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

Victor do Valle Cunha

Protocolo para Negociação de Atributos e Ambiente de Execução na Identidade Fiduciária e Análise Formal de Segurança

Florianópolis
2024

Victor do Valle Cunha

Protocolo para Negociação de Atributos e Ambiente de Execução na Identidade Fiduciária e Análise Formal de Segurança

Trabalho de Conclusão de Curso submetido ao Curso de Graduação em Ciência da Computação do Centro Tecnológico da Universidade Federal de Santa Catarina como requisito para obtenção do título de Bacharel em Ciência da Computação.
Orientador: Prof. Frederico Schardong, Dr.
Coorientador: Prof. Ricardo Custódio, Dr.

Florianópolis
2024

Ficha de identificação da obra elaborada pelo autor,
através do Programa de Geração Automática da Biblioteca Universitária da UFSC.

Ravenhurst, Omar
   Template LaTeX seguindo a RN 46/2019/CPG da UFSC / Omar Ravenhurst ; orientador, Ben Trovato, coorientador, Lars Thørväld, 2019.
   666 p.

   Tese (doutorado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2019.

   Inclui referências.

   1. Ciência da Computação. 2. Documentação. 3. LaTeX. I. Trovato, Ben. II. Thørväld, Lars. III. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. IV. Título.

Victor do Valle Cunha

Protocolo para Negociação de Atributos e Ambiente de Execução na Identidade Fiduciária e Análise Formal de Segurança

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, 13 de Dezembro de 2024.

_____

Prof. Márcio Bastos Castro, Dr
Coordenador do Curso

Banca Examinadora:

_____

Prof. Frederico Schardong, Dr.
Orientador
Universidade Federal de Santa Catarina

_____

Maurício de Vasconcelos Barros
Avaliador
Universidade Federal de Santa Catarina

_____

Brendon Vicente Rocha Silva
Avaliador
Universidade Federal de Santa Catarina

À minha mãe e à minha namorada.

AGRADECIMENTOS

Concluir esta monografia não teria sido possível sem o apoio daqueles que estiveram ao meu lado ao longo de toda a jornada. Agradeço, primeiramente, aos amigos que tive a sorte de conhecer durante esse período – Cris, João, Léo, Luan, Dudu e Caz –, com quem compartilhei momentos de aprendizado, companheirismo e incentivo mútuo. Um agradecimento especial vai para Dani, que me apoiou em momentos difíceis e me ofereceu forças quando mais precisei. Aos amigos de longa data, Bia, Rondon, Barcelos e Henrique, agradeço pela confiança e pelo incentivo constante, por nunca terem desistido de mim e por me inspirarem a seguir em frente, independentemente dos desafios.

Também expresso minha profunda gratidão aos meus familiares – minha tia e meu irmão, que sempre foi um exemplo de dedicação e persistência. À minha namorada, Maria, que esteve ao meu lado de maneira incondicional, agradeço pelo amor, paciência e pelo encorajamento constantes, fundamentais para enfrentar todos os momentos difíceis. Por fim, agradeço especialmente à minha mãe, cujo esforço e comprometimento inabaláveis permitiram que seus filhos alcançassem o sonho da formação acadêmica.

"Parabéns mamãe, seu projeto de homem feliz deu certo"
Leandro Roque

RESUMO

O modelo de Identidade Fiduciária representa uma abordagem inovadora para o desenvolvimento de sistemas de identificação, com foco em solucionar dois desafios centrais dos modelos convencionais: a usabilidade e a proteção da privacidade dos dados do usuário. A inclusão de um novo ator, responsável por atender a essas demandas, permite a definição de propriedades que facilitam o alcance dessas metas. No entanto, a definição dessas propriedades, por si só, é insuficiente. Para que sejam aplicadas de maneira eficaz, é imprescindível contar com protocolos de comunicação seguros entre as três entidades envolvidas no processo de autenticação do usuário: o Emissor, o Fiduciário e o Verificador. Além disso, é essencial que esses protocolos incluam propriedades de segurança específicas, capazes de validá-los em diferentes cenários de uso. Diante dessas considerações, este trabalho de conclusão de curso adapta e expande o protocolo OpendID para Apresentações Verficáveis (OIDC4VP) para o contexto deste novo modelo de identidade, possibilitando a negociação de dados sensíveis no processo de autenticação e a manipulação desses dados em um ambiente de confiança para o usuário. Dessa forma, busca-se uma experiência mais fluida, em que o usuário não precise interagir constantemente com formulários de autorização, mas que ainda preserve as propriedades de segurança essenciais para o cenário proposto.

Palavras-chave: Identidade Eletrônica. Identidade Digital. Gestão de Identidade. Identidade Autossoberana. Identidade Fiduciária.

ABSTRACT

The Fiduciary Identity model represents an innovative approach to developing identification systems, focusing on addressing two central challenges of conventional models: usability and the protection of user data privacy. The inclusion of a new actor, responsible for meeting these demands, enables the definition of properties that facilitate the achievement of these goals. However, defining these properties alone is not sufficient. To apply them effectively, it is essential to rely on secure communication protocols among the three entities involved in the user authentication process: the Issuer, the Fiduciary, and the Verifier. Additionally, it is crucial that these protocols incorporate specific security properties capable of validating them across different use scenarios. In light of these considerations, this thesis adapts and expands the OpenID Connect for Verifiable Presentation (OIDC4VP) protocol for the context of this new identity model, enabling the negotiation of sensitive data in the authentication process and the handling of such data within a trusted environment for the user. This approach seeks to provide a smoother experience, where the user does not need to constantly interact with authorization forms, while still preserving the essential security properties for the proposed scenario.

Keywords: Electronic Identity. Digital Identity. Identity Management. Self-Sovereign Identity. Fiduciary Identity.

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CRS	Common Reference String
DA	Definição de Apresentação
DID	Identificador Descentralizado
DSL	Domain Specific Language
DV	Dolev-Yao
HTTP	HyperText Transfer Protocol
IAM	Identity and Access Management
IdP	Identity Provider
JSON	JavaScript Object Notation
JSON-LD	JSON for Linked Data
JWS	JSON Web Signature
JWT	JSON Web Token
MPC	Multiparty Computation
MRPTWG	Machine Readable Privacy Terms Working Group
OIDC	OpenID Connect
OIDC-D	OpenID Connect Discovery
OIDC-DCR	OpenID Connect Dynamic Client Registration
OIDC4CI	OpenID para Emissão de Credenciais Verificáveis
OIDC4VC	OpenID Connect com Credenciais Verificáveis
OIDC4VP	OpenID para Apresentações Verificáveis
OPAL	Open Algorithms
PGP	Pretty Good Privacy
SIOPv2	Self-Issued OpenID Provider v2
SP	Service Provider
SSI	Self-Sovereign Identity
SSO	Single Sign-On
VC	Verifiable Credentials
VP	Verifiable Presentation
WIM	Web Infrastructure Model
ZK-SNARK	Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge
ZK-STARK	Zero-Knowledge Scalable Transparent ARguments of Knowledge
ZKP	Zero-Knowledge Proofs

SUMÁRIO

1 INTRODUÇÃO

A identificação de pessoas no ambiente digital é um processo essencial para plataformas online, desempenhando um papel crucial tanto na segurança quanto na personalização das interações dos usuários. Conforme o número de serviços digitais e transações online cresce exponencialmente, garantir que os indivíduos sejam corretamente identificados e autenticados tornou-se uma prioridade para empresas e governos. Esse processo não só assegura que o acesso aos sistemas seja restrito a usuários autorizados, protegendo dados sensíveis e prevenindo fraudes, mas também possibilita a entrega de experiências personalizadas, adaptadas às necessidades e preferências de cada usuário, tornando a interação mais fluida e satisfatória.

Ao longo dos anos, diferentes modelos de identidade – representações abstratas que descrevem formas de gerenciar as identidades dos usuários em sistemas computacionais (El Jaouhari; BOUABDALLAH; BONNIN, 2017) – foram propostos e desenvolvidos para atender às crescentes demandas por segurança, privacidade, escalabilidade e usabilidade no ambiente digital. Entre esses modelos, o Modelo Terceirizado destaca-se como o mais amplamente adotado atualmente. Esse modelo é caracterizado pela intermediação de grandes empresas que funcionam como elos entre o usuário e o Provedor de Serviços, do inglês Service Provider (SP), encarregado de fornecer o serviço desejado. Empresas como Google e Facebook atuam como Provedores de Identidade, do inglês Identity Provider (IdP), simplificando o processo de autenticação e aumentando a usabilidade ao permitir que os usuários utilizem as mesmas credenciais em múltiplos serviços. Esse processo é viabilizado pelo uso de tokens, que são dados digitais gerados para representar a identidade de um usuário. Em vez de transmitir diretamente as credenciais, os IdP emitem tokens que podem ser validados pelos Provedores de Serviços, garantindo que a identidade do usuário seja confirmada de forma segura e eficiente.

No entanto, essa conveniência envolve a transferência do armazenamento e gerenciamento de identidades virtuais para corporações, o que levanta preocupações quanto à privacidade e ao controle de grandes volumes de dados sensíveis por essas entidades. Para superar essas limitações, novos paradigmas estão sendo explorados, como o da Identidade Auto-Soberana, conhecida como Self-Sovereign Identity (SSI), que busca oferecer aos próprios titulares um controle mais direto sobre seus dados (Dock.io, 2024). Em vez de confiar em uma entidade terceira para armazenar suas informações, essa abordagem permite que os dados sejam mantidos de forma segura em dispositivos pessoais, como smartphones, utilizando tecnologias de criptografia e blockchain para assegurar sua integridade e autenticidade. Assim, o SSI inevitavelmente transfere a responsabilidade de gerenciar dados pessoais do IdP para o próprio usuário, o que pode gerar frustração em usuários com pouca familiaridade tecnológica.

Em resposta às limitações do modelo Terceirizado — que apresenta desafios quanto ao controle e à privacidade das informações — e do modelo de SSI, que transfere ao usuário a responsabilidade de gerenciar seus próprios dados, surge o modelo de identidade Fiduciário. Um novo padrão de gestão de identidades que introduz uma entidade de confiança, denominada Fiduciário (SCHARDONG; CUSTODIO, 2024), que atua de forma transparente para garantir a autenticação e a autorização nas infraestruturas web desejadas pelo usuário, aliviando-o da tarefa de administrar seus dados pessoais.

Para que o modelo Fiduciário funcione de maneira eficaz e segura, torna-se essencial a criação de um protocolo de comunicação que estabeleça normas claras para a interação entre o Fiduciário, os Provedores de Serviço e os usuários. Esse protocolo precisa definir como as informações de identidade devem ser trocadas de forma segura e garantir que os dados pessoais mantidos pelo Fiduciário estejam acessíveis somente aos serviços autorizados pelo usuário. A ausência de tal protocolo poderia gerar inconsistências e vulnerabilidades, comprometendo a privacidade e a confiabilidade do modelo.

Dentro desse escopo, o objetivo desta monografia é desenvolver um esquema que descreva de forma detalhada a comunicação entre o Fiduciário e o SP, buscando alcançar um equilíbrio eficaz entre usabilidade e segurança. A pesquisa visa aprofundar o nível de detalhamento necessário para uma possível implementação prática, explorando as diferentes possibilidades de comunicação entre essas entidades sem comprometer a privacidade dos usuário.

1.1 OBJETIVOS GERAIS

Desenvolver um protocolo que permita aos representantes legítimos dos usuários, denominados Fiduciários, autenticar os usuários junto aos Provedores de Serviços e facilitar o compartilhamento seguro e controlado de informações dentro dessa nova infraestrutura de identidade eletrônica.

1.2 OBJETIVOS ESPECÍFICOS

i. Propor um mecanismo para a transferência de Credenciais Verificáveis por meio de Apresentações Verificáveis.

ii. Estabelecer mecanismos eficazes para assegurar a implementação do princípio de Minimização de Dados dentro do contexto de um Modelo Fiduciário.

iii. Realizar uma análise formal de segurança.

1.3 MÉTODO DE PESQUISA

Este estudo utilizará uma abordagem metodológica qualitativa, descritiva e explicativa para a criação de uma interface de comunicação baseada nas propriedades do modelo Fiduciário. A pesquisa será conduzida por meio de uma abordagem qualitativa, caracterizada pela coleta e análise de dados descritivos, permitindo uma compreensão aprofundada do tema estudado. Para tanto, serão revisados modelos existentes de gerenciamento de identidades no ambiente digital, bem como protocolos que possibilitem a implementação efetiva dessa gestão.

A metodologia descritiva tem como objetivo detalhar as características e processos envolvidos na proposta de um protocolo que aumente a privacidade dos usuários sem comprometer sua experiência no ambiente digital. Nesse contexto, essa metodologia será empregada para descrever minuciosamente o funcionamento do protocolo, suas principais características, as regras que o governam, os tipos de dados que ele pode transmitir, entre outros aspectos relevantes.

Por fim, o estudo irá utilizar uma abordagem explicativa por meio uma análise formal de segurança, essencial para demonstrar que o protocolo é seguro em relação à definição de segurança sob determinadas suposições e decisões de modelagem. Uma vez comprovadas como verdadeiras, essas propriedades permanecem válidas, diferentemente dos testes tradicionais, que cobrem apenas cenários específicos. Esse aprimoramento permitirá que o protocolo seja implementado em sistemas de Gerenciamento de Identidades e Acessos em larga escala, popularmente conhecidos como Identity and Access Management (IAM).

1.4 MOTIVAÇÃO E JUSTIFICATIVA

A segurança da informação tornou-se uma preocupação na era digital, à medida que indivíduos enfrentam ameaças cada vez mais sofisticadas. Tornou-se necessário a criação de mecanismos que mitiguem seus efeitos, visando fortalecer a resiliência do usuário a ameaças digitais emergentes. Escândalos de vazamentos de dados pessoais se tornaram cada vez mais comuns,o escândalo do Facebook e Cambridge Analytica (G1, 2018) envolvendo a coleta e uso inadequado de dados pessoais de milhões de usuários do Facebook sem o seu consentimento explícito, foi um grande marco nesse sentido. No Brasil a proteção da privacidade é princípio constitucional previsto pelos incisos X, XI e XII, do artigo 5º, da Constituição Federal de 1988 (1). Assim como afirma Maria Eugênia Finkelstein e Claudio Finkelstein (FINKELSTEIN; FINKELSTEIN, 2019):

> "Cada vez que o usuário trafega na Rede, para que possa usufruir de seus benefícios, deverá preencher formulários virtuais, nos quais informa seus dados pessoais, seus há-

> bitos de consumo e, às vezes, seus dados patrimoniais e preferências. Dessa forma, os sites que se dedicam ao comércio eletrônico organizam verdadeiros bancos de dados acerca de seus usuários, cuja utilização encontra-se numa zona cinzenta, uma vez que nem o usuário nem o Poder Público sabem exatamente a forma da utilização destas informações."

.

Para contribuir com o avanço da pesquisa no campo de IAM, o modelo Fiduciário — um paradigma inovador em identidade eletrônica — requer o desenvolvimento de um protocolo que, além de alcançar seus objetivos como modelo, ofereça uma base formal de segurança confiável.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 O DESAFIO DA IDENTIFICAÇÃO DIGITAL

A prática de identificar pessoas para a prestação de serviços é muito comum e se manifesta em diversos contextos, desde o check-in de embarque até a recepção em estabelecimentos de saúde, onde são solicitados a apresentar documentos que confirmem sua identidade. Essa abordagem tornou-se imperativa para assegurar que as empresas possam aperfeiçoar a qualidade dos serviços oferecidos aos clientes, proporcionando uma experiência mais segura e personalizada. De maneira análoga, a identificação na web viabilizou um ambiente digital mais interativo e adaptado às preferências individuais. A exibição de telas com sugestões de produtos em uma loja eletrônica ou a orientação sobre quais perfis seguir em uma rede social são apenas alguns exemplos dentre a infinidade de possibilidades desse processo de personalização.

No entanto, esse procedimento suscita preocupações substanciais no que diz respeito à privacidade dos indivíduos. O monitoramento constante das ações dos usuários, muitas vezes conduzido de forma despercebida, pode resultar em uma intrusão significativa em suas vidas pessoais, minando a confiança essencial na relação entre eles e as plataformas digitais. Além disso, o compartilhamento não autorizado desses dados com terceiros representa uma ameaça significativa à segurança e à integridade das informações pessoais. A falta de controle sobre para quem e como esses dados são repassados pode conduzir a uma série de consequências prejudiciais, incluindo o seu potencial uso indevido, vazamento de informações sensíveis e até mesmo a manipulação de dados para fins maliciosos.

Nesse contexto, o avanço contínuo dos modelos de identidade na área de IAM é impulsionado pela necessidade de abordar essas preocupações e encontrar soluções inovadoras. À medida que a tecnologia evolui, surgem métodos cada vez mais sofisticados e seguros para autenticação e autorização. Entre as abordagens exploradas estão as Provas de Conhecimento Zero, ou Zero-Knowledge Proofs (ZKP), e a tecnologia blockchain, que visam equilibrar a segurança com a proteção da privacidade do usuário. As Provas de Conhecimento Zero constituem uma técnica criptográfica que permite verificar a veracidade de uma informação sem necessidade de revelá-la, preservando a privacidade dos dados. Já a blockchain funciona como uma estrutura de dados distribuída que registra informações de forma transparente e imutável, garantindo a integridade e rastreabilidade dos dados sem recorrer a uma autoridade central. Dessa maneira, o desenvolvimento de modelos de identidade mais robustos busca não apenas aprimorar a segurança, mas também oferecer uma experiência digital personalizada, preservando a confidencialidade e a integridade das informações pessoais.

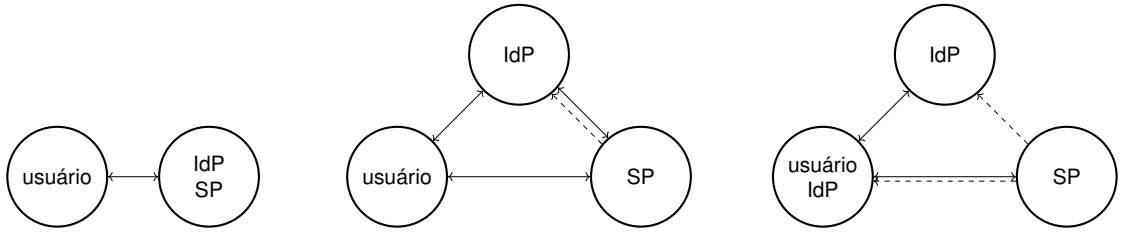
2.2 MODELOS DE IDENTIDADES

Modelo de identidade refere-se a uma representação abstrata que descreve como as identidades são gerenciadas, autenticadas e autorizadas em um sistema de computação. A identidade, predominantemente representada por meio de uma **credencial** contendo afirmações acerca do usuário, conhecidas por **claims**, e são de importância crucial para o controle de acesso e a segurança em ambientes computacionais. Essa representação possibilita a determinação de quais entidades possuem permissão para acessar recursos específicos e realizar ações designadas. Portanto, os modelos de identidade são estruturados em entidades que desempenham papéis específicos ao longo do ciclo de vida da credencial (BERTINO; TAKAHASHI, 2009). Essas entidades incluem:

- **Usuário:** Representa o indivíduo interessado em utilizar um serviço. No entanto, o acesso a esse serviço requer a sua identificação.
- **SP**: Entidade que solicita a identificação do usuário para permitir o acesso aos seus serviços.
- **IdP**: Responsável por emitir credenciais (identificação) para os usuários que desejam acessar algum serviço.

Os modelos são categorizados em três tipos distintos (SCHARDONG, 2022): centralizado, terceirizado e auto-soberano. Cada uma dessas classificações representa uma abordagem única em termos de interação e organização dos elementos mencionados anteriormente. A Figura 1 oferece uma representação visual de cada modelo respectivamente, os quais serão detalhadamente descritos nas próximas subseções.

Figura 1 – Modelos de identidade.



(a) Modelo Centralizado (b) Modelo Terceirizado (c) Modelo SSI

Fonte: Schardong (2022)

2.2.1 Centralizado

Na década de 1960, a internet ainda estava em seus primórdios, e poucos serviços web estavam disponíveis. Nessa época, era comum que cada site mantivesse seu próprio sistema interno para gerenciar as credenciais dos usuários, desempenhando simultaneamente as funções de SP e IdP. Assim, cada sistema decidia quais informações eram necessárias para identificar o usuário em seu contexto específico, enquanto os próprios usuários eram responsáveis por administrar suas credenciais, mantendo uma senha única para cada aplicativo que utilizavam (WANGHAM et al., 2010).

No final dos anos 1990 e início dos anos 2000, uma série de serviços e plataformas surgiram, tornando a internet mais atrativa. Entre eles estavam os serviços de e-mail como Yahoo Mail e Hotmail, os motores de busca como Google, o comércio eletrônico com Amazon e eBay, e as redes sociais, começando com MySpace e seguido pelo Facebook em 2004. Essa explosão de novos serviços resultou em um aumento significativo no número de credenciais que as pessoas precisavam gerenciar. Como consequência, muitos optaram por senhas mais simples e fáceis de lembrar, ou até mesmo começaram a reutilizar senhas entre diferentes serviços (HARDMAN, 2018). Essa prática comprometeu consideravelmente a segurança das informações pessoais, pois a violação de apenas uma senha poderia permitir acesso a vários aspectos da vida digital.

Além disso, esse modelo enfrentou desafios ao tentar oferecer uma melhor experiência de navegação ao usuário. Cada aplicação da época tinha seu próprio procedimento para a criação de credenciais de acesso, o que levava muitas pessoas a preencherem os formulários de maneira incompleta ou incorreta, na tentativa de obter acesso ao serviço o mais rapidamente possível. Isso prejudicava a capacidade de personalização da experiência, impactando negativamente na apresentação de conteúdos e funcionalidades adaptadas às preferências e necessidades individuais (WANGHAM et al., 2010).

Para superar as dificuldades do modelo centralizado, um novo paradigma foi proposto e foi possível melhorar a experiência do usuário, reduzir a carga administrativa de múltiplas credenciais e aumentar a segurança dos dados pessoais. Ele ficou conhecido como modelo terceirizado.

2.2.2 Terceirizado

A proliferação de provedores de serviços destacou de forma evidente os problemas previamente descritos. Gerenciar uma senha para cada plataforma acessada tornou-se um ônus significativo para os indivíduos, enquanto que, para os desenvolvedores, representava uma sobrecarga substancial. Isso se devia à necessidade de

alocar recursos consideráveis para a manutenção e segurança das credenciais em seus sistemas. A solução para esse problema foi terceirizar essa responsabilidade para uma nova entidade, o IdP, levando esse modelo a ser conhecido por modelo terceirizado (SCHARDONG, 2022). Nesse sentido, diversos protocolos foram criados para padronizar as interações entre IdP, SP e usuários finais, como os protocolos de troca de tokens, como OAuth 2.0 (HARDT, 2012) e OpenID Connect (SAKIMURA et al., 2014).

Além disso, começou a ser utilizado o conceito de autenticação única, conhecido como Single Sign-On (SSO). O SSO permite que os indivíduos façam login uma única vez para acessar múltiplos provedores de serviços, sem a necessidade de se autenticar novamente em cada um deles, até que as credenciais expirem. Esse avanço significativo simplificou consideravelmente a experiência do usuário, tornando o processo de autenticação mais eficiente e conveniente. Com o SSO, o compartilhamento de identidades entre diferentes provedores de serviços tornou-se mais fluido, aumentando a usabilidade e a satisfação do usuário.

Além de melhorar a experiência do usuário, o SSO também proporciona benefícios adicionais, como a redução da fadiga de senha, já que os indivíduos não precisam lembrar de múltiplas senhas para diferentes serviços. Isso, por sua vez, pode melhorar a segurança, pois os eles são menos propensos a utilizar senhas fracas ou repetidas em diferentes plataformas. No entanto, é crucial garantir a segurança das credenciais de SSO, pois o comprometimento de uma única conta pode potencialmente dar acesso a múltiplos serviços.

Apesar dos progressos alcançados, é importante destacar algumas desvantagens relacionadas à privacidade. O risco se concentra na possibilidade de que os dados pessoais coletados durante o processo de verificação de identidade e compartilhamento de informações sejam comprometidos ou divulgados de maneira inadequada, ou ainda, mal utilizados por uma das partes que têm acesso a eles, tipicamente IdP e SP (PANDEY; SAINI, 2012). Sobre a prerrogativa de oferecer melhores serviços, fazer melhorias e personalizar a experiência, muitos IdPs, como o Google (SIDELL, 2020), coletam dados sobre a forma como usuário usa seus dispositivos, apps e serviços, incluindo o comportamento de navegação.

2.2.3 Auto-Soberano

O modelo Auto-Soberano é um paradigma de gestão de identidades que tem se destacado nos últimos anos com o avanço das tecnologias de armazenamento de dados descentralizados, como blockchains. Nesse modelo, o controle das informações pessoais é transferido para os indivíduos, permitindo-lhes determinar com quem, quando e de que maneira compartilham seus dados (BOSCH, 2024). Em contraste com os modelos tradicionais, nos quais os dados são geridos por entidades centraliza-

das, o modelo Auto-Soberano proporciona maior privacidade e autonomia ao usuário, pois elimina intermediários e centraliza a gestão de identidade no próprio indivíduo.

No contexto da SSI, o modelo de Credenciais Verificáveis, conforme proposto pela W3C (SPORNY et al., 2024), é amplamente adotado. Esse modelo faz uso das **Verifiable Credentials (VC)**, também chamadas por Credencias Verificáveis, que são documentos digitais emitidos pelo IdP e contêm informações específicas sobre o indivíduo que podem ser verificadas criptograficamente por qualquer entidade. Além disso, o titular das credenciais tem a capacidade de gerar **Verifiable Presentation (VP)**, ou em português Apresentações Verificávies, que são estruturas para a apresentação de dados de identidade digital, codificadas de modo a garantir que a autoria das informações seja verificável criptograficamente, assegurando assim sua autenticidade e integridade sem a necessidade de revelar informações adicionais além do necessário.

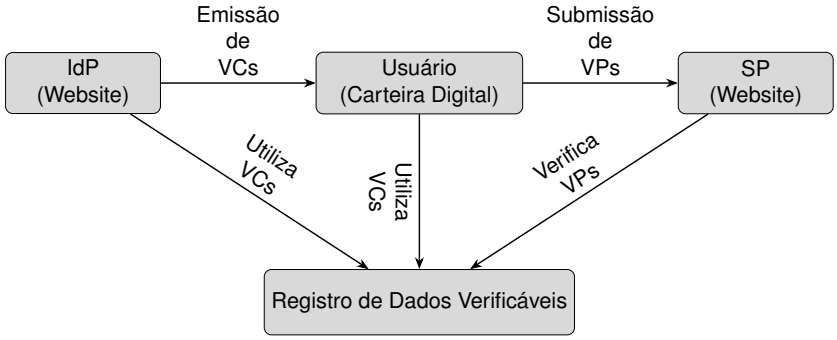
As VPs utilizam duas técnicas principais para minimizar a exposição de informações sensíveis: **Divulgação Seletiva** e **ZKP**. A Divulgação Seletiva permite ao titular compartilhar apenas partes específicas dos dados contidos nas credenciais, restringindo a divulgação ao mínimo necessário para a verificação, enquanto o restante dos dados permanece protegido. Essa abordagem proporciona um controle granular sobre quais informações são reveladas, fortalecendo a privacidade e a segurança. Por exemplo, em uma VC que contenha nome, data de nascimento e endereço, o titular pode optar por compartilhar apenas o nome e omitir os demais detalhes. A Prova de Zero-Conhecimento é outra técnica poderosa empregada nas VPs, o qual é possível provar a veracidade de uma afirmação sem revelar os dados subjacentes. Por exemplo, pode-se demonstrar que uma pessoa é maior de idade sem expor sua data de nascimento específica. Este conceito será explorado mais detalhadamente na Subsubseção 2.2.3.2.

O fluxo de VCs e VPs entre os atores IdP, Usuário e SP está representado na (Figura 2). O processo inicia-se com o IdP, que desempenha o papel de entidade responsável pela criação e emissão da VC. O IdP gera a credencial associada a um identificador. Normalmente, um **identificador** é usado para obter uma chave pública criptográfica, necessária para verificar a assinatura digital na credencial. As opção mais conhecida para identificador são os certificados X.509, mas também há outros exemplos como Identificador Descentralizado (DID) que utilizam um Registro de Dados Verificáveis para obter a chave. A associação é feita pela assinatura é fundamental para assegurar tanto a autenticidade quanto a integridade da credencial, garantindo que qualquer entidade verificadora possa, posteriormente, confirmar que a credencial foi emitida por uma fonte legítima e que não sofreu alterações.

Após a emissão, a VC é entregue ao titular, que a armazena em uma carteira digital. Esta carteira digital, cuja funcionalidade será detalhada na Subsubseção 2.2.3.1, é um progrma2 específico destinado ao armazenamento seguro das cre-

denciais. Quando o SP exige que determinada informação seja compartilhada ou provada, ele solicita a apresentação de um VP correspondente. O titular, então, gera e compartilha a VP de maneira segura por meio da carteira digital. Utilizando o identificador, o titular recupera a chave pública e com a assinatura anexada à VP, o SP pode verificar instantaneamente a autenticidade e a integridade da credencial apresentada.

Figura 2 – Modelo de Credenciais Verificáveis W3C.



Fonte: Baseado em Sporny et al. (2024)

Contudo, a adoção deste modelo exige que os usuários adotem uma postura mais proativa e participativa na gestão de suas informações pessoais. Essa necessidade de envolvimento direto pode representar um desafio significativo, especialmente para o público leigo que utiliza ambientes digitais sem grande preocupação com as complexidades subjacentes de segurança (SCHARDONG, 2022). A situação é análoga aos desafios enfrentados pelo Pretty Good Privacy (PGP), uma tecnologia de criptografia desenvolvida nos anos 1990 para garantir a segurança e privacidade de e-mails e arquivos. O PGP utiliza um sistema de criptografia de chave pública, permitindo que os usuários troquem informações de forma segura. No entanto, apesar de ser uma ferramenta poderosa, o PGP enfrentou dificuldades de adoção devido à complexidade envolvida na configuração e gestão segura dos pares de chaves, o que limitou seu uso entre o público geral (ALLEN, 2016). O gerenciamento adequado de chaves públicas e privadas é tecnicamente exigente e requer um nível de conhecimento que muitos usuários não possuem, o que pode limitar a adoção ampla de sistemas baseados no modelo Auto-Soberano.

2.2.3.1 Carteiras Digitais

Carteiras digitais surgem no âmbito da IAM com propósitos variados, adaptados a diferentes modelos de uso. De maneira geral, uma carteira digital é um programa que armazena informações de identidade digital, permitindo que usuários gerenciem suas credenciais e realizem transações seguras online (IDENTITY MANAGEMENT INSTITUTE, 2024).

Com as carteiras digitais, é possível obter o armazenamento seguro para material criptográfico associado aos dados de identidade digital, como chaves de assinatura criptográficas. O usuário consegue controlar e gerenciar suas informações, incluindo remover e revisar os dados armazenados na carteira, além de selecionar explicitamente quais dados deseja armazenar ou compartilhar, tanto dentro quanto fora da carteira. (PODGORELEC; ALBER; ZEFFERER, 2022). Além disso, as carteiras digitais também são caracterizadas por sua acessibilidade, sendo utilizáveis através de dispositivos móveis, como smartphones e tablets. Elas frequentemente suportam múltiplas formas de autenticação, incluindo senhas, biometria (como impressão digital e reconhecimento facial) e autenticação multifator (EMUDHRA, 2022).

Portanto, as carteiras digitais representam uma evolução significativa na forma como as identidades digitais são gerenciadas e protegidas, promovendo a descentralização e conferindo maior controle individual sobre dados pessoais. Nesse modelo, o próprio usuário mantém o controle de suas informações, sem depender de uma entidade central para gerenciar ou armazenar seus dados. Assim, a identidade digital é centralizada apenas para o usuário, enquanto o gerenciamento de identidade como um todo se torna mais descentralizado no ecossistema digital.

2.2.3.2 Prova de Zero Conhecimento

Prova de Zero Conhecimento (ZKP) trata-se de um método no qual uma parte, denominada provador, é capaz de demonstrar a outra parte, denominada verificador, a veracidade de uma determinada afirmação, sem revelar qualquer informação adicional além da própria afirmação em questão (PETKUS, 2019). Esse esquema de prova é utilizado de forma ampla em criptomoedas, como ZCash, para garantir a privacidade, e também nos sistemas de identidade digital para realizar a autenticação sem revelar informações de maneira excessiva.

Para garantir a privacidade e a segurança das informações dos usuários em sistemas de identidade digital, existem diferentes tipos de prova que podem ser utilizadas para permitir que os detentores de identidade provem certos atributos de suas credenciais sem revelar os dados completos, como a de igualdade, desigualdade e prova de pertencimento a um conjunto (LODDER, 2018). A prova de igualdade é usada para mostrar que um atributo é igual a um valor específico, como provar que está empregado apenas com uma resposta de afirmação ou negação. A prova de desigualdade demonstra que um atributo numérico está dentro de um certo intervalo sem revelar o valor exato, como provar que uma pessoa possui mais de 21 anos sem informar a idade exata. A prova de pertencimento a um conjunto confirma que um atributo pertence a um grupo específico sem revelar o valor exato, como mostrar que alguém reside em um país europeu sem dizer qual país. Essas técnicas aumentam a confiança e a segurança ao permitir que os usuários compartilhem informações verifi-

cáveis sem expor dados sensíveis desnecessariamente.

Nessa perspectiva esse esquema de prova abre um caminho para uma redução significativamente do risco de ataques de phishing e engenharia social, bem como o vazamento de dados. Isso ocorre porque as ZKPs permitem que os usuários verifiquem suas identidades ou atributos sem revelar informações sensíveis durante o processo de autenticação (JUNIOR; VASCONCELOS; RIBEIRO, 2024). Como as informações secretas nunca são compartilhadas, mesmo em interações com terceiros, os atacantes não têm acesso a dados críticos que poderiam ser usados para fraudes ou ataques direcionados. Esse nível de privacidade e segurança impede que dados confidenciais sejam interceptados ou manipulados, mitigando efetivamente os riscos associados a esses tipos de ameaças.

Fundamentos e Funcionamento

As ZKPs baseiam-se em três propriedades fundamentais: completude, solidez e zero-conhecimento. A completude assegura que, se uma declaração é verdadeira, o verificador aceitará o resultado da prova. A solidez garante que, se a declaração for falsa, o provador não conseguirá criar uma prova falsa para enganar o verificador. Por fim, a propriedade de zero-conhecimento assegura que o verificador não obtém nenhuma informação adicional sobre o provador além do que está explicitamente contido na prova. O processo geral envolve quatro etapas principais: autenticação, troca de provas, verificação e conclusão (JUNIOR; VASCONCELOS; RIBEIRO, 2024).

1. Autenticação: O provador (quem detém a informação) e o verificador (quem precisa confirmar a veracidade) estabelecem um canal de comunicação seguro. O provador prepara a prova baseada na informação que deseja validar.
2. Troca de provas: O provador gera uma prova criptográfica e a envia ao verificador. Esta prova é estruturada de forma que demonstre a veracidade da declaração sem revelar a própria informação.
3. Verificação: O verificador analisa a prova recebida utilizando um algoritmo que confirma se a prova é válida. Esse processo assegura que a prova é consistente com a declaração sem necessitar de acesso à informação subjacente.
4. Conclusão: Se a prova é validada, o verificador aceita a declaração como verdadeira. Caso contrário, a declaração é rejeitada. Importante notar que o verificador não adquire conhecimento adicional sobre a informação que o provador está protegendo.

Essas etapas garantem a segurança e a privacidade das informações, pois o verificador pode confiar na autenticidade da declaração sem comprometer a confidencialidade dos dados.

Protocolos Existentes

Os protocolos ZKPs são amplamente utilizados em diversas aplicações, especialmente em cenários onde a privacidade e a segurança são essenciais. Existem dois tipos de protocolos, sendo estes classificados como interativos ou não interativos. Nos protocolos interativos, a verificação envolve uma série de interações dinâmicas entre o provador e o verificador. Durante esse processo, o verificador faz uma série de desafios, e o provador responde a cada um deles em tempo real, provando passo a passo o conhecimento da informação sem revelá-la. Em contraste, os protocolos não-interativos permitem que o provador prepare uma prova completa e independente, que pode ser verificada posteriormente por qualquer verificador, sem necessidade de interação adicional. Esses protocolos geralmente dependem do uso de funções de hash para gerar provas verificáveis sem interação contínua (PETKUS, 2019).

O exemplo de protocolo não-interativo mais conhecido é o Zero-Knowledge Succinct Non-Interactive Arguments of Knowledge (ZK-SNARK) (JUNIOR; VASCONCELOS; RIBEIRO, 2024) devido à sua capacidade de executar provas extremamente compactas. Por essa razão, são considerados ideais para adoção em sistemas com restrições de armazenamento e largura de banda limitadas.

Atualmente, algumas bibliotecas e ferramentas facilitam a implantação do ZK-SNARK de maneira prática e eficiente em sistemas reais. Essas bibliotecas permitem que desenvolvedores utilizem ZKPs sem precisar de um conhecimento profundo da criptografia subjacente, tornando a tecnologia mais acessível. As mais famosas são: snarkjs e circom. A snarkjs é uma biblioteca JavaScript para a criação e verificação de ZK-SNARK, enquanto a circom é uma linguagem específica de domínio, Domain Specific Language (DSL), projetada para descrever circuitos aritméticos utilizados nos ZK-SNARK (BELLES-MUNOZ et al., 2023).

A principal desvantagem dos ZK-SNARKs é a necessidade de uma configuração inicial confiável, *trusted setup*. Durante essa fase, um conjunto inicial de parâmetros públicos, chamado Common Reference String (CRS), é gerado. A segurança de todo o sistema depende da integridade desse processo. Se o CRS for comprometido, por exemplo, ou se os dados temporários (resíduos tóxicos) gerados durante a configuração não forem destruídos corretamente, uma entidade maliciosa poderia criar provas falsas e enganar o sistema.

Nesse sentido, os protocolos Zero-Knowledge Scalable Transparent ARguments of Knowledge (ZK-STARK) são uma variante mais recente, projetada para superar algumas limitações dos ZK-SNARK, especialmente em termos de transparência e escalabilidade. Considerados resistentes aos ataques de computadores quânticos. Esses argumentos são muito úteis em contextos onde os conceitos de transparência e escalabilidade são críticos, além de possuírem destaque na aplicabilidade e eficiência (JUNIOR; VASCONCELOS; RIBEIRO, 2024). No entanto, as bibliotecas atuais para a

implementação de ZK-STARK em sistemas reais não são tão fáceis e intuitivas, especialmente quando comparadas às bibliotecas snarkjs e circom, que oferecem um nível mais elevado de abstração e facilidade de uso para ZK-SNARK. Essa complexidade adicional representa um desafio significativo para os desenvolvedores que desejam implementar ZK-STARK de maneira eficaz.

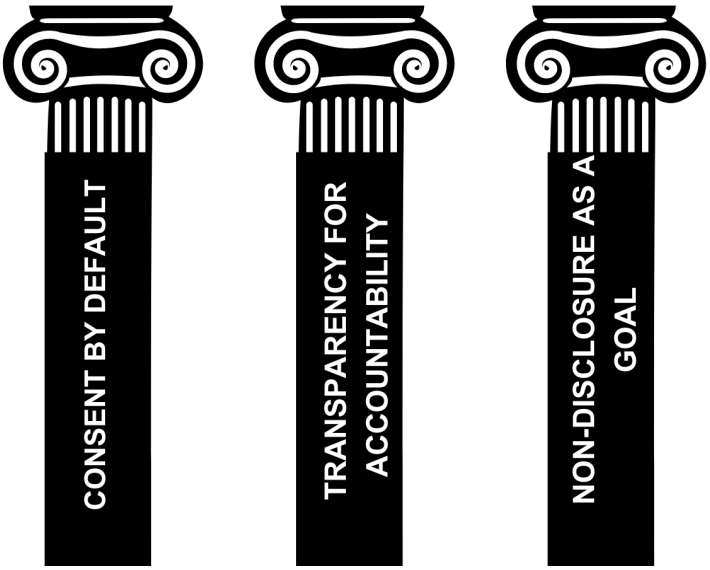
2.2.4 Fiduciário

O quarto e último modelo a ser apresentado representa um conjunto de esforços destinados a superar as limitações observadas nos modelos anteriores. Conhecido como Fiduciário, esse novo paradigma busca encontrar uma solução para a dicotomia entre a experiência do usuário durante os procedimentos de autenticação e manipulação de dados, e o fortalecimento da preservação da privacidade desses dados. Enquanto sistemas de gerenciamento de identidades baseados em modelos terceirizados oferecem uma experiência de navegação satisfatória, facilitada pelos mecanismos de SSO, eles enfrentam dificuldades em evitar a divulgação desnecessária de dados em certos contextos. Em contrapartida, as propostas existentes de SSI conseguem solucionar esse problema, mas apresentam uma experiência de usuário insatisfatória (SCHARDONG; CUSTODIO, 2024), com interações complexas e demoradas com os provedores de serviços e as carteiras.

O modelo fiduciário propõe que a gestão das identidades do usuário deve ser realizada por um novo agente de confiança, conhecido como fiduciário. Este, por sua vez, estabelece um relacionamento que dá origem a seu nome, fiduciário, que estabelece uma relação de confiança e responsabilidade em que uma parte (o fiduciário) tem o dever legal e ético de agir no melhor interesse da outra parte (o beneficiário). Esse tipo de relacionamento é comum em várias áreas, como no direito, na administração de empresas, saúde e na gestão de investimentos, e é ilustrado na Figura 4 através das linhas duplas tracejadas. Este modelo também se caracteriza por inúmeras características de segurança e usabilidade, porém elas podem agrupadas nos seguintes princípios fundamentais: *Consent by Default*, *Transparency for Accountability* e *Non-Disclosure as a Goal*.

O primeiro princípio, *Consent by Default*, dita que todas as ações do fiduciário devem ser guiadas pelos consentimentos que o indivíduo definiu. Isso é implementado por meio de um artefato computacional chamado Política de Consentimento, que estabelece um conjunto de declarações feitas pelo usuário que definem o que pode ser feito com seus dados em termos de coleta, armazenamento e compartilhamento com outras entidades. Por meio deste artefato, o Fiduciário consegue decidir quais credenciais são mais adequadas para serem usadas em determinados contextos, seguindo a orientação do que o titular considera sensível sobre seus dados, possibilitando que ele não perca o controle sobre sua identidade digital. Essa política pode ser definida

Figura 3 – Pilares do Modelo Fiduciário.



Fonte: O Autor

previamente, antes que as interações entre o usuário e SP ocorram, ou sob demanda, caso o Fiduciário não consiga determinar claramente se sua ação pode ou não prejudicar a pessoa.

À primeira vista, a introdução de uma nova entidade responsável pelo gerenciamento dos dados pode parecer um retrocesso, retirando o controle dos indivíduos sobre suas próprias credenciais. Contudo, o objetivo é proporcionar suporte àqueles que enfrentam dificuldades ou preferem não se preocupar com as exigências de segurança relacionadas à manutenção de credenciais. As Políticas de Consentimento são projetadas para garantir que os titulares dos dados tenham todo o controle desejado, permitindo que flexibilizem ou restrinjam as ações do Fiduciário. Dessa forma, o Fiduciário pode atuar tanto como uma carteira digital presente nos modelos de SSI quanto como um elo de confiança, ciente de suas responsabilidades e deveres, capacitado a tomar as melhores decisões em benefício do dono dos dados.

O segundo princípio, *Transparency for Accountability*, estabelece que deve ser viável rastrear de forma precisa as ações realizadas sobre determinados dados, especificando o momento em que ocorreram e para quais finalidades foram utilizadas. Para esse propósito, é definido um artefato computacional denominado *Evidências*, que serve para auditar toda e qualquer atividade realizada pelo fiduciário com os dados dos utilizadores. Através desse mecanismo, é possível realizar um processo sistemático de exame e avaliação das atividades e modificações de dados efetuadas pelo fiduciário, com o objetivo de verificar a conformidade com as políticas de consentimento e boas práticas, bem como avaliar a eficácia, eficiência e integridade dessas

manipulações.

Ao permitir que o Fiduciário tome decisões de forma autônoma, ele pode se basear em solicitações anteriores semelhantes ou em outras métricas, visando reduzir as interrupções constantes aos usuários. Contudo, a autonomia conferida ao Fiduciário não o isenta de cometer erros na utilização dos dados dos indivíduos. Na prática, essa autonomia e os registros das ações garante ao beneficiário que seu representante será responsabilizado caso ocorra qualquer atitude delituosa. Adicionalmente, um aspecto significativo deste novo artefato é a sua capacidade de aprimorar a rastreabilidade das ações de um invasor que consiga comprometer o ambiente de execução do Fiduciário e conseguisse agir como tal agente. Isso se deve ao fato de que todas as atividades realizadas pelo Fiduciário serão registradas de forma detalhada, permitindo uma análise minuciosa e precisa das ações executadas durante a intrusão.

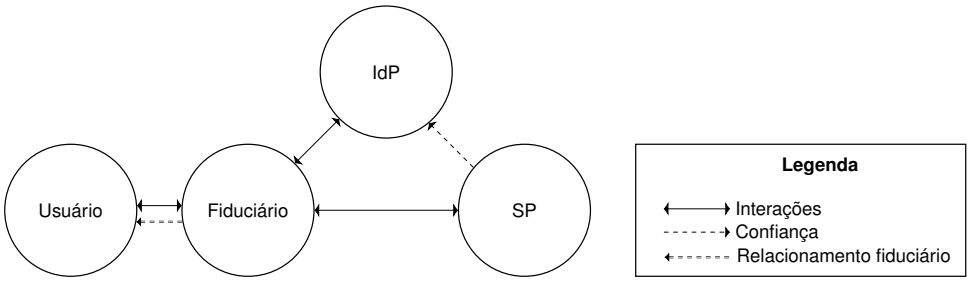
O terceiro princípio, *Non-Disclosure as a Goal*, determina que o principal objetivo do modelo é manter a confidencialidade e evitar a revelação de determinadas informações, seja na construção de novos protocolos ou interfaces que utilizam o modelo como estrutura. Para alcançar esse objetivo, três mecanismos podem ser empregados: Divulgação Seletiva, ZKP (Prova de Conhecimento Zero) e Seleção de Ambiente de Execução. Embora cada um desses mecanismos possa ser aplicado individualmente, a combinação deles potencializa significativamente o grau de preservação das informações dos beneficiários.

A primeira estratégia fundamenta-se na capacidade do Fiduciário de revelar apenas partes específicas das informações do usuário, sem expor a totalidade de seus dados. Para alcançar esse objetivo, busca-se a utilização de VC que disponibilizem mecanismos para a construção de VP que contenham apenas um subconjunto de atributos, em vez de incluir todos os atributos disponíveis. A segunda estratégia é baseada na utilização de ZKP, conforme detalhado na Subsubseção 2.2.3.2, pois apesar de também ser uma solução de divulgação seletiva, essa abordagem reduz de forma significativa a exposição de dados em comparação à seleção de atributos para uma VP. Independentemente do mecanismo adotado, o modelo visa proporcionar ao indivíduo maior controle sobre seus dados pessoais, permitindo-lhe fornecer apenas as informações estritamente necessárias e, assim, minimizar a exposição de dados supérfluos ou potencialmente invasivos.

A terceira estratégia explora a possibilidade de processamento de dados fora do ambiente do SP. Tradicionalmente, o processamento de dados relacionados à identidade ocorre em um ambiente de confiança controlado pelo SP, sem oferecer ao usuário a opção de escolher o ambiente que considera mais seguro, resultando na transferência obrigatória de dados de um ambiente para outro. No entanto, o modelo proposto introduz duas alternativas: a primeira permite que o processamento ocorra em um ambiente de confiança do próprio titular dos dados, em consonância com a proposta de Hardjono e Pentland (2019). A segunda possibilita a computação colabo-

rativa entre diferentes entidades por meio de técnicas seguras de Computação Multipartidária, conhecida em inglês como Multiparty Computation (MPC), assegurando que os dados de cada participante permaneçam privados e confidenciais ao longo de todo o processo.

Figura 4 – Modelo Fiduciário.



Fonte: Inspirado em Schardong (2022)

2.3 PROTOCOLOS DE IDENTIDADE DIGITAL

2.3.1 OAuth 2.0

OAuth 2.0 é um protocolo de autorização que fornece à aplicações a capacidade de acessar um recurso de usuário por meio de tokens, evitando que o indivíduo precise compartilhar a sua credencial de acesso com aplicação. Dessa forma, não há necessidade de compartilhar credenciais sensíveis (HARDT, 2012). Por exemplo, um leitor pode autorizar um aplicativo de notícias à acessar sua conta ou realizar postagens em seu nome na sua conta de sua rede social sem revelar para o aplicativo qual é a senha. Caso o aplicativo de notícias sofra algum tipo de vazamento de dados, a senha da rede social desse leitor continua resguardada no IdP, a rede social.

Para que o protocolo funcione adequadamente, quatro papéis principais são essenciais no processo. Primeiramente, temos o **Proprietário do recurso**, que corresponde ao indivíduo que deseja autorizar uma aplicação a acessar seus dados. No cenário ilustrativo de leitura de notícias, o Proprietário do recurso seria o leitor que pretende permitir que um aplicativo acesse suas preferências de leitura. Em segundo lugar, o **Cliente** é a aplicação que almeja utilizar os dados do indivíduo. Mantendo o exemplo anterior, o cliente seria o aplicativo de notícias que deseja acessar as preferências do leitor para personalizar o conteúdo apresentado. O terceiro papel é desempenhado pelo **Servidor de Autorização**, que é o responsável por autenticar o usuário e fornecer tokens que permitem ao cliente acessar os recursos do indivíduo. Este servidor atua como uma ponte de confiança entre o Proprietário do recurso e o cliente, assegurando que apenas aplicativos devidamente autorizados possam acessar os dados. Finalmente, temos o **Servidor de Recurso**, que é o servidor responsável

por armazenar os dados do indivíduo. Frequentemente, o servidor de recurso coincide com o servidor de autorização, embora possam ser distintos dependendo da arquitetura do serviço implementado. Comparando com o modelo terceirizado da Figura 1, o Servidor de Autorização e o Servidor de Recurso correspondem ao IdP, o Proprietário do recurso representa o Usuário, e o SP é o Cliente.

Além desses papéis, o protocolo utiliza tokens, credenciais que representam a autorização, para gerenciar o acesso aos recursos. Alguns deles são **Código de Autorização**, **Token de Acesso** e o **Token de Atualização**. O Código de Autorização (Authorization Grant) é um token fornecido ao cliente após a autenticação do usuário perante o servidor de autorização. Este código é utilizado pelo cliente para obter um Token de Acesso e se trata de uma sequência de caracteres que comprova que o usuário autorizou o cliente a agir em seu nome. O Código de Acesso (Access Token) é o token que determina o tipo de acesso que o cliente terá sobre os recursos do indivíduo. Este token geralmente inclui informações sobre o escopo específico de acesso, o tempo de validade e outros atributos relevantes; no entanto, o provedor pode personalizar o conteúdo do token conforme suas necessidades. É com este token que o cliente realiza efetivamente o acesso aos dados. Por fim, os tokens de atualização (Refresh Tokens) permitem que a aplicação obtenha novos Tokens de Acesso sem a necessidade de reautenticação do usuário. O uso de Tokens de Atualização reduz a frequência com que o usuário precisa interagir com o processo de autenticação, melhorando a segurança, já que o usuário não precisa fornecer suas credenciais repetidamente.

O OAuth 2.0 utiliza o protocolo HyperText Transfer Protocol (HTTP) para transmitir parâmetros durante o processo de autorização. Quando um cliente inicia uma solicitação de autorização, ele redireciona o navegador do usuário para o Ser3vidor de Autorização utilizando o método HTTP GET, incluindo alguns parâmetros na URL da requisição. Três parâmetros importantes nesse contexto são o **`scope`**, **`response_type`** e **`client_id`**. O parâmetro `scope` define o nível de acesso que o cliente está solicitando, especificando as permissões desejadas. Por exemplo, uma aplicação pode solicitar acesso apenas à leitura de emails ou à leitura e escrita de contatos.

O parâmetro `response_type` define o fluxo de mensagens para autenticação e autorização, determinando o tipo de resposta que o Cliente deve esperar do Servidor de Autorização. Existem quatro fluxos distintos: **Fluxo de Código de Autorização** (Authorization Code Flow), **Fluxo Implícito** (Implicit Flow), **Fluxo de Credenciais de Senha do Proprietário do Recurso** (Resource Owner Password Credentials Flow) e **Fluxo de Credenciais do Cliente** (Client Credential Flow). Assim, dependendo `response_type` escolhido e outros parâmetros, algum desses fluxo poderá ser utilizado. Por exemplo, se o `response_type` é `code`, utiliza-se o Fluxo de Código de Autorização, onde o cliente recebe um código de autorização que será trocado por um Token de Acesso. Já se for o `response_type=token` é utilizado no Fluxo

Implícito, onde o cliente recebe diretamente um Token de Acesso.

Outro parâmetro crucial é o `client_id`, que identifica de forma única a aplicação cliente que está solicitando acesso aos recursos do usuário. Este identificador é fornecido pelo Servidor de Autorização quando a aplicação é registrada. Em combinação com o `client_secret`, que é um segredo conhecido apenas pela aplicação e pelo Servidor de Autorização, o `client_id` ajuda a autenticar a aplicação durante o processo de obtenção do Token de Acesso. Além disso, as requisições ao servidor de autorização geralmente incluem a `redirect_uri`, que é a URL para a qual o usuário será redirecionado após o processo de autenticação e autorização. Esta URL deve ser registrada previamente no Servidor de Autorização e corresponder exatamente ao que foi registrado para evitar ataques de redirecionamento malicioso.

Há também a definição de vários endpoints usados ao longo do processo de autorização. O **Authorization Endpoint** é o URL onde o usuário é redirecionado para fornecer o consentimento e autenticar. Este endpoint processa as solicitações de autorização e emite Códigos de Autorização ou tokens, dependendo do fluxo. O **Token Endpoint** é utilizado para trocar um Código de Autorização por um Token de Acesso, sendo também usado para renovar Tokens de Acesso expirados. O **Resource Endpoint** é onde os recursos protegidos são acessados. O cliente usa o Token de Acesso para solicitar recursos do servidor de recursos.

Esses papéis, artefatos, parâmetros e endpoints permitem construir uma implementação robusta do OAuth 2.0, assegurando acesso seguro e controlado a recursos, sem comprometer as credenciais dos usuários. O protocolo é flexível e adaptável, equilibrando segurança e usabilidade conforme as necessidades das aplicações.

A Figura 5, descrita a seguir, demonstra o funcionamento do Fluxo de Código de Autorização. Nesse fluxo, ocorre a obtenção de um Código de Autorização, que é subsequentemente trocado por um Token de Acesso.

1. O usuário acessa a aplicação e especifica seu provedor de identidade.

2. O frontend do aplicação redireciona o utilizador para o Servidor de Autorização, fornecendo seu identificador, `client_id`.

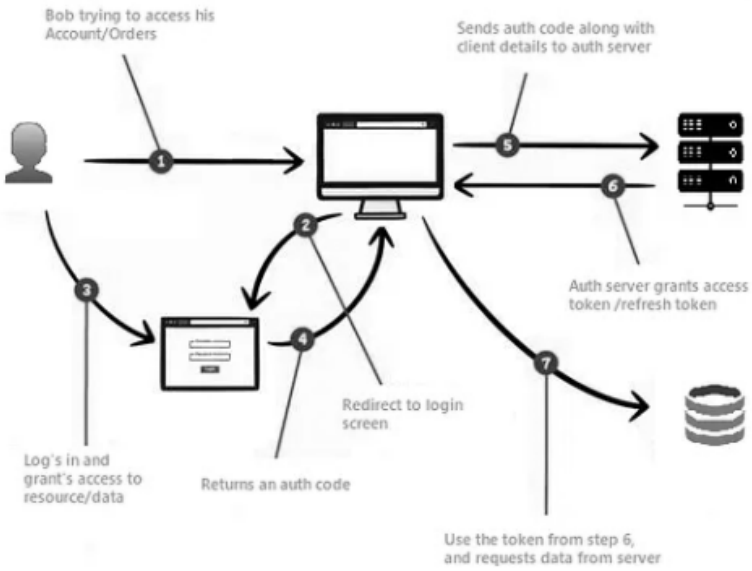
3. O indivíduo autentica-se perante o Servidor de Autorização e concede acesso ao seu recurso para o aplicação.

4. O Servidor de Autorização fornece o token de autorização, que é então encaminhado para o backend da aplicação.

5. O backend do aplicação solicita o token de acesso correspondente ao seu código de autorização para o Servidor de Autorização.

6. O Servidor de Autorização encaminha o código de acesso.

Figura 5 – Fluxo de Código de Autorização no OAuth.



Fonte: Chinta (2024)

7. O aplicação solicita ao Proprietário do recursos o acesso aos dados utilizando seu código de acesso. Se bem-sucedido, o recurso é fornecido para o aplicação.

2.3.2 OpenID Connect

OpenID Connect (OIDC) é um protocolo de autenticação baseado na família de especificações OAuth 2.0, permitindo que as aplicações autentiquem usuários e obtenham informações sobre eles, proporcionando uma experiência de SSO (SAKIMURA et al., 2014). Ele é amplamente adotado por grandes provedores de identidade, como Google, Microsoft e Facebook, proporcionando uma interoperabilidade robusta e simplificada entre diversas plataformas e serviços, facilitando a integração e melhorando a segurança na autenticação de usuários em aplicações web e móveis.

Um novo token é definido chamado de `id_token` para identificar os usuários para a aplicação. Este token está organizado em uma estrutura de JSON Web Token (JWT) que contém informações sobre a autenticação do usuário, incluindo o emissor do token, o público-alvo representado por um `client_id`, um identificador único do usuário final no IdP, a data e hora da autenticação e os tempos de emissão e expiração do token. Para emitir esse token, novos fluxos também são definidos estendendo a especificação do parâmetro `response_type`. No OAuth o valor de `response_type` é code ou token. O OIDC adiciona o `id_token`, e permite qualquer combinação de code, token e `id_token`. Um valor especial, none, também é adicionado.

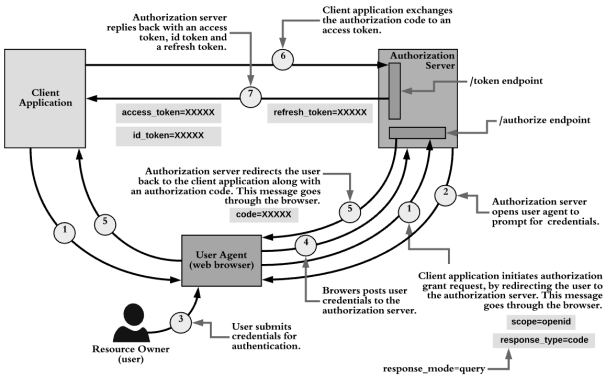
Também é adicionado o valor **openid** para o parâmetro `scope`, que é utili-

zado para definir as permissões específicas que uma aplicação está solicitando em relação aos dados do usuário. Especificamente, este valor permite que a aplicação obtenha um `id_token`, que contém informações autenticadas sobre o usuário, como seu identificador único, e possivelmente outras informações adicionais se escopos adicionais como profile, email ou address forem incluídos. O escopo profile permite que a aplicação acesse uma variedade de informações de perfil básicas sobre o usuário. Isso pode incluir o nome completo, apelido, foto de perfil, gênero, data de nascimento, idioma preferido e outras informações de perfil. A inclusão deste escopo proporciona uma visão mais completa do usuário. O escopo email concede à aplicação acesso ao endereço de e-mail do usuário e a verificação se o endereço de e-mail foi verificado. Especificamente, a aplicação pode obter o endereço de e-mail e um indicador booleano (email_verified) que informa se o endereço foi verificado. Isso é útil para aplicações que necessitam de um contato confiável com o usuário ou para validação de identidade. O escopo address permite que a aplicação acesse o endereço físico do usuário. Isso pode incluir informações como a rua, cidade, estado, código postal e país. A inclusão deste escopo é útil para aplicações que requerem informações de envio ou para personalizar ofertas e serviços baseados na localização do usuário.

Ademais, também é exigido a obrigatoriedade do `redirect_uri` como medida de segurança para garantir que as Respostas da Autorização sejam enviadas apenas para URLs previamente autorizados. Isso ajuda a prevenir ataques como a homem no meio (man-in-the-middle) e a ataque de redirecionamento, onde um atacante poderia tentar interceptar ou redirecionar as respostas de autorização para um local malicioso.

A Figura 6, descrita a seguir, demonstra o funcionamento do Fluxo de Código de Autorização no conxtexto do OIDC. Nesse fluxo, ocorre a obtenção de um Código de Autorização, que é subsequentemente trocado por um `id_token`.

Figura 6 – Fluxo de Código de Autorização no OIDC.



Fonte: Siriwardena (2020)

1. O titular acessa a aplicação, que realiza uma requisição de Código de Autorização, redirecionando-o para o endpoint /authorize do Servidor de Autorização.

2. O Servidor de Autorização abre o navegador para solicitar credenciais, geralmente um nome de usuário e uma senha.

3. O indivíduo autentica-se perante o Servidor de Autorização.

4. A pessoa encaminha suas credenciais e concede acesso às informações pessoais para a aplicação.

5. O Servidor de Autorização fornece o token de autorização. Essa mensagem é enviada pelo navegador e encaminha o token de autorização para o backend da aplicação.

6. O backend da aplicação solicita o token de acesso correspondente ao seu código de autorização para o provedor de identidade no token endpoint.

7. O Servidor de Autorização verifica o token de autorização e, caso esteja de acordo, encaminha o token de acesso, juntamente com `id_token` e o refresh_token.

8. A aplicação solicita ao Servidor de Autorização acesso às informações adicionais sobre o usuário, utilizando seu `id_token` no endpoint UserInfo.

2.3.3 OIDC4VC: O OpenID Connect no modelo SSI

O OpenID Connect com Credenciais Verificáveis (OIDC4VC) representa um conjunto de protocolos avançados projetados para estabelecer um ecossistema de SSI (YASUDA et al., 2022). Esse sistema capitaliza as funcionalidades robustas e amplamente validadas do OpenID Connect, amplamente adotado no modelo terceirizado de autenticação, oferecendo uma experiência segura e eficiente para a gestão de identidades digitais. Ao integrar Credenciais Verificáveis e, consequentemente, Apresentações Verificáveis ao OIDC, o OIDC4VC proporciona um mecanismo poderoso e flexível, permitindo que os indivíduos autentiquem e compartilhem suas identidades com uma combinação de segurança, privacidade e conveniência. Essa abordagem garante que apenas os dados estritamente necessários sejam compartilhados em cada transação, oferecendo aos usuários um controle refinado sobre suas interações digitais.

O problema do protocolo OIDC dentro do modelo Terceirizado é que ele levanta preocupações significativas sobre a redução de privacidade causada pela centralização de controle de credencias e a dificuldade de gerenciamento das mesmas em diferentes provedores. Durante o processo de autenticação usando o OIDC, informações sensíveis sobre o indivíduo podem ser inadvertidamente compartilhadas com o IdP que atua como Servidor de Autorização, como detalhes sobre quais serviços online a pessoa acessa (YASUDA et al., 2022, Página 7). Isso ocorre porque a sua

arquitetura centralizada concentra o controle das identidades digitais no IdP (Servidor de Autorização).

Cada provedor emite e gerencia suas próprias credenciais de forma isolada, o que impede a unificação ou combinação de atributos provenientes de diferentes fontes em uma única credencial ou token. Por exemplo, um usuário pode ter uma credencial emitida por um banco que contém informações financeiras e outra credencial emitida por uma rede social com dados pessoais. No modelo OIDC, esses atributos permanecem separados, e o usuário não tem uma maneira eficiente de combiná-los em um único artefato que possa ser utilizado para uma verificação mais simplificada de sua identidade em diversos contextos. Essa limitação não apenas aumenta a complexidade do gerenciamento de identidades, mas também restringe a flexibilidade e a utilidade das credenciais digitais, dificultando a criação de experiências de usuário mais integradas e seguras.

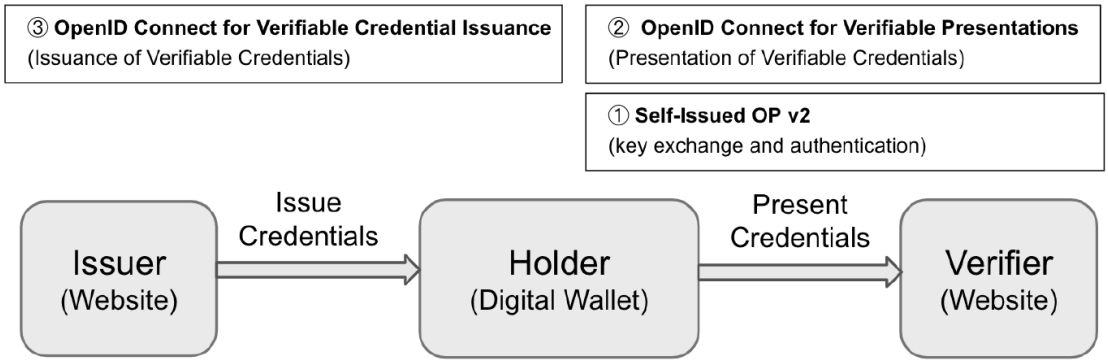
O OIDC4VC visa abordar esses desafios de maneira eficaz, especialmente ao redefinir a responsabilidade do envolvida na emissão, armazenamento e apresentação das credenciais. Ao restringir o papel de IdP a apenas emitir e delegar aos usuários a tarefa de armazenar e apresentar suas credenciais por meio de carteiras digitais, o novo ecossistema permite que os indivíduos exerçam um controle mais rigoroso sobre suas informações pessoais. Essa mudança possibilita que os usuários compartilhem seus dados diretamente com os serviços, de acordo com suas necessidades, garantindo que apenas as informações estritamente necessárias sejam divulgadas no momento oportuno. Essa abordagem fortalece a autonomia do usuário e promove um ambiente mais seguro e privado para a gestão de identidades digitais.

Ao empregar VPs, os protocolos mitigam o problema da fragmentação das credenciais emitidas por diferentes provedores, possibilitando a unificação e combinação de atributos em um único artefato digital. Essa abordagem permite que o usuário consolide informações derivadas de múltiplas credenciais emitidas por diversas fontes, como exemplificado pela integração de dados provenientes de um banco e de uma rede social em uma única VP. Dessa forma, o usuário pode fornecer, em suas interações com serviços online, uma representação mais completa e integrada de sua identidade. As VP aprimoram o gerenciamento de identidades ao permitir que o usuário selecione e combine apenas os atributos necessários para cada contexto específico, promovendo uma experiência mais fluida e segura, ao mesmo tempo em que preservam a privacidade e mantêm o controle sobre as informações compartilhadas.

2.3.3.1 Ecossistema

Para sustentar essa estrutura, o OIDC4VC estabelece uma série de protocolos e extensões que integram as funcionalidades do OpenID Connect ao modelo de Credenciais Verificáveis, conforme ilustrado na Figura 7.

Figura 7 – Ecossistema do OIDC4VC.



Fonte: Yasuda et al. (2022)

1. **Self-Issued OpenID Provider v2 (SIOPv2)**: É uma especificação que permite aos usuários atuarem como seus próprios IdP, emitindo e gerenciando suas credenciais diretamente sem depender de um provedor de identidade terceirizado. Com ele o usuário pode gerar e assinar atributos como nome, e-mail, sem que sejam verificadas por uma entidade central. Além de conseguir provar que realmente controla a chave privada associada a essa assinatura (prova de posse) (YASUDA; JONES; LODDERSTEDT, 2023).

   O SIOPv2 permite uma ampla variedade de arquiteturas de carteiras. Por exemplo, as carteiras podem ser executadas no dispositivo do usuário, mas também podem ser hospedadas na nuvem. A experiência do usuário pode ser fornecida por meio de um aplicativo móvel ou uma aplicação web. Do ponto de vista do protocolo, elas podem utilizar todos os fluxos do OpenID Connect, ou seja, há uma grande variedade de opções para atender às necessidades da respectiva implantação e dos casos de uso.

2. **OpenID para Apresentações Verificáveis (OIDC4VP)**: Extensão do OpenID Connect para permitir os serviços solicitarem e receberem Apresentações Verificáveis (YASUDA; LODDERSTEDT, 2023b).

3. **OpenID para Emissão de Credenciais Verificáveis (OIDC4CI)**: Esta especificação propõe uma API destinada à emissão de VCs em diversos formatos, incluindo o do W3C (SPORNY et al., 2024). As VCs são emitidas por meio de uma carteira digital, previamente autorizada pelo usuário através do protocolo OAuth 2.0. Essa abordagem visa aproveitar as características comprovadas de segurança, simplicidade e flexibilidade inerentes ao protocolo OAuth 2.0 (YASUDA; LODDERSTEDT, 2023a).

   Embora seja viável autorizar a emissão de credenciais conforme os serviços acessados requeiram informações do usuário, como ocorre no modelo terceiri-

zado atual, a proposta das carteiras digitais visa proporcionar um maior controle ao usuário. Nesse modelo, os usuários armazenarão suas credenciais e gerarão VPs individualizadas para cada serviço utilizado, o que reduz significativamente a possibilidade de os IdPs monitorarem quais serviços são acessados e com que frequência isso ocorre.

2.3.3.2 OIDC4VP

OIDC4VP amplia as capacidades do OpenID ao permitir a apresentação de Credenciais Verificáveis na forma de Apresentações Verificáveis. Essa extensão oferece uma série de funcionalidades que reforçam a flexibilidade e a interoperabilidade do sistema, como a compatibilidade com todos os fluxos do OpenID Connect, o suporte a diferentes formatos de VCs e VPs, a capacidade de utilizar múltiplos métodos de transporte e o reaproveitamento do parâmetro claims para definir a sintaxe de solicitações. Nos parágrafos subsequentes, será detalhado cada um desses itens para fornecer uma compreensão mais profunda de suas funcionalidades e benefícios.

O protocolo é projetado para ser compatível com todos os fluxos do OpenID Connect. Isso significa que, independentemente do fluxo de autenticação escolhido, desde aqueles mais comuns como o Fluxo de Código de Autorização até naqueles em que o usuário é seu próprio provedor de identidade (SIOPv2), OIDC4VP pode ser integrado de maneira eficaz, garantindo flexibilidade e interoperabilidade. Ademais, a sintaxe das solicitações reaproveita o parâmetro claims do OIDC e utiliza a especificação DIF Presentation Exchange (DIF, 2024) para formatar VPs.

O sistema oferece suporte a diferentes formatos de credenciais e apresentações, permitindo codificações em JavaScript Object Notation (JSON) ou JSON for Linked Data (JSON-LD), bem como assinaturas utilizando JSON Web Signature (JWS) ou Linked Data Proofs. A capacidade de operar com diversos formatos e métodos de assinatura amplia significativamente a interoperabilidade do sistema, facilitando sua adaptação a diferentes padrões e requisitos técnicos. Por exemplo, o JSON é amplamente utilizado em diversas aplicações web devido à sua simplicidade e eficiência (BRAY, 2017), enquanto o JSON-LD é preferido em contextos que demandam maior semântica e interconectividade de dados (SPORNY et al., 2020). A escolha de formato é uma característica importante, porque determina se uma credencial suporta propriedades que reduzem a exposição de dados (Authlete, 2023), como a Divulgação Seletiva e ZKP descritas na Subseção 2.2.3.

A flexibilidade robusta na transmissão de VCs e VPs, suportando múltiplos métodos de transporte, é uma característica essencial. As informações podem ser integradas diretamente no `id_token` ou na resposta do Userinfo, permitindo uma integração transparente com os fluxos de autenticação já existentes. Alternativamente, as credenciais podem ser transmitidas por meio de um token dedicado, o

`redirect_uri`, que pode ser retornado juntamente com o `id_token` a partir dos endpoints de autorização ou token. Independentemente da forma que essa VP seja encaminhada, para que a carteira saiba quais atributos são necessários, dentro da Requisição de Autorização, é reutilizado o parâmetro claims do OIDC para conter um `redirect_uri` com um **presentation_definition**. Esse último contém uma Definição de Apresentação (DA) (DIF, 2024), que é uma estrutura que define como as apresentações devem ser enviadas para o SP.

2.3.3.3 Funcionamento

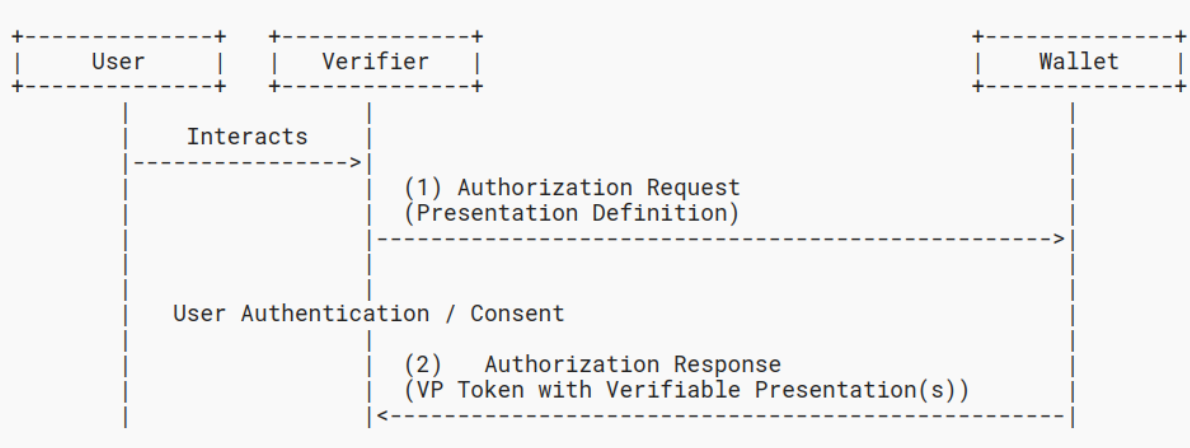
No contexto desta especificação, a Carteira desempenha a função de Servidor de Autorização OAuth 2.0 em relação ao SP, que, por sua vez, atua como Cliente OAuth 2.0 (YASUDA; LODDERSTEDT, 2023b). Essa distinção é significativa, pois, em outras especificações, como no OIDC4CI, a Carteira assume o papel de Cliente OAuth 2.0. Ademais, com essa definição, os SP podem usufruir dos mecanismos existentes para facilitar o registro de clientes, como especificado no OpenID Connect Discovery (OIDC-D) (SAKIMURA et al., 2023) e OpenID Connect Dynamic Client Registration (OIDC-DCR) (SAKIMURA; BRADLEY; JONES, 2023)

As especificações OIDC-D e OIDC-DCR desempenham papéis complementares no ecossistema de autenticação do OpenID Connect. O OIDC-D possibilita que os clientes descubram automaticamente as configurações e endpoints do provedor de identidade por meio de um documento de descoberta, facilitando a configuração e integração com o sistema de autenticação. Já o OIDC Dynamic Client Registration permite que clientes sejam registrados de forma dinâmica e automática com o provedor de identidade, sem a necessidade de intervenção manual, o que é particularmente útil em ambientes com alta demanda de novos clientes. Juntas, essas especificações tornam a gestão de autenticação e autorização mais ágil, flexível e escalável.

O funcionamento desse protocolo é estruturado em dois fluxos distintos. O primeiro, denominado **same-device flow**, refere-se ao processo no qual o usuário apresenta uma credencial a um SP utilizando o mesmo dispositivo em que sua carteira digital está instalada. Nesse fluxo, tanto o SP quanto a Carteira interagem diretamente no dispositivo do usuário, empregando redirecionamentos simples para a troca da Solicitação de Autorização (Authorization Request) e da Resposta de Autorização (Authorization Response) entre ambas as partes. Esse fluxo é exemplificado na figura Figura 8. O segundo fluxo, conhecido como **cross-device flow**, ocorre quando o SP e a Carteira residem em dispositivos distintos. Nesse contexto, o SP gera uma Solicitação de Autorização e a apresenta ao usuário na forma de um código QR (QR Code). O usuário, então, utiliza a Carteira em seu dispositivo para escanear o código QR e iniciar o processo de verificação, permitindo a continuidade da interação de forma segura e eficiente.

O protocolo introduz novos tipos de resposta (`response_type`) para suportar a transmissão de tokens de apresentação verificável (`vp_token`) em diferentes cenários de autenticação. Quando o valor de `response_type` é `vp_token`, a VP é retornada diretamente na Resposta de Autorização (Authorization Response). No caso em que o valor de `response_type` é `redirect_uri id_token` e o parâmetro `scope` contém `openid`, o `redirect_uri` é retornado na Resposta de Autorização juntamente com um ID Token autoemitido, conforme definido no (YASUDA; JONES; LODDERSTEDT, 2023). Além disso, quando o `response_type` é **code**, seguindo o Fluxo de Código de Autorização, o `redirect_uri` é fornecido na Resposta de Token (Token Response), após a troca bem-sucedida do código de autorização pelo token correspondente. Nesse exemplo, o `response_type` é `redirect_uri`.

Figura 8 – Fluxo com Carteira e SP no mesmo dispositivo.



Fonte: Yasuda e Lodderstedt (2023b)

1. O SP representado por um verficador, realiza um **Solicitação de Autorização** (Authorization Request) no **Endpoint de Autorização** para a carteira digital. Nesse caso, o parâmetro `response_type` inclui o valor `vp_token`, enquanto o `presentation_definition` contém a DA que define os critérios que as VPs devem cumprir, como o tipo de credencial exigido, o formato de apresentação e os atributos específicos das credenciais que devem ser compartilhados.

2. A Carteira retorna a Resposta de Autorização, *Authorization Response*. Nesse caso, com o `response_type` definido como `vp_token`, o valor de `vp_token` é enviado diretamente.

   O SP realiza a verificação da vinculação do titular (holder binding), bem como da integridade e autenticidade da credencial antes de seu processamento. Os procedimentos específicos a serem adotados variam conforme o formato da credencial, o esquema criptográfico utilizado e o mecanismo de revogação empre-

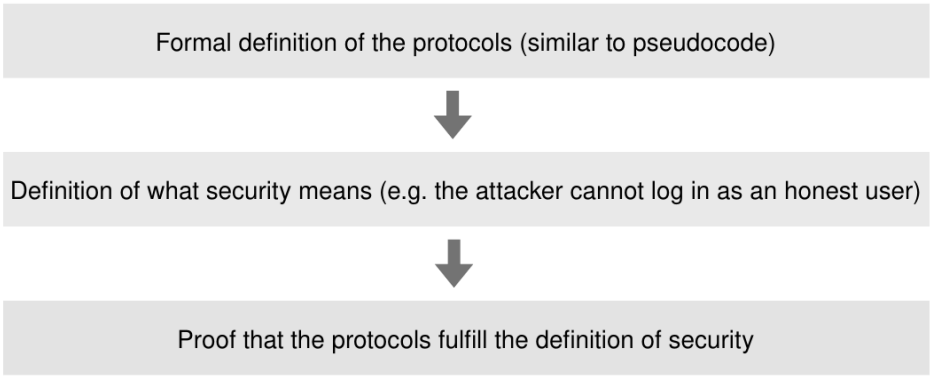
gado, os quais estão além do escopo definido pelo OIDC4VP (YASUDA et al., 2022).

2.4 ANÁLISE FORMAL DE SEGURANÇA

A análise formal de segurança é um processo sistemático utilizado para garantir que protocolos atendam a requisitos de segurança específicos. O objetivo é verificar, de maneira matemática ou lógica, que um sistema se comporta corretamente (KULIK et al., 2020). Diferente das abordagens empíricas, como testes de penetração, que se concentram em identificar vulnerabilidades conhecidas, a análise formal permite detectar vulnerabilidades anteriormente desconhecidas, proporcionando uma garantia de segurança significativamente superior (HAUCK, 2023). Isso ocorre porque, enquanto os testes empíricos são limitados a cenários e ameaças específicas, a análise considera o comportamento completo do sistema, baseando-se em provas rigorosas, oferecendo uma segurança mais abrangente e fundamentada em teorias robustas.

A análise envolve três etapas, ilustrado na Figura 9. A primeira é a **modelagem**, que consiste em criar uma representação abstrata do sistema, descrevendo suas operações, interações e comportamentos possíveis. A segunda é a **especificação de propriedades** em que são definidas, de forma precisa, as propriedades que o sistema deve atender em um contexto específico, como "um atacante não pode se autenticar como um usuário legítimo". Por fim, as **provas** que garantem que o modelo formal respeite as especificações de segurança definidas.

Figura 9 – Etapas da Análise Formal de Segurança.



Fonte: Hauck (2023)

Para ilustrar o funcionamento desse tipo de análise, será reproduzida uma parte da Prova de Autenticação de Apresentação (Presentation Authentication) conforme descrita em (HAUCK, 2023). O objetivo deste capítulo não é fornecer uma explicação detalhada e exaustiva da prova, mas sim uma compreensão intuitiva do processo de análise. Este capítulo é, portanto, dividido em três subseções. Na Subseção 2.4.1, será apresentado Web Infrastructure Model (WIM) utilizado como base para definir formalmente o OIDC4VP, em um formato similar a pseudo-código. Na Sub-

seção 2.4.2 será especificado o significado da Autenticação de Apresentação neste contexto, e, finalmente, na Subseção 2.4.3, será apresentada a prova.

2.4.1 Definição Formal de Protocolos Usando WIM

O modelo WIM segue uma abordagem baseada no modelo Dolev-Yao (DV), capturando padrões web amplamente utilizados, como HTTP e HTML. A Figura 10 ilustra a arquitetura do modelo. Nele, cada entidade é representada por um processo que monitora um ou mais endereços IP e processa eventos correspondentes. Um evento é composto por uma mensagem e pelos endereços IP do remetente e do destinatário. Em cada etapa de processamento, um evento é selecionado de maneira não determinística de uma lista de eventos e entregue ao processo apropriado. A entidade processa o evento e gera um ou mais novos eventos, que são, então, adicionados à lista para processamento subsequente (FETT; KUSTERS; SCHMITZ, 2014).

Para modelar o OIDC4VP, o comportamento do atacante e do navegador web são especificados como processos. O processo do atacante é um processo Dolev-Yao não determinístico que registra todas as mensagens que recebe e gera todos os eventos que pode derivar das mensagens registradas. Mais formalmente, um processo atacante (I, Z, R, s_0) que recebe um evento de entrada e_{in} em um estado s gera o novo estado $s' = \langle e_{in}, E_{out}, s\rangle$ e os eventos $E_{out} = \{\langle a_1, f_1, m_1\rangle, \ldots, \langle a_n, f_n, m_n\rangle\}$ para algum $n \geq 0$, onde os endereços do remetente $f_1, \ldots, f_n$ são escolhidos de forma não determinística a partir de I, os endereços do destinatário $a_1, \ldots, a_n$ são escolhidos não deterministicamente entre todos os endereços IP, e as mensagens $m_1, \ldots, m_n$ são escolhidas não deterministicamente de $d(\{e_{in}\} \cup \{s\})$ (FETT; KÜSTERS; SCHMITZ, 2014, Seção 2.5). Dessa forma, um processo atacante é capaz de realizar todos os ataques que qualquer processo Dolev-Yao poderia executar, exceto quebrar criptografia.

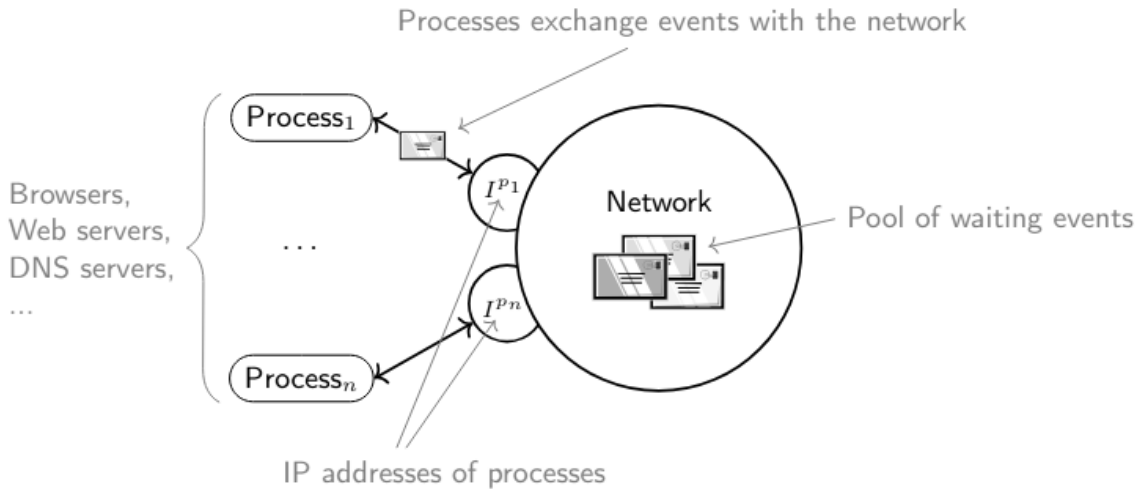
Nesta análise, utiliza-se o modelo de atacante de rede, que tem acesso a todos os endereços IP, permitindo-lhe interceptar mensagens destinadas a outras partes e falsificar os endereços dos remetentes. Em redes reais, esses ataques podem incluir ARP-spoofing, onde o atacante redireciona tráfego em redes locais ao falsificar a correspondência entre endereços IP e endereços MAC, e ações de adversários patrocinados por estados-nação, que exploram infraestrutura de telecomunicação para interceptar e manipular dados em larga escala.

O atacante pode, ainda, corromper qualquer processo legítimo, ganhando acesso completo ao estado da entidade comprometida. Isso é representado pelo envio de uma mensagem especial, $m =$ CORRUPT, ao processo correspondente (FETT; KÜSTERS; SCHMITZ, 2014, Seção 2.5).. Após receber essa mensagem, o processo começa a atuar como um processo atacante, utilizando o último estado disponível. Os detalhes sobre como ocorre a corrupção variam de acordo com o modelo específico

do processo.

O navegador web faz parte de um sistema que formaliza a infraestrutura da web e suas aplicações associadas. Para simplificar o modelo, o WIM inclui um servidor HTTPS genérico, responsável, entre outras funções, por receber e enviar requisições HTTPS.

Figura 10 – Modelo WIM.



Fonte: (FETT; KUSTERS; SCHMITZ, 2014)

2.4.2 Propriedades de segurança

Uma etapa crucial para a comprovação de segurança é definir o que ela significa em um contexto específico, o que se faz por meio da identificação das propriedades de segurança relevantes. No caso do OIDC4VP, uma dessas propriedades é a **Autenticação de Apresentação** (Presentation Authentication), que garante que um atacante não consiga se passar por um usuário legítimo para acessar uma aplicação. Em um ambiente web, essa propriedade é comprometida se um atacante obtiver posse de um cookie de sessão associado a um ID de usuário legítimo.

2.4.3 Prova de segurança

Para demonstrar a segurança da Autenticação de Apresentação, são estabelecidas duas premissas: (1) o usuário é perfeito, ou seja, ele sempre mantém o controle de quais fluxos iniciou; e (2) as entidades Browser, Carteira e Verificador são honestas, ou seja, não divulgam informações. Com base nessas premissas, deve-se provar que o atacante não consegue obter um cookie de sessão vinculado à identidade de um usuário legítimo. Isso ocorre porque as sessões são tipicamente identificadas por

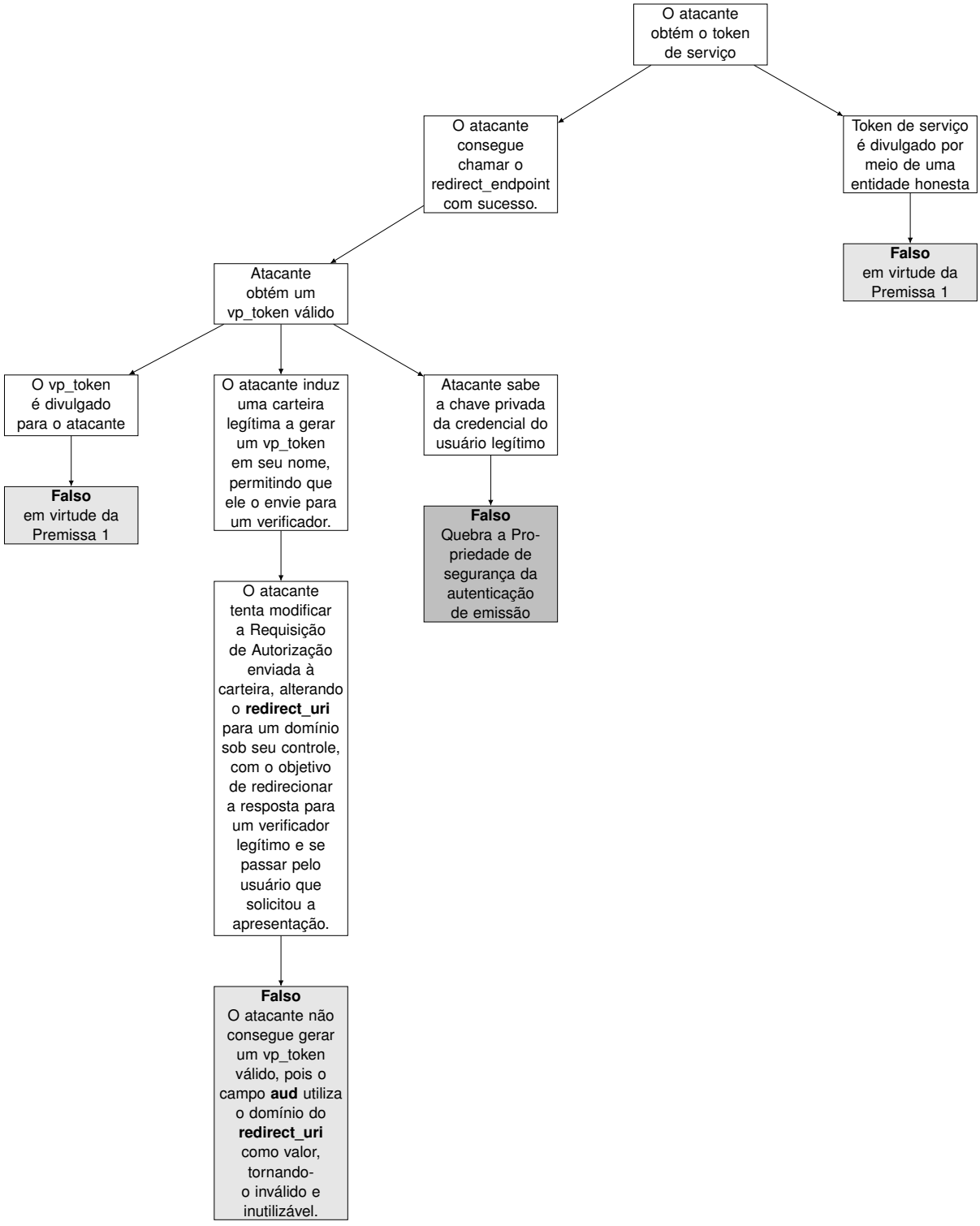
um nonce armazenado no navegador do usuário como um cookie. Se um atacante obtiver acesso a esse cookie, ele pode se passar pelo usuário legítimo, reutilizando o cookie comprometido para acessar a sessão ativa no sistema e, assim, obter os mesmos privilégios e acessos que o usuário original.

O cookie é gerado apenas no endpoint de redirecionamento (redirect endpoint), onde o verificador recebe a resposta de autorização (Authorization Response). Para que o atacante obtenha esse cookie, existem duas opções: a primeira seria que uma das partes honestas vazasse o cookie de sessão, o que é prevenido pela premissa (2); a segunda seria o atacante conseguir utilizar o redirect endpoint do verificador com sucesso. No entanto, isso só pode ser feito por meio de uma resposta de autorização contendo um **`redirect_uri`**.

Para que o atacante obtenha um **`redirect_uri`** com a identidade de um usuário legítimo, a possibilidade de uma entidade honesta divulgá-lo é descartada, conforme descrito em (1). Além disso, o atacante poderia tentar acessar a chave privada associada à chave pública da credencial do usuário, o que também é impedido pela **Propriedade de Emissão Autenticada**, detalhada em (HAUCK, 2023). Outra opção seria induzir uma carteira legítima a criar um **`redirect_uri`** arbitrário.

Uma das táticas que o atacante pode usar para induzir uma carteira a gerar um **`redirect_uri`** envolve o uso de um ataque de phishing. Nesse cenário, o atacante engana o usuário para que ele clique em um link ou visite uma página maliciosa, desencadeando involuntariamente um processo de autenticação com sua carteira digital. Durante esse processo, a carteira recebe uma requisição de autorização com o parâmetro **`redirect_uri`** direcionado para um domínio controlado pelo atacante. No entanto, ao obter o **`redirect_uri`**, o atacante não consegue utilizá-lo, pois o parâmetro **aud** (audience) é preenchido com o domínio do **`redirect_uri`**, tornando o ataque ineficaz.

Figura 11 – Prova de Autenticação da Apresentação.



Fonte: O Autor

3 TRABALHOS RELACIONADOS

3.1 OPEN ALGORITHMS

O Open Algorithms (OPAL) é uma iniciativa do MIT Media Lab que visa permitir o uso de dados sensíveis e pessoais para análises, assegurando ao mesmo tempo a privacidade dos indivíduos. O projeto adota alguns princípios fundamentais dentro do seu modelo, como o de deslocar os algoritmos para os dados. Este princípio sugere que, em vez de transferir dados de vários repositórios para um local centralizado para processamento, os algoritmos devem ser direcionados aos repositórios de dados para serem processados localmente. Dessa forma, busca-se garantir que informações pessoais não sejam transferidas em seu formato bruto, mas apenas as análises e percepções (insights) derivadas desses dados. Esse método preserva a privacidade ao permitir que apenas resultados agregados sejam retornados após a execução dos algoritmos.

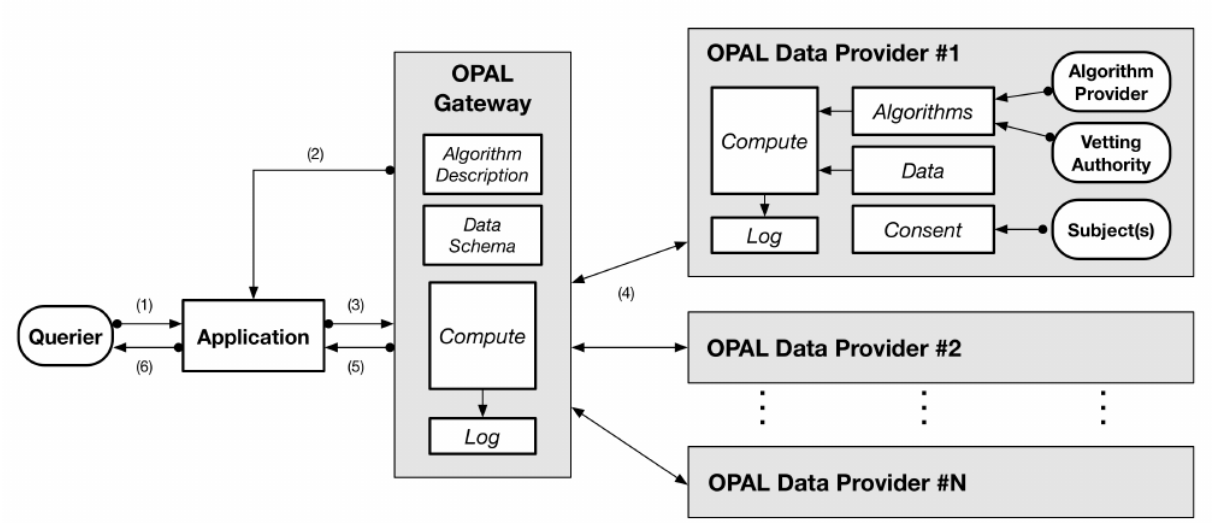
Para que isso ocorra, um dos principais conceitos introduzidos para o tratamento de dados é o de **Algoritmos Verificados** (Vetted Algorithms). Este conceito estabelece que os algoritmos utilizados no ecossistema são previamente revisados e aprovados por especialistas, conhecidos como **Autoridade Verificadora** (Vetting Authority), que centralizam a decisão sobre quais algoritmos devem ou não ser empregados. Essa abordagem visa, sobretudo, assegurar a qualidade dos algoritmos sob a perspectiva de viés, injustiça, e outros possíveis efeitos colaterais não intencionais ou imprevistos.

Outro aspecto importante abordado é o atual isolamento dos dados. Sugere-se que percepções mais aprofundadas podem ser alcançadas quando há uma integração de dados de diferentes domínios — como dados de saúde, financeiros, redes sociais, etc. A ideia é que, ao combinar e analisar esses dados em conjunto, torna-se possível identificar padrões, correlações e tendências que não seriam detectáveis ao se considerar apenas um único conjunto de dados. Por exemplo, ao combinar dados de compra provenientes de diferentes lojas e bancos, as empresas podem obter uma compreensão mais abrangente dos comportamentos de consumo, o que permite a implementação de campanhas de marketing mais eficazes e personalizadas.

A concepção geral desse ecossistema é estabelecer dois atores principais: o **Provedor de Dados** (Data Provider) e o **Serviço de Dados OPAL** (OPAL Data Service), que podem ser a mesma entidade ou entidades distintas. A interação entre esses atores e outras entidades é representada na Figura 12. O Consultor, geralmente representado como SP em aplicações web, que deseja obter informações, utiliza a Aplicação no Passo 1 para selecionar um ou mais algoritmos e os dados desejados (Passo 2). Em seguida, o Consultor usa a Aplicação para transmitir essas seleções ao Serviço de Dados, também chamado de OPAL Gateway, no Passo 3. O Serviço de

Dados, então, interage com os Provedores de Dados relevantes no Passo 4 para executar a solicitação. Finalmente, o Serviço de Dados envia a resposta para a Aplicação e o Consultor no Passo 5.

Figura 12 – Ecossistema OPAL.



Fonte: Hardjono e Pentland (2019)

3.2 MACHINE READABLE PERSONAL PRIVACY TERMS

Machine Readable Privacy Terms Working Group (MRPTWG) é um grupo de trabalho da IEEE que está desenvolvendo o padrão Machine Readable Personal Privacy Terms. Apesar de ainda não ter nenhuma publicação, o objetivo descrito em seu resumo é tratar de como os termos de privacidade pessoal são estruturados, buscando possibilitar que sejam compreendidos e aceitos por máquinas. Essa abordagem permitiria que dispositivos e sistemas em uma rede concordassem de maneira clara e precisa sobre a divulgação e o uso de informações. A relevância e o impacto desse padrão dependem, evidentemente, de sua implementação prática, o que o torna um tópico de interesse significativo para a evolução das políticas de privacidade digital. Portanto, merece ser mencionado aqui.

3.3 COMPARAÇÃO ENTRE PROPOSTAS

A Tabela 2 oferece uma comparação entre as duas propostas em relação a diferentes aspectos. Na coluna intitulada **Nível de Atuação**, observa-se a primeira grande diferença entre as propostas. Enquanto a OPAL é descrita como um modelo, o *Protocolo para Negociação de Atributos e Ambiente de Execução* é classificado como um protocolo. Essa distinção é crucial, pois destaca o objetivo principal de cada solução proposta. Em geral, um modelo está primariamente preocupado em descrever

Tabela 2 – Comparação entre Propostas

Proposta	**Nível de Atuação**	**Ambiente de Execução**	**Dados Compartilhados**	**Algoritmos Verificados**
OPAL	Modelo	Provedor de Dados	sim	sim
Machine Readable Personal Privacy Terms	Padrão	Sem informações	Sem informações	Sem informações
Novo Protocolo para Modelo Fiduciário	Protocolo	SP, Fiduciário ou ambos	não	não

Fonte: O Autor

um ecossistema de forma abstrata, enfatizando seus princípios e fundamentos, ao invés de se concentrar em como implementar o modelo de maneira prática. Por outro lado, um protocolo aprofundará a descrição oferecida por um modelo, visando facilitar sua implementação prática.

Em termos de **Ambiente de Execução**, o OPAL opera exclusivamente no Provedor de Dados, o que sugere uma mudança drástica em relação ao cenário atual. Por outro lado, o novo protocolo pode ser implementado no SP, no fiduciário ou em ambos, o que indica uma transição mais gradual para a execução de algoritmos em ambientes que oferecem maior confiança ao usuário. Na coluna **Dados Compartilhados**, o OPAL possibilita o compartilhamento de dados, alinhando-se com práticas que promovem a colaboração. Em contraste, o novo protocolo não apresenta mecanismos específicos para o compartilhamento de dados. No entanto, o Modelo Fiduciário não impõe restrições a esse respeito, pois é viável que um fiduciário, com usuários de diferentes domínios, permita o compartilhamento de suas credenciais por meio de sua Política de Consentimento, visando adquirir novas análises sobre os dados.

Na coluna **Algoritmos Verificados**, o OPAL também restringe seu escopo de atuação ao exigir, como princípio básico, a verificação de algoritmos. Essa exigência, por um lado, contribui para garantir a qualidade e a integridade das análises realizadas; por outro, pode atrasar o desenvolvimento de algoritmos mais atualizados e eficientes. Em contraste, o novo protocolo não incorpora nenhuma funcionalidade similar para a verificação de algoritmos, permitindo que o SP apresente sua solução desejada, enquanto o fiduciário avalia se a proposta está de acordo com as definições de transparência e com as restrições impostas pelo seu beneficiário.

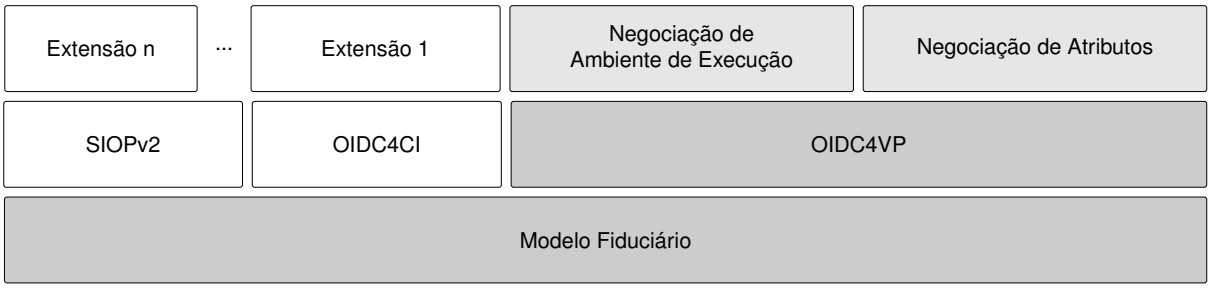
4 UM NOVO PROTOCOLO PARA IDENTIDADE FIDUCIÁRIA

O novo Protocolo para Identidade Fiduciária propõe uma modificação no protocolo OIDC4VP, parte integrante do ecossistema OIDC4VC, para adequá-lo ao Modelo Fiduciário. Através dessa adaptação facilitará a construção de uma camada de apresentação de credenciais de maneira simplificada, segura e amigável para os desenvolvedores. Destaca-se também a capacidade de reutilizar a infraestrutura existente, além do fácil acesso a uma vasta gama de códigos e bibliotecas desenvolvidas com base na especificação central do OpenID, o que amplifica sua adoção e integração. Essas características tornam o protocolo não apenas uma solução técnica eficiente, mas também uma abordagem estratégica que potencializa a escalabilidade e a interoperabilidade dentro do ecossistema de identidades digitais.

A nova especificação introduz duas extensões alinhadas às estratégias da Subseção 2.2.4 para a preservação da privacidade dos usuários: a *Negociação de Atributos* e a *Negociação do Ambiente de Execução*. A primeira utiliza a estrutura de DAs em conjunto com a definição de novas mensagens e endpoints, visando à redução do risco de uso indevido de dados, garantindo que somente as informações estritamente necessárias sejam compartilhadas. Já a segunda extensão, também baseada em endpoints e mensagens, permite que dados sensíveis sejam preferencialmente processados em ambientes confiáveis ao usuário.

Essas adaptações são representadas na Figura 13, que adota uma abordagem modular como solução para atingir as propriedades e comportamentos definidos pelo modelo. Embora o foco esteja no OIDC4VP, destacado em tons de cinza, deixa-se em aberto a viabilidade de explorar as demais soluções mencionadas em Subseção 2.2.4, representadas em branco, para atender aos demais princípios do modelo. Essa solução visa não apenas cumprir os requisitos de segurança, mas também aprimorar a confiança do usuário no sistema, ao assegurar que sua privacidade seja rigorosamente protegida em todas as etapas do processo.

Figura 13 – Pilha: Modelo, protocolos, extensões.



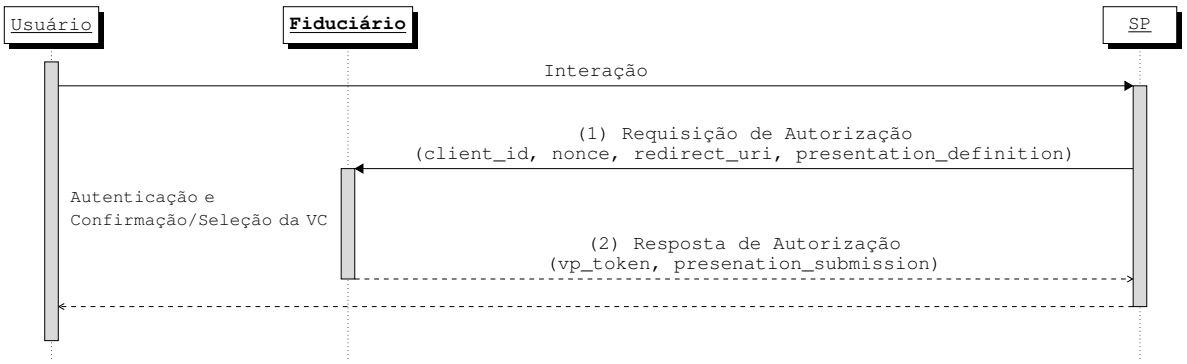
Fonte: O Autor

4.1 INTEGRAÇÃO DO OIDC4VP COM O MODELO FIDUCIÁRIO

A transição do modelo original de SSI, no qual o OIDC4VP foi originalmente proposto, para o Modelo Fiduciário, é um processo bastante natural como mostrado na Figura 14. Isso ocorre porque esta adaptação foca exclusivamente na redução da divulgação de dados proposto por *Non-Disclousure as Goal*, sem exigir, por exemplo, modificações nas mensagens trocadas pelo Fiduciário como solução para garantir maior controle sobre as ações realizadas com os dados do usuário, conforme previsto no princípio de *Transparency for Accountability*. Dessa forma, não foram necessárias alterações significativas, permitindo que a SP continue desempenhando seu papel como a entidade responsável por solicitar, receber e validar as VPs. No contexto do OIDC4VP, essa função é exercida por meio do papel de Cliente OAuth.

O Fiduciário, por sua vez, possui as mesmas funções e capacidades que a Carteira, conforme discutido anteriormente na Subseção 2.2.4, incluindo receber, armazenar, apresentar e gerenciar as VCs, bem como as chaves criptográficas associadas. Portanto, no contexto do OIDC4VP, deve assumir o papel Servidor de Autorização que é realizado pela Carteira dentro desse contexto. No entanto, com a prioridade de, além de adquirir um mecanismo para entregar VPs, integrar as extensões de *Negociação de Atributos* e *Negociação do Ambiente de Execução* nos fluxos existentes, visando atender o pilar *Non-Disclosure as Goal*.

Figura 14 – Fluxo com Fiduciário no lugar da Carteira.



Fonte: O Autor

4.2 USO DE PROVAS DE CONHECIMENTO ZERO

A utilização da especificação OIDC4VP no Modelo Fiduciário visa aproveitar a flexibilidade e a concepção agnóstica em relação ao formato das Credenciais Verificáveis utilizadas. Isso significa que sua adaptação ao modelo não exige diretamente o uso de Provas de Conhecimento Zero; ao contrário, proporciona um ambiente que permite a integração de VCs que possuam essa capacidade. Como o OIDC4VP não

estabelece requisitos específicos para a implementação de ZKP, cabe aos formatos de VCs adotados determinar se oferecem suporte ou não para essa tecnologia. O uso dessa tecnologia é recomendado especialmente em contextos que exigem altos níveis de privacidade e segurança na verificação das credenciais, permitindo que apenas as informações estritamente necessárias sejam divulgadas.

4.3 NEGOCIAÇÃO DE ATRIBUTOS

Quando um Fiduciário recebe uma Requisição de Autorização é possível que a Definição de Apresentação solicite informações do usuário as quais não estejam de acordo com as Políticas de Consentimento definidas previamente. Por exemplo, é comum que Provedores de Serviço realizem a solicitação da data de nascimento dos usuários apenas para verificar sua maioridade. Este tipo de requisição é invasiva e pode estar contra os desejos do indivíduo. Dessa forma, a extensão **Negociação de Atributos** permite definir quais informações sobre os usuários são relevantes para a plataforma que está oferecendo serviços sem comprometer a privacidade e o sigilo dos dados do dono desses recursos.

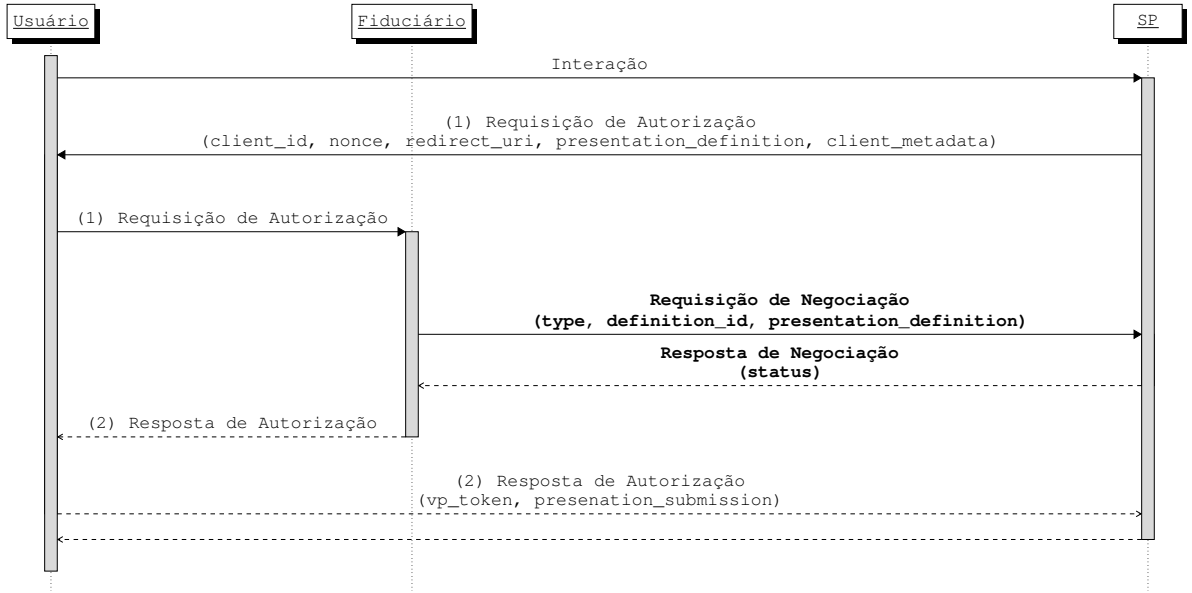
Esse mecanismo está centrado na estrutura de DAs, que no modelo Fiduciário são chamados de **Declarações de Requisitos** (Requirements Statements). Neste texto, os termos DAs e Declarações de Requisitos serão tratados como sinônimos. Com essa estrutura, os Fiduciários conseguem definir quais informações requisitadas não estão em conformidade com as Políticas de Consentimento e propor novos que satisfaçam as intenções. Para que isso seja possível, são definidos o endpoint Negociação (Negotiation Endpoint), a Solicitação de Negociação (Negotiation Request) e sua respectiva resposta, Resposta de Negociação (Negotiation Response). Essa nova requisição, iniciada pelo Fiduciário e respondida pelo SP, ocorre entre a Requisição de Autorização e sua respectiva resposta, conforme destacado na Figura 15.

4.3.1 Endpoint de Negociação

Este endpoint é utilizado para negociar Declaração de Requisitos previamente encaminhada em casos os quais o Fiduciário identifica alguma divergência do que foi requisitado e o que é autorizado dentro das Políticas de Consentimento. A comunicação com o Endpoint de Neogicação deve utilizar TLS.

Para o Fiduciário obter o *Endpoint de Negociação* do Provedor de Serviços, este texto define o parâmetro `negotiation_endpoint`, que pode ser adquirido por dois meios. O primeiro é pelo mecanismo tradicional de registro OIDC-D de (SAKIMURA et al., 2023), no qual os Clientes OAuth fornecem seus metadados aos Servidores de Autorização. O segundo é por meio do parâmetro `client_metadata`,

Figura 15 – Fluxo para Negociação de Atributos.



Fonte: O Autor

conforme definido na seção 5 (YASUDA; LODDERSTEDT, 2023b), enviado na Requisição de Autorização e contendo os metadados do SP.

- **negotiation_endpoint**: URL do Endpoint de Negociação do Provedor de Serviços. Esta URL deve usar o esquema HTTPS e pode conter componentes de porta, caminho e parâmetros de consulta. Se omitido, o provedor não oferece suporte a esse endpoint e não está em conformidade com as propriedades do Modelo Fiduciário.

Figura 16 – Exemplo de parâmetro `negotiation_enpoint` no `client_metadata` (em UTF-8).

```
1 GET /authorize?
2   client_id=client.example.org
3   &client_metadata=%7B%22vp_formats%22%3A%7B%22jwt_vp_json%22%3A%7B%22alg%22%3A%5B%22EdDSA%22%
      2C%22ES256K%22%5D%7D%7D%2C%22negotiation_endpoint%22%3A%22https%3A%2F%2Fexample.com%22%7
      D
4   &request_uri=https%3A%2F%2Fclient.example.org%2Frequest%2Fvapof4ql2i7m41m68uep
5   &request_uri_method=post HTTP/1.1
```

Fonte: O Autor

4.3.2 Solicitação de Negociação

A Solicitação de Negociação constitui uma requisição HTTP POST enviada pelo Fiduciário ao SP, com o tipo de mídia `application/json`. Os parâmetros utilizados na solicitação são os seguintes:

Figura 17 – Solicitação de Negociação.

```
POST /negotiate HTTP/1.1
Host: example.com
Content-Type: application/json

{
  "type": "attribute",
  "definition_id": "8xLOxBtZp8",
  "presentation_definition": "..."
}
```

Fonte: O Autor

- **type**: OBRIGATÓRIO. Este parâmetro deve ser utilizado na Solicitação de Negociação para especificar o tipo de negociação que está sendo realizada. No contexto da *Negociação de Atributos*, deve-se utilizar o valor `attribute`.
- **definition_id**: OPCIONAL. Trata-se de uma string que identifica uma Declaração de Requisitos previamente encaminhada ao Fiduciário. Após o acordo ser firmado entre o Fiduciário e o Provedor, o SP deve invalidar o `definition_id`. Este parâmetro se torna OBRIGATÓRIO quando o valor de type é `attribute`.
- **presentation_definition**: OPCIONAL. Este parâmetro contém um objeto JSON de Declaração de Requisitos, conforme a sintaxe estabelecida na (DIF, 2024). Este parâmetro se torna OBRIGATÓRIO quando o valor de type é `attribute`.

4.3.3 Resposta de Negociação

O Provedor de Serviços possui a prerrogativa de aceitar ou recusar a proposta de alteração da Declaração de Requisitos. Em determinados casos, o Fiduciário aceita a proposta que contém a nova Declaração de Requisitos e responde com uma mensagem com tipo de mídia `application/json` contendo em seu corpo o JSON com o parâmetro status indicando a aceitação da sugestão de declaração e o código de status HTTP 201 sinalizando a criação de um novo recurso para conseguir receber a nova declaração, que é diferente daquele que ele enviou.

Figura 18 – Resposta de Negociação com Sucesso.

```
HTTP/1.1 201 Accepted
Content-Type: application/json

{
    "status": "accepted"
}
```

Fonte: O Autor

- **status**: OBRIGATÓRIO. A semântica desse campo irá depender da negociação definida em type. Nesse exemplo, indica se o SP aceitou ou recusou a nova

proposta para a Declaração de Requisitos. O código de status em formato ASCII selecionado entre as duas opções abaixo:

- **accepted**: Indica que a proposta para a nova Declaração de Requisitos foi aceita pelo SP. Nesse caso, o processo de alteração da Declaração de Requisitos será iniciado, e o recurso associado pode ser atualizado ou criado conforme necessário.
- **refused**: Indica que a proposta para a nova Declaração de Requisitos foi recusada pelo SP. Nesse caso, nenhuma alteração será realizada, e a Declaração de Requisitos permanecerá como está. O motivo da recusa pode ser detalhado em um campo adicional de erro.

Em outros casos, o SP pode rejeitar a proposta por não concordar com os novos sugeridos. Nessas situações, a resposta HTTP deve utilizar o parâmetro status rejeitando a declaração e o código de status HTTP 400 (Bad Request) incluindo os seguintes parâmetros no corpo da resposta codificada em JSON.

Resposta de Recusa de Negociação

Se a Solicitação de Negociação não for aceita ou for considerada inválida, o SP deverá definir o campo de status como `refused` e incluir os campos `error` e `interval`, conforme ilustrado abaixo.

Figura 19 – Resposta de Negociação com Recusa.

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
    "status": "refused",
    "error": "invalid_negotiation_request",
    "interval": 10
}
```

Fonte: O Autor

- **Error**: OBRIGATÓRIO. O parâmetro de erro deve conter um único código de erro em formato ASCII selecionado a partir da lista a seguir:
  - **negotiation_request_denied**: A semântica desse campo irá depender da negociação definida em type. Nesse contexto, indica que a Solicitação de Negociação com a nova Declaração de Requisitos não foi aceita pelo provedor de serviços.
  - **invalid_negotiation_request**: A Solicitação de Negociação está incompleta, com falta de um parâmetro obrigatório, inclui um parâmetro ou valor

de parâmetro não suportado, repete o mesmo parâmetro ou está de outra forma malformada.

  - **unsupported_definition**: A Declaração de Requisitos contida na Solicitação de Negociação contém um formato que não é reconhecido ou suportado pelo SP. Esse erro acontece quando é utilizada uma versão desatualizada ou por não seguir a sintaxe e os padrões estabelecidos.
  - **expired_definition_id**: Esse erro indica que o `definition_id` fornecido na Solicitação de Negociação já expirou. Isso ocorre quando a negociação associada a esse `definition_id` já foi concluída e o identificador pode ser invalidado para impedir que novas solicitações sejam feitas com base em um estado anterior.

- **error_description**: OPCIONAL. O parâmetro `error_description` deve ser um texto ASCII, fornecendo informações adicionais para ajudar os implementadores do Fiduciário a entender o erro ocorrido. Ele pode incluir espaços, caracteres de pontuação e símbolos comuns, mas não pode incluir caracteres como o caractere aspas duplas ("), barra invertida (\), ou qualquer caractere de controle esteja fora dos intervalos `%x20-21 / %x23-5B / %x5D-7E`, onde o prefixo `%` indica valores hexadecimais da tabela ASCII.

- **interval**: OBRIGATÓRIO. A resposta de erro deve também conter o parâmetro `interval`, que determina o tempo mínimo em segundos que o Fiduciário deve aguardar antes de enviar uma nova solicitação ao Endpoint de Negociação.2

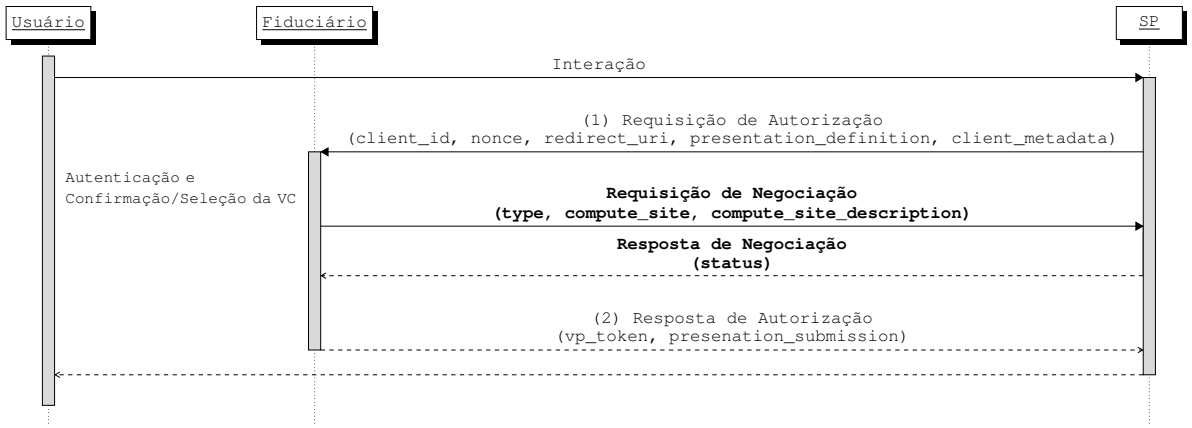
Se o Fiduciário e o Provedor de Serviços não chegarem a um acordo sobre a Declaração de Requisitos, ambos continuarão a trocar mensagens até atingirem um limite estipulado por uma das partes. Assim, recomenda-se fortemente que o Provedor de Serviços adote práticas flexíveis, solicitando apenas as informações estritamente necessárias. Caso contrário, a ausência de um acordo pode resultar na perda de acesso ao serviço pelo usuário, o que pode afetar negativamente a percepção pública da marca, gerando comentários desfavoráveis e reclamações que podem se espalhar rapidamente.

4.4 NEGOCIAÇÃO DO AMBIENTE DE EXECUÇÃO

As arquiteturas tecnológicas vigentes, de modo geral, adotam a lógica de que a manipulação dos dados dos usuários deve ocorrer exclusivamente no lado do SP, centralizando o processamento e o armazenamento das informações. No entanto, novas alternativas estão surgindo e possibilitam uma mudança nesse paradigma, como é o caso da proposta de execução dentro do Fiduciário, visto na Subseção 2.2.4, e o uso de MPC. Essas soluções permitem que o processamento de dados sensíveis seja realizado de forma descentralizada e segura, protegendo a privacidade do usuário ao restringir o acesso direto aos dados pelo provedor de serviços e permitindo que o processamento ocorra de maneira distribuída e controlada.

Nesse sentido, a especificação de **Negociação do Ambiente de Execução** viabiliza esses dois paradigmas dentro do Modelo Fiduciário, possibilitando que o beneficiário decida o local (Fiduciário ou SP) e forma (tradicional ou MPC) apropriada para o processamento de suas informações privadas. O objetivo dessa mudança é manter o paradigma atual em operação, enquanto abre espaço para a integração dessas novas abordagens em serviços compatíveis. A especificação está centrada nas solicitações e endpoints mencionados na Subseção 4.3.2 e possibilita o usuário decidir em qual local é o melhor para que suas informações sejam processadas, conforme pode ser visto na Figura 20.

Figura 20 – Fluxo para Negociação de Ambiente de Execução.



Fonte: O Autor

4.4.1 Solicitação de Negociação

Essa extensão fará uso da Solicitação de Negociação descrita em Subseção 4.3.2. Nesse contexto, o valor atribuído a `type` será **env**. Consulte o exemplo apresentado abaixo.

Figura 21 – Solicitação de Negociação com `type=env`.

```
POST /negotiate HTTP/1.1
Host: example.com
Content-Type: application/json

{
  "type": "env",
  "computer_site": "fiduciary"
}
```

Fonte: O Autor

- **compute_site**: OBRIGATÓRIO. Indica o local em que as manipulações de dados serão realizadas. O valor de `compute_site` deve estar no formato ASCII, escolhido entre as três opções a seguir:

  - **sp**: Indica que a manipulação de dados do usuário será realizada em um ambiente de confiança gerido pelo Provedor de Serviços. Caso o Fiduciário não faça uma proposta, este é o mecanismo padrão utilizado pelo provedor.
  - **fiduciary**: Indica que a manipulação de dados do usuário será realizada em um ambiente de confiança gerido pelo Fiduciário.
  - **both**: Indica que a manipulação de dados do usuário será realizada de forma colaborativa em um ambiente de confiança gerido tanto pelo Fiduciário quanto pelo Provedor de Serviços. Quando essa opção for selecionada, é necessário especificar o parâmetro `compute_site_description`.

4.4.2 Resposta de Negociação

Essa extensão utilizará a Resposta de Negociação, conforme descrito em Subseção 4.3.2. Nesse contexto, o campo `status` indica se o SP aceitou ou rejeitou a nova proposta sobre o modo e o local de execução. O valor do `status` é representado em formato ASCII e pode assumir uma das duas opções: `accepted` ou `refused`.

Figura 22 – Resposta de Negociação com Sucesso.

```
HTTP/1.1 201 Accepted
Content-Type: application/json

{
    "status": "accepted"
}
```

Fonte: O Autor

4.4.2.1 *Resposta de Sucesso para Computação Multipartidária*

Se a Solicitação de Negociação com o parâmetro `compute_site` definido como both for verificada e considerada válida pelo SP, ele deverá incluir o campo **compute_site_description** em sua Resposta de Negociação, conforme ilustrado abaixo.

Figura 23 – Resposta de Negociação com `compute_site_description`.

```
HTTP/1.1 202 Accepted
Content-Type: application/json

{
    "status": "accepted",
    "compute_site_description":...
}
```

Fonte: O Autor

- **compute_site_description**: Esse parâmetro é um JSON que deve ser especificado de acordo com os requisitos estabelecidos pelo SP para disponibilizar a computação Multipartidária. A responsabilidade pela definição do formato deste JSON recai sobre os implementadores, que deverão garantir que ele atenda aos critérios e necessidades específicas do SP e do Fiduciário.

4.4.2.2 *Resposta de Recusa para Execução em outros Ambientes*

Se a Solicitação de Negociação não for aceita ou for considerada inválida, o SP pode reutilizar alguns dos campos estabelecidos Seção 4.3.3, além dos códigos de erro descritos a seguir. Abaixo está um exemplo de recusa.

- **negotiation_request_denied**: A interpretação deste campo dependerá do tipo de negociação especificado em `type`. Neste contexto, ele indica que a Solicitação de Negociação referente ao novo local de execução não foi aceita pelo Provedor de Serviços.
- **not_supported**: Este parâmetro indica que o Provedor de Serviços carece de mecanismos necessários para realizar uma computação do tipo MPC.

4.5 ANÁLISE FORMAL DE SEGURANÇA

Esta seção apresenta a prova formal de segurança para os módulos de negociação apresentados acima. Ela utiliza o modelo formal do protocolo OIDC4VP, apresentado na Subseção 2.4.1, para demonstrar que as propriedades de segurança definidas a seguir na Subseção 4.5.1 não podem ser comprometidas por um atacante.

Figura 24 – Resposta de Recusa.

```
HTTP/1.1 400 Bad Request
Content-Type: application/json

{
    "status": "refused",
    "error": "negotiation_request_denied",
    "interval": 5
}
```

Fonte: O Autor

Essa prova oferece garantias de segurança para as novas extensões dentro protocolo OIDC4VP. Neste capítulo, as ideias principais das provas formais são explicadas de maneira resumida e organizadas em uma estrutura em árvore. A versão completa e detalhada das prova de segurança pode ser encontrada no Apêndice A.

4.5.1 Propriedade de Segurança: Autenticação da Negociação

Em alto nível, a propriedade de Autenticação da Negociação é dizer que um atacante não pode negociar com um verificador se passando por um Fiduciário legítimo. O atacante quebra essa propriedade quando consegue utilizar o `definition_id` associado a um ID de Sessão.

4.5.2 Prova da Autenticação da Negociação

a segurança de uma definição de apresentação (pd) em um sistema de autenticação, mostrando que informações sensíveis são protegidas contra vazamentos para terceiros não autorizados, inclusive potenciais atacantes. A análise foca em como as comunicações entre o navegador, o verificador e o fiduciário são protegidas. Utilizando conexões HTTPS, esses dados são criptografados e acessíveis apenas para as partes legítimas, conforme demonstrado pelo Lema 1 de Fett, Kusters e Schmitz (2014).

O sistema garante que somente scripts confiáveis das origens autorizadas podem acessar pd, e esses scripts não manipulam ou divulgam a informação. Além disso, o fiduciário valida pd de acordo com suas restrições de segurança antes de criar qualquer nova definição de apresentação, protegendo a integridade dos dados ao longo do processo de autenticação. Assim, a prova conclui que pd permanece seguro e inacessível para qualquer agente malicioso, assegurando que apenas o Fiduciário honesto pode negociar Definições de Apresentações.

4.6 LIMITAÇÕES E ESFORÇO PARA MUDANÇA

A transição de um paradigma para outro fator que é sempre difícil de mensurar. Embora as facilidades oferecidas pelo modelo OIDC4VP ajudem e agilizem esse processo, ainda é um desafio convencer todos os interessados de que essa mudança é benéfica não apenas do ponto de vista do usuário. Frequentemente, é do interesse extremo dos SP manter toda a manipulação dos dados dos usuários sob uma infraestrutura de sua confiança. Isso torna mais conveniente para eles não oferecer o serviço a determinados usuários do que abdicar do controle que possuem atualmente sobre os dados.

Em resumo, apesar dos benefícios claros do novo protocolo, como maior segurança e usabilidade, a transição exige esforços consideráveis em termos de mudanças técnicas, culturais e operacionais. Para uma adoção bem-sucedida, é crucial que os benefícios a longo prazo sejam claramente comunicados e que haja um suporte contínuo para superar as barreiras iniciais de implementação.

4.7 CONCLUSÕES

Este trabalho apresenta uma adaptação do OpenID Connect com Apresentações Verificáveis dentro do Modelo Fiduciário, resultando em um mecanismo flexível para a transmissão de credenciais e apresentações, com suporte à divulgação seletiva de informações e a Zero-Knowledge Proofs (ZKP). As modificações propostas não apenas permitem que os usuários negociem os atributos que desejam disponibilizar ao SP em VPs, mas também sugerem uma metodologia para a negociação do ambiente de execução conforme descrito pelo novo modelo.

Além disso, foi realizada uma análise formal de segurança na extensão proposta para o protocolo OpenID para Apresentações Verificáveis. Para essa análise, definiu-se a propriedade de Autenticação da Negociação e provou-se que ela é mantida dentro dos limites do modelo formal. Essa análise demonstrou que a extensão proposta oferece uma segurança no contexto do modelo definido. Considerando que esses protocolos contribuem significativamente para o avanço do Modelo Fiduciário, esta pesquisa também se torna um incentivo relevante para o desenvolvimento de novas soluções no campo da IAM.

Trabalhos Futuros

Uma das direções promissoras é investigar o suporte dos protocolos para incorporar regras de consentimento específicas utilizadas pelos Fiduciários. Esse aprimoramento permitiria que os protocolos reconhecessem e aplicassem automaticamente políticas de consentimento conforme definidas por diferentes Fiduciários, promovendo um maior alinhamento com o Modelo Fiduciário.

Outra possibilidade é expandir a análise formal para cobrir propriedades relacionadas à negociação do ambiente de execução. Isso incluiria a definição e validação de propriedades que assegurem a integridade e a conformidade do ambiente em que a negociação ocorre, garantindo que todos os elementos envolvidos na transação sigam as especificações de segurança e privacidade necessárias. Esse tipo de análise pode fortalecer a proteção contra ambientes inseguros ou comprometidos, proporcionando garantias adicionais para as partes envolvidas e consolidando a segurança dos protocolos frente a cenários de execução complexos e distribuídos.

REFERÊNCIAS

ALLEN, C. **The Path to Self-Sovereign Identity**. 2016. https://www.lifewithalacrity.com/article/the-path-to-self-sovereign-identity/. Acessado em 10 de julho, 2024.

Authlete. **OpenID for Verifiable Credential Issuance (OID4VCI) - Access Token Issuance Details**. 2023. Acessado: 13 de agosto de 2024. Disponível em: https://www.authlete.com/developers/oid4vci/#24-access-token-issuance-details.

BELLES-MUNOZ, M. et al. Circom: A robust and scalable language for building complex zero-knowledge circuits. October 30 2023.

BERTINO, E.; TAKAHASHI, K. **Identity Management: Concepts, Technologies, and Systems**. Norwood, MA, USA: Artech House, 2009. ISBN 978-1-59693-318-5.

BOSCH. **Self-Sovereign Identities**. 2024. https://www.bosch.com/stories/self-sovereign-identities/. Acessado: 10 de julho de 2024.

BRAY, T. **The JavaScript Object Notation (JSON) Data Interchange Format**. Internet Engineering Task Force (IETF), 2017. Request for Comments: 8259. Disponível em: https://www.rfc-editor.org/rfc/rfc8259.txt.

CHINTA, M. **Understanding OAuth 2.0: Step-by-Step Gude**. 2024. Acessado em 06 de Abril, 2024. Disponível em: https://medium.com/codenx/oauth-2-0-4cddd6c7471f.

DIF. **FPresentation Exchange 2.1.0**. [S.l.]: GitHub, 2024. https://github.com/decentralized-identity/presentation-exchange.

Dock.io. **Self-Sovereign Identity: The Ultimate Guide 2024**. 2024. Acessado: 18 de julho de 2024. Disponível em: https://www.dock.io/post/self-sovereign-identity.

El Jaouhari, S.; BOUABDALLAH, A.; BONNIN, J.-M. Chapter 14 - security issues of the web of things. In: **Managing the Web of Things**. [S.l.]: Morgan Kaufmann, 2017.

EMUDHRA. **The Future of Digital Wallets and Identity Management**. 2022. Acessado: 15 de julho de 2024. Disponível em: https://emudhra.com/blog/the-future-of-digital-wallets-and-identity-management.

FETT, D.; KÜSTERS, R.; SCHMITZ, G. An expressive model for the web infrastructure: Definition and application to the browserid SSO system. **CoRR**, 2014.

FETT, D.; KUSTERS, R.; SCHMITZ, G. The web infrastructure model (wim). In: **Proceedings of the IEEE Symposium on Security and Privacy (S&P)**. [S.l.: s.n.], 2014.

FINKELSTEIN, M. E.; FINKELSTEIN, C. Privacidade e lei geral de proteÇÃo de dados pessoais / privacy and general personal data protection law. **Revista de Direito Brasileira**, 2019.

G1. **Entenda o escândalo de uso político de dados que derrubou valor do Facebook e o colocou na mira de autoridades**. 2018. Acessado: 15 de agosto de

2024. Disponível em: https://g1.globo.com/economia/tecnologia/noticia/entenda-o-escandalo-de-uso-politico-de-dados-que-derrubou-valor-do-facebook-e-o-colocou-na-mira-de-autoridades.ghtml.

HARDJONO, T.; PENTLAND, A. MIT Open Algorithms. In: **Trusted Data: A New Framework for Identity and Data Sharing**. The MIT Press, 2019. ISBN 9780262356053. Disponível em: https://doi.org/10.7551/mitpress/12439.003.0005.

HARDMAN, D. **The Three Models of Digital Identity Relationships**. 2018. https://medium.com/evernym/the-three-models-of-digital-identity-relationships-ca0727cb5186. Acessado em 10 de julho, 2024.

HARDT, D. **The OAuth 2.0 Authorization Framework**. RFC Editor, 2012. RFC 6749. (Request for Comments, 6749). Disponível em: https://www.rfc-editor.org/info/rfc6749.

HAUCK, F. **OpenID for Verifiable Credentials: Formal Security Analysis using the Web Infrastructure Model**. Dissertação (Mestrado) — Informatik, 2023.

IDENTITY MANAGEMENT INSTITUTE. **Digital Identity Wallet Benefits and Risks**. 2024. Acessado: 15 de julho de 2024. Disponível em: https://identitymanagementinstitute.org/digital-identity-wallet-benefits-and-risks/.

JUNIOR, T. A. F.; VASCONCELOS, R. O.; RIBEIRO, A. de R. L. Um estudo comparativo entre zero-knowledge proof (zkp) e ring signatures visando as implicações legais e regulatórias com a lei geral de proteção de dados (lgpd) e general data protection regulation (gdpr). **Revista Observatorio de la Economia Latinoamericana**, Curitiba, v. 22, n. 2, p. 01–24, 2024. ISSN 1696-8352.

KULIK, T. et al. A survey of practical formal methods for security. 2020.

LODDER, M. **The Sovrin Network and Zero Knowledge Proofs**. 2018. Acessado: 02 de agosto de 2024. Disponível em: https://sovrin.org/the-sovrin-network-and-zero-knowledge-proofs/.

PANDEY, A.; SAINI, J. R. An investigation of challenges to online federated identity management systems. **Int. J. Eng. Innov. Res**, v. 1, n. 2, p. 50–54, 2012.

PETKUS, M. Why and how zk-snark works: Definitive explanation. July 2019.

PODGORELEC, B.; ALBER, L.; ZEFFERER, T. What is a (digital) identity wallet? a systematic literature review. **Journal of Digital Identity Studies**, 2022.

SAKIMURA, N.; BRADLEY, J.; JONES, M. **OpenID Connect Dynamic Client Registration 1.0 incorporating errata set 2**. [S.l.], 2023. NAT.Consulting (was at NRI), Yubico (was at Ping Identity), Self-Issued Consulting (was at Microsoft).

SAKIMURA, N. et al. **OpenID Connect Core 1.0 incorporating errata set 2**. 2014. Disponível em: https://openid.net/specs/openid-connect-core-1_0.html.

SAKIMURA, N. et al. **OpenID Connect Discovery 1.0 incorporating errata set 2**. [S.l.], 2023. NAT.Consulting (was at NRI), Yubico (was at Ping Identity), Self-Issued Consulting (was at Microsoft), Illumila.

SCHARDONG, F. Self-sovereign identity: A systematic review, mapping and taxonomy. **Sensors 22(15)**, 2022.

SCHARDONG, F.; CUSTODIO, R. From self-sovereign identity to fiduciary identity: A journey towards greater user privacy and usability. In: **Proceedings of the 39th ACM/SIGAPP Symposium on Applied Computing**. New York, NY, USA: Association for Computing Machinery, 2024. (SAC '24), p. 687–694. ISBN 9798400702433. Disponível em: https://doi.org/10.1145/3605098.3636061.

SIDELL, E. P. **How Google uses your data**. 2020. Acessado: 18 de julho de 2024. Disponível em: https://www.avast.com/pt-br/c-how-google-uses-your-data.

SIRIWARDENA, P. **OpenID Connect Authentication Flows**. 2020. Acessado em 11 de Abril, 2024. Disponível em: https://medium.facilelogin.com/openid-connect-authentication flows-936f52380f38.

SPORNY, M. et al. **Verifiable Credentials Data Model v2.0**. [S.l.], 2024. Https://www.w3.org/TR/2024/CRD-vc-data-model-2.0-20240513/.

SPORNY, M. et al. **JSON-LD 1.1: A JSON-based Serialization for Linked Data**. World Wide Web Consortium (W3C), 2020. W3C Recommendation. Disponível em: https://www.w3.org/TR/2020/REC-json-ld11-20200716/.

WANGHAM, M. et al. Gerenciamento de identidades federadas. **Anais do X Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais**, 10 2010.

YASUDA, K.; JONES, M.; LODDERSTEDT, T. Self-issued openid provider v2 - draft 13. Acessado em 11 de Maio, 2024. 2023.

YASUDA, K.; LODDERSTEDT, T. Openid for verifiable credential issuance - draft 13. Acessado em 11 de Maio, 2024. 2023.

YASUDA, K.; LODDERSTEDT, T. Openid for verifiable presentations - draft 20. Acessado em 11 de Abril, 2024. 2023.

YASUDA, K. et al. **OpenID for Verifiable Credentials**. [S.l.], 2022.

APÊNDICE A – ANÁLISE FORMAL DE SEGURANÇA

A.1 SISTEMA DE CREDENCIAIS VERIFICÁVEIS NA WEB

Os seguintes algoritmos modelam a extensão com Negociações para o modelo OpenID com Credenciais Verificáveis. O modelo é baseado na análise formal do OAuth 2.0 e o OpenID Connect da dissertação Fett, Küsters e Schmitz (2014) e OpenID Connect para Credenciais Verificáveis de Hauck (2023).

A.1.1 Visão Geral

Esse protocolo é modelado em um sistema de credenciais verificáveis na web $\mathcal{VCWS}^n = (\mathcal{W}, \mathcal{S}, \text{script}, E^0)$. O sistema $\mathcal{W} = \text{Hon} \cup \text{Net}$ modela um processo de ataque na rede Net. A estrutura $\text{Hon} = \text{Issuers} \cup \text{Fiduciary} \cup \text{Verifiers} \cup \text{B}$ inclui um conjunto finito de entidades definidas como: emissores (Issuers), fiduciários (Fiduciary), verificadores (Verifiers) e navegadores (B). Os processos são descritos em mais detalhes nas seções seguintes. Servidores DNS não foram modelados, conforme indicado em (HAUCK, 2023). O conjunto de scripts $\mathcal{S}$ e o mapeamento script permaneceram inalterados e são equivalentes aos apresentados em (HAUCK, 2023, Seções A.1). Da mesma forma, o conjunto E^0 permaneceu inalterado e é equivalente à Definição 42 de (FETT; KÜSTERS; SCHMITZ, 2014). A seguir, apresentam-se as definições dos conjuntos que devem ser incorporadas ao modelo:

- IDO (Input Descriptor Objects) é um conjunto finito de descritores de objeto associado a cada atacante de rede em Net, a cada verificador em Verifiers e cada fiduciário em Fiduciary. Os Objetos de Descritores de Entrada especificam as informações exigidas por uma Definição de Apresentação apresentada em (DIF, 2024). Por exemplo, para um verificador são as informações que ele requer de um titular ou para um fiduciário as que ele deseja negociar com um provedor de serviço. Denotamos por inputDescriptorObject a atribuição de processos atômicos ao conjunto IDO. Por exemplo, para $o \in \text{inputDescriptorObject}(fid)$, sendo $fid \in$ Fiduciary, teríamos $\text{inputDescriptorObject}(fid)$ como um subconjunto de IDO com apenas objetos de fid e o sendo um único objeto que o fiduciário pode utilizar para construir sua definição de apresentação.
- Constraints consiste em uma coleção de regras que limitam ou definem as condições específicas que devem ser atendidas para cada processo no sistema. No caso do fiduciário, são as regras de consentimento; já no caso de um verificador, elas incluem as condições que a aplicação requer para autenticar o usuário durante uma sessão. Também denotamos por constraints a atribuição de processos atômicos ao conjunto de Constraints, semelhante ao inputDescriptorObject.

- FiduciaryDoms $= \{d \mid d \in \text{dom}(fid) \wedge fid \in \text{Fiduciary}\}$ é o conjunto de domínios usados pelos processos de fiduciários.

Além dos conjuntos, apresentamos a seguir as modificações realizadas nas definições de funções, bem como a adição de novas funções, conforme exigido pelo modelo:

- browserOfFiduciary : Fiduciary $\rightarrow$ B função que associa cada fiduciário a exatamente um navegador.

- Em uma execução ρ de um sistema de credenciais verificáveis na web $\mathcal{VCWS}^n$, dentro de uma relação de uma fiduciário fid, dizemos que validateRequest$(x) \equiv \top$ se e somente se existe uma configuração (S, E, N) que contém um estado com $x \in S(b).\texttt{authStarted}$ e $b =$ browserOfFiduciary(fid).

- inputDescriptor : Fiduciary $\cup$ Verifier $\times\, 2^{\text{Constraints}} \rightarrow 2^{\text{IDO}}$ é uma função responsável por mapear cada processo com suas respectivas restrições do sistema para um conjunto de descritores de entrada (Input Descriptors), que especificam as informações ou credenciais requeridas. Neste contexto, o domínio da função apresenta um elmento de Fiduciary ou Verifier que representam, respectivamente, os participantes fiduciários e verificadores envolvidos, enquanto $2^{\text{Constraints}}$ denota o conjunto das restrições aplicáveis. A função inputDescriptor associa essas entidades e suas restrições a um conjunto de descritores, que detalham os dados necessários para o cumprimento dos requisitos para a autenticação.

- followConstraints : Constraints $\times$ inputDescriptor $\rightarrow \{\bot, \top\}$ é uma função responsável por verificar se um conjunto de objetos descritores respeita um conjunto de restrições (constraints) associado a um processo.

A.1.2 Identidades e Segredos

Esta seção descreve o processo de inicialização para identidades, chaves e segredos no sistema de credenciais verificáveis na web $\mathcal{VCWS}^n$. As subseções a seguir foram baseadas da Seção A.1 de (HAUCK, 2023).

A.1.2.1 Identidades

As definições de identidades permaneceram inalteradas e são equivalentes às apresentadas na Seção A.1.1 de (HAUCK, 2023).

A.1.2.2 Chaves e Segredos

O conjunto N de nonces continua sendo particionado em conjuntos disjuntos, um conjunto infinito N, e conjuntos finitos K_{TLS}, K_{sign} e $Passwords$:

$$N = N \uplus K_{TLS} \uplus K_{sign} \uplus Passwords$$

O conjunto K_{sign}, no entanto, contém as chaves que serão utilizadas por emissores para assinarem credenciais e para as fiduciários assinarem apresentações. Seja signkey : Issuers $\cup$ Fiduciary $\to K_{sign}$ uma função injetora que atribui uma chave de assinatura (diferente) a cada emissor e fiduciário.

A.1.2.3 Senhas

As definições de senhas mantiveram-se inalteradas, sendo equivalentes às expostas na Seção A.1.3 de (HAUCK, 2023).

A.1.2.4 Navegadores (Web Browsers)

As definições de navegadores mantiveram-se inalteradas, sendo equivalentes às expostas na Seção A.1.4 de (HAUCK, 2023), exceto pela alteração a seguir no estado inicial de um navegador.

Definição A.1 (Estado inicial do Navegador no $\mathcal{VCWS}^n$)**.** O estado inicial de um processo de navegador web $b \in B$ é definido conforme a Definição 33 apresentada em (HAUCK, 2023), sujeitando-se, adicionalmente, à seguinte restrição: s_0^b.secrets contém uma entrada $\langle\langle d, S\rangle, \langle \text{Secrets}^{b,p}\rangle\rangle$ para cada $p \in$ Issuer $\cup \{fid \in$ Fiduciary $\mid b =$ browserOfFiduciary$(fid)\}$ e cada domínio $d \in$ dom(p)

A.1.2.5 Função auxiliar

Para simplificar a descrição dos algoritmos, utilizamos a seguinte função auxiliar.

Algoritmo A.1 – Relação de um verificador R^v: Função para construir definição de apresentação.

```
1 function BUILD_PRESENTATION_DEFINITION(a, constraints, id)
2     let inputDescriptor := inputDescriptor(a, constraints)
3     let presentationDefinition := ⟨id, inputDescriptor⟩
4 return presentationDefinition
```

A.1.3 Emissores

As definições de Emissores mantiveram-se inalteradas, sendo equivalentes às expostas na Seção A.2 de (HAUCK, 2023).

A.1.4 Fiduciário

Um fiduciário $fid \in$ Fiduciary é modelado como um servidor web representado por um processo atômico DY $(I^{fid}, Z^{fid}, R^{fid}, s_0^{fid})$, com endereço $I^{fid} :=$ addr(fid). A definição a seguir especifica os estados Z^{fid} de fid, bem como o estado inicial s_0^{fid} de fid. A Tabela 3 mostra a lista de placeholders usados pelo Fiduciários.

Placeholder	Uso
ν_1	Novo ID da Definição de Apresentação
ν_2	Novo ID de Sessão
ν_3	Novo HTTP nonce

Tabela 3 – Lista de placeholders usados pelo Fiduciários

Definição A.2. Um estado $s \in Z^{fid}$ de um fiduciário fid é um termo da forma

$$\left\langle \begin{array}{l} pendingDNS,\ pendingRequests,\ DNSaddress,\ keyMapping,\ tlskeys, \\ corrupt,\ credentials,\ holderKey,\ userPins,\ sessions,\ transactionIds \end{array} \right\rangle$$

com $pendingDNS$, $pendingRequests$, $DNSaddress$, $keyMapping$, $tlskeys$, $corrupt$ como definido na Definição 43 em (FETT; KÜSTERS; SCHMITZ, 2014) e $credentials$, $holderKey$, $userPins$, $sessions$, e $transactionIds$ definido na seção A.3.1 em (HAUCK, 2023).

Um estado inicial s_0^{fid} de fid é um estado de fid com $s_0^{fid}.\texttt{pendingDNS} = \langle\rangle$, $s_0^{fid}.\texttt{pendingRequests} = \langle\rangle$, $s_0^{fid}.\texttt{DNSaddress} \in IPs$, $s_0^{fid}.\texttt{keyMapping} = \langle\{\langle d, \mathsf{pub}(\mathsf{tlskey}(d))\rangle \mid d \in \mathsf{Doms}\}\rangle$, $s_0^{fid}.\texttt{tlskeys} = tlskeys^{fid}$, $s_0^{fid}.\texttt{corrupt} = \bot$, $s_0^{fid}.\texttt{credentials} = \langle\rangle$, $s_0^{fid}.\texttt{holderKey} = \mathsf{signkey}(fid)$, $s_0^{fid}.\texttt{userPins} = \langle\{\langle i.\texttt{domain}, \mathsf{userPinOfId}(i)\rangle \mid \forall i \in s_0^b.\texttt{ids}$ com $b \in browserOfFiduciary(fid)\}\rangle$, $s_0^{fid}.\texttt{sessions} = \langle\rangle$, e $s_0^{fid}.\texttt{transactionIds} = \langle\rangle$.

A estrutura relacional de um fiduciário fid continua fundamentando-se no servidor HTTPS genérico, conforme delineado em (FETT; KÜSTERS; SCHMITZ, 2014). O Algoritmo A.2 subsequente modifica e expande as funcionalidade do referido servidor HTTPS com suporte para credenciais verificáveis proposto em (HAUCK, 2023) para suportar também o esquema de Negociação de Atributos. Métodos não modificados mantêm as definições originais estabelecidas pela Carteira.

Algoritmo A.2 – Modificação na relação de um Fiduciário R^{fid}: Processamento de Requisições HTTPS

```
70  if responseMode ≡ fragment then
71      let authRespUri := m.body[redirect_uri]
72      let previousPd := m.body[presentation_definition]
73      let clientMetadata := m.body[client_metadata]
74      let negotiationEndpoint := clientMetadata[negotiation_endpoint]
75      if negotiationEndpoint ≢ ⟨⟩ then
76          let type := {attribute, env}
```

```
77          if type ≡ attribute ∧ followConstraints(constraints(f), previousPd) ≡ ⊥ then
78              // Save information to respond to the Authorization request later.
79              let nonceForVp := m.body[nonce]
80              let nonceForHttpResp := m.nonce
81              let aud := authRespUri.host
82              let host := m.host
83              let state := m.body[state]
84              let authReqParameters := ⟨⟨nonceForVp, nonceForVp⟩⟩
85              let authReqParameters := authReqParameters +⟨⟩ ⟨nonceForHttpResp, nonceForHttpResp⟩
86              let authReqParameters := authReqParameters +⟨⟩ ⟨redirect_uri, authRespUri⟩
87              let authReqParameters := authReqParameters +⟨⟩ ⟨aud, aud⟩
88              let authReqParameters := authReqParameters +⟨⟩ ⟨host, host⟩
89              let authReqParameters := authReqParameters +⟨⟩ ⟨state, state⟩
90              let sessionID := ν2
91              let s'.sessions := ⟨⟨sessionID, authReqParameters⟩⟩
92              // Create Negotiation Request
93              let previousId := previousPd[id]
94              let idFiduciaryPd := ν1
95              let newPd := BUILD_PRESENTATION_DEFINITION(a, constraints(f), idFiduciaryPd)
96              let negotiationEndpoint := client_metadata[negotiation_endpoint]
97              let parameters := ⟨type, attribute⟩
98              let parameters := parameters +⟨⟩ ⟨previous_id, previousId⟩
99              let parameters := parameters +⟨⟩ ⟨presentation_definition, newPd⟩
100             let message :=
101             ⟨HTTPReq, ν3, POST, negotiationEndpoint.host, negotiationEndpoint.path, parameters, ⟨⟩, ⟨⟩⟩
102             call HTTPS_SIMPLE_SEND([responseTo:NEGOTIATE, session:sessionID], message, s' )
103     else
```

A.1.5 Verificador

A estrutura relacional de um Verificador v continua fundamentando-se no servidor HTTPS genérico, conforme delineado em (FETT; KÜSTERS; SCHMITZ, 2014). Os algoritmos A.3 e A.4 subsequentes modificam e expandem as funcionalidade do referido servidor HTTPS com suporte para credenciais verificáveis proposto em (HAUCK, 2023) para suportar também o esquema de Negociação de Atributos. Métodos não modificados mantêm as definições originais estabelecidas. A Tabela 4 lista de placeholders usados pelo verificador.

Placeholder	Uso
ν_1	Novo ID da Definição de Apresentação

Tabela 4 – Lista de placeholders usados pelo Verificador

Algoritmo A.3 – Modificação 1 na relação de um verificador R^v: Processamento de Requisições HTTPS

```
21  let domain ← FiduciaryDoms
22  let authUrl := ⟨URL, S, domain, /vp/authentication, ⟨⟩, ⊥⟩
23  let negotiationEndpoint ← {⟨URL, S, d, /negotiate, ⟨⟩, ⊤⟩ | d ∈ dom(v)}
24  let clientMetadata := clientMetadata +⟨⟩ ⟨negotiation_endpoint, negotiationEndpoint⟩
25  let idVerifierPd := ν1
26  let presentationDefinition := BUILD_PRESENTATION_DEFINITION(a, constraints(f), idVerifierPd)
27  let parameters := parameters +⟨⟩ ⟨client_metadata, clientMetadata⟩
28  let parameters := parameters +⟨⟩ ⟨presentation_definition_id, presentationDefinition[id]⟩
29  let session := session +⟨⟩ ⟨presentation_definition, presentationDefinition⟩
```

Algoritmo A.4 – Modificação 2 na relação de um verificador R^v: Processamento de Requisições HTTPS

```
101 else if m.path ≡ /negotiate ∧ m.method ≡ POST then
102   if m.body[type] ≡ attribute then
103     if m.body[previous_id] ≡ s'.session[presentation_definition][id] then
104       let validatePresentationDefinition ← {⊤,⊥}
105       if validatePresentationDefinition ≡ ⊤ then
106         let session := session + ⟨⟩⟨presentation_definition, m.body[presentation_definition]⟩
107         let body := ⟨status, ACCEPTED⟩
108         let m' := enc_s(⟨HTTPResp, m.nonce, 201, ⟨⟩, body⟩, k)
109         stop ⟨⟨f, a, m'⟩⟩, s'
110       else
111         let body := ⟨status, REJECTED⟩
112         let error ← {negotiation_request_denied, invalid_negotiation_request,
113         unsupported_definition, expired_defintion_id}
114         let body := body +⟨⟩⟨error, error⟩
115         let m' := enc_s(⟨HTTPResp, m.nonce, 400, ⟨⟩, body⟩, k)
116         stop ⟨⟨f, a, m'⟩⟩, s'
```

A.2 PROPRIEDADES FORMAIS DE SEGURANÇA

Autenticação da Negociação

De forma informal, a propriedade de segurança de Autenticação da Negociação implica que um atacante não pode realizar negociações em nome de um fiduciário honesto em um verificador honesto, desde que determinadas partes envolvidas nos processos de autenticação permaneçam íntegras. A Definição A.3 se aproxima da propriedade de autenticação apresentada em B.1.2 de (HAUCK, 2023), embora apresente algumas diferenças sutis, mas relevantes para esta análise.

Definição A.3 (O atacante não negocia Definições de Apresentações em nome de Fiduciário)**.** Seja $\mathcal{VCWS}^n$ um sistema web de Credenciais Verificáveis com um atacante de rede. O atacante não negocia Definições de Apresentações se, e somente se, para cada execução ρ de $\mathcal{VCWS}^n$, cada configuração (S_j, E_j, N_j) em ρ, cada $u \in$ ID e o navegador b que possui u não esteja totalmente corrompido em S_j (ou seja, o valor de `isCorrupted` não seja `FULLCORRUPT`), e $fid \in$ Fiduciary sendo um fiduciário honesto em S_j, não existe $l \leq j$, (S_l, E_l, N_l) sendo um estado em ρ tal que $pd \equiv \langle id, input\_descriptor \rangle$ e $pd \in d_\varnothing(S^l(attacker))$

A.3 PROVA DE PROPRIEDADES DE SEGURANÇA

Prova da Autenticação da Negociação

Esta seção apresenta a prova formal da propriedade de segurança de Autenticação da Negociação da Definição A.3. A propriedade é demonstrada direta-

mente, provando que ela se mantém em um sistema de credenciais verificáveis na web, $\mathcal{VCWS}^n$.

Lema A.1 (Negociação da Autenticação é válida)**.** Definção A.3 é válida.

Prova De acordo com a Definição A.3, existe uma definição de apresentação arbitrária pd no formato $\langle id, input\_descriptor \rangle$. Para mostrar que pd não é derivável pelo atacante ($pd \notin d_{\emptyset}(S_j(atacante))$), rastreamos todas as maneiras pelas quais um agente malicioso poderia controlar a pd.

Este parágrafo demonstra que tanto o navegador quanto o verificador não vazam pd em sua comunicação. O verificador honesto v envia uma requisição HTTPS criptografada contendo uma presentation_definition no formato $\langle id, input\_descriptor \rangle$ (Linha 25 do Algoritmo A.3) para o Fiduciário honesto fid através do navegador b. Esse mecanismo impede que o atacante derive o valor de pd, já que o algoritmo de criptografia não pode ser violado, as chaves de criptografia são mantidas em sigilo, e o mapeamento do domínio para a chave pública é devidamente configurado no estado inicial de cada processo. Uma prova mais aprofundada da segurança das mensagens HTTPS trocadas entre um navegador e um servidor HTTPS genérico, sem vazamento de informações, é apresentada no Lema 1 em (FETT; KUSTERS; SCHMITZ, 2014), assegurando que a resposta seja visível apenas para v e b, sem exposição a outras partes. Além disso, pela definição do navegador, podemos ver que apenas scripts carregados das origens $\langle d, \mathrm{S} \rangle$ podem acessar pd. Portanto, temos que apenas os scripts script_verifier_get_fragment e script client_verifier_index podem acessar pd (se carregados de suas respectivas origens) e que o navegador não utiliza ou vaza pd de nenhuma outra forma. Nenhum desses scripts utiliza pd, então o navegador não vaza esta definição por eles.

Este parágrafo demonstra que tanto o navegador quanto o fiduciário não vazam pd em sua comunicação. Da mesma forma mostrada acima, ao estabelecer uma conexão HTTPS entre b e fid, o mesmo lema 1 em (FETT; KUSTERS; SCHMITZ, 2014) garante que a resposta permanece protegida contra vazamentos para qualquer outra parte. O fid realiza uma validação da definição com `followConstraints(` `constraints(`fid`)`, pd) se $input\_descriptor$ de pd está de acordo com `constraints(`fid`)` (Linha 78 do Algoritmo A.2). Caso a condição não seja satisfeita, o sistema descarta pd procede construindo uma requisição com uma nova definição de apresentação pd' no mesmo formato $\langle id', input\_descriptor' \rangle$ juntamente com id de pd (Linha 80 do Algoritmo A.2). Como id e pd não vazam na etapa anterior, então somente fid consegue sugerir uma nova definição por meio de uma requisição POST criptografada para o domínio que foi especificado em negotiationEndpoint (Linha 23 Algoritmo A.3).

Uma vez que essa requisição chega ao domínio de destino, a definição de apresentação é submetida a um processo de validação por v com `followConstraints(` `constraints(`v`)`, pd'), garantindo que ela atende a todos os requisitos e critérios es-

tabelecidos para ser considerada válida em `constraints(`v`)`. Se a entidade verificadora v estiver de acordo com os termos da definição de apresentação, ela então armazena essa definição em seu estado interno (Linha 106 do Algoritmo A.4) para processamento da autenticidade do vp_token no futuro. Portanto, pd não é exposto por v, b ou fid ao atacante ou a qualquer outra parte, garantindo a proteção das informações. Dessa forma, o lema está comprovado. □