnologia, e já participou de competições e projetos envolvendo Aprendizagem de Máquina, Criptografia e FPGAs. Além disso, ele é fascinado por Cibersegurança e, em seu tempo livre, gosta de estudar assuntos relacionados a *Ethical Hacking* e *Red Teaming*.

REFERÊNCIAS

ALAWIDA, M. et al. Digital cosine chaotic map for cryptographic applications. **IEEE Access**, v. 7, p. 150609–150622, out. 2019.

BAJARD, J. C.; DUQUESNE, S. Montgomery-friendly primes and applications to cryptography. **Journal of Cryptographic Engineering**, v. 11, n. 4, p. 399–415, nov. 2021.

BAO, H. et al. Discrete memristor hyperchaotic maps. **IEEE Transactions on Circuits and Systems I: Regular Papers**, v. 68, n. 11, p. 4534–4544, nov. 2021.

BASSHAM III, L. E. et al. **SP 800-22 Rev. 1a. A Statistical Test Suite for Random and Pseudorandom Number Generators for Cryptographic Applications**. Gaithersburg, MD, United States: Nat. Inst. Std. & Technol., 2010.

BLAHUT, R. E. **Fast Algorithms for Signal Processing**. 1. ed. [S.l.]: Cambridge University Press, 2010.

CHAVES, D. P. B.; SOUZA, C. E. C.; PIMENTEL, C. A smooth chaotic map with parameterized shape and symmetry. **EURASIP Journal on Applied Signal Processing**, v. 2016, n. 1, p. 122, dez. 2016.

EVANGELISTA, J. V. de C. **Physical-layer authentication Using chaotic maps**. Dissertação (Mestrado) — Programa de Pós-Graduação em Engenharia Elétrica, Universidade Federal de Pernambuco, Recife, PE, Brasil, ago. 2016.

GARCIA-BOSQUE, M. et al. Chaos-Based Bitwise Dynamical Pseudorandom Number Generator On FPGA. **IEEE Transactions on Instrumentation and Measurement**, v. 68, n. 1, p. 291–293, jan. 2019.

GU, Z. et al. Physical layer authentication for non-coherent massive SIMO-enabled industrial IoT communications. **IEEE Transactions on Information Forensics and Security**, v. 15, p. 3722–3733, jun. 2020.

GUPTA, M. D.; CHAUHAN, R. K. Coupled variable-input LCG and clock divider-based large period pseudo-random bit generator on FPGA. **IET Computers & Digital Techniques**, v. 15, n. 5, p. 349–361, abr. 2021.

KALANADHABHATTA, S. et al. PUF-Based Secure Chaotic Random Number Generator Design Methodology. **IEEE Transactions on Very Large Scale Integration (VLSI) Systems**, v. 28, n. 7, p. 1740–1744, jul. 2020.

KOPPARTHI, V. R. et al. Hardware architecture of a digital piecewise linear chaotic map with perturbation for pseudorandom number generation. **AEU - International Journal of Electronics and Communications**, v. 147, p. 154138, fev. 2022.

KOYUNCU, I. et al. Design, FPGA implementation and statistical analysis of chaos-ring based dual entropy core true random number generator. **Analog Integrated Circuits and Signal Processing**, v. 102, p. 445–456, fev. 2020.

LAN, R. et al. A parameter-selection-based chaotic system. **IEEE Transactions on Circuits and Systems II: Express Briefs**, v. 66, n. 3, p. 492–496, mar. 2019.

LIU, J.; WANG, X. Physical layer authentication enhancement using two-dimensional channel quantization. **IEEE Transactions on Wireless Communications**, v. 15, n. 6, p. 4171–4182, jun. 2016.

MACHICAO, J. et al. A visual analysis method of randomness for classifying and ranking pseudo-random number generators. **Information Sciences**, v. 558, p. 1–20, maio 2021.

MAENG, S. J. et al. Precoder design for physical-layer security and authentication in massive MIMO UAV communications. **IEEE Transactions on Vehicular Technology**, v. 71, n. 23, p. 2949–2964, mar. 2022.

MASSEY, J. L. **Lecture Notes on Applied Digital Information Technology II**. 1998.

MAURER, U. Authentication theory and hypothesis testing. **IEEE Transactions on Information Theory**, v. 46, n. 4, p. 1350–1356, 2000.

PAAR, C.; PELZL, J. **Understanding Cryptography: A Textbook for Students and Practitioners**. 1. ed. Berlin, Germany; London, United Kingdom: Springer Publishing Company, Incorporated, 2009.

PANDA, A. K.; RAY, K. C. A Coupled Variable Input LCG Method and its VLSI Architecture for Pseudorandom Bit Generation. **IEEE Transactions on Instrumentation and Measurement**, v. 69, n. 4, p. 1011–1019, abr. 2020.

PERAZZONE, J. B. et al. Cryptographic side-channel signaling and authentication via fingerprint embedding. **IEEE Transactions on Information Forensics and Security**, v. 13, n. 9, p. 2216–2225, set. 2018.

PREISHUBER, M. et al. Depreciating Motivation and Empirical Security Analysis of Chaos-Based Image and Video Encryption. **IEEE Transactions on Information Forensics and Security**, v. 13, n. 9, p. 2137–2150, set. 2018.

QAISI, M.; ALTHUNIBAT, S.; QARAQE, M. Phase-assisted dynamic tag-embedding message authentication for IoT networks. **IEEE Internet of Things Journal**, v. 9, n. 20, p. 20620–20629, out. 2022.

RIVEST, R. L. Permutation Polynomials Modulo 2^w. **Finite Fields and Their Applications**, v. 7, n. 2, p. 287–292, abr. 2001.

SENIGAGLIESI, L.; BALDI, M.; GAMBI, E. Comparison of statistical and machine learning techniques for physical layer authentication. **IEEE Transactions on Information Forensics and Security**, v. 16, p. 1506–1521, abr. 2021.

SHANNON, C. E. A mathematical theory of communication. **The Bell System Technical Journal**, v. 27, n. 3, p. 379–423, jul. 1948.

SOUZA, C. E. C.; CHAVES, D. P. B.; PIMENTEL, C. One-Dimensional Pseudo-Chaotic Sequences Based on the Discrete Arnold's Cat Map Over $\mathbb{Z}_{3^m}$. **IEEE Transactions on Circuits and Systems II: Express Briefs**, v. 68, n. 1, p. 491–495, jul. 2021.

SOUZA, C. E. C. et al. Pseudo-Chaotic Sequences Generated by the Discrete Arnold's Map Over $\mathbb{Z}_{2^m}$: Period Analysis and FPGA Implementation. **IEEE Transactions on Instrumentation and Measurement**, v. 71, p. 1–10, ago. 2022.

SOUZA, C. E. C. et al. q-Analogs Over Finite Fields: Definition, Algebraic Properties, and Application in Pseudo-Random Number Generators. **IEEE Transactions on Circuits and Systems II: Express Briefs**, p. 1–1, fev. 2023.

STROGATZ, S. H. **Nonlinear Dynamics and Chaos: With Applications to Physics, Biology, Chemistry and Engineering**. 2. ed. Boulder, CO, United States: Westview Press, 2015.

TANENBAUM, A. S.; WETHERALL, D. **Computer Networks**. 5. ed. Boston, MA, United States: Prentice Hall, 2011.

XIE, N.; CHEN, C.; MING, Z. Security model of authentication at the physical layer and performance analysis over fading channels. **IEEE Transactions on Dependable and Secure Computing**, v. 18, n. 1, p. 253–268, fev. 2021.

XIE, N. et al. Multi-user physical-layer authentication and classification. **IEEE Transactions on Wireless Communications**, jan. 2023.

XIE, N. et al. Privacy-preserving physical-layer authentication for non-orthogonal multiple access systems. **IEEE Journal on Selected Areas in Communications**, v. 40, n. 4, p. 1371–1385, abr. 2022.

YU, P. L.; BARAS, J. S.; SADLER, B. M. Physical-Layer Authentication. **IEEE Transactions on Information Forensics and Security**, v. 3, n. 1, p. 38–51, mar. 2008.

YU, P. L.; SADLER, B. M. MIMO Authentication via Deliberate Fingerprinting at the Physical Layer. **IEEE Transactions on Information Forensics and Security**, v. 6, n. 3, p. 606–615, set. 2011.

YU, P. L.; VERMA, G.; SADLER, B. M. Wireless physical layer authentication via fingerprint embedding. **IEEE Communications Magazine**, v. 53, n. 9, p. 48–53, jun. 2015.

ZHANG, P. et al. Physical layer authentication jointly utilizing channel and phase noise in MIMO systems. **IEEE Transactions on Communications**, v. 68, n. 4, p. 2446–2458, abr. 2020.

ZHU, S. et al. A class of quadratic polynomial chaotic maps and its application in cryptography. **IEEE Access**, v. 7, p. 34141–34152, mar. 2019.