

UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
DEPARTAMENTO DE INFORMÁTICA E ESTATÍSTICA
CIÊNCIA DA COMPUTAÇÃO

Alex Davis Neuwiem da Silva

**Provas de Conhecimento Zero: Um Estudo sobre zk-SNARKs e sua Aplicação em
Sistemas de Autenticação**

Florianópolis
2025

Alex Davis Neuwiem da Silva

Provas de Conhecimento Zero: Um Estudo sobre zk-SNARKs e sua Aplicação em Sistemas de Autenticação

Trabalho de Conclusão de Curso submetido ao
Curso de Graduação em Ciência da Computa-
ção do Centro Tecnológico da Universidade
Federal de Santa Catarina como requisito para
obtenção do título de Bacharel em Ciência da
Computação.

Orientadora: Prof^ª. Thaís Bardini Idalino,
Dr^ª.

Coorientador: Gustavo Zambonin, Me.

Florianópolis

2025

Ficha catalográfica gerada por meio de sistema automatizado gerenciado pela BU/UFSC.
Dados inseridos pelo próprio autor.

Neuwiem da Silva, Alex Davis
Provas de Conhecimento Zero : Um Estudo sobre zk-SNARKs
e sua Aplicação em Sistemas de Autenticação / Alex Davis
Neuwiem da Silva ; orientadora, Thaís Bardini Idalino,
coorientador, Gustavo Zambonin, 2025.
84 p.

Trabalho de Conclusão de Curso (graduação) -
Universidade Federal de Santa Catarina, Centro Tecnológico,
Graduação em Ciências da Computação, Florianópolis, 2025.

Inclui referências.

1. Ciências da Computação. 2. Criptografia. 3. Provas de
Conhecimento Zero. 4. Sistema de Autenticação. I. Bardini
Idalino, Thaís. II. Zambonin, Gustavo. III. Universidade
Federal de Santa Catarina. Graduação em Ciências da
Computação. IV. Título.

Alex Davis Neuwiem da Silva

**Provas de Conhecimento Zero: Um Estudo sobre zk-SNARKs e sua Aplicação em
Sistemas de Autenticação**

Este Trabalho de Conclusão de Curso foi julgado adequado para obtenção do Título de Bacharel em Ciência da Computação e aprovado em sua forma final pelo curso de Graduação em Ciência da Computação.

Florianópolis, 19 de Novembro de 2025.

Prof. Renato Cislighi, Dr.
Coordenador do Curso

Banca Examinadora:

Prof^ª. Thaís Bardini Idalino, Dr^a.
Orientadora
Universidade Federal de Santa Catarina

Gustavo Zambonin, Me.
Coorientador
Universidade Federal de Santa Catarina

Prof. Ricardo Felipe Custódio, Dr.
Avaliador
Universidade Federal de Santa Catarina

Dedico este trabalho aos meus pais e à minha irmã.

AGRADECIMENTOS

Gostaria de agradecer aos meus pais, Patricio Queiroz da Silva e Kellen Luciany Neuwiem. Vocês são os principais responsáveis por eu ter chegado tão longe. Obrigado por acreditarem em mim e por todo o seu apoio. Também quero agradecer à minha maravilhosa irmã, Anna Lúcia Neuwiem da Silva, pelo incentivo constante, pela amizade e por estar sempre ao meu lado. Um agradecimento especial ao meu fiel companheiro, meu cachorro Bilbo, que, com seu rostinho alegre, tornou os dias de estudo e cansaço muito mais toleráveis.

Aos meus grandes amigos e companheiros de graduação, Luan Diniz Moraes e Lucas Castro Truppel Machado, sou imensamente grato por terem caminhado ao meu lado. Juntos, superamos os períodos mais difíceis desta trajetória. A parceria de vocês foi fundamental para que eu não desistisse.

À minha orientadora, Thaís Bardini Idalino, expresso minha profunda gratidão por ser sempre tão atenciosa. Obrigado pela paciência e por guiar meus passos com sabedoria. Agradeço também ao meu coorientador, Gustavo Zambonin, por todo o suporte e por me ajudar a superar as dificuldades técnicas e teóricas que encontrei pelo caminho.

Aos professores Jean Everson Martina e Ricardo Felipe Custódio, obrigado por todo o apoio, pelos conselhos e pelos ensinamentos valiosos que contribuíram significativamente para a minha formação profissional e pessoal.

Por fim, gostaria de agradecer aos meus colegas do LabSEC, Arthur Gabriel Crippa Milanez, Gabriel Holstein Meireles, Lucas Coelho Pini de Sousa, Gustavo Ribeiro Bôdi, Vitor Praxedes Calegari e João Paulo Oliveira Melo. Obrigado pelas risadas no dia a dia que tornam o ambiente de pesquisa repleto de bons momentos.

A você, prezado leitor, e a todos que, direta ou indiretamente, fizeram parte desta jornada, muito obrigado.

*“É estranho, mas as coisas boas e os dias agradáveis são narrados depressa,
e não há muito o que ouvir sobre eles, enquanto as coisas desconfortáveis,
palpitantes e até mesmo horríveis podem dar uma boa história e levar um
bom tempo para contar.”*
J.R.R. Tolkien

RESUMO

As Provas de Conhecimento Zero (PCZs) representam uma ferramenta criptográfica poderosa para garantir privacidade e segurança em sistemas digitais. No entanto, a literatura sobre o tema encontra-se fragmentada entre artigos científicos e implementações técnicas, dificultando uma visão consolidada da área. Este trabalho aborda essa questão ao apresentar um estudo que conecta os fundamentos teóricos das PCZs à sua aplicação prática. A monografia contribui em duas frentes principais: primeiramente, realiza uma análise aprofundada dos conceitos, propriedades e premissas de segurança dos principais protocolos zk-SNARKs (*Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge*), a classe de provas de conhecimento zero mais difundida atualmente. Em segundo lugar, para demonstrar a aplicabilidade desses conceitos, o projeto detalha o desenvolvimento de um sistema de autenticação biométrica facial que preserva a privacidade do usuário ao remover a necessidade de armazenamento de dados sensíveis. A solução proposta utiliza o protocolo **Groth16**, um zk-SNARK, para permitir que um usuário prove sua identidade sem revelar os dados biométricos ao verificador. O resultado é um trabalho autocontido que serve como uma referência sólida sobre o tema em língua portuguesa, unindo a análise teórica a uma demonstração prática de seu potencial.

Palavras-chave: Criptografia. Provas de Conhecimento Zero. Sistema de Autenticação.

ABSTRACT

Zero-Knowledge Proofs (ZKPs) represent a powerful cryptographic tool for ensuring privacy and security in digital systems. However, the literature on the topic is fragmented between scientific articles and technical implementations, hindering a consolidated view of the area. This work addresses this issue by presenting a study that connects the theoretical foundations of ZKPs to their practical application. The undergraduate thesis contributes on two main fronts: first, it conducts an in-depth analysis of the concepts, properties, and security assumptions of the main zk-SNARKs (Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge) protocols, the currently most widespread class of zero-knowledge proofs. Second, to demonstrate the applicability of these concepts, the project details the development of a facial biometric authentication system that preserves user privacy by removing the need for sensitive data storage. The proposed solution utilizes the **Groth16** protocol, a zk-SNARK, to allow a user to prove their identity without revealing biometric data to the verifier. The result is a self-contained work that serves as a solid reference on the subject in the Portuguese language, uniting theoretical analysis with a practical demonstration of its potential.

Keywords: Cryptography. Zero-Knowledge Proofs. Authentication System.

LISTA DE SIGLAS

CMP	Computação Multipartidária
FRC	Fita de Referência Comum
IA	Inteligência Artificial
PCZ	Prova de Conhecimento Zero
QAP	<i>Quadratic Arithmetic Program</i>
R1CS	<i>Rank-1 Constraint System</i>
SNARK	<i>Succinct Non-Interactive Argument of Knowledge</i>
STARK	<i>Scalable Transparent Argument of Knowledge</i>
zk	<i>Zero-Knowledge</i>

LISTA DE FIGURAS

Figura 1 – Circuito aritmético para $f(s_1, s_2, s_3) = (s_1 + s_2) \times s_3$ segundo (CHEN et al., 2023)	33
Figura 2 – Arquitetura de um protocolo interativo baseado no diagrama de (GOLDWASSER; MICALI; RACKOFF, 1989)	39
Figura 3 – Grafo de exemplo sem coloração	40
Figura 4 – Grafos representando a solução 1	41
Figura 5 – Escolha de nós adjacentes da solução 1	42
Figura 6 – Grafo representando a solução 2	43
Figura 7 – Diagrama de processamento de um zk-SNARK segundo (BONEH, 2022) . .	52
Figura 8 – Circuito aritmético da fórmula $x^3 + 2x + 7$ com $x = 2$	54
Figura 9 – Arquitetura do projeto de autenticação biométrica com Provas de Conhecimento Zero na fase de registro	77
Figura 10 – Arquitetura do projeto de autenticação biométrica com Provas de Conhecimento Zero na fase de autenticação	79

LISTA DE TABELAS

Tabela 1 – Tempo médio das operações do sistema	81
---	----

LISTA DE EQUAÇÕES

Equação 1 – Cálculo da Similaridade por Cosseno	46
---	----

SUMÁRIO

1	INTRODUÇÃO	25
1.1	OBJETIVO GERAL	26
1.2	OBJETIVOS ESPECÍFICOS	26
1.3	METODOLOGIA DE PESQUISA	26
2	FUNDAMENTAÇÃO TEÓRICA	29
2.1	PRIMITIVAS MATEMÁTICAS	29
2.1.1	Circuitos Aritméticos e o Lema de Schwartz-Zippel	33
2.2	PRIMITIVAS CRIPTOGRÁFICAS	35
2.2.1	Esquemas Criptográficos e Funções Hash	35
2.2.2	Provas de Conhecimento	37
2.2.2.1	<i>Componentes de um Sistema de Prova de Conhecimento</i>	37
2.2.2.2	<i>Interações do Protocolo de Prova de Conhecimento</i>	38
2.2.2.3	<i>Extrator de Conhecimento</i>	39
2.2.3	Prova de Conhecimento Zero Interativa	40
2.2.3.1	<i>Comprometimento com a coloração</i>	41
2.2.3.2	<i>Propriedades das Prova de Conhecimento Zero</i>	42
2.2.4	Esquema de Comprometimento Polinomial	43
2.3	PRIMITIVAS DE INTELIGÊNCIA ARTIFICIAL	44
2.3.1	O Processo de Reconhecimento Facial	45
2.3.2	Extração de <i>Embeddings</i> Faciais	45
2.3.3	Identificação Facial com Similaridade por Cosseno	46
2.3.3.1	<i>Limiar de Similaridade</i>	46
3	PROTOCOLOS ZK-SNARKS	49
3.1	INTRODUÇÃO AOS SNARKS	49
3.1.1	Propriedades Fundamentais dos zk-SNARKs	49
3.1.2	Etapas de Funcionamento de um zk-SNARK	50
3.1.3	Diagrama de Funcionamento de um zk-SNARK	52
3.2	TIPOS DE PROTOCOLOS ZK-SNARKS	52
3.2.1	Pinocchio	53
3.2.1.1	<i>Aritmetização em Pinocchio</i>	53
3.2.1.2	<i>Cerimônia de Confiança em Pinocchio</i>	59
3.2.1.3	<i>Geração de Prova em Pinocchio</i>	61
3.2.1.4	<i>Verificação de Prova em Pinocchio</i>	62
3.2.2	Groth16	64
3.2.2.1	<i>Aritmetização em Groth16</i>	64
3.2.2.2	<i>Diferenças em relação ao Pinocchio</i>	64

3.2.2.3	<i>Cerimônia de Confiança em Groth16</i>	65
3.2.2.4	<i>Geração de Prova em Groth16</i>	66
3.2.2.5	<i>Verificação de Prova em Groth16</i>	67
4	PROTOCOLOS DE AUTENTICAÇÃO COM PROVAS DE CONHE- CIMENTO ZERO	69
4.1	EUROPEAN UNION DIGITAL IDENTITY WALLET	69
4.1.1	Provas de Conhecimento Zero na EUDI Wallet	69
4.1.2	Exemplo de caso de uso	70
4.2	A PRIVACY-PRESERVING BIOMETRIC AUTHENTICATION SYSTEM WITH BINARY CLASSIFICATION IN A ZERO KNOWLEDGE PROOF PROTOCOL	70
4.3	ANÁLISE DAS LACUNAS IDENTIFICADAS	72
5	PROTOCOLO DE AUTENTICAÇÃO BIOMÉTRICA FACIAL COM PROVAS DE CONHECIMENTO ZERO	73
5.1	JUSTIFICATIVA E POSICIONAMENTO DA PROPOSTA	73
5.2	PROPOSTA DO SISTEMA	74
5.3	COMPONENTES DO SISTEMA	75
5.4	FUNCIONAMENTO DO SISTEMA	76
5.4.1	Fase de Registro	76
5.4.2	Fase de Autenticação	78
5.4.3	Propriedades e Suposições Explícitas	80
5.5	RESULTADOS OBTIDOS	81
6	CONCLUSÃO	83
6.1	TRABALHOS FUTUROS	83
	REFERÊNCIAS	85
	APÊNDICE A - CÓDIGO FONTE DA IMPLEMENTAÇÃO	89
	APÊNDICE B - ARTIGO DO TCC	91

1 INTRODUÇÃO

A rápida evolução das tecnologias digitais e a crescente necessidade de proteger dados sensíveis impulsionam o desenvolvimento de técnicas criptográficas cada vez mais avançadas. Nesse contexto, as Provas de Conhecimento Zero (PCZs), introduzidas formalmente por (GOLDWASSER; MICALI; RACKOFF, 1989), destacam-se como uma das inovações mais promissoras. As PCZs permitem que uma parte (o provador) demonstre para outra (o verificador) que uma afirmação é verdadeira, sem revelar nenhuma informação adicional além da veracidade dessa afirmação. Isso é possível graças às três propriedades fundamentais das PCZs: primeiramente, a *completude* garante que, se a afirmação apresentada pelo provador for verdadeira e o protocolo for seguido corretamente, o verificador sempre aceitará a prova. A *solidez* assegura que, se a afirmação for falsa, nenhum provador, mesmo agindo de forma maliciosa, conseguirá convencer o verificador, exceto com probabilidade negligenciável. Por fim, o *conhecimento zero* garante que o verificador não aprende nenhuma informação nova além do fato de que a afirmação é verdadeira, ou seja, nenhuma outra propriedade do problema ou solução é revelada durante a interação. Isso torna as PCZs ferramentas valiosas em diversos cenários onde a privacidade é essencial, como em sistemas de autenticação e em tecnologias *blockchain*.

De maneira geral, as técnicas de PCZs dividem-se em provas interativas e não interativas, cada uma com características distintas que afetam sua aplicabilidade em diferentes contextos. As provas interativas baseiam-se na troca contínua de mensagens entre o provador e o verificador. Por mais que sejam métodos eficazes, elas dependem de comunicação em tempo real, o que limita seu uso em redes distribuídas ou em sistemas com alta latência. Para contornar essas limitações, (BLUM; FELDMAN; MICALI, 1988) introduziu as provas não interativas, que foram desenvolvidas com o intuito de permitir que uma única mensagem convença o verificador, eliminando a necessidade de comunicação constante.

Nos últimos anos, a literatura sobre PCZs tem-se expandido rapidamente, cobrindo desde avanços teóricos até implementações práticas. Este crescimento, no entanto, tem gerado uma dispersão das contribuições, que se encontram espalhadas entre artigos científicos, relatórios técnicos, repositórios de código e implementações de sistemas, dificultando uma visão consolidada e acessível da área. Nesse contexto, torna-se relevante realizar um estudo teórico aprofundado que permita organizar e compreender os fundamentos das PCZs, bem como identificar as premissas de segurança que sustentam diferentes abordagens e os contextos em que podem ser utilizadas.

Este projeto, portanto, tem como objetivo conduzir um estudo teórico sobre os protocolos PCZs mais relevantes da atualidade, com ênfase na análise das premissas criptográficas subjacentes a esquemas já estabelecidos. Em particular, o trabalho foca no estudo de *Zero-Knowledge (zk) Succinct Non-Interactive Arguments of Knowledge* (SNARKs), uma família de protocolos de conhecimento zero sucintos e não interativos introduzida por (BITANSKY et al., 2012). Além disso, para demonstrar a aplicabilidade e o potencial prático dessa tecnologia, o trabalho culmina no desenvolvimento de um sistema de autenticação biométrica facial. A apli-

cação proposta utiliza o protocolo **Groth16**, um zk-SNARK, para validar a identidade de um usuário sem nunca expor seus dados biométricos, enfrentando diretamente o desafio de proteger informações sensíveis.

Um diferencial importante deste projeto está na produção de um material em língua portuguesa, o que amplia a acessibilidade para estudantes e profissionais que desejam aprofundar-se no tema, mas enfrentam barreiras linguísticas em materiais predominantemente disponíveis em inglês. Ademais, trata-se de um projeto autocontido, reunindo todo o embasamento matemático necessário para o entendimento da pesquisa e apresentando os conceitos de forma clara e integrada, o que facilita o aprendizado e a aplicação prática das PCZs.

Ao final, espera-se que este trabalho contribua significativamente para o entendimento e a utilização de PCZs, oferecendo uma referência sólida tanto para estudos acadêmicos quanto para o desenvolvimento de novas soluções criptográficas.

1.1 OBJETIVO GERAL

O objetivo principal desta monografia consiste na realização de um estudo teórico aprofundado sobre PCZs, com ênfase na análise das premissas de segurança subjacentes a diferentes esquemas, especialmente os classificados como zk-SNARKs, bem como desenvolver um sistema de autenticação biométrica facial que preserve a privacidade do usuário, utilizando o protocolo **Groth16**, um zk-SNARK, para validar a identidade sem revelar dados sensíveis.

1.2 OBJETIVOS ESPECÍFICOS

- Estudar os fundamentos teóricos das PCZ do tipo zk-SNARK, com foco nas premissas criptográficas que os sustentam;
- Projetar e implementar um circuito zk-SNARK que encapsule a lógica de verificação de similaridade por cosseno, permitindo a comparação de *embeddings* faciais de forma privada;
- Desenvolver uma aplicação de autenticação completa que integre o modelo de extração de *embeddings* e o protocolo de PCZ, validando sua funcionalidade e eficácia na proteção da privacidade em um cenário prático.

1.3 METODOLOGIA DE PESQUISA

Este trabalho adota uma abordagem metodológica dividida em duas etapas principais: a condução de um estudo teórico sobre os zk-SNARKs e o desenvolvimento de um protocolo de autenticação biométrica baseado na implementação do esquema **Groth16**. O objetivo é integrar análise conceitual e experimentação prática, promovendo uma compreensão mais ampla dos fundamentos e aplicações das PCZs.

Na primeira etapa, foi realizado um levantamento da literatura técnica e científica, com a identificação das premissas de segurança que fundamentam diferentes esquemas classificados como zk-SNARK. O foco desta fase inicial é o estudo dos protocolos **Pinocchio** e **Groth16**.

Na etapa seguinte, foi projetado um sistema de autenticação biométrica facial baseado na técnica de similaridade por cosseno. O protocolo **Groth16** foi selecionado para a implementação de uma aplicação demonstrativa desse sistema de autenticação facial e foram utilizadas bibliotecas criptográficas adequadas ao contexto, buscando evidenciar o funcionamento prático da técnica escolhida.

2 FUNDAMENTAÇÃO TEÓRICA

Os sistemas de PCZs fundamentam-se em um ambiente aritmético definido por estruturas algébricas modernas e conceitos matemáticos abstratos. Essas ferramentas são vitais para o estudo da mecânica interna dos protocolos zk-SNARKs. Com isso, as seções a seguir introduzem as primitivas matemáticas e criptográficas que sustentam os protocolos que serão estudados em capítulos posteriores. Também serão discutidas primitivas de inteligência artificial, que são indispensáveis para compreender o sistema de autenticação biométrica desenvolvido.

2.1 PRIMITIVAS MATEMÁTICAS

Para o início do estudo das PCZs, é necessário entender os **Princípios da Álgebra Abstrata**. Esta área da matemática fornece o formalismo para descrever as operações criptográficas de forma precisa. O ponto de partida para este estudo é a definição das estruturas algébricas fundamentais que dão forma aos protocolos. A primeira e mais básica dessas estruturas é o grupo.

Definição 2.1.1 (Grupo). Um grupo $(\mathbb{G}, \cdot)$ é um conjunto de elementos com uma operação binária denotada por $\cdot$ que associa cada par ordenado $(a, b) \in \mathbb{G} \times \mathbb{G}$ a um elemento $a \cdot b \in \mathbb{G}$, de modo que os seguintes axiomas sejam obedecidos (STALLINGS, 2022).

- Fechamento: Para todos $a, b \in \mathbb{G}$, $a \cdot b$ também está em $\mathbb{G}$.
- Associatividade: Para todos $a, b, c \in \mathbb{G}$, $(a \cdot b) \cdot c = a \cdot (b \cdot c)$.
- Elemento Identidade: Para todo $a \in \mathbb{G}$, existe um elemento $e \in \mathbb{G}$ tal que $a \cdot e = e \cdot a = a$.
- Elemento Inverso: Para todo $a \in \mathbb{G}$, existe um inverso $a^{-1} \in \mathbb{G}$ tal que $a \cdot a^{-1} = e$.

Uma propriedade fundamental de qualquer grupo é o seu tamanho, conhecido como a ordem do grupo.

Definição 2.1.2 (Ordem de um grupo). A ordem do grupo $\mathbb{G}$ é igual ao número de elementos no grupo e é denominada por $|\mathbb{G}|$. Se um grupo possui um número finito de elementos, ele é chamado de grupo finito. Caso contrário, o grupo é um grupo infinito (MENEZES; VANSTONE; OORSCHOT, 1996).

Além dos axiomas básicos, alguns grupos satisfazem propriedades adicionais. Um tipo particularmente importante é o grupo abeliano, que introduz a propriedade da comutatividade.

Definição 2.1.3 (Grupo Abeliano). Um grupo é considerado abeliano se satisfaz a seguinte condição adicional (STALLINGS, 2022).

- Comutatividade: Para todos $a, b \in \mathbb{G}$, $a \cdot b = b \cdot a$.

Dentro da estrutura de um grupo, é comum realizarmos operações repetidas. Isso leva à definição de exponenciação, que é fundamental para os protocolos criptográficos a serem estudados.

Definição 2.1.4 (Exponenciação em um grupo). Define-se exponenciação dentro de um grupo como uma aplicação repetida do operador de grupo, de modo que $a^k = a \cdot a \cdot \dots \cdot a$, k vezes para qualquer inteiro k . Além disso, define-se $a^0 = e$ como o elemento identidade, e $a^{-n} = (a^{-1})^n$, onde a^{-1} é o elemento inverso de a dentro do grupo (STALLINGS, 2022).

O conceito de exponenciação está diretamente ligado à noção de grupos cíclicos. Nesses grupos, um único elemento é capaz de gerar todos os outros elementos do grupo através da exponenciação.

Definição 2.1.5 (Grupo Cíclico). Um grupo $\mathbb{G}$ é cíclico se cada elemento de $\mathbb{G}$ é uma potência g^k , sendo k um inteiro, de um elemento fixo $g \in \mathbb{G}$. Diz-se que o elemento g gera o grupo $\mathbb{G}$ ou é um gerador de $\mathbb{G}$. Um grupo cíclico é sempre abeliano e pode ser finito ou infinito (STALLINGS, 2022).

Enquanto os grupos definem uma estrutura com uma única operação, muitas aplicações criptográficas exigem uma estrutura mais rica que suporte duas operações: adição e multiplicação. Essa estrutura é conhecida como corpo.

Definição 2.1.6 (Corpo). Um corpo $(\mathbb{F}, +, \times)$ é um conjunto de elementos com duas operações binárias, *adição* e *multiplicação*, tais que para todos os $a, b, c \in \mathbb{F}$ os seguintes axiomas são obedecidos (STALLINGS, 2022) e (THALER, 2022).

- Fechamento: Para todos $a, b \in \mathbb{F}$, $a + b$ e $a \times b$ também estão em $\mathbb{F}$.
- Associatividade: Para todos $a, b, c \in \mathbb{F}$, $(a + b) + c = a + (b + c)$ e $(a \times b) \times c = a \times (b \times c)$.
- Comutatividade: Para todos $a, b \in \mathbb{F}$, $a + b = b + a$ e $a \times b = b \times a$.
- Elementos Identidade: Existem elementos distintos $0, 1 \in \mathbb{F}$ tais que para todo $a \in \mathbb{F}$, $a + 0 = a$ e $a \times 1 = a$.
- Elementos Inversos: Para todo $a \in \mathbb{F}$, existe um inverso aditivo $-a \in \mathbb{F}$ tal que $a + (-a) = 0$. Para todo $a \in \mathbb{F}$ com $a \neq 0$, existe um inverso multiplicativo $a^{-1} \in \mathbb{F}$ tal que $a \times a^{-1} = 1$.
- Distributividade: Para todos $a, b, c \in \mathbb{F}$, $a \times (b + c) = (a \times b) + (a \times c)$.

Para aplicações práticas em criptografia, o interesse recai sobre corpos que não possuem um número infinito de elementos. Estes são os corpos finitos.

Definição 2.1.7 (Corpo Finito). Um corpo finito é um corpo $\mathbb{F}$ que contém uma quantidade finita de elementos. A ordem de $\mathbb{F}$ é o número de elementos em $\mathbb{F}$ e é definida como $|\mathbb{F}|$. (MENEZES; VANSTONE; OORSCHOT, 1996). Se $\mathbb{F}$ é um corpo finito, então $\mathbb{F}$ contém p^m elementos para algum primo p e inteiro $m \geq 1$ (MENEZES; VANSTONE; OORSCHOT, 1996).

Em essência, um corpo é um conjunto de elementos nos quais é possível realizar operações de adição, subtração, multiplicação e divisão sem sair do conjunto (STALLINGS, 2022).

A segurança de muitos sistemas criptográficos baseia-se na dificuldade computacional de reverter certas operações dentro dessas estruturas. Um dos problemas fundamentais que surgem em grupos multiplicativos de corpos finitos é o **Problema do Logaritmo Discreto**.

Problema 2.1.1 (Logaritmo Discreto). Seja p um número primo e seja $\mathbb{G} = \mathbb{F}_p^*$ o grupo multiplicativo de um corpo finito, que consiste em seus $p - 1$ elementos não nulos, seja g um gerador de $\mathbb{G}$ e seja h um elemento de $\mathbb{G}$. O Problema do Logaritmo Discreto (DLP) consiste em encontrar o único inteiro x tal que $g^x = h$ (MENEZES; VANSTONE; OORSCHOT, 1996).

Este valor x é chamado de *logaritmo discreto* de h na base g . A operação de exponenciação g^x é computacionalmente fácil de realizar, mesmo para valores grandes de x . No entanto, encontrar x a partir de g e h é considerada computacionalmente inviável para corpos finitos de tamanho apropriado.

No contexto das PCZs, os conceitos de grupos e a intratabilidade do DLP são aplicados de forma mais eficiente. Ao invés de se basearem em grupos multiplicativos de corpos finitos genéricos (como $\mathbb{F}_p^*$), os protocolos modernos são implementados sobre estruturas algébricas chamadas **Curvas Elípticas**. Elas fornecem grupos onde o DLP é significativamente mais difícil, permitindo níveis de segurança equivalentes com chaves muito menores.

Definição 2.1.8 (Curvas Elípticas). Uma curva elíptica é o conjunto de pontos (x, y) que satisfazem uma equação cúbica específica. A forma mais comum é (MILLER, 1986):

$$y^2 = x^3 + ax + b$$

onde a e b são constantes e

$$4a^3 + 27b^2 \neq 0$$

Enquanto as curvas elípticas podem ser estudadas sobre os números reais, para a criptografia, elas são definidas sobre um corpo finito $\mathbb{F}_p$. Isso significa que as constantes a e b , bem como as coordenadas x e y de cada ponto na curva, são elementos do corpo finito $\mathbb{F}_p$ (MILLER, 1986). O conjunto de todos os pontos (x, y) que satisfazem a equação sobre $\mathbb{F}_p$, juntamente com um ponto especial chamado “ponto no infinito” O , forma um grupo abeliano finito sob uma operação de adição de pontos.

Esta operação de “adição” não é a adição aritmética padrão, mas sim uma operação geométrica definida sobre os pontos da curva, conforme descrito a seguir.

Definição 2.1.9 (Soma de dois pontos distintos). Para somar dois pontos distintos $P = (x_P, y_P)$ e $Q = (x_Q, y_Q)$ na curva, traça-se uma linha reta através deles. Devido à natureza cúbica da equação da curva, esta linha irá interceptar a curva em exatamente um terceiro ponto, que denotamos por $-R$. A soma $P + Q$ é então definida como o ponto R , que é a reflexão de $-R$ sobre o eixo x . Algebricamente, isso corresponde a (SEROUSSI, 1999):

$$\lambda = \frac{y_Q - y_P}{x_Q - x_P}, \quad x_R = \lambda^2 - x_P - x_Q, \quad y_R = \lambda(x_P - x_R) - y_P$$

A definição anterior cobre o caso em que os pontos P e Q são distintos. Um caso especial dessa operação, conhecido como “dobra” do ponto, ocorre quando $P = Q$.

Definição 2.1.10 (Soma de dois pontos iguais). Para dobrar um ponto P , ou seja, calcular $P + P$, a linha secante se torna uma linha tangente à curva no ponto P . O processo é o mesmo: encontra-se o outro ponto de intersecção da tangente com a curva e reflete-se sobre o eixo x . As fórmulas são ligeiramente diferentes (SEROUSSI, 1999):

$$\lambda = \frac{3x_P^2 + a}{2y_P}, \quad x_R = \lambda^2 - 2x_P, \quad y_R = \lambda(x_P - x_R) - y_P$$

Com base nas operações de adição de pontos distintos e de dobra de ponto, pode-se definir a multiplicação escalar, ou produto, de um ponto por um inteiro k .

Definição 2.1.11 (Produto entre um ponto e um inteiro). A multiplicação entre o ponto P e o inteiro k é definida como a adição repetida de um ponto P a si mesmo k vezes, ou seja, $k \cdot P = P + P + \dots + P$, k vezes (KOBLOITZ, 1987).

A operação de multiplicação escalar $Q = k \cdot P$ é computacionalmente eficiente. No entanto, o problema inverso (encontrar o escalar k dados os pontos P e Q) é considerado computacionalmente intratável. Essa assimetria é a base de segurança da Criptografia de Curvas Elípticas (ECC) e é formalizada no **Problema do Logaritmo Discreto em Curvas Elípticas**.

Problema 2.1.2 (Logaritmo Discreto em Curvas Elípticas). A segurança da ECC repousa na dificuldade do Problema do Logaritmo Discreto em Curvas Elípticas (ECDLP). Formalmente, dados dois pontos P e Q no grupo da curva elíptica, onde P é um gerador de um subgrupo, o ECDLP consiste em encontrar o inteiro k tal que $Q = k \cdot P$.

Este problema é o análogo do DLP em corpos finitos, mas em um contexto diferente. A estrutura geométrica e a operação de adição de pontos não se prestam às mesmas técnicas de fatoração que tornam o DLP em corpos finitos vulnerável (KOBLOITZ, 1987).

Embora a intratabilidade do ECDLP seja fundamental para a segurança, certos protocolos zk-SNARKs mais avançados exigem uma ferramenta adicional que possa “mover” o problema de um grupo para outro de forma controlada. Essa ferramenta é o emparelhamento bilinear.

Definição 2.1.12 (Emparelhamentos). Um emparelhamento bilinear é uma função e de mapeamento não trivial de pares de elementos de dois grupos cíclicos $\mathbb{G}_1$ e $\mathbb{G}_2$ para um terceiro grupo cíclico $\mathbb{G}_T$ (BOWE; GABIZON; GREEN, 2017):

$$e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$$

Em contextos criptográficos, $\mathbb{G}_1$ e $\mathbb{G}_2$ são tipicamente grupos aditivos de pontos em uma curva elíptica, enquanto $\mathbb{G}_T$ é um grupo multiplicativo de elementos em um corpo finito. Todos os três grupos têm a mesma ordem prima p .

Para ser criptograficamente útil, o emparelhamento deve satisfazer a propriedade de bilinearidade.

Definição 2.1.13 (Bilinearidade). Para quaisquer geradores $g_1 \in \mathbb{G}_1$, $g_2 \in \mathbb{G}_2$, $g_T \in \mathbb{G}_T$ e quaisquer escalares $a, b \in \mathbb{F}_p$, a propriedade de bilinearidade é definida como (BOWE; GABIZON; GREEN, 2017):

$$e(a \cdot g_1, b \cdot g_2) = (g_T)^{ab}$$

2.1.1 Circuitos Aritméticos e o Lema de Schwartz-Zippel

Esta seção detalha dois conceitos que conectam as estruturas algébricas abstratas a um protocolo zk-SNARK prático. Os **Circuitos Aritméticos** fornecem o método para representar qualquer programa computacional como um conjunto de equações algébricas. Por sua vez, o **Lema de Schwartz-Zippel** é a ferramenta matemática que permite verificar essas equações de forma sucinta, garantindo a eficiência dos protocolos.

Definição 2.1.14 (Circuito Aritmético). Um circuito aritmético é um grafo acíclico direcionado, com nós como portas de adição e multiplicação, e arestas como fios, sobre um corpo finito $\mathbb{F}_p$. Fios conectam as saídas de uma porta às entradas de outra. Cada porta possui duas entradas e um fio de saída, e o circuito possui um fio de saída final. A Figura 1 é um exemplo de um circuito aritmético (CHEN et al., 2023):

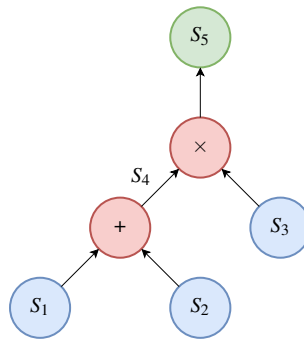


Figura 1 – Circuito aritmético para $f(s_1, s_2, s_3) = (s_1 + s_2) \times s_3$ segundo (CHEN et al., 2023)

Para que um zk-SNARK possa provar o conhecimento da computação representada pelo circuito, ela precisa ser descrita algebricamente. Isso é feito através de um sistema de restrições.

Definição 2.1.15 (Restrições). Para um circuito de m portas e n fios, definem-se como **restrições** um conjunto de m elementos que modelam as relações que as portas de adição e multiplicação de um circuito impõem sobre seus fios de entrada e saída (CHEN et al., 2023).

Dada a estrutura de restrições que define o circuito, a “testemunha” (*witness*) é a atribuição de valores específicos aos fios que satisfaz todas essas restrições.

Definição 2.1.16 (Testemunha). Para um circuito de m portas e n fios, define-se como testemunha $s = (s_1, \dots, s_n)$ como sendo as atribuições aos n fios do circuito, de modo que as entradas e saídas de cada porta satisfaçam a **restrição** definida pela operação da porta (CHEN et al., 2023).

Uma vez que uma computação é representada como um circuito aritmético com suas restrições, o desafio central dos zk-SNARKs é permitir que uma entidade, o provador, convença uma outra entidade, o verificador, de que ele conhece uma testemunha válida que satisfaz todas as restrições do circuito.

Para alcançar a **sucintez** (o ‘S’ de SNARK), os protocolos não verificam cada restrição individualmente. Em vez disso, eles utilizam uma técnica que converte todo o sistema de restrições do circuito em uma única identidade entre três polinômios. O problema de verificar o circuito torna-se, então, um problema de verificar se uma equação polinomial é válida. Essa técnica de conversão será estudada mais a fundo em capítulos posteriores, pois o objetivo agora é apresentar como o **Lema de Schwartz-Zippel** garante a igualdade entre dois polinômios de maneira sucinta.

Verificar essa igualdade polinomial de forma completa exigiria comparar todos os coeficientes, o que não seria sucinto. É aqui que a verificação probabilística se torna essencial. Em vez de comparar os polinômios inteiros, o verificador simplesmente avalia os polinômios em um ponto aleatório e verifica se a identidade se mantém.

O **Lema de Schwartz-Zippel** estabelece um limite superior para a probabilidade de que um polinômio não nulo assumo o valor zero em uma avaliação aleatória. Esse resultado é fundamental em protocolos de verificação probabilística, pois permite testar a igualdade de polinômios de maneira sucinta e com erro desprezível.

Lema 2.1.1 (Schwartz-Zippel). *Seja $P(x_1, \dots, x_n)$ um polinômio não nulo de grau d sobre um corpo $\mathbb{F}$, e seja $S \subseteq \mathbb{F}$ um subconjunto não vazio. Então:*

$$\Pr[P(r_1, \dots, r_n) = 0] \leq \frac{d}{|S|}$$

onde $r_1, \dots, r_n$ são elementos aleatórios de S e $|S|$ é a ordem de S (ATSERIAS; TZAMERET, 2024).

Esse resultado, juntamente com o fato de que dois polinômios distintos de grau no máximo d podem coincidir em, no máximo, d pontos distintos, garante que ao avaliar dois polinômios em um ponto selecionado aleatoriamente dentro de um conjunto grande S , a probabilidade de coincidirem é limitada por $d/|S|$ (PETKUS, 2019).

Exemplo 2.1.1. Por exemplo, ao considerar que x seja escolhido em um intervalo de inteiros compreendido entre 1 e 10^{77} , o número de pontos em que as avaliações dos dois polinômios diferem é igual a $10^{77} - d$. Assim, a probabilidade de que o valor de x selecionado aleatoriamente coincida com um dos d pontos em que os polinômios são iguais é dada por $\frac{d}{10^{77}}$, que é um valor considerado desprezível (PETKUS, 2019).

2.2 PRIMITIVAS CRIPTOGRÁFICAS

Apoiada nas fundações matemáticas introduzidas anteriormente, esta seção introduz as primitivas criptográficas que utilizam essas estruturas para construir sistemas seguros. O foco são dois dos objetivos centrais da segurança da informação: a confidencialidade, que impede o acesso não autorizado aos dados, e a integridade, que garante que os dados não foram alterados. Para isso, serão apresentados os conceitos de esquemas de criptografia e de funções de resumo criptográfico.

2.2.1 Esquemas Criptográficos e Funções Hash

Definição 2.2.1 (Esquema de criptografia). Um esquema de criptografia com um espaço de chaves K consiste em um conjunto $\{E_e : e \in K\}$ de transformações de criptografia e um conjunto correspondente $\{D_d : d \in K\}$ de transformações de descryptografia com a propriedade de que para cada $e \in K$ existe uma chave única $d \in K$ tal que $D_d = E_e^{-1}$. Ou seja, $D_d(E_e(m)) = m$ para todo $m \in M$, onde M é o conjunto de mensagens (MENEZES; VANSTONE; OORSCHOT, 1996).

Um esquema de criptografia pode também ser chamado de cifra. As chaves e e d na definição anterior são chamadas de “par de chaves”. Observe que e e d podem ser iguais. Para construir um esquema de criptografia, é necessário selecionar um espaço de mensagem M , um espaço de texto cifrado C , um espaço de chave K , um conjunto de transformações de criptografia $\{E_e : e \in K\}$ e um conjunto correspondente de transformações de descryptografia $\{D_d : d \in K\}$ (MENEZES; VANSTONE; OORSCHOT, 1996).

A principal distinção entre os tipos de esquemas de criptografia reside na relação entre a chave de criptografia (e) e a de descryptografia (d). Essa relação define duas categorias fundamentais: criptografia de chave simétrica e de chave pública (ou assimétrica).

Definição 2.2.2 (Criptografia simétrica). Considere um esquema de criptografia que consiste nos conjuntos de transformações de criptografia e descryptografia $\{E_e : e \in K\}$ e $\{D_d : d \in K\}$, respectivamente, onde K é o espaço de chaves. O esquema de criptografia é considerado de chave simétrica se, para cada par de chaves de criptografia/descryptografia associado (e, d) , for computacionalmente viável determinar d conhecendo apenas e , e determinar e a partir de d . Como $e = d$ nos esquemas práticos de criptografia de chave simétrica, o termo chave simétrica torna-se apropriado (MENEZES; VANSTONE; OORSCHOT, 1996).

A principal limitação da criptografia simétrica é a necessidade de um canal seguro preexistente para que as partes possam compartilhar a chave secreta e (ou d , nesse caso). Para resolver este problema de distribuição de chaves, a criptografia assimétrica, ou de chave pública, introduz um paradigma diferente, utilizando um par de chaves matematicamente relacionadas, mas distintas.

Definição 2.2.3 (Criptografia assimétrica). Considere um esquema de criptografia que consiste nos conjuntos de transformações de criptografia e descriptografia $\{E_e : e \in K\}$ e $\{D_d : d \in K\}$, respectivamente. O método de criptografia é considerado um esquema de criptografia de chave pública se, para cada par de criptografia/descriptografia associado (e, d) , uma chave e , a chave pública, for disponibilizada publicamente, enquanto a outra d , a chave privada, for mantida em segredo. Para que o esquema seja seguro, ele deve ser computacionalmente inviável calcular d a partir de e (MENEZES; VANSTONE; OORSCHOT, 1996).

Os esquemas de criptografia, tanto simétricos quanto assimétricos, são projetados primariamente para garantir a confidencialidade da informação, transformando uma mensagem legível em um formato ininteligível. No entanto, a criptografia abrange outros objetivos essenciais, como a garantia de integridade dos dados, ou seja, a capacidade de verificar que uma mensagem não foi corrompida ou maliciosamente alterada durante a transmissão.

Para atender a essa necessidade, utiliza-se uma primitiva criptográfica conhecida como **função de resumo criptográfico**.

Definição 2.2.4 (Função *Hash*). Uma função de resumo criptográfico, que também pode ser denominada como “função *hash*”, H aceita um bloco de dados m de comprimento variável como entrada e produz um resultado de tamanho fixo $h = H(m)$, denominado “valor *hash*”. Essa função tem a propriedade de que os resultados da aplicação da função a um grande conjunto de entradas produzirão saídas uniformemente distribuídas e aparentemente aleatórias. Ou seja, uma alteração em qualquer *bit* em m resulta em uma alteração no valor *hash*. Além disso, é computacionalmente inviável calcular (STALLINGS, 2022):

1. Um bloco de dados x que mapeia para um resultado *hash* pré-especificado h de modo que $h = H(x)$.
2. Um bloco de dados y que mapeia para um mesmo resultado *hash* de um bloco de dados x pré-especificado, de modo que $x \neq y$ e $H(x) = H(y)$.
3. Dois blocos de dados x e y que mapeiam para o mesmo resultado *hash* de modo que $H(x) = H(y)$.

Em termos gerais, o principal objetivo de uma função *hash* criptográfica é a integridade dos dados. Devido às suas características, funções *hash* são frequentemente usadas para determinar se os dados foram alterados ou não (STALLINGS, 2022).

2.2.2 Provas de Conhecimento

A noção de “prova de conhecimento” foi introduzida por (GOLDWASSER; MICALI; RACKOFF, 1989), sendo posteriormente formalizada por (BELLARE; GOLDREICH, 1993). Intuitivamente, um protocolo entre duas partes constitui um “sistema para provas de conhecimento” se “em algum momento” uma parte (chamada de verificador) estiver “convencida” de que a outra parte (chamada de provador) de fato “sabe” “algo”.

Para demonstrar essa ideia, considera-se uma relação binária $R \subseteq \{0, 1\}^* \times \{0, 1\}^*$. Um sistema de prova de conhecimento para R é definido por um conjunto de interações entre um provador P e um verificador V , onde V aceita uma interação, com uma probabilidade de erro e , apenas se P realmente “conhece” uma testemunha w válida para a relação R .

A interação entre P e V em uma entrada comum $x \in \{0, 1\}^*$ consiste em uma sequência de “movimentos” em cada um dos quais um participante envia uma mensagem ao outro. Os participantes alternam movimentos e, para simplificar, assume-se que o provador move primeiro e o verificador por último.

2.2.2.1 Componentes de um Sistema de Prova de Conhecimento

A fim de proporcionar uma melhor compreensão dos sistemas de prova de conhecimento, esta seção apresenta os componentes necessários para a execução correta de um protocolo, de acordo com os estudos de (GOLDWASSER; MICALI; RACKOFF, 1989) e (BELLARE; GOLDREICH, 1993).

O ponto central da interação é a entrada x . No contexto de provas de conhecimento, a entrada x representa o dado sobre o qual P deve demonstrar conhecimento ou uma propriedade específica. No contexto de relações binárias, x é descrito como uma definição de regras, um exemplo são problemas de decisão em que é verificado se uma cadeia de caracteres pertence a uma linguagem.

Juntamente com a entrada, há a testemunha w , que é uma informação auxiliar que serve para demonstrar a veracidade de uma afirmação, sendo utilizada em conjunto com a entrada x para permitir que V valide se determinada propriedade ou afirmação é verdadeira. No contexto de relações binárias, se x é um problema de decisão, então w é a solução que satisfaz x .

Os participantes do protocolo são o provador P e o verificador V . O provador P é modelado como uma máquina de Turing (pode ser computacionalmente limitada) que tenta convencer V de que possui o conhecimento sobre uma testemunha w relacionada à entrada x tal que $(x, w) \in R$. Por sua vez, o verificador V é modelado como uma máquina de Turing que verifica se P possui o conhecimento sobre w que satisfaça x em relação à R com uma probabilidade de erro e . Este erro é o erro de conhecimento e , uma medida da probabilidade de que V aceite uma interação em que P não possui o conhecimento genuinamente.

2.2.2.2 Interações do Protocolo de Prova de Conhecimento

Como apresentado por (GOLDWASSER; MICALI; RACKOFF, 1989), os dois participantes de um protocolo geralmente são modelados como máquinas de Turing interativas. Porém, nessa seção, eles serão introduzidos como *funções interativas*, seguindo a formalização de (BELLARE; GOLDREICH, 1993). A ideia é separar o aspecto computacional da entidade de seu comportamento de entrada/saída. Isso facilita o esclarecimento das definições posteriores.

Durante a interação entre o provador P e o verificador V , denota-se por α_i a mensagem enviada por P em seu i -ésimo movimento. Similarmente, utiliza-se β_i para denotar a mensagem de V . Assume-se que cada mensagem seja definida pela função interativa citada anteriormente (P ou V) como uma função dependente da entrada comum x e das mensagens anteriores.

Mais precisamente:

$$\alpha_i = P_x(\alpha_1 \beta_1 \cdots \alpha_{i-1} \beta_{i-1}) (i = 1, 2, \dots)$$

$$\beta_i = V_x(\alpha_1 \beta_1 \cdots \alpha_{i-1} \beta_{i-1} \alpha_i) (i = 1, 2, \dots)$$

Adota-se a convenção de que há símbolos especiais que uma função interativa pode emitir para indicar coisas como aceitação ou rejeição. Assume-se que existe uma função $t_V(\cdot)$ (o número de “rodadas”) tal que o $t_V(x)$ -ésimo movimento do verificador contém seu veredito sobre aceitação ou rejeição. A transcrição da interação, denotada $tr_{P,V}(x)$, é a mensagem que registra a conversa até o veredito do verificador. Ou seja, $tr_{P,V}(x) = \alpha_1 \beta_1 \cdots \alpha_{t_V(x)} \beta_{t_V(x)}$.

A transcrição da interação entre P e V contém a sequência de mensagens trocadas durante a interação, mas não informações que estão disponíveis apenas para uma parte, como a entrada auxiliar w ou suas variáveis aleatórias, a menos que tenham sido enviadas para a outra parte. Essas variáveis aleatórias são definidas sobre as escolhas probabilísticas de ambas as funções interativas.

Como se assume que a transcrição contém o veredito do verificador sobre aceitar ou rejeitar, é possível, para cada entrada comum x , falar do conjunto de transcrições aceitantes, denotado $ACC_V(x)$, e do conjunto de transcrições rejeitantes, denotado $REJ_V(x)$. Assim, a probabilidade de que o verificador aceite é, por definição, $Pr[tr_{P,V}(x) \in ACC_V(x)]$. O protocolo chega ao fim quando o verificador entra em um estado de aceitação, ou seja, no momento em que V estiver “convencido” da prova de P .

A Figura 2 apresenta o diagrama da arquitetura de um sistema de prova de conhecimento interativo de acordo com (GOLDWASSER; MICALI; RACKOFF, 1989). Nesse diagrama, A e B representam as máquinas de Turing, as setas com o símbolo R representam as interações com as fitas de apenas leitura, as setas com o símbolo W representam as interações com as fitas de apenas escrita e as setas sem símbolo representam as interações com as fitas de leitura e escrita.

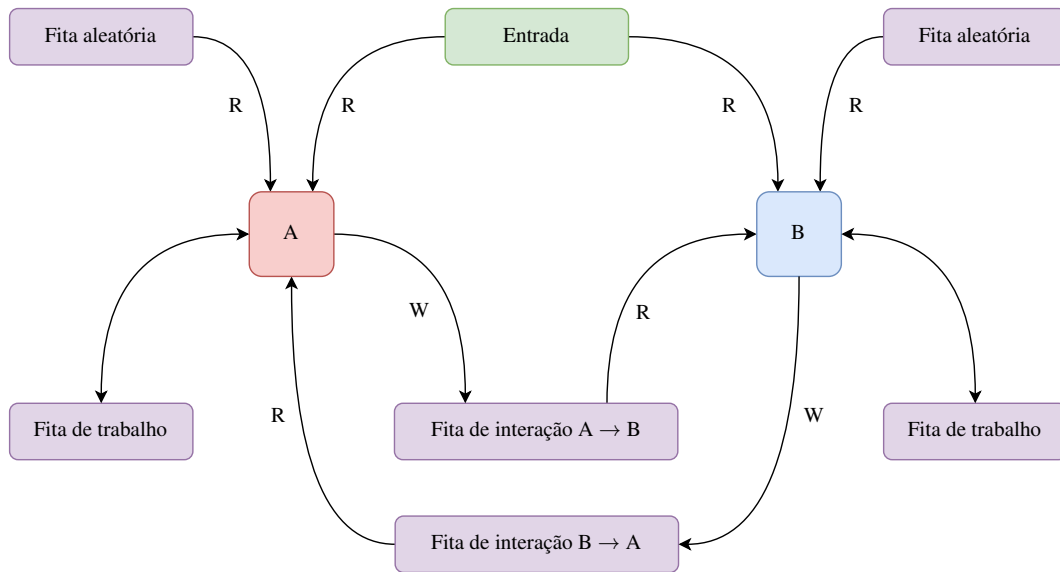


Figura 2 – Arquitetura de um protocolo interativo baseado no diagrama de (GOLDWASSER; MICALI; RACKOFF, 1989)

2.2.2.3 Extrator de Conhecimento

A ideia de que um provador “sabe” algo, no contexto das provas de conhecimento, serve para formalizar a noção de que ele detém informações capazes de sustentar uma afirmação diante de um verificador. Segundo (BELLARE; GOLDREICH, 1993), esse “saber” vai além do simples conhecimento teórico, pois trata-se de uma capacidade demonstrável e operacional.

Essa noção pode ser entendida, primeiramente, como a posse de uma testemunha. Em termos práticos, isso significa que o provador possui uma evidência ou prova relacionada diretamente à entrada do problema. Por exemplo, no caso de uma fórmula booleana, “saber” implica que o provador conhece uma atribuição de valores que satisfaz essa fórmula.

Além disso, o conhecimento em questão envolve a habilidade de transferir essa informação de forma convincente. O provador não apenas detém a informação, mas também consegue utilizá-la para convencer o verificador. Isso se concretiza através de respostas, interações ou soluções que comprovam de forma eficaz o conhecimento declarado.

No plano formal, a teoria das provas de conhecimento introduz o conceito de *extração de conhecimento*. Um extrator de conhecimento é um mecanismo hipotético que, ao interagir com o provador, é capaz de recuperar a testemunha correspondente. Essa capacidade é o que distingue um provador que realmente detém uma prova válida: sua informação pode ser extraída de maneira eficiente por meios definidos.

De forma mais técnica, o extrator de conhecimento pode ser entendido como um algoritmo ou máquina de Turing que, ao interagir com um provador, deve ser capaz de obter uma testemunha válida por meio de um número limitado de trocas de mensagens. Esse processo tem como finalidade garantir que, sempre que o provador consegue convencer o verificador, é possível extrair uma prova correspondente de forma eficiente. Isso serve como evidência de que o provador não apenas foi convincente, mas, de fato, possuía o conhecimento necessário.

Frequentemente, o extrator é modelado como tendo acesso a um “oráculo”, que funciona como uma interface com o provador. Essa abordagem permite tratá-lo como uma caixa-preta capaz de responder a perguntas específicas, facilitando o processo de obtenção de dados relevantes. Com base nas informações coletadas, e considerando também o contexto da entrada do problema, o extrator processa essas respostas para construir uma testemunha válida.

A eficiência do extrator está diretamente relacionada à taxa de sucesso do provador. Quanto maior a probabilidade de o provador convencer o verificador, mais rápido e com maior probabilidade o extrator será capaz de realizar a extração da testemunha.

2.2.3 Prova de Conhecimento Zero Interativa

Esta seção apresenta o conceito de protocolo interativo de PCZ, onde há múltiplas rodadas de comunicação entre o provador e o verificador. Esse tipo de protocolo possui importância histórica na evolução do conceito das PCZs e oferece uma base para a compreensão de seus princípios fundamentais.

Para ilustrar um sistema de PCZ interativo, (GOLDREICH; MICALI; WIGDERSON, 1991) utiliza o problema de coloração de grafos como exemplo prático. Mais especificamente, a obra apresenta um protocolo para o problema no qual se deseja verificar se é possível colorir os vértices de um grafo com, no máximo, três cores, de forma que nenhum par de vértices adjacentes compartilhe a mesma cor.

Problema 2.2.1 (Coloração de grafos). Seja $G = (V, E)$ um grafo com conjunto de vértices $V = \{1, \dots, n\}$, o Problema da Coloração de Grafos indaga se existe uma função de coloração $\phi : V \rightarrow \{1, 2, 3\}$ tal que $\phi(u) \neq \phi(v)$ para toda aresta $\{u, v\} \in E$ (GOLDREICH; MICALI; WIGDERSON, 1991).

Esse protocolo é executado por dois participantes: o provador, que possui uma coloração válida do grafo, e o verificador, que busca se convencer dessa validade. O objetivo central do protocolo é permitir que o provador prove a existência de uma coloração válida sem revelar nenhuma informação sobre ela, exceto sua validade.

A Figura 3 demonstra o grafo que será utilizado como exemplo para a explicação do protocolo apresentado.

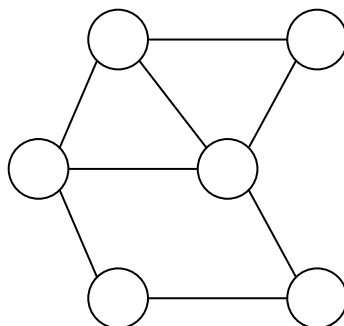


Figura 3 – Grafo de exemplo sem coloração

2.2.3.1 Comprometimento com a coloração

O provador começa com uma coloração válida $\phi : V \rightarrow \{1, 2, 3\}$ para os vértices do grafo $G = (V, E)$, que possui n vértices e m arestas. Para proteger essa coloração, ele aplica uma permutação aleatória π às cores, resultando em uma nova coloração permutada. Em seguida, para cada vértice v , o provador escolhe uma chave criptográfica r_v e calcula o valor criptografado $F_v = f(\pi(\phi(v)), r_v)$, utilizando uma função de criptografia segura f . A sequência $F_1, F_2, \dots, F_n$ é enviada ao verificador, funcionando como um compromisso criptográfico com a coloração, ocultando suas cores reais.

A Figura 4 (a) demonstra um exemplo de coloração válida para o grafo da Figura 3. Perceba que, por mais que a Figura 4 (b) apresente uma permutação dessa coloração, ambas as figuras correspondem à mesma solução, chamada de solução 1 neste exemplo.

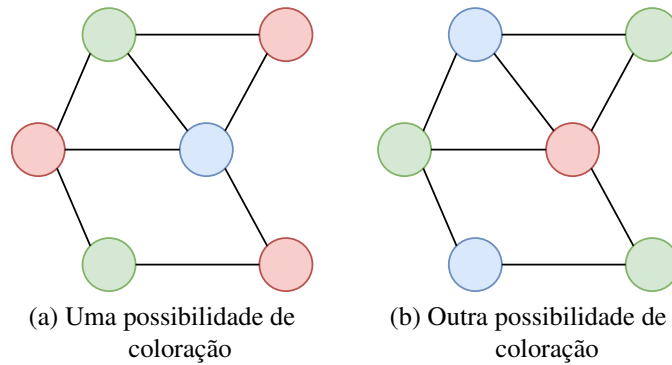


Figura 4 – Grafos representando a solução 1

Em seguida, o verificador seleciona aleatoriamente uma aresta $e = (u, v) \in E$ e solicita que o provador revele as cores dos vértices dessa aresta. O provador responde revelando $\pi(\phi(v))$, $\pi(\phi(u))$ e os respectivos valores aleatórios r_v e r_u utilizados para descriptografar F_v e F_u . Isso permite que o verificador confirme que:

- As cores reveladas correspondem às codificações previamente enviadas.
- As cores dos vértices adjacentes são diferentes, conforme exigido por uma coloração válida.
- Ambas as cores estão dentro do conjunto permitido $\{1, 2, 3\}$.

A Figura 5 (a) apresenta uma possibilidade de escolha de nós adjacentes com base na coloração da Figura 4 (a). Já a Figura 5 (b) possui outra possível escolha de nós adjacentes, porém com base na coloração permutada da Figura 4 (b).

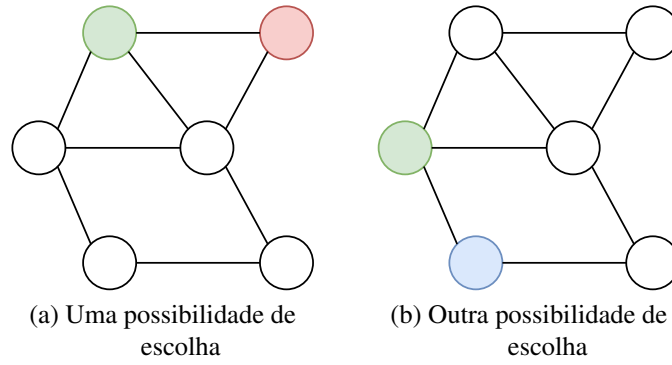


Figura 5 – Escolha de nós adjacentes da solução 1

O provador e o verificador devem repetir o procedimento de revelação dos vértices adjacentes, aplicando sucessivas permutações e recriptografias, até que o verificador esteja convencido da validade da solução. Devem ser realizadas pelo menos m^2 interações para garantir confiabilidade estatística.

2.2.3.2 Propriedades das Prova de Conhecimento Zero

Como observado, o verificador realiza todas as verificações mencionadas e aceita ou rejeita a iteração com base na consistência e validade da revelação. Dessa forma, o protocolo de (GOLDREICH; MICALI; WIGDERSON, 1991) é um sistema de prova de conhecimento zero interativo pois satisfaz as condições de **completude**, **solidez** e **conhecimento zero**.

Definição 2.2.5 (Completude). Seja R uma relação binária da linguagem $L \subseteq \{0, 1\}^*$, um par de máquinas de Turing (P, V) apresenta a propriedade de **completude** sobre L se, para qualquer entrada $x \in L$ e polinômio $p(\cdot)$ (WU; WANG, 2014),

$$Pr[(P, V)(x) = 1] \geq 1 - \frac{1}{p(|x|)}.$$

Ou seja, se o grafo satisfaz o problema e o provador age honestamente, todas as iterações serão aceitas pelo verificador garantidamente.

Definição 2.2.6 (Solidez). Seja R uma relação binária da linguagem $L \subseteq \{0, 1\}^*$, um par de máquinas de Turing (P, V) apresenta a propriedade de **solidez** sobre L se, para qualquer entrada $x \notin L$ e polinômio $p(\cdot)$ (WU; WANG, 2014),

$$Pr[(P, V)(x) = 1] < 1 - \frac{1}{p(|x|)}.$$

Se o grafo não satisfaz o problema, o provador não conseguirá evitar a detecção de inconsistências em algumas iterações. A chance de um verificador não detectar um erro após m^2 iterações é exponencialmente pequena.

Definição 2.2.7 (Conhecimento zero). Seja R uma relação binária da linguagem $L \subseteq \{0, 1\}^*$, um par de máquinas de Turing (P, V) apresenta a propriedade de **conhecimento zero** sobre L

se, para qualquer entrada $x \in L$ e polinômio $p(\cdot)$, existe um algoritmo de tempo polinomial probabilístico M tal que, para qualquer $x \in L$ (WU; WANG, 2014),

$$(P, V)(x) \approx M(x).$$

A segurança da função de criptografia e a permutação aleatória das cores em cada rodada garantem que o verificador não aprende nada sobre a coloração original. Cada iteração é estatisticamente independente, e mesmo ao final de todas as execuções, o verificador não é capaz de reconstruir nenhuma parte da coloração.

Segundo (FEIGE; SHAMIR, 1990), um protocolo possui uma testemunha indistinguível se a parte verificadora não consegue discernir qual das várias testemunhas secretas (soluções) a parte provadora está utilizando para demonstrar uma afirmação.

Definição 2.2.8 (Indistinguibilidade de testemunha). Formalmente, um sistema de prova (P, V) possui uma testemunha indistinguível sobre uma relação R de uma linguagem L se, para qualquer verificador V' , qualquer entrada x , e quaisquer duas testemunhas w_1 e w_2 pertencentes ao conjunto de testemunhas $w(x)$, os registros das interações geradas por V' ao interagir com P usando w_1 e P usando w_2 são polinomialmente indistinguíveis. Ou seja (WU; WANG, 2014),

$$Pr[(P(w_1), V)(x)] \approx Pr[(P(w_2), V)(x)]; x \in L; w_1, w_2 \in w(x).$$

Com isso, tem-se a Figura 6, que representa a solução 2 e é indistinguível da solução 1 apresentada na Figura 4 (a), caso o protocolo de (GOLDREICH; MICALI; WIGDERSON, 1991) seja executado corretamente. Note que a solução 2 não é uma permutação da solução 1, pois elas são testemunhas diferentes.

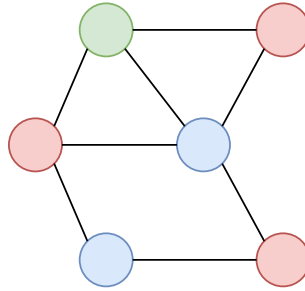


Figura 6 – Grafo representando a solução 2

2.2.4 Esquema de Comprometimento Polinomial

Um esquema de comprometimento polinomial permite que uma parte se comprometa com um polinômio, mantendo-o oculto, mas verificável (KATE; ZAVERUCHA; GOLDBERG, 2010). Esta seção apresenta a base dos esquemas de comprometimento polinomial utilizados pelos protocolos zk-SNARKs estudados neste trabalho. Para isso, será demonstrado o esquema de (PETKUS, 2019).

Segundo (PETKUS, 2019), pelo *Teorema Fundamental da Álgebra*, qualquer polinômio pode ser fatorado em termos lineares. Assim, considerando o polinômio $P(x) = c_0 + c_1x +$

$\dots + c_d x^d$, provar que $P(x)$ possui as raízes de um polinômio alvo $T(x)$ equivale a mostrar que a divisão $H(x) = P(x)/T(x)$, para algum polinômio $H(x)$, não tem resto.

De acordo com o **Lema 2.1.1**, a verificação em um único ponto s é suficiente para o comprometimento polinomial. Assim, o verificador escolhe s , computa $T(s)$ e solicita $P(s)$ e $H(s)$. No entanto, o provador pode forjar respostas, por isso utiliza-se uma **transformação homomórfica**:

$$g^{a \cdot b} = (g^a)^b, \quad g^{a+b} = g^a \cdot g^b.$$

Nesse caso, g é o gerador de um grupo cíclico $\mathbb{G}_p$, sendo as operações aritméticas realizadas em módulo p , um número primo. De acordo com o Problema 2.1.1, a dificuldade de inverter $g^{s \bmod p}$ garante a segurança. O protocolo é descrito por:

1. Verificador envia $g^{s^0}, g^{s^1}, \dots, g^{s^d}$
2. Provador calcula $g^{P(s)} = (g^{s^0})^{c_0} \cdot (g^{s^1})^{c_1} \dots (g^{s^d})^{c_d} = g^{c_0 \cdot s^0} \cdot g^{c_1 \cdot s^1} \dots g^{c_d \cdot s^d}$ e, similarmente, $g^{H(s)}$
3. Verificador calcula $T(x)$ e analisa se $g^{P(s)} = (g^{H(s)})^{T(s)} = g^{H(s) \cdot T(s)}$

Para impedir a forja de provas, o verificador escolhe aleatoriamente um valor α e envia as ocultações $g^{s^0}, g^{s^1}, \dots, g^{s^d}$ e também $g^{\alpha \cdot s^0}, g^{\alpha \cdot s^1}, \dots, g^{\alpha \cdot s^d}$. O provador retorna $a = g^{P(s)}$ e $b = g^{\alpha \cdot P(s)}$, e o verificador confirma se $b = a^\alpha$.

Caso as equações $g^{P(s)} = g^{H(s) \cdot T(s)}$ e $b = a^\alpha$ sejam verdadeiras, o verificador será convencido. Este método de verificação é conhecido como **conhecimento por expoente**.

Para garantir a propriedade de **conhecimento zero**, o provador aplica uma aleatorização, elevando $g^{P(x)}, g^{\alpha \cdot P(x)}, g^{H(x)}$ a uma potência aleatória δ que não é revelada ao verificador em momento algum. Mesmo após mascarar os valores, ambas as equações se mantêm:

$$(g^{P(s)})^\delta = ((g^{H(s)})^\delta)^{T(x)}, \quad b^\delta = a^{\alpha \cdot \delta}$$

2.3 PRIMITIVAS DE INTELIGÊNCIA ARTIFICIAL

A Inteligência Artificial (IA), e mais especificamente o subcampo de aprendizagem profunda (*Deep Learning*, em inglês), transformou radicalmente a área de visão computacional, capacitando máquinas a interpretar e compreender o mundo visual com uma precisão sem precedentes (WANG; DENG, 2021). Uma das aplicações mais impactantes e difundidas dessa tecnologia é o reconhecimento facial, um método de identificação biométrica que utiliza as características únicas do rosto de um indivíduo para verificar sua identidade. Esta seção detalhará as primitivas de IA que compõem um sistema moderno de reconhecimento facial, desde o processo geral de detecção até a transformação de uma face em uma representação vetorial e a subsequente comparação matemática para identificação.

2.3.1 O Processo de Reconhecimento Facial

Segundo (YI et al., 2014) e (ZHANG et al., 2016), o reconhecimento facial consiste em um fluxo de trabalho composto por várias etapas sequenciais que transformam uma imagem bruta em uma decisão de identidade. O primeiro passo consiste em localizar a presença de uma ou mais faces em uma imagem ou quadro de vídeo. Algoritmos de detecção de objetos são empregados para identificar regiões da imagem que contêm padrões característicos de um rosto humano. Uma vez detectada, a face é normalizada através de um processo de alinhamento, que rotaciona e dimensiona a imagem para que características faciais chave, como os olhos e a boca, estejam em posições padronizadas. Este pré-processamento é necessário para melhorar a precisão das etapas subsequentes.

No modelo de (SCHROFF; KALENICHENKO; PHILBIN, 2015), após o isolamento e alinhamento da face, o sistema analisa sua geometria para extrair um conjunto de características distintivas e mensuráveis. Essas características são então convertidas em uma representação vetorial compacta e única, frequentemente chamada de *embedding*. Esta representação é tipicamente um vetor de números de ponto flutuante, que serve como uma assinatura matemática da face. O estudo de (WANG; DENG, 2021) detalha que no processo de correspondência facial, a *embedding* recém-gerada é então comparada com outras *embeddings* pré-cadastradas em um banco de dados. Com isso, é calculada uma pontuação de similaridade entre o vetor da imagem de entrada e os vetores registrados no banco de dados. Com base nessa pontuação, o sistema pode realizar uma **verificação**, comparando a *embedding* de entrada com uma única *embedding* no banco de dados para confirmar se a pessoa é quem ela afirma ser, ou uma **identificação**, comparando a *embedding* de entrada com todas as *embeddings* no banco de dados para encontrar a correspondência mais próxima e determinar a identidade da pessoa.

2.3.2 Extração de *Embeddings* Faciais

O conceito central por trás dos sistemas de reconhecimento facial é a transformação de uma imagem facial em um *embedding*. De acordo com (SCHROFF; KALENICHENKO; PHILBIN, 2015), um *embedding* é uma representação da face em um espaço vetorial que captura as características semânticas essenciais do dado original. No contexto facial, a *embedding* é uma assinatura numérica que encapsula as características únicas da identidade de uma pessoa, de forma que seja invariante a fatores irrelevantes como iluminação, ângulo da câmera ou expressão facial. Em vez de comparar imagens *pixel a pixel*, o sistema compara esses vetores compactos e ricos em informação.

A genialidade da abordagem de (SCHROFF; KALENICHENKO; PHILBIN, 2015) reside na conversão de um problema de percepção visual (reconhecer um rosto) em um problema de geometria espacial (medir a distância entre pontos em um espaço vetorial). A rede neural profunda, através do treinamento, aprende a modelar um espaço vetorial multidimensional, tipicamente com 128 ou 512 dimensões, onde a proximidade geométrica entre os pontos

Equação 1 – Cálculo da Similaridade por Cosseno

$$\text{similaridade}(\vec{A}, \vec{B}) = \cos(\theta) = \frac{\vec{A} \cdot \vec{B}}{|\vec{A}| |\vec{B}|} = \frac{\sum_{i=1}^n \vec{A}_i \vec{B}_i}{\sqrt{\sum_{i=1}^n \vec{A}_i^2} \sqrt{\sum_{i=1}^n \vec{B}_i^2}}$$

Fonte: (NGUYEN; BAI, 2010)

corresponde diretamente à similaridade de identidade facial.

2.3.3 Identificação Facial com Similaridade por Cosseno

Uma vez que as faces foram convertidas em representações vetoriais, o passo final é compará-las para determinar a similaridade. Embora a **distância euclidiana** seja bastante popular, uma métrica muito comum e eficaz para a comparação final é a **similaridade por cosseno**.

Como demonstra o método proposto por (NGUYEN; BAI, 2010), a similaridade por cosseno é uma medida que calcula o cosseno do ângulo entre dois vetores em um espaço multidimensional. Sua principal vantagem é que ela mede a orientação dos vetores, ignorando suas magnitudes. Isso é particularmente útil para comparar *embeddings* faciais, pois a direção do vetor no espaço de características é o que codifica a identidade, enquanto a magnitude pode variar devido a fatores como a qualidade da imagem.

A fórmula matemática para a similaridade por cosseno entre dois vetores, $\vec{A}$ e $\vec{B}$, é dada pelo produto escalar dos vetores dividido pelo produto de suas normas, como demonstra a Equação 1.

O resultado da Equação 1 varia no intervalo de -1 a 1. Quando o valor é igual a 1, isso indica que os vetores possuem a mesma orientação, formando um ângulo de 0 graus e representando a máxima similaridade. No caso do reconhecimento facial, isso corresponderia a uma identidade perfeitamente coincidente. Já um valor igual a 0 significa que os vetores são ortogonais, evidenciando a ausência de similaridade entre eles. Por fim, um valor de -1 significa que os vetores apontam em direções opostas, formando um ângulo de 180 graus e representando, assim, a máxima dissimilaridade.

Na prática, as *embeddings* são normalizadas para terem comprimento unitário, simplificando o cálculo da similaridade por cosseno para apenas o produto escalar dos vetores. O resultado é uma **pontuação de similaridade** que quantifica a probabilidade das duas faces pertencerem à mesma pessoa.

2.3.3.1 Limiar de Similaridade

De acordo com (NGUYEN; BAI, 2010), o cálculo da similaridade por cosseno resulta em um valor que, por si só, não é suficiente para que o sistema tome uma decisão binária de

aceitação ou rejeição de identidade. Para viabilizar essa decisão, é necessário definir um **limiar de similaridade** (*threshold*, em inglês). Assim, quando a pontuação de similaridade entre dois *embeddings* excede esse limiar, é possível interpretar a comparação como uma correspondência positiva. Caso contrário, considera-se que não há correspondência.

O estudo de (WANG; DENG, 2021) informa que a escolha de um bom limiar é uma etapa crítica, pois esse parâmetro influencia diretamente as taxas de erro do sistema. Quando o limiar é estabelecido como um valor muito baixo, isso torna o sistema mais permissivo e, consequentemente, vulnerável a identificar incorretamente duas faces de pessoas distintas como pertencentes a uma mesma identidade. Por outro lado, um limiar excessivamente elevado aumenta a probabilidade do sistema não reconhecer duas faces da mesma pessoa como correspondentes.

A definição do limiar ideal, portanto, não é uma decisão puramente técnica, mas uma decisão que depende do contexto da aplicação, entrando no domínio da análise de risco e da definição de políticas de segurança específicas para cada caso de uso.

3 PROTOCOLOS ZK-SNARKS

Dando continuidade à exploração das PCZs, este capítulo se dedica a um tipo específico de protocolo de prova não interativo: os zk-SNARKs. Sigla para *Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge* (Argumentos de Conhecimento Sucintos e Não-Interativos de Conhecimento Zero, em português), os zk-SNARKs representam uma classe de sistemas de prova criptográfica proposta inicialmente por (BITANSKY et al., 2012). Esses sistemas derivam de conceitos clássicos de PCZs e têm como diferencial três propriedades-chave: sucintez, não-interatividade e a natureza de argumento (em oposição a prova absoluta), além, é claro, da própria propriedade de conhecimento zero.

As próximas seções apresentam um estudo dos protocolos zk-SNARKs, iniciando pela introdução das propriedades fundamentais que sustentam seu funcionamento. Na sequência, são examinados os principais protocolos que marcaram a evolução dessa tecnologia: **Pinocchio** e **Groth16**.

3.1 INTRODUÇÃO AOS SNARKS

As fundações teóricas para as PCZs foram estabelecidas no prestigiado trabalho de (GOLDWASSER; MICALI; RACKOFF, 1985). No entanto, foram necessárias décadas de avanços criptográficos para que esses conceitos se tornassem práticos. O termo “zk-SNARK” foi formalmente introduzido por (BITANSKY et al., 2012), apresentando uma nova classe de protocolos de prova conhecimento zero.

O grande passo inicial em direção às implementações práticas foi o protocolo **Pinocchio**, de (PARNO et al., 2013), que foi um dos primeiros a tornar os zk-SNARKs aplicáveis à computação geral, embora ainda fosse considerado lento para muitas aplicações do mundo real.

O ponto de virada ocorreu em 2016 com a introdução do **Groth16** por (GROTH, 2016). Este protocolo representou um avanço notável em questão de eficiência, reduzindo o tamanho da prova para apenas três elementos de grupo de curva elíptica e otimizando significativamente o tempo de verificação em relação ao **Pinocchio**. O **Groth16** tornou-se rapidamente o padrão da indústria em termos de desempenho e foi a tecnologia que permitiu a primeira aplicação em larga escala de um sistema zk-SNARK (LAVIN et al., 2024).

3.1.1 Propriedades Fundamentais dos zk-SNARKs

Os zk-SNARKs são classificados como “argumentos de conhecimento” e não como “provas de conhecimento” devido à natureza de sua propriedade de **solidez**. Segundo (NITU-LESCU, 2021) e (CHEN et al., 2023), a distinção entre provas e argumentos reside na forma como essa solidez é garantida. Nos sistemas de prova, exige-se solidez estatística, o que significa que mesmo um provador com poder computacional ilimitado não consegue convencer o verificador de uma declaração falsa, salvo com probabilidade insignificante. Essa garantia

é extremamente forte, porém difícil de alcançar em problemas complexos. Já nos sistemas de argumento, como é o caso dos zk-SNARKs, exige-se apenas solidez computacional, ou seja, a segurança é garantida contra provadores com poder computacional limitado. Em teoria, um provador com capacidade ilimitada poderia forjar provas falsas.

Os zk-SNARKs baseiam sua segurança em suposições criptográficas sobre a dificuldade computacional de certos problemas, e por isso são consideradas sistemas de argumento. Essa classificação se baseia em um fator de eficiência, pois alcançar provas concisas e não interativas com solidez perfeita é geralmente impraticável em sistemas computacionais modernos, enquanto a solidez computacional permite construções mais eficientes em termos de tamanho da prova e tempo de verificação.

Outra propriedade essencial dos zk-SNARKs é a **sucintez**. Segundo (CHEN et al., 2023), a sucintez refere-se ao fato de que a prova gerada é muito menor do que o circuito, tanto em tamanho quanto em tempo de verificação, sendo possível verificá-la em tempo logarítmico em relação à complexidade do circuito original. Essa característica é fundamental para viabilizar o uso prático das provas em sistemas computacionais de alto desempenho e em aplicações que exigem respostas rápidas, como autenticação em tempo real.

Além disso, os zk-SNARKs apresentam a propriedade de **não-interatividade**. Diferentemente das provas interativas, uma única mensagem do provador ao verificador é suficiente, caracterizando o que (BLUM et al., 1991) descreve como “interações unidirecionais”. O estudo de (LIANG et al., 2025) afirma que, para alcançar a não-interatividade, muitos protocolos zk-SNARK utilizam uma Fita de Referência Comum (FRC). A FRC é uma série de elementos criptográficos gerados em uma fase de cerimônia prévia, que é publicamente disponível para ambos, provador e verificador. Essa cerimônia é frequentemente realizada por uma terceira parte confiável ou como uma cerimônia de Computação Multipartidária (CMP).

Por fim, a propriedade de **conhecimento zero** garante que o verificador aprende somente a veracidade da afirmação, e nada sobre o dado subjacente (a testemunha) que justifica essa veracidade. De acordo com (NITULESCU, 2021), o conhecimento zero é formalmente definido em termos de um simulador. Um simulador é uma máquina que pode gerar uma prova “falsa” sem ter acesso à testemunha real do provador. Se a prova gerada pelo provador real for computacionalmente indistinguível da prova gerada pelo simulador, então o verificador não aprendeu nada além da verdade da afirmação.

3.1.2 Etapas de Funcionamento de um zk-SNARK

O funcionamento de um zk-SNARK envolve uma sequência estruturada de etapas, que se inicia com a **aritmética** da computação original. Os estudos de (NITULESCU, 2021) e (LAVIN et al., 2024) apresentam a aritmética como um processo que consiste em converter a computação em uma representação matemática verificável. Uma computação arbitrária pode ser representada como um circuito aritmético sobre um corpo finito modelado por um grafo acíclico direcionado. Nesse grafo, os nós correspondem a portas de adição e multiplicação, e

as arestas conectam saídas e entradas dessas portas. As entradas do circuito incluem valores conhecidos e testemunhas, e a única saída final representa o resultado da computação.

Com a computação convertida em um formato aritmético, o próximo passo é a realização da **cerimônia de confiança**, essencial para tornar o sistema não interativo. De acordo com (CHEN et al., 2023), a maioria dos protocolos zk-SNARKs modernos exige uma cerimônia de confiança inicial, que gera uma FRC contendo parâmetros criptográficos públicos que o provador e o verificador utilizam para gerar e verificar provas, respectivamente. A geração da FRC permite que a prova não seja interativa, pois o provador já tem os valores necessários sem precisar de mais comunicação com o verificador.

Durante a cerimônia, valores aleatórios são gerados. Esses valores são chamados de “lixo tóxico” porque devem ser descartados com segurança após a geração da FRC. Se um atacante obtiver esses valores, ele poderá forjar provas, por isso a geração da FRC é frequentemente realizada por uma parte confiável ou através de um protocolo de CMP para garantir que o “lixo tóxico” seja descartado. Na CMP, diferentes entidades geram a FRC cooperativamente, e basta que uma delas descarte sua parte do segredo para que a segurança seja mantida. Ao fim de toda cerimônia de confiança, duas chaves públicas são geradas: a chave de prova, que o provador utiliza para construir a prova, e uma chave de verificação, utilizada pelo verificador.

A etapa seguinte é a **geração da prova**, em que o provador utiliza os dados da computação aritmetizada e os parâmetros gerados na FRC. Conforme (NITULESCU, 2021) e (LIANG et al., 2025), o processo começa com a solução da representação matemática da computação, na qual o provador utiliza a testemunha para satisfazer um sistema de equações polinomiais que descrevem a relação entre entrada, saída e estados intermediários da execução. Em seguida, os valores solucionados são criptograficamente codificados por meio de estruturas como emparelhamentos bilineares e curvas elípticas, permitindo operações homomórficas sobre dados ocultos com base nos parâmetros públicos da FRC.

Para assegurar a propriedade de conhecimento zero, o provador aplica aleatoriedade, tornando os componentes da prova indistinguíveis de valores aleatórios e ocultando completamente a testemunha. Por fim, a prova é construída com poucos elementos criptográficos, cujo tamanho independe da complexidade da computação original. Essa prova sucinta e não interativa é enviada ao verificador em uma única mensagem.

Por fim, ocorre a **verificação da prova**. Segundo (CHEN et al., 2023), o verificador checa se os valores codificados na prova foram gerados de forma consistente a partir dos parâmetros da FRC e da estrutura matemática definida pela aritmetização. Isso garante que o provador não está forjando arbitrariamente os valores da prova, mas realmente os derivou de forma válida. Em seguida, ocorre a verificação da condição matemática principal, na qual se testa se a relação polinomial que representa a computação original é satisfeita, garantindo, com alta probabilidade, que a prova não foi forjada.

A solidez probabilística é garantida pelo Lema 2.1.1, que afirma que dois polinômios diferentes se cruzam em um número limitado de pontos, tornando altamente improvável que um provador desonesto consiga forjar uma prova que passe por esta checagem.

3.1.3 Diagrama de Funcionamento de um zk-SNARK

A Figura 7 apresenta um diagrama que exemplifica o processamento de um esquema zk-SNARK de acordo com (BONEH, 2022). Nesse exemplo, tem-se o circuito aritmético público: $C(x, w) \rightarrow \mathbb{F}$, onde x representa os parâmetros públicos (entradas conhecidas tanto pelo provador quanto pelo verificador), w é a testemunha (informação secreta conhecida apenas pelo provador) e $\mathbb{F}$ é um corpo finito sobre o qual o circuito é definido.

Durante o pré-processamento, um algoritmo S , que realiza a cerimônia de confiança, é executado sobre o circuito C para gerar um conjunto de parâmetros: $S(C) \rightarrow (S_p, S_v)$. Aqui, S_p é a chave de prova, utilizada pelo provador para gerar a prova criptográfica, e S_v é a chave de verificação, que permite ao verificador validar a prova recebida. Esses parâmetros encapsulam os detalhes do circuito e garantem que o protocolo possa ser executado de forma não interativa e com conhecimento zero.

Com os parâmetros (S_p, S_v) disponíveis, o provador pode usar a chave de prova S_p , juntamente com os valores x e w , para gerar uma prova π que atesta a validade da computação: $\pi \leftarrow \text{Prove}(S_p, x, w)$. Esta prova é então enviada ao verificador, que, utilizando a chave de verificação S_v , executa a função $\text{Verify}(S_v, x, \pi)$ para confirmar se π é válida em relação à computação definida por C . A verificação retorna um valor booleano, aceitando a prova se e somente se o circuito for satisfazível para os dados fornecidos.

Esse processo assegura que o verificador pode confiar na veracidade da computação sem ter acesso à testemunha w .

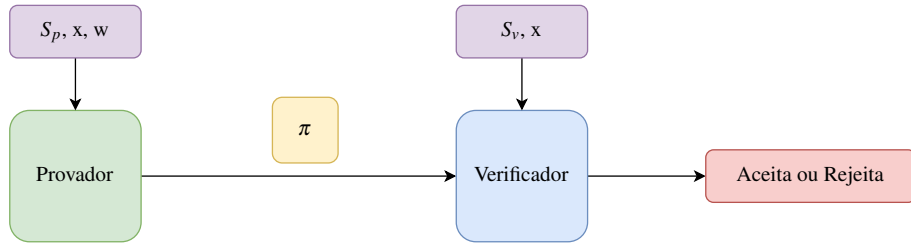


Figura 7 – Diagrama de processamento de um zk-SNARK segundo (BONEH, 2022)

3.2 TIPOS DE PROTOCOLOS ZK-SNARKS

A fase da cerimônia de confiança é um dos aspectos mais críticos e debatidos no *design* de sistemas zk-SNARK. É o processo inicial que gera os parâmetros criptográficos públicos, conhecidos como FRC, que são subsequentemente utilizados pelo provador para gerar provas e pelo verificador para validá-las.

O cerne da questão reside na geração desses parâmetros. Todos os esquemas zk-SNARK requerem a geração de aleatoriedade secreta durante a cerimônia de confiança. Essa aleatoriedade, se conhecida por qualquer parte após o término da operação, constitui uma vulnerabilidade que compromete a **solidez** de todo o sistema.

Os zk-SNARKs podem ser classificados através de como cada esquema aborda o desa-

fio da cerimônia de confiança, revelando um trilema fundamental entre eficiência, flexibilidade e segurança (NITULESCU, 2021). É extremamente difícil otimizar simultaneamente para o desempenho máximo (provas pequenas e verificação rápida), a máxima flexibilidade (uma cerimônia para todos os programas) e as mínimas suposições de segurança criptográfica.

A abordagem estudada nesta seção é a **cerimônia de confiança específica do circuito** (*Circuit-Specific Trusted Setup*), utilizada pelos protocolos **Pinocchio** e **Groth16**. Nesse modelo, cada circuito aritmético distinto exige a execução de uma nova cerimônia de confiança, o que garante a geração das provas mais eficientes conhecidas até hoje: extremamente pequenas (apenas três elementos de grupo no caso do **Groth16**) e com verificação muito rápida, baseada em poucas operações de emparelhamento.

3.2.1 Pinocchio

Desenvolvido por (PARNO et al., 2013), o protocolo **Pinocchio** introduziu um modelo eficiente e generalizável de provas sucintas e não interativas, capaz de transformar programas de computador em representações matemáticas verificáveis. Isso tornou o protocolo um marco conceitual, servindo de base para sistemas modernos de privacidade e verificação, como o **Groth16**.

Esta seção apresenta uma análise do esquema **Pinocchio** inspirada nos estudos de (PARNO et al., 2013) e na abordagem de (LAMBDA, 2024), com foco nas quatro fases do protocolo: aritmetização, cerimônia de confiança, geração da prova e verificação da prova.

3.2.1.1 Aritmetização em Pinocchio

A aritmetização é a primeira das quatro fases do protocolo. Seu objetivo é traduzir uma computação em uma representação algébrica formal verificável. O primeiro passo nesse processo de tradução é decompor a computação em suas operações básicas de adição e multiplicação, formando um circuito aritmético. Para ilustrar este processo de forma didática, será utilizado o exemplo 3.2.1, que é baseado em (KALMYKOV, 2023).

Exemplo 3.2.1. O provador quer convencer o verificador de que a seguinte afirmação é verdadeira “Conheço x , tal que a fórmula cúbica $x^3 + 2x + 7$ resulta em 19”. Nesse exemplo, a testemunha do provador é o valor $x = 2$, que é um elemento do corpo finito $\mathbb{F}_p$. O verificador conhece a equação e o resultado, mas não o valor de x .

Com base no exemplo acima, a Figura 8 apresenta o circuito aritmético da fórmula $x^3 + 2x + 7$ com $x = 2$.

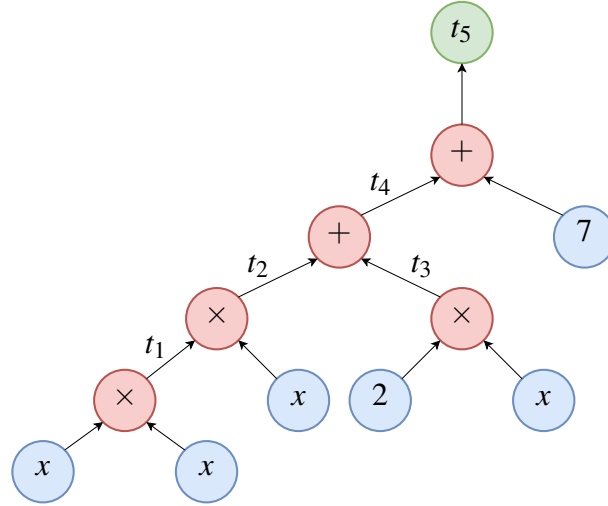


Figura 8 – Circuito aritmético da fórmula $x^3 + 2x + 7$ com $x = 2$

Com base no circuito aritmético de exemplo, são definidas as seguintes restrições:

$$\begin{array}{lll}
 t_1 = x \times x & \Rightarrow & 4 = 2 \times 2 \\
 t_2 = t_1 \times x & \Rightarrow & 8 = 4 \times 2 \\
 t_3 = 2 \times x & \Rightarrow & 4 = 2 \times 2 \\
 t_4 = t_2 + t_3 & \Rightarrow & 12 = 8 + 4 \\
 t_5 = t_4 + 7 & \Rightarrow & 19 = 12 + 7
 \end{array}$$

Essa lista de restrições descreve a computação porta por porta. No entanto, para fins criptográficos, é ineficiente lidar com essa representação diretamente. O próximo passo é converter esse conjunto de restrições em um formato equivalente, mais estruturado e algébrico, chamado *Rank-1 Constraint System* (R1CS), ou “Sistema de Restrições de Ranque 1”, em português. O R1CS é uma forma de representar a computação como um conjunto de equações lineares que podem ser verificadas eficientemente. O conceito por trás deste formato é criar restrições para o circuito, ou seja, definir condições e regras que uma computação deve satisfazer. Cada restrição em um R1CS segue a forma:

$$A \cdot w \times B \cdot w = C \cdot w$$

Onde $A, B, C \in \mathbb{F}_p^{n \times m}$ são matrizes de coeficientes que definem as n restrições do circuito. O vetor $w \in \mathbb{F}_p^m$ é o vetor testemunha, que contém todas as m variáveis do circuito, como entradas, saídas e valores intermediários. A operação $\times$ é definida como o produto de elemento a elemento. Por convenção, o primeiro elemento do vetor w é sempre a constante 1.

Para que esta equação vetorial possa representar o circuito, é preciso primeiro organizar todas as variáveis da computação (a constante 1, entradas públicas, saídas e valores intermediários) em um único vetor ordenado. Com isso, essas variáveis são agrupadas no vetor testemunha $w = [1, w_1, \dots, w_m]$. Em seguida, a matriz $A \in \mathbb{F}_p^{n \times m}$ é construída. A regra de construção estabelece que cada elemento A_{ij} da matriz é o coeficiente da variável w_j no termo à esquerda da multiplicação na restrição de índice i .

De forma análoga, constrói-se a matriz B , cujas linhas são formadas pelos coeficientes do termo à direita da multiplicação de cada restrição. Por fim, a matriz C é elaborada, onde cada linha representa os coeficientes do valor resultante de cada restrição.

Para o exemplo apresentado, o vetor testemunha w é definido como:

$$w = [1, x, t_1, t_2, t_3, t_4, t_5]$$

Com a solução $x = 2$, o vetor testemunha preenchido é:

$$w = [1, 2, 4, 8, 4, 12, 19]$$

Para cada uma das n portas do circuito, é construída uma linha das matrizes A , B e C da seguinte forma:

$$t_1 = x \times x \Rightarrow (1 \cdot t_1) = (1 \cdot x) \times (1 \cdot x)$$

$$A_1 = [0, 1, 0, 0, 0, 0, 0] \text{ (Segunda posição corresponde ao valor de } x \text{ no vetor } w)$$

$$B_1 = [0, 1, 0, 0, 0, 0, 0]$$

$$C_1 = [0, 0, 1, 0, 0, 0, 0] \text{ (Terceira posição corresponde ao valor de } t_1 \text{ no vetor } w)$$

$$t_2 = t_1 \times x \Rightarrow (1 \cdot t_2) = (1 \cdot t_1) \times (1 \cdot x)$$

$$A_2 = [0, 0, 1, 0, 0, 0, 0]$$

$$B_2 = [0, 1, 0, 0, 0, 0, 0]$$

$$C_2 = [0, 0, 0, 1, 0, 0, 0] \text{ (Quarta posição corresponde ao valor de } t_2 \text{ no vetor } w)$$

$$t_3 = 2 \times x$$

$$A_3 = [2, 0, 0, 0, 0, 0, 0] \text{ (Primeira posição corresponde às constantes do sistema)}$$

$$B_3 = [0, 1, 0, 0, 0, 0, 0]$$

$$C_3 = [0, 0, 0, 0, 1, 0, 0] \text{ (Quinta posição corresponde ao valor de } t_3 \text{ no vetor } w)$$

$$t_4 = t_2 + t_3 \Rightarrow (1 \cdot t_4) = 1 \times (1 \cdot t_2 + 1 \cdot t_3)$$

$$A_4 = [0, 0, 0, 1, 1, 0, 0] \text{ (Em uma soma, os valores da mesma linha são selecionados)}$$

$$B_4 = [1, 0, 0, 0, 0, 0, 0]$$

$$C_4 = [0, 0, 0, 0, 0, 1, 0] \text{ (Sexta posição corresponde ao valor de } t_4 \text{ no vetor } w)$$

$$t_5 = t_4 + 7$$

$$A_5 = [7, 0, 0, 0, 0, 1, 0]$$

$$B_5 = [1, 0, 0, 0, 0, 0, 0]$$

$$C_5 = [0, 0, 0, 0, 0, 0, 1] \text{ (Sétima posição corresponde ao valor de } t_5 \text{ no vetor } w)$$

O resultado do processo R1CS é um conjunto de três matrizes, A , B e C , onde as linhas de cada matriz são os vetores de cada restrição, respectivamente:

$$A = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 2 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 1 & 0 & 0 \\ 7 & 0 & 0 & 0 & 0 & 1 & 0 \end{bmatrix},$$

$$B = \begin{bmatrix} 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix},$$

$$C = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix}.$$

Perceba que para cada linha i das matrizes, a equação $A_i \cdot w \times B_i \cdot w = C_i \cdot w$ é satisfeita. Por exemplo, quando $i = 4$, temos:

$$A_4 \cdot w = [0, 0, 0, 1, 1, 0, 0] \cdot [1, 2, 4, 8, 4, 12, 19] = 8 + 4 = 12$$

$$B_4 \cdot w = [1, 0, 0, 0, 0, 0, 0] \cdot [1, 2, 4, 8, 4, 12, 19] = 1$$

$$C_4 \cdot w = [0, 0, 0, 0, 0, 1, 0] \cdot [1, 2, 4, 8, 4, 12, 19] = A_4 \cdot w \times B_4 \cdot w = 12$$

Este exemplo demonstra como a i -ésima linha de cada matriz A , B e C isola e impõe a i -ésima restrição do circuito sobre o vetor testemunha w . Se w for uma testemunha válida, todas as n equações serão verdadeiras.

Perceba também que o problema de provar a execução do circuito foi transformado em provar que se conhece um vetor w que satisfaz o sistema matricial $A \cdot w \times B \cdot w = C \cdot w$.

Embora o R1CS seja uma excelente representação algébrica, ele ainda consiste em n restrições separadas. Para um circuito grande, verificar cada uma das n linhas seria computacionalmente caro. O objetivo do zk-SNARK é a sucintez, ou seja, realizar uma única verificação que comprove a validade de todas as restrições de uma só vez.

A etapa final da aritmetização alcança exatamente isso. Ela converte o conjunto de n restrições lineares em uma única equivalência que permitirá verificar a satisfatibilidade de todas as restrições ao mesmo tempo. Essa transição é realizada através de uma técnica matemática chamada *Quadratic Arithmetic Program* (QAP), ou “Programa de Aritmética Quadrática”, em português. A ideia é utilizar as colunas das matrizes A , B e C para construir um conjunto

de pontos e, com isso, encontrar um polinômio de grau mínimo que passe por esses pontos utilizando a *Interpolação de Lagrange*.

Segundo o QAP, é possível interpretar cada coluna de uma matriz como avaliações de um polinômio sobre algum domínio adequado. Dada uma matriz $M \in \mathbb{F}_p^{n \times m}$, amostra-se o domínio $D = \{x_0, x_1, \dots, x_n\}$ sobre $\mathbb{F}_p$ e define-se o polinômio $P(x)$ como o polinômio de grau máximo n tal que $P(x_i) = M_{ij}$, sendo $i \in \{1, \dots, n\}$, $j \in \{1, \dots, m\}$ e $i, j \in \mathbb{F}_p$. Por convenção, seleciona-se como domínio de interpolação D as n -ésimas raízes unitárias. Assim, para cada coluna j da matriz M , constrói-se um conjunto de pontos K da forma:

$$K_j = \{(i, M_{ij})\}, \text{ para todo } 1 < i \leq n \text{ e } 1 < j \leq m$$

Voltando ao exemplo, para a matriz A , os pontos $(1, a_{1j}), (2, a_{2j}), \dots, (n, a_{nj})$ são usados para construir um polinômio único de grau no máximo n , o qual chamamos de $L_j(x)$. De forma análoga, os polinômios $R_j(x)$ e $O_j(x)$ são gerados a partir da interpolação das colunas correspondentes das matrizes B e C , respectivamente. Como demonstração, considere a primeira coluna da matriz A :

$$A = \begin{bmatrix} 0 \\ 0 \\ 2 \\ 0 \\ 7 \end{bmatrix}$$

Nesse caso, o conjunto K_1 de A é:

$$K_1 = \{(1, 0), (2, 0), (3, 2), (4, 0), (5, 7)\}$$

O próximo passo é aplicar a *interpolação de Lagrange* nesses pontos. O polinômio resultante da interpolação dos pontos do exemplo anterior é:

$$27 - 53.5833 \cdot x + 34.7083 \cdot x^2 - 8.91667 \cdot x^3 + 0.791667 \cdot x^4$$

Repetindo esse processo para todas as colunas da matriz A , B e C e colocando todos os coeficientes em linhas de uma matriz, temos:

$$L(x) = \begin{bmatrix} 27 & -53.5833 & 34.7083 & -8.91667 & 0.791667 \\ 5 & -6.41667 & 2.95833 & -0.583333 & 0.0416667 \\ -10 & 17.8333 & -9.83333 & 2.16667 & -0.166667 \\ -5 & 10.1667 & -6.83333 & 1.83333 & -0.166667 \\ -5 & 10.1667 & -6.83333 & 1.83333 & -0.166667 \\ 1 & -2.08333 & 1.45833 & -0.416667 & 0.0416667 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$R(x) = \begin{bmatrix} -4 & 8.08333 & -5.375 & 1.41667 & -0.125 \\ 5 & -8.08333 & 5.375 & -1.41667 & 0.125 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

$$O(x) = \begin{bmatrix} 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 \\ 5 & -6.41667 & 2.95833 & -0.583333 & 0.0416667 \\ -10 & 17.8333 & -9.83333 & 2.16667 & -0.166667 \\ 10 & -19.5 & 12.25 & -3 & 0.25 \\ -5 & 10.1667 & -6.83333 & 1.83333 & -0.166667 \\ 1 & -2.08333 & 1.45833 & -0.416667 & 0.0416667 \end{bmatrix}$$

Ao substituir a variável x nas linhas da matriz A por qualquer elemento de D , o resultado será uma das restrições. Exemplo quando $x = 1$:

$$L(1) = \begin{bmatrix} 27 & - & 53.5833 & + & 34.7083 & - & 8.91667 & + & 0.791667 \\ 5 & - & 6.41667 & + & 2.95833 & - & 0.583333 & + & 0.0416667 \\ -10 & + & 17.8333 & - & 9.83333 & + & 2.16667 & - & 0.166667 \\ -5 & + & 10.1667 & - & 6.83333 & + & 1.83333 & - & 0.166667 \\ -5 & + & 10.1667 & - & 6.83333 & + & 1.83333 & - & 0.166667 \\ 1 & - & 2.08333 & + & 1.45833 & - & 0.416667 & + & 0.0416667 \\ 0 & + & 0 & + & 0 & + & 0 & + & 0 \end{bmatrix} = \begin{bmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$

Este exemplo demonstra que a avaliação dos polinômios no ponto $x = 1$ recupera a primeira linha (restrição) das matrizes R1CS originais. De forma geral, avaliar os polinômios em $x = i$ recupera a i -ésima restrição.

O próximo passo é combinar todos os polinômios de coluna (representados pelas linhas nas matrizes $L(x), R(x), O(x)$) com a testemunha w do provador. Isso cria três polinômios agregados, que representam toda a computação ponderada pela testemunha. Eles são denotados por $L(x) \cdot w$, $R(x) \cdot w$, e $O(x) \cdot w$:

$$L(x) \cdot w = \begin{bmatrix} -51 & 21.917 & -63.2084 & 15.5833 & 1.29167 \end{bmatrix}$$

$$R(x) \cdot w = \begin{bmatrix} 6 & -8.08333 & 5.375 & -1.41667 & 0.375 \end{bmatrix}$$

$$O(x) \cdot w = \begin{bmatrix} -61 & 121.417 & -72.125 & 17.0833 & -1.37501 \end{bmatrix}$$

Com base nesses polinômios, define-se um novo polinômio $P(x)$ que agrega as verificações de todas as restrições em uma única equação polinomial:

$$P(x) = L(x) \cdot w \times R(x) \cdot w - O(x) \cdot w$$

Para finalizar o exemplo, calcula-se:

$$\begin{aligned} P(x) = & -245 + 422.335x - 758.413x^2 + 777.405x^3 - 522.259x^4 \\ & + 191.966x^5 - 52.7223x^6 + 7.67361x^7 - 0.484376x^8 \end{aligned}$$

Perceba que, se existir uma solução válida w para o R1CS, o polinômio $P(x)$, que é construído usando w , será avaliado como 0 em todo o domínio $D = \{1, 2, \dots, 5\}$. Isso ocorre porque, para qualquer i em D , a avaliação $P(i)$ é exatamente $A_i \cdot w \times B_i \cdot w - C_i \cdot w$, que sabemos ser 0 se w for válido.

Portanto, um w válido implica que $P(x)$ tem raízes em todos os pontos $\{1, \dots, n\}$. Isso nos leva diretamente à definição do polinômio nulo $T(x)$. Pelo Teorema Fundamental da Álgebra, se $P(x)$ é zero em todos os pontos $\{1, \dots, n\}$, ele deve ser divisível por cada fator $(x-1), (x-2), \dots, (x-n)$. Consequentemente, $P(x)$ deve ser divisível pelo produto de todos esses fatores, que é exatamente o polinômio $T(x)$.

O polinômio nulo, $T(x)$, é definido como:

$$T(x) = (x-1)(x-2) \dots (x-n)$$

Finalmente, a condição de satisfação do R1CS é traduzida para o domínio dos polinômios. O vetor w é uma solução válida para o R1CS se, e somente se, o polinômio $P(x)$ for divisível por $T(x)$. Isso significa que deve existir um polinômio $H(x)$ tal que $P(x) = H(x) \cdot T(x)$. Na prática, a verificação é feita de forma probabilística, escolhendo um ponto aleatório $s \in \mathbb{F}_p$ e verificando se a igualdade $P(s) = H(s) \cdot T(s)$ se mantém. A tarefa do provador é convencer o verificador de que ele conhece um $H(x)$ válido sem revelar a parte privada da testemunha.

3.2.1.2 Cerimônia de Confiança em Pinocchio

Para que o protocolo **Pinocchio** seja não-interativo, é necessária uma fase de configuração inicial, conhecida como **cerimônia de confiança**. Este processo, que é executado apenas uma vez por circuito, gera a chave de prova e a chave de verificação, denominadas como parâmetros públicos, que são utilizados para gerar a prova e realizar a sua verificação, respectivamente.

Previamente à construção da FRC, o provador e o verificador devem concordar em uma curva elíptica amigável a emparelhamento e geradores dos grupos $\mathbb{G}_1$ e $\mathbb{G}_2$ denotados por g_1 e g_2 , respectivamente. Em implementações formais, costuma-se utilizar curvas como **BN254** ou **BLS12-381**.

A seguir “[” e “]” são usados para denotar vetores e M_k é usado para denotar a k -ésima linha da matriz M . Os l valores de w são parâmetros públicos.

A parte principal da cerimônia reside na geração de um conjunto de valores aleatórios secretos. O responsável pela cerimônia de confiança inicia o processo ao selecionar oito elementos aleatórios privados $s, \alpha_L, \alpha_R, \alpha_O, \beta, r_L, r_R, \gamma$ de $\mathbb{F}_p$, definindo $r_O = r_L \cdot r_R$. Este conjunto de elementos é chamado de *lixo tóxico* e deve ser descartado e totalmente esquecido assim que as chaves forem geradas.

A **chave de prova** é composta por 7 vetores de pontos em $\mathbb{G}_1$ e 2 vetores de pontos em $\mathbb{G}_2$, de modo que

$$\begin{aligned} & [r_L \cdot L_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \\ & [r_R \cdot R_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \\ & [r_O \cdot O_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \end{aligned}$$

são vetores de pontos das $m - 1$ linhas das matrizes L, R e O (parâmetros privados), e juntamente com os vetores

$$\begin{aligned} & [r_L \cdot \alpha_L \cdot L_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \\ & [r_R \cdot \alpha_R \cdot R_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \\ & [r_O \cdot \alpha_O \cdot O_k(s) \cdot g_1]_{k \in \{1, \dots, m-1\}} \end{aligned}$$

permitem calcular s^α e s para a checagem $b = a^\alpha$ (conhecimento por expoente). Além disso, temos

$$\begin{aligned} & [(r_L \cdot \beta \cdot L_k(s) + r_R \cdot \beta \cdot R_k(s) + r_O \cdot \beta \cdot O_k(s)) \cdot g_1]_{k \in \{1, \dots, m-1\}} \\ & [r_R \cdot R_k(s) \cdot g_2]_{k \in \{1, \dots, m-1\}} \end{aligned}$$

que são necessários para decidir se a solução é válida na etapa final do protocolo. Por fim, para que o provador consiga calcular $P(x)$, é preciso do vetor

$$[s^d \cdot g_2]_{d \in \{0, \dots, n-1\}}$$

que contém as n potências de s .

A **chave de verificação** é composta por 6 pontos em $\mathbb{G}_2$, 1 ponto em $\mathbb{G}_1$, dois vetores de pontos em $\mathbb{G}_1$ e 1 vetor de pontos em $\mathbb{G}_2$, sendo que

$$g_2, \quad \gamma \cdot g_2, \quad \beta \cdot \gamma \cdot g_2$$

são valores usados para garantir que os elementos da prova foram construídos corretamente, enquanto

$$\alpha_L \cdot g_2, \quad \alpha_R \cdot g_2, \quad \alpha_O \cdot g_2$$

são necessários para a etapa de conhecimento por expoente. Já o ponto

$$r_O \cdot T(s) \cdot g_1$$

corresponde ao resultado do polinômio nulo $T(x)$ avaliado em s . Por fim, temos

$$\begin{aligned} [r_L \cdot L_k(s) \cdot g_1]_{k \in \{0, \dots, l-1\}} \\ [r_R \cdot R_k(s) \cdot g_2]_{k \in \{0, \dots, l-1\}} \\ [r_O \cdot O_k(s) \cdot g_1]_{k \in \{0, \dots, l-1\}} \end{aligned}$$

que são vetores de pontos das l linhas das matrizes L , R e O (parâmetros públicos).

Com a cerimônia de confiança concluída e o lixo tóxico destruído, o sistema está pronto. A chave de prova é distribuída aos provadores e a chave de verificação, aos verificadores. A seção a seguir descreve como um provador utiliza sua chave para construir uma evidência criptográfica (a prova π) para uma testemunha w específica que ele alega conhecer.

3.2.1.3 Geração de Prova em Pinocchio

Em posse da chave de prova e de uma testemunha válida $w = (w_0, \dots, w_{m-1})$, o provador pode gerar a prova $\pi = (L, R_1, O, R_2, L', R', O', T, H)$. O primeiro passo é definir o polinômio alvo $T(x)$, que deve ser zero em todos os pontos de avaliação do QAP:

$$T(x) = (x - 1) \cdot \dots \cdot (x - n)$$

Para o próximo passo, o objetivo do provador é construir os polinômios $P(x)$ e $H(x)$ que representam sua testemunha. No entanto, para garantir a propriedade de **conhecimento zero** e evitar que a prova revele informações sobre w , o provador deve mascarar esses polinômios. Isso é feito amostrando os valores aleatórios δ_L , δ_R e δ_O e adicionando múltiplos aleatórios de $T(x)$ aos valores de $L(x)$, $R(x)$ e $O(x)$. Esta alteração não afeta a validade da equação do QAP nos pontos de interesse, ou seja, onde $T(x) = 0$. Com isso:

$$\begin{aligned} L(x) &:= L(x) + \delta_L \cdot T(x) \\ R(x) &:= R(x) + \delta_R \cdot T(x) \\ O(x) &:= O(x) + \delta_O \cdot T(x) \end{aligned}$$

Continuando o processo de geração da prova π , o provador calcula:

$$\begin{aligned} P(x) &= L(x) \cdot w \times R(x) \cdot w - O(x) \cdot w \\ H(x) &= \frac{P(x)}{T(x)} \end{aligned}$$

Se $P(x)$ é de fato divisível por $T(x)$, então $H(x)$ será um polinômio válido, e o provador confirma que sua testemunha w satisfaz o QAP. O desafio agora é provar isso ao verificador sem revelar w .

Para isso, são utilizados os parâmetros da chave de prova, que são as avaliações dos polinômios $L(s)$, $R(s)$, $O(s)$ já criptografados no grupo. Cada elemento da prova π corresponde à avaliação dos polinômios L, R, O ou H no ponto secreto s , combinados com a testemunha w e

mascarados pelos fatores aleatórios da cerimônia. Perceba que as operações a seguir são feitas em vetores (usando a testemunha w), mas resultam em um único ponto em $\mathbb{G}_1$ ou $\mathbb{G}_2$.

$$L = \sum_{k=l}^{m-1} (w_k \cdot \underbrace{[r_L \cdot L_k(s) \cdot g_1]}_{\text{Da chave de prova}})$$

$$R_1 = \sum_{k=l}^{m-1} (w_k \cdot [r_R \cdot R_k(s) \cdot g_1])$$

$$O = \sum_{k=l}^{m-1} (w_k \cdot [r_O \cdot O_k(s) \cdot g_1])$$

Os termos acima correspondem ao cálculo dos valores de $L(s)$, $R(s)$ e $O(s)$ no grupo $\mathbb{G}_1$, utilizando a parte privada da testemunha w .

$$R_2 = \sum_{k=l}^{m-1} (w_k \cdot [r_R \cdot R_k(s) \cdot g_2])$$

Este é o termo $R(s)$ avaliado no grupo $\mathbb{G}_2$. É separado de R_1 pois será usado no outro lado da equação de emparelhamento.

$$L' = \sum_{k=l}^{m-1} (w_k \cdot [\alpha_L \cdot r_L \cdot L_k(s) \cdot g_1])$$

$$R' = \sum_{k=l}^{m-1} (w_k \cdot [\alpha_R \cdot r_R \cdot R_k(s) \cdot g_1])$$

$$O' = \sum_{k=l}^{m-1} (w_k \cdot [\alpha_O \cdot r_O \cdot O_k(s) \cdot g_1])$$

Estes três termos (L', R', O') são usados para a checagem do conhecimento por expoente, provando que L, R_1, O foram construídos usando os polinômios corretos.

$$T = \sum_{k=l}^{m-1} (w_k \cdot [(r_L \cdot \beta \cdot L_k(s) + r_R \cdot \beta \cdot R_k(s) + r_O \cdot \beta \cdot O_k(s)) \cdot g_1])$$

O termo T é usado para a checagem de consistência, garantindo que a mesma testemunha w foi usada para construir L, R_1 e O .

$$H = H(s) \cdot g_2$$

Finalmente, este é o polinômio quociente $H(x)$ avaliado no ponto secreto s e criptografado em $\mathbb{G}_2$, provando que $P(s)$ é divisível por $T(s)$. Após a geração de todos os seus componentes, a prova $\pi = (L, R_1, O, R_2, L', R', O', T, H)$ é enviada para o verificador.

3.2.1.4 Verificação de Prova em Pinocchio

O verificador, de posse da chave de verificação e dos parâmetros públicos do circuito (as l primeiras entradas de w), deve executar três checagens para se convencer da validade da

prova. Para simplificar, será denotado e para o emparelhamento cujo primeiro argumento é um ponto de $\mathbb{G}_1$, e o segundo de $\mathbb{G}_2$.

Primeiramente, é preciso garantir que a condição polinomial $L(s) \cdot R(s) - O(s) = H(s) \cdot T(s)$ seja satisfeita e, em segundo lugar, que os elementos da prova tenham sido gerados honestamente a partir dos polinômios do QAP e da mesma testemunha w . A primeira checagem foca em validar a condição polinomial. Para isso, o verificador primeiro calcula a contribuição dos parâmetros públicos L_{pub} , R_{pub} e O_{pub} , que são as l linhas das matrizes calculadas no ponto s :

$$\begin{aligned} L_{pub} &= \sum_{k=0}^{l-1} (w_k \cdot r_L \cdot L_k(s) \cdot g_1) \\ R_{pub} &= \sum_{k=0}^{l-1} (w_k \cdot r_R \cdot R_k(s) \cdot g_2) \\ O_{pub} &= \sum_{k=0}^{l-1} (w_k \cdot r_O \cdot O_k(s) \cdot g_1) \end{aligned}$$

Em seguida, ele combina esses valores com os elementos da prova L , R_2 , H e O na verificação principal:

$$e(L_{pub} + L, R_{pub} + R_2) = e(r_O \cdot T(s) \cdot g_1, H) \cdot e(O_{pub} + O, g_2)$$

No entanto, esta checagem por si só é insuficiente. Um provador malicioso poderia fornecer valores arbitrários para L , R_2 e O que simplesmente satisfazem a equação anterior. A segunda checagem, conhecida como conhecimento por expoente, é necessária para determinar se o provador de fato usou os polinômios corretos do QAP para construir L , R_1 e O :

$$\begin{aligned} e(L', g_2) &= e(L, \alpha_L \cdot g_2) \\ e(R', g_2) &= e(R_1, \alpha_R \cdot g_2) \\ e(O', g_2) &= e(O, \alpha_O \cdot g_2) \end{aligned}$$

Se estas igualdades forem verdadeiras, o verificador tem certeza de que, por exemplo, $L' = \alpha_L \cdot L$, o que implica que a única maneira pela qual o provador poderia ter construído L e L' é usando uma combinação linear de $L(s)$. De forma semelhante, ele é convencido sobre R_1 e O .

Mas isso ainda não é suficiente. A etapa final é verificar se o provador usou o mesmo conjunto de coeficientes (ou seja, a mesma testemunha w) em cada combinação linear (L , R_1 e O). A terceira checagem garante essa consistência:

$$e(T, \gamma \cdot g_2) = e(L + R_1 + O, \beta \cdot \gamma \cdot g_2)$$

Se todas as três equações de emparelhamento forem satisfeitas, o verificador se convence, com altíssima probabilidade, de que o provador conhece uma testemunha válida w que satisfaz o QAP, sem que o verificador tenha aprendido qualquer informação sobre ela. Com isso, a verificação do protocolo **Pinocchio** está completa.

3.2.2 Groth16

O protocolo **Groth16**, introduzido por (GROTH, 2016), emergiu como uma implementação altamente otimizada do **Pinocchio**. Ele rapidamente se tornou o padrão adotado para aplicações que exigiam máxima eficiência, oferecendo provas com apenas três elementos de curvas elípticas e apenas três emparelhamentos no processo de verificação. Essa eficiência o tornou a escolha ideal para ambientes com recursos computacionais e de armazenamento restritos, como as *blockchains* (LAVIN et al., 2024).

Esta seção fornece uma exploração do sistema de prova **Groth16** inspirada nos estudos de (GROTH, 2016) e na abordagem de (RARE, 2023). A análise se concentra em como o protocolo otimiza o **Pinocchio**, partindo da mesma base de aritmetização (QAP) para construir uma equação de verificação fundamentalmente mais eficiente, que é a principal fonte de sua performance.

3.2.2.1 Aritmetização em Groth16

A aritmetização em **Groth16** é idêntica à do **Pinocchio**. O processo de conversão da computação em um circuito aritmético, sua transformação em R1CS e a subsequente redução para um QAP, resultando nos conjuntos de polinômios $L(x)$, $R(x)$ e $O(x)$, é exatamente o mesmo.

A otimização do **Groth16** reside na estrutura da etapa de verificação e na construção da cerimônia de confiança para suportá-la. Para isso, continua-se a adotar um corpo finito $\mathbb{F}_p$, uma curva elíptica favorável ao pareamento (como a **BN254** e a **BLS12-381**), os grupos $\mathbb{G}_1$ e $\mathbb{G}_2$ e os geradores g_1 e g_2 , como descrito na Seção 3.2.1.

3.2.2.2 Diferenças em relação ao Pinocchio

A principal melhoria que o **Groth16** apresenta é o número inferior de checagens em comparação ao **Pinocchio**. Isso é devido à introdução de uma nova equação de verificação que utiliza apenas três emparelhamentos: $e(A \cdot g_1, B \cdot g_2) = e(\alpha \cdot g_1, \beta \cdot g_2) \cdot e(C \cdot g_1, g_2)$.

Para gerar essa nova equação, primeiramente é necessário alterar a fórmula QAP para incorporar os valores aleatórios α e β . Inicialmente tem-se o seguinte:

$$w \cdot L(x) \times w \cdot R(x) = w \cdot O(x) + H(x) \cdot T(x)$$

neste ponto, o protocolo introduz α e β no lado esquerdo da equação:

$$(\alpha + w \cdot L(x)) \times (\beta + w \cdot R(x)) = \alpha \cdot \beta + \alpha \cdot w \cdot R(x) + \beta \cdot w \cdot L(x) + w \cdot L(x) \times w \cdot R(x)$$

e substitui os termos mais à direita usando a definição QAP original:

$$(\alpha + w \cdot L(x)) \times (\beta + w \cdot R(x)) = \alpha \cdot \beta + \alpha \cdot w \cdot R(x) + \beta \cdot w \cdot L(x) + w \cdot O(x) + H(x) \cdot T(x)$$

Esta equação é algebricamente equivalente à do QAP original, mas sua estrutura é diferente. A verificação do **Groth16** nada mais é do que a verificação desta nova identidade em um ponto secreto τ (o análogo ao s do Pinocchio), criptografada nos grupos da curva elíptica.

Se definirmos os componentes da prova A, B, C como as avaliações desses polinômios em τ , obtemos a equação de verificação:

$$\begin{aligned} e(\underbrace{(\alpha + w \cdot L(\tau)) \cdot g_1}_A, \underbrace{(\beta + w \cdot R(\tau)) \cdot g_2}_B) = \\ e(\underbrace{\alpha \cdot g_1, \beta \cdot g_2}_D) \cdot e(\underbrace{(\alpha \cdot w \cdot R(\tau) + \beta \cdot w \cdot L(\tau) + w \cdot O(\tau) + H(\tau) \cdot T(\tau))}_{C}, g_2) \end{aligned}$$

Perceba que a equação acima mistura os termos da testemunha pública (índices 0 a l) com os da testemunha privada (índices $l + 1$ a m). Para separar o cálculo em duas partes, a cerimônia de confiança introduz dois novos escalares secretos, γ e δ , e força uma divisão na equação de maneira que os termos públicos são associados a γ e os termos privados a δ . Isso é feito dividindo, ou seja, multiplicando pelo inverso modular, os termos que dependem da testemunha pública por γ , e os termos que dependem da testemunha privada por δ .

Como o termo $H(s) \cdot T(s)$ está embutido em C , que depende da testemunha privada, ele também é dividido por δ . Se δ e γ forem secretos, um provador malicioso não pode “mover” um termo da parte privada para a pública, ou vice-versa, sem quebrar a equação.

3.2.2.3 Cerimônia de Confiança em Groth16

Com todos os elementos do lixo tóxico definidos $(\alpha, \beta, \tau, \gamma, \delta)$, a cerimônia de confiança pode calcular os parâmetros públicos que compõem a chave de prova e a chave de verificação. O termo τ é usado para criar as potências secretas necessárias para avaliar os polinômios e os outros escalares são usados para construir os componentes de verificação da chave.

Primeiramente, são gerados os escalares de verificação. Os valores $\alpha, \beta, \gamma, \delta$ são criptografados e publicados. Eles formam a base para a equação de verificação do **Groth16**:

$$\alpha \cdot g_1, \beta \cdot g_1, \beta \cdot g_2, \gamma \cdot g_2, \delta \cdot g_1, \delta \cdot g_2$$

Em seguida, é gerada a FRC para o grupo $\mathbb{G}_1$, que consiste nas n potências do ponto secreto τ criptografadas em $\mathbb{G}_1$. Esses termos são essenciais para que o provador possa calcular a avaliação de seus polinômios $L(x)$, $R(x)$ e $O(x)$ no ponto secreto τ :

$$[\tau^{n-1} \cdot g_1, \tau^{n-2} \cdot g_1, \dots, \tau \cdot g_1, g_1]$$

De forma similar, é gerada a FRC para o grupo $\mathbb{G}_2$. Ela é necessária principalmente para que o provador construa o componente B da prova, que será usado no emparelhamento pelo verificador:

$$[\tau^{n-1} \cdot g_2, \tau^{n-2} \cdot g_2, \dots, \tau \cdot g_2, g_2]$$

Além disso, é calculada a FRC para o polinômio quociente $H(x)$. Para realizar a verificação $P(\tau) = H(\tau) \cdot T(\tau)$ e ao mesmo tempo mascarar o resultado com δ , a FRC fornece esses termos já multiplicados por $T(\tau)$ e divididos por δ , todos criptografados em $\mathbb{G}_1$:

$$\left[\frac{\tau^{n-2} \cdot T(\tau)}{\delta} \cdot g_1, \frac{\tau^{n-3} \cdot T(\tau)}{\delta} \cdot g_1, \dots, \frac{\tau \cdot T(\tau)}{\delta} \cdot g_1, \frac{T(\tau)}{\delta} \cdot g_1 \right]$$

Após as potências de τ , a FRC deve incluir os termos específicos do circuito, que representam a porção pública da testemunha. Para os l valores públicos da testemunha, a FRC fornece o polinômio correspondente avaliado em τ , combinado com α e β , e dividido pelo escalar γ :

$$\begin{aligned} \sigma_0 &= \frac{\alpha \cdot L_0(\tau) + \beta \cdot R_0(\tau) + O_0(\tau)}{\gamma} \cdot g_1 \\ \sigma_1 &= \frac{\alpha \cdot L_1(\tau) + \beta \cdot R_1(\tau) + O_1(\tau)}{\gamma} \cdot g_1 \\ &\vdots \\ \sigma_{l-1} &= \frac{\alpha \cdot L_l(\tau) + \beta \cdot R_l(\tau) + O_l(\tau)}{\gamma} \cdot g_1 \end{aligned}$$

O verificador usará esses termos, juntamente com os valores públicos de w , para construir sua parte da equação de verificação. Finalmente, a FRC inclui os termos para a porção privada da testemunha. A estrutura é idêntica à da porção pública, mas todos os termos são divididos pelo escalar δ :

$$\begin{aligned} \sigma_l &= \frac{\alpha \cdot L_{l+1}(\tau) + \beta \cdot R_{l+1}(\tau) + O_{l+1}(\tau)}{\delta} \cdot g_1 \\ \sigma_{l+1} &= \frac{\alpha \cdot L_{l+2}(\tau) + \beta \cdot R_{l+2}(\tau) + O_{l+2}(\tau)}{\delta} \cdot g_1 \\ &\vdots \\ \sigma_{m-1} &= \frac{\alpha \cdot L_m(\tau) + \beta \cdot R_m(\tau) + O_m(\tau)}{\delta} \cdot g_1 \end{aligned}$$

O provador usará esses termos e os $m - l$ valores da testemunha para construir o componente C da prova. Com isso, chega-se ao fim da cerimônia de confiança e os seguintes parâmetros são tornados públicos:

$$(\alpha \cdot g_1, \beta \cdot g_1, \beta \cdot g_2, \gamma \cdot g_2, \delta \cdot g_1, \delta \cdot g_2, \{\tau^i \cdot g_1\}_{i=0}^{m-1}, \{\tau^i \cdot g_2\}_{i=0}^{m-1}, \{\frac{\tau^i \cdot T(\tau)}{\delta} \cdot g_1\}_{i=0}^{n-2}, \{\sigma_i\}_{i=0}^{m-1})$$

3.2.2.4 Geração de Prova em Groth16

Com a FRC gerada, um provador que conhece a testemunha $w = (w_0, \dots, w_{m-1})$ pode construir a prova $\pi = (A, B, C)$.

Para garantir o **conhecimento zero**, o provador gera dois escalares aleatórios, r e s . Ele os utiliza para mascarar os componentes da prova A , B e C , adicionando múltiplos aleatórios de

δ . A prova é então calculada da seguinte forma:

$$\begin{aligned} A &= \alpha \cdot g_1 + \sum_{i=0}^{m-1} w_i \cdot L_i(\tau) \cdot g_1 + r \cdot \delta \cdot g_1 \\ B' &= \beta \cdot g_1 + \sum_{i=0}^{m-1} w_i \cdot R_i(\tau) \cdot g_1 + s \cdot \delta \cdot g_1 \\ B &= \beta \cdot g_2 + \sum_{i=0}^{m-1} w_i \cdot R_i(\tau) \cdot g_2 + s \cdot \delta \cdot g_2 \\ C &= \sum_{i=l}^{m-1} w_i \cdot \sigma_i + H(\tau) \cdot T(\tau) + s \cdot A + r \cdot B' - r \cdot s \cdot \delta \cdot g_1 \end{aligned}$$

Note como o provador usa sua testemunha para combinar linearmente os elementos pré-calculados da FRC. O componente C é o mais complexo, pois ele agrupa a parte da testemunha privada ($\sum_{i=l}^{m-1} w_i \cdot \sigma_i$), o polinômio $H(\tau)$, e os termos r e s para neutralizar o ruído introduzido em A e B , garantindo que a equação de verificação final se mantenha válida. Com isso, o provador calcula a prova, $\pi = (A, B, C)$, e os parâmetros públicos da testemunha, $\phi = [w_0, \dots, w_{l-1}]$, e publica (π, ϕ) .

3.2.2.5 Verificação de Prova em Groth16

Primeiramente, o verificador utiliza a parte pública da testemunha ϕ e $\sigma_0, \dots, \sigma_{l-1}$ (a porção pública da FRC em $\mathbb{G}_1$) para calcular W , que representa a contribuição da entrada pública para a equação de verificação:

$$W = \sum_{i=0}^{l-1} w_i \cdot \sigma_i$$

Finalmente, o verificador executa a verificação de emparelhamento única, que é a otimização central do **Groth16**. Ele insere a prova π , sua parte calculada W , e os elementos da chave de verificação na equação:

$$e(A, B) = e(\alpha \cdot g_1, \beta \cdot g_2) \cdot e(W, \gamma \cdot g_2) \cdot e(C, \delta \cdot g_2)$$

Se esta equação for válida, o verificador se convence (com altíssima probabilidade) de que o provador conhece uma testemunha privada w que satisfaz a computação original, sem aprender nada sobre a parte privada da testemunha.

4 PROTOCOLOS DE AUTENTICAÇÃO COM PROVAS DE CONHECIMENTO ZERO

Este capítulo tem como objetivo apresentar e contextualizar pesquisas e soluções já desenvolvidas que abordam o uso de PCZs em sistemas de autenticação. Esses trabalhos servem de base para a proposta desta monografia, que visa implementar um sistema de autenticação mais seguro e privado utilizando PCZs. As abordagens analisadas neste capítulo tratam de temas como a aplicação de PCZs em protocolos de verificação de identidade, autenticação sem revelação de segredos e uso de criptografia avançada, sendo diretamente relacionadas à temática da aplicação escolhida neste projeto. A análise desses estudos fornece fundamentos teóricos e práticos que orientam o desenvolvimento da solução proposta.

4.1 EUROPEAN UNION DIGITAL IDENTITY WALLET

A *European Union Digital Identity Wallet* (EUDI Wallet), ou “Carteira de Identidade Digital Europeia” em português, é uma iniciativa da União Europeia (UE) destinada a fornecer aos cidadãos, residentes e empresas um meio seguro e conveniente de identificação digital (EUDI, 2025c). Este projeto permite que os usuários armazenem e gerenciem diversos documentos digitais, como credenciais educacionais, documentos de viagem e informações bancárias, em um único aplicativo, facilitando o acesso a serviços públicos e privados em toda a UE e assegurando que os usuários mantenham o controle sobre seus dados pessoais e compartilhem apenas as informações necessárias (EUDI, 2025b).

4.1.1 Provas de Conhecimento Zero na EUDI Wallet

O Artigo de Discussão para o Grupo de Cooperação da Identidade Digital Europeia sobre o Tópico G (EUDI, 2025a) discute a relevância da integração das PCZs na Identidade Digital Europeia. Essa análise baseia-se no plano de desenvolvimento do *Architecture and Reference Framework*, que identifica as PCZs como uma tecnologia com forte potencial para o aprimoramento da privacidade, já que as PCZs alinham-se com os requisitos legais estabelecidos pelo Regulamento (UE) 2024/1183 (PARLIAMENT; UNION, 2024).

A proposta de integração destaca que as PCZs possibilitam propriedades necessárias para sistemas de identidade digital, como divulgação seletiva de atributos, ocultação do emissor e provas de intervalo, características que atendem às exigências do Regulamento (UE) 2024/1183, que proíbe qualquer forma de rastreamento ou correlação não autorizada.

O documento também examina tecnologias específicas para viabilizar a implementação das PCZs na EUDI Wallet, com foco em zk-SNARKs. Os zk-SNARKs são valorizados por sua eficiência e provas compactas, mas apresentam limitações importantes, como a necessidade de uma cerimônia de confiança e dificuldades de implementação em *hardware* seguro, o que compromete sua viabilidade prática no contexto europeu atual.

4.1.2 Exemplo de caso de uso

Exemplo 4.1.1. Um caso de uso ilustrativo da *EUDI Wallet* envolve a verificação seletiva de atributos, como a maioridade legal, sem a revelação de dados pessoais adicionais. Suponha que Alice deseje adquirir uma bebida alcoólica de Bob, comerciante autorizado a vender apenas para indivíduos com idade igual ou superior a 18 anos. Alice, preocupada com a proteção de sua privacidade, opta por não revelar informações desnecessárias, como seu nome completo ou data de nascimento.

Utilizando sua *EUDI Wallet*, Alice possui uma credencial digital emitida por uma autoridade confiável que atesta sua idade. Essa credencial incorpora estruturas criptográficas que permitem a geração de provas específicas por meio de técnicas de PCZs. Com isso, Alice consegue derivar uma prova não interativa que demonstra unicamente o predicado “ $idade \geq 18$ ”, sem divulgar sua data de nascimento ou qualquer outro identificador pessoal.

No momento da transação, Alice utiliza seu aplicativo de identidade digital para gerar e transmitir essa prova criptográfica a Bob, que, por sua vez, usa um verificador compatível com o protocolo de PCZ para validar a prova recebida. O processo é projetado de modo que a prova só possa ser construída se Alice realmente atender ao critério de idade.

Esse modelo apresentado satisfaz as três propriedades fundamentais das PCZs:

- **Completeness:** Se Alice for de fato maior de 18 anos e seguir corretamente o protocolo, o verificador aceitará a prova como válida.
- **Solidez:** Caso Alice não satisfaça o critério de idade, ela não conseguirá gerar uma prova válida, mesmo tentando manipular os dados.
- **Conhecimento zero:** Bob não aprende nada além do fato de que Alice satisfaz a condição “ $idade \geq 18$ ”, mantendo todos os outros dados pessoais ocultos.

4.2 A PRIVACY-PRESERVING BIOMETRIC AUTHENTICATION SYSTEM WITH BINARY CLASSIFICATION IN A ZERO KNOWLEDGE PROOF PROTOCOL

A autenticação biométrica tem se consolidado como uma das principais formas de verificação de identidade, principalmente pela sua conveniência e resistência a fraudes. Entretanto, a proteção dos dados biométricos continua sendo um desafio, uma vez que tais dados são permanentes e sensíveis. Em resposta a essa preocupação, (TRAN et al., 2022) propõe um sistema que combina técnicas de IA, mais especificamente classificadores binários, com protocolos criptográficos de PCZ.

No sistema proposto, são utilizados classificadores binários como *Support Vector Machines* (SVMs) e *Multi-layer Perceptrons* (MLPs). SVM é um algoritmo de aprendizado de máquina que busca separar dados em diferentes classes por meio de uma fronteira ótima. Já

o MLP é um tipo de rede neural artificial composta por múltiplas camadas que aprende padrões complexos nos dados. Esses classificadores são treinados com representações binárias dos dados biométricos, que podem ser extraídas por métodos tradicionais ou por meio da técnica *Composite Features Retrieval* (CFR), que combina características de várias amostras de um mesmo indivíduo para gerar uma representação mais precisa.

O processo de autenticação apresentado se baseia na classificação da entrada biométrica como “genuína” ou “impostora”. Para isso, o classificador é alimentado com exemplos positivos (do mesmo usuário) e negativos (de outros indivíduos). Na autenticação real, a amostra biométrica é transformada em uma sequência binária e avaliada pelo classificador, que decide se aceita ou rejeita a identidade apresentada. Esse método garante decisões objetivas e evita o compartilhamento direto de informações sensíveis.

O ponto central da proposta é a integração com o protocolo de PCZ interativa de Chaum-Pedersen (CHAUM; PEDERSEN, 1992). Mais especificamente, após passar pela autenticação do classificador, a saída binária é processada por um código de correção de erros (ECC) e convertida em um resumo criptográfico (*hash*), que não é armazenado em lugar algum e funciona como o segredo a ser provado no protocolo PCZ. O protocolo Chaum-Pedersen é então utilizado para permitir que o usuário prove a autenticidade desse *hash* sem expô-lo diretamente.

Configuração Inicial

1. É estabelecido um grupo cíclico $\mathbb{G}_p$, gerado por um gerador $g \in \mathbb{G}_p$.
2. O verificador V utiliza um espaço de desafio C , que é um subconjunto de $\mathbb{Z}_p$.
3. O provador P produz uma tripla (u, v, w) , onde $v = g^S$ e $w = u^S$. Neste contexto, S representa o *hash* biométrico secreto que o usuário possui (o compromisso do provador). Esta tripla é um parâmetro público acessível por ambos, P e V .

Processo de Prova

1. Quando P precisa provar a V que possui o *hash* biométrico privado S , ele escolhe aleatoriamente um número S_t de $\mathbb{Z}_p$.
2. P calcula $v_t \leftarrow g^{S_t}$ e $w_t \leftarrow u^{S_t}$ e os envia para o verificador V .
3. Ao receber v_t e w_t , V gera um desafio $c \in C$ e o envia para P .
4. Com o desafio c , P calcula $S_z \leftarrow S_t + S \cdot c$ e o envia de volta para V como resposta.
5. Finalmente, V verifica se as seguintes igualdades se mantêm: $g^{S_z} = v_t \cdot v^c$ e $u^{S_z} = w_t \cdot w^c$.
6. Se as igualdades forem verdadeiras, P é autenticado como o detentor do *hash* biométrico S .

Embora o artigo demonstre que a abordagem é eficaz com base em testes em bancos de dados públicos, ele não detalha a fundo como a implementação técnica do sistema é feita, nem justifica de maneira clara por que o protocolo específico de Chaum-Pedersen foi escolhido em vez de outras variantes de PCZ. Essa ausência de discussão limita a compreensão completa da arquitetura e das decisões de projeto adotadas.

Em resumo, o estudo de (TRAN et al., 2022) representa uma contribuição relevante ao integrar técnicas de aprendizado de máquina com mecanismos criptográficos para fortalecer a segurança em sistemas de autenticação biométrica. No entanto, a falta de aprofundamento na implementação e na escolha do protocolo criptográfico utilizado são pontos que poderiam ser melhor explorados em trabalhos futuros.

4.3 ANÁLISE DAS LACUNAS IDENTIFICADAS

A revisão dos trabalhos correlatos evidencia duas abordagens predominantes para a autenticação com preservação de privacidade e mostra lacunas que este trabalho busca solucionar. A proposta apresentada em (TRAN et al., 2022) fundamenta-se no protocolo de Chaum-Pedersen, de natureza *interativa*, o que exige trocas de mensagens entre as partes envolvidas, como compromisso, desafio e resposta. Essa característica aumenta a latência em arquiteturas cliente-servidor assíncronas, nas quais se busca minimizar a comunicação entre os componentes, tornando o método menos adequado nesse tipo de sistema.

Outro aspecto importante é que a abordagem de (TRAN et al., 2022) atua de forma indireta, pois o usuário realiza a classificação biométrica e, posteriormente, demonstra conhecimento sobre um *hash* (S) derivado do resultado obtido. Embora isso comprove que o usuário possui um resultado genuíno, não assegura que a computação biométrica tenha sido executada corretamente ou com os dados apropriados.

Em contraste, a EUDI *Wallet* introduz o conceito de **divulgação seletiva de atributos**, que representa uma referência conceitual relevante. Apesar disso, trata-se de um *framework* de identidade digital de escopo mais amplo, sem oferecer uma implementação voltada especificamente à autenticação biométrica não interativa.

Diante desse panorama, observa-se uma lacuna evidente na literatura, que consiste na ausência de um protocolo de autenticação biométrica capaz de ser simultaneamente **não interativo**, garantindo eficiência em redes, e de **comprovar diretamente a integridade da computação** realizada na verificação biométrica, neste caso, a similaridade por cosseno, em vez de apenas validar um resultado derivado.

5 PROTOCOLO DE AUTENTICAÇÃO BIOMÉTRICA FACIAL COM PROVAS DE CONHECIMENTO ZERO

Este é o capítulo de apresentação de uma das principais contribuições deste trabalho: o sistema de autenticação biométrica facial com Provas de Conhecimento Zero, que tem como proposta desenvolver um protocolo de autenticação que combina um modelo de IA para realizar o reconhecimento biométrico facial com um esquema PCZ. O objetivo é garantir um método de verificação de identidade que preserve a privacidade dos usuários, eliminando a necessidade de armazenar ou transmitir dados biométricos completos.

Para isso, propõe-se um sistema em que os dados biométricos dos usuários sejam armazenados de forma criptografada, minimizando os riscos de comprometimento em caso de ataques. Com o uso de PCZs, é possível validar a identidade do usuário sem a necessidade de expor os dados biométricos. Diferente de senhas, esses dados são identificadores únicos e irreversíveis, portanto sua proteção é fundamental. O método proposto proporciona uma garantia de privacidade, tornando-se mais robusto frente aos ataques no contexto de autenticação biométrica.

Este projeto foi planejado com o intuito de apresentar uma aplicação prática relevante que complemente a pesquisa teórica. Além disso, o tema permite explorar a segurança e a eficiência das PCZs em um contexto real de autenticação, o que contribui para a construção de um estudo mais aprofundado e alinhado com os desafios atuais da área. O código fonte da implementação pode ser encontrado ao acessar o *link* disponível no apêndice deste documento.

5.1 JUSTIFICATIVA E POSICIONAMENTO DA PROPOSTA

Com base nas lacunas identificadas no capítulo anterior, este trabalho propõe um protocolo de autenticação biométrica facial que sintetiza e evolui os conceitos analisados. O protocolo se posiciona na interseção da identidade digital soberana (inspirado na EUDI *Wallet*) e da autenticação biométrica com privacidade, oferecendo melhorias arquiteturais significativas.

A principal contribuição é aplicar o paradigma de **divulgação seletiva de atributos** da EUDI *Wallet* à biometria: o usuário prova um predicado computacional (a similaridade por cosseno) sem jamais revelar os dados-fonte (as *embeddings* faciais).

Para endereçar a limitação da interatividade, o sistema apresentado, ao contrário de (TRAN et al., 2022), utiliza o protocolo **Groth16**, que é *não-interativo*. Isso é fundamental para arquiteturas cliente-servidor, onde a latência e a complexidade da comunicação devem ser mínimas.

Além disso, na proposta deste trabalho, o cálculo de verificação biométrica (a função de similaridade por cosseno) é encapsulado no circuito zk-SNARK, de forma que o verificador possa conferir que a computação foi executada corretamente e que o resultado é legítimo.

5.2 PROPOSTA DO SISTEMA

No contexto do reconhecimento facial, cada rosto é representado por um vetor de características, também denominado *embedding*. Tais vetores são gerados por redes neurais previamente treinadas com o objetivo de capturar feições distintivas (SCHROFF; KALENICHENKO; PHILBIN, 2015). A comparação entre esses vetores é realizada por meio da métrica de similaridade por cosseno, que calcula o cosseno do ângulo entre dois vetores no espaço vetorial conforme a Equação 1. O valor resultante dessa métrica varia entre -1 (vetores opostos) e 1 (vetores idênticos), sendo que valores próximos de 1 indicam alta semelhança entre os vetores, o que sugere similaridade entre os rostos correspondentes (NGUYEN; BAI, 2010).

Em um sistema tradicional de autenticação baseado em similaridade por cosseno, a imagem fornecida no momento da autenticação é convertida em um vetor $\vec{A}$. Esse vetor é então comparado a um vetor $\vec{B}$, que foi previamente armazenado durante a fase de registro do usuário. A autenticação é considerada bem-sucedida se a similaridade por cosseno entre $\vec{A}$ e $\vec{B}$ exceder um limiar pré-definido τ , que é estabelecido com base nas características do modelo de IA adotado.

Entretanto, o armazenamento desprotegido de *embeddings* faciais acarreta riscos significativos à privacidade e segurança dos usuários. Como essas *embeddings* constituem representações vetoriais únicas e funcionam como identificadores biométricos, sua exposição pode permitir que agentes mal-intencionados reconstruam imagens faciais ou simulem autenticações fraudulentas. Além disso, diferentemente de senhas tradicionais que podem ser substituídas em caso de comprometimento, *embeddings* biométricas são, por natureza, permanentes e irre recuperáveis.

Com base nos estudos de (NGUYEN; BAI, 2010), (SCHROFF; KALENICHENKO; PHILBIN, 2015) e (GROTH, 2016), o presente projeto propõe uma solução para esse problema por meio da incorporação do cálculo de similaridade por cosseno em um circuito de PCZ. Nesse modelo, a verificação da condição “ $\text{similaridade_por_cosseno}(\vec{A}, \vec{B}) > \tau$ ” é encapsulada no circuito da prova. O limiar de similaridade τ é considerado um parâmetro público, enquanto os vetores $\vec{A}$ e $\vec{B}$, que representam as *embeddings* faciais, são mantidos como entradas privadas. Essa abordagem permite a realização da autenticação facial sem a necessidade de revelar qualquer informação sensível, garantindo que o verificador valide a correspondência sem jamais acessar diretamente os vetores originais.

Essa abordagem apresenta benefícios substanciais, pois assegura privacidade total, uma vez que o verificador apenas confirma a veracidade da prova sem ter acesso à identidade do usuário. Além disso, também proporciona elevada resistência a vazamentos de dados, visto que nenhuma informação biométrica sensível é armazenada em formato acessível ou passível de recuperação por terceiros.

5.3 COMPONENTES DO SISTEMA

O protocolo proposto é composto pela interação entre três entidades: o usuário U (parte que deseja provar sua identidade), o modelo de IA M , dispositivo auxiliar sem conexão a uma rede externa, e o servidor S (parte que realiza a verificação da prova de identidade).

O modelo de IA permite gerar a representação vetorial da identidade do usuário. Essa representação é chamada de *embedding*. Suas responsabilidades incluem: gerar a *embedding* a partir dos dados fornecidos pelo usuário e gerar a prova zk-SNARK que demonstre a similaridade entre duas *embeddings* sem revelar seu conteúdo.

Já o usuário é a entidade que deseja se registrar e, posteriormente, se autenticar no sistema. Sua identidade é representada de forma vetorial por meio de *embeddings* extraídas pelo modelo de IA. Suas responsabilidades incluem: solicitar o registro e a autenticação, proteger sua chave de criptografia pessoal, armazenar seu ID e lidar com a criptografia da *embedding*.

Por fim, tem-se o servidor, que atua como ponto central de armazenamento e validação. Ele não possui acesso direto às *embeddings* originais e tampouco pode descriptografá-las. Suas responsabilidades incluem: realizar a cerimônia de confiança do protocolo zk-SNARK, armazenar a *embedding* criptografada, fornecer a *embedding* criptografada e a chave de prova quando solicitadas para autenticação e verificar a prova zk-SNARK enviada pelo usuário.

Entidades do sistema

- U : Usuário do sistema de autenticação biométrica.
- S : Servidor que valida a identidade do usuário U .
- M : Dispositivo de reconhecimento facial e extração de *embeddings*.

Algoritmos

- $Emb(f) \rightarrow e$: Função de extração de *embedding*, mapeia a face f em um vetor $e \in R^d$.
- $GenKey() \rightarrow k$, $Enc_k(m) \rightarrow c$, $Dec_k(c) \rightarrow m$: Esquema de criptografia simétrica com chave k , converte o texto plano m em texto criptografado c e vice-versa.
- $Setup(C) \rightarrow (S_p, S_v)$: Procedimento de pré-processamento do circuito C para gerar chave de prova S_p e chave de verificação S_v . É a cerimônia de confiança do protocolo **Groth16**.
- $Prove(S_p, x, w) \rightarrow \pi$: Função de geração da prova π com a chave de prova S_p , entradas públicas x e testemunha privada w conforme o protocolo **Groth16**.
- $Verify(S_v, x, \pi) \rightarrow r$: Função de verificação da prova π com a chave de verificação S_v e entradas públicas x , onde $r \in \{0, 1\}$, conforme o protocolo **Groth16**.
- $cos(e_{reg}, e_{cur})$: Função de cálculo do cosseno do ângulo entre os vetores e_{reg} e e_{cur} , onde $e_{reg}, e_{cur} \in R^d$ e d corresponde às dimensões de um vetor.

- $GenId() \rightarrow id$: Função que gera um número inteiro único utilizado como identificador.

Notações

- $A \xrightarrow{m} B$: Indica que a entidade A envia a mensagem m para a entidade B .
- $A \xleftarrow{m} B$: Indica que a entidade B envia a mensagem m para a entidade A .
- $y := f(x)$: Indica a atribuição do resultado da operação $f(x)$ à y .
- $y = x$: Indica a cópia do valor de x para y .

5.4 FUNCIONAMENTO DO SISTEMA

Antes do início do protocolo, duas etapas preparatórias são essenciais para garantir a funcionalidade do sistema: o treinamento do modelo de IA para reconhecimento facial e a realização da cerimônia de confiança. O treinamento da IA é responsável por capacitar a identificação e verificação de rostos com precisão, enquanto a cerimônia de confiança estabelece os parâmetros criptográficos iniciais necessários para o funcionamento correto das PCZs, como a geração das chaves e a configuração de parâmetros públicos.

1. Primeiramente, o circuito C que verifica a condição de similaridade é definido como:

$$C(x = \tau, w = (e_{reg}, e_{cur})) : \cos(e_{cur}, e_{reg}) \geq \tau, \text{ tal que } e_{reg}, e_{cur} \in R^d \text{ e } \tau \in R$$

2. Com isso, é realizada a cerimônia de confiança sobre o circuito C :

$$(S_p, S_v) = Setup(C)$$

Com essas etapas concluídas, o sistema proposto passa a operar em duas fases principais: a fase de registro e a fase de autenticação.

5.4.1 Fase de Registro

Primeiramente, o modelo de IA extrai a *embedding* do usuário, convertendo dados biométricos em representações vetoriais. Em seguida, o usuário criptografa sua *embedding* utilizando um algoritmo de criptografia simétrica e envia essa *embedding* criptografada para o servidor. O servidor, por sua vez, armazena a *embedding* criptografada e gera um ID (inteiro qualquer) para o usuário. Por fim, o usuário guarda sua chave criptográfica e o ID. A Figura 9 apresenta o diagrama de fluxo do processo de registro discutido nesta seção.

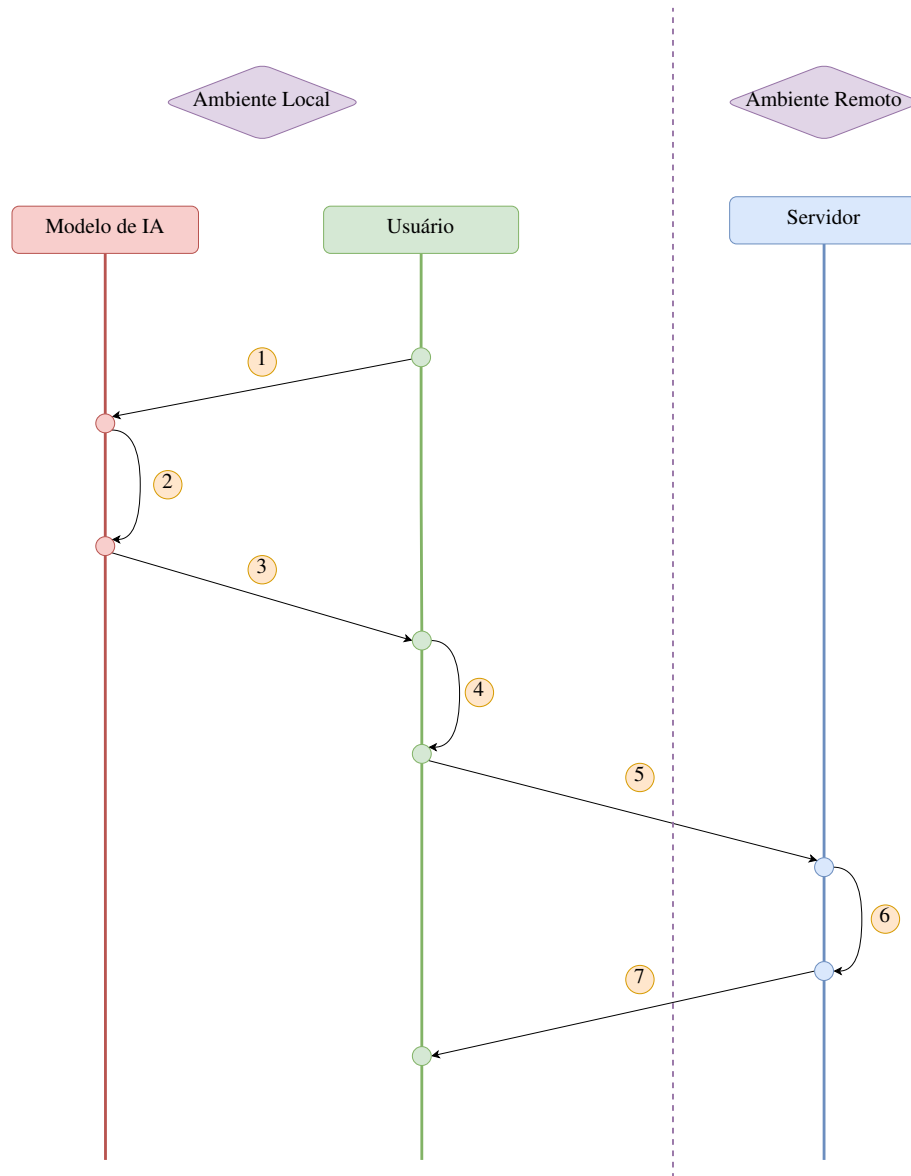


Figura 9 – Arquitetura do projeto de autenticação biométrica com Provas de Conhecimento Zero na fase de registro

Legenda de notações da fase de registro:

1. O Usuário U inicia o processo de registro enviando sua face f_{reg} para o Modelo de IA M :

$$M \xleftarrow{f_{reg}} U$$

2. M extrai a *embedding* e_{reg} de f_{reg} ao calcular:

$$e_{reg} := Emb(f_{reg})$$

3. M envia a *embedding* e_{reg} para U :

$$M \xrightarrow{e_{reg}} U$$

4. U gera uma chave simétrica k e criptografa sua *embedding* ao computar:

$$c_{reg} := Enc_k(e_{reg})$$

5. U envia a *embedding* criptografada c_{reg} para o Servidor S :

$$U \xrightarrow{c_{reg}} S$$

6. S armazena a *embedding* criptografada c_{reg} e gera um identificador único id :

$$id := GenId()$$

7. S envia o identificador id para U :

$$U \xleftarrow{id} S$$

5.4.2 Fase de Autenticação

Para iniciar a autenticação, o usuário envia seu ID ao servidor, que, por sua vez, disponibiliza uma cópia da *embedding* criptografada relativa ao ID enviado e a chave de prova. Depois disso, o usuário descriptografa a *embedding* recebida, utilizando sua chave simétrica localmente e a seguinte prova é gerada utilizando um protocolo PCZ, como zk-SNARK, por meio do modelo de IA:

$$similaridade_por_cosseno(embedding_atual, embedding_antiga) > limiar_de_similaridade$$

Por fim, a prova é enviada para o servidor, que verifica a sua validade. Se a prova for aceita, o usuário é autenticado. A Figura 10 contém o diagrama de fluxo do processo de autenticação apresentado.

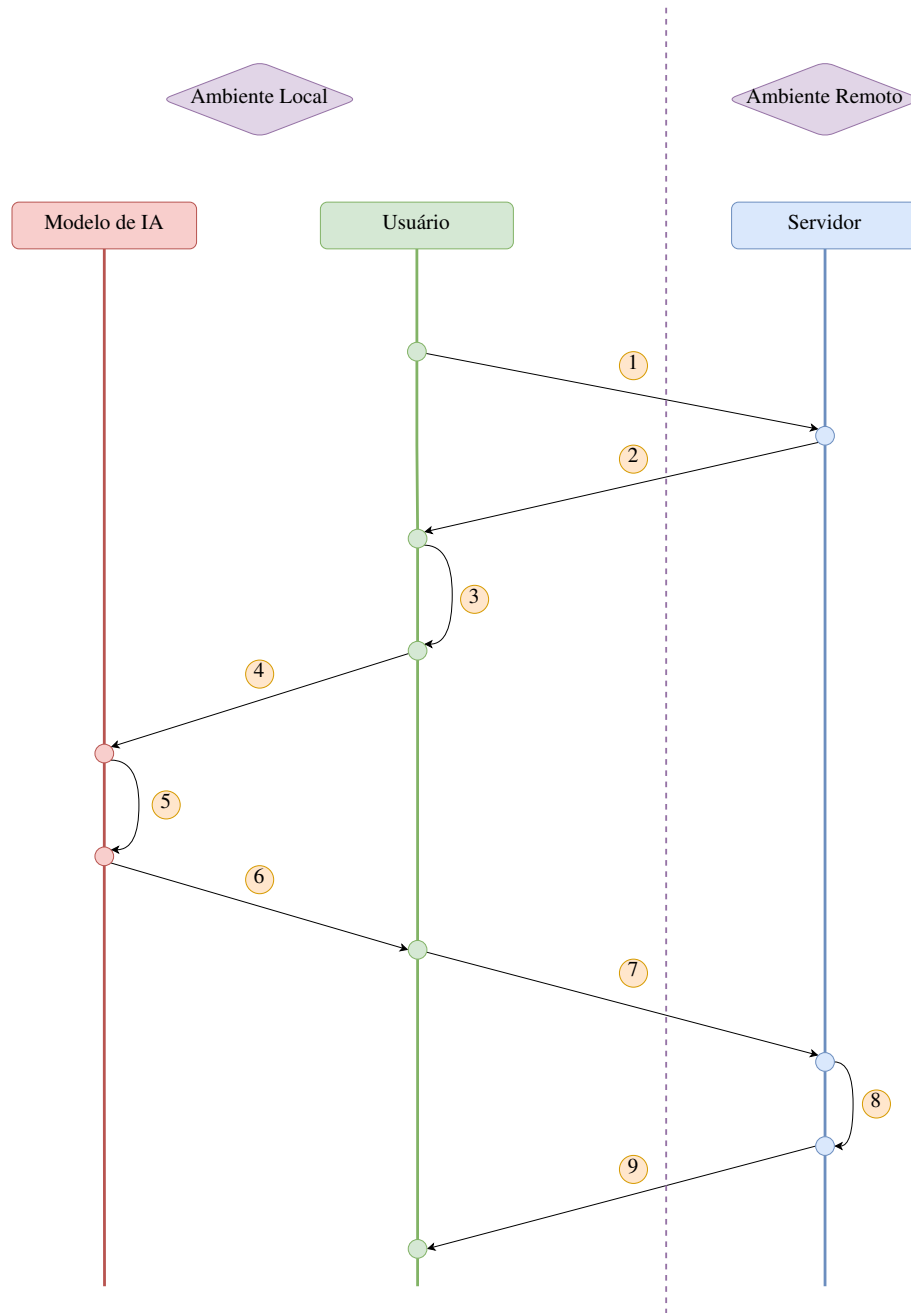


Figura 10 – Arquitetura do projeto de autenticação biométrica com Provas de Conhecimento Zero na fase de autenticação

Legenda de notações da fase de autenticação:

1. O Usuário U inicia o processo de autenticação ao enviar o identificador id ao Servidor S :

$$U \xrightarrow{id} S$$

2. S envia a chave de prova S_p e uma cópia da *embedding* criptografada c_{reg} , que é correspondente ao id recebido, a U :

$$U \xleftarrow{S_p, c_{reg}} S$$

3. U descriptografa a *embedding* e_{reg} recebida ao calcular:

$$e_{reg} := Dec_k(c_{reg})$$

4. U solicita a prova do protocolo zk-SNARK ao enviar a chave de prova S_p , a *embedding* recém descryptografada e_{reg} e uma nova captura de sua face f_{cur} para o Modelo de IA M :

$$M \xleftarrow{S_p, e_{reg}, f_{cur}} U$$

5. M extrai a *embedding* e_{cur} a partir da face f_{cur} e gera a prova zk-SNARK π ao computar:

$$e_{cur} := Emb(f_{cur})$$

$$\pi := Prove(S_p, x = \tau, w = (e_{cur}, e_{reg}))$$

6. M envia a prova zk-SNARK π a U :

$$M \xrightarrow{\pi} U$$

7. U envia a prova zk-SNARK π a S :

$$U \xrightarrow{\pi} S$$

8. S verifica a validade da prova π ao calcular:

$$r := Verify(S_v, x = \tau, \pi)$$

9. S envia o veredito da autenticação r para U :

$$U \xleftarrow{r} S$$

5.4.3 Propriedades e Suposições Explícitas

O protocolo garante propriedades de segurança que se baseiam em suposições criptográficas explícitas. A privacidade é uma característica central, assegurada pelo fato de que a entidade S não obtém conhecimento sobre os valores e_{reg} nem e_{cur} . Isto é alcançado pois S interage apenas com uma prova de conhecimento zero, que valida uma afirmação sobre estes dados sem revelá-los. De forma complementar, a confidencialidade de e_{reg} é protegida através do uso de criptografia simétrica. O texto cifrado c_{reg} não revela e_{reg} a nenhuma parte que não possua a chave k , uma vez que c_{reg} é o resultado de $Enc_k(e_{reg})$.

A integridade da prova é garantida pela função de verificação $Verify(S_v, x, \pi)$. Este algoritmo é projetado para que a prova π seja aceita somente se o provador de fato possuir uma testemunha válida que satisfaça a computação descrita pelo circuito C . Isso impede que um provador malicioso convença S de uma afirmação falsa para a qual não possui a testemunha correspondente.

Todas essas garantias dependem diretamente das suposições criptográficas adotadas. A confidencialidade de c_{reg} depende da segurança do esquema de criptografia simétrica $Enc_k(\cdot)$ e $Dec_k(\cdot)$. As garantias de integridade e privacidade dependem, por sua vez, das propriedades de solidez, completude e conhecimento zero do sistema **Groth16**.

5.5 RESULTADOS OBTIDOS

Para avaliar a viabilidade e o desempenho prático do protocolo proposto, foi calculado o tempo médio de execução da implementação em 20 iterações. As métricas foram coletadas para as principais operações do sistema, divididas por entidade. O Modelo de IA foi responsável pela geração das embeddings e da prova zk-SNARK. Ao Servidor coube a geração da FRC e o processo de verificação da prova. Por fim, o Usuário calculou o tempo de registro de um novo usuário, bem como o tempo do processo completo de autenticação. O tempo médio para a realização de cada operação pode ser encontrado na Tabela 1.

Tabela 1 – Tempo médio das operações do sistema

Métrica	Tempo Médio (s)	Responsável
Geração de <i>embeddings</i>	0.31	Modelo de IA
Geração da FRC	19.74	Servidor
Geração da prova	1.42	Modelo de IA
Verificação da prova	0.52	Servidor
Etapa de registro	0.39	Usuário
Etapa de autenticação	2.27	Usuário

Os resultados foram obtidos de um dispositivo Dell-G15, com processador 13th Gen Intel Core i5-13450HX (2.40 GHz), 8 GB de memória RAM em um sistema operacional Ubuntu 24.04 LTS e placa de vídeo NVIDIA GeForce RTX 3050 (6GB). A análise dos resultados demonstra a viabilidade do protocolo para um caso de uso prático. A geração da FRC é a etapa mais custosa, exigindo em torno de 20 segundos. Contudo, esta é uma operação de cerimônia de confiança executada uma única vez pelo Servidor para o circuito, não afetando a experiência do Usuário.

As operações voltadas ao Usuário apresentaram alta performance. O tempo da etapa de registro foi consistentemente rápido, por volta de 0.40 segundos. Esta etapa, que inclui a geração de *embeddings* faciais (tempo médio de 0.31 segundos), criptografia e armazenamento, é praticamente instantânea.

O tempo da etapa de autenticação total, métrica mais crítica para a experiência do Usuário, manteve-se baixo, com um tempo médio de 2.27 segundos. Este tempo é composto principalmente pela geração da prova **Groth16** (1.42 segundos), que é a operação criptográfica mais intensiva do lado do cliente (realizada pelo Modelo de IA). Por fim, o tempo de verificação no lado do servidor foi adequado, ficando com 0.52 segundos, o que possibilita que o servidor possa lidar com validações de forma escalável.

6 CONCLUSÃO

Este Trabalho de Conclusão de Curso teve como propósito principal estudar e demonstrar o potencial das PCZs na construção de sistemas de autenticação que conciliam segurança e privacidade. Para alcançar esse objetivo, buscou-se consolidar uma base teórica sobre os fundamentos dos protocolos zk-SNARKs e aplicar esses conceitos na implementação de um sistema de autenticação biométrica facial que preserva a confidencialidade dos dados do usuário.

A contribuição teórica deste trabalho manifesta-se na organização e integração de conceitos matemáticos e criptográficos que formam o alicerce dos sistemas modernos de PCZ. Essa base foi utilizada para descrever o funcionamento dos protocolos **Pinocchio** e **Groth16**, evidenciando suas etapas desde a geração dos parâmetros públicos e construção das provas até o processo de verificação. Além disso, o texto destacou suas propriedades fundamentais de completude, solidez e conhecimento zero. Essa sistematização contribui como um material de referência em língua portuguesa, abordando de forma acessível um tema de alta complexidade técnica.

No campo prático, a principal contribuição foi a concepção e prototipagem de um sistema de autenticação biométrica facial com preservação de privacidade, fundamentado no protocolo **Groth16**. O sistema projetado utiliza um circuito zk-SNARK que encapsula o cálculo da similaridade por cosseno entre *embeddings* faciais, permitindo que o processo de verificação ocorra sem a exposição dos dados biométricos. Essa arquitetura demonstra, de forma concreta, a aplicabilidade das PCZs na mitigação de riscos de vazamento de informações sensíveis, um problema clássico em sistemas de autenticação tradicionais, nos quais dados biométricos explícitos permanecem armazenados em servidores.

Com isso, os objetivos propostos foram integralmente atingidos. A pesquisa teórica proporcionou uma compreensão aprofundada dos fundamentos matemáticos e criptográficos dos protocolos zk-SNARKs, enquanto a aplicação prática validou a viabilidade do uso dessas técnicas em um cenário real de autenticação. Os resultados obtidos, apresentados na Seção 5.5, foram cruciais para essa validação, demonstrando que o tempo médio de autenticação para o usuário, de 2.27 segundos, é plenamente aceitável para um caso de uso prático, mesmo com 1.42 segundos para a geração da prova.

Por fim, o estudo reforça a relevância das PCZs como um componente central das infraestruturas de segurança modernas. A possibilidade de provar a veracidade de uma afirmação sem revelar informações sensíveis representa um avanço significativo para a proteção de dados pessoais e para a construção de sistemas mais confiáveis e éticos.

6.1 TRABALHOS FUTUROS

Apesar dos resultados alcançados, algumas limitações identificadas no desenvolvimento deste projeto abrem espaço para novas investigações e aprimoramentos. Uma questão relevante diz respeito à segurança da entrada de dados no modelo. A implementação atual

assume que a *embedding* fornecida ao circuito PCZ é legítima e proveniente de uma captura facial em tempo real. No entanto, foi identificada uma vulnerabilidade em que um usuário mal-intencionado poderia injetar vetores arbitrários, sejam eles não-biométricos ou pertencentes a terceiros, diretamente no fluxo de entrada do modelo de IA, contornando o sensor da câmera.

Para mitigar esse risco, trabalhos futuros podem investigar a adoção de esquemas mais robustos de garantia de procedência dos dados. Isso pode incluir a implementação de etapas rigorosas de detecção de vivacidade (*liveness detection*, em inglês) antes da geração da prova, ou a integração com ambientes de execução confiáveis e *hardware* assinado, garantindo que o vetor inserido no circuito zk-SNARK tenha sido genuinamente gerado pelo sensor físico autorizado naquele instante.

Além disso, os esquemas **Pinocchio** e **Groth16**, utilizados neste trabalho, requerem uma cerimônia de confiança específica para cada circuito, o que representa um desafio operacional e uma potencial vulnerabilidade, pois depende da honestidade dos participantes desse processo. Como direcionamento futuro, propõe-se investigar o uso de protocolos de PCZ que adotem uma cerimônia transparente (*Transparent Setup*), eliminando a necessidade dessa etapa de confiança. Nesse contexto, os *Scalable Transparent Arguments of Knowledge* (STARKs), propostos por (BEN-SASSON et al., 2018), surgem como uma alternativa promissora, oferecendo segurança baseada em funções *hash* e resistência a ataques quânticos.

Há também um crescente interesse da indústria na adoção de provas de conhecimento zero em larga escala, como demonstra o projeto **Longfellow**, atualmente em desenvolvimento pela empresa Google (FRIGO; SHELAT, 2025). Essa iniciativa visa integrar protocolos de PCZ em aplicações de autenticação e identidade digital, reforçando a tendência de adoção dessas tecnologias por grandes corporações.

Dessa forma, um trabalho futuro natural consiste em adaptar o sistema de autenticação biométrica desenvolvido neste estudo para empregar protocolos do tipo zk-STARK, permitindo uma análise comparativa em relação à eficiência, tamanho das provas e complexidade de implementação. Tal evolução poderá aprimorar a segurança e a escalabilidade da solução, além de consolidar o papel das PCZs como uma tecnologia essencial na autenticação digital e na proteção da privacidade dos usuários.

REFERÊNCIAS

- ATSERIAS, A.; TZAMERET, I. **Feasibly Constructive Proof of Schwartz-Zippel Lemma and the Complexity of Finding Hitting Sets**. 2024. Disponível em: <https://arxiv.org/abs/2411.07966>.
- BELLARE, M.; GOLDREICH, O. On defining proofs of knowledge. In: BRICKELL, E. F. (Ed.). **Advances in Cryptology — CRYPTO' 92**. Berlin, Heidelberg: Springer Berlin Heidelberg, 1993. p. 390–420. ISBN 978-3-540-48071-6.
- BEN-SASSON, E. et al. Scalable, transparent, and post-quantum secure computational integrity. **IACR Cryptol. ePrint Arch.**, v. 2018, p. 46, 2018. Disponível em: <https://api.semanticscholar.org/CorpusID:44557939>.
- BITANSKY, N. et al. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In: **Proceedings of the 3rd Innovations in Theoretical Computer Science Conference**. New York, NY, USA: Association for Computing Machinery, 2012. (ITCS '12), p. 326–349. ISBN 9781450311151. Disponível em: <https://doi.org/10.1145/2090236.2090263>.
- BLUM, M.; FELDMAN, P.; MICALI, S. Non-interactive zero-knowledge and its applications. In: **Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing**. New York, NY, USA: Association for Computing Machinery, 1988. (STOC '88), p. 103–112. ISBN 0897912640. Disponível em: <https://doi.org/10.1145/62212.62222>.
- BLUM, M. et al. Noninteractive zero-knowledge. **SIAM Journal on Computing**, v. 20, n. 6, p. 1084–1118, 1991. Disponível em: <https://doi.org/10.1137/0220068>.
- BONEH, D. **ZK Whiteboard Sessions - Module One: What is a SNARK?** 2022. Module One of the ZK Whiteboard Sessions. Acesso em: 20 de out. de 2025. Disponível em: <https://zkhack.dev/whiteboard/module-one/>.
- BOWE, S.; GABIZON, A.; GREEN, M. D. **A multi-party protocol for constructing the public parameters of the Pinocchio zk-SNARK**. 2017. Cryptology ePrint Archive, Paper 2017/602. Disponível em: <https://eprint.iacr.org/2017/602>.
- CHAUM, D.; PEDERSEN, T. P. Wallet databases with observers. In: **Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology**. Berlin, Heidelberg: Springer-Verlag, 1992. (CRYPTO '92), p. 89–105. ISBN 3540573402.
- CHEN, T. et al. **A Review of zk-SNARKs**. 2023. Disponível em: <https://arxiv.org/abs/2202.06877>.
- EUDI. **Official GitHub Organization of the European Digital Identity project**. 2025. Disponível em: <https://github.com/eu-digital-identity-wallet>.
- EUDI. **An official website of the European Union**. 2025. Disponível em: https://commission.europa.eu/strategy-and-policy/priorities-2019-2024/europe-fit-digital-age/european-digital-identity_en.
- EUDI. **A personal digital wallet for EU citizens, businesses and residents**. 2025. Disponível em: <https://ec.europa.eu/digital-building-blocks/sites/display/EUDIGITALIDENTITYWALLET/EU+Digital+Identity+Wallet+Home>.

FEIGE, U.; SHAMIR, A. Witness indistinguishable and witness hiding protocols. In: **Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing**. New York, NY, USA: Association for Computing Machinery, 1990. (STOC '90), p. 416–426. ISBN 0897913612. Disponível em: <https://doi.org/10.1145/100216.100272>.

FRIGO, M.; SHELAT abhi. **The Longfellow Zero-knowledge Scheme**. [S.l.], 2025. Work in Progress. Disponível em: <https://datatracker.ietf.org/doc/draft-google-cfrg-libzk/01/>.

GOLDREICH, O.; MICALI, S.; WIGDERSON, A. Proofs that yield nothing but their validity or all languages in np have zero-knowledge proof systems. **J. ACM**, Association for Computing Machinery, New York, NY, USA, v. 38, n. 3, p. 690–728, jul 1991. ISSN 0004-5411. Disponível em: <https://doi.org/10.1145/116825.116852>.

GOLDWASSER, S.; MICALI, S.; RACKOFF, C. The knowledge complexity of interactive proof-systems. In: **Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing**. New York, NY, USA: Association for Computing Machinery, 1985. (STOC '85), p. 291–304. ISBN 0897911512. Disponível em: <https://doi.org/10.1145/22145.22178>.

GOLDWASSER, S.; MICALI, S.; RACKOFF, C. The knowledge complexity of interactive proof systems. **SIAM Journal on Computing**, v. 18, n. 1, p. 186–208, 1989. Disponível em: <https://doi.org/10.1137/0218012>.

GROTH, J. **On the Size of Pairing-based Non-interactive Arguments**. 2016. Cryptology ePrint Archive, Paper 2016/260. Disponível em: <https://eprint.iacr.org/2016/260>.

KALMYKOV, A. **Zero-Knowledge Proof: Groth16**. 2023. Workshop presentation part of the Dora Dōjō Workshop Series. Acesso em: 20 de out. de 2025. Disponível em: <https://www.youtube.com/watch?v=VQyDSxB9Bl8>.

KATE, A.; ZAVERUCHA, G. M.; GOLDBERG, I. Constant-size commitments to polynomials and their applications. In: ABE, M. (Ed.). **Advances in Cryptology - ASIACRYPT 2010**. Berlin, Heidelberg: Springer Berlin Heidelberg, 2010. p. 177–194. ISBN 978-3-642-17373-8. Disponível em: <https://www.iacr.org/archive/asiacrypt2010/6477178/6477178.pdf>.

KOBLITZ, N. Elliptic curve cryptosystems. **Mathematics of Computation**, American Mathematical Society, v. 48, n. 177, p. 203–209, 1987. ISSN 00255718, 10886842. Disponível em: <http://www.jstor.org/stable/2007884>.

LAMBDA. **Pinocchio: verifiable computation revisited**. 2024. Pinocchio presentation page from the LambdaClass Blog. Acesso em: 20 de out. de 2025. Disponível em: <https://blog.lambdaclass.com/pinocchio-verifiable-computation-revisited/>.

LAVIN, R. et al. **A Survey on the Applications of Zero-Knowledge Proofs**. 2024. Disponível em: <https://arxiv.org/abs/2408.00243>.

LIANG, J. et al. **SoK: Understanding zk-SNARKs: The Gap Between Research and Practice**. 2025. Cryptology ePrint Archive, Paper 2025/172. Disponível em: <https://eprint.iacr.org/2025/172>.

MENEZES, A. J.; VANSTONE, S. A.; OORSCHOT, P. C. V. **Handbook of Applied Cryptography**. 1st. ed. USA: CRC Press, Inc., 1996. ISBN 0849385237.

MILLER, V. S. Use of elliptic curves in cryptography. In: **Lecture Notes in Computer Sciences; 218 on Advances in Cryptology—CRYPTO 85**. Berlin, Heidelberg: Springer-Verlag, 1986. p. 417–426. ISBN 0387164634.

NGUYEN, H. V.; BAI, L. Cosine similarity metric learning for face verification. In: **Proceedings of the 10th Asian Conference on Computer Vision - Volume Part II**. Berlin, Heidelberg: Springer-Verlag, 2010. (ACCV'10), p. 709–720. ISBN 9783642193088.

NITULESCU, A. **zk-SNARKs: A Gentle Introduction**. 2021. 50 p.

PARLIAMENT, E.; UNION the Council of the E. **Regulation (EU) 2024/1183 of the European Parliament and of the Council of 11 April 2024 amending Regulation (EU) No 910/2014 as regards establishing the European Digital Identity Framework**. Official Journal of the European Union, 2024. OJ L, 2024/1183, 30.4.2024. Disponível em: <https://eur-lex.europa.eu/eli/reg/2024/1183/oj/eng>.

PARNO, B. et al. Pinocchio: Nearly practical verifiable computation. In: **2013 IEEE Symposium on Security and Privacy**. [S.l.: s.n.], 2013. p. 238–252.

PETKUS, M. **Why and How zk-SNARK Works**. 2019. Disponível em: <https://arxiv.org/abs/1906.07221>.

RARE. **Groth16 Explained**. 2023. Groth16 presentation page from the RareSkills Blog. Acesso em: 20 de out. de 2025. Disponível em: <https://rareskills.io/post/groth16>.

SCHROFF, F.; KALENICHENKO, D.; PHILBIN, J. Facenet: A unified embedding for face recognition and clustering. In: **2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)**. IEEE, 2015. p. 815–823. Disponível em: <http://dx.doi.org/10.1109/CVPR.2015.7298682>.

SEROUSSI, G. Elliptic curve cryptography. In: **1999 Information Theory and Networking Workshop (Cat. No.99EX371)**. [S.l.: s.n.], 1999. p. 41–.

STALLINGS, W. **Cryptography and Network Security: Principles and Practice**. 8th. ed. USA: Pearson, 2022. ISBN 1292437480.

THALER, J. **Proofs, Arguments, and Zero-Knowledge**. [S.l.: s.n.], 2022. ISBN 978-1-63828-124-5.

TRAN, Q. N. et al. A privacy-preserving biometric authentication system with binary classification in a zero knowledge proof protocol. **IEEE Open Journal of the Computer Society**, v. 3, p. 1–10, 2022.

WANG, M.; DENG, W. Deep face recognition: A survey. **Neurocomputing**, Elsevier BV, v. 429, p. 215–244, mar. 2021. ISSN 0925-2312. Disponível em: <http://dx.doi.org/10.1016/j.neucom.2020.10.081>.

WU, H.; WANG, F. A survey of noninteractive zero knowledge proof system and its applications. **TheScientificWorldJournal**, v. 2014, p. 560484, 05 2014.

YI, D. et al. **Learning Face Representation from Scratch**. 2014. Disponível em: <https://arxiv.org/abs/1411.7923>.

ZHANG, K. et al. Joint face detection and alignment using multitask cascaded convolutional networks. **IEEE Signal Processing Letters**, Institute of Electrical and Electronics Engineers (IEEE), v. 23, n. 10, p. 1499–1503, out. 2016. ISSN 1558-2361. Disponível em: <http://dx.doi.org/10.1109/LSP.2016.2603342>.

APÊNDICE A - CÓDIGO FONTE DA IMPLEMENTAÇÃO

O repositório com o código-fonte do sistema de autenticação biométrica facial desenvolvido neste trabalho pode ser acessado por meio do seguinte *link* do GitHub:

<https://github.com/AlexDavisNeuwiem/INE5448>

Ou pelo repositório GitLab da UFSC:

<https://codigos.ufsc.br/alex.davis.neuwiem/ine5448>

Para a utilização correta do programa, recomenda-se seguir as instruções contidas no arquivo **README.md**. Além disso, uma demonstração da execução da aplicação pode ser encontrada no seguinte vídeo: **<https://youtu.be/NZuAtn9qfy4>**.

APÊNDICE B - ARTIGO DO TCC

Provas de Conhecimento Zero: Um Estudo sobre zk-SNARKs e sua Aplicação em Sistemas de Autenticação

Alex Davis Neuwiem da Silva¹

¹Departamento de Informática e Estatística
Universidade Federal de Santa Catarina (UFSC)
Florianópolis – SC – Brasil

alexavisneuwiem@gmail.com

Abstract. *Zero-Knowledge Proofs (ZKPs) are powerful cryptographic tools for ensuring privacy. However, the literature is often fragmented between theoretical concepts and practical implementations. This work connects these two fronts by analyzing the fundamental concepts of zk-SNARKs protocols, specifically Pinocchio and Groth16. Furthermore, it details the development of a privacy-preserving facial biometric authentication system. The proposed solution uses the Groth16 protocol to prove the similarity between facial embeddings without revealing the biometric data itself. The results demonstrate the viability of the system, with authentication times compatible with real-world applications.*

Resumo. *As Provas de Conhecimento Zero (PCZs) representam ferramentas criptográficas poderosas para garantir privacidade. No entanto, a literatura encontra-se fragmentada entre conceitos teóricos e implementações práticas. Este trabalho conecta essas duas frentes ao realizar uma análise dos conceitos fundamentais dos protocolos zk-SNARKs, especificamente Pinocchio e Groth16. Além disso, detalha o desenvolvimento de um sistema de autenticação biométrica facial que preserva a privacidade. A solução proposta utiliza o protocolo Groth16 para provar a similaridade entre embeddings faciais sem revelar os dados biométricos em si. Os resultados demonstram a viabilidade do sistema, com tempos de autenticação compatíveis com aplicações reais.*

1. Introdução

A rápida evolução das tecnologias digitais e a crescente necessidade de proteger dados sensíveis impulsionam o desenvolvimento de técnicas criptográficas cada vez mais avançadas. Nesse contexto, as Provas de Conhecimento Zero (PCZs), introduzidas formalmente por [Goldwasser et al. 1989], destacam-se como uma das inovações mais promissoras. Elas permitem que uma parte, denominada provador, demonstre para outra, o verificador, que uma afirmação é verdadeira, sem revelar nenhuma informação adicional além da própria veracidade dessa afirmação. Para que um protocolo seja considerado uma PCZ, ele deve satisfazer obrigatoriamente três propriedades fundamentais. A primeira é a completude, que assegura que se a afirmação for verdadeira e o protocolo for seguido corretamente, o verificador será convencido. A segunda é a solidez, que garante que se a afirmação for falsa, a probabilidade de um provador malicioso enganar o verificador é negligenciável. Por fim, a propriedade de conhecimento zero assegura que o verificador não aprende nada além do fato de que a afirmação é verdadeira.

Embora as provas interativas tenham sido as primeiras a surgir, elas dependem de comunicação constante, o que limita seu uso em redes distribuídas ou sistemas com alta latência. Para contornar essas limitações, foram introduzidas as provas não interativas [Blum et al. 1988], que permitem que uma única mensagem convença o verificador. Este artigo apresenta um estudo teórico sobre os protocolos zk-SNARKs (*Zero-Knowledge Succinct Non-interactive Arguments of Knowledge*) [Bitansky et al. 2012], uma classe de provas não interativas e sucintas, focando nos esquemas Pinocchio [Parno et al. 2013] e Groth16 [Groth 2016]. Além da análise teórica, o trabalho propõe e implementa um sistema de autenticação biométrica facial que utiliza o protocolo Groth16 para validar identidades sem expor dados biométricos sensíveis, preenchendo a lacuna entre a teoria matemática complexa e a aplicação prática de segurança.

2. Fundamentação Teórica

A compreensão dos zk-SNARKs exige o domínio de um conjunto de primitivas matemáticas e criptográficas. Os protocolos modernos operam sobre estruturas algébricas chamadas grupos e corpos finitos [Stallings 2022]. A segurança desses sistemas repousa na dificuldade computacional de problemas específicos, como o Problema do Logaritmo Discreto (DLP), que consiste em encontrar o expoente de um gerador que resulta em um determinado elemento do grupo [Menezes et al. 1996]. Em implementações práticas, utilizam-se Curvas Elípticas [Miller 1986], que oferecem um maior nível de segurança que os de corpos finitos tradicionais. Uma ferramenta crucial nesse contexto são os emparelhamentos bilineares (*pairings*), que permitem mapear elementos de dois grupos distintos para um terceiro grupo alvo, preservando relações lineares essenciais para a verificação de equações polinomiais de forma eficiente [Bowe et al. 2017].

Outro pilar teórico é o conceito de Circuitos Aritméticos e o Lema de Schwartz-Zippel. Qualquer computação pode ser representada como um circuito aritmético, um grafo onde os nós são operações de adição e multiplicação [Chen et al. 2023]. O desafio dos zk-SNARKs é provar que se conhece uma atribuição de valores (testemunha) que satisfaz esse circuito. Para tornar essa verificação sucinta, o sistema de restrições do circuito é convertido em polinômios. O Lema de Schwartz-Zippel fornece a base probabilística para verificar a igualdade desses polinômios [Atserias and Tzameret 2024]: ao avaliá-los em um ponto aleatório de um domínio suficientemente grande, a probabilidade de que dois polinômios diferentes produzam o mesmo resultado é extremamente baixa. Isso permite verificar a integridade de toda a computação através de uma simples avaliação pontual [Petkus 2019].

No contexto da aplicação proposta, também são fundamentais as primitivas de Inteligência Artificial. O reconhecimento facial moderno baseia-se na extração de características faciais por redes neurais profundas, resultando em vetores numéricos conhecidos como *embeddings* [Schroff et al. 2015]. A comparação entre faces não é feita pixel a pixel, mas sim calculando a distância ou similaridade entre esses vetores. A métrica utilizada neste trabalho é a similaridade por cosseno, que mede o cosseno do ângulo entre dois vetores [Nguyen and Bai 2010]. Se o valor resultante superar um limiar pré-definido, considera-se que as faces pertencem à mesma pessoa.

3. Protocolos zk-SNARKs

Os zk-SNARKs são argumentos de conhecimento caracterizados por serem sucintos e não interativos. A sucintez refere-se ao fato de que o tamanho da prova é muito pequeno e o tempo de verificação é rápido, independentemente da complexidade da computação original [Chen et al. 2023]. A não-interatividade é alcançada através de um modelo de Fita de Referência Comum (FRC), gerada em uma fase inicial conhecida como Cerimônia de Confiança (*Trusted Setup*) [Liang et al. 2025]. O funcionamento de um zk-SNARK segue um fluxo estruturado que inicia com a Aritmetização, onde a computação é convertida em um Sistema de Restrições de Ranque 1 (R1CS). Subsequentemente, utiliza-se a técnica de Programa de Aritmética Quadrática (QAP) para transformar essas restrições em polinômios, permitindo verificar todas as regras do circuito através de uma única identidade polinomial. Por fim, ocorrem as fases de geração da prova pelo provador e sua validação pelo verificador [Nitulescu 2021].

3.1. Pinocchio

O protocolo Pinocchio [Parno et al. 2013] foi um marco no desenvolvimento de provas de conhecimento zero práticas, sendo um dos primeiros a fornecer um método eficiente para verificar computações gerais. Ele introduziu o uso de QAPs para traduzir a satisfatibilidade do circuito em um problema de divisibilidade polinomial. No Pinocchio, o provador deve demonstrar que conhece um polinômio $H(x)$ tal que $P(x) = H(x) \cdot T(x)$, onde $P(x)$ representa a computação e $T(x)$ é um polinômio alvo. A verificação baseia-se em emparelhamentos bilineares para checar se essa equação se sustenta em um ponto secreto τ , definido durante a cerimônia de confiança, e se os termos da prova foram construídos consistentemente usando a mesma testemunha.

3.2. Groth16

O protocolo Groth16, introduzido por [Groth 2016], surgiu como uma otimização significativa do Pinocchio e rapidamente se estabeleceu como o padrão da indústria para aplicações que exigem alta performance [Lavin et al. 2024]. Sua principal inovação está na redução drástica do tamanho da prova, que consiste em apenas três elementos de grupo de curva elíptica, e na eficiência da verificação, que requer apenas três operações de emparelhamento, independentemente do tamanho do circuito. Essa eficiência é alcançada através de uma reestruturação da equação de verificação e da cerimônia de confiança. Assim como o Pinocchio, o Groth16 exige uma configuração confiável específica para cada circuito, o que significa que qualquer alteração na lógica da computação requer uma nova cerimônia de geração de parâmetros.

4. Sistema de Autenticação Biométrica Proposto

A aplicação desenvolvida neste trabalho busca solucionar o problema da privacidade em sistemas de autenticação biométrica. Em abordagens tradicionais, o vazamento de um banco de dados biométrico é um problema irreversível, pois as características faciais de um indivíduo são imutáveis. A proposta apresentada utiliza o protocolo Groth16 para permitir que um usuário prove sua identidade sem que o servidor de autenticação tenha acesso aos dados biométricos em momento algum, diferindo de abordagens interativas anteriores como a de [Tran et al. 2022].

4.1. Arquitetura e Fluxo de Execução

O sistema é composto por três entidades principais: o Usuário, o Servidor e o Modelo de IA, que atua localmente com o usuário. O fluxo de operação divide-se em duas fases distintas. Na fase de registro, o modelo de IA captura a face do usuário e extrai a *embedding* de referência. O usuário então gera uma chave simétrica, criptografa essa *embedding* e envia apenas o dado cifrado para o servidor. O servidor armazena o dado criptografado e gera um identificador, sem jamais ter acesso à biometria em texto claro.

A fase de autenticação inicia-se quando o usuário solicita acesso enviando seu identificador. O servidor responde enviando a *embedding* criptografada e a chave de prova pública do circuito. O usuário descriptografa a *embedding* de referência localmente e captura uma nova imagem de sua face para gerar a *embedding* atual. Ambas as *embeddings* (registrada e atual) são inseridas como entradas privadas (testemunhas) no circuito zk-SNARK. Este circuito encapsula a lógica matemática da similaridade por cosseno e verifica se o resultado da comparação excede um limiar público de aceitação. O resultado desse processo é uma prova criptográfica sucinta, que é enviada ao servidor. O servidor, por sua vez, executa a função de verificação utilizando a prova recebida e a chave de verificação. Se a prova for válida, a identidade do usuário é confirmada, garantindo que ele possui uma biometria correspondente à armazenada, sem que os dados biométricos tenham trafegado pela rede ou sido processados pelo servidor.

5. Resultados e Discussão

Para validar a proposta, o sistema foi implementado e submetido a testes de desempenho em um dispositivo Dell-G15, com processador 13th Gen Intel Core i5-13450HX (2.40 GHz), 8 GB de memória RAM em um sistema operacional Ubuntu 24.04 LTS e placa de vídeo NVIDIA GeForce RTX 3050 (6GB). Foram realizadas 20 iterações para calcular o tempo médio das operações críticas do protocolo. A geração da Fita de Referência Comum (FRC), que corresponde à cerimônia de confiança, mostrou-se a etapa mais custosa, levando aproximadamente 19,74 segundos. No entanto, é importante ressaltar que esta operação é executada uma única vez durante a configuração inicial do sistema e não impacta a experiência do usuário.

Do ponto de vista da usabilidade, as métricas obtidas foram satisfatórias. A etapa de registro do usuário é praticamente instantânea, levando cerca de 0,39 segundos. O processo completo de autenticação, que inclui a captura da face, geração da prova no cliente e verificação no servidor, levou em média 2,27 segundos. Deste total, a geração da prova pelo modelo de IA consome a maior parte do tempo (1,42 segundos), o que é esperado devido à complexidade das operações criptográficas envolvidas. A verificação no servidor foi extremamente eficiente, com um tempo médio de 0,52 segundos, demonstrando que a solução possui boa escalabilidade para lidar com múltiplas requisições. Esses resultados, apresentados na Tabela 1, confirmam que a utilização de zk-SNARKs em sistemas de autenticação biométrica é viável para cenários reais, oferecendo um ganho substancial em privacidade com um custo computacional aceitável.

Tabela 1. Tempo médio das operações do sistema

Operação	Tempo Médio (s)	Entidade Responsável
Geração de <i>embeddings</i>	0.31	Modelo de IA
Geração da FRC	19.74	Servidor
Geração da prova	1.42	Modelo de IA
Verificação da prova	0.52	Servidor
Etapa de registro	0.39	Usuário
Etapa de autenticação	2.27	Usuário

6. Conclusão

Este trabalho explorou a interseção entre criptografia avançada e inteligência artificial. A análise teórica dos protocolos Pinocchio e Groth16 forneceu a base necessária para compreender como transformar computações arbitrárias em provas verificáveis. A implementação prática do sistema de autenticação biométrica comprovou que o protocolo Groth16 é capaz de validar a similaridade facial de forma não interativa e segura.

O sistema proposto mitiga efetivamente os riscos associados ao vazamento de dados biométricos, uma vez que o servidor armazena apenas dados criptografados e interage apenas com provas de conhecimento zero. Apesar da dependência de uma cerimônia de confiança específica para o circuito, os tempos de execução obtidos indicam que a solução é prática para uso cotidiano. Como trabalhos futuros, sugere-se a investigação de protocolos zk-STARKs (*Scalable Transparent Argument of Knowledge*) [Ben-Sasson et al. 2018], que eliminam a necessidade da cerimônia de confiança e oferecem resistência a ataques de computação quântica, potencialmente aumentando a transparência e a longevidade da segurança do sistema.

Referências

- Atserias, A. and Tzameret, I. (2024). Feasibly constructive proof of schwartz-zippel lemma and the complexity of finding hitting sets.
- Ben-Sasson, E., Bentov, I., Horesh, Y., and Riabzev, M. (2018). Scalable, transparent, and post-quantum secure computational integrity. *IACR Cryptol. ePrint Arch.*, 2018:46.
- Bitansky, N., Canetti, R., Chiesa, A., and Tromer, E. (2012). From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again. In *Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS '12*, page 326–349, New York, NY, USA. Association for Computing Machinery.
- Blum, M., Feldman, P., and Micali, S. (1988). Non-interactive zero-knowledge and its applications. In *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing, STOC '88*, page 103–112, New York, NY, USA. Association for Computing Machinery.
- Bowe, S., Gabizon, A., and Green, M. D. (2017). A multi-party protocol for constructing the public parameters of the pinocchio zk-SNARK. *Cryptology ePrint Archive*, Paper 2017/602.
- Chen, T., Lu, H., Kunpittaya, T., and Luo, A. (2023). A review of zk-snarks.

- Goldwasser, S., Micali, S., and Rackoff, C. (1989). The knowledge complexity of interactive proof systems. *SIAM Journal on Computing*, 18(1):186–208.
- Groth, J. (2016). On the size of pairing-based non-interactive arguments. Cryptology ePrint Archive, Paper 2016/260.
- Lavin, R., Liu, X., Mohanty, H., Norman, L., Zaarour, G., and Krishnamachari, B. (2024). A survey on the applications of zero-knowledge proofs.
- Liang, J., Hu, D., Wu, P., Yang, Y., Shen, Q., and Wu, Z. (2025). SoK: Understanding zk-SNARKs: The gap between research and practice. Cryptology ePrint Archive, Paper 2025/172.
- Menezes, A. J., Vanstone, S. A., and Oorschot, P. C. V. (1996). *Handbook of Applied Cryptography*. CRC Press, Inc., USA, 1st edition.
- Miller, V. S. (1986). Use of elliptic curves in cryptography. In *Lecture Notes in Computer Sciences; 218 on Advances in Cryptology—CRYPTO 85*, pages 417–426, Berlin, Heidelberg. Springer-Verlag.
- Nguyen, H. V. and Bai, L. (2010). Cosine similarity metric learning for face verification. In *Proceedings of the 10th Asian Conference on Computer Vision - Volume Part II, ACCV'10*, pages 709–720, Berlin, Heidelberg. Springer-Verlag.
- Nitulescu, A. (2021). zk-SNARKs: A Gentle Introduction.
- Parno, B., Howell, J., Gentry, C., and Raykova, M. (2013). Pinocchio: Nearly practical verifiable computation. In *2013 IEEE Symposium on Security and Privacy*, pages 238–252.
- Petkus, M. (2019). Why and how zk-snark works.
- Schroff, F., Kalenichenko, D., and Philbin, J. (2015). Facenet: A unified embedding for face recognition and clustering. In *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 815–823. IEEE.
- Stallings, W. (2022). *Cryptography and Network Security: Principles and Practice*. Pearson, USA, 8th edition.
- Tran, Q. N., Turnbull, B. P., Wang, M., and Hu, J. (2022). A privacy-preserving biometric authentication system with binary classification in a zero knowledge proof protocol. *IEEE Open Journal of the Computer Society*, 3:1–10.