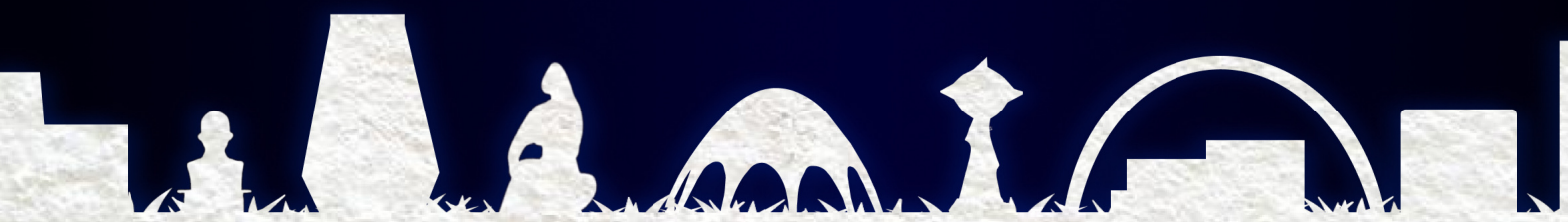
12 a 15
de Setembro

UFSM -
Santa Maria/RS

Minicursos SBSEG 2022

XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais



Por Carlos Raniery Paula dos Santos, Walter Priesnitz Filho,
Paulo André da Silva Gonçalves e Marcia Henke

XXII SBSeg 2022

Santa Maria - RS

Minicursos do XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais

Editora

Sociedade Brasileira de Computação (SBC)

Organização

Carlos Raniery Paula dos Santos, UFSM
Walter Priesnitz Filho, UFSM
Paulo André da Silva Gonçalves, UFPE
Marcia Henke, UFSM

Realização

Sociedade Brasileira de Computação (SBC)
Universidade Federal de Santa Maria (UFSM)

Promoção

Sociedade Brasileira de Computação (SBC)

12 a 15 de setembro de 2022,
Santa Maria - Rio Grande do Sul

Capa (Edição): Ana Paula Militz Dorneles (UFSM)
Produção Editorial: Vinicius Fulber Garcia (UFPR), Carlos Raniery Paula dos Santos (UFSM)

Cópias Adicionais:
Sociedade Brasileira de Computação (SBC)
Av. Bento Gonçalves, 9500- Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia - CEP 91.509-900 - Porto Alegre - RS
Fone: (51) 3308-6835
E-mail: sbc@sbc.org.br

Dados Internacionais de Catalogação na Publicação (CIP)

S612 Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (22. : 12 – 15 set. 2022 : Santa Maria)

XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais [recurso eletrônico] : Minicursos do SBSeg 2022 / organização: Carlos Raniery Paula dos Santos ... [et al.]. Dados eletrônicos. – Porto Alegre: Sociedade Brasileira de Computação, 2022.

189 p. : il. : PDF ; 18MB

Modo de acesso: World Wide Web.
Inclui bibliografia
ISBN 978-85-7669-510-3 (e-book)

1. Computação – Brasil – Simpósio. 2. Segurança da informação. 3. Sistemas computacionais. I. Santos, Carlos Raniery Paula dos. II. Priesnitz Filho, Walter. III. Gonçalves, Paulo André da Silva. IV. Henke, Marcia. V. Sociedade Brasileira de Computação. VI. Universidade Federal de Santa Maria. VII. Título.

CDU 004.56(063)

Ficha catalográfica elaborada por Jéssica Paola Macedo Müller – CRB-10/2662

Biblioteca Digital da SBC – SBC OpenLib

Índices para catálogo sistemático:

1. Ciência e tecnologia dos computadores : Informática : Segurança da informação – Publicação de conferências, congressos e simpósios etc. ... 004.56(063)

Contato
Av. Bento Gonçalves, 9500
Setor 4 - Prédio 43.412 - Sala 219
Bairro Agronomia
91.509-900 – Porto Alegre RS
CNPJ: 29.532.264/0001-78
http://www.sbc.org.br

Comissão Especial de Segurança da Informação e de Sistemas Computacionais – CESeg

Coordenação
Michele Nogueira (UFMG), Coordenadora
Igor Moraes (UFF), Vice-Coordenador

Comitê Gestor
Aldri Luiz dos Santos (UFMG) - Convidado
Altair Olivo Santin (PUCPR)
Daniel Macêdo Batista - (USP)
Eduardo Luzeiro Feitosa - (UFAM) - Convidado
Fábio Borges de Oliveira (LNCC)
Luciano Paschoal Gaspary - (UFRGS)
Luis Antonio Brasil Kowada (UFF)
Marco Aurelio Amaral Henriques (UNICAMP)
Roberto Samarone dos Santos Araujo (UFPA)

Mensagem dos Coordenadores Gerais

É com imensa alegria e satisfação que recebemos todos no XXII Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais (SBSeg 2022), realizado pela primeira vez em formato híbrido, entre os dias 12 e 15 de Setembro de 2022. Esta edição do SBSeg é especial, pois após dois anos afastados, temos novamente a possibilidade de nos encontrar presencialmente e contribuir com a consolidação do SBSeg como o mais importante evento nacional na área de segurança.

A organização de um evento como o SBSeg nunca é um ato individual, mas sim um ato de dedicação à comunidade realizado por diversas pessoas. Assim, gostaríamos de agradecer a todos os envolvidos, pelo apoio, dedicação e comprometimento durante a organização do evento. Primeiro, aos patrocinadores que confiaram no SBSeg 2022, pelo incentivo à pesquisa, desenvolvimento e inovação, tão importantes nos dias de hoje: o Comitê Gestor da Internet no Brasil (CGI.br) e o Núcleo de Informação e Coordenação do Ponto BR (NIC.br), a Coordenação de Aperfeiçoamento de Pessoal de Nível Superior (CAPES), o Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), a XLabs Security, a Tempest, a Ávato, a Ripple e a Faculdade IBPTech.

Nosso especial agradecimento à Universidade Federal de Santa Maria (UFSM), ao Centro de Tecnologia, e ao Colégio Técnico Industrial de Santa Maria pelo indispensável suporte à realização do evento. Nosso agradecimento também para os competentes e incansáveis coordenadores do comitê de programa, Alberto Egon Schaeffer-Filho (UFRGS) e Daniel Macêdo Batista (IME-USP). Aos coordenadores do Capture the Flag Michelle Wangham (Univali) e Weverton Cordeiro (UFRGS). Aos coordenadores de minicursos, Marcia Henke (UFSM) e Paulo André da Silva Gonçalves (UFPE). Aos coordenadores de Palestras e Turoriais, Lucas Cordeiro (University of Manchester) e Tiago Rizzetti (UFSM). Aos coordenadores do Salão de Ferramentas, Diego Kreutz (UNIPAMPA) e Raul Ceretta (UFSM). À coordenadora do Workshop de Trabalhos de Iniciação Científica e de Graduação, Rossana Maria de Castro Andrade (UFC). Ao coordenador do Concurso de Teses e Dissertações em Segurança da Informação e Sistemas Computacionais, Altair Santin (PUCPR). Aos coordenadores do Workshop de Gestão de Identidades Digitais, Emerson Ribeiro de Mello (IFSC) e Jean Martina (UFSC). Aos coordenadores do Workshop de Forense Computacional, Giuliano Giova (IBPTech) e Marcos Antônio Simplicio Junior (USP). Aos coordenadores do Fórum de Segurança Corporativa, Roberto Alves Gallo Filho (KRYPTUS Segurança) e Rafael Timóteo (UNB). Finalmente, o apoio da Sociedade Brasileira de Computação (SBC) e da Coordenação e do Comitê Gestor da Comissão Especial de Segurança da Informação (CESeg) da SBC foram determinantes para o sucesso do evento.

Todos os envolvidos contribuíram incansavelmente para fornecer uma programação rica e diversificada, discutindo temas relevantes no cenário nacional e internacional. A contribuição da comunidade científica brasileira foi de fundamental importância para manter a qualidade técnica dos trabalhos e fortalecer a ciência, a tecnologia e a inovação no Brasil. Após um cuidadoso processo de avaliação, foram selecionados 27 artigos completos e 4 artigos curtos (organizados em sessões técnicas), e 13 ferramentas para apresentação durante o Salão de Ferramentas. Além disso, o evento contou com 4

palestras principais proferidas por pesquisadores internacionalmente renomados, 1 Concurso de Teses e Dissertações, 4 minicursos, 2 tutoriais, bem como 3 Workshops, 1 competição do tipo Capture the Flag e 1 Fórum de Segurança Corporativa; este último, estabelecendo um diálogo entre a comunidade acadêmica, o setor produtivo e os órgãos do Governo encarregados da execução das políticas de segurança dos sistemas e redes governamentais.

Agradecemos também o excelente trabalho da equipe de organização local. Por fim, desejamos a todos um produtivo SBSeg.

Carlos Raniery P. dos Santos e Walter Priesnitz Filho
Coordenadores Gerais do SBSeg 2022

Mensagem dos Coordenadores de Minicursos

É com grande alegria e satisfação que apresentamos a seleção de Minicursos para o XXII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais (SBSeg) sediado em Santa Maria - RS de 12 a 15 de setembro de 2022. Recebemos 14 propostas de minicursos e 4 foram selecionadas para publicação e apresentação, representando uma taxa de aceitação de cerca de 28%. O Comitê de Avaliação foi composto por 10 pesquisadores para a elaboração dos pareceres. Cada proposta recebeu 3 pareceres, gerando ao todo 42 revisões.

Cada minicurso selecionado corresponde a um capítulo de 40 a 50 páginas deste livro com conteúdo ministrado, por especialistas, de forma presencial e online durante o evento. O Capítulo 1 apresenta e discute aspectos conceituais de gestão da identidade e de acesso sob a luz de novas necessidades e desafios provenientes de evoluções tecnológicas, demandas sociais e regulatórias. O Capítulo 2, contextualizado na área de segurança em Cidades Inteligentes, apresenta conceitos, princípios, aplicações, desafios e tendências sobre mobilidade elétrica e tarifação inteligente com NFTs (*Non-Fungible Tokens*). Em sua parte prática capacita audiência com um *hands-on* sobre emissão de NFTs. O Capítulo 3 apresenta conceitos, técnicas, ferramentas e casos práticos voltados para a análise de artefatos maliciosos no ambiente Windows. Faz uso intensivo da abordagem *hands-on* a fim de promover a capacitação e o aperfeiçoamento técnico da audiência. O Capítulo 4 foca na área de segurança de Redes Definidas por Software. Mostra e discute como o conceito de programabilidade do plano de dados pode ser explorado para a melhoria da segurança das redes. Apresenta desafios de segurança associados e estudos de caso.

Os textos produzidos para os Minicursos são tradicionalmente reconhecidos por sua qualidade, tornando-os referências para trabalhos acadêmicos, formação e treinamento complementares de estudantes, pesquisadores e profissionais. Além disso, os Minicursos demonstram sua valiosa contribuição para todos aqueles que visam adquirir ou atualizar conhecimentos teóricos e práticos em tópicos atuais e relevantes em Segurança da Informação e de Sistemas Computacionais.

Aproveitamos o ensejo para agradecer aos membros do Comitê de Programa e aos revisores por terem contribuído de forma voluntária neste desafio. Agradecemos-lhes também pelas discussões, empenho e competência na realização dos processos de avaliação e de seleção dos minicursos. Expressamos também nossos agradecimentos aos coordenadores gerais do SBSeg 2022, Carlos Raniery P. dos Santos (UFSM) e Walter Priesnitz Filho (UFSM), pela disponibilidade, ajustes operacionais e orientações providas. Expressamos também os nossos agradecimentos a todos os autores que submeteram suas propostas de minicursos e que nos motivam a realizar anualmente este evento de interesse, visibilidade e sucesso crescentes.

Saudamos a todos os participantes dos Minicursos do XXII SBSeg com os votos de excelentes cursos. É um enorme prazer tê-los em Santa Maria ou online participando dos minicursos.

Paulo André da S. Gonçalves (UFPE) e Márcia Henke (UFSM)
Coordenadores dos Minicursos do SBSeg 2022

Comitê de Organização

Coordenadores Gerais
Carlos Raniery Paula dos Santos (UFSM)
Walter Priesnitz Filho (UFSM)

Coordenação de Minicursos
Paulo André da Silva Gonçalves (UFPE)
Marcia Henke (UFSM)

Comitê de Programa dos Minicursos
Aldri dos Santos (UFMG)
Altair Santin (PUCPR)
Diogo Mattos (UFF)
Eduardo Feitosa (UFAM)
Eduardo Souto (UFAM)
Eduardo Viegas (PUCPR)
Igor Moraes (UFF)
Luiz Fernando Rust da Costa Carmo (Inmetro)
Marcia Henke (UFSM)
Paulo André da Silva Gonçalves (UFPE)
Rafael de Sousa Junior (UnB)
Raul Ceretta Nunes (UFSM)
Routo Terada (USP)

Sumário

Capítulo 1

Autenticação e Autorização: antigas demandas, novos desafios e tecnologias emergentes

Emerson Ribeiro de Mello*, Shirlei Aparecida de Chaves*, Carlos Eduardo da Silva†, Michelle Silva Wangham‡§, Andrey Brito¶ e Marco Aurélio Amaral Henriques‖

Abstract

Identity and access management integrates policies, business processes, and technologies to enable authentication and authorization of subjects before and during an online transaction. Technological developments, and social and regulatory demands, such as personal data protection regulations, constantly pose challenges for identity management. This chapter begins with a characterization of identity and access management models, which includes the decentralized identity model. It then presents some technologies and standards to meet new demands and challenges regarding security, privacy, usability, and user empowerment. It also characterizes software identity, the use of authorization and access control in web applications, ending with an overview of the topics covered.

Resumo

A gestão de identidade e de acesso integra políticas, processos de negócios e tecnologias para permitir a autenticação e autorização de sujeitos antes e durante uma transação online. Evoluções tecnológicas, demandas sociais e regulatórias, como as leis de proteção de dados pessoais, impõem constantemente desafios para a gestão de identidade. Este capítulo inicia-se com uma caracterização dos modelos de gestão de identidade e de acesso, o que inclui o modelo de identidade descentralizada, para depois apresentar

*Instituto Federal de Santa Catarina. Email: mello@ifsc.edu.br, shirlei.chaves@ifsc.edu.br
†Sheffield Hallam University, UK. Email: C.DaSilva@shu.ac.uk
‡Universidade do Vale do Itajaí. Email: michelle.wangham@rnp.br
§Rede Nacional de Ensino e Pesquisa
¶Universidade Federal de Campina Grande. Email: andrey@computacao.ufcg.edu.br
‖Universidade Estadual de Campinas. Email: maah@unicamp.br

algumas tecnologias e padrões para atender novas demandas e desafios relacionados à segurança, privacidade, usabilidade e empoderamento dos usuários. Também se caracteriza a identidade de software, utilização de autorização e controle de acesso em aplicações web, finalizando com um apanhado geral sobre os temas tratados.

1.1. Introdução

De acordo com a meta 16.9 dos Objetivos de Desenvolvimento Sustentável das Nações Unidas, os países signatários da Agenda 2030 devem fornecer identidade legal para todos, incluindo identidade digital. Cada pessoa tem o direito de participar plenamente na sua sociedade e de ser reconhecida como uma pessoa perante a lei. No entanto, cerca de um bilhão de pessoas em todo o mundo não têm meios de provar sua identidade, o que é essencial para protegerem os seus direitos e permitir acesso a serviços (GROUP, 2018).

Uma identidade digital é uma representação única de uma entidade que seja suficiente para identificar esta entidade em uma transação *online* (GRASSI; GARCIA; FENTON, 2020). Uma entidade é qualquer coisa existente no mundo real (uma pessoa, máquina, aplicação, objeto físico, empresa), sendo que esta pode possuir múltiplas identidades. A prova de identidade estabelece que uma entidade é quem ela afirma ser em um processo de autenticação digital (GRASSI; GARCIA; FENTON, 2020).

Conforme apresentado pelo relatório do Fórum Econômico Mundial intitulado *A Blueprint for Digital Identity* (FORUM, 2016), uma das lacunas do cenário de identidade digital é confundir autenticação com identidade. Soluções de autenticação utilizam processos de coleta de atributos e identificação do usuário (*onboarding*) preexistentes, baseados em modelos de identidade digital.

Segundo Allan (2020), a gestão de identidade e de acesso (*Identity And Access Management* – IAM) consiste em um conjunto de processos e tecnologias que visa permitir o relacionamento e confiança entre pessoas, serviços ou coisas (como no contexto de *Internet of Things* – IoT). A IAM visa garantir a identidade de uma entidade (usuário, dispositivo, software), garantir a qualidade das informações de uma identidade (identificadores, credenciais e atributos) e prover procedimentos de autenticação, autorização e auditoria (ITU, 2009).

Diante da transformação digital acelerada, decorrente da pandemia de Covid-19, da redefinição dos perímetros de segurança da informação nas instituições, das necessidades de usuários e de empresas por segurança, proteção de dados pessoais e usabilidade, a área de IAM se mostra relevante e desperta interesse da academia, do governo e das empresas. Embora os processos para implementar a IAM muitas vezes sejam complexos, estes são extremamente necessários, em especial, para lidar com ataques cada vez mais sofisticados e para implementação do modelo de confiança zero, uma estratégia de segurança que vem sendo muito recomendada (ROSE et al., 2020). Este modelo orienta a "nunca confiar e sempre verificar", ou seja, desconfiar "por padrão" e confiar "por exceção".

Segundo o relatório anual sobre violação de dados da IBM (2022), 83% das organizações sofreram mais de uma violação, sendo que o custo médio de uma violação de dados foi de 4,35 milhões dólares em 2022. O uso de credenciais roubadas ou comprometidas continua sendo o principal vetor de ataque em 19% das violações, tendo um custo médio

de 4,50 milhões de dólares. Essas violações de credenciais tiveram o ciclo de vida mais longo — 243 dias para identificar a violação e outros 84 dias para conter a violação. De acordo com relatório, organizações que fizeram uso de soluções de IAM, bem como a adoção de autenticação multifator, conseguiram reduzir os custos com a violação de dados em 224 mil dólares, 187 mil dólares, respectivamente.

De acordo com a pesquisa "Pesquisa Global de Identidade e Fraude 2021[1]"da Serasa (2021), 55% dos consumidores disseram que a segurança é o aspecto mais importante de sua experiência online e 33% disseram estar preocupados com roubo de identidade. A pesquisa constatou um aumento do conforto e preferência que os consumidores têm por métodos de segurança físicos e baseados em comportamento - ou invisíveis. Os consumidores, com base em sua segurança percebida, classificaram os seguintes métodos como os três mais seguros para autenticação:

- 74% dos consumidores disseram biometria física, que inclui principalmente reconhecimento facial e impressões digitais em dispositivos móveis;
- 72% dos consumidores disseram senhas de uso único (*One-Time Password* – OTP) enviados para dispositivos móveis;
- 66% dos consumidores disseram análise comportamental, que aproveita os comportamentos observados passivamente em navegadores e dispositivos móveis, sem fricção.

Os modelos de gestão de identidade sofreram evoluções constantes para adequarem-se aos novos serviços, modelos de negócio e tecnologias bem como às novas restrições impostas por meio de leis, como o Regulamento Geral de Proteção de Dados da União Europeia (*General Data Protection Regulation* – GDPR) (UNION, 2016) e a Lei Geral de Proteção de Dados Pessoais (LGPD) (BRASIL, 2018) no Brasil. Segundo Allen (2016), tal evolução pode ser dividida em quatro estágios: identidade centralizada, identidade federada, identidade centrada no usuário e identidade digital descentralizada.

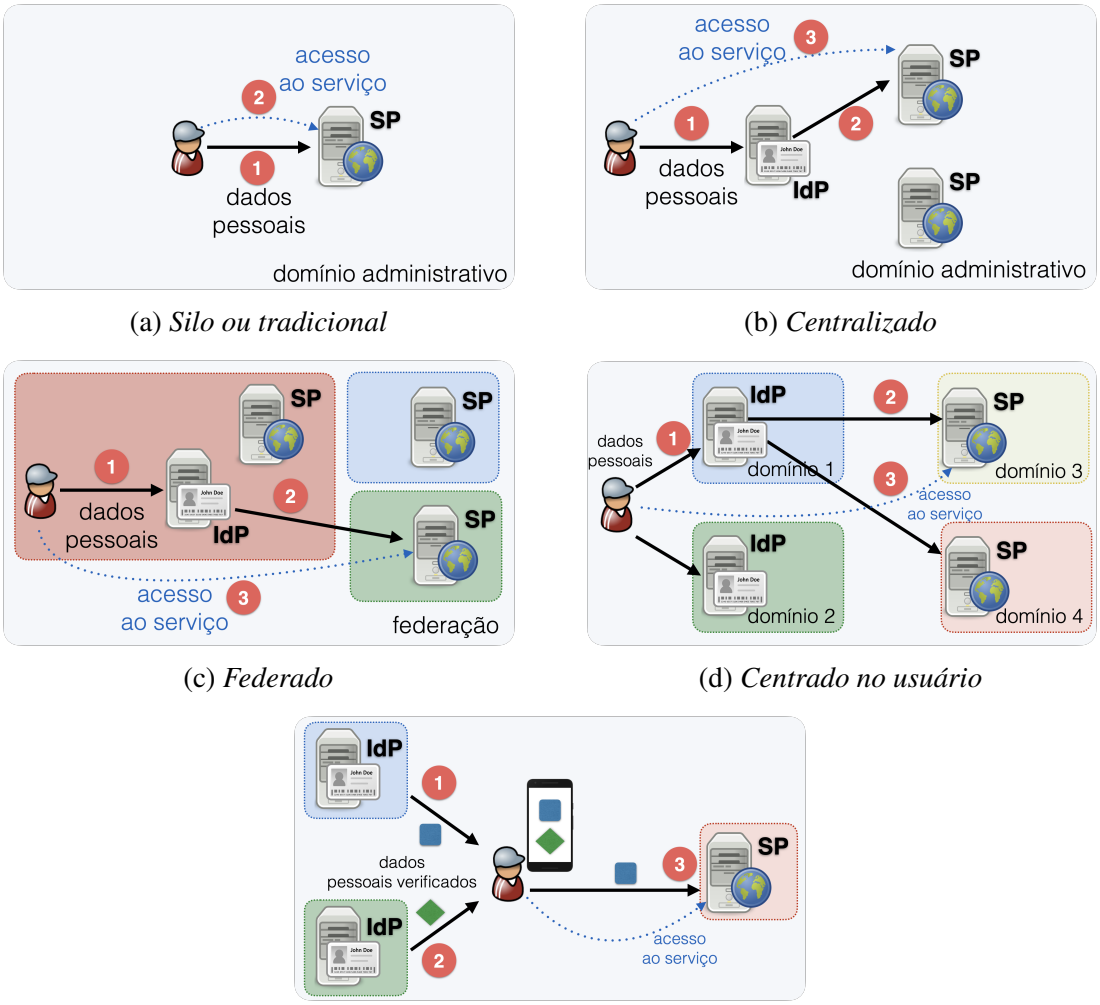
No minicurso apresentado no SBSeg de 2010 (WANGHAM; MELLO et al., 2010), discorreu-se sobre o gerenciamento de identidades federadas, apresentando os principais conceitos e tecnologias da época para implementação do modelo. Em 2013, o minicurso (WANGHAM; DOMENECH; MELLO, 2013) analisou infraestruturas de autenticação e autorização na IoT. No SBSeg 2019, o minicurso (NAKAMURA et al., 2019) foi específico para o modelo de identidade digital descentralizado, apresentando conceitos e implementação de alguns casos de uso. Por fim, em 2021, o minicurso de Falcão et al. (2021) abordou o tema de identidade de *software*.

O objetivo geral deste capítulo é apresentar as demandas históricas, novos desafios e tecnologias empregadas para autenticação, autorização e controle de acesso em sistemas distribuídos. Os principais problemas que permeiam a gestão de identidade e de acesso serão analisados, tais como: modelos de gestão de identidade, robustez do processo de autenticação, autenticação contínua e dinâmica, usabilidade e distribuição de responsabilidade, autenticação e autorização no cenário da Internet das coisas.

[1]A pesquisa teve como base 3 estudos produzidos entre junho de 2020 e janeiro de 2021. Foram entrevistados 9 mil consumidores e 2700 executivos de empresas de 10 países, incluindo o Brasil.

1.1.1. Modelos de gestão de identidade

Na literatura (WANGHAM; MELLO et al., 2010; ALLEN, 2016), os modelos são classificados como uma progressão de estágios. Apesar da literatura apresentar uma pequena divergência na nomenclatura para alguns destes modelos, Preukschat e Reed (2021) destacam que uma das principais diferenças entre os modelos é referente à forma como o usuário se relaciona com a organização ou serviço no qual ele estabelece a sua identidade digital. Esse relacionamento se refere ao quanto os dados da identidade do usuário estão efetivamente sob o seu controle e também ao quanto desses dados ele precisa compartilhar, direta ou indiretamente, para a utilização de um serviço. Os modelos de GId, normalmente, envolvem três atores, a saber: usuário que deseja acessar um recurso ou serviço; provedor de identidade (*Identity Provider* – IdP); e provedor de serviço (*Service Provider* – SP). O IdP é responsável pela autenticação e gerenciamento de informações do usuário. Os SPs, também conhecidos como terceiras partes confiáveis (*Relying Party* – RP), são entidades que fornecem serviços aos usuários e que delegam a autenticação destes aos IdPs.



(a) *Silo ou tradicional*

(b) *Centralizado*

(c) *Federado*

(d) *Centrado no usuário*

(e) *Descentralizado*

Figura 1.1: *Classificação dos modelos de gestão de identidade. Adaptado de (WANGHAM; MELLO et al., 2010)*

Na Figura 1.1 são representados os fluxos com dados pessoais do usuário (seta de linha contínua) e de acesso ao serviços (seta de linha pontilhada) nos diferentes modelos de gestão de identidade. O detalhamento sobre cada um destes modelos é apresentado logo a seguir.

O primeiro estágio consiste no modelo tradicional ou modelo baseado em silo (veja Figura 1.1a). Neste modelo, o provedor de serviço (*Service Provider* – SP) é o responsável por gerenciar seus usuários. Não existe o compartilhamento de identidades de usuários por diferentes serviços. Assim, o usuário precisará criar uma identidade digital específica para cada provedor de serviço com quem for interagir. Alguns autores também classificam este modelo como sendo centralizado.

O segundo estágio consiste no modelo centralizado (veja Figura 1.1b). O provedor de identidade (*Identity Provider* – IdP) é o único que possui controle administrativo sobre as identidades digitais dos usuários e compartilha os dados destes usuários com os demais provedores de serviços que obrigatoriamente estão dentro do mesmo domínio administrativo onde se encontra o IdP. Para o usuário tem-se a facilidade em não precisar agora criar uma identidade digital para cada SP com quem for interagir, contudo o IdP é quem de fato está controlando seus dados pessoais. O *Central Authentication Service* (CAS) *Protocol* (BRAMHALL et al., 2017) pode ser usado para implementar este modelo.

O modelo federado é considerado como o terceiro estágio (veja Figura 1.1c) e assemelha-se muito com o segundo estágio, contudo, o usuário poderá acessar também SPs que não fazem parte de sua instituição de origem, mas que fazem parte da federação na qual seu provedor de identidade também participa. O IdP continua controlando os dados do usuário e sempre participa da interação entre o usuário e o SP. O SAML (OPEN, 2022) pode ser usado para implementar este modelo, sendo este usado pela Comunidade Acadêmica Federada (CAFe) da RNP e por diversas outras federações acadêmicas no mundo.

Como quarto estágio tem-se o modelo de identidade centrada no usuário (veja Figura 1.1d), o qual apresenta esforços para que a experiência do usuário seja melhor e para que haja uma maior descentralização das informações e da confiança. Com este modelo começaram a surgir as ideias de que uma identidade digital deveria ficar totalmente sob o controle de seu dono. Entretanto, o foco maior está em duas frentes: consentimento do usuário, dando a este a visibilidade dos atributos que são compartilhados pelo IdP ao SP, e interoperabilidade, para facilitar a autenticação entre múltiplos provedores de serviços.

Neste estágio é possível dar ao usuário o controle total de sua identidade, mas ao custo de ele ter que criar seu próprio serviço de autenticação centrada no usuário, como a instalação de um serviço OpenID (OPENID, 2022), por exemplo. Como a maior parte dos usuários não têm condições de criar tal serviço, o caminho natural demonstrado na prática, é o uso de serviços deste tipo disponibilizados por grandes provedores já consolidados como Facebook, Google, Apple e a conta gov.br[2], por exemplo. O conceito *Bring you own identity* (BYOI), em tradução livre, "traga a sua própria identidade", foi cunhado como uma forma de uso propiciada por este modelo de gestão de identidade. Para implementar este modelo é comum o uso do OpenID Connect (OPENID, 2022), que apresenta uma

[2]https://www.gov.br/governodigital/pt-br/conta-gov-br/conta-gov-br/

camada de identidade sobre o protocolo OAuth2 (HARDT, 2012).

Dessa forma, os dados do usuário ainda continuam nas mãos dos provedores, que detêm o controle sobre os mesmos, e estes podem desabilitar um usuário a qualquer momento, mesmo sem apresentar justificativas para tal. Além disso, observa-se uma centralização ainda maior do processo de autenticação em alguns poucos e grandes provedores, o que nos remete a um dos problemas básicos dos primeiros estágios, que é a centralização dos dados no provedor de identidade, sem o controle do usuário.

O conceito de Identidade Digital Descentralizada (IDD), também chamada de Identidade Autossoberana, do inglês *Self-Sovereign Identity* (SSI), apresenta-se como o último estágio dos modelos de gestão de identidade (veja Figura 1.1e). Cabe destacar que o termo identidade autossoberana pode ser mal interpretado de forma a dar a entender que o indivíduo poderia emitir sua própria identidade. A sociedade está organizada em sistemas políticos que possuem estruturas governamentais com papéis bem definidos e cabem somente a estas a soberania para identificação de seus cidadãos. Desta forma, a identidade autossoberana propõe-se em dar ao usuário a soberania para administrar suas identidades digitais e não em criá-las (LÓPEZ, 2020).

Segundo (ALLEN, 2016), apesar do modelo centrado no usuário ter permitido que identidades centralizadas pudessem ser usadas como identidades federadas interoperáveis, e respeitando o consentimento do usuário, ainda era necessário um modelo no qual o usuário estivesse no centro do processo autenticação, cabendo somente a ele ditar as regras de uso sobre seus dados pessoais e que não houvesse qualquer intermediação do IdP no acesso ao SP. No modelo descentralizado, o próprio usuário é o responsável por manter suas identidades digitais, por exemplo em um aplicativo de carteira digital em seu telefone inteligente, cujos atributos são atestados criptograficamente por seus emissores (e.g. Secretaria de Segurança Pública) e poderão ter sua integridade e autenticidade verificada pelos provedores de serviço.

Os identificadores descentralizados (*Decentralized Identifiers* – DIDs) (W3C, 2022a) e as credenciais verificáveis (*Verifiable Credentials* – VC) (W3C, 2022b) são dois padrões da W3C que estão sendo considerados como pilares para soluções de gestão de identidade descentralizada. Ainda existem diversas tecnologias e *frameworks*, abertos e proprietários, sendo que muitos fazem uso de livro razão distribuído (e.g. *blockchain*) e o Hyperledger Indy[3] é um que tem despertado bastante interesse da comunidade.

Identificadores descentralizados

Segundo (W3C, 2022a), pessoas e organizações usam identificadores únicos globais, nos mais variados contextos. Por exemplo, pessoas usam endereço de email, nomes de usuários em redes sociais, número de telefone, CPF etc. Como identificadores para produtos ou serviços tem-se o número serial, URI (NOTTINGHAM, 2020), UUID (LEACH; MEALLING; SALZ, 2005) etc. Porém, os identificadores amplamente usados por pessoas, nas interações com serviços na Internet, não estão de fato sob seu controle, uma vez que são emitidos e controlados por autoridades externas. Por exemplo, o endereço de email de

[3]https://www.hyperledger.org/use/hyperledger-indy

uma pessoa pode ser excluído de uma organização assim que esta deixar de fazer parte do quadro de funcionários. Tais identificadores também podem relevar informações pessoais mais do que é necessário na interação com um serviço, ou ainda, podem ser replicados de forma fraudulenta, podendo assim resultar no roubo da identidade de uma pessoa em um serviço.

Os identificadores descentralizados (DIDs) (W3C, 2022a) consistem em identificadores únicos globais, que podem referenciar qualquer tipo de entidade (e.g. pessoa, dispositivo, organização, software etc.), e que possibilitam aos detentores (e.g. pessoas, organizações etc.) serem seus controladores. Desta forma, não existe aqui a dependência de uma autoridade externa ou mesmo centralizada para emissão, gestão ou manutenção de DID. O DID, portanto, consiste em um identificador que apresenta *simultaneamente* as quatro propriedades a seguir:

- **Descentralizado** – não há necessidade de uma autoridade central para emiti-lo;
- **Persistente** – não há necessidade em ter uma organização mantenedora para que continue a existir;
- **Resolvível** – pode ser usado no processo de resolução para recuperar metadados associados a este (chamados de documentos DID);
- **Criptograficamente verificável** – é possível comprovar criptograficamente seu controle e posse.

Um DID consiste de uma URI – *Uniform Resource Identifier* – que é uma cadeia de caracteres segmentada em três partes (veja Figura 1.2): 1) identificador do esquema (*did:*); 2) identificador do método DID; e 3) identificador único determinado pelo método DID. O método DID é definido em uma especificação própria, e externa à especificação do próprio DID, na qual são detalhadas como DID e documentos DID são criados, atualizados, resolvidos e desativados.

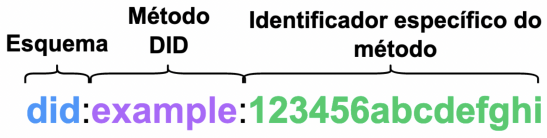


Figura 1.2: *Exemplo de um DID. Adaptado de W3C (2022a)*

O DID pode ser usado na resolução de metadados que estejam associados a este, sendo estes metadados chamados de documentos DID. Nestes documentos estão contidas informações de interesse do controlador, bem como métodos (tipicamente baseados em criptografia de chave pública) que permitem a verificação destas informações. A Figura 1.3 apresenta uma visão geral da arquitetura DID e o relacionamento entre seus componentes básicos. Um resumo de cada um dos componentes é apresentado a seguir, de acordo com W3C (2022a).

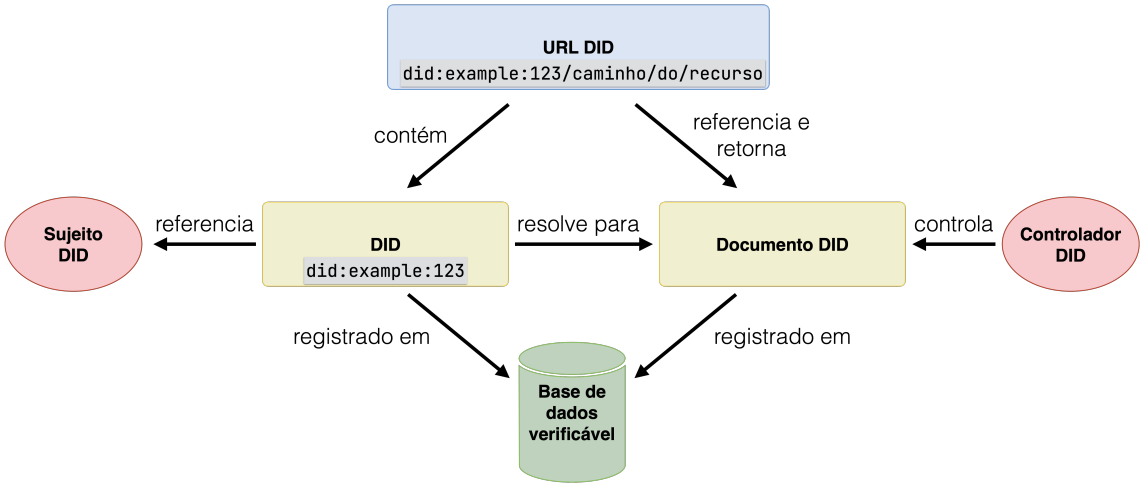


Figura 1.3: *Visão geral da arquitetura DID e relacionamentos entre seus componentes básicos. Adaptado de W3C (2022a)*

- **Sujeito DID** – é a entidade (pessoa, grupo, organização, coisa, conceito etc.) identificada pelo DID;
- **Documento DID** – contém informações associadas ao sujeito identificado pelo DID - como por exemplo, chaves públicas para verificações, serviços relevantes para interação com o sujeito DID etc.;
- **Controlador DID** – a entidade (pode ser o próprio sujeito, mas não necessariamente) com capacidade de fazer alterações no documento DID;
- **URL DID** – estende a sintaxe básica de um DID, incorporando outros componentes padrões de uma URI, como *path*, *query* e *fragment*;
- **Base de dados verificável (*Verifiable Data Registry*)** – solução subjacente de armazenamento que permite a criação, verificação, atualização e desativação de DID e documentos DID (e.g. livro razão distribuído, redes P2P, sistema de arquivos distribuídos etc.).

Credenciais verificáveis

As Credenciais Verificáveis (VCs) (W3C, 2022b) são o ícone mais visível da infraestrutura que se convencionou chamar de Identidade Autossoberana e os DIDs permitem criar identificadores que possuam os atributos desejados de um identificador global único. As VCs foram propostas para permitir expressar credenciais digitais de forma similar às credenciais físicas, como documentos de identidade, carteira de habilitação etc. (veja Figura 1.4), de modo que sejam criptograficamente seguras, respeitem a privacidade e que possam ser interpretadas por máquinas.

O modelo de dados das VCs prevê que estas podem ser representadas, bem como o material criptográfico associado, como documentos JSON-LD (W3C, 2020) ou na forma de *tokens* JWT (JONES; BRADLEY; SAKIMURA, 2020) (veja Listagem 1.1).

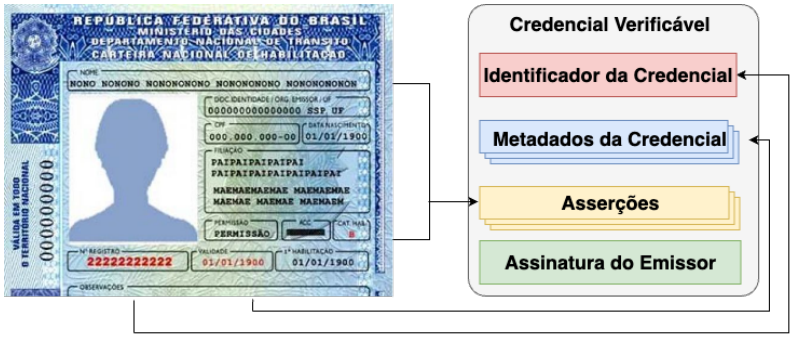


Figura 1.4: *Mapeamento de conceitos de Credenciais Verificáveis para o equivalente de credencial física. Adaptado de Preukschat e Reed (2021)*

```
{
    "@context": [
        "https://www.w3.org/2018/credentials/v1",
        "https://www.w3.org/2018/credentials/examples/v1"
    ],
    "id": "http://example.edu/credentials/1872",
    "type": [
        "VerifiableCredential",
        "AlumniCredential"
    ],
    "issuer": "https://example.edu/issuers/565049",
    "issuanceDate": "2010-01-01T19:23:24Z",
    "credentialSubject": {
        "id": "did:example:ebfeb1f712ebc6f1c276e12ec21",
        "alumniOf": {
            "id": "did:example:c276e12ec21ebfeb1f712ebc6f1",
            "name": [
                {
                    "value": "Example University",
                    "lang": "en"
                },
                {
                    "value": "Exemplo de Universidade",
                    "lang": "pt_BR"
                }
            ]
        }
    },
    "proof": {
        "type": "RsaSignature2018",
        "created": "2017-06-18T21:19:10Z",
        "proofPurpose": "assertionMethod",
        "verificationMethod": "https://example.edu/issuers/565049#key-1",
        "jws": "eyJhbGciOiJSUzI1NiIsImI2NCI6ZmFsc2UsImNyaXQiOlsiYjY0Il19..TCYt5X
    sITJX1CxPCT8yAV-TVkIEq_PbChOMqsLfRoPsnsgw5WEuts01mq-pQy7UJiN5mgRxD-WUc
    X16dUEMGlv50aqzpqh4Qktb3rk-BuQy72IFLOqV0G_zS245-kronKb78cPN25DGlcTwLtj
    PAYuNzVBAh4vGHSrQyHUdBBPM"
    }
}
```

Listagem 1.1: Exemplo de uma VC representada com JSON-LD. Fonte: (W3C, 2022b)

Quando comparado com o gerenciamento de identidades federadas, a arquitetura de VCs apresenta uma terminologia similar (veja Figura 1.5). Podemos relacionar no modelo de VCs o Verificador (*Verifier*) como sendo equivalente ao SP, o Emissor (*Issuer*) como sendo equivalente ao IdP e o usuário como sendo o Detentor (*holder*). Porém, as

semelhanças em termos de modelo de confiança e comunicação não podem ser analisadas como equivalentes, pois são fundamentalmente diferentes.

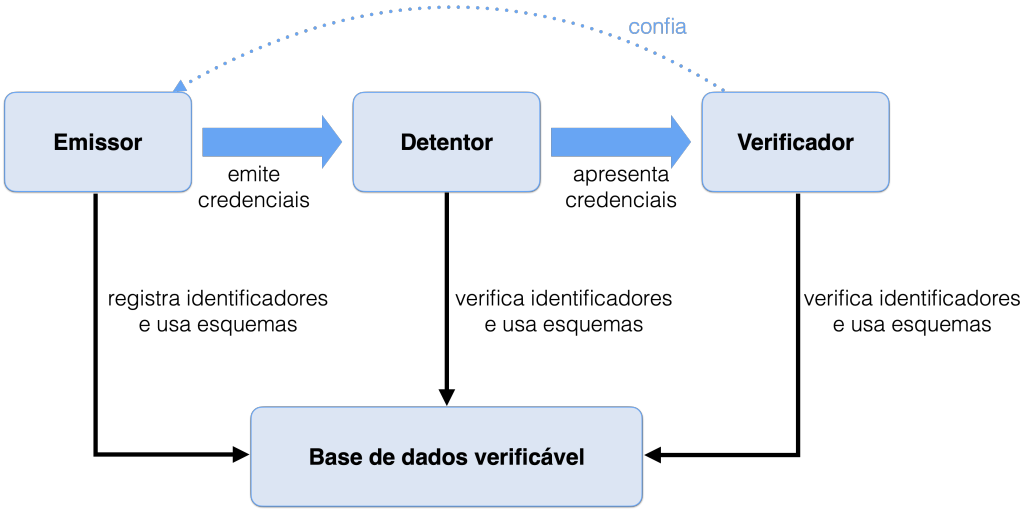


Figura 1.5: *Papéis e fluxos de comunicação com VCs. Adaptado de W3C (2022b)*

Na Figura 1.5 é possível notar que o usuário, detentor da credencial, está no centro da comunicação. Quando o Detentor apresenta sua credencial ao Verificador, a exemplo do que acontece no mundo físico quando se apresenta um documento a uma parte interessada, não há uma comunicação com o emissor. A recuperação da informação necessária para validar a credencial é feita diretamente pelo Verificador, acessando a base de dados verificável. Isto é, o Emissor não toma conhecimento para qual serviço ou em que situação o Detentor está apresentando sua credencial. Neste caso, tanto o Detentor, como o Emissor e o Verificador confiam na base de dados verificável.

O Verificador confia no Emissor, de acordo com suas próprias regras de confiança. Por exemplo, um conselho profissional pode confiar em uma universidade como autoridade para emissão de uma VC que ateste que uma pessoa obteve um título necessário para exercer determinada profissão. A universidade deve disponibilizar publicamente (por exemplo, divulgando seu DID) o material criptográfico necessário para que a VC emitida possa ser verificada. Cabe ao Verificador ir até à base de dados verificável para obter a informação necessária para verificar se a VC apresentada pelo Detentor está íntegra e é autêntica, ou seja, emitida por um Emissor no qual confia.

O Detentor é o responsável por manter suas VCs e pode, por exemplo, fazer uso de *softwares* como carteiras digitais instaladas em dispositivos que possui, como um telefone inteligente. Segundo Grüner et al. (2020), o modelo de gestão de identidade descentralizado possui desafios para que possam ser amplamente utilizados, sendo o armazenamento de credenciais pelo usuário um destes. No modelo descentralizado, os dados do usuário ficam armazenados sob sua responsabilidade, em sua carteira digital, ao contrário do modelo centralizado e federado, em que esses dados são armazenados pelo IdP em bases de dados próprias.

1.1.2. Autorização e controle de acesso

Quando se discute políticas e mecanismos para controle de acesso é comum utilizar uma abstração chamada de modelo de controle de acesso. Tal abstração permite descrever as características principais dos mecanismos de controle de acesso. Eles constituem um tipo de linguagem universal que permite que usuários, fornecedores e desenvolvedores interajam entre si, fornecendo um entendimento comum para o desenho e implementação de mecanismos de controle de acesso.

Existem diversos modelos de controle de acesso e, neste capítulo, iremos discutir os dois principais modelos que podem ser encontrados na maioria das ferramentas disponíveis. Antes de apresentar os modelos é necessário a definição de algumas terminologias que serão utilizadas.

- **Sujeito**: Também conhecido como principal, o sujeito é uma representação de uma entidade que deseja acessar um sistema. Normalmente é um usuário que deseja realizar alguma operação mas também pode ser um processo, outro sistema ou uma coisa;
- **Objeto**: Uma representação de um recurso que é acessado por um sujeito por meio de uma operação e protegido pelo mecanismo de controle de acesso;
- **Operação**: Representa uma ação realizada pelo sujeito no objeto;
- **Permissão** (privilégio): é a autorização para realizar uma ação em um sistema. É normalmente associada com o par operação-objeto.

A matriz de controle de acesso constitui a abstração mais básica quando se desenha e analisa políticas de controle de acesso. Ela pode ser interpretada como uma tabela onde cada linha representa um sujeito, e cada coluna representa um objeto no sistema protegido. Cada célula da tabela contém o conjunto de direitos de acesso que cada sujeito possui para o respectivo objeto.

Uma matriz de controle de acesso pode ser representada de várias maneiras, por exemplo, agrupando as permissões por linhas onde as mesmas são associadas a cada sujeito (também conhecido como lista de capacidades ou controle de acesso baseado em identidade), ou através do agrupamento por colunas, associadas aos objetos. Este último é também conhecido como listas de controle de acesso (*Access Control List - ACL*) e é a estratégia normalmente utilizada para o controle de permissões em sistemas de arquivos. Entretanto, a implementação de matrizes de controle de acesso através de listas de capacidades e ACLs são soluções não indicadas para sistemas *web* e distribuídos. Tais soluções não são adequadas para ambientes de larga escala, que possuem um grande número de usuários e objetos protegidos.

No modelo de controle de acesso baseado em papéis (*Role-Based Access Control - RBAC*) as decisões de acesso são baseadas na definição de papéis que os usuários assumem dentro de uma organização (FERRAIOLO; KUHN, 1996). Neste modelo existe uma ligação indireta entre o usuário e a permissão de acesso. Em uma política RBAC um conjunto de permissões é associado a um determinado papel. Por outro lado, usuários são

alocados a esses papéis, fazendo com que cada usuário tenha direito às permissões de seu papel. Deste modo, a administração das políticas de controle de acesso pode acontecer de maneira independente à administração dos usuários do sistema. Por exemplo, uma determinada permissão pode ser adicionada a um papel existente, autorizando seu uso por todos os usuários com esse papel, sem a necessidade de alterar os dados de todos os usuários afetados.

O modelo RBAC foi padronizado em 2004 pelo NIST (ANSI, 2004) e conta com um conjunto de extensões para prover suporte a diferentes casos de uso para políticas de controle de acesso. As extensões principais são o modelo hierárquico e as separações de responsabilidades estática e dinâmica.

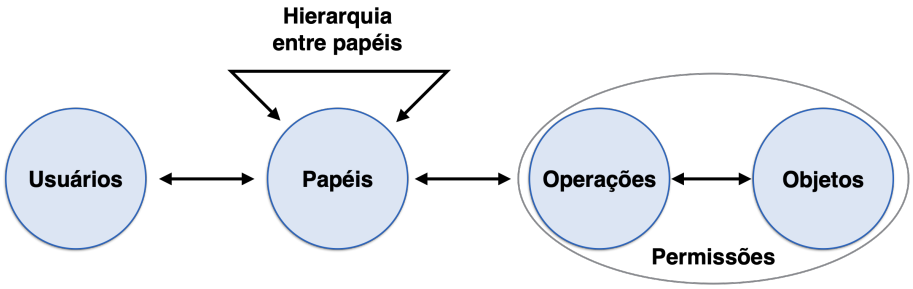


Figura 1.6: *Modelo RBAC hierárquico. Adaptado de (ANSI, 2004)*

O modelo RBAC hierárquico (apresentado na Figura 1.6) adiciona o conceito de hierarquias entre os papéis, de modo que um papel pode herdar as permissões associadas a outro papel enquanto adiciona outras permissões. Com isso, se consegue capturar mais facilmente a estrutura organizacional de uma instituição. Neste caso, existe responsabilidades e privilégios em comum entre dois ou mais papéis, mas um destes papéis possui outras permissões que não estão disponíveis para os outros. Por exemplo, um papel funcionário pode ter permissão para leitura de uma informação enquanto um papel diretor herda esta permissão e adiciona uma operação de escrita.

Outras extensões preveem a inclusão de restrições nas políticas de controle de acesso. Tais restrições são associadas ao conceito de separação de responsabilidades (*Separation of Duties - SOD*). No modelo RBAC um usuário pode estar associado a um ou mais papéis que são ativados durante uma sessão de uso do sistema (por exemplo, após a autenticação do usuário). Alguns sistemas permitem que usuário troquem seu papel ativo, enquanto que outras implementações fazem com que usuários tenham todos seus papéis ativos ao mesmo tempo. A restrição de SOD diz respeito a um conjunto de papéis que não podem ser ativados simultaneamente por um usuário.

Na SOD estática, a restrição acontece através de regras que são definidas junto à política de controle de acesso, por exemplo, quais papéis não podem ser alocados para um mesmo usuário simultaneamente, ou restrições baseadas no objeto sendo acessado. Por exemplo, o RBAC pode garantir que os usuários não possam ser membros da função de compra e da função de aprovação simultaneamente. A SOD estática garante que a mesma pessoa não possa comprar e aprovar a compra.

Na SOD dinâmica a restrição é normalmente associada aos papéis que podem estar

ativos ao mesmo tempo no sistema. Tal restrição pode ser associada a um usuário, que não pode ativar dois papéis diferentes em uma mesma sessão, ou à quantidade de usuários com um determinado papel que podem estar ativo simultaneamente. Neste caso, o sistema permite que a mesma pessoa esteja na função de compras e na função de aprovação, mas essas estariam proibidas de aprovar suas próprias compras. Elas só poderiam aprovar as compras de terceiros.

O modelo RBAC simplifica a definição de políticas de controle de acesso de acordo com as funções de um usuário em uma organização. Uma vez que uma política de controle de acesso foi definida, com seus respectivos papéis, regras, hierarquias e restrições, as tarefas relacionadas com a administração dos mecanismos de controle de acesso se resumem a adicionar ou remover usuários de determinados papéis.

Em 2012, o NIST publicou uma extensão ao modelo RBAC, denominado RBAC *Policy-Enhanced* (RPE) (ANSI, 2012), incorporando algumas funcionalidades que não constavam no modelo original mas que foram identificadas ao longo dos anos após sua publicação. A especificação RPE define e incorpora suporte a um conjunto de regras para os dois tipos de restrições: estática e dinâmica. Para restrições SOD estática, a especificação define regras que permitem restringir a alocação de diversos papéis a um mesmo usuário, diversas permissões a um mesmo papel, ou até mesmo quais usuários não podem ser alocados a determinados papéis. Para restrições SOD dinâmica, temos a adição de regras que consideram informações que são fornecidas por elementos externos ao do mecanismo de controle de acesso. Dentre as novas regras encontram suporte para restrições que influenciam quando um determinado papel pode ser ativado considerando, por exemplo, o horário do dia, a localização geográfica ou um determinado valor de um atributo.

Entretanto, o modelo RBAC apresenta algumas limitações, que acabaram motivando o desenvolvimento de outros modelos. Dentre essas limitações podemos mencionar a dificuldade associada com a tarefa conhecida como "engenharia de papéis", ou seja, a definição de todos os papéis e suas respectivas permissões em uma organização. Existe um conflito entre a facilidade de administração e a definição de políticas de segurança "mais fortes". Este último exige a definição de papéis e permissões mais granulares, o que resulta em uma maior quantidade de papéis que precisam ser alocados aos usuários, dificultando a administração das políticas de controle de acesso.

Uma evolução do modelo RBAC é o controle de acesso baseado em atributos (*Attribute-Based Access Control - ABAC*) (HU et al., 2014). Definido pelo NIST em 2014, no modelo ABAC as decisões de acesso são tomadas sobre um conjunto de regras baseados nos valores de atributos do sujeito que está requisitando acesso, do objeto sendo acessado, da operação sendo realizada e de condições ambientais tais como, hora do dia, dia da semana, localização geográfica ou qualquer outro atributo disponível no sistema. Neste caso, uma política de controle de acesso é composta por um conjunto de regras *booleanas* com os atributos e seus valores. O ABAC baseia-se nos modelos mencionados anteriormente, no sentido que cada um dos elementos considerados pelos modelos anteriores podem ser vistos como um atributo, mas os estendem ao permitir o uso de outros atributos. O modelo ABAC permite a implementação do modelo RBAC ao se utilizar somente o atributo papel nas regras de uma política de controle de acesso.

Não considerado estritamente como um modelo de controle de acesso, mas uma prática que vem ganhando adeptos, devido ao seu uso pelo Google, é o conceito de autorização de três fatores[4] (*Three-Factor Authorization - 3FA*) (ADKINS et al., 2020). A ideia geral é que algumas decisões de acesso exijam uma autorização explícita por um ser humano (que atua como autorizador) após sua avaliação pelos mecanismos de controle de acesso. Tal autorização deve ser realizada usando um sistema (ou dispositivo) diferente do que o utilizado para se realizar a operação. Por exemplo, uma prática adotada pela Google para algumas tarefas mais críticas é o uso de 3FA exigindo uma autorização explícita do usuário através de seu smartphone. A mesma estratégia é utilizada para usuários comuns se autenticando em sua conta Google, onde é enviada uma notificação, em um aplicativo no telefone inteligente do usuário o qual já esteja autenticado junto ao Google, questionando se o mesmo autoriza a realização da operação de *login* em um outro dispositivo.

Infraestrutura de autorização e controle de acesso

Com o crescente uso de regras de controle de acesso mais complexas, sua implementação em código torna-se mais arriscada. Isso tem contribuído para a adoção de infraestruturas de controle de acesso, nas quais as decisões de acesso são realizadas por um componente separado do código que implementa a lógica de negócio da aplicação (HU et al., 2014; ADKINS et al., 2020). Tais serviços podem ser encontrados, por exemplo, nas plataformas de nuvem dos grandes provedores e são conhecidos como *frameworks* de autorização. Por exemplo, as plataformas de nuvem da Google[5] e da Amazon[6] oferecem um serviço de *Identity & Access Management* (IAM).

Esses serviços implementam o conceito de mecanismo de autorização, que engloba um conjunto de funções lógicas bem definidas. Sua função principal é receber uma requisição de acesso do sujeito para executar uma determinada operação em um objeto, verificar se tal operação é permitida de acordo com as regras definidas na política de controle de acesso e retornar um resultado positivo ou negativo que é então seguido pelo sistema.

Essas funções são capturadas em uma arquitetura de referência (HU et al., 2014) composta por quatro componentes principais: *Policy Enforcement Point* (PEP), *Policy Decision Point* (PDP), *Policy Information Point* (PIP), e *Policy Administration Point* (PAP). Na Figura 1.7 é apresentada a arquitetura de referência com esses componentes que serão detalhados a seguir.

O PDP é responsável pela tomada de decisão sobre uma determinada requisição utilizando como base uma política de controle de acesso junto com os atributos disponíveis (do sujeito, do objeto, da operação e do ambiente). O PEP atua como um porteiro, interceptando as requisições de acesso de um sujeito que são então enviadas ao PDP, e cumprindo as decisões recebidas como respostas do PDP. O PIP é responsável por obter as informações que o PDP necessita para tomar suas decisões, podendo utilizar uma ou

[4]Não confundir com a autenticação de dois fatores (*Two-factor Authentication* – 2FA), que é comentado na Subseção 1.2.2.

[5]https://cloud.google.com/iam

[6]https://aws.amazon.com/iam/

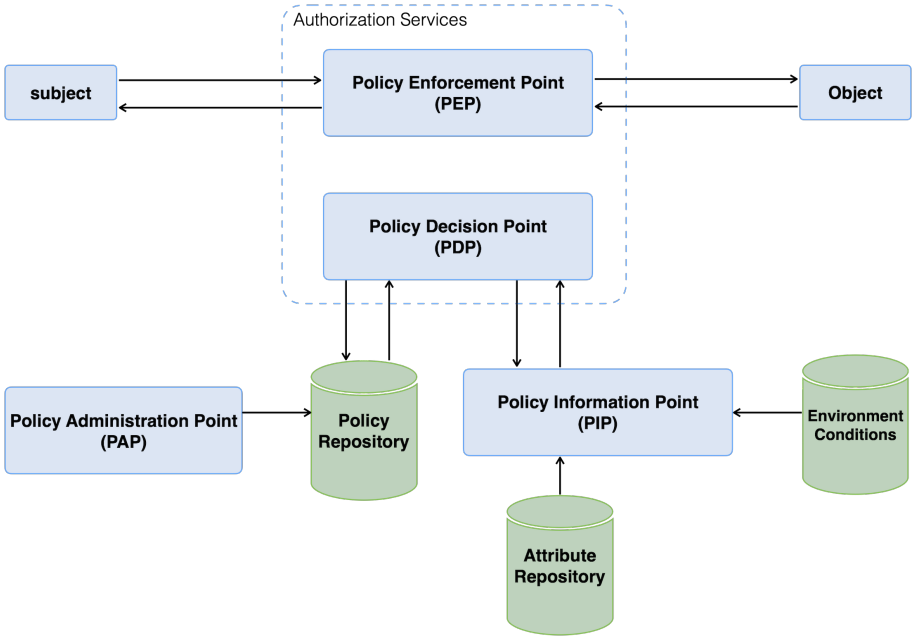


Figura 1.7: *Arquitetura de referência para mecanismos de controle de acesso. Fonte: (HU et al., 2014)*

mais fontes de atributos dentro da organização. O PAP corresponde a uma interface para a administração de políticas, que são então armazenadas em um repositório. Todos os componentes podem ser implementados de maneira centralizada ou distribuída, podendo estar logicamente e fisicamente separados uns dos outros ou agrupados em um único componente.

Tecnologias

Dentre as tecnologias consideradas como referências para autorização e controle de acesso a que mais se destaca é a XACML (RISSANEN, 2017). A *eXtensible Access Control Markup Language* é um padrão definido pela OASIS e compreende a definição de uma linguagem para criação de políticas de controle de acesso em formato interoperável, uma arquitetura e um modelo para o processamento de políticas e tomada de decisão. XACML define uma linguagem em XML para a especificação de políticas ABAC, e um protocolo para realizar requisições sobre decisões de controle de acesso (trocas de mensagens entre PEP e PDP), sendo implementado por diversas infraestruturas de autorização. Atualmente, em sua versão 3.0, a linguagem XACML define também um conjunto de extensões, incluindo um perfil para representação de políticas em formato JSON, perfil para definições de políticas RBAC, assim como um perfil para seu uso em arquiteturas RESTful.

O OAuth2 (HARDT, 2012) é um padrão para delegação de acessos em sistemas distribuídos. Sua principal funcionalidade é permitir que uma aplicação tenha acesso a recursos hospedados em outras aplicações em nome de um usuário. É considerado pela indústria o padrão *de facto* para autorização, sendo utilizado pelos grandes provedores de serviço da Internet (Amazon, Google, Facebook, Microsoft, e Twitter) para o compartilha-

mento de informações de contas de usuários para provedores de serviços de terceiros (veja o modelo de gestão de identidade centrado no usuário apresentado na Subseção 1.1.1).

A especificação OAuth2 provê suporte a diversos tipos de clientes (por exemplo, aplicações rodando em navegadores *web*, aplicações *web* executando em um servidor, aplicações em dispositivos móveis etc.) e define um conjunto de papéis, tipos de autorização e fluxos de autorização permitindo uma padronização e compartilhamento de informações entre organizações diferentes. Os fluxos definem as mensagens trocadas entre participantes, enquanto que os papéis identificam as responsabilidades dos participantes dentro do fluxo. Eles includem a identificação de proprietário de recurso (*Resource owner*) que delega o acesso a um cliente através de um servidor de autorização, o qual emite um *token* de acesso. O cliente então apresenta este *token* de acesso ao servidor do recurso que valida o *token* antes de liberar acesso ao recurso para o cliente.

O OAuth2 e o XACML desempenham funções distintas e podem ser facilmente utilizados em conjunto. Enquanto o OAuth2 pode ser utilizado para a delegação de acesso, políticas XACML podem ser utilizadas para decidir se um determinado acesso deve ser permitido ou não.

1.1.3. Privacidade e usabilidade

Do ponto de vista da engenharia de software, a usabilidade é considerada como um requisito de qualidade. Em IEEE... (1998), a usabilidade é definida como o quão fácil um usuário é capaz de aprender a operar, fornecer as entradas e interpretar as saídas de um sistema ou produto. Jøsang, Zomai e Suriadi (2007) definem um conjunto de princípios sobre usabilidade da segurança, classificando tais princípios como ações e conclusões. Eles consideram que o usuário deve entender quais ações de segurança precisa tomar e deve ter conhecimento e prática suficientes para tomar as ações corretas; além disso, a carga física e cognitiva para tomada dessas ações, mesmo que repetitivas, deve ser tolerável. As conclusões de segurança têm relação com a observação feita pelo usuário e, a partir desta, ele deve obter evidências sobre a segurança do sistema. Assim, em termos de usabilidade, para o usuário deve ser possível derivar a conclusão de segurança a partir da informação provida.

Segundo Schaar (2010), a maioria das pessoas possui pouca afinidade com tecnologia da informação e assim, não estaria na melhor posição para tomar decisões que impactam na proteção de seus dados pessoais e de outras pessoas. O conceito de privacidade desde a concepção (*Privacy by Design*), cunhado por Ann Cavoukian na década de 90 (CAVOUKIAN, 2009), considera que a privacidade dos usuários deve estar em foco desde a concepção do sistema ou produto e mantida até sua execução. Assim, considera que os sistemas ou produtos devam ser projetados de forma a minimizar a quantidade de informação pessoal tratada, além de fazer uso de mecanismos de segurança para garantir a proteção dos dados pessoais de seus usuários. Tais conceitos também estão presentes na Lei Geral de Proteção de Dados Pessoais (LGPD) (BRASIL, 2018, art. 46, §2) e na GPDR Europeia (UNION, 2016).

Como apresentado na Seção 1.2, a linha que separa o abuso do uso legítimo dos *cookies* pelos *sites* na Internet é tênue. A leis de proteção de dados pessoais exigem em muitas situações o consentimento do usuário antes que seus dados possam ser tratados. O

uso de *cookies* nos *sites* geralmente entram nessa situação, pois normalmente se enquadram nas situações em que o consentimento é exigido, haja visto que muitos são usados para propaganda e análise de comportamento e, portanto, precisam apresentar um termo de consentimento ao usuário. Para que o correto funcionamento do *site* não seja limitado ou completamente impedido, existe o conceito de *cookies* estritamente necessários, que se aplica ao que é essencialmente necessário armazenar para prover o serviço ao usuário, como por exemplo *cookies* para *login* em áreas protegidas ou *cookies* que mantém estado de carrinhos de compras[7]. Ainda assim, o usuário precisa ser avisado sobre o uso dos *cookies*.

O termo de consentimento também é apresentado aos usuários em soluções de autenticação que seguem o modelo de gestão de identidade federado (veja Subseção 1.1.1), como as federações acadêmicas, baseadas no protocolo SAML, ou que seguem o modelo centrado no usuário, como as aplicações que fazem uso do *login* social (contas Google, Facebook, Apple etc.), baseadas no protocolo OpenID Connect, que podem ainda questionar quais atributos o usuário deseja compartilhar com o provedor de serviço.

Os mecanismos usados atualmente pelos sistemas na *web*, para garantir a aderência às leis de proteção de dados, impactam diretamente na usabilidade destes sistemas. Em muitos casos eles podem até nem serem de fato aderentes à legislação, pois tornam os usuários habituados a consentirem por padrão, sem que o consentimento seja de fato livremente dado, específico, informado e sem ambiguidade, conforme dita a legislação. Por exemplo, os usuários são frequentemente inundados com *banners* de consentimento para o uso de *cookies*, com diferentes interfaces, apresentando-se como uma barreira entre o usuário e o serviço que ele deseja acessar. Em Utz et al. (2019) é apresentado o resultado de um estudo com mais 80.000 usuários sobre os formulários de consentimento para uso de *cookies* no qual foi notado que a maioria dos usuários está inclinada a sempre aceitar todos os *cookies* a ter que escolher individualmente quais *cookies* deseja permitir. Utz et al. (2019) consideram ainda que não basta apenar exigir o consentimento de usuário, também é necessário providenciar um guia de forma a padronizar uma interface para tal.

1.2. Demandas, desafios e tecnologias

Para que aplicações possam continuar operando de maneira adequada, é necessária a adequação aos diversos novos desafios, como demandas sociais e regulatórias de proteção de dados pessoais, novas formas de integração de serviço e diversidade de dispositivos utilizados pelo usuário, incluindo dispositivos que atuam em nome do usuário. Nesta seção são apresentadas as demandas históricas na área de gestão de identidade e de acesso, os novos desafios e as novas tecnologias e padrões para atender tais demandas, considerando o legado histórico de sistemas e a afinidade dos usuários com a tecnologia.

1.2.1. Autenticação federada, privacidade e novos mecanismos dos navegadores *web*

Muitas tecnologias e padrões usados para permitir autenticação e autorização federada ou centrada no usuário em aplicações *web* (e.g. SAML, OpenID Connect), apesar de terem sido desenvolvidas de maneira independente, foram projetadas para usufruir de funcionalidades de propósito geral presentes nos padrões *web*, como redirecionamento de

[7]https://gdpr.eu/cookies/

URL, parâmetros de URL, *iframe* e *cookies*. Nesta seção é feita uma breve apresentação sobre tais funcionalidades para depois apresentar os principais desafios para autenticação e autorização federada diante de alterações que já estão sendo implementadas pelos navegadores *web*, em particular o redirecionamento e restrições sobre *cookies*.

Na especificação do HTML5 (LAWSON et al., 2021), é dito que um contexto de navegação consiste em um ambiente no qual objetos do tipo `Document` são apresentados para o usuário. Assim, quando se abre uma nova janela ou aba em um navegador *web*, tem-se ali um contexto de navegação. O elemento HTML `iframe` representa um contexto de navegação aninhado, permitindo assim que uma janela do navegador *web* seja dividida em segmentos, cada qual podendo exibir um objeto `Document` diferente, sendo que cada documento pode ainda vir de diferentes servidores *web*. Toda vez que acessamos um *site* e este apresenta propagandas, oriundas do serviço *Google AdSense*, estamos presenciando o uso do elemento `iframe`. Tal elemento pode ser usado também para permitir a autenticação federada sem fricção, assunto que será apresentado logo mais nesta seção.

Os redirecionamentos de URL estão previstos na especificação HTTP (FIELDING; NOTTINGHAM; RESCHKE, 2022) para indicar ao agente do usuário (normalmente um navegador *web*) que este deve tomar alguma ação para que possa atender o pedido. O agente do usuário é informado por meio de um código de resposta da classe `3XX`. Por exemplo, para indicar que o recurso desejado (e.g. página HTML) está disponível em uma outra URL, é feito uso dos códigos `301`, movido permanentemente, ou `307`, redirecionamento temporário.

O formato de uma URL é definido por scheme://host:port/path?queryString#fragment. O elemento `path` consiste em segmentos de texto, delimitados pelo caractere `/`, que identificam um recurso na *web*. O elemento *query string* consiste em uma lista de parâmetros que é separada do recurso desejado pelo caractere `?`. Cada parâmetro consiste em um par (`nome=valor`) e são delimitados entre si pelo caractere `&`. Por exemplo: *https//www.exemplo.com/page?chave1=val1&chave2=val2&chave3=val3*.

A lista de parâmetros pode ser usada para alterar o comportamento de uma aplicação *web*. Por exemplo, o agente do usuário ao acessar a URL *https://www.exemplo.com/produtos?ordem=maior-valor*, indica a aplicação *web* que deseja receber a lista de produtos ordenada pelo maior valor. Como também pode ser usada em uma campanha de *marketing*, para saber de onde um usuário veio para chegar em uma determinada página. Por exemplo, a lista de parâmetros da URL *https://www.exemplo.com/?utm_source=twitter&utm_medium=tweet&utm_campaign=summer-sale* indica que o usuário chegou em `exemplo.com` a partir de um *tweet*, sobre uma liquidação de verão, publicado na rede social Twitter.

Os exemplos apresentados acima são considerados casos de propósito geral dos padrões *web*. Casos de uso específico, como o fluxo de autenticação federada, combinam o redirecionamento de URL com lista de parâmetros para permitir que seus usuários naveguem entre provedores de serviço e provedores de identidade. Por exemplo, quando um usuário acessa o serviço de periódicos da CAPES, ofertado por um provedor de

serviço na federação CAFe[8], este é redirecionado ao seu provedor de identidade, para que se autentique e, após isto, volta a ser redirecionado ao serviço da CAPES. O caminho de redirecionamentos é ditado pela lista de parâmetros, como apresentado nesta URL exemplo: *https://www.periodicos.capes.gov.br/Shibboleth.sso/Login?target=https://www.periodicos.capes.gov.br/secure&entityID=https://shibboleth.ifsc.edu.br/idp/shibboleth*.

Um *cookie* HTTP (BARTH, 2011) consiste de um conjunto de pares (*nome=valor*) e surgiu com o intuito de permitir ao agente do usuário (e.g. navegador *web*) manter o estado da aplicação na interação com um servidor HTTP, uma vez que o protocolo HTTP não mantém estado entre pedidos subsequentes. Por meio do campo *Set-Cookie* no cabeçalho HTTP, um servidor pode enviar um conjuntos de pares (nome=valor) e metadados associados para o navegador *web* do usuário, cabendo a este último decidir se armazena localmente ou simplesmente ignora as informações recebidas.

Atualmente, os *cookies* HTTP são usados devido a três principais motivos: gerenciamento de sessão – informações que geram facilidades na interação e só ficam persistidas durante a interação do usuário com a aplicação; personalização – preferências do usuário, como o idioma do conteúdo, e que devem ficar persistidas mesmo após o usuário finalizar a execução do navegador *web*; rastreamento – registros e análises do comportamento do usuário, podendo tais informações serem usadas com o intuito de *marketing* direcionado.

Os *cookies* podem ser ainda classificados como primários (*first-party*) ou de terceiros (*third-party*). Os *cookies* primários são criados e gerenciados pelo servidor HTTP, responsável pelo domínio *web* (*site*) que o usuário acessou diretamente. Os *cookies* de terceiros são criados por empresas detentoras de outros domínios *web*, cujo conteúdo é acessado indiretamente pelo usuário. Por exemplo, o usuário acessa o *site example.com*, cuja página HTML contém um *banner* de propaganda carregado a partir do domínio *example.net*. Esse segundo *site* solicita a persistência de um *cookie*, no caso, cria-se aqui um *cookie* de terceiro. Assim, quando este navegador *web* visitar outros *sites*, que também incluam recursos de *example.net*, este último conseguirá rastrear essa navegação (veja Figura 1.8).

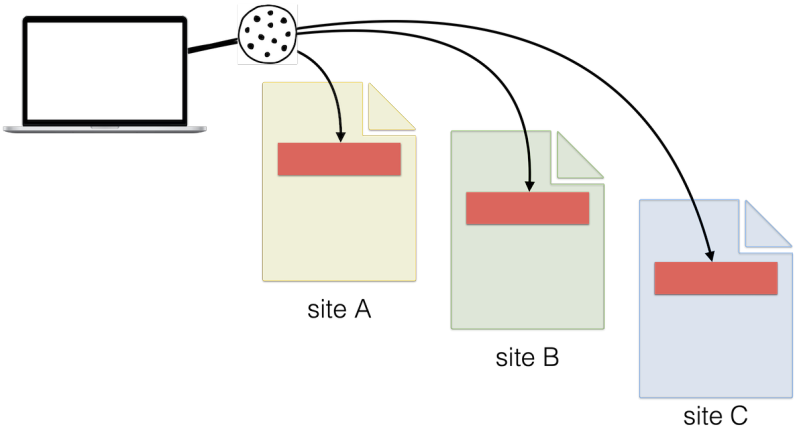


Figura 1.8: *Cookie de terceiro presente em diferentes sites que o usuário navega. Adaptado de Merewood (2020)*

[8]https://www.rnp.br/servicos/cafe

Ao criar um *cookie*, por meio do campo *Set-Cookie* no cabeçalho HTTP, o servidor pode definir algumas propriedades, como a *SameSite* (MEREWOOD, 2020), que no momento da escrita deste capítulo está em voga pelos principais navegadores *web*, porém ainda está como um *draft* IETF[9]. Esta propriedade permite que um *cookie* não possa ser enviado com requisições entre *sites*. Tem como objetivo proteger o usuário contra ataques de requisição forjada entre *sites* (*Cross-Site Request Forgery* – CSRF) e de *cookies* que objetivam fazer o rastreamento do usuário.

O redirecionamento de URL, parâmetros na URL e *cookies* são primitivas básicas das plataformas *web* e o uso legítimo destes possibilitaram uma melhor experiência de uso e a disponibilização de recursos mais ricos aos usuários. Porém, como são primitivas que podem ser usadas para uma grande variedade de finalidades, ao longo dos anos foram ocorrendo abusos no uso desses mecanismos, especialmente com propósitos de rastreamento e perfilamento de usuários, para propaganda direcionada, por exemplo. Isso têm causado um aumento de pressão social e especialmente regulatória por mais privacidade e proteção de dados.

Os desenvolvedores de navegadores *web* acabam intervindo e tomando medidas para mitigar a situação. Inclusive, tomam para si muitas vezes, conforme pontuado por Geradin, Katsifis e Karanikioti (2020), o papel de um "regulador de privacidade de fato", ao assumirem a interpretação dessas regulamentações e tomarem decisões de implementação para atendê-las, como a recente decisão dos principais desenvolvedores dos navegadores *web* de tornar por padrão o bloqueio total de *cookies* de terceiros.

A Google criou a iniciativa chamada *Privacy SandBox*[10] que tem por objetivo propor novas tecnologias que permitam proteger a privacidade dos usuários na *web* ao mesmo tempo que dá ferramentas para que desenvolvedores e empresas possam conduzir seus negócios na *web*. A empresa está buscando alternativas aos *cookies* de terceiros, iniciando com a proposta chamada *Federated Learning of Cohorts* (FloC) que posteriormente foi substituída pela proposta chamada *Topics*[11].

Em 2017, a empresa Apple implementou o *Intelligent Tracking Prevention* (ITP) (WILANDER, 2019) em seu navegador *web* Safari que teve como foco *cookies* de terceiros, porém, atualmente, a solução é mais ampla e bloqueia alternativas de contorno encontradas pelas empresas de anúncios, como os *link decoration*. A Mozilla implementou o *Enhanced Tracking Protection* (ETP) em seu navegador *web* Firefox, que bloqueia rastreadores de mídia social, *cookies* de rastreamento *cross-site*, *fingerprinters*, mineradores de criptomoedas, rastreadores de conteúdo e mais recentemente, remove parâmetros de URLs, a partir de uma lista conhecida, conforme configuração definida pelo usuário.

A decoração de *links* (*link decoration*) consiste na prática em usar parâmetros de URL para rastrear o usuário e assim contornar as restrições impostas com o bloqueio de *cookies* de terceiros. Os parâmetros podem ser adicionados na URL de maneira estática ou dinâmica, usando rotinas em Javascript, quando o usuário clica em um determinado elemento da página, como um anúncio.

[9]https://datatracker.ietf.org/doc/html/draft-west-first-party-cookies-07
[10]https://privacysandbox.com/
[11]https://blog.google/products/chrome/get-know-new-topics-api-privacy-sandbox/

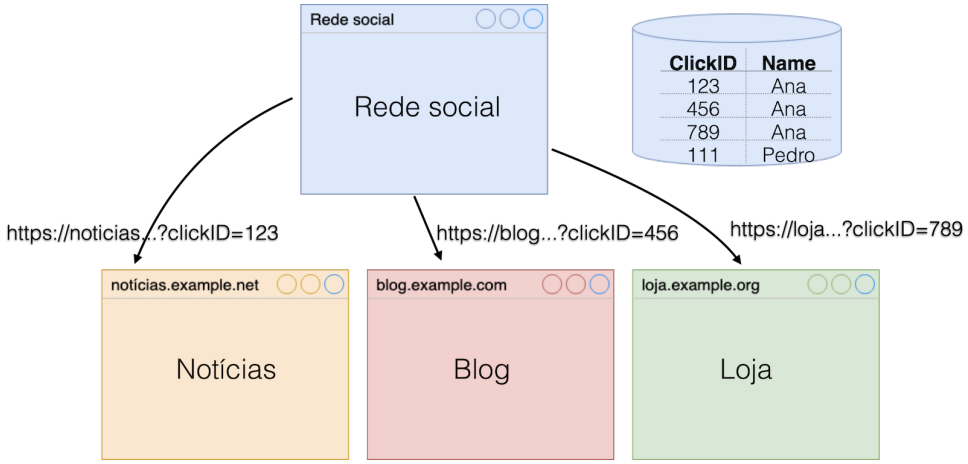


Figura 1.9: *Exemplo de link decoration com finalidade de rastreamento. Adaptado de Wilander (2019)*

No exemplo apresentado na Figura 1.9, a rede social adiciona o parâmetro *clickID* em todos os *links* externos ao seu domínio e o valor associado a este parâmetro tem ligação direta com um usuário real desta rede social. Quando o usuário clica no *link* para a ir ao site *loja.example.org*, a rede social faz uso de rotinas Javascript, já embutidas no *site* da Loja por algum outro motivo que seja interessante para este site, para extrair o parâmetro *clickID* da URL e o persiste como *cookie* primário do site da loja. O mesmo ocorre quando o usuário acessa os demais *links* a partir da rede social. Assim, toda vez que o usuário retornar à loja, a rotina Javascript da rede social lê o *cookie* primário e o encaminha para seus servidores, possibilitando assim rastrear o usuário.

Tem-se aqui mais um caso de abuso das primitivas da *web* e alguns navegadores *web*, como Safari e Firefox, estão criando mecanismos para coibir tal prática, identificando e removendo da URL parâmetros que são reconhecidos como rastreadores. Alguns sites, como o Facebook, começaram a cifrar toda URL de forma que estes mecanismos de bloqueio não consigam distinguir a parte útil da URL dos parâmetros que foram colocados unicamente com o propósito de rastreamento.

Iniciativas da comunidade de gestão de identidade

Redirecionamento e parâmetros de URL são essenciais para o funcionamento dos principais protocolos usados na autenticação federada. O grande desafio então é como manter o funcionamento das federações de identidade, com as políticas de privacidade de navegadores cada vez mais restritivas, respeitando a privacidade do usuário e melhorando sua usabilidade.

Embora o objetivo da arquitetura de identidade federada não seja rastrear o usuário, ela possibilita que o usuário possa ser rastreado pelo IdP, o qual sempre é contatado antes que consiga acessar os recursos providos pelo SP. Assim, o IdP toma conhecimento dos serviços que o usuário acessa, sem que isso seja de fato uma necessidade. Os SPs também podem fazer conluio para diretamente ou probabilisticamente ligarem seus usuários e assim fazer a prática de enriquecimento de perfil, adicionando mais dados do que realmente precisam sobre seus usuários.

Iniciativas de debates na comunidade de GId têm sido realizadas na forma de eventos, como por exemplo o *Federated Identity and Browser Workshop*, apoiado pela REFEDs (*Research and Education FEDerations group*) e realizado em 2021. A partir desse *workshop* houve uma articulação para criar um grupo comunitário da W3C sobre o tema. Atualmente este grupo está trabalhando na *Federated Credential Management*[12] que consiste em uma API para permitir que usuários façam login em *sites web* com suas contas federadas e ainda assim tendo a sua privacidade preservada.

O projeto SeamlessAccess (SEAMLESSACCESS, 2021) é uma iniciativa conjunta de quatro entidades ligadas à educação e publicação de pesquisa científica, a GÉANT[13], Internet2[14], NISO[15] (*National Information Standards Organization*) e STM[16] (*International Association of STM Publishers*), com o objetivo de resolver os atuais problemas de usabilidade das federações acadêmicas baseadas em asserções SAML, para permitir uma experiência de autenticação única (*Single Sign-On* – SSO) verdadeira e transparente.

Nessa proposta, o serviço de descoberta de IdP (*Discovery Service* – DS) continua a existir, porém os SPs podem escolher um dos três modos como o DS seria apresentado aos seus usuários: modo limitado, semelhante ao que se tem em alguns serviços na federação CAFe, ou seja, com o redirecionamento HTTP para o DS bem evidente ($SP \rightarrow DS \rightarrow IdP \rightarrow SP$); modo padrão, no qual o DS aparece embarcado e integrado com a página *web* do próprio SP e o redirecionamento ao IdP só aconteceria no primeiro acesso do usuário a um serviço federado durante aquela sessão do navegador *web* ($SP \rightarrow IdP \rightarrow SP$); modo avançado, com comportamento semelhante ao modo padrão, porém, permite ao SP indicar a lista de IdP que confia.

Alguns provedores de serviços presentes na federação CAFe, como o da Escola Superior de Redes, possuem o DS embarcado em sua própria página, permitindo que o usuário siga o fluxo $SP \rightarrow IdP \rightarrow SP$. Porém, se o usuário acessar outros provedores de serviços, durante a mesma sessão no navegador *web*, este ainda terá o passo adicional em indicar seu IdP novamente, em um DS embarcado ou dedicado, antes que consiga acessar o recurso desejado.

Com o projeto SeamlessAccess, a escolha do IdP do usuário é persistida no recurso de armazenamento local do navegador, o *localStorage*, e este permite que o usuário possa acessar diferentes provedores de serviços sem que tenha a necessidade de indicar o seu IdP em cada acesso. Ao contrário dos *cookies*, a informação armazenada no *localStorage* não é encaminhada em cada requisição e somente o domínio que a escreveu é quem tem permissão de lê-la. Assim, a solução proposta pelo SeamlessAccess continua efetiva mesmo diante das restrições de bloqueio de *cookies* de terceiros por parte dos navegadores *web*.

[12]https://fedidcg.github.io/FedCM
[13]https://geant.org/
[14]urlhttps://internet2.edu/
[15]https://niso.org/
[16]https://stm-assoc.org/

1.2.2. Robustez do processo de autenticação

Segundo NIST (2017a), o processo de autenticação digital busca garantir que um determinado sujeito possui controle sobre um ou mais autenticadores (e.g. senha, chave privada etc.) que estejam associados à sua identidade digital. Os sistemas de autenticação estão fundamentados sobre três fatores de autenticação:

- Aquilo que você sabe – senha, número de identificação pessoal (*Personal Identification Number* – PIN) etc;
- Aquilo que você possui – carteira de identificação, *token* criptográfico etc;
- Aquilo que você é – biometria do sujeito.

O primeiro uso do par *(username, password)* em sistemas computacionais é atribuído a Fernando Corbató na década de 60 (YADRON, 2014), quando buscava um meio para permitir o compartilhamento de computadores do tipo *Compatible Time-Sharing System* (CTSS) por vários usuários, de forma que cada usuário tivesse uma área privativa onde seus arquivos não poderiam ser vistos pelos demais usuários.

Apesar das fragilidades, o par *(username, password)*, ainda é amplamente empregado em processos de autenticação digital de sujeitos (DASGUPTA; ROY; NAG, 2017). Assim, sistemas de autenticação que dependem exclusivamente deste par precisam fazer uso de outros mecanismos de segurança para lidar com usuários que fazem uso de estratégias ruins para escolha de senha, que geram facilidade para ataques de força bruta, com possíveis *malwares* no dispositivo do usuário, que visam a captura da senha, ou ainda, com técnicas de engenharia social como o *phishing*, que também visam a captura da senha. Em Mello e Chaves (2020), notou-se que alguns provedores de serviços comerciais optaram por senha de uso único (*One-Time Password* – OTP) que é enviada para o email do sujeito durante o processo de autenticação. Assim, assumem que somente o usuário teria acesso à sua caixa postal, não precisam implementar processos para recuperar senhas e evitam a escolha de senhas frágeis por parte de seus usuários.

Fatores de autenticação do tipo “aquilo que você possui” ou “aquilo que você é” não estão suscetíveis aos mesmos ataques que fatores do tipo “aquilo que você sabe” estão. Por exemplo, um atacante remoto, que faz uso de força bruta para descoberta de senha, seria capaz de fazer o comprometimento em massa de contas de usuários de um serviço, considerando que não existam outros mecanismos de segurança para detecção e mitigação do ataque. Porém, se a autenticação do usuário também exigir algo que esteja fisicamente perto do usuário (e.g. um *token* criptográfico, sua impressão digital), tal ataque não teria sucesso.

A autenticação multifator (*Multi-Factor Authentication* – MFA), que em alguns casos também é chamada de autenticação com dois fatores (*Two Factor Authentication* – 2FA), visa aumentar a robustez do processo de autenticação por meio da combinação de mais de um fator de autenticação. Parte-se do pressuposto que aumenta muito o grau de dificuldade para que o atacante consiga comprometer mais de um fator.

Apesar do conceito de autenticação com dois fatores não ser algo novo, foi somente há pouco tempo que de fato ganhou popularidade. Atualmente, os principais provedores

de serviços comerciais na Internet exigem ou permitem que seus usuários façam uso da autenticação com dois fatores. Como primeiro fator é comum o uso de senhas escolhidas pelo próprio usuário (aquilo que você sabe) e como segundo fator as senhas de uso único (OTP), podendo estas serem enviadas por email, SMS, ligação telefônica ou ainda gerenciadas por meio de aplicativos (e.g. Google Authenticator ou Authy) em dispositivos móveis (veja Figura 1.10). Neste caso, assume-se que o sujeito está em posse do dispositivo do qual conseguirá obter a senha que deverá ser apresentada no processo de autenticação.

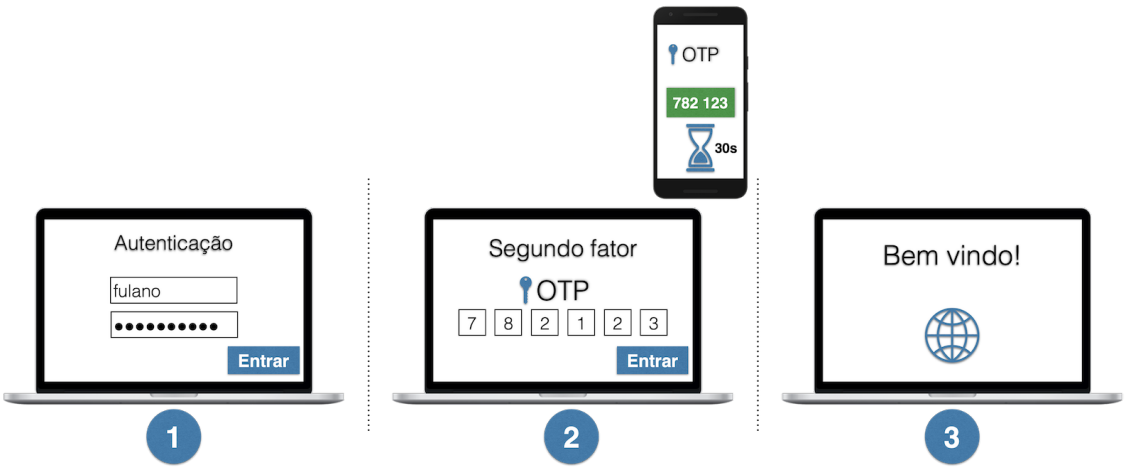


Figura 1.10: *Autenticação com dois fatores, tendo senha de uso único como segundo fator*

Segundo NIST (2017a), o uso de dois fatores de autenticação é o suficiente para um sistema de autenticação atingir o maior nível de confiança, na escala de um a três, definido em (NIST, 2017b), desde que os fatores usados sejam resistentes ao ataque de *phishing*. O uso de outras informações, como a localização geográfica ou o identificador do dispositivo usado pelo sujeito, ajudam na análise de risco para determinar se o sujeito é de fato detentor dos autenticadores, porém tais informações não são consideradas fatores de autenticação.

As senhas de uso único, enviadas por meio do serviço telefônico (ligação ou SMS) são consideradas de uso restrito pelo NIST, uma vez que existe um risco sobre a personificação da estação móvel do usuário. Assim, considera-se alinhado ao primeiro nível de confiança (o mais baixo nível), um sistema de autenticação que faz uso de uma senha como primeiro fator e senhas de uso único como segundo fator, desde que se faça uso de *tokens* físicos dedicados ou aplicativos TOTP (M'RAIHI et al., 2011) em telefones inteligentes. Apesar das senhas de uso único como segundo fator aumentarem a robustez do processo de autenticação, soluções baseadas no envio desta senha por email, ligação telefônica ou SMS ainda estão sujeitas a ataques de engenharia social, onde agentes maliciosos iniciam o processo de autenticação e induzem a vítima a fornecer o código OTP recebido.

O uso de aplicativos dedicados para geração do código OTP podem tornar este tipo de ataque mais difícil, pois exige uma consciência muito maior da vítima, podendo esta perceber que trata-se de um ataque. Porém, estes aplicativos apresentam um ponto negativo com relação a usabilidade, pois para cada autenticação iniciada por um usuário

correto, em um serviço correto, o usuário precisará procurar, abrir o aplicativo e transcrever o código que vê, dentro da janela de 30 segundos, no formulário de autenticação. A usabilidade fica ainda mais prejudicada se o usuário estiver usando seu telefone inteligente para acessar o serviço desejado e neste dispositivo estiver o aplicativo gerador de código OTP (REESE et al., 2019). Por fim, usuários de aplicações *web* que fazem uso deste tipo de solução ainda estão suscetíveis a ataque de *phishing* combinado com o ataque do homem no meio (*man-in-the-middle*) e sequestro de sessão por meio de *cookies* (GRIMES, 2019; MICROSOFT, 2022).

Soluções como o *Google Prompt* ou *Duo Push* surgiram como uma solução para prover uma melhor experiência de uso, quando comparados com aplicativos gerenciadores de códigos OTP. Durante o processo de autenticação, depois do usuário fornecer o primeiro fator, uma notificação aparece no telefone inteligente do usuário e basta esse pressionar o botão `SIM` para confirmar que é ele quem está querendo autenticar-se. Tal solução está suscetível a ataques, batizados de *MFA prompt bombing* (GOODIN, 2022), que apostam na falta de atenção da vítima para as notificações que aparecem em seu dispositivo.

A partir de 2015, a *Fast IDentity Online Alliance* (FIDO) publicou um conjunto de especificações abertas com o intuito de permitir que a autenticação de usuários na *web* seja simples e robusta. As especificações estão fundamentadas sobre criptografia de chave pública, autenticadores baseados em *hardware* seguro para armazenamento do material criptográfico, como *Secure Element* (SE), *Trusted Execution Environment* (TEE) e *Trusted Platform Module* (TPM). No caso é criado um par de chaves criptográficas, que não pode ser extraído do dispositivo FIDO, para cada *site* ou aplicativo (chave fica vinculada à URI do *site* ou aplicativo) que o usuário for se autenticar. Tem-se ainda a facilidade para o usuário evitar *phishing*, pois este não é responsável por confirmar se um *site* é realmente quem ele afirma ser, sendo esta uma responsabilidade do dispositivo FIDO.

Foram publicados os padrões *Universal Second Factor* (U2F) (SRINIVAS et al., 2017), *Universal Authentication Framework* (UAF) (MACHANI et al., 2020) e o *Client to Authenticator Protocol* (CTAP) (BRADLEY et al., 2021), sendo este último um complemento à especificação *Web Authentication* (WebAuthn) (HODGES et al., 2021) da W3C, que algumas vezes também é referenciada como FIDO2. Atualmente a recomendação é que provedores de serviços façam uso do WebAuthn e CTAP e não usem mais UAF e U2F, porém tais padrões serão explicados na sequência dada sua grande relevância histórica e para que o leitor entenda as implicações de cada alternativa que surgiu na evolução dos padrões FIDO.

A especificação U2F teve como foco dispositivos físicos (*tokens* criptográficos) que pudessem ser usados exclusivamente como segundo fator de autenticação, uma vez que o dispositivo em si não possui qualquer tipo de mecanismo de autenticação local que impeça seu uso por qualquer pessoa. De acordo com a especificação, os dispositivos U2F devem possuir um mecanismo físico, normalmente um botão capacitivo, para confirmar que o usuário está presente e participando de forma ativa de um pedido de autenticação (veja Figura 1.11). O motivo para isto é que, como os dispositivos U2F podem estar constantemente conectados na porta USB do computador do usuário, um aplicativo malicioso neste computador poderia iniciar um pedido de autenticação e usar o dispositivo sem que o usuário soubesse.

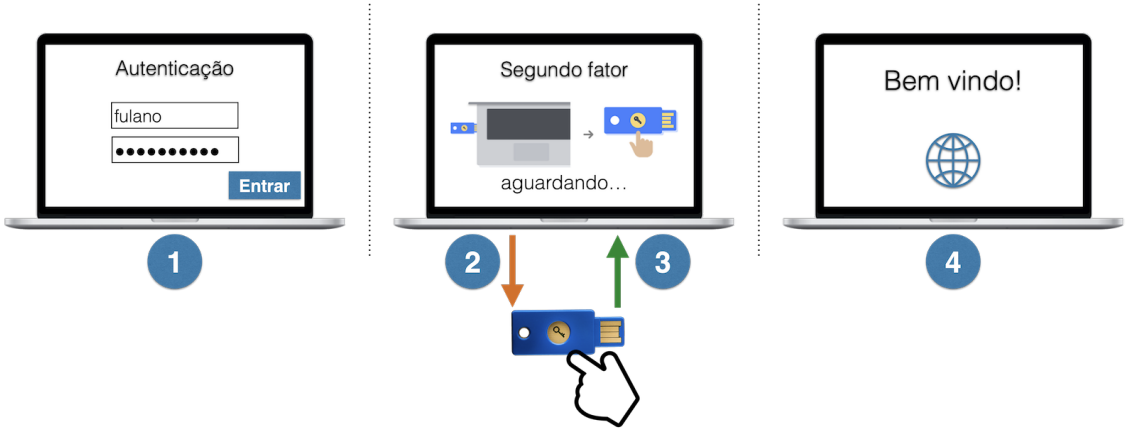


Figura 1.11: *Autenticação com dois fatores, tendo dispositivo U2F como segundo fator*

Dispositivos U2F podem possuir interface USB (A ou C), Bluetooth BLE ou *Near Field Communication* (NFC), permitindo assim que possam ser usados em computadores, telefones inteligentes ou *tablets*. Na Figura 1.12 são apresentados alguns exemplares de produtos que implementam a especificação U2F e também FIDO2. Todos os produtos implementam interface de comunicação NFC, sendo que os produtos da Google e da Yubiko possuem interface de comunicação USB C e da Solokey interface USB A.

Figura 1.12: *Chaves FIDO2 Google Titan, Yubiko Yubikey e Solokey*

A especificação UAF teve como foco proporcionar uma experiência de autenticação totalmente sem senha, isto é, um dispositivo que esteja de acordo com a UAF pode ser usado como o único fator de autenticação. No caso, este dispositivo (i.e computador ou telefone inteligente) precisa possuir um hardware de seguro (e.g. SE, TEE, TPM) para armazenar o material criptográfico e uma forma de autenticação local no dispositivo, normalmente por meio de algum leitor biométrico, para que seja destravada a chave privada para uso em um processo de autenticação. Dispositivos UAF estão aderentes ao terceiro nível de confiança de sistemas de autenticação definido em (NIST, 2017b).

Os padrões UAF e U2F são complementares e, apesar do U2F ter tido uma maior adoção no mercado, o número de provedores de serviços que fizeram uso deste ainda era pequeno. A especificação WebAuthn conduzida pela W3C em conjunto com a FIDO Alliance surgiu como uma solução para permitir a ampla adoção dos padrões FIDO, uma vez que a especificação padroniza uma API para os navegadores *web*. Atualmente, os principais navegadores *web*, de dispositivos móveis ou de computadores, possuem suporte

a especificação WebAuthn[17] e um grande número de provedores de serviços comerciais (e.g. Dropbox, GitHub, Google, iCloud etc.) já permitem o uso do WebAuthn como um fator de autenticação.

A especificação WebAuthn permite dispositivos como chaves de segurança USB (chamados de autenticadores externos) ou dispositivos como um telefone inteligente ou computador (chamados autenticadores de plataforma) que possuam um hardware seguro (e.g. TEE) mais um leitor que permita autenticação biométrica (e.g. leitor de impressão digital, reconhecimento facial etc.). Os dispositivos WebAuthn podem ser usados como o único fator de autenticação, quando for um telefone ou computador, ou como segundo fator de autenticação, quando for uma chave USB que não permite realizar a autenticação local do sujeito. Assim, autenticação baseada nos padrões FIDO conseguem atingir o mais alto nível de confiança (nível 3) descrito em (NIST, 2017b).

Dispositivos WebAuthn (de plataforma ou externos), na fase de registro (e.g. criação da conta do usuário em um provedor de serviço) sempre irão gerar um novo par de chaves criptográficas exclusivo para aquela conta de usuário e para aquele provedor de serviço. Se o dispositivo WebAuthn tivesse uma única chave privada e não gerasse uma nova por provedor de serviço, os provedores poderiam formar um conluio para tentar rastrear as atividades deste usuário, sendo que a chave privada atuaria como um identificador único e universal deste usuário.

O fato de sempre gerar um novo par de chaves na fase de registro, permite que dispositivos WebAuthn possam ser compartilhados, por exemplo, por todos os membros de uma família, sem possibilitar que um usuário consiga usar as chaves criptográficas do outro usuário. Assim, um mesmo dispositivo WebAuthn pode ser usado por um sujeito em diferentes provedores de serviço, por diferentes sujeitos em um mesmo provedor de serviço ou por diferentes sujeitos em diferentes provedores de serviços.

Se o dispositivo WebAuthn tiver capacidade de armazenamento, a chave privada pode ser armazenada nele, porém no caso de chaves USB como aquelas apresentadas na Figura 1.12, informações usadas para gerar a chave privada (mas não a própria chave) são encapsuladas e armazenadas no provedor de serviço. Durante a fase de autenticação, essas informações são encaminhadas pelo provedor de serviço ao computador do usuário e este as encaminha à chave USB, a qual será capaz de derivar a chave privada a partir destas informações, para então empregá-la no processo de autenticação.

Em NIST (2017b), é apresentado um conjunto de eventos que podem ocorrer durante do ciclo de vida de um autenticador (e.g. uma senha ou um dispositivo WebAuthn) que afeta diretamente o uso deste autenticador. Os eventos incluem a associação do autenticador à conta do sujeito no provedor de serviço, a perda, o roubo, a duplicação não autorizada, a expiração e a revogação.

Se o autenticador for uma senha e seu detentor vier a perdê-la, este poderia recorrer ao processo de redefinição de senha, que geralmente é composto por três passos: pedido de recuperação; verificação se o pedido não foi feito por um *bot*; redefinição de senha. O último passo pode fazer uso de perguntas de segurança previamente cadastradas pelo sujeito ou o envio de um código temporário para o email do sujeito o qual lhe permitirá

[17] https://webauthn.me/browser-support

redefinir a senha (MAQBALI; MITCHELL, 2018).

Se o autenticador perdido for do tipo "aquilo que você possui", como um *token* criptográfico, uma chave USB WebAuthn ou a semente usada para geração das senhas de uso único (OTP), o processo de redefinição não pode ser semelhante ao de redefinir uma senha, uma vez que tal processo não é tão robusto quanto o próprio processo de autenticação, tornando-se assim um possível alvo de ataques.

Alguns aplicativos gerenciadores de senhas de uso único, como o Authy[18], permitem que o usuário faça uma cópia de segurança das sementes que possui em seu telefone inteligente na nuvem, ou mesmo sincronizar as sementes por múltiplos dispositivos. A cópia de segurança é protegida unicamente por uma senha, que se for perdida, não poderá ser recuperada e, por consequência, tornará a cópia de segurança permanentemente indisponível.

Segundo o relatório de 2021 da Gartner (PHILLIPS, 2021), o WebAuthn venceu o pico das expectativas infladas e situa-se no vale da desilusão, uma vez que sua adoção vem avançando e conquistando um grande número de clientes. Porém, as chaves USB WebAuthn ainda enfrentam alguns desafios, do ponto de vista dos usuários finais. As chaves estão disponíveis para venda em poucos mercados, restringindo-se principalmente aos Estados Unidos da América e alguns países na Europa. As chaves custam em média US$ 35, assim observa-se uma adoção maior por empresas, que adquirem para seus funcionários, e um menor interesse quando a pessoa precisa adquirir com recursos próprios.

Além da logística e do custo monetário, as chaves USB geram uma dificuldade extra para os usuários caso estes venham a perdê-la. Ciente que uma mesma chave USB pode ser usada em diferentes provedores de serviços, a perda deste dispositivo pode deixar o usuário sem conseguir acessar um grande número de serviços, caso este não tenha configurado outros fatores de autenticação como opção de *backup*. Problema semelhante ocorre quando se usa autenticadores de plataforma, embarcados em telefones ou computadores. Apesar de ser mais difícil perder tais dispositivos, a troca destes, por exemplo quando o sujeito adquire um novo, é algo mais comum. Assim, este sujeito ficaria impossibilitado de passar pelo processo de autenticação com este novo dispositivo e para evitar isto, e não depender de fatores mais frágeis, precisaria ter também uma chave USB WebAuthn (autenticação externo) associada à sua conta.

O WebAuthn é um fator de autenticação robusto, resistente a *phishing*, simples de usar, mas a impossibilidade de recuperar as chaves privadas de dispositivo perdido faz com que os usuários continuem dependentes de outros fatores. Para o usuário comum, a autenticação é algo que deve simplesmente funcionar sem que necessite adquirir dispositivos adicionais ou lidar com inconveniências.

Em 2022, a FIDO Alliance e a W3C propuseram uma nova versão do nível 3 da especificação WebAuthn, chamada de *Multi-device FIDO Credentials* - para a qual algumas empresas estão usando o termo chaves de acesso (*passkeys*) (FIDO, 2022). Na proposta, considera-se que as chaves de acesso (*passkeys*) serão capazes de substituir as senhas (*password*) até mesmo em cenários que exigem um alto nível de segurança e confiança nos autenticadores.

[18]https://authy.com/

A proposta busca avançar em dois cenários principais: permitir que telefones inteligentes possam ser usados como autenticadores externos (como uma chave *Bluetooth*) e propor alterações na implementação dos autenticadores de plataforma (computadores e telefones) para permitir que as credenciais FIDO (chaves privadas) possam ser sincronizadas por múltiplos dispositivos, permitindo que o usuário as transfira facilmente para um novo computador ou telefone recém adquiridos.

A experiência do usuário com as chaves de acesso será semelhante a experiência com gerenciadores de senhas, que podem ter sua base sincronizada por diversos dispositivos, são protegidos por uma única senha principal e que permitem o preenchimento automático do formulário de autenticação. Desta forma, os usuários conseguirão acessar suas chaves de acesso (*passkeys*) nos seus diferentes dispositivos, mesmo em novos dispositivos, sem a necessidade de passar pelo processo de registro de credenciais para cada *site* ou aplicativo que já o tenha feito anteriormente.

Se o usuário possuir dispositivos de um mesmo fabricante, por exemplo, um telefone Android e um laptop Chromebook, então haverá a sincronização automática das chaves de acesso por meio de sua Conta Google. Porém, também é possível o cenário no qual o usuário possui um Chromebook e um telefone com o sistema iOS, ou seja, de diferentes fabricantes. Por exemplo, o usuário possui uma chave de acesso criada para o site *example.com* quando ele o acessou por meio de seu telefone com iOS. Quando este usuário for acessar pela primeira vez o site *example.com* por meio de seu computador (que também possui um autenticador de plataforma), o navegador *web* apresentará um diálogo questionando se deseja usar seu telefone como chave *Bluetooth*. Se sim, este usuário fará autenticação local em seu telefone (e.g. usando o reconhecimento facial) e, autenticando com sucesso, o site *example.com* pode associar uma nova chave de acesso específica para este computador. Desta forma, nos próximos acessos por meio desse computador não será mais necessário fazer uso do telefone inteligente como uma chave *Bluetooth*.

A segurança e o sincronismo das chaves de acesso de um usuário depende diretamente do sistema operacional subjacente do autenticador (plataforma) para suas contas *online* (e.g. Conta Google) e do método de segurança para restabelecer o acesso às chaves de acesso quando todos os dispositivos foram perdidos (FIDO, 2022). Apple, Google e Microsoft firmaram compromisso para acelerar a adoção das chaves de acesso e é esperado que os primeiros serviços, e adequações nos sistemas operacionais, estejam disponíveis ao longo do ano de 2023 (APPLE, 2022).

Por fim, a proposta também possibilita o cenário onde um provedor de serviço, por questões regulatórias ou de segurança, deseja realizar passos adicionais para ter certeza sobre o usuário, quando este apresentar uma chave de acesso por meio de novo dispositivo. Neste caso, a proposta permitirá que o provedor de serviço crie uma chave criptográfica adicional vinculada ao dispositivo. Assim, nos pedidos de autenticação posteriores, esta chave do dispositivo é também usada e ajudará ao provedor de serviço ter certeza de que o usuário está fazendo uso de um dispositivo já conhecido e não de um novo dispositivo.

Sistemas Adaptativos de Autenticação

Sistemas adaptativos de autenticação[19] estão aptos a modificarem dinamicamente seu comportamento para escolha do(s) melhor(es) mecanismo(s) em resposta a fatores contextuais, tais como localização, proximidade de dispositivos e outros atributos (CABARCOS; KRUPITZER; BECKER, 2019). Tornar adaptativa a autenticação nos provedores de identidade permite que esses ofereçam diversidade de mecanismos e de fatores de autenticação, monitoramento em tempo real (de modo a validar o usuário durante a sessão estabelecida – autenticação contínua) e extensibilidade para novas tecnologias e fatores de autenticação, sem que seja necessário a mudança completa da arquitetura do sistema.

Um sistema adaptativo compreende duas partes: um conjunto de recursos gerenciados e a lógica de adaptação. Mapeando para o domínio de autenticação, os recursos gerenciados são os autenticadores (disponíveis nos dispositivos e em aplicativos do usuário) e a lógica de adaptação é a camada de *software* encarregada de orquestrar seu uso de acordo com a situação detectada. Além disso, a lógica de adaptação pode ser executada no mesmo dispositivo da aplicação que deseja utilizar os autenticadores, ou em outro dispositivo, possibilitando diferentes casos de uso. Um exemplo de autenticação adaptativa com lógica no dispositivo ocorre quando um *smartphone* que detecta quando o usuário está em casa e desativa a proteção por senha (modificação automática de comportamento) até que ele se mude para um local diferente (HAYASHI et al., 2013). Outro exemplo, quando a lógica de adaptação é distribuída, é um sistema no qual um usuário se autentica com o leitor de impressão digital do *smartphone* para acessar seu *laptop*, quando os dois dispositivos estão próximos. A autenticação dinâmica, além de aumentar o nível robustez do processo de autenticação do ponto de vista de segurança computacional, também pode melhorar a usabilidade do sistema, visto que pode se adaptar para promover baixa fricção do usuário ao se autenticar.

De acordo com Dasgupta, Roy e Nag (2017), a autenticação contínua, também conhecida como autenticação ativa, foi introduzida em 2012 para abordar novas formas de validar a identidade dos usuários, em vez de usar apenas senhas tradicionais. O foco estava principalmente na biometria de comportamento baseada em *software* que capturava os dados da sessão para determinar se o usuário legítimo estava usando o sistema em um determinado momento. Segundo o Programa de Autenticação Ativa da DARPA (GUIDORIZZI, 2013), uma pessoa pode ser autenticada em intervalos regulares por:

- Aspectos físicos – impressão digital, geometria facial etc.;
- Interação com o sistema – padrão de pressionamento de tecla, padrão de digitação e movimento do mouse etc.;
- Contexto existente do usuário – análise semântica estrutural, como o usuário constrói sentenças, forense de autoria etc.; ou uso de dados de suas experiências, sendo linguística computacional ou como o usuário usa a língua.

[19]Na literatura, termos semanticamente similares são: autenticação ciente e baseada em contexto, autenticação ciente e baseada em risco.

A inclusão de diversos conjuntos de autenticadores melhora a flexibilidade do sistema adaptativo, favorecendo sua aplicabilidade a diferentes cenários. Alguns sistemas se concentram em melhorar a usabilidade por meio da adaptação, para a qual a biometria comportamental é uma boa candidata, uma vez que o usuário pode ser implicitamente "sentido" sem exigir interação explícita (RYU et al., 2021). A autenticação implícita possibilita que o usuário se autentique em algum serviço sem precisar que o mesmo coloque suas credenciais manualmente ou requer pouco envolvimento ativo do usuário (JAKOBSSON et al., 2009), o que proporciona uma grande aceitação (CABARCOS; KRUPITZER; BECKER, 2019).

As abordagens de autenticação implícita do usuário são ótimas candidatas para realizar a autenticação contínua. Por exemplo, as atividades na *web* de um usuário podem ser continuamente verificadas quanto a irregularidades em seu fluxo de trabalho normal e padrões de interação da interface do usuário. Os aplicativos de *smartphone* podem verificar regularmente a bio-assinatura ou o padrão de comportamento baseado em localização do usuário para detectar um impostor. A autenticação implícita e contínua pode ser utilizada para aumentar a qualidade do processo de autenticação, dado que tanto características comportamentais (JAKOBSSON et al., 2009) quanto contextuais (WU et al., 2019) podem ser utilizadas com a finalidade de detectar se o usuário conectado ainda é o usuário inicialmente autenticado. A autenticação implícita pode ainda ser usada para melhorar a experiência do usuário quando se faz uso de um segundo fator de autenticação.

Outro exemplo de abordagem adaptativa é a autenticação baseada em risco (*Risk-Based Authentication* - RBA), no qual um sistema captura e armazena diferentes tipos de informações (e.g. dados do usuário, do dispositivo, endereço IP, geolocalização, metadados do navegador, tipo de operação a ser executada no sistema etc.), a partir disso calcula uma pontuação e gera uma classificação de risco (e.g. baixo, médio ou alto), e então decide o nível apropriado de segurança e o mecanismo ou fator de autenticação a ser utilizado (NIST, 2017b). Muitas soluções RBA usam aprendizado de máquina. Os algoritmos dessas ferramentas monitoram e aprendem o comportamento do usuário ao longo do tempo para criar um perfil preciso dos padrões de login de um determinado usuário. Estas podem monitorar em tempo real dispositivos, horários típicos de login do usuário ou locais de trabalho habituais para identificar anomalias nos padrões de autenticação do mesmo. Eles verificam endereços IP e reputações de rede, além de dados de ameaças para essas redes no caminho da autenticação (como redes comprometidas).

Na Figura 1.13 é ilustrado um exemplo de RBA, no qual, se o risco for considerado baixo (por exemplo, dispositivo comum, local e horário usuais), exige-se somente uma autenticação baseada no par (usuário e senha). Em um risco médio (por exemplo, dispositivo desconhecido em local e em horário usuais), o serviço RBA solicita ao usuário informações adicionais (por exemplo, verificação de endereço de e-mail). Se a pontuação de risco for considerada alta (por exemplo, dispositivo desconhecido em local irreal e em horário incomum), uma nova requisição com um segundo fator pode ser solicitada ao usuário. No caso de uma pontuação de risco alto e operação a ser executada for crítica, o serviço poderá bloquear a tentativa de acesso (WIEFLING; DÜRMUTH; LO IACONO, 2020).

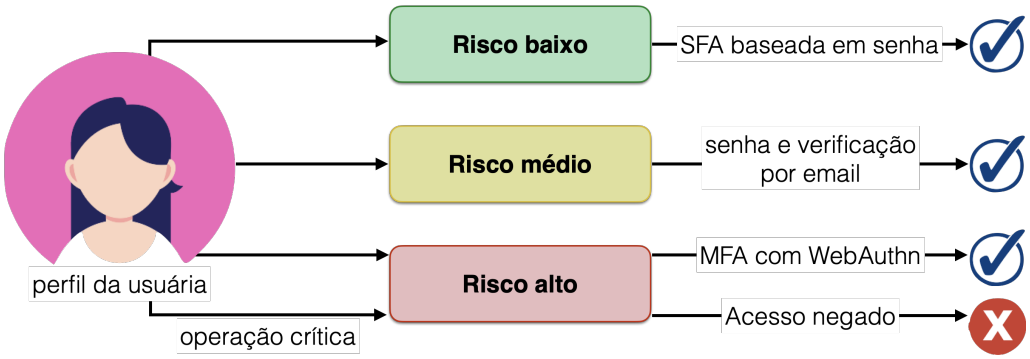


Figura 1.13: *Exemplo de Sistema de Autenticação Baseado em Riscos*

Algumas plataformas de identidade comerciais, como por exemplo a OKTA[20], a Azure Active Directory[21] e OneLogin[22], possibilitam a criação de políticas de acesso contextual que avaliam fatores de risco, como dispositivo, rede, localização, viagem, IP e outros contextos, em cada etapa do processo de autenticação. Em seguida, analisa o nível de risco com as configurações de autenticação apropriadas, como solicitar MFA ou usar autenticação sem senha para um acesso de baixo risco.

O *framework Shibboleth*, que implementa o padrão SAML e o modelo de identidades federadas, permite aos IdPs implementarem novos fluxos de autenticação, além daqueles já existentes por padrão (e.g. senha, certificado X.509, endereço IP). Esse *framework* possui ainda um tipo especial chamado *MFA Flow*, o qual fornece uma maneira programável de combinar diferentes tipos de fluxos de autenticação, bem como orquestrar sequências de execução destes fluxos para criarem cenários adaptativos de autenticação, inclusive baseados em riscos, que levem em consideração o contexto e o comportamento do usuário. Um cenário possível para a implementação de autenticação dinâmica em IdPs *Shibboleth* é utilizar autenticação implícita baseada em contexto (endereço IP, geolocalização e *user agent*) como segundo fator de autenticação. O IdP autentica o usuário com usuário e senha e armazena as informações de endereço IP, *user agent* e geolocalização. Para isso, o provedor de serviço (SP) executa, em segundo plano, um *software* responsável por solicitar periodicamente (p.ex., a cada 60 minutos) a autenticação implícita do usuário. Este SP pode ser, por exemplo, um aplicação de registro de presença de alunos. Caso esse processo de autenticação implícita falhe, o IdP redireciona o usuário para uma autenticação explícita, essa podendo ser feita com OTP, WebAuthn, entre outros.

1.2.3. Cenário internacional de federações acadêmicas

Universidades, instituições de pesquisa e empresas estão gerando grandes quantidades de dados que precisam estar disponíveis por meio de ambientes colaborativos de pesquisa que vão além dos limites de uma única organização (BROEDER et al., 2012) e até mesmo de um único país (ATHERTON et al., 2022). O serviço *eduGAIN* possibilita que pesquisadores possam usar suas credenciais institucionais para acessar inúmeros provedores de serviços

[20]https://www.okta.com/products/adaptive-multi-factor-authentication/

[21]https://docs.microsoft.com/en-us/azure/active-directory/identity-protection/howto-identity-protection-configure-risk-policies

[22]https://www.onelogin.com/learn/what-why-adaptive-authentication

disponíveis na interfederação acadêmica.

O conceito de *e-science* pode ser definido como uma forma de colaboração global em determinadas áreas da ciência e a ciberinfraestrutura que a suporta (TAYLOR, 2001). Nessa infraestrutura baseada na Internet, recursos computacionais e processamento de alto desempenho são compartilhados nas instituições ou em provedores de serviços em nuvem. Como resultado deste cenário, tem-se um grupo sem fronteiras que atua como uma rede de pessoas e instituições conectadas que colaboram com o objetivo de resolver problemas complexos e fazer ciência. Em alguns trabalhos, este grupo é chamado de Organização Virtual (*Virtual Organizations* – VO).

A maioria dos membros das federações acadêmicas são universidades ou centros de pesquisa; no entanto, profissionais autônomos e empresas também podem colaborar com a pesquisa em *e-science* (ATHERTON, C. J. et al., 2018). Profissionais autônomos muita vezes utilizam do login social (e.g. Google, Github, Linkedin) , os quais empregam geralmente o protocolo OpenID Connect e não o SAML, comumente usado nas federações acadêmicas. Diante da diversidade de mecanismos de autenticação e autorização, é comum que pesquisadores precisem gerenciar diversas credenciais de acesso e usá-las em sistemas de controle de acesso de forma desarticulada, gerando assim uma dificuldade para colaborações em *e-science* (BASNEY et al., 2019).

De acordo com Christopher John Atherton et al. (2018), os sistemas de gestão de identidade federada não foram projetados para serem usados em ambientes abertos e dinâmicos como as OVs. Existem problemas a serem resolvidos quando um usuário precisa colaborar em um ambiente no qual mais de uma federação está envolvida. Esta colaboração não deve tratar apenas dos aspectos tecnológicos, mas também das políticas organizacionais.

Um dos principais desafios relacionados ao uso de identidades federadas em OVs são os diversos atributos exigidos pelos diferentes SPs. A decisão de acesso a um serviço de *e-science* em uma OV depende não apenas dos atributos definidos pelo IdP do usuário, mas também dos atributos definidos na própria OV (CHAGAS et al., 2019). Por exemplo, o atributo de associação que identifica um usuário de instituição é membro de uma OV específica e o atributo que identifica o seu papel na OV. A falta de um padrão amplamente suportado para expressar o pertencimento a uma OV nas federações acadêmicas e na eduGAIN estreita o potencial dos sistemas federados para pesquisa colaborativa.

O projeto, financiado pela União Europeia, sobre Autenticação e Autorização para Pesquisas Colaborativas (*Authentication and Authorisation for Research and Collaboration*[23] – AARC), foi lançado em 2015 e finalizado em dezembro de 2019, e teve como objetivo: (i) identificar os requisitos necessários em pesquisa colaborativa internacional, indo além das capacidades de acesso federado atuais; (ii) entregar um modelo de arquitetura (*AARC Blueprint Architecture*), (iii) definir um grupo de diretrizes e políticas para contribuir com a interoperabilidade no contexto de autenticação e autorização; bem como (iv) avaliar os benefícios de uma plataforma abrangente para comunidades de pesquisa, por meio de casos de uso da comunidade e pilotos de integração da infraestrutura.

A GÉANT foi uma das principais colaboradoras do projeto AARC e, com base

[23]https://aarc-project.eu/

na AARC BPA, desenvolveu o serviço eduTEAMS[24], o qual expande a eduGAIN. O eduTEAMS é oferecido como serviço somente para usuários e comunidades de pesquisa na Europa e permite que pesquisadores possam criar e gerenciar times virtuais, utilizando provedores de identidade da eduGAIN e outros provedores de identidades confiáveis. As comunidades de pesquisa podem gerenciar seus usuários, organizá-los em grupos, atribuir papéis a eles e gerenciar de forma centralizada os direitos de acesso aos recursos Serviços *Web* e nativos. eduTEAMS atende usuários vindos da indústria ou cientistas que não tenham acesso a eduGAIN, por meio de um *proxy* que integra provedores de *logins* sociais, provedor ORCID e outros provedores comerciais.

O projeto de código aberto CILogon[25], fundamentado sobre o *framework* Shibboleth[26], o COManage[27] e baseada na arquitetura AARC, consiste em uma plataforma para gestão de identidade e de acesso para permitir a pesquisa colaborativa. A plataforma provê suporte a diferentes tecnologias de autenticação, fluxos para registro de usuários, vinculação de identidade (p.ex. com ORCID) e gerenciamento de grupos (BASNEY et al., 2019).

Unity [28] é um serviço de autenticação que oferece suporte a identidades federadas (SAML, OpenID), juntamente com o gerenciamento de grupos de usuários, atributos e credenciais. Permite a integração via LDAP, OAuth2, SAML e PAM. Também é possível integrá-lo com o *middleware* de computação científica UNICORE. O Unity é um software de código aberto que pode ser instalado e operado pela OV (colaboração de pesquisa), em contraste com o modelo de software como serviço do CILogon e eduTEAMS.

Periodicamente, a REFEDS apoia atividades de grupos de trabalho com o intuito de promover o diálogo abordando problemas e objetivos específicos sobre interfederação acadêmica. Com o objetivo de identificar o valor da federação e definir recomendações para melhorias no futuro, a REFEDS criou o grupo de trabalho Federação 2.0. Este grupo seguiu um processo estruturado para reunir contribuições de uma ampla gama de fontes de informação e perspectivas individuais, a fim de revisar os estados passados e atuais e formular possíveis cenários futuros para a evolução das federações acadêmicas e da interfederação. Esses dados foram analisados e sintetizados no relatório "*Academic Interfederation into the 2030s*" para articular o valor da federação acadêmica, identificar mudanças potenciais que podem aumentar esse valor e recomendar ações que as federações podem tomar para aumentar seu valor ao longo do tempo (ATHERTON et al., 2022).

1.2.4. Empoderamento dos usuários

Na Subseção 1.2.1, foram apresentadas e discutidas diversas questões relativas ao fluxo de autenticação e autorização envolvendo os mecanismos de plataformas *web*, os quais estão em mudanças devido ao fato de o aspecto tecnológico e financeiro sobreporem aos interesses do indivíduo, incluindo direitos básicos como o direito à privacidade e proteção de dados.

[24]https://www.eduteams.org
[25]https://www.cilogon.org/
[26]https://www.shibboleth.net/
[27]https://www.incommon.org/software/comanage/
[28]https://unity-idm.eu/

O próprio conceito de privacidade e proteção de dados pessoais em diversos momentos é tomado como sinônimo de segurança da informação. Porém, embora não haja privacidade e proteção de dados sem segurança da informação, os conceitos de privacidade e proteção de dados são muito mais amplos. Barth, Ionita e Hartel (2022) destacam que contemporaneamente privacidade refere-se fundamentalmente à informação, além de que seu escopo está permanentemente ampliando devido aos avanços sociais e tecnológicos. Entre os avanços sociais estão diversas legislações sobre proteção de dados pessoais, como as leis LGPD (BRASIL, 2018) e RGPD (UNION, 2016).

O ponto comum nestas legislações é o objetivo de empoderar o usuário, destacando direitos como o da auto-determinação informativa, o qual garante ao usuário o controle sobre a emissão e utilização de dados pessoais. Para que esse direito possa ser exercido, o usuário precisa ser informado sobre quais dados serão coletados, por quais motivos, se serão compartilhados e com quem, por quanto tempo serão mantidos, como serão descartados etc. Assim, muitas vezes a porta de entrada ou o primeiro contato com o usuário, para a coleta desses dados, é pela funcionalidade de criação de conta de um sistema de GId.

Os modelos de GId apresentados em Subseção 1.1.1 possuem diversas características que os colocam em maior ou menor grau com foco em tecnologia e pouco em questões humano-políticas. O modelo baseado em silo coloca um fardo grande no usuário na questão de gerenciamento de inúmeras contas, além do fato de possibilitar que seus dados pessoais estejam espalhados por diversas bases, com mecanismos diversos de segurança, aumentando o risco de vazamento de seus dados.

Nos modelos federado e centrado no usuário, o usuário passa a ter um fardo muito menor no gerenciamento de suas credenciais de acesso, além de passar poucas vezes (número de IdPs com o qual pretende interagir) pelo processo de cadastro, onde precisa fornecer seus dados pessoais. Os termos de consentimento, apresentados ao usuário pelo IdP, quando este pretende acessar um provedor de serviço, dão ao usuário o poder para ver quais dados o provedor de serviço está solicitando ao IdP e se concorda em compartilhá-los. O fato do usuário criar uma conta em um IdP e poder utilizá-la em vários provedores de serviços passa uma impressão de portabilidade de dados, mas na verdade o gerenciamento e controle dos dados ainda é atribuição única do IdP.

O modelo descentralizado é o que almeja deixar o usuário em controle desses dados, porém de acordo com o estudo conduzido por Ostern e Cabinakova (2019), a maioria dos trabalhos na literatura discutem sobre a viabilidade técnica destes sistemas de gestão de identidade descentralizada, mas não levam em consideração os requisitos de usuários e não apresentam uma avaliação sobre a usabilidade destes sistemas de forma que possam verificar se os usuários possuem as habilidades necessárias para proteger sua privacidade. Tal tipo de preocupação é essencial, uma vez que neste modelo o usuário é responsável pelo gerenciamento de seus dados.

Embora existam outras tecnologias para a implantação de um sistema de gerenciamento de identidades descentralizadas, a tecnologia *blockchain* é a mais utilizada nas soluções que estão sendo estudadas e desenvolvidas. A tecnologia provê um livro-razão distribuído (*Distributed Legder Technology* – DLT), protegido por provas criptográficas, o qual facilita troca de informações de modo seguro e confiável entre partes que não confiam

entre si, sem a necessidade de uma autoridade central para validar as transações. No estudo Dunphy e Petitcolas (2018), com relação a soluções de GId baseadas em DLT, os autores concluíram que há uma grande suposição que os usuários estão aptos a fazer um uso efetivo e correto de chaves criptográficas, bem como compreendem intuitivamente as implicações em referenciar atributos de identidade em uma DLT.

Segundo Angulo et al. (2011), o conceito de privacidade online não é simples de compreender e muitas vezes é necessário assistir o usuário de uma maneira não intrusiva. Acquisti et al. (2017) corrobora essa questão, destacando que decisões sobre privacidade e segurança são especialmente complexas *online*, sendo um dos motivos o fato de que tecnologia e ameaças evoluem constantemente. Além disso, Acquisti et al. (2017) pontua também que raramente segurança e privacidade são os objetivos principais do usuário, tendo recursos limitados para avaliar todas as opções e consequências. Portanto, de modo geral o usuário está em uma posição assimétrica de recursos e poder e por esse motivo evangelistas e reguladores sobre privacidade e proteção de dados são unânimes ao requerer que as soluções levem isso em consideração em todo o ciclo de vida do sistema.

Resumindo, qualquer que seja o modelo de GId adotado, para que ele seja centrado no usuário, que efetivamente o empodere e que também esteja de acordo com as modernas regulamentações de proteção de dados, não basta que seja disponibilizado um longo texto de política de privacidade com um botão para o usuário clicar e indicar que *leu e concordou* com os termos.

1.2.5. Identidades de Software

Um dos novos desafios na área de autenticação está no provisionamento de identidades para serviços e componentes de software. Assim como os usuários humanos, componentes de software precisam acessar sistemas como bancos de dados ou APIs diversas. Para controlar estes acessos, um dos componentes básicos de segurança na concepção é o princípio do menor privilégio, que define que cada entidade deve ter acesso apenas às informações e aos recursos necessários para o seu propósito (CAVOUKIAN, 2009). Assim, já que atualmente é cada vez mais comum que componentes de software tenham responsabilidades específicas e mínimas (e.g., paradigma de micro-serviços), componentes devem ter identidades únicas.

Outra tendência que reforça a necessidade de identidades específicas para componentes de software é o modelo de confiança zero (ROSE et al., 2020). Este modelo define que todos os recursos de um sistema devem ter identidades e que a segurança da rede não deve ser orientada a perímetros considerados confiáveis, mas sim que cada serviço deve usar autenticação e autorização forte em todas as comunicações. Isto significa que mesmo componentes em uma mesma rede devem se autenticar em todas as comunicações. Além disso, uma arquitetura de confiança zero deve considerar o monitoramento de seus recursos e que o acesso seja determinado com base em políticas dinâmicas.

No entanto, diferentemente dos usuários humanos, os componentes de software podem ter ciclos de vida bastante dinâmicos. Em uma aplicação de grande porte, instâncias de componentes de software podem ser criadas e extintas em escalas de milhares por dia, orquestrados por sistemas como Kubernetes[29]. Consequentemente, não só as identidades

[29]https://kubernetes.io

para componentes de software precisam ser específicas e robustas como as identidades para usuários humanos, mas também precisam ser provisionadas de forma automatizada.

Atualmente, diversas alternativas estão disponíveis para a geração de identidades para componentes de software. Entre elas destacamos o Kubernetes, Google BeyondProd[30], e SPIFFE/SPIRE (FELDMAN et al., 2020). Estas identidades estão tipicamente embutidas em certificados X.509 ou tokens JWT. Os certificados X.509 são especialmente populares pois podem ser usados em diferentes tipos de aplicações e até de forma transparente para aplicações legadas, por exemplo, colocando um componente *proxy* para encapsular conexões entre aplicações legadas ou como terminadores de conexões TLS que usam tais certificados. Os *tokens* JWT, por outro lado, estão mais sujeitos a ataques de reutilização e não são suportados de forma transparente. No entanto ainda são populares quando incorporados já no processo de desenvolvimento da aplicação.

Kubernetes

Kubernetes é, atualmente, a ferramenta mais popular para orquestração de aplicações em ambientes de nuvem. Uma vez que a aplicação é especificada através de um manifesto escrito em YAML, o Kubernetes faz a criação dos recursos computacionais (volumes, contêineres, balanceadores de carga etc.) e monitora estes recursos para recriá-los em caso de falhas.

O Kubernetes tem uma API para geração de certificados de forma automatizada, conhecida como a *Certificate API*. Esta API pode ser usada em combinação com controladores Kubernetes para automatizar a emissão de certificados da seguinte maneira: (1) um usuário ou aplicação gera uma chave localmente e submete um pedido de assinatura de certificado (*Certificate Signing Request* – CSR) para esta API especificando uma entidade que assinaria o certificado; (2) um administrador pode aprovar manualmente este certificado ou pode haver um controlador associado àquela entidade de assinatura que assina o certificado; (3) quando o status do certificado estiver atualizado, o usuário ou aplicação pode resgatar o certificado assinado.

Este processo de geração de certificados é pouco utilizado por aplicações e mais utilizados para geração de certificados aprovados manualmente para acesso à API do Kubernetes ou para geração de certificados aprovados automaticamente que servirão de identidade para outros componentes do próprio Kubernetes (como o Kubelet). Uma alternativa mais utilizada para aplicações é a utilização de contas de serviço (*Service Accounts*). Seguindo esta estratégia, estas contas podem ser geradas pelo desenvolvedor ou operador, atribuídas a uma aplicação durante a escrita do manifesto e, então, se tornam acessíveis de maneira programática pelos componentes de software executando no Kubernetes. Tipicamente, todo componente executando em Kubernetes tem uma identidade de serviço padrão (denominada *default*). As identidades são montadas automaticamente para serem acessíveis pelo componente executando no *pod* e o componente pode usar esta identidade (através de um *token* JWT associado) para acessar funcionalidades da própria API. Para uso com autenticação, um serviço poderia usar a API de validação de tokens do próprio

[30]https://cloud.google.com/docs/security/beyondprod

Kubernetes (a *Token Review API*) para validar tokens apresentados pelos componentes clientes.

Tanto a solução com a *Certificate API* como a solução baseada em *Service Account* têm limitações consideráveis, como não considerar características específicas da aplicação (apenas a associação entre a *Service Account* e o *pod* no manifesto) e depender de lógica inserida na aplicação para resgatar, usar e verificar os *tokens*. Finalmente, elas seguem o modelo centralizado de gestão de identidades, apenas aplicações no mesmo *cluster* reconheceriam as identidades de software.

BeyondProd

BeyondProd é um modelo de segurança para serviços recomendado pela Google. Ele é baseado em princípios de confiança zero e é definido como uma extensão do modelo BeyondCorp (WARD; BEYER, 2014). O BeyondCorp define que a autenticação dos usuários deve ser baseada em um contexto e não apenas em credenciais ou da localização. Este contexto pode considerar características do acesso como o horário e a origem, assim como características do dispositivo que está sendo usado, como configurações de segurança e nível de atualização. Seguindo também a abordagem de confiança zero, BeyondCorp reforça que não deve haver confiança implícita (e.g., comunicações locais na rede) e, portanto, não depende de VPNs.

O BeyondProd trata os micro-serviços da forma que o BeyondCorp trata os usuários: cada serviço precisa ser autenticado com base não só em credenciais estáticas, mas também no contexto onde está sendo executado. Assim, o BeyondProd estende o modelo de segurança das identidades de software providas pelo Kubernetes e detalhadas acima.

A principal forma de implementação do BeyondProd é a utilização de conexões TLS (*Transport Layer Security*[31]), mutualmente autenticadas (ou ATLS, uma variante do TLS proposta pela Google para conexões RPC[32]). Cada componente de software deve receber uma identidade e usar esta identidade para autenticar mutualmente o serviço que está acessando e a si mesmo. A implementação destas conexões mutualmente autenticadas podem ser diretamente na aplicação ou utilizando *proxies* terminadores de TLS que encapsulam a conexão entre dois pontos dentro de uma conexão TLS mutualmente autenticada. Assim, as aplicações da ponta não precisam ter conhecimento das identidades. Um exemplo de *proxy* de código aberto é o Envoy[33]. O Envoy pode ser usado tanto isoladamente para encapsular conexões quanto como parte de malhas de serviço (gerenciada por sistemas como o Istio[34] sobre um *cluster* Kubernetes).

No caso da Google, as identidades de software que serão entregues para o *proxy* TLS são gerenciadas pelo Borg (VERMA et al., 2015), uma ferramenta interna da Google e que deu origem ao Kubernetes. Ao contrário do Kubernetes puro, que entrega identidades puramente com base em um mapeamento estático, o Borg pode usar mecanismos

[31]https://www.rfc-editor.org/rfc/rfc8446.html
[32]https://cloud.google.com/docs/security/encryption-in-transit/application-layer-transport-security
[33]https://www.envoyproxy.io/
[34]https://istio.io

como a Autorização Binária (BOB[35]). O BOB utiliza várias estratégias para garantir a confiabilidade do código, por exemplo, permitindo o condicionando do fornecimento de identidade a características como a cadeia de suprimento para os artefatos de software (usando o padrão SLSA[36]). Estas evidências de integridade devem estar embutidas em políticas de segurança e associadas a imagens assinadas dos contêineres que contém os componentes de software que receberão as identidades. Para serviços em execução na nuvem da Google, o serviço de Autorização Binária está disponível e é compatível com serviços de orquestração de aplicações como o Google Kubernetes Engine ou o Anthos Service Mesh.

SPIFFE/SPIRE

Finalmente, SPIFFE (*Secure Production Identity Framework For Everyone*) é um conjunto de padrões para identificar serviços[37]. Considerando a natureza heterogênea de componentes de software, o objetivo do SPIFFE é prover interoperabilidade para estas identidades, sendo agnóstica quanto às plataformas e tecnologias.

As identidades SPIFFE (SPIFFE ID) são implementadas como URIs (*Uniform Resource Identifiers*). Por exemplo, a identidade *spiffe://example.com/database* poderia ser associada a um servidor de banco de dados que está localizado no domínio administrativo `example.com`. O nome do componente do software (`database`) é definido no registro da identidade e pode ser tanto um nome ou uma hierarquia de nomes amigáveis para humanos (como `database/server` e `database/client`), como sequências opacas de caracteres (como um valor de *hash*).

Além do formato da identidade, outros componentes chave do SPIFFE são as definições dos formatos de identidades verificáveis, os SVIDs (*SPIFFE Verifiable IDs*), e a especificação da API para emitir ou resgatar SVIDs (conhecida como *Workload API*). Por exemplo, no momento desta escrita, dois formatos para identidades verificáveis estão definidos X.509 e JWT.

SPIFFE, assim como sua implementação de referência SPIRE (*SPIFFE Runtime Environment*), são projetos graduados da *Cloud Native Computing Foundation* (CNCF). Projetos graduados são considerados estáveis para uso em produção. O SPIRE fornece APIs que permitem o estabelecimento de confiança entre componentes de software através da atestação das propriedades destes componentes e a emissão de identidades verificáveis, os SVIDs, para os mesmos. Por exemplo, de posse de SVIDs tipo X.509, os componentes podem criar conexões TLS mutualmente autenticadas.

De forma semelhante ao BeyondCorp, ele permite que clientes e servidores legados possam integrar um ambiente de confiança zero através de *proxies* que intermedeiam todas as conexões. No caso do SPIRE (e de outras implementações que usam identidades SPIFFE, como o Istio), as identidades são providas por um processo de atestação que pode ser baseada em propriedades do ambiente ou da própria aplicação. Para o caso do

[35]https://cloud.google.com/docs/security/binary-authorization-for-borg
[36]https://slsa.dev/
[37]https://github.com/spiffe/spiffe

ambiente, por exemplo, uma identidade específica pode ser provida para o componente caso ele seja executado em uma máquina virtual que pertence a um determinado grupo de segurança em um provedor público de nuvem ou em uma máquina com verificação de integridade (ex., via um *Trusted Platform Module* – TPM). Caso ele esteja executando em outro ambiente, o mesmo componente receberia uma identidade diferente, refletindo um nível de confiança diferente.

Além da validação do ambiente, o provisionamento de identidade pode ser condicionado a propriedades da própria aplicação, como o *hash* da imagem do contêiner ou até o *hash* do binário do componente. Finalmente, identidades emitidas por um domínio (ex., `example.com`) podem ser reconhecidas em outro (ex., `example.org`) caso os seus servidores SPIRE sejam federados. Neste caso, os servidores SPIRE trocam informações sobre seus certificados e as aplicações, ao receber suas identidades, também recebem os conjuntos de certificados-raiz dos outros domínios. Assim, o SPIRE pode enquadrar-se tanto na categoria de modelo centralizado de gestão de identidade, quanto no modelo federado (veja Seção 1.1.1).

O SPIRE pode ser usado junto com outras soluções de código aberto. Em especial, integra-se com o orquestrador de serviços Kubernetes e com *proxies* terminadores de TLS como o Envoy e o Ghostunnel[38]. Alternativamente, as identidades podem ser recuperadas e manipuladas diretamente pelos componentes de software, usando SDKs disponíveis para diversas linguagens de programação[39].

1.2.6. Autorização e controle de acesso

Assim como na autenticação, a utilização de uma infraestrutura externa para autorização e controle de acesso, combinado com o modelo ABAC (veja Subseção 1.1.2), traz diversos benefícios (BAILEY; CHADWICK; LEMOS, 2014), como por exemplo, um controle de acesso de baixa granularidade e regras que se aplicam a diversos sistemas dentro de uma instituição. Tal modelo vem sendo utilizado em diversos domínios de aplicação e vem sendo adotado pelos grandes provedores de serviços de computação em nuvem.[40] Na sequência apresentamos como tais mecanismos podem ser utilizados para proteger aplicações *web* e de Internet das coisas (IoT). Discutiremos também uma preocupação envolvendo a mitigação de ameaças internas.

Utilização em aplicações *Web*

Atualmente a maioria das aplicações *web* são baseadas no estilo de arquitetura REST (FIELDING, 2000), onde uma aplicação cliente interage com um servidor por meio de uma API. Existem diversos *frameworks*, nas mais variadas linguagens de programação, que oferecem suporte para este estilo arquitetural, por exemplo, Java Spring, Python Django, JavaScript Node.JS etc. Estes *frameworks* simplificam o desenvolvimento de aplicações Web e implementam algum mecanismo de controle de acesso, normalmente baseado no

[38]https://github.com/ghostunnel/ghostunnel

[39]https://spiffe.io/docs/latest/deploying/libraries/

[40]Por exemplo, o serviço *Identity and Access Management* da Amazon e o *Azure Active Directory* da Microsoft.

modelo RBAC com algumas extensões específicas de sua respectiva tecnologia.

Frameworks como Spring, Django e Node.JS permitem ao programador inserir uma anotação no código fonte para estabelecer as permissões de acesso para as operações oferecidas pela API. Node.JS, em conjunto com o *framework* Express, oferecem o conceito de *middleware* onde regras de controle de acesso são encapsuladas em módulos JavaScript que são então adicionados como interceptadores para as diversas funções oferecidas através de uma API REST. Já o *framework* Django explora o conceito de decoradores onde as permissões são adicionadas às definições das funções junto ao código fonte, com suporte adicional a instruções *if* arbitrárias definidas pelo programador.

Tais abordagens foram amplamente estudadas na literatura e são reconhecidas hoje por apresentarem diversos problemas, tais como, a possibilidade de erros ao definir políticas mais complexas e a dificuldade em se alterar uma política de acesso. De fato, os benefícios no uso de uma infraestrutura de autorização externa à aplicação são bem conhecidos (HU et al., 2014), sendo, por exemplo, uma das recomendações da Google em seu guia para o design e desenvolvimento de aplicações mais robustas (ADKINS et al., 2020) e adotado atualmente pelos grandes provedores de nuvem.

As soluções disponíveis normalmente envolvem a utilização de um componente PEP (veja Subseção 1.1.2) para interceptar as chamadas à API. Por exemplo, Silva, Medeiros e Sampaio (2019) desenvolveram PEP4Django, um *middleware* para o *framework* Django que atua como PEP de acordo com o padrão XACML. A implementação com integração ao servidor de autorização *WSO2 Identity service* está disponível como código-aberto no GitHub[41].

Outras soluções exploram o padrão de *API gateway* para onde as requisições a uma determinada API são encaminhadas. Esta é a abordagem normalmente oferecida pelos provedores de serviço de nuvem. Por exemplo, o serviço IAM da Amazon oferece uma implementação do modelo ABAC que pode ser utilizado para proteger acessos a outros serviços de nuvem ou a APIs REST através do serviço API Gateway que atua como um PEP.

Lidando com ameaças internas

Embora os mecanismos de controle de acesso sejam bastante efetivos para proteção de recursos, eles não são suficientes para detectar ameaças internas. Uma ameaça é considerada interna quando usuários autorizados a acessar um sistema abusam de suas permissões para comprometer o sigilo, a integridade ou a disponibilidade dos recursos de uma organização (CAPPELLI; MOORE; TRZECIAK, 2012). Estes usuários podem causar danos de maneira intencional ou devido a erros por falta de treinamento ou negligência.

Em geral os mecanismos de controle de acesso não são capazes de detectar usuários que abusam de suas permissões e, mesmo quando ferramentas externas são empregadas para detectar tais situações, os mecanismos de controle de acesso não são capazes de mitigá-las (SALEM; HERSHKOP; STOLFO, 2008; HOMOLIAK et al., 2019). Esta é uma situação que exige mecanismos de controle de acesso dinâmicos, no sentido que políticas

[41] https://github.com/welkson/PEP4Django

de acesso possam ser dinamicamente modificadas em respostas a ameaças internas. É importante não confundir a modificação dinâmica de políticas de acesso com a tomada de decisão de uma política baseado em informações em tempo de execução.

Alguns trabalhos têm focado neste problema nos últimos anos. Por exemplo, Bailey, Chadwick e Lemos (2014) apresentaram um *framework* para suportar a reconfiguração dinâmica de políticas de controle de acesso. Já Silva, Diniz et al. (2018) aplicam esses conceitos para detectar abusos por usuários autorizados em plataformas de nuvem Openstack, realizando uma análise do impacto de se efetivar determinadas mudanças como auxílio ao processo de tomada de decisão para responder a uma ameaça interna.

Outra abordagem para tal foi apresentada por Silva, Silva et al. (2017). O SARBAC explora *logs* de processos de negócio implementados por um sistema para construir um modelo que capture o comportamento de seus usuários. Com isso, modelos probabilísticos são utilizados para se comparar o comportamento de um usuário com o restante dos usuários de um sistema. Através do cálculo do intervalo de confiança dos modelos sendo comparados é possível afirmar que o comportamento de um usuário é estatisticamente diferente dos outros usuários, o que indica uma possível anomalia que é respondida de acordo com uma política de adaptação como, por exemplo, remover o papel do usuário em questão ou remover uma determinada permissão de um papel.

1.3. Considerações finais

As questões apresentadas neste capítulo se referem à busca de soluções para um velho problema: provar sua identidade *online*, garantindo acesso aos recursos ou serviços de maneira segura e somente ao que se tem autorização, através do estabelecimento de uma relação de confiança entre as partes. Este problema é profundamente enraizado, pois conforme as palavras de Kim Cameron, Chefe de Arquitetura de Identidade da Microsoft de 2004 à 2019, a Internet foi construída sem uma camada de identidade (CAMERON, 2005) – não há como saber com quem ou o que a conexão está sendo estabelecida. Saber apenas o endereço IP do dispositivo (o qual pode ser forjado) não informa nada em termos de qual entidade está acessando o serviço.

Questões como se a entidade tem os atributos mínimos como idade, por exemplo, precisam ser tratados de outra maneira. E essa maneira historicamente tem sido a construção de diversos modelos de GId, combinando tecnologias diversas de autenticação e autorização. A constante transformação digital da sociedade mantém esse antigo problema ainda não resolvido em evidência e busca por novas soluções, como a proposta da Identidade Autossoberana, a qual busca colocar o ser humano no centro do controle de sua identidade digital, representando um modelo de GId totalmente descentralizado. Entretanto, ainda é difícil dizer se o usuário está preparado para (e/ou disposto a) assumir a grande responsabilidade e os custos de gerir seus dados pessoais de forma totalmente autônoma e soberana.

As ações recentes realizadas pela FIDO Alliance e W3C, que permitem o uso de telefones inteligentes como autenticadores externos e a proposta de sincronismo de chaves por múltiplos dispositivos FIDO, também chamado de *passkey*, foram feitas com o intuito de acabar com a dependência do par usuário e senha pelos usuários. Porém, como observado no estudo conduzido por Owens et al. (2021), a ampla adoção WebAuthn ainda

dependerá da percepção relativa dos usuários sobre a usabilidade *versus* segurança, quando comparado com as tradicionais senhas. Usuários estão habituados com senhas, mesmo que façam um mau uso destas. Assim tem-se uma resistência ou dificuldade implícita para aprender uma nova forma de autenticar-se nos serviços, uma vez que o benefício do WebAuthn não é algo que possa ser facilmente observado por usuários leigos.

Cabarcos, Krupitzer e Becker (2019) e Ryu et al. (2021) apresentam revisões sistemáticas da literatura sobre autenticação adaptativa. No primeiro estudo, os autores afirmam que os sistemas analisados são difíceis de estender ou reutilizar (por exemplo, incluir novos autenticadores, estratégias de adaptação ou contextos) e concluem que estes são desafios significativos para o uso prático de soluções de autenticação dinâmica. Na segunda revisão, que aborda especificamente autenticação biométrica multimodal contínua, os autores concluem que muitos sistemas não avaliam adequadamente a usabilidade (aceitação e satisfação do usuários) e a viabilidade das soluções (avaliação extensiva com dados reais sem restrições), requisitos-chave para o sucesso e implantação real da solução.

Com relação à identidades de software, a combinação de padrões e boas práticas permitem ter identidades mais fortemente vinculadas a propriedades do componente de software e a limitar o impacto de vazamentos de identidades. Consequentemente, torna-se mais fácil ter ambientes com máquinas confiáveis e que executam código de origem conhecida. Tais garantias são especialmente úteis para componentes que acessam dados confidenciais. Além disso, a utilização de ferramentas que aplicam conceitos como atestação e que emitem identidades de forma automatizadas permite que esta confiança não dependa de mudanças no código ou de atividades repetitivas de operadores. As identidades podem então ser a cola que permitirá que mecanismos de autenticação e autorização isolem serviços do ambiente e, ao mesmo tempo, integrem componentes de uma mesma aplicação de forma segura e transparente.

Referências

ACQUISTI, Alessandro et al. Nudges for privacy and security: Understanding and assisting users' choices online. **ACM Computing Surveys (CSUR)**, ACM New York, NY, USA, v. 50, n. 3, p. 1–41, 2017.

ADKINS, Heather et al. **Building secure and reliable systems: best practices for designing, implementing, and maintaining systems**. First edition. Beijing [China] ; Boston [MA]: O'Reilly, 2020. 519 p. ISBN 9781492083122. Disponível em: <https://sre.google/books/building-secure-reliable-systems/>.

ALLAN, Ant. **Hype Cycle for Identity and Access Management Technologies**. Gartner, jul. 2020. Disponível em: <https://www.gartner.com/en/documents/3987655/hype-cycle-for-identity-and-access-management-technologi>. Acesso em: 5 mai. 2021.

ALLEN, Christopher (Ed.). **The Path to Self-Sovereign Identity**. Abr. 2016. Disponível em: <http://www.lifewithalacrity.com/2016/04/the-path-to-self-soverereign-identity.html>. Acesso em: 14 mai. 2021.

ANGULO, Julio et al. Towards Usable Privacy Policy Display & Management-The PrimeLife Approach. In: HAISA. 2011. p. 108–118.

ANSI. **Role Based Access Control**. 2004. ANSI/INCITS 359-2004.

ANSI. **Role Based Access Control - Policy-Enhanced**. 2012. ANSI/INCITS 494-2012.

APPLE. **Apple, Google, and Microsoft commit to expanded support for FIDO standard to accelerate availability of passwordless sign-ins**. Mai. 2022. Disponível em: <https://www.apple.com/newsroom/2022/05/apple-google-and-microsoft-commit-to-expanded-support-for-fido-standard/>. Acesso em: 3 ago. 2022.

ATHERTON, Christopher John et al. **Federated Identity Management for Research Collaborations**. Jun. 2018. DOI: 10.5281/zenodo.1307551. Disponível em: <https://doi.org/10.5281/zenodo.1307551>.

ATHERTON et al. **Academic Interfederation into the 2030s**. Zenodo, mai. 2022. DOI: 10.5281/zenodo.6584587. Disponível em: <https://doi.org/10.5281/zenodo.6584587>.

BAILEY, Christopher; CHADWICK, David W.; LEMOS, Rogério de. Self-adaptive federated authorization infrastructures. **Journal of Computer and System Sciences**, v. 80, n. 5, p. 935–952, 2014. ISSN 0022-0000. DOI: 10.1016/j.jcss.2014.02.003. Disponível em: <http://www.sciencedirect.com/science/article/pii/S0022000014000154>.

BARTH, A. **HTTP State Management Mechanism**. Abr. 2011. Disponível em: <http://www.rfc-editor.org/rfc/rfc6265.txt>.

BARTH, Susanne; IONITA, Dan; HARTEL, Pieter. Understanding Online Privacy—A Systematic Review of Privacy Visualizations and Privacy by Design Guidelines. **ACM Comput. Surv.**, Association for Computing Machinery, New York, NY, USA, v. 55, n. 3, fev. 2022. ISSN 0360-0300. DOI: 10.1145/3502288. Disponível em: <https://doi.org/10.1145/3502288>.

BASNEY, Jim et al. CILogon: Enabling federated identity and access management for scientific collaborations. English (US). **Proceedings of Science**, Sissa Medialab Srl, v. 351, 2019. ISSN 1824-8039. DOI: 10.22323/1.351.0031.

BRADLEY, John et al. (Ed.). **Client to Authenticator Protocol (CTAP)**. Jun. 2021.

BRAMHALL, Susan et al. **CAS Protocol 3.0 Specification**. Dez. 2017. Disponível em: <https://apereo.github.io/cas/6.5.x/protocol/CAS-Protocol-Specification.html>. Acesso em: 3 ago. 2022.

BRASIL. Lei nº 13.709, de 14 de agosto de 2018. **Diário Oficial da República Federativa do Brasil**, Brasília, DF, 14 ago. 2018. Disponível em: <http://www.planalto.gov.br/ccivil_03/_ato2015-2018/2018/lei/l13709.htm>. Acesso em: 28 mai. 2021.

BROEDER, Daan et al. **Federated Identity Management for Research Collaborations**. Geneva, abr. 2012. CERN-OPEN-2012-006. Disponível em: <https://cds.cern.ch/record/1442597>.

CABARCOS, Patricia; KRUPITZER, Christian; BECKER, Christian. A Survey on Adaptive Authentication. **ACM Computing Surveys**, v. 52, p. 1–30, set. 2019. DOI: 10.1145/3336117.

CAMERON, Kim. **The Laws of Identity**. 2005. Disponível em: <https://www.identityblog.com/stories/2005/05/13/TheLawsOfIdentity.pdf>. Acesso em: 28 fev. 2022.

CAPPELLI, Dawn; MOORE, Andrew; TRZECIAK, Randall. **The CERT guide to insider threats: how to prevent, detect, and respond to information technology crimes (theft, sabotage, fraud)**. Upper Saddle River, NJ: Addison-Wesley, 2012. 389 p. (The Sei series in software engineering). OCLC: ocn752067994. ISBN 9780321812575.

CAVOUKIAN, Ann. **Privacy by Design: The 7 Foundational Principles**. Ago. 2009. Information and Privacy Commissioner of Ontario. Disponível em: <https://www.ipc.on.ca/wp-content/uploads/Resources/7foundationalprinciples.pdf>. Acesso em: 3 ago. 2022.

CHAGAS, M. et al. SM4VO: A Security Management Mechanism for Virtual Organizations. **2019 9th Latin-American Symposium on Dependable Computing (LADC)**, p. 1–10, 2019. DOI: 10.1109/LADC48089.2019.8995732.

DASGUPTA, Dipankar; ROY, Arunava; NAG, Abhijit. Multi-Factor Authentication. In: ADVANCES in User Authentication. Cham: Springer International Publishing, 2017. p. 185–233. ISBN 978-3-319-58808-7. DOI: 10.1007/978-3-319-58808-7_5. Disponível em: <https://doi.org/10.1007/978-3-319-58808-7_5>.

DUNPHY, Paul; PETITCOLAS, Fabien AP. A first look at identity management schemes on the blockchain. **IEEE security & privacy**, IEEE, v. 16, n. 4, p. 20–29, 2018.

FALCÃO, Eduardo et al. Autenticando aplicações nativas da nuvem com identidades SPIFFE. In: MINICURSOS XXI Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. 2021. p. 100–144.

FELDMAN, Daniel et al. **Solving the Bottom Turtle: a SPIFFE way to establish trust in your infrastructure via universal identity**. 2020. ISBN 978-0-578-77737-5. Disponível em: <thebottomturtle.io>.

FERRAIOLO, David F.; KUHN, D. Richard. Future directions in role-based access control. In: THE FIRST ACM WORKSHOP. **Proceedings of the first ACM Workshop on Role-based access control - RBAC '95**. Gaithersburg, Maryland, United States: ACM Press, 1996. 8–es. ISBN 9780897917599. DOI: 10.1145/270152.270165. Disponível em: <http://portal.acm.org/citation.cfm?doid=270152.270165>. Acesso em: 12 ago. 2022.

FIDO. **How FIDO Addresses a Full Range of Use Cases**. Mar. 2022. Disponível em: <https://media.fidoalliance.org/wp-content/uploads/2022/03/How-FIDO-Addresses-a-Full-Range-of-Use-Cases-March24.pdf>. Acesso em: 3 ago. 2022.

FIELDING, R.; NOTTINGHAM, M.; RESCHKE, J. **HTTP Semantics**. Jun. 2022. Disponível em: <http://www.rfc-editor.org/rfc/rfc9110.txt>.

FIELDING, Roy Thomas. **REST: Architectural Styles and the Design of Network-based Software Architectures**. 2000. Doctoral dissertation – University of California, Irvine. Disponível em: <http://www.ics.uci.edu/~fielding/pubs/dissertation/top.htm>.

FORUM, World Economic. **A Blueprint for Digital Identity**. 2016. Disponível em: <https://www3.weforum.org/docs/WEF_A_Blueprint_for_Digital_Identity.pdf>.

GERADIN, Damien; KATSIFIS, Dimitrios; KARANIKIOTI, Theano. **Google as a de facto privacy regulator: Analyzing Chrome's removal of third-party cookies from an antitrust perspective**. TILEC Discussion Paper No. DP2020-034, 2020.

GOODIN, Dan. **Lapsus$ and SolarWinds hackers both use the same old trick to bypass MFA**. Disponível em: <https://arstechnica.com/information-technology/2022/03/lapsus-and-solar-winds-hackers-both-use-the-same-old-trick-to-bypass-mfa/>. Acesso em: 3 ago. 2022.

GRASSI, Paul; GARCIA, Michael; FENTON, James. **NIST Special Publication 800-63-3 Digital Identity Guidelines**. 2020. https://doi.org/10.6028/NIST.SP.800-63-3.

GRIMES, Roger. **12+ ways to hack multi-factor authentication**. KnowBe4, 2019. Disponível em: <https://info.knowbe4.com/12-way-to-hack-two-factor-authentication>. Acesso em: 3 ago. 2022.

GROUP, World Bank. **Principles on identification for sustainable development: toward the digital age - First Edition (English)**. 2018.

GRÜNER, A. et al. A Comparative Analysis of Trust Requirements in Decentralized Identity Management. **Advances in Intelligent Systems and Computing**, v. 926, p. 200–213, 2020. ISBN: 9783030150310 Publisher: Springer Verlag. ISSN 21945357. DOI: 10.1007/978-3-030-15032-7_18.

GUIDORIZZI, Richard P. Security: Active Authentication. **IT Professional**, v. 15, n. 4, p. 4–7, 2013. DOI: 10.1109/MITP.2013.73.

HARDT, D. **The OAuth 2.0 Authorization Framework**. Out. 2012. http://www.rfc-editor.org/rfc/rfc6749.txt. Disponível em: <http://www.rfc-editor.org/rfc/rfc6749.txt>. Acesso em: 3 ago. 2022.

HAYASHI, Eiji et al. CASA: Context-Aware Scalable Authentication. In: PROCEEDINGS of the Ninth Symposium on Usable Privacy and Security. Newcastle, United Kingdom: Association for Computing Machinery, 2013. (SOUPS '13). ISBN 9781450323192. DOI: 10.1145/2501604.2501607. Disponível em: <https://doi.org/10.1145/2501604.2501607>.

HODGES, Jeff et al. (Ed.). **Web Authentication: An API for accessing Public Key Credentials Level 2**. Abr. 2021.

HOMOLIAK, Ivan et al. Insight Into Insiders and IT: A Survey of Insider Threat Taxonomies, Analysis, Modeling, and Countermeasures. **ACM Computing Surveys**, v. 52, n. 2, 30:1–30:40, 2 abr. 2019. ISSN 0360-0300. DOI: 10.1145/3303771. Disponível em: <https://doi.org/10.1145/3303771>. Acesso em: 9 jun. 2022.

HU, Vincent C. et al. **Guide to Attribute Based Access Control (ABAC) Definition and Considerations**. Jan. 2014. DOI: 10.6028/NIST.SP.800-162. Disponível em: <https://nvlpubs.nist.gov/nistpubs/SpecialPublications/NIST.SP.800-162.pdf>. Acesso em: 12 ago. 2022.

IBM. **Cost of a Data Breach Report 2022**. 2022. Disponível em: <https://www.ibm.com/security/data-breach>.

IEEE Standard for a Software Quality Metrics Methodology. **IEEE Std 1061-1998**, 1998. DOI: 10.1109/IEEESTD.1998.243394.

ITU. **NGN identity management framework**. International Telecommunication Union (ITU), 2009. Recommendation Y.2720. Disponível em: <https://www.itu.int/rec/T-REC-Y.2720-200901-I>.

JAKOBSSON, Markus et al. Implicit authentication for mobile devices. In: USENIX ASSOCIATION. PROCEEDINGS of the 4th USENIX conference on Hot topics in security. 2009. v. 1, p. 25–27.

JONES, M.; BRADLEY, J.; SAKIMURA, N. (Ed.). **Introduction to JSON Web Tokens**. Disponível em: <https://jwt.io/introduction/>. Acesso em: 2 jun. 2020.

JØSANG, Audun; ZOMAI, Muhammed Al; SURIADI, Suriadi. Usability and privacy in identity management architectures. In: AUSTRALIAN COMPUTER SOCIETY, INC. PROCEEDINGS of the fifth Australasian symposium on ACSW frontiers-Volume 68. 2007. p. 143–152. Disponível em: <http://dl.acm.org/citation.cfm?id=1274548>.

LAWSON, Bruce et al. **HTML 5.3**. Jan. 2021. Disponível em: <https://www.w3.org/TR/html53/>. Acesso em: 11 ago. 2022.

LEACH, Paul J.; MEALLING, Michael; SALZ, Rich. **A Universally Unique IDentifier (UUID) URN Namespace**. Jul. 2005. Disponível em: <https://tools.ietf.org/html/rfc4122>.

LÓPEZ, M Allende. **Self-sovereign identity: The future of identity: Self-sovereignity, digital wallets, and blockchain**. v. 10. 2020. p. 0002635. Disponível em: <https://publications.iadb.org/publications/english/document/Self-Sovereign-Identity-The-Future-of-Identity-Self-Sovereignity-Digital-Wallets-and-Blockchain.pdf>. Acesso em: 6 ago. 2022.

M'RAIHI, D. et al. **TOTP: Time-Based One-Time Password Algorithm**. Mai. 2011. Disponível em: <http://www.rfc-editor.org/rfc/rfc6238.txt>.

MACHANI, Salah et al. (Ed.). **FIDO UAF Architectural Overview**. Out. 2020.

MAQBALI, Fatma Al; MITCHELL, Chris J. **Web password recovery — a necessary evil?** arXiv, 2018. DOI: 10.48550/ARXIV.1801.06730. Disponível em: <https://arxiv.org/abs/1801.06730>.

MELLO, Emerson Ribeiro de; CHAVES, Shirlei Aparecida. O uso do endereço de email pelos sites mais acessados pelo público brasileiro e os possíveis impactos na privacidade de seus usuários. In: XI Computer on the Beach. 2020.

MEREWOOD, Rowan. **Cookies SameSite explicados**. Mai. 2020. Disponível em: <https://web.dev/samesite-cookies-explained/>. Acesso em: 3 ago. 2022.

MICROSOFT. **From cookie theft to BEC: Attackers use AiTM phishing sites as entry point to further financial fraud**. Jul. 2022. Disponível em: <https://www.microsoft.com/security/blog/2022/07/12/from-cookie-theft-to-bec-attackers-use-aitm-phishing-sites-as-entry-point-to-further-financial-fraud>. Acesso em: 3 ago. 2022.

NAKAMURA, Emili Tissato et al. Identidade Digital Descentralizada: conceitos, aplicações, iniciativas, plataforma de desenvolvimento e implementação de caso de uso. In: MINICURSO - SBSeg 2019 - Petrópolis - RJ. 2019. Disponível em: <https://sol.sbc.org.br/livros/index.php/sbc/catalog/view/85/372/636-1>.

NIST. **Digital Identity Guidelines**. National Institute of Standards e Technology, jun. 2017. NIST Special Publication 800-63-3. DOI: https://doi.org/10.6028/NIST.SP.800-63-3.

_____. **Digital Identity Guidelines: Authentication and Lifecycle Management**. National Institute of Standards e Technology, jun. 2017. NIST Special Publication 800-63B. DOI: https://doi.org/10.6028/NIST.SP.800-63b.

NOTTINGHAM, M. **URI Design and Ownership**. Jun. 2020. Disponível em: <https://www.rfc-editor.org/rfc/rfc8820>.

OPEN, OASIS. **OASIS Security Services (SAML) TC**. Disponível em: <https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=security>. Acesso em: 8 ago. 2022.

OPENID. **Specifiations**. Disponível em: <https://openid.net/developers/specs/>. Acesso em: 8 ago. 2022.

OSTERN, Nadine; CABINAKOVA, Johana. Pre-prototype testing: empirical insights on the expected usefulness of decentralized identity management systems. In: PROCEEDINGS of the 52nd Hawaii International Conference on System Sciences. 2019.

OWENS, Kentrell et al. User Perceptions of the Usability and Security of Smartphones as FIDO2 Roaming Authenticators. In: SEVENTEENTH Symposium on Usable Privacy and Security (SOUPS 2021). USENIX Association, ago. 2021. p. 57–76. ISBN 978-1-939133-25-0. Disponível em: <https://www.usenix.org/conference/soups2021/presentation/owens>. Acesso em: 9 ago. 2022.

PHILLIPS, Tricia. **Hype Cycle for Identity and Access Management Technologies**. Gartner, jul. 2021. Disponível em: <https://www.gartner.com/en/documents/4004062>. Acesso em: 3 ago. 2022.

PREUKSCHAT, Alex; REED, Drummond. **Self-sovereign identity**. Manning Publications, 2021.

REESE, Ken et al. A Usability Study of Five Two-Factor Authentication Methods. In: FIFTEENTH Symposium on Usable Privacy and Security (SOUPS 2019). Santa Clara, CA: USENIX Association, ago. 2019. p. 357–370. ISBN 978-1-939133-05-2. Disponível em: <https://www.usenix.org/conference/soups2019/presentation/reese>.

RISSANEN, Erik (Ed.). **eXtensible Access Control Markup Language (XACML) Version 3.0 Plus Errata 01**. Jul. 2017. OASIS Standard incorporating Approved Errata. Disponível em: <http://docs.oasis-open.org/xacml/3.0/xacml-3.0-core-spec-en.html>.

ROSE, Scott et al. **Zero Trust Architecture**. Gaithersburg, MD, 2020. DOI: 10.6028/NIST.SP.800-207. Disponível em: <https://tsapps.nist.gov/publication/get_pdf.cfm?pub_id=930420>.

RYU, Riseul et al. Continuous Multimodal Biometric Authentication Schemes: A Systematic Review. **IEEE Access**, v. 9, p. 34541–34557, 2021. DOI: 10.1109/ACCESS.2021.3061589.

SALEM, Malek Ben; HERSHKOP, Shlomo; STOLFO, Salvatore J. A Survey of Insider Attack Detection Research. In: STOLFO, Salvatore J. et al. (Ed.). **Insider Attack and Cyber Security**. Boston, MA: Springer US, 2008. v. 39. p. 69–90. DOI: 10.1007/978-0-387-77322-3_5. Disponível em: <http://link.springer.com/10.1007/978-0-387-77322-3_5>. Acesso em: 12 ago. 2022.

SCHAAR, Peter. Privacy by Design. **Identity in the Information Society**, v. 3, n. 2, p. 267–274, 2010. DOI: 10.1007/s12394-010-0055-x.

SEAMLESSACCESS. **SeamlessAccess enables true Single Sign-On**. Disponível em: <https://seamlessaccess.org>. Acesso em: 5 mai. 2021.

SERASA. **Pesquisa Global de Identidade e Fraude 2021**. 2021. Disponível em: <https://www.serasaexperian.com.br/images-cms/wp-content/uploads/2021/06/Pesquisa-Global-de-Identidade-e-Fraude-2021.pdf>.

SILVA, Carlos Eduardo da; DINIZ, Thomás et al. Self-adaptive authorisation in OpenStack cloud platform. **Journal of Internet Services and Applications**, v. 9, n. 1, p. 19, 16 set. 2018. ISSN 1869-0238. DOI: 10.1186/s13174-018-0090-7. Disponível em: <https://doi.org/10.1186/s13174-018-0090-7>.

SILVA, Carlos Eduardo da; MEDEIROS, Welkson Renny de; SAMPAIO, Silvio Costa. PEP4Django - a policy enforcement point for python web applications. In: IX workshop de gestão de identidades digitais (WGID). São Paulo, SP, Brazil, 2019.

SILVA, Carlos Eduardo da; SILVA, José Diego Saraiva da et al. Self-adaptive role-based access control for business processes. In: PROCEEDINGS of the 12th international symposium on software engineering for adaptive and self-managing systems (SEAMS2017). Buenos Aires, Argentina: IEEE Press, 2017. (SEAMS 2017), p. 193–203. ISBN 978-1-5386-1550-8. DOI: 10.1109/SEAMS.2017.13. Disponível em: <https://dl.acm.org/citation.cfm?id=3105537>.

SRINIVAS, S. et al. (Ed.). **Universal 2nd Factor (U2F) Overview**. Abr. 2017. Disponível em: <https://fidoalliance.org/specs/fido-u2f-v1.2-ps-20170411/fido-u2f-overview-v1.2-ps-20170411.pdf>.

TAYLOR, John. News from the e-Science Programme, first phase. **Social Science Information**, RCUK website, v. 47, n. 2, p. 131–157, 2001.

UNION, European. Regulamento (UE) 2016/679 do Parlamento Europeu e do Conselho. **Jornal Oficial da União Europeia**, European Union, 4 mai. 2016. Disponível em: <https://eur-lex.europa.eu/legal-content/PT/TXT/?uri=uriserv%5C%3AOJ.L_.2016.119.01.0001.01.POR>.

UTZ, Christine et al. (Un)Informed Consent: Studying GDPR Consent Notices in the Field. In: PROCEEDINGS of the 2019 ACM SIGSAC Conference on Computer and Communications Security. London, United Kingdom: Association for Computing Machinery, 2019. (CCS '19), p. 973–990. ISBN 9781450367479. DOI: 10.1145/3319535.3354212.

VERMA, Abhishek et al. Large-scale cluster management at Google with Borg. In: PROCEEDINGS of the Tenth European Conference on Computer Systems. 2015. p. 1–17.

W3C. **A JSON-based Serialization for Linked Data**. Jul. 2020. Disponível em: <https://www.w3.org/TR/json-ld/>. Acesso em: 23 jul. 2022.

_____. **Decentralized Identifiers (DIDs) v1.0**. 2022. Disponível em: <https://www.w3.org/TR/did-core/>. Acesso em: 21 jul. 2022.

_____. **Verifiable Credentials Data Model v1.1**. 2022. Disponível em: <https://www.w3.org/TR/vc-data-model/>. Acesso em: 23 jul. 2022.

WANGHAM, Michelle S; DOMENECH, Marlon C; MELLO, Emerson R de. Infraestruturas de Autenticação e de Autorização para Internet das Coisas. In: MINICURSOS do XIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais - SBSeg 2013. 2013.

WANGHAM, Michelle Silva; MELLO, Emerson Ribeiro de et al. Gerenciamento de identidades federadas. In: MINICURSOS X Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais. 2010. p. 1–52.

WARD, Rory; BEYER, Betsy. BeyondCorp: A new approach to enterprise security, 2014. https://storage.googleapis.com/pub-tools-public-publication-data/pdf/43231.pdf.

WIEFLING, Stephan; DÜRMUTH, Markus; LO IACONO, Luigi. More Than Just Good Passwords? A Study on Usability and Security Perceptions of Risk-Based Authentication. In: ANNUAL Computer Security Applications Conference. Austin, USA: Association for Computing Machinery, 2020. (ACSAC '20), p. 203–218. ISBN 9781450388580. DOI: 10.1145/3427228.3427243. Disponível em: <https://doi.org/10.1145/3427228.3427243>.

WILANDER, John. **Intelligent Tracking Prevention 2.2**. Abr. 2019. Disponível em: <https://webkit.org/blog/8828/intelligent-tracking-prevention-2-2/>. Acesso em: 3 ago. 2022.

WU, Cong et al. ICAuth: Implicit and Continuous Authentication When the Screen Is Awake. In: ICC 2019 - 2019 IEEE International Conference on Communications (ICC). 2019. p. 1–6. DOI: 10.1109/ICC.2019.8761435.

YADRON, Danny. **Man Behind the First Computer Password: It's Become a Nightmare**. Edição: The Wall Street Journal. Mai. 2014. Disponível em: <https://www.wsj.com/articles/BL-DGB-35227>. Acesso em: 22 jul. 2022.

Capítulo 2

Provendo Segurança em Cidades Inteligentes: Aplicações, Desafios e Tendências em Mobilidade Elétrica e Tarifação Inteligente com NFTs

Paulo Mann (UFF), Guilherme Scofano (UFF), Yona Lopes (UFF), Helio N. Cunha Neto (UFF), Diogo M. F. Mattos (UFF) e Natalia C. Fernandes (UFF)

Abstract

This theoretical-practical short course presents the principles, applications, main challenges, and trends in electric mobility and smart charging with NFT (Non-Fungible Token). In the theoretical part of the short course, the main challenges and motivations related to electric mobility and dynamic pricing for the use of blockchain and NFT are addressed. The main concepts about NFT and its commercial applications, such as games, health, artistic productions, and certificates, are described. Specific applications in electric mobility and dynamic pricing, such as carbon credit management, are detailed, highlighting how NFTs allow the management of device identities, the generation of billing tokens, and the recording of activities in a secure, private, and legal way. The main research challenges in this area, such as anonymization and the high number of transactions, will be discussed in detail. Finally, to bring practical knowledge on the subject, we present a hands-on for issuing NFTs with the participants of the short course.

Resumo

Este minicurso teórico-prático apresenta os princípios, as aplicações e os principais desafios e tendências em mobilidade elétrica e tarifação inteligente com token não-fungível (Non-Fungible Token - NFT). Na parte teórica do minicurso, são abordados os principais desafios relacionados à mobilidade elétrica e tarifação dinâmica e as motivações para o uso de blockchain e NFT neste contexto. Os principais conceitos sobre NFT e suas aplicações comerciais, como jogos, saúde, produções artísticas e certificados, são descritos. São detalhadas aplicações específicas em mobilidade elétrica e tarifação dinâmica, como a gestão de crédito carbono, destacando como os NFTs permitem a gestão de identidades de dispositivos, a geração de tokens de cobrança e o registro de atividades de forma segura, privada e em consonância com a LGPD. Os principais desafios de pesquisa nessa área, como a anonimização e o número elevado de transações serão discutidos em detalhes. Por fim, visando trazer conhecimento prático sobre o assunto, é apresentado um hands-on sobre emissão de NFTs para ser realizado com os participantes do minicurso.

Este capítulo foi realizado com recursos do CNPq, CAPES, FAPERJ e FAPESP.

2.1. Introdução

A transição energética envolve mudanças que vão desde a geração de energia limpa até seu consumo consciente e eficiente. A transformação digital do sistema elétrico envolve a modernização do sistema, impulsionando a criação de novas aplicações com potencial para melhorar a confiabilidade, eficiência, flexibilidade e controle de recursos energéticos cada vez mais distribuídos e renováveis.

Nesse cenário, são introduzidas quatro forças motrizes da indústria de energia para a transformação do sistema, conhecidos como os "quatro Ds", descarbonização, descentralização, digitalização e desregulamentação (ou democratização). Os avanços tecnológicos em sistemas de geração e armazenamento de energia, internet das coisas, inteligência artificial, dentre outros, prometem acelerar a transição energética nesses quatro eixos. A mobilidade elétrica e a tarifação inteligente são elementos essenciais neste cenário e os pilares de destaque para o desenvolvimento de soluções nestes eixos são a garantia de segurança, interoperabilidade, singularidade e proveniência.

- A **descarbonização** é o elemento chave para limitar o aquecimento global. A redução das emissões de gases de efeito estufa da sociedade, principalmente dióxido de carbono, é uma estratégia real para desacelerar as mudanças climáticas. Uma vez que não se pode parar ou reverter o aquecimento global, tem-se que criar estratégias para retardá-lo. Nesse sentido, quanto mais rápido a energia for estruturalmente descarbonizada, melhor. Em um primeiro momento, a ampla utilização de Fonte de Energia Renovável (FER) é o que se aborda quando se trata de descarbonizar o setor de energia. O crescimento constante da utilização de fontes de energia renováveis permite que um recorde seja alcançado, chegando a um quarto da geração total de energia global segundo a *International Energy Agency* (IEA)[2]. No entanto, apenas esse crescimento não é suficiente, sendo primordiais a adoção de tecnologias associadas às FERs, como o armazenamento de bateria e veículos elétricos, e medidas para remover ou compensar o carbono inerente presente em nossa infraestrutura, como o crédito carbono. Nesse contexto, espera-se que a ampla adoção de Veículos Elétricos (VE) leve a uma importante eletrificação do transporte. Soluções de crédito carbono com a implementação destes modais de transporte com baixo impacto ambiental quando alinhados a serviços de economia compartilhada trazem ganhos e mudanças de paradigma importantes no setor de mobilidade.

- A **descentralização** da energia refere-se a produção de energia mais eficiente, flexível e resiliente por meio de Geração Distribuída (GD) e armazenamento em uma arquitetura de rede descentralizada. Esse modelo pode ser viabilizado através de microrredes locais autônomas, usinas elétricas locais limpas, como fazendas solares. A descentralização do fornecimento de energia de uma região traz muitos benefícios. A implantação de usinas solares locais, pequenos parques eólicos, armazenamento de baterias e usinas combinadas de calor e energia tornam o mercado de energia mais competitivo, reduzindo os preços para o consumidor. A utilização de geração distribuída para apoio a mobilidade elétrica por exemplo, pode trazer redução de custos ao se alimentar os postos de recarga de VEs com energia limpa. Sis-

[2]Disponível em https://www.iea.org/fuels-and-technologies/renewables

temas renováveis descentralizados não são apenas melhores para o meio ambiente, mas também tendem a ser mais confiáveis do ponto de vista elétrico. Problemas que ocorrem no sistema local permanecem locais, de modo que as falhas podem ser reparadas mais rapidamente resultando um fornecimento de energia mais confiável. Ressalta-se que a redução das emissões de gases de efeito estufa também envolve uma produção de energia mais eficiente, flexível e resiliente.

- A **digitalização** do setor de energia envolve o aumento do uso de tecnologia para melhor controlar, gerenciar e proteger o sistema de energia. Uma indústria de energia descentralizada exige que os dados sobre a produção, transmissão, distribuição e o consumo de energia sejam disponibilizados em tempo real e em vários pontos da rede de maneira acessível e confiável. Para viabilizar a digitalização, iniciativas como as da norma IEC 61850 [IEC TC 57, 2022], visam padronizar todo o sistema de automação de energia garantindo interoperabilidade e reusabilidade. Para isso, cada elemento/função do sistema é representado logicamente — através dos chamados nós lógicos — de forma única com um modelo de dados padronizado. Efetivamente, para que um sistema elétrico digitalizado, como uma microrrede de VEs, seja implementado, a garantia de interoperabilidade entre seus elementos é essencial. Afinal, para averiguar que as inciativas de mudança climática implementadas estão funcionando, é necessário que a quantidade de energia utilizada seja medida, e que a quantidade de carbono que esta sendo economizado possa ser calculada.

- A **democratização** é a ação de tornar o setor de energia mais justo para os consumidores. Em um mercado de energia regulado, o governo tem controle sobre os preços da eletricidade, o que deixa pouco espaço para concorrência e pouca escolha para os clientes. No entanto, com o aumento da demanda global de energia, em algumas jurisdições, novos atores podem participar do mercado de energia. Neste cenário, novamente a mobilidade elétrica sustentável aparece como solução para democratizar o setor e impulsionar soluções de tarifação inteligente descentralizadas e seguras.

Existe uma sobreposição entre os conceitos dos quatro Ds, principalmente quando se observa aplicações em mobilidade elétrica e tarifação inteligente. A mobilidade elétrica engloba a eletrificação dos veículos, toda a infraestrutura de recarga necessária para que estes sejam abastecidos e as iniciativas propostas para solucionar problemas da mobilidade urbana, como o compartilhamento de VEs. No conceito de mobilidade elétrica a infraestrutura de carregamento é externa, onde os postos de recarga podem ser associados a geração distribuída. Para que esta infraestrutura seja implementa é necessário um sistema de tarifação inteligente, seguro e eficiente que inclui a gestão da energia gerada, a forma com que é comercializada e a relação do usuário com a tarifação. Sendo assim, a mobilidade elétrica e a tarifação inteligente estão fortemente presentes nas quatro forças motrizes da industria de energia para transformação do sistema, sendo grande foco da indústria e do governo atualmente.

Essas tecnologias prometem mudar disruptivamente o mercado de energia tradicional, seja em nível de infraestrutura, aplicações ou serviços. O potencial advindo dessas tecnologias disruptivas, no entanto, introduzem novas implicações e desafios de segurança

cibernética para os operadores de serviços públicos e de mercado. Sendo assim, melhorar a segurança, a disponibilidade e principalmente a integridade dos dados da rede elétrica e é imprescindível para atender aos requisitos emergentes dos sistemas elétricos modernos, oferecendo operações seguras e confiáveis.

A maioria das soluções existentes em mobilidade elétrica [Ruggieri et al., 2021] se concentra no manuseio e compartilhamento seguro dos dados. Desafios como o uso de dados pessoais sem o consentimento ou conhecimento do proprietário até o acesso ou manipulação de dados por partes não autorizadas são recorrentes.

Neste minicurso, a tecnologia de cadeia de blocos (*blockchain*) é apresentada como uma opção para prover segurança em mobilidade elétrica e tarifação inteligente, pois permite a construção de soluções transparentes e descentralizadas de forma segura quando combinada a contratos inteligentes. A cadeia de blocos garante que transações sejam executadas de forma autônoma.

Visando garantir a singularidade, surgiram os *Non-Fungible Token*s (NFTs), que são registros/direitos negociáveis de ativos digitais únicos, tais como imagens, músicas, vídeos, entre diversos outros, onde a propriedade é registrada em contratos inteligentes em uma cadeia de blocos [Dowling, 2022]. Portanto, funcionam como certificados exclusivos de autenticidade registrados na cadeia de blocos e, usualmente, emitidos pelos criadores dos ativos, que podem ser de natureza digital ou física [Ante, 2022]. Por suas características como unicidade, interoperabilidade e também por sua rastreabilidade, que permite que o histórico completo de transações seja mantido, o NFT tem sido apontado como opção revolucionária em novos domínios, apesar de ter sua origem e destaque no mercado de ativos digitais relacionados à arte e aos jogos. O NFT permite o registro de proveniência, com características sobre o ativo que está sendo registrado e vendido, tais como origem, data de origem, tecnologia utilizada, entre outros. Tais características permitem solucionar desafios de segurança relacionados às áreas de controle, de mercado e tecnológicos de diversas aplicações dos sistemas de energia e de cidades inteligentes.

O objetivo deste minicurso teórico-prático é apresentar os princípios, as aplicações e os principais desafios e tendências em mobilidade elétrica e tarificação inteligente com NFTs. Na parte teórica do minicurso, serão abordados os principais desafios relacionados à mobilidade elétrica e tarifação dinâmica e às motivações para o uso das cadeias de blocos e NFTs em redes elétricas inteligentes. Os principais conceitos sobre NFTs e suas aplicações comerciais como jogos, saúde, produções artísticas e certificados serão descritos. Serão detalhadas aplicações específicas em mobilidade elétrica e tarifação dinâmica, como a gestão de crédito carbono, destacando como os NFTs permitem a gestão de identidades de dispositivos, a geração de *tokens* de cobrança e o registro de atividades de forma segura, privada e em consonância com a Lei Geral de Proteção de Dados Pessoais (LGPD). Os principais desafios de pesquisa nessa área, como a anonimização e o número elevado de transações serão discutidos em detalhes. Por fim, visando trazer conhecimento prático sobre o assunto, será realizada um *hands-on* de emissão de NFTs com os participantes do minicurso.

Espera-se que ao final do minicurso os participantes sejam capazes de (i) conhecer os principais fundamentos de NFT e cadeias de blocos, (ii) conhecer as principais aplicações comerciais de NFTs, (iii) conhecer as possíveis aplicações de NFTs para mobilidade

elétrica e tarifação inteligente, (iv) compreender os principais desafios de pesquisa e tendências no assunto, (v) aprender, de forma prática, como é realizada a emissão de um NFT.

O restante desse minicurso está organizado como descrito a seguir. Na Seção 2.2, são apresentados os principais conceitos sobre cadeias de blocos, enquanto que a Seção 2.3 conceitua os NFTs. As aplicações tradicionais de NFT são discutidas na Seção 2.4, enquanto que as novas aplicações em mobilidade elétrica e tarifação são discutidas na Seção 2.5. Na sequência, a Seção 2.6 traz os desafios e tendências de pesquisa. Por fim, a Seção 2.7 apresenta uma exemplificação prática da criação de NFTs, enquanto que a Seção 2.8 traz as considerações finais do texto.

2.2. Conceitos gerais da tecnologia de cadeias de blocos

A cadeia de blocos é uma tecnologia de registro distribuído sem a necessidade de uma autoridade confiável. Essa tecnologia possibilita o desenvolvimento de aplicações distribuídas seguras em cenários nos quais há desconfiança mútua entre entidades [Nofer et al., 2017]. Dois elementos principais definem a tecnologia: i) a estrutura de dados distribuída e ii) a rede par-a-par formada pelos nós participantes da rede. A estrutura de dados distribuída é formada por uma sequência imutável de registros encadeados, que originam um livro-razão digital compartilhado, distribuído e descentralizado. Sendo assim, a cadeia de blocos funciona como um banco de dados distribuído seguro, permitindo o armazenamento da informação. É importante destacar que a estrutura de dados sozinha não é suficiente para garantir que haja o acordo entre todos os participantes do sistema sobre a versão mais atual da cadeia de blocos. Porém, a estrutura de dados provida pela cadeia de blocos possui papel essencial para garantir o ordenamento das transações e evitar o gasto duplo nos sistemas de pagamento distribuídos. Também não há garantia de acordo entre os participantes sobre qual é o bloco atual que deve ser inserido na cadeia de blocos.

O acordo entre os participantes do sistema de pagamento sobre a correta versão dos blocos a serem inseridos na cadeia de blocos é um desafio dos mecanismos de consenso do sistema. Mecanismos de consenso são algoritmos que permitem alcançar um acordo sobre um único dado ou sobre o estado de um sistema distribuído [Rebello et al., 2019]. O encadeamento dos registros associados aos mecanismos de consenso garante a integridade da informação e permitem que todos os participantes da rede possuam uma réplica idêntica da cadeia de blocos, criando uma visão global da informação armazenada.

A solução proposta para alcançar o consenso no sistema de pagamento da criptomoeda Bitcoin é que o primeiro participante que resolva um desafio computacional seja o responsável por adicionar o bloco por ele proposto na cadeia de blocos divulgada para todos os demais participantes. O desafio computacional corresponde à adição de um número aleatório ao bloco candidato a ser adicionado na cadeia de blocos, porém, após a adição desse número aleatório, o resumo criptográfico do bloco deve ser iniciado com um número predeterminado de zeros. Como as funções *hash* de resumo criptográfico têm um comportamento pseudo-aleatório, não é possível correlacionar os valores de entrada aos valores de saída. Assim, não é possível ter pistas de qual valor aleatório deva ser adicionado ao bloco candidato para que se possa ter a condição de saída necessária para finalizar o desafio. Dessa forma, a busca pelo número aleatório que cumpre o desafio proposto pelo

sistema é necessariamente baseada em tentativas e erros. O desafio computacional para provar que um participante realmente comprometeu seu trabalho computacional e, portanto, é quem detém o direito de adicionar um novo bloco à cadeia é chamado de Prova de Trabalho (*Proof of Work* - PoW).

É importante ressaltar que os participantes da rede não podem falsificar o quanto de poder computacional dispõem. Portanto, a proposta de consenso se baseia em um desafio computacional que envolve o cálculo consecutivo de valores *hash* através de tentativas e erros. Ademais, o poder computacional está intimamente ligado a questões físicas que impõe custos reais ao sistema, como aquisição de equipamento e custos energéticos. Logo, é necessário que haja incentivos aos participantes que comprometem seu poder de computação para adicionar blocos à cadeia de blocos. O Bitcoin implementa a política de que, se um participante pode mostrar que satisfez a condição, então realizou uma quantidade de trabalho e, portanto, tem o direito de adicionar um bloco novo à cadeia e receber uma recompensa. O trabalho realizado pelo participante é pago em criptomoedas Bitcoin. A prova de trabalho desencoraja o desperdício do recurso compartilhado. Nota-se que o desafio criptográfico não é uma proposta restrita à cadeia de blocos Bitcoin, mas antecede as criptomoedas sendo previamente usado em protocolos de autenticação [Nita et al., 2018] ou mecanismos de prevenção de spam (mensagens de e-mail não solicitadas) [Yoon et al., 2010].

A argumentação pelo uso do desafio computacional para realizar o consenso da rede é que o custo real para atacar o sistema é grande o suficiente para que o atacante que estiver disposto a atacar seja o maior prejudicado [Oliveira et al., 2020]. Contudo, diferentes sistemas de livro-razão distribuídos têm necessidades distintas em garantir o consenso sobre os ativos transacionados e, portanto, outros mecanismos de consenso também são propostos para diferentes plataformas de cadeias de blocos.

A base criptográfica da estrutura de dados da cadeia de blocos é fortemente baseada em duas premissas criptográficas: as funções de resumo criptográfico (*hash*) e o algoritmo de assinatura digital. A partir dessas premissas, desenvolvem-se os conceitos de estruturas de dados baseadas em *hash*, prova de comprometimento, *Proof of Work* (PoW) e endereçamento de participantes na cadeia de blocos. A função de resumo criptográfico tem como objetivo realizar o mapeamento de uma cadeia de bytes (*string*) de tamanho arbitrário em uma nova cadeia de bytes com um tamanho fixo e, possivelmente, menor que a cadeia de bytes original. O tamanho da *string* de saída é fixo e, para funções *hash* atuais, tal como a SHA256, tem comprimento de 256 bytes na saída, independentemente do comprimento da entrada. Já os algoritmos de criptografia assimétrica permitem criptografar uma mensagem para que apenas o destinatário específico seja capaz de lê-la, garantindo a propriedade da confidencialidade. Além disso, é possível gerar uma assinatura digital para comprovar que uma origem específica gerou aquela mensagem, garantindo as propriedades do não-repúdio e da autenticidade.

Outro conceito importante em cadeia de blocos é a rede par-a-par. A rede par-a-par é responsável por interconectar os participantes de uma cadeia de blocos; transações, que são mensagens geradas pelos participantes que contêm alguma instrução, como a transferência de ativos digitais; bloco, que é formado por um conjunto de transações; conta, que é uma entidade capaz de gerar transações; e a carteira, que são aplicações que

permitem um usuário interagir com a conta.

Não obstante o Bitcoin apresenta uma revolução tecnológica e financeira, suas limitações rapidamente emergiram. Em particular, o Bitcoin permitia apenas a troca de ativos fungíveis, e apenas de um tipo, a moeda $BTC. Com o passar do tempo, os desenvolvedores perceberam a necessidade de criar suas próprias moedas digitais — criptomoedas — que poderiam residir também dentro da cadeia de blocos, lado a lado com outras moedas. Uma das aplicações mais bem-sucedidas que implementou este conceito foi a cadeia de blocos Ethereum.

O Ethereum elaborou e consolidou uma nova primitiva para as cadeias de blocos, revolucionando todo o mercado de criptomoedas desde sua concepção. O Ethereum permitiu que *scripts* fossem escritos e executados em linguagem de programação de alto nível dentro da cadeia de blocos por meio de transações [Saingre et al., 2022]. Na prática, esses *scripts* são chamados de "contratos inteligentes", que também possuem um endereço único. O contrato é executado ao enviar uma transação que chama uma determinada função com os parâmetros corretos para o endereço do contrato [Saingre et al., 2022]. Essa capacidade de executar código na cadeia de blocos conferiu a criação de aplicações descentralizadas - *Distributed Application*s (DApps) - que podem utilizar a rede para efetuar cálculos arbitrários. O termo "contrato" também pode ser visto como um pedaço de código que implementa um algoritmo para o qual todos da rede que o utilizam estão mutuamente de acordo.

Um dos contratos inteligentes mais utilizados no Ethereum é o *Ethereum Request for Comments - 20* (ERC-20), criado em 2015[3]. Este contrato habilitou a criação de novas moedas que podem coexistir com a moeda dita nativa do Ethereum — o Ether. Essas moedas que podem ser criadas pelos usuários recebem o nome de *tokens*. O resultado disso é visto em diversas aplicações cujos *tokens* possuem alta capitalização de mercado atualmente — como o *token* $UNI do DApp Uniswap[4]. Além disso, a criação de um *token* próprio concede um maior controle sobre a emissão e destruição deste *token*, isto é, da economia do DApp. O maior controle pode estar em dissonância com o conceito de descentralização, mas isso também pode ser resolvido por meio da criação de um sistema *Decentralized Autonomous Organization* (DAO). O DAO irá representar de maneira democrática os interesses individuais dos usuários para opinar sobre quais direções o DApp deve tomar. O DAO permite que as pessoas, em posse de uma quantia de *tokens*, tenham poder sobre o futuro da aplicação. Além disso, muitas vezes, também é possível investir em um projeto apenas comprando o seu *token*, o que representaria algo similar a comprar ações de uma empresa na bolsa de valores.

O contrato ERC-20 foi importante por adotar uma padronização de alto nível para todos os *tokens* criados no Ethereum. Em princípio, qualquer pessoa poderia criar um padrão que define a governança de *tokens*. Entretanto, a ideia de que haja apenas um único padrão tem o objetivo de aumentar a interoperabilidade e facilitar o desenvolvimento de DApps. Isso, no entanto, incorre em alguns problemas, caso haja alguma limitação com o padrão adotado.

[3] https://eips.ethereum.org/EIPS/eip-20
[4] https://uniswap.org/

O cômputo das instruções de um contrato inteligente no Ethereum tem um custo associado. A taxa de transação — ou popularmente chamado de "gás" — é coletada pelos mineradores que mantêm a rede em funcionamento. Essa taxa é o incentivo financeiro que proporciona uma vantagem para os mineradores emprestarem seu poder computacional. A taxa de transação é calculada como $taxa\_transacao = custo\_gas \cdot preco\_gas$ [Saingre et al., 2022]. O *preco_gas* é definido por quem inicia a transação — em geral o usuário. O *custo_gas* é igual à soma do custo de cada instrução do contrato inteligente que é executada na função chamada do contrato [Saingre et al., 2022]. Isso significa que quanto maior o número de instruções a serem executadas maior será o custo do gás, e portanto maior a taxa de transação. O usuário tem um papel ativo na definição do *preço do gás*, pois a rede também possui um limite de transações por bloco, ou até transações por segundo. Com o maior número de pessoas utilizando a rede, ocorre um efeito de "afunilamento" ou "congestionamento", onde passam apenas algumas transações, sobretudo aqueles que tiverem um maior *preco_gas* definido pelo usuário. De maneira geral, os mineradores priorizam as transações que têm maior recompensa financeira. Em outras situações, o usuário também pode querer que a sua transação seja processada rapidamente, e portanto, quanto maior o *preco_gas* definido, maior a prioridade.

Um dos problemas das cadeias Bitcoin e Ethereum é o mecanismo de consenso, que consome muita energia. De fato, a maior parte dos mineradores utiliza *hardware* dedicado, como *Graphics Processing Unit*s (GPUs) ou *Application Specific Integrated Circuit*s (ASICs), para obter lucros significativos, aumentando ainda mais os danos ecológicos. O rápido crescimento das criptomoedas baseadas em PoW chamou a atenção da imprensa, academia e da indústria, devido às iniciativas políticas internacionais recentes para a redução de emissões de gases que causam o efeito estufa. As cadeias de bloco passaram a ser mal vistas, tanto pela alta necessidade de energia, quanto pelo lixo eletrônico que será gerado pelos altos investimentos em fazendas de mineração [Platt et al., 2021, Miraz et al., 2021]. Para superar esse problema, foi proposto um mecanismo de consenso chamado de Prova de Participação - *Proof of Stake* (PoS), que fornece a possibilidade de "validadores" participarem do consenso de uma transação com um consumo mínimo de energia. No Prova de Participação, não é necessário usar um hardware dedicado, mas apenas ter a criptomoeda nativa e um computador simples. Quanto maior a quantidade de criptomoedas, mais chance o validador tem de validar um bloco e receber uma recompensa por isso. Pelo fato da recompensa estar associada com a maior quantidade de criptomoedas que o validador possui, os validadores são estimulados a captarem usuários com o objetivo de aumentar a recompensa total. Dessa forma, os usuários podem "delegar" suas criptomoedas para um validador em troca de receberem parte das recompensas que o validador recebe. Delegar é vantajoso para usuários que não queiram ter o trabalho de gerenciar um validador. Caso o validador valide dados fraudulentos, eles podem perder parte de suas moedas e/ou recompensas — a depender da implementação. O mecanismo de consenso Prova de Participação também pode ser visto à luz de uma pessoa que investe em sua poupança: quanto mais dinheiro a pessoa guardar, maior a recompensa.

Como resultado, a tecnologia de cadeia de blocos está em constante desenvolvimento. O Bitcoin cementou e viabilizou o uso de um sistema financeiro sem a necessidade de uma autoridade confiável — como um banco. O Ethereum desenvolveu esse

conceito e popularizou a criação de aplicações totalmente descentralizadas. Atualmente, embora de difícil execução, é possível criar aplicações que sejam governadas pelos usuários, armazenadas em serviços descentralizados, e cujo futuro é sobretudo determinado pela comunidade. Por outro lado, tem-se serviços altamente centralizados em que as funcionalidades são desenvolvidas por um grupo muito seleto que irão impactar a vida de milhões de usuários.

Dentro desse contexto de rápida evolução das cadeias de bloco, surge a tecnologia dos NFTs. Antes, todos os *tokens* das cadeias de blocos eram fungíveis, de forma que não era possível garantir, com a cadeia, a unicidade e a posse de um determinado bem digital com características únicas. A Seção 2.3 aborda como foi a primeira tentativa de criar ativos não fungíveis no Bitcoin. Em seguida, é apresentado o funcionamento dos NFTs no Ethereum, sua concepção e evolução até os dias atuais.

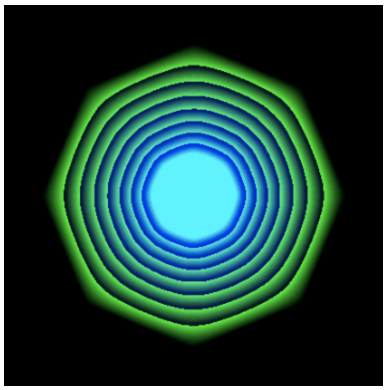
2.3. *Non Fungible Tokens* (NFT)

Um *token* não fungível (*Non-Fungible Token*) é um tipo especial de *token* criptográfico que representa um bem cuja singularidade, identidade e proveniência são cruciais [Apolinário, 2019]. Para entender esse conceito, pode-se considerar que o valor de 1 $BTC não difere de outro 1 $BTC. Igualmente, é possível trocar 1 $BTC por dois 0.5 $BTC sem prejuízos, uma vez que um bem fungível é passível de ser substituído por outro bem da mesma espécie, qualidade, quantidade e valor. Diferentemente de uma criptomoeda, um NFT não pode ser substituído por outro da mesma natureza: uma arte digital emitida por um artista nunca será igual a outra arte digital também emitida pelo mesmo artista. Ainda que o artista utilize o mesmo conteúdo digital, o NFT ainda terá outra identificação na cadeia de blocos. O NFT pode assumir qualquer tipo de conteúdo digital ou físico, como música, código, ou qualquer produção implementada como código, tal como arte generativa, jogos, páginas web ou outras.

Num cenário com quatro forças motrizes da industria de energia, temos em especial consonância a descentralização e digitalização com os NFTs. A representação digital "tokenizada" dos nós lógicos na cadeia de blocos é uma das possíveis formas de criar maior interoperabilidade no setor de energia, facilitando a tarifação inteligente. Essa seção discute a história dos NFTs, sua motivação, os padrões adotados em algumas das cadeias de blocos disponíveis, a criação e o uso de NFTs, a reusabilidade de NFTs em contextos da descentralização e o consumo de energia para emitir e negociar NFTs.

A discussão sobre a origem dos NFTs é uma área cinzenta de difícil definição. Contudo, há certo consenso que os NFTs atuais são desdobramentos da proposta *Colored Coins* [Rosenfeld et al., 2012]. A proposta das *Colored Coins* visava colorir determinadas moedas da rede bitcoin ($BTC) pelo rastreamento de sua proveniência, o que naturalmente implicaria na sua distinção, através de cores, de outras moedas da rede. As moedas coloridas eram assim definidas por terem um propósito distinto das moedas não coloridas, seja para serem usadas como moedas alternativas, certificados de *commodities* ou para representar outros instrumentos financeiros, como ações [Rosenfeld et al., 2012]. A coloração determina que a propriedade da moeda é de um bem não fungível, isto é, de identidade singular, indivisível e insubstituível. No entanto, as limitações do Bitcoin impossibilitaram o uso prático das *Colored Coins*. Na época, embora restrito apenas a uma

(a) Um *rare pepe*



(b) Um quadro de Quantum, por Kevin McCoy.

Figura 2.1. Dois dos NFTs mais antigos já emitidos.

aplicabilidade teórica, os fundamentos estabelecidos pelas *Colored Coins* deflagraram um movimento de ideias que culminou em aplicações práticas e funcionais, inclusive, no que entende-se como NFT atualmente. Por outro lado, existem pessoas que consideram o NFT Quantum — criado em maio de 2014 na rede Namecoin por Kevin MacCoy — como a origem dos NFTs. Um exemplo de quadro do NFT Quantum é mostrado na Figura 2.1b. A hipótese é que a obra Quantum foi a primeira implementação prática e funcional na rede principal (*mainnet*) pública da rede Namecoin, e portanto, poderia ser livremente negociada e oficialmente chamada de "a origem dos NFTs".

Em seguida, vários eventos se desdobraram a partir de 2015 no mundo de NFTs. A fundação do Counterparty, uma plataforma financeira distribuída cujo protocolo foi construído em cima da rede do Bitcoin, possibilitou a criação de NFTs que poderiam ser negociados livremente pela plataforma. Com efeito, dois jogos de cartas emitiram parte de seus conteúdos via NFTs: *Spells of Genesis*[5] e *Force of Will*. Ambos os jogos foram pioneiros na aplicabilidade de recursos digitais escassos emitidos via NFTs em jogos digitais. Além disso, em 2016, usuários do Counterparty começaram a criar NFTs de memes, em particular o que atualmente é conhecido como *rare pepes*, como exemplificado na Figura 2.1a, onde os usuários se especializaram em reconhecer a raridade dos *pepes* e a negociá-los livremente. Discutivelmente, esse poderia ter sido o primeiro rastro da comercialização de NFTs como colecionáveis digitais escassos. Isso demonstrou que existia não apenas uma demanda, mas também erigiu um novo nicho onde colecionadores de raridades digitais se encontravam e discutiam sobre as produções artísticas e culturais emitidas como NFTs.

No entanto, a despeito dos esforços em criar ativos que poderiam representar recursos digitais ou físicos dentro da cadeia de blocos, um problema começou a surgir por meio do excesso de heterogeneidade. A heterogeneidade é um problema porque cria um ambiente de desconfiança e de não interoperabilidade. Considere, portanto, que a em-

[5]Um *Trading Card Game* (TCG) cujos desenvolvedores se autointitulam "o primeiro jogo mobile baseado em cadeia de blocos criado" [Spells of Genesis, 2022].

presa A criou um padrão P_A de certificação digital, enquanto as empresas B e C criaram os padrões P_B e P_C. Nesse cenário, existe uma grande dificuldade de intercâmbio de produtos digitais emitidos, por exemplo, no padrão P_A para o padrão P_B ou P_C. Para haver interoperabilidade, as empresas A, B e C devem ter consenso e confiança mutua que todas as certificações digitais são válidas. Essa situação exige uma confiança e reconhecimento dos diferentes padrões adotados pelas diferentes empresas, que, no entanto, frequentemente têm dificuldade de cooperarem entre si. Com efeito, a adoção de um padrão único de emissão de NFTs foi crucial para o desenvolvimento e adoção dos NFTs, onde, às vezes, dentro de uma mesma cadeia de blocos existiam um ou mais padrões diferentes que eram usados simultaneamente.

Dessa forma, um ponto de inflexão deflagrou uma grande mudança no que hoje entende-se por NFT, iniciando-se com o padrão EIP-721[6], criado em 2017 na rede Ethereum. O EIP-721, uma implementação específica de contratos inteligentes, possibilitou a emissão e negociação de NFTs na rede Ethereum. Em seguida, os padrões EIP-1155[7] e EIP-2981[8] foram desenvolvidos com o propósito de abranger os casos de uso do EIP-721. O EIP-1155 insere um novo paradigma de uso para NFTs com a adoção do padrão *multi-tokens* e operações menos custosas — menor uso de gás — que reduzem a emissão de carbono em 90% [Valeonti et al., 2021]. Um *multi-token* permite a existência de NFTs semi-fungíveis, que implica a existência de uma classe de ativo que possui uma ou mais edições. Para todas as edições de um mesmo NFT, é possível trocá-los entre si, pois são equivalentes; mas entre classes diferentes de ativos — de *multi-token* para *multi-token* — ainda podemos considerar o ativo insubstituível.

Com os padrões EIP-721 e EIP-1155, os *royalties* são distribuídos de maneira individualizada por cada plataforma de negociação — não há uma padronização. Com efeito, o EIP-2981 cria uma maneira universal de garantir o direito financeiro dos *royalties* por meio do contrato que o NFT foi gerado. Isso garante que, a despeito de como a plataforma irá lidar com *royalties*, o criador do NFT irá receber os *royalties*. Originalmente, na ausência de um padrão que habilitava o uso de NFTs, o padrão mais adotado no Ethereum era a implementação ERC-20, que representa a emissão de moedas[9] fungíveis. Embora este padrão não seja capaz de emitir NFTs, foi uma evolução perante ao bitcoin — pois limitava-se à uma única moeda na cadeia de blocos, o $BTC.

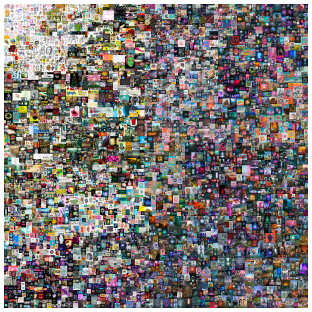
Uma das primeiras aplicações de sucesso que usou o padrão ERC-721 foi o CryptoKitties, um jogo cujo objetivo é reproduzir gatos digitais, onde os filhotes criados terão as características herdadas dos gatos pais. O jogo permitia ao jogador ter a posse dos gatos como NFT, o que, por consequência, habilitava o jogador a negociá-lo em uma plataforma de negociação, ou até mesmo trocar com seus amigos. O sucesso do jogo fez com que fosse gerado um número imenso de transações na rede Ethereum, congestionando-a e aumentando o preço necessário para processar uma transação — popularmente conhecido como gás. Embora a proposta do jogo não fosse para obter ganhos financeiros, o

[6] `https://eips.ethereum.org/EIPS/eip-721`

[7] `https://eips.ethereum.org/EIPS/eip-1155`

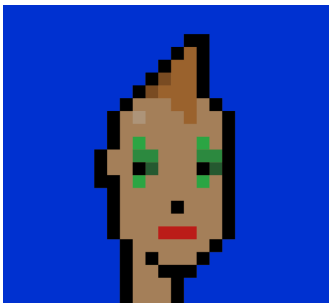
[8] `https://eips.ethereum.org/EIPS/eip-2981`

[9] Essas moedas muitas vezes recebem o nome de "*tokens*", e são moedas emitidas e controladas livremente pela pessoa — ou a organização — que os criaram, o que difere particularmente da moeda nativa do Ethereum — Ether.



(a) 5000 *Everydays*' por Beeple.

(b) Diamante destruído por Tascha Che.



(c) CryptoPunk #7610 comprado pela VISA Inc.

Figura 2.2. Em (a), um dos NFTs mais caros já vendidos. Em (b), um diamante físico que foi destruído para emitir um NFT. Em (c), um colecionável chamado CryptoPunks comprado pela VISA Inc.

mercado secundário (revenda dos gatos a outros interessados) estava super aquecido enquanto os gatos eram vendidos por altos preços; um gato chegou a ser vendido por US$ 170,000 [Valeonti et al., 2021].

Os efeitos do sucesso do ERC-721 ainda são sentidos atualmente, uma vez que cerca de 97% das vendas de NFTs ainda são realizadas no Ethereum [CoinTelegraph Research, 2021]. Com efeito, a partir de 2021, ocorreu uma sucessão de vendas de alto valor de NFTs. Em Fevereiro de 2021, ocorreu a venda de um meme popularmente conhecido como "Nyan Cat" por aproximadamente US$ 590,000 no dia da venda [Etherscan, 2021b]. Em seguida, a Christie's, uma famosa casa de leilões de arte, vendeu o NFT "5000 Everydays" do artista popularmente conhecido como Beeple por US$ 69.3 milhões [Beeple, 2021]. A arte vendida por Beeple é a união de 5000 mil artes individuais produzidas diariamente em seu perfil do Instagram por mais de treze anos, como pode ser observado pela Figura 2.2a. Em uma outra situação, a VISA Inc. comprou um NFT colecionável chamado CryptoPunks, mostrado na Figura 2.2c, por US$ 150,000 [Browne, 2021, VisaNews, 2021]. Além disso, outras utilidades inovadoras surgiram ao migrar o conteúdo físico para o digital por meio dos NFTs. Como exemplo, a Tascha Che comprou um diamante de US$ 5.000,00 e o destruiu para emitir um NFT com uma imagem e o certificado de compra do diamante [Che, 2021], mostrada na Figura 2.2b. Como resultado, o NFT que representa o diamante foi vendido por 5.5125 Ether (equivalente a US$ 18,009.28 no valor do dia da venda) no OpenSea [Etherscan, 2021a]. Em outra situação, a obra Fumeur V (1964) de Pablo Picasso foi queimada para ser emitida como um NFT [Yahoo, 2021], embora não tenha sido vendida. A obra Fumeur V original pode ser observado pela Figura 2.3a, enquanto a obra destruída (incinerada) pode ser observada pela Figura 2.3b.

Há, no entanto, uma discussão entorno do quanto um NFT consegue funcionar como uma reserva de valor em face da perda da utilidade física. O diamante possui valor pela sua utilidade física, raridade, durabilidade, mas também como reserva de valor porque existe um consenso social e financeiro que habilita o seu valor. Ao destruí-lo, a utilidade física é perdida, enquanto, em tese, a raridade, durabilidade e consenso social são transmitidos diretamente ao NFT. Porém, isso é apenas verdade enquanto o consenso

(a) Fumeur V, obra original de Pablo Picasso.

(b) Fumeur V queimado, obra original de Pablo Picasso.

Figura 2.3. Fumeur V, por Pablo Picasso em (a). A mesma obra, porém queimada em (b).

social e a confiança sustentarem o que chamamos atualmente de NFT. Mas uma coisa é inegável: é muito mais simples transferir um NFT que representa a propriedade de uma casa para outra pessoa do que passar pela burocracia para realizar a mesma tarefa sem NFTs.

Qualquer tipo de investimento ou ativo financeiro possui seus riscos inerentes, sobretudo algo tão emergente como NFTs. Entretanto, pode-se observar que todas essas aplicações inovadoras ou altos valores de vendas não somente sinalizam a adoção dos NFTs, mas também põe em pauta a presença institucional no processo de comercialização de NFTs, o que em última instância promove sua seriedade e utilidade.

2.3.1. Consumo de energia

Com o crescente número de pessoas interessadas em NFTs, particularmente inclinadas pela promessa de lucros exorbitantes, somado ao fato de Ethereum ser uma rede de alta capitalização de mercado, e portanto de maior confiabilidade, houve um crescimento vertiginoso no número de transações na rede Ethereum entre 2019–2022. Com um maior número de transações, mas com a mesma capacidade de processamento de blocos de outrora, a rede tornou-se congestionada, que por consequência fez o preço da transação (gás) aumentar em função da alta demanda. Em outras palavras, havia um número muito maior de pessoas tentando fazer transações do que a cadeia de blocos poderia suportar; a maneira, portanto, de ter sua transação processada seria aumentar o valor pago como uma forma de "suborno" aos mineradores para processar sua transação com maior prioridade na fila de transações. Isso, no entanto, fez com que o gás inviabilizasse o uso da rede para uma parcela dos usuários, já que para ter sua transação processada, frequentemente, era necessário pagar US$10–US$50, como mostrado na Figura 2.4. Pagar este valor para usuários de países subdesenvolvidos é, na maioria dos casos, uma quantia muito alta para apenas processar uma transação, causando uma grave exclusão social e econômica das tecnologias emergentes de cadeia de blocos. Além disso, como a taxa de gás é variável em função do uso da rede, o uso se torna imprevisível e portanto não confiável para determinadas aplicações.

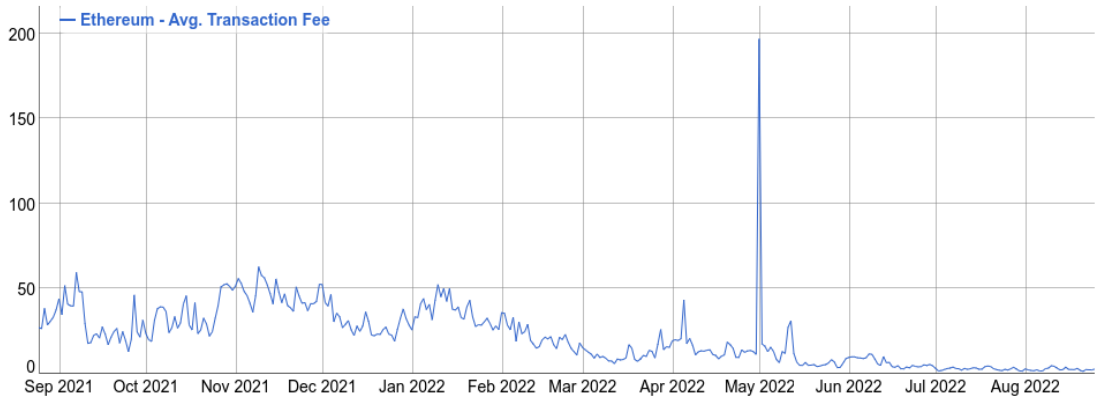


Figura 2.4. Preço de gás médio das transações do Ethereum em US$ de Setembro de 2021 até Agosto de 2022. Fonte: Bitinfocharts.

Ao mesmo tempo que o alto valor de gás dificulta o uso da rede, também propicia maiores recompensas financeiras para os mineradores. Esse maior incentivo resulta em mais pessoas participando da rede de mineração, o que em geral implica no aumento de consumo de energia da rede por meio do maior uso de GPUs — que são utilizadas para minerar criptomoedas cujas redes utilizam o consenso de Prova de Trabalho. Com o recente declínio no valor e uso das criptomoedas, houve uma queda no valor das placas de vídeo da NVIDIA de 50% no mercado secundário [Bloomberg, 2022]. Isso indica uma certa correlação do interesse das pessoas em manter suas GPUs enquanto as recompensas financeiras forem vantajosas ao minerar criptomoeda; ou vendê-las caso contrário.

Em 2018, uma pesquisa comparou o consumo de energia de mineração das redes Bitcoin e Ethereum com o consumo da mineração de metais convencionais como alumínio, cobre, ouro e platina durante o período de 2016 até 2018. Nesse estudo, foi estimado que minerar Bitcoin e Ethereum consomem em média 17 e 7 MJ respectivamente para gerar US$1, enquanto os materiais da mineração convencionais consomem em média 122, 4, 5, e 7 MJ respectivamente para gerar US$1 [Krause e Tolaymat, 2018]. O Bitcoin apenas consome menos energia do que a produção de alumínio. Por outro lado, estima-se que o Bitcoin consumiu a mesma energia que a Angola ou Panamá em 2017 [Krause e Tolaymat, 2018]. Enquanto isso, o Ethereum, que é a principal rede usada para negociação de NFTs, consumiu cerca de 2.4x menos energia do que o Bitcoin para o mesmo período. Em 26 de agosto de 2022, o índice de consumo de energia do Ethereum (*Ethereum Energy Consumption Index*), criado por Alex de Vries, aponta que o consumo de energia anualizado do Ethereum equivale ao consumo de energia do Chile [Digiconomist, 2022]. Ainda segundo o Digiconomist, a emissão de carbono de uma única transação do Ethereum equivale a 251,954 transações financeiras da VISA Inc. ou ao equivalente a 19 mil horas de vídeos assistidos no YouTube [Digiconomist, 2022]. O gráfico do consumo anualizado do Ethereum pode ser visto na Figura 2.5. Ademais, a taxa de *hash* das redes estão em tendência de alta, o que implica na necessidade cada vez maior de poder de processamento para a mineração, e portanto, maior consumo de energia com o passar do tempo [Krause e Tolaymat, 2018].

Com efeito, o problema do alto valor de gás aliado ao grande consumo de energia proporcionado pelo mecanismo de consenso Prova de Trabalho transformou o Ethereum

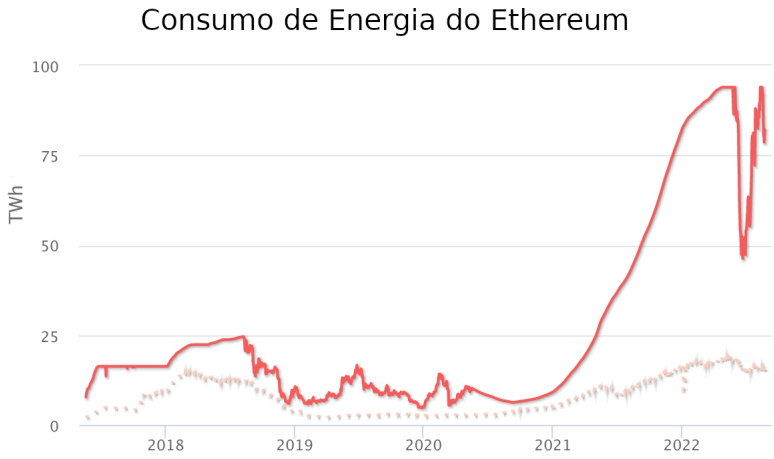


Figura 2.5. Índice de consumo de energia do Ethereum. O eixo vertical representa o consumo em TWh por ano. A linha pontilhada representa o mínimo de consumo em TWh por ano. A linha não pontilhada representa a estimativa. Adaptado de [Digiconomist, 2022].

em um grande vilão. O consumo de energia, e por consequência a emissão de carbono, é um tema que é frequentemente discutido ao falar sobre cadeias de blocos ou NFTs. O Ethereum ainda é a referência principal para emissão e negociação de NFTs. No entanto, o Ethereum, no momento de escrita deste texto, ainda é regido pelo mecanismo de consenso Prova de Trabalho. A promessa de que o Ethereum irá se tornar Prova de Participação com a atualização chamada "*The Merge*" implicaria numa redução de 99,95% do consumo de energia de toda a rede [Ethereum.org, 2022]. Mas enquanto essa solução ainda não é efetivamente implementada na rede principal do Ethereum, outras cadeias de blocos que já adotaram a Prova de Participação se movimentaram para criar seus próprios padrões de NFT, alegando serem amigáveis ao meio ambiente com o *slogan* "*green NFTs*". Este foi o caso da rede Tezos, uma cadeia de blocos que implementa a Prova de Participação, cujo padrão para NFTs é intitulado de FA2[10] e implementado como um contrato inteligente. Diferentemente do Ethereum, o FA2 é capaz de operar sobre uma miríade de tipos de *tokens*: *multi-tokens*, *tokens* fungíveis, e *tokens* não fungíveis. O consumo de energia da rede Tezos anualizado é estimado em 0.001 TWh [Tezos Foundation, 2022], tornando-se uma alternativa mais ecologicamente sustentável para o uso de NFTs com baixas taxas de transação. A rede Solana também se demonstrou eficiente em termos de uso de energia, consumindo apenas 2,707 J por transação (em comparação, uma única busca no Google consome cerca de 1,080 J [Solana, 2022]). O consumo de energia anualizado do Solana está estimado em 0.01105 TWh [Flow, 2022b]. Além disso, Solana também possui um padrão definido para *multi-tokens*, *tokens* fungíveis e não fungíveis implementados via contratos inteligentes. Uma outra rede em que as transações também possuem baixo custo e baixo consumo de energia é a rede Flow, originalmente concebida pelos mesmos criadores de CryptoKitties [Flow, 2022a]. O consumo anualizado da rede Flow está estimado em 0.00018 TWh [Flow, 2022b], ou seja, uma única busca no Google é equivalente à emissão de 12.5 NFTs no Flow.

[10]https://gitlab.com/tezos/tzip/-/blob/master/proposals/tzip-12/tzip-12.md

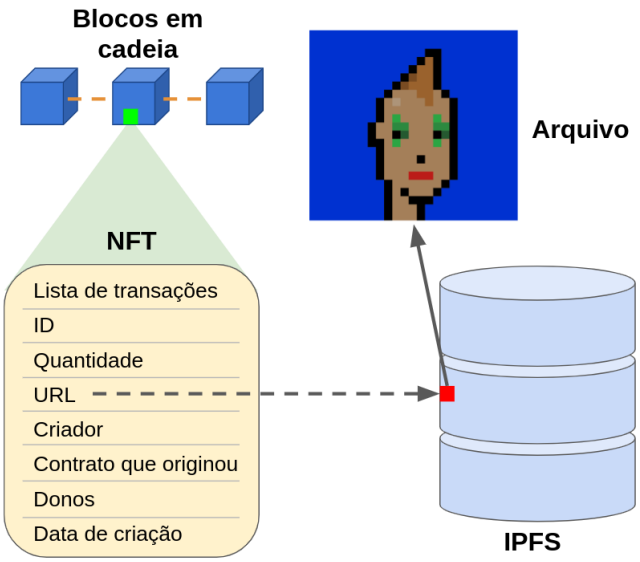


Figura 2.6. Esquema genérico do conteúdo do NFT armazenado na cadeia (url, id, quantidade, etc.) vs. o conteúdo armazenado fora da cadeia — o documento digital em si.

Embora as redes que implementam um mecanismo de consenso mais ecologicamente sustentável sejam uma boa solução para o consumo de energia e para a emissão de carbono, ainda resta um problema a ser discutido. Em linhas gerais, os NFTs armazenam metadados associados ao dado digital em questão, mas o arquivo digital não é salvo na rede de blocos. Isso ocorre em função do alto custo para armazenar até um dado muito pequeno na rede; por exemplo, no Ethereum, em julho de 2020, armazenar apenas 1 MB de dados custava mais de US$ 13,000 em taxas [Valeonti et al., 2021]. Para tanto, é necessário salvar o arquivo digital fora da cadeia de blocos em algum serviço de armazenamento, que poderia ser, por exemplo, o *InterPlanetary File System* (IPFS), ou um serviço próprio de armazenamento — frequentemente centralizado. Na prática, o NFT terá apenas uma referência (link) para o dado salvo no serviço de armazenamento escolhido. Como exemplo, alguns dos possíveis dados salvos na cadeia de blocos podem ser observados na Figura 2.6, como a URL que irá apontar para o conteúdo digital fora da cadeia; o número de edições (quantidade) que o NFT possui; o registro de todos, e quem, transferiu este NFT; os donos do NFT; a sua data de criação; o contrato cujo NFT foi emitido — aquele que rege seu comportamento; quem é o criador deste NFT, e mais outras possíveis informações a depender da cadeia de blocos ou do padrão adotado.

Embora aparentemente desprezível, o consumo de energia e a emissão de carbono associados ao uso de uma tecnologia fora da cadeia não deve ser desprezado. Para o caso de aplicações de uso prático com grandes emissões de NFTs, o armazenamento pode se tornar um grande consumidor de energia sobretudo em situações de alta escalabilidade. Além disso, o armazenamento do conteúdo digital em uma plataforma terceirizada levanta questionamentos sobre o quão descentralizado os NFTs realmente são. Por um lado, o IPFS seria a alternativa mais descentralizada, que no entanto não garante a disponibilidade do dado; por outro lado, um servidor de armazenamento pago pelo criador do NFT pode deixar de disponibilizar o dado assim que o pagamento do serviço for suspenso.

Com efeito, surgiram outras cadeias de blocos que são especializadas em armazenar os dados que, em tese, ficariam fora da cadeia original. Dessa forma, tornou-se possível armazenar o dado do NFT de uma cadeia em outra cadeia, garantindo uma maior capacidade de descentralização. O Filecoin é uma cadeia cujo propósito é fornecer uma rede de operadores que compartilham e armazenam os conteúdos requisitados pelos usuários por incentivo financeiro — a criptomoeda $FIL [Filecoin, 2022]. O Filecoin é orquestrado em cima da tecnologia do IPFS, com a exceção que no IPFS não há incentivo financeiro. Por outro lado, também existe a possibilidade de fazer cálculos computacionais fora da cadeia e acessar dados do mundo real de maneira segura por meio da cadeia Chainlink, que também oferece incentivos financeiros aos participantes [Chainlink, 2022a]. Contudo, o uso dessas cadeias em paralelo também deve ser incluído no cálculo do consumo de energia e de emissão de carbono.

2.3.2. NFTs, armazenamento e portabilidade

Uma situação particularmente interessante de casos de uso de NFTs é a reusabilidade no contexto de descentralização. O NFT emitido por uma aplicação reside na cadeia independentemente da existência da aplicação. No contexto em que a aplicação deixe de existir por motivos diversos, o NFT pode ser utilizado em outras aplicações que sejam compatíveis, ou desenvolvidas para serem compatíveis. Por outro lado, a aplicação que uma vez já existiu poderia ser recriada pela comunidade — um exemplo dos benefícios de código aberto, sobretudo em contextos de redes de blocos. Mais recentemente, o criador do *marketplace* Hic et Nunc[11] descontinuou o desenvolvimento da plataforma, delegando o código para a comunidade de desenvolvedores. A comunidade se reuniu e criou um sistema DAO para controlar a governança da plataforma de maneira livre e aberta entre os membros da comunidade. O resultado pode ser diretamente observado pelo relançamento da plataforma sob o nome Teia[12]. Embora o Hic et Nunc tenha deixado de existir por um breve período, a posse dos NFTs ainda pertenciam às pessoas que inicialmente os compraram ou emitiram. Contudo, o armazenamento da mídia digital, em geral, é controlado pela plataforma de negociação em que o artista originalmente emitiu o NFT. Isto é, a plataforma de negociação de NFTs é frequentemente responsável por pagar o serviço de armazenamento das artes criadas na mesma, que, em geral, são subsidiadas pelos *royalties* cobrados pela plataforma. No entanto, quando o desenvolvimento da plataforma é descontinuado, fica a critério dos usuários — compradores ou artistas e desenvolvedores — de manterem o conteúdo digital persistido. Isso ocorre pois a disponibilidade do conteúdo digital, que é normalmente armazenado no IPFS, depende da existência de pelo menos uma máquina na rede IPFS que armazene o conteúdo. Existem serviços pagos que garantem, com certa probabilidade, a persistência do conteúdo no IPFS. Embora o conteúdo possa não estar disponível, o IPFS garante o armazenamento da impressão digital (do inglês, *fingerprint*) do conteúdo; isso permite que o conteúdo, embora não persistido, possa eventualmente ser religado à rede, como evidenciado pela seta na Figura 2.6.

A maneira que o conteúdo digital é armazenado levanta diversos questionamentos. Em primeiro lugar, é possível criar um NFT com o mesmo conteúdo digital de um outro NFT já existente. Neste cenário, o que diferenciará um NFT do outro serão os metadados

[11] Aplicação descentralizada da rede de bloco Tezos.

[12] `https://teia.art/`

associados ao NFT, sobretudo o endereço de quem o criou, como mostrado na Figura 2.6. Um "NFT falso" terá o endereço do criador falso, que em geral aponta para o mesmo conteúdo digital armazenado no IPFS que o NFT original. Com efeito, usuários são regularmente vítimas de golpes (também conhecidos como *scam*), onde os perpetradores criam NFTs fraudulentos ao copiar as artes de artistas famosos. De maneira semelhante, os NFTs que residem em redes de blocos diferentes mas que apontam para o mesmo conteúdo digital também são outra forma de cópia fraudulenta. Ademais, é trabalho dos colecionadores (ou usuários) verificar os direitos autorais da obra que eles se propõem a negociar. Portanto, os usuários devem estar atentos à proveniência do NFT antes de negociá-lo.

Além disso, um outro problema se cria com relação à posse e propriedade do NFT. Ao alugar um apartamento, o locatário tem a posse mas não a propriedade do apartamento — de exclusividade do locador. Caso um bem seja roubado, é senso comum que a propriedade ainda seja daquele que foi roubado, embora o perpetrador tenha a posse do bem. No caso dos NFTs, não existe distinção — para a maioria das cadeias de blocos, sobretudo para o Ethereum — entre posse e propriedade. Isso implica que, ao roubar um NFT, o perpetrador tem automaticamente a posse e propriedade do NFT, o que dificulta o processo de reversão do crime.

No entanto, como já discutido anteriormente, a característica de posse e propriedade garante que o NFT possa ser utilizado por uma aplicação, ou até mesmo reutilizado em outra aplicação. Ao passo que esta portabilidade constitui um potencial de interoperabilidade, ela também é de difícil execução. Por exemplo, o modelo 3D de um personagem confere uma série de dificuldades para proporcionar a portabilidade. Uma simples mudança de motor de jogos que leia os eixos X, Y ou Z diferentemente de quem emitiu o NFT já resulta em incompatibilidade. Podemos ainda mencionar as texturas e *shaders* que irão depender dos formatos das entradas, saídas, parâmetros e *buffers*, o que dificulta ainda mais a portabilidade. Por outro lado, existem projetos inovadores que propõem o uso de um conteúdo simples e genérico que pode ser utilizado e projetado de diferentes formas em diferentes aplicações (ou jogos). Nesse cenário, Loot[13] propõe o uso de NFTs que representam equipamentos para aventureiros em um jogo — qualquer jogo que implemente as diretrizes do NFT. O projeto propositalmente omite quaisquer outras informações com a intenção de promover a criação de jogos que façam implementações individualizadas e inovadoras com poucas diretrizes, como apenas fornecer os nomes dos equipamentos. Cabe destaque, que, dentro do contexto de aplicações de energia e de cidades inteligentes, muito pouco foi discutido sobre interoperabilidade dos NFTs entre diferentes plataformas, muito embora esse seja um dos problemas em aberto para pesquisa.

Embora os NFTs forneçam uma série de benefícios, ainda pode-se dizer que é uma tecnologia incipiente. Como todo avanço tecnológico na história da humanidade, o produto pode ser utilizado de maneira eticamente questionável. Com NFTs, isso não é diferente. Contudo, num contexto de colaboração mundial, independente e de código aberto, acredita-se que o desenvolvimento da tecnologia de criptomoedas e NFTs andará a passos largos para evitar os diversos problemas existentes atualmente. As tecnologias de bancos e comércios digitais também tiveram seu período de fricção em suas primeiras

[13] `https://www.lootproject.com/`

versões, cujos problemas de segurança geraram centenas de fraudes. Ainda assim, os benefícios de tais tecnologias eram tão evidentes que a segurança da informação evoluiu para assegurar o uso dessas aplicações. Assim, é forte a possibilidade que a evolução da tecnologia de cadeias de blocos pode levar a uma adoção em massa da tecnologia. Nesse contexto, a próxima seção mostra como aplicações comerciais estão gerando receita e demanda a partir de ideias inovadoras.

2.4. Aplicações comerciais de destaque baseadas em NFT

Os NFTs ganharam muita atenção das pessoas, em particular da indústria financeira, entre 2020 e 2021, sobretudo com vendas de artes digitais como NFTs de alto valor. Diferentemente, outras formas de NFT também podem ser comercializadas, como música digital (Doja Cat), conteúdo de jogos digitais (CryptoKitties, Axie Infinity), ou ainda para bem-estar, como um aplicativo cujo objetivo é encorajar as pessoas a correr com incentivo financeiro (STEPN). Essa seção demonstra como a identificação inequívoca de um bem não fungível na cadeia de blocos pode ser utilizada em diversos cenários da indústria para fomentar a descentralização e prosperar aplicações da chamada Web 3.0.

2.4.1. Aplicações em jogos digitais

O NFT possui grande potencial para a indústria de jogos digitais. Atualmente existem cripto jogos como Axie Infinity[14], Stepn[15], [16], Gods Unchained[17], TradeStars[18] e Decentraland[19]. Nesses jogos, o jogador pode colecionar, negociar e fazer reprodução de criaturas digitais.

No jogo Axie Infinity, o jogador coleta criaturas chamadas de Axies. Com essas criaturas, o jogador pode realizar batalhas, criar e construir reinos para os Axies. Além disso, o jogador pode reproduzir seus Axies e criar espécies. Semelhantes ao Axie Infinity, no jogo CryptoKitties, o jogador compra um gato disponível no catálogo e esse gato será guardado em sua carteira. A comercialização dos gatos no jogo é feita através da criptomoeda Ethereum e alguma plataforma de negociação que opere com o EIP-721. Em posse dos gatos, o jogador pode reproduzir, selecionando dois de seus gatos, ou utilizar um genitor público para a reprodução. A Figura 2.7 mostra alguns CryptoKitties colecionáveis.

Já o jogo Gods Unchained é um jogo de cartas tático, onde o jogador tem a propriedade de seus itens do jogo, possibilitando a negociação desses itens em plataformas de negociação de NFT. O jogo possui sua própria criptomoeda chamada $GODS para negociação de itens.

A proposta do Stepn é manter as pessoas em movimento. Foi o primeiro projeto que implementou com sucesso o conceito de movimente-se para ganhar. Os jogadores que caminham ou correm ao ar livre, ganham a criptomoeda do jogo, chamada de $GMT,

[14]https://axieinfinity.com/
[15]https://stepn.com/
[16]https://www.cryptokitties.co/
[17]https://godsunchained.com/
[18]https://tradestars.app/
[19]https://decentraland.org/

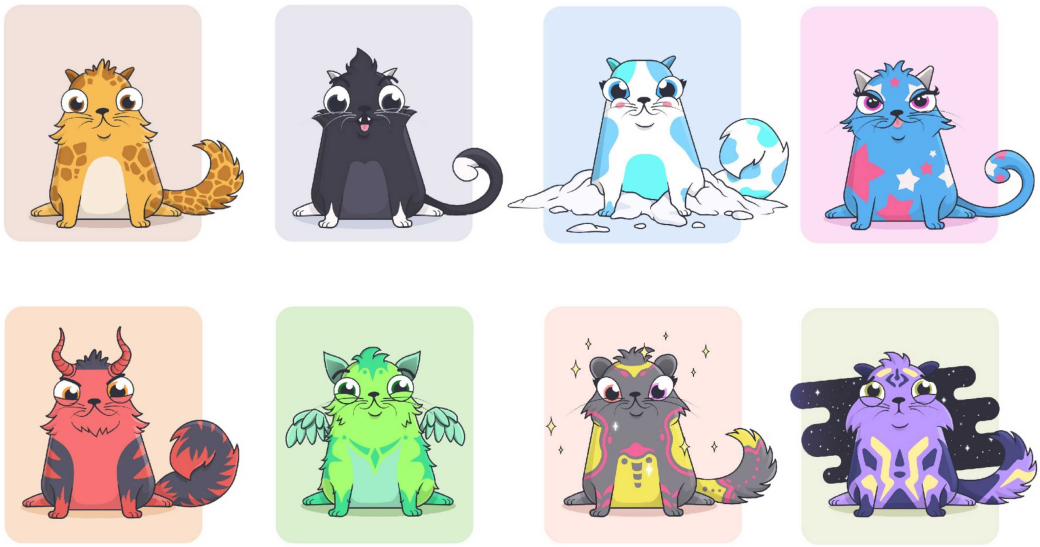


Figura 2.7. Em CryptoKitties é possível procriar novas raças de gatos.

que pode ser usada no próprio jogo ou vendida para obter lucro em moedas fiduciárias. A moeda do jogo é utilizada como mecanismo de controle da própria economia do jogo. Para participar, é preciso comprar um tênis — que é um NFT — apto a registrar seus passos, o qual é vendido pelos criadores.

O TradeStart é jogo social descentralizado onde o conhecimento dos jogadores sobre esporte é avaliado. O jogo utiliza estatísticas reais do mundo do esporte. A plataforma do jogo não é controlada por uma organização individual. A plataforma foi construída utilizando cadeia de blocos, logo, nenhum agente do jogo pode modificar as regras do jogo, os valores dos ativos ou impedir que um jogador acesse a plataforma.

O Decentraland é um jogo estilo Sandbox, onde os jogadores são livres para modificar e explorar o mundo virtual. O jogo é governado por uma organização autônoma descentralizada e foi desenvolvido na cadeia de blocos do Ethereum. O diferencial do Decentraland é que os jogadores podem ser "donos" de suas terras. O jogo também possui sua própria criptomoeda chamada de $MANA. Os ativos e terras dentro do jogo são negociadas utilizando a $MANA. O Decentraland foi um dos primeiros metaversos descentralizados que foi construído, governado por meio da participação de seus usuários.

O metaverso é um espaço compartilhado virtual coletivo que permite todos os tipos de atividades digitais. Geralmente, abrange um conjunto de técnicas como a realidade aumentada e a internet para estabelecer o mundo virtual. O conceito vem das últimas décadas e tem um grande progresso com o desenvolvimento da cadeia de blocos. A cadeia de blocos fornece um ambiente descentralizado ideal para o mundo virtual online. Os participantes dessas realidades alternativas conectadas pela cadeia de blocos podem ter muitos tipos de casos de uso intrigantes, como desfrutar de jogos, exibir artes feitas por eles mesmos, negociar ativos e propriedades virtuais, como, por exemplo artes, terras, vídeos e vestimentas. Além disso, os usuários também têm oportunidades de obter lucros

com a economia virtual. Além do Decentraland, existem outros metaversos que utilizam cadeia de blocos, como, por exemplo, [20], Somnium Space[21], MegaCryptoPolis[22] e Sandbox[23]. A descentralização nesses espaços virtuais é imprescindível para que não haja um monopólio financeiro e político que domina e controla todos os participantes, como ocorre com as grandes empresas de tecnologia atualmente.

2.4.2. Aplicações em produções artísticas e culturais

Além dos jogos, existem também os cripto colecionáveis como Meebit[24] e CryptoPunks[25]. Os CryptoPunks foi classificado pela OpenSea[26] como a maior coleção de NFT de todos os tempos por volume de negociação (2,2 bilhões de dólares). Os CryptoPunks são dez mil figuras únicas em pixel art. As figuras foram geradas aleatoriamente através de um algoritmo. Por mais que alguns CryptoPunks compartilhem algumas características, nenhum compartilha todas as características de outro. Já os Meebit, são modelos 3D estilo voxel. Existem 20 mil Meebits, onde aproximadamente 11 mil foram distribuídos gratuitamente para donos de CryptoPunks.

O comércio de produções artísticas e colecionáveis pode ser potencializado com NFT. Tradicionalmente, os artistas possuem poucos canais para exibir seus trabalhos. Os preços não refletem o verdadeiro valor das obras devido à falta de recursos. Além do mais, plataformas e anúncios cobram seus trabalhos publicados nas redes sociais com taxas intermediárias. Os NFTs transformam esses trabalhos em formatos digitais com identidades integradas. Os artistas não precisam transferir a propriedade e o conteúdo para os agentes. Isso lhes dá impulso com muitos lucros. Exemplos típicos incluem o REPLICATOR de Mad Dog Jones [27] (vendido por 4,1 milhões de dólares), os trabalhos de Grimes [28] (movimentando, no total, cerca de 6 milhões de dólares) e outros como o Trevor Jones. Além disso, por meios tradicionais, os artistas em geral não recebem royalties de vendas futuras de suas obras; ou quando recebem, o depósito ocorre semestralmente, trimestralmente, ou até anualmente. Em contraste, os NFTs podem ser programados para que o artista receba uma taxa de royalties predeterminada toda vez que sua obra de arte digital for negociada. Sendo assim, uma maneira eficiente de gerenciar, proteger e negociar obras-primas digitais. Além disso, várias plataformas como Mintbase[29] e Mintable[30] estabeleceram ferramentas para apoiar pessoas comuns a criar seus próprios trabalhos NFT de modo facilitado. A Figura 2.8 mostra alguns CryptoPunks e um Meebit.

[20] https://www.voxels.com/
[21] https://somniumspace.com/
[22] https://mcp3d.com/
[23] https://www.sandbox.game/en/
[24] https://meebits.app/
[25] https://www.larvalabs.com/cryptopunks
[26] Plataforma de negociação de NFTs online sediada em Nova Iorque.
[27] https://www.phillips.com/detail/mad-dog-jone
[28] https://www.theverge.com/2021/3/1/22308075/grimes-nft-6-million-sales-nifty-gateway-warnymph
[29] https://www.mintbase.io/
[30] https://mintable.app/

(a) CryptoPunks.

(b) Meebit.

Figura 2.8. Os CryptoPunks e Meebits foram criados aleatoriamente por um algoritmo na Larva Labs.

2.4.3. Certificados e *tickets* de eventos

Eventos tradicionais contam com empresas centralizadas que são provedores do meio tecnológico. Embora a cadeia de blocos esteja presente em diversos tipos de atividades, como arrecadação monetária, suas aplicações ainda são restritas a uma pequena gama de eventos. Os NFTs estendem o escopo das aplicações de cadeia de blocos com a ajuda de suas propriedades adicionais (*e.g.*, singularidade, propriedade, liquidez). Isso permite que cada indivíduo se vincule a um evento específico, assim como os padrões em nossa vida real. Ao comprar ingressos para eventos de maneira tradicional, o consumidor deve confiar em terceiros. Portanto, existe o risco da compra de bilhetes fraudulentos ou inválidos, que podem ser falsificados ou cancelados. O mesmo ingresso pode ser vendido múltiplas vezes ou obtido através de imagens de ingressos publicados online.

Um ingresso baseado em NFT representa um ingresso emitido pela cadeia de blocos para demonstrar o direito de acesso a qualquer evento, como cultura ou esportes. Um bilhete baseado em NFT é único e escasso, que embora possa ser revendido, é possível limitar sua negociação via contratos inteligentes, caso seja assim desejado. O contrato inteligente baseado em cadeia de blocos fornece uma plataforma transparente de negociação de ingressos para as partes interessadas. Os consumidores podem comprar e vender o bilhete criptográfico do contrato inteligente em vez de depender de terceiros de maneira eficiente e confiável.

Empresas como Oveit e Seatlab comercializam ingressos baseados em NFT para eventos. A Oveit fornece também um produto chamado *Ticket Addons*, onde qualquer ticket sobre um evento pode ser centralizado em um único lugar. Por exemplo, o cliente pode comprar o *ticket* do evento, as bebidas que serão consumidas, camisetas e outro souvenir, com todos esses itens registrados como *tickets*, centralizados no aplicativo do evento. Assim, o cliente pode ir ao evento apenas com o celular, evitando perda de seus pertences.

2.4.4. Aplicações de saúde

O NFT também possui um grande potencial para aplicações da saúde [Musamih et al., 2022]. Uma aplicação potencial é o uso do NFT para cunhar informações de pacientes. Em aplicações tradicionais um paciente deixa suas informações clínicas com o hospital ou uma clínica. O hospital pode vender esses dados genéticos a terceiros para fins de pesquisa. No entanto, ao vender esses dados a empresa pode ganhar muito dinheiro que nunca será compartilhado com os donos desses dados. Além disso, à medida que esses dados confidenciais são transmitidos ao longo de uma cadeia de transações, o risco de manuseio incorreto das informações aumenta. Agora, se esses dados são cunhados como NFTs, a informação virá com uma característica inerente a ser rastreada. O paciente seria capaz de ver onde o dado foi utilizado e responsabilizar aqueles que o usaram sem sua permissão, pois é o único proprietário dos dados. Além disso, o proprietário do NFT pode habilitar um recurso para ganhar dinheiro sempre que ocorrer uma transação com os dados.

Com uma abordagem NFT, as empresas podem oferecer serviços de saúde digital incentivando os pacientes a participar de estudos contribuindo com seus dados e ganhando com eles. Outros terceiros interessados em utilizar os dados para pesquisa ou desenvolvimento de novos produtos podem entrar em contato diretamente com os pacientes em uma plataforma digital. A principal diferença em comparação com a abordagem tradicional é que os pacientes realmente têm a opção de compartilhar seus dados de maneira transparente.

Embora muito sobre NFT na área da saúde ainda seja especulativo neste momento, existem algumas empresas que estão explorando esse potencial. Uma dessas empresas é a Aimedis, que possui um mercado NFT médico onde os pacientes podem participar de transações envolvendo seus dados de saúde. O aplicativo de monitoramento de saúde Go!, desenvolvido pela Enjin e Health Hero, pode coletar dados individuais de atividades e bem-estar de aplicativos populares como Apple Health, Google Fit e Fitbit. Estes podem até ser negociados no mercado aberto.

No entanto, existem diversos potenciais obstáculos para a adoção em massa da tecnologia de NFTs, especialmente na área da saúde. Tal como está, a tecnologia NFT atualmente funciona de forma bastante ineficiente, com grandes quantidades de energia necessárias para pequenas transações. Como tal, os NFTs podem não ser comercialmente viáveis em um futuro próximo. Mas alternativas para a cunhagem de NFT estão em andamento e podem usar uma fração do poder de computação atualmente envolvido em suas transações. Outro ponto é a falta de interesse na adoção do NFT pelas empresas que oferecem serviços de saúde digital. Essas empresas podem não ser particularmente atraídos pela ideia de compartilhar seus lucros com os pacientes, de quem tradicionalmente lucram e não o contrário.

2.5. Aplicações do NFT em mobilidade elétrica e tarifação

Embora o uso mais amplamente difundido dos NFTs seja destinado a jogos e obras de arte, vislumbra-se, atualmente, que essa tecnologia tem grandes potenciais para aumentar a segurança das aplicações de energia e de cidades inteligentes baseadas em cadeias de blocos. Os NFTs auxiliam na gestão de identidade e no controle seguro e transparente

de uso de créditos. A seguir, são apresentadas algumas das principais aplicações de NFTs dentro do contexto de mobilidade elétrica e tarifação.

2.5.1. Gestão de identidade de veículos e estações de recarga

A mobilidade elétrica depende fortemente do uso deveículos elétricos e da geração distribuída [Lopes et al., 2015] em estações de recarga. Nesse sentido, para fins de criação de contratos inteligentes e registros de dados nas cadeias de blocos, há a necessidade da associação de uma identidade de forma inequívoca a um objeto dentro da cadeia. Para esse fim, vem-se utilizando os *Identity Non-Fungible Token*s (I-NFTs), que permitem a utilização de componentes criptográficos e operações protegidas para dar suporte aos ativos da rede em todas as etapas da atividade [Gourisetti et al., 2021].

Um I-NFT consiste de um registro NFT que visa armazenar dados sobre a identidade de uma pessoa, um dispositivo ou uma entidade do sistema. Esse NFT é, então, utilizado em contratos inteligentes como forma de identificar a origem do pedido. A posse da chave criptográfica associada ao NFT já é suficiente para a realização da autenticação. Um ponto interessante para o uso de NFTs para esse tipo de processo é manter as informações pessoais fora da cadeia de blocos, mas autenticáveis pelo uso da cadeia. Gourisetti *et al.* sugerem a criação de I-NFTs a cada transação, a fim de evitar que o usuário ou o dispositivo possam ser rastreados ao longo de suas atividades por quem tem acesso aos blocos da cadeia [Gourisetti et al., 2021].

Arcenegui *et al.* generalizam os I-NFTs para a autenticação de dispositivos. Os autores colocam a necessidade de associar a identidade virtual ao dispositivo de forma inequívoca. Assim, os autores propõem o uso de *Physical Unclonable Function*s (PUFs), que é um método que permite a associação difícil de quebrar entre *tokens* e dispositivos. Dessa forma, os registros e tokens relacionados ao dipositivo passam a poder ser rastreados durante a vida útil do dispositivo, permitindo, entre outros, o registro autenticado e seguro de mudanças de propriedade [Arcenegui et al., 2021]. A proposta vai além da representação do dispositivo *Internet of Things* (IoT) na cadeia de blocos, contemplando também sua interação com outros endereços na cadeia de blocos, sejam esses endereços associados a pessoas, organizações ou outros dispositivos IoT. O dispositivo IoT passa a ser capaz de receber e prover informações e assinar transações. Carros, semáforos, câmeras e quaisquer elementos que devam participar da reconstrução digital do sistema físico de forma a manter sua fidedignidade podem ser representados por seus NFTs e as interações entre esses componentes do sistema de transporte podem ser realizadas por meio de contratos inteligentes executados na cadeia de blocos.

Gao *et al.* citam algumas formas de se implementar uma PUF. Uma delas, particularmente interessante para implementação em sistemas embarcados, baseia-se no projeto de *chipsets* de memória RAM estática [Gao et al., 2020]. Essas possibilidades remontam ao conceito de NFTs inteligentes mencionados por Arcenegui *et al.* [Arcenegui et al., 2021], no qual o NFT é fisicamente interligado ao seu dispositivo IoT respectivo graças ao PUF.

Outros trabalhos também ressaltam a necessidade da identificação segura dos dispositivos na cadeia de blocos [Karger et al., 2021]. Yuan e Wang tratam diretamente a gestão de identificadores de veículos elétricos, propondo um modelo conceitual organi-

zado em camadas para *Sistema de Transporte Inteligente*s (STIs) baseados em cadeias de blocos. O modelo contém uma camada física composta por entidades tais como veículos, câmeras, semáforos e demais componentes da malha de transportes [Yuan e Wang, 2016].

Uma aplicação mais objetiva das cadeias de blocos em sistemas de transportes pode ser encontrada em [Zhang e Wang, 2019]. Nesse trabalho, os autores trazem conceitos de *Vehicular Ad Hoc Network*s (VANETs) e as associam a cadeias de blocos com o intuito de realizar o controle inteligente do tempo de luz verde dos semáforos a partir das condições de trânsito. Para tal, é necessário que os veículos sejam equipados com *Unidade de Bordo*s (UBs) e que as estradas tenham *Unidade de Acostamento*s (UAs) espalhadas. As UBs, as UAs e o departamento de trânsito ingressam na cadeia de blocos. As UAs se comunicam com as UBs dos veículos e registram dados criptografados relativos às condições de trânsito na cadeia de blocos. O departamento de trânsito, por sua vez, recolhe, descriptografa e analisa as informações de tráfego registradas na cadeia de blocos e dispara a execução de contratos inteligentes que regulam a duração do tempo de luz verde dos semáforos de forma dinâmica. A representação dos componentes do sistema de trânsito como NFTs na cadeia de blocos também se faz coerente se aplicada neste trabalho. O autor de [Zhang e Wang, 2019] propõe, ainda, que o departamento de trânsito conceda recompensas por veículos "honestos", que fornecem informações precisas quanto à sua localização, e exponha na rede informações de veículos maliciosos. Os incentivos podem se concretizar na forma de certificados digitais únicos que, por sua vez, também podem ser implementados por meio de um NFT associado ao NFT do veículo, atestando que tal veículo fornece informações corretas.

Indo além das propostas de [Zhang e Wang, 2019], é possível, ainda, que as VANETs registrem na cadeia de blocos infrações cometidas por veículos, tais como exceder um limite de velocidade ou cruzar um semáforo fechado, na forma de NFTs. O NFTs da infração, embora não seja físico como o veículo ou o semáforo, é uma indicação de que a violação foi cometida e pode ser associada a um veículo, um semáforo, uma câmera de monitoramento e até mesmo a um vídeo documentando a ocorrência.

2.5.2. Gestão segura de créditos de carbono

Outras atividades importantes dizem respeito à geração de *tokens* de **créditos de carbono** a serem negociados entre entidades que promovam a redução de emissão de gases do efeito estufa e empresas poluidoras. Para essas aplicações, é possível o uso de sistemas baseados em NFT ou ainda, híbridos, mesclando NFT e *Fungible Token* (FT) [Karandikar et al., 2021].

2.5.2.1. Créditos de carbono

O crédito de carbono foi criado dentro do contexto do Protocolo de Kyoto, visando reduzir as emissões de gases que aumentam o efeito estufa. A pegada de carbono de um indivíduo ou uma empresa é o total de gases do efeito estufa geradas por todas as suas atividades. Dentro dos protocolos internacionais contra o aquecimento global, as empresas devem reduzir ou compensar a sua pegada de carbono. As reduções e compensações são medidas como crédito de carbono. Portanto, o crédito de carbono é a moeda utilizada no

mercado de carbono, sendo equivalente a uma tonelada de dióxido de carbono ou gases equivalentes. Nesse contexto, empresas que possuem um nível de emissão de gases que causam o efeito estufa muito alto e poucas opções para a redução da emissão desses gases devem optar pela compra de créditos de carbono para compensar suas emissões, conforme mostrado na Figura 2.9.

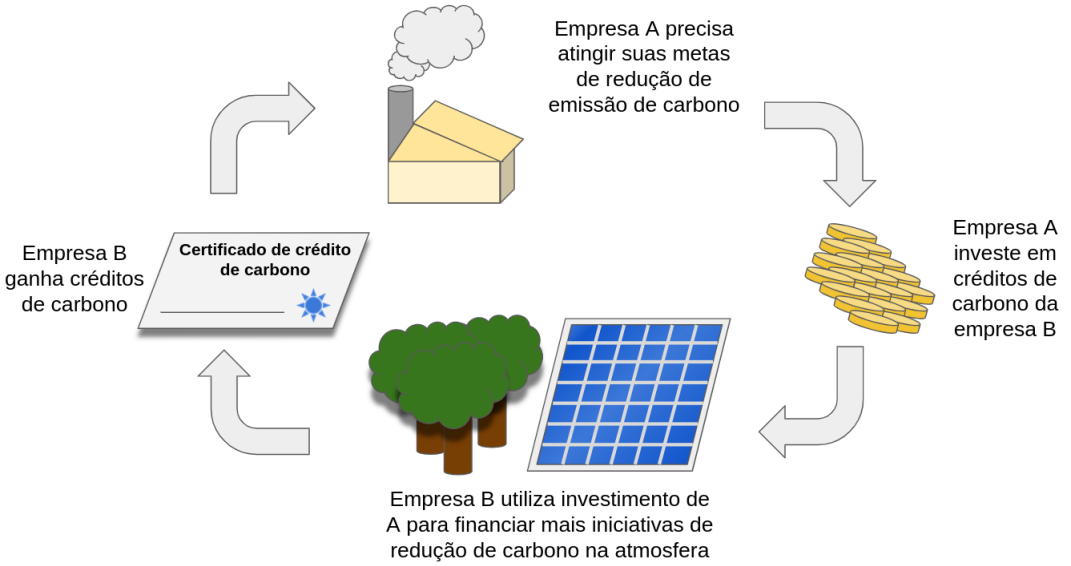


Figura 2.9. Geração e utilização dos créditos de carbono. Adaptado de [Save Planet Earth, 2021].

Existem dois tipos de crédito de carbono: redução de emissões voluntária - *Voluntary Emissions Reduction* (VER) e redução de emissões certificada - *Certified Emissions Reduction* (CER). O VER é aplicável no mercado de créditos voluntário - *Voluntary Carbon Market* (VCM), onde indivíduos ou empresas investem em projetos de redução ou sequestro de carbono de forma voluntária. O CER é emitido por meio de arcabouços regulatórios [Carbon Credits, 2021].

2.5.2.2. NFTs para créditos de carbono

Os mercados de carbono transformam as emissões de CO2 em um ativo ambiental negociável, dando-lhe um preço. Com isso, os Mercados Voluntários de Carbono - VCM - cresceram, atingindo movimentações da ordem US$ 1 bilhão em 2021, com expectativas de crescer 15 vezes até 2030 [Thomason, 2022].

Um crédito de carbono pode ser revendido várias vezes até que seja retirado pelo usuário final que deseja reivindicar o impacto da compensação. Para tanto, é necessário que os registros de créditos de carbono sejam emitidos por verificadores terceirizados, independentes e certificados internacionalmente. As cadeias de bloco e os NFTs entram como a forma de registrar a geração, as transações e o uso do crédito para compensação. Esse tipo de esquema aumenta a transparência e o monitoramento dos créditos de carbono, evita falsas alegações de eficiência energética e uso de créditos ineficazes, além de evitar a dupla contagem dos créditos de carbono devido à falta de protocolos contábeis completos

e alinhamento entre as jurisdições do mercado [Thomason, 2022]. Outra vantagem é a redução de custos em toda a cadeia.

2.5.3. Gestão segura da geração de energia

O aumento no uso de veículos elétricos está diretamente relacionado a impactos na rede elétrica, que deverá passar a suprir a energia necessária para uma atividade que outrora era suprida pela queima direta de combustíveis fósseis. Além do aumento de capacidade nas unidades de consumo e do aumento do fluxo de potência circulando pela rede elétrica para atender essa nova demanda, há de se considerar que, num contexto de popularização da GD, essa carga pode ser parcialmente atendida por uma FER. Ademais, as baterias automotivas conectadas à rede elétrica trazem a possibilidade de armazenar energia, podendo mitigar uma das principais deficiências de geradores eólicos ou fotovoltaicos, que é a intermitência, através do fornecimento *Vehicle to Grid* (V2G) [Gao et al., 2014]. Esse contexto de grandes modificações no sistema traz desafios e oportunidades que podem ser solucionados e explorados por cadeias de blocos, criptomoedas e NFTs.

Em [Li et al., 2020] o autor apresenta um gêmeo digital utilizado para estimar o estado de carga e a degradação de baterias. O gêmeo digital recebe os dados das baterias sob análise por meio de dispositivos IoT e é executado na nuvem, podendo usufruir de uma capacidade de processamento mais alta e executar algoritmos mais sofisticados para realizar suas estimativas. Em contribuições futuras, uma bateria veicular pode ser representada digitalmente por um NFT na cadeia de blocos de forma similar ao trabalho de [Arcenegui et al., 2021], bem como os dispositivos IoT físicos, e a interação entre eles se concretizaria por meio de contratos inteligentes.

Em um contexto de popularização da tecnologia V2G, as representações digitais por meio de NFTs das baterias e dos demais ativos elétricos em uma cadeia de blocos podem participar de uma reconstrução virtual do sistema de potência, garantindo ao operador maior observabilidade sobre o sistema. Dados de operação podem ser coletados e o despacho da geração pode levar em consideração a quantidade de FERs e baterias conectadas dispostas a participar do V2G. Em [Karandikar et al., 2021], o autor descreve um sistema de potência com diversos atores, cada qual com diversas capacidades, representados em uma cadeia de blocos. Tais capacidades variam entre gerenciamento pelo lado da demanda, disponibilidade de bancos de baterias, capacidade de geração a partir de fontes renováveis, dentre outros.

Um sistema de compra e venda de energia entre veículos elétricos baseado em cadeias de blocos foi proposto em [Kang et al., 2017]. Uma das inovações propostas era a figura de nós autorizados a estabelecer o livro-razão distribuído intitulados agregadores locais. A cadeia de blocos baseada nos agregadores locais era capaz de auditar e verificar registros de transações entre os veículos elétricos. Um mecanismo de leilão duplo iterativo maximiza o bem-estar comunitário otimizando o preço da energia e o estado de carga das baterias dos veículos. Neste artigo, são registrados dados, incluindo pseudônimos dos veículos, na cadeia de blocos proposta. O autor desenvolveu a moeda "energy coin" para representar digitalmente um bem utilizado para trocar energia.

Dentro desse contexto de mercado de energias renováveis controladas por cadeias de bloco, existe a tendência dos certificados de energia renovável - *Renewable Energy*

Certificate (REC), criados como *tokens*, por meio de NFTs. Um REC é um contrato único emitido por algum gerador que não emite gás carbônico, o qual é vendido a multinacionais e geradores fósseis. Na forma tradicional de gerenciar os RECs, isso demanda muitos contratos e processos complexos.

O problema associado aos RECs é a natureza dos sistemas elétricos, nos quais não é possível saber de onde a energia que está sendo consumida efetivamente veio, ou ainda, se veio de uma fonte renovável. Para sobrepor esse problema e incentivar a energia renovável, Certificados de Atributo de Energia - *Energy Attribute Certificate*s (EACs) começaram a ser emitidos, permitindo ao consumidor comprar a energia a partir do tipo de geração. Um EAC especifica um megawatt hora de energia produzida em uma determinada localização, marcando também o tipo de tecnologia e o mês de produção. Contudo, os EACs não tem a granularidade temporal ideal para demonstrar as variações na geração de energia renovável, a qual depende da força do vento ou da intensidade do sol. Assim, passou-se a considerar a necessidade de EAC que representam não apenas o mês, mas também o dia e a hora de geração. Esses EAC ou REC são, então, registrados digitalmente em um NFT [Rossi, 2022].

Um exemplo prático de NFTs para energia renovável é provido pela plataforma RESpring[31], que provê EAC com granularidade de hora para criar um mercado de energia renovável imutável e certificável. Outro exemplo é da *Restart Energy Democracy* (RED), que lançou uma plataforma descentralizada de fornecimento de energia para recompensar certificados de energia verde para consumidores de energia renovável. Essa plataforma permite o comércio direto ponto a ponto entre consumidores e fornecedores por meio de RECs em NFTs [RED, 2021]. A REX também criou a REX NFT[32], uma coleção de NFTs para REC, desenvolvida sob a plataforma Polygon, com a moeda REX Coin.

2.5.4. Tarifação dinâmica e tokens de cobrança/premiação

O registro seguro de valores para tarifação dinâmica e para crédito de carbono para diferentes tipos de atividade ao longo do tempo já é possível por meio da associação entre NFT e *Decentralized Finance* (DeFi), permitindo a desassociação entre a criptomoeda e o valor aplicado ao ativo registrado na transação. Essa nova tecnologia, ainda imatura, permite também a criação de *tokens* de cobrança basedos em NFT, que podem permitir diversos tipos de ação em plataformas de negociação de energia, como a criação de *Virtual Power Plant*s (VPPs) [Lopes et al., 2015].

Outro uso interessante é a aplicação de *tokens* para premiar usuários que participam de programas de gerenciamento pelo lado da demanda - *Demand Side Management* (DSM) [Chainlink, 2022b]. Ações de gerenciamento de energia pelo lado da demanda, permitem, entre outros, que determinadas cargas do sistema usuário sejam reduzidas ou desconectadas em momentos em que haja dificuldades em manter o balanço entre geração e demanda a um preço economicamente atrativo. No caso, o NFT gerado e atribuído ao usuário comprova o seu comportamento favorável à redução do consumo de energia, e, consequentemente, registrando a redução da pegada de carbono do usuário. A facilitação da participação de um cliente, em especial, os de grande porte, em progra-

[31] https://www.flexidao.com/respring
[32] https://www.rexcreditnft.io/

mas de gestão pelo lado da demanda é de particular interesse para a concessionária e para o operador do sistema elétrico. Tal carga se torna sensível ao preço da energia, sendo conectada somente em períodos em que haja disponibilidade de uma fonte geradora de energia de baixo custo.

Em [Karandikar et al., 2021], o autor propõe *tokens* fungíveis para representar energia trocada entre pontos do sistema e *tokens* não fungíveis para representar que um cliente foi capaz de: i - participar de uma ação de gestão pelo lado da demanda; ii - estimar sua demanda com precisão; e/ou iii - disponibilizar um de banco de baterias para armazenamento de energia do sistema. Os *tokens* recebidos podem, por sua vez, ser trocados por criptomoedas.

2.6. Desafios e tendências de pesquisa

Dado o grau de inovação que permeia as cadeias de blocos, os contratos inteligentes e os NFTs, é necessário avaliar os resultados obtidos com as primeiras implementações desses conceitos e observar possíveis pontos de aprimoramento. Esta seção discorre sobre os desafios associados a essas tecnologias e projetos de que as utilizam como forma de soluções de pesquisa.

2.6.1. Desafios relacionados ao uso de NFTs

Os principais desafios das aplicações em cadeia de blocos e da tecnologia de NFT dentro do contexto de mobilidade elétrica e cidades inteligentes são descritos nesta seção. Entre os desafios abordados, tem-se questões como o congestionamento da rede e o seu impacto sobre as aplicações, a anonimização e a legislação, os impactos dos ataques, entre outros.

2.6.1.1. Riscos associados aos contratos inteligentes

Apesar de os contratos inteligentes serem amplamente utilizados, eles ainda apresentam diversos desafios em aberto relacionados à privacidade, ataques e ao próprio gasto de energia das principais cadeias de bloco [Medeiros et al., 2019]. Esses riscos ficam ainda mais claros em aplicações de amplo uso dos contratos inteligentes, tais como o DeFi e os NFTs. Com base nas observações sobre os usos nas diversas áreas, é possível discutir algumas questões sobre o uso dos contratos e dos NFTs em sistemas de energia. As aplicações de contratos inteligentes incluem coordenação de carregamento de veículos elétricos inteligentes, resposta automatizada pelo lado da demanda, comércio de energia ponto a ponto e alocação de tarefas de controle entre os operadores de rede [Kirli et al., 2022].

Um dos principais problemas associados aos contratos inteligentes é a existência de códigos maliciosos ou com erros, os quais podem gerar impactos que, nos piores casos, podem levar a uma exaustão dos fundos de um ou mais participantes. Hoje, já existem diversos exemplos desse tipo de problemas em contratos de DeFi e de venda de NFTs, onde os desenvolvedores dos contratos deixaram *backdoors* que permitem a manipulação fraudulenta de *tokens*. Dentro do contexto dos NFTs, existem exemplos de contratos de vendas parciais de NFTs que definem códigos auto-executáveis que especificam que em

novas re-vendas, parte do preço de venda deve ser transferido para o vendedor inicial de forma não explícita, resultando em uma perda para o comprador [Kirli et al., 2022]. Ataques que permitem o vazamento de criptomoedas para entidades não autorizadas durante a execução do contrato inteligente são chamados de Ataques de Vazamento, enquanto que ataques que permitem ao atacante finalizar o contrato quando do seu interesse são chamados de Ataques Suicidas [Liu et al., 2021].

Cabe ressaltar que, até o momento, ainda não existem relatos de fraudes em contratos inteligentes para o sistema elétrico, mas isso se deve ao fato de essas aplicações ainda serem poucas e com pequena escala, além de muitas delas serem executadas em cadeias privadas ao invés de nas cadeias públicas.

Os contratos inteligentes também podem apresentar problemas relacionados à programação. Por exemplo, eventos conhecidos como Desordem de Exceção são caracterizados quando ocorrem problemas no tratamento das exceções. Quando um contrato A chama uma função de um contrato B e, por alguma razão, essa função não está disponível, o comportamento correto do contrato A seria reverter todas as transações já realizadas. Contudo, se existir uma ou mais chamadas de função de baixo nível, tal como *call()* ou *send()*, a reversão das transações para na última chamada de função de baixo nível, deixando todas as chamadas subsequentes sem reversão. Com isso, as transações restantes não serão revertidas, gerando prejuízos para quem chamou o contrato A [Liu et al., 2021]. Outra vulnerabilidade conhecida está relacionada às funções reentrantes. Tais funções podem mudar o estado de um contrato. Contudo, se uma função não reentrante for chamada como uma função reentrante, isso pode levar ao roubo de criptomoedas [Liu et al., 2021]. Um exemplo famoso da ocorrência desse tipo de ataque foi o "The DAO" na rede Ethereum, que permitiu o roubo de mais de 3,64 milhões de unidades de Ether, o equivalente à 45 milhões de dólares na época (2016) [Zhao et al., 2017].

Além da preocupação com padrões de ataques já conhecidos contra os contratos, outro ponto importante, em especial para o contexto das redes elétricas inteligentes e cidades inteligentes, é que se use sistemas de autenticação e autorização fortes. Além disso, os diversos aspectos de segurança tradicionais devem ser considerados, incluindo uma análise detalhada do código do contrato antes da sua disponibilização e uso na cadeia de blocos.

2.6.1.2. Congestionamento da rede e escalabilidade da cadeia de blocos

A escalabilidade é um dos principais problemas atuais para o uso das cadeias mais populares, em especial as que são baseadas em prova de trabalho. Em comparação, uma rede de cartões de crédito consegue processar milhares de transações por segundo, enquanto que as cadeias de bloco mais populares, como o Ethereum e o Bitcoin, são bastante limitadas em quantas transações por segundo podem fazer.

A rede Ethereum cresceu muito rapidamente e se tornou uma das mais importantes cadeias de bloco, tendo importância equivalente ao Bitcoin no mercado. Esse rápido crescimento se deu pela possibilidade de criar contratos inteligentes e também a criação dos NFTs. O aparecimento do DeFi e do NFT criou uma segunda onda de uso do Ethereum, causando congestionamentos na rede e, consequentemente, alta no preço do gás

para uso da rede. Tal crescimento gera problemas de escalabilidade para a rede. Esses problemas são associados a um "trilema", que coloca que não é possível ter descentralização, escalabilidade e segurança simultaneamente [Kiong e Xiang, 2021]. O uso do mecanismo de consenso prova de trabalho garante os requisitos de descentralização e segurança, mas também implica em uma baixa escalabilidade, já que esse tipo de consenso limita o número de transações por segundo (*Transactions per Second* (TPS)).

O trilema da escalabilidade ganhou visibilidade em 2017, quando a aplicação CryptoKitties foi capaz de praticamente congelar o uso da Ethereum em seu primeiro pico de popularidade em vendas de NFTs. Com o rápido aumento do número de requisições de transações, os mineradores ficam sobrecarregados, implicando em filas e no aumento do custo em gás para realizar uma determinada operação. Eventos similares ocorrem com certa frequência com novos lançamentos de NFTs por aplicativos populares, entre outros eventos, fazendo com o que o custo das redes mais populares cresça muito rapidamente.

Entre as soluções para esse problema, estão as chamadas soluções de camada um, soluções de camada dois e a utilização de novos protocolos de consenso tais como Prova de Participação e Tolerância a Falhas Bizantinas, que substituem o ambiente energeticamente caro e pouco amigável do Prova de Trabalho. As soluções de camada um implicam na construção de uma nova cadeia de blocos, visando sobrepujar os desafios encontrados nas cadeias originais, tais como as cadeias Polkadot[33], Solana[34], Cosmos[35], Theta[36], Algorand[37], Fantom[38], Avalanche[39], NEAR[40], TRON[41] e CELO[42]. As soluções de camada dois são construídas sobre a cadeia já existente, como um *overlay*. Outro tipo de solução implica na criação de uma segunda cadeia de blocos que funciona em paralelo com a original, como é o caso do *Binance Smart Chain*[43] e da *Huobi Eco Chais* (HECO)[44], que são *forks* da cadeia Ethereum.

Entre as soluções de camada dois, tem-se:

1. *Rollups* de camada 2 – São soluções que realizam transações fora da cadeia principal (cadeia de camada um) antes de submeter uma transação para a cadeia de camada um. Com isso, existe uma redução da carga sobre a cadeia de camada um, reduzindo também os custos associados para a manutenção do serviço (já que espera-se que os custos de camada um sejam maiores que os de camada dois). Nesse modelo, a segurança continua sendo provida pelo registro dos dados na cadeia de camada um. Existem dois tipos de *rollups*: os *rollups* de conhecimento zero, que executam operações que demandam poder computacional fora da cadeia de camada

[33]https://polkadot.network/
[34]https://solana.com/
[35]https://cosmos.network/
[36]https://www.thetatoken.org/
[37]https://www.algorand.com/
[38]https://fantom.foundation/
[39]https://www.avax.network/
[40]https://near.org/
[41]https://tron.network/
[42]https://celo.org/pt
[43]https://www.binance.com/pt-BR
[44]https://www.hecochain.com/

um e apenas registram nessa cadeia as provas de validade das operações; e o *rollup* otimista, que assume que as transações são válidas por padrão e apenas executam cálculos na cadeia de camada um em eventos de desafio, como uma prova contra fraudes [Kiong e Xiang, 2021].

2. Canais de Estado (*State Channels*) – São canais de comunicação bidirecionais entre os participantes, por meio do qual eles podem realizar inúmeras transações fora da cadeia e, ao final, ambos registram o resultado final na cadeia de camada um [Kiong e Xiang, 2021]. Com esse tipo de abordagem, aumenta-se a velocidade de transações e reduz-se a sobrecarga da cadeia de camada um. Exemplos incluem o Connext[45], Kchannels[46], Perun[47] e Raiden[48].

3. Cadeias Paralelas (*Sidechains*) – São cadeias de blocos independentes da cadeia de camada um (em oposição às soluções por *fork* da cadeia principal), mas conectadas com ela por uma ponte bidirecional. Essas cadeias paralelas usualmente utilizam seus próprios algoritmos de consenso menos custosos, como Prova de Autoridade, Prova de Participação, Tolerância de Falhas Bizantinas, entre outros. Exemplos de cadeias paralelas incluem Skale[49], POA Network[50] e xDai[51].

4. Plasma – Essa solução foi criada para a rede Ethereum, funcionando como uma cadeia filha (cadeia de nível 2) para a cadeia pai (cadeia de nível 1) [Poon e Buterin, 2017]. As cadeias pai e filha coexistem independentemente, de forma que a cadeia filha pode ter os seus próprios contratos inteligentes, criando as suas próprias regras de negócio sem estar presa a taxa de transações ou ao preço de transação da cadeia principal. A segurança é aplicada pelo monitoramento da cadeia de nível 1 e da cadeia de nível 2, de tal forma que comportamentos fraudulentos na cadeia de nível 2 são penalizados. No Plasma, cadeias filhas podem criar filhas, gerando uma árvore que confere maior flexibilidade na criação do modelo de negócio, como mostrado na Figura 2.10. Iniciativa semelhante também foi criada no Bitcoin para resolver os problemas de escalabilidade, sendo chamada de *Lightning Network* [Poon e Dryja, 2016].

2.6.1.3. Anonimização

A anonimização é um dos pontos chaves para o uso de cadeias de blocos para aplicações de energia e de cidades inteligentes. Nesse contexto, é importante garantir que as chaves públicas e transações não possam ser associadas a um determinado usuário, expondo sua privacidade, especificamente, os locais por onde anda ou seu padrão de consumo de energia.

[45] https://www.connext.network/
[46] https://www.kchannels.io/
[47] https://perun.network/
[48] https://raiden.network/
[49] https://skale.space/
[50] https://www.poa.network/
[51] https://developers.gnosischain.com/

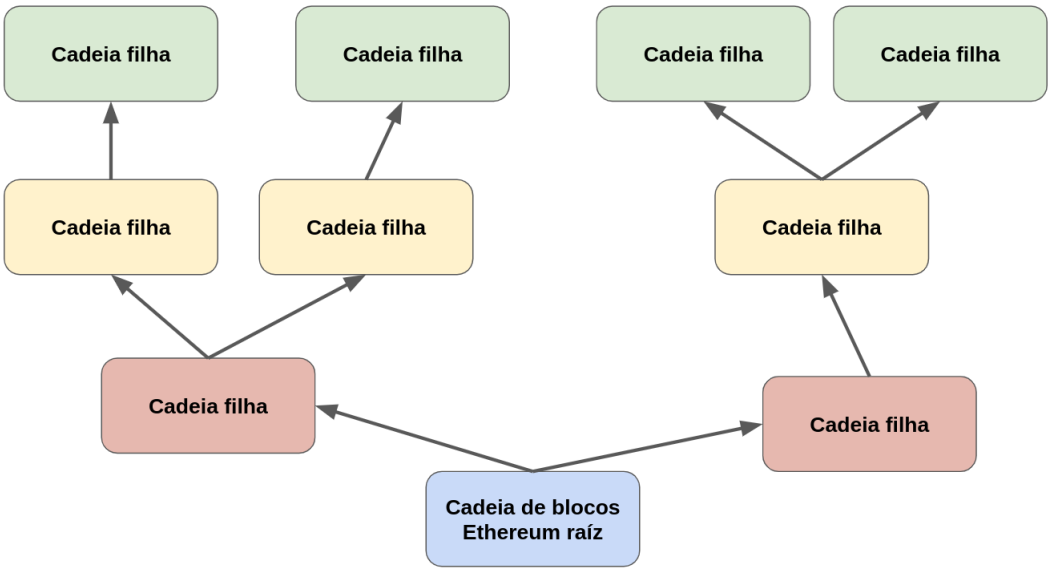


Figura 2.10. Estrutura em árvore de cadeias, gerada com a proposta do Plasma sobre a cadeia Ethereum.

A anonimização é especificamente difícil de se manter em cadeias de blocos públicas, aonde todos os registros são expostos a todos os usuários. Uma vez descoberta a identidade de um usuário, associando um nome a uma chave pública, todas as atividades daquele usuário são automaticamente expostas. Dessa forma, embora seja interessante armazenar transações e créditos de carbono na cadeia de blocos, não é recomendável, mesmo com o uso de criptografia, registrar dados pessoais, tais como consumo de energia, uso de veículos elétricos, entre outros.

Dessa forma, todas as aplicações baseadas em cadeias de blocos devem prover soluções concretas e bem estruturadas para evitar expor dados do usuário, seja pelo uso de cadeias privadas paralelas às públicas ou banco de dados paralelos. Além disso, tanto o controle de acesso quanto o sistema de armazenamento precisam ser cuidadosamente projetados para evitar vazamentos de dados e para permitir a remoção de dados pessoais sempre que necessário, conforme determinam as legislações vigentes, tais como a LGPD no Brasil.

Indo em uma direção contrária a da necessidade de anonimização dos usuários, existe a necessidade de se criar sistemas fortes de gestão de identidades para IoT, pois aplicações como o registro de créditos de carbono ou transações comerciais de energia demandam que os equipamentos e entidades que fazem registros nas cadeias de bloco sejam verificáveis e resistentes à violações (*tamper proof*). Portanto, aplicações de mobilidade elétrica e tarifação inteligente precisam ser capazes de identificar a origem dos dados e autenticar as informações contra violações, especialmente por questões legais associadas a esses tipos de aplicações.

2.6.1.4. Ataques contra a blockchain

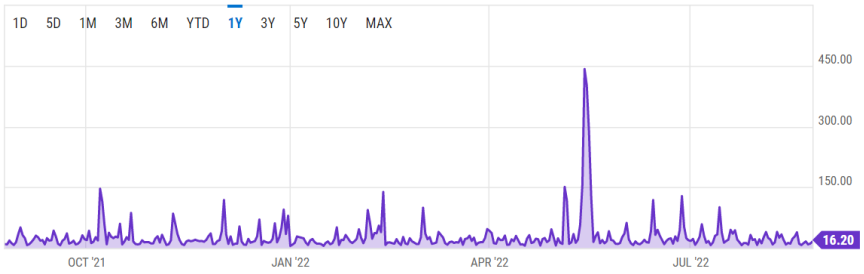
Uma das principais áreas para pesquisa e desenvolvimento para cadeias de blocos aplicadas ao sistema elétrico é a segurança cibernética. Especificamente, existem ataques que podem corromper as lógicas de negócio sendo executadas por meio de contratos inteligentes e NFTs.

Entre os principais ataques já conhecidos que podem afetar as aplicações do setor elétrico e das cidades inteligentes, destacam-se:

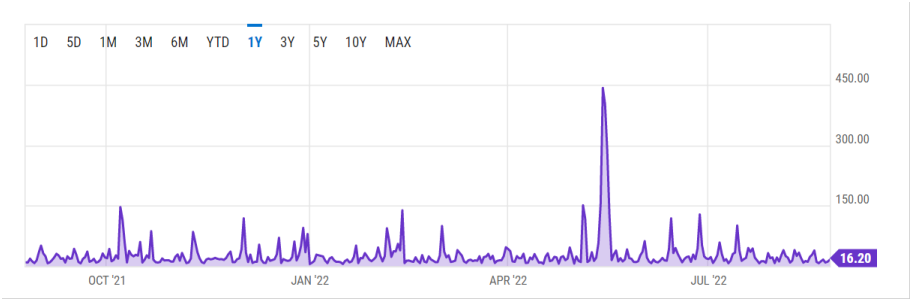
1. Ataque da Vivacidade (*Liveness Attack* - atrasa o tempo de confirmação da transação, visando obter vantagens. Esse ataque, que depende de um maior poder computacional do atacante, ocorre em três etapas: preparação, negação de transação e atraso de cadeia. Primeiramente, na preparação, o atacante tenta obter uma vantagem potencial contra usuários honestos para construir sua cadeia privada. Nesse caso, está se assumindo o uso de algoritmos de consenso que privilegiam o poder computacional do minerador. Em seguida, inicia-se a fase de negação da transação, na qual o atacante tenta atrasar um bloco genuíno que contém a transação alvo do ataque. Quando o invasor decide que o atraso não pode ser maior sem levantar suspeitas dos demais mineradores, ele prossegue para a fase de renderização da cadeia de blocos, onde tenta diminuir a taxa de transações na cadeia [Singh et al., 2021]. Atrasos em transações de leilões ou transações relacionadas a preços variáveis podem causar prejuízos significativos ao alvo.

2. Ataques de gasto duplo: busca utilizar o mesmo fundo para realizar mais de uma transação rápida sem validação. Esse ataque pode ocorrer em cadeias que tem um tempo de validação da transação lento, ou mais lento que a velocidade da transação no mundo real. Seria o caso, por exemplo, de uma transação de pagamento em um mercado, no qual o cliente paga e vai embora, sem esperar alguns minutos até confirmar que transação foi validada na cadeia. Em cenários como esse, o usuário poderia fazer diversas compras rápidas com o mesmo crédito, aproveitando-se de que não seria viável validar a transação em tempo real [Karame, 2012]. Essa possibilidade acaba por tornar a implementação de muitas aplicações inviáveis em cadeias de blocos, pois não validar a transação em tempo real permite a realização de fraudes. Como ilustração, a Figura 2.11 mostra o tempo de validação em minutos de transações na Bitcoin e no Ethereum, mostrando que são tempos relativamente longos e altamente variáveis. Dada essa realidade, existem esforços em pesquisa para reduzir e homogenizar o tempo de validação de transações em novas cadeias de bloco, possibilitando o uso de cadeias de bloco em aplicações com transações de validação rápida.

3. Roubo da Chave Privada: Todas as transações e acesso a fundos em cadeias de blocos são feitas por meio do uso de criptografia assimétrica. A perda de uma chave privada leva a perda de acesso ao fundo, sem possibilidade de recuperação. Assim, muitos ataques a computadores recentes visam encontrar e roubar chaves privadas armazenadas pelos usuários, o que permite o acesso à cadeia pelo agente malicioso [Singh et al., 2021].



(a) Tempo de confirmação de uma transação no Bitcoin em minutos.



(b) Tempo de confirmação de uma transação no Ethereum em minutos.

Figura 2.11. Estatísticas do último ano de tempo de transação nas duas principais cadeias de bloco atuais. Fonte: [Ycharts.com, 2022a, Ycharts.com, 2022b]

4. Ataque de Colisão: Esse ataque se aplica a cadeias de bloco baseadas em Prova de Trabalho. Nesse caso, se o atacante possuir mais de 50% do poder de mineração na rede, ele pode redefinir a cadeia da forma que achar mais vantajosa para si, realizando outros ataques, como por exemplo, a realização de Ataques de Gasto Duplo, por meio da criação de ramos na cadeia. Uma versão diferente do ataque se aplica a cadeias baseadas no consenso Tolerância de Falhas Bizantinas. Nesse caso, se o atacante controla pelo menos 50% dos nós de validação escolhidos pelo líder de consenso, ele pode validar algo errado ou não validar algo correto, trazendo problemas para as aplicações e regras de negócios executadas sobre a cadeia de blocos [Liu et al., 2021].

5. Ataque Sybil: Nesse ataque, o atacante cria um grande número de identidades falsas, visando levar vantagem em algoritmos de consenso baseados em votação, tais como os grupos de validação *Practical Bizantine Fault Tolerant* (PBFT), ou sistemas DAO.

6. Inserção de dados falsos: As aplicações no domínio de sistemas de energia e de cidades inteligentes podem se beneficiar das propriedades de segurança que as cadeias de blocos trazem em inúmeros sentidos. Contudo, as cadeias de bloco, seus contratos inteligentes e os NFTs apenas garantem a segurança dos registros no mundo virtual. Não existe, na tecnologia, uma garantia que os registros cibernéticos de fato correspondem aos eventos do mundo real. Assim, atacantes podem fabricar/falsificar dados e enviá-los para a cadeia de blocos [Gourisetti et al., 2021]. Dessa forma, há que existir uma camada de segurança ciber-física que garanta, entre

outros, a não-violação dos dispositivos aptos a fazer registros na cadeia de blocos em uma determinada aplicação.

Cabe destacar que qualquer ataque que possa influenciar as regras de negócio ou levar a perdas financeiras são potencialmente negativos para as aplicações do setor energético e de cidades inteligentes. Nesse sentido, vários esforços em pesquisa vem sendo aplicados para tentar minimizar os impactos desses ataques, assim como em melhorar o desempenho das cadeias de blocos.

2.6.1.5. Regulamentação

Embora as cadeias de blocos tenham grande potencial para moldar novos modelos no mercado de energia, ainda existe uma grande restrição no que diz respeito à legislação e regulamentação.

Usualmente, o setor elétrico dos países costuma ser altamente regulamentado, com órgãos de operação, regulação e controle de mercado, que definem um vasto conjunto de regras e normas de operação. Essas regulações visam garantir que o acesso à energia seja confiável, seguro e acessível [Stanwell, 2022].

Dentro desse contexto, as cadeias de bloco ainda não são vistas com bons olhos dentro do setor, por serem consideradas como uma tecnologia emergente, ainda pouco testada e não suficientemente estável. Essa desconfiança acaba por retardar significativamente a regulação e a integração dessa nova tecnologia ao mercado de energia. Além disso, observa-se que a natureza descentralizada das cadeias de bloco dificultam o desenvolvimento de estruturas e modelos legais que regulamentem o seu uso, tornando-se um desafio para os setores regulatórios [World Economic Forum, 2018]. Por exemplo, não é possível responsabilizar ou punir um responsável por ataque em uma cadeia de bloco que venha a prejudicar consumidores e geradores de energia se não for possível identificá-lo no mundo físico.

Outro ponto crucial relacionado à legalização das cadeias de blocos, contratos inteligentes e NFTs para aplicações de energia diz respeito à questão da anonimização e da persistência dos dados. Leis como a *European General Data Protection Regulation* (GDPR), na Europa, e a Lei Geral de Proteção de Dados (LGPD), no Brasil, trazem questões desafiadoras para o contexto das cadeias de bloco [Zemler e Westner, 2019, Morte et al., 2020]. Essas leis determinam fortes restrições protegendo dados privados e confidenciais de usuários, exigindo a garantia de privacidade e dando ao usuário o direito de deletar dados privados quando achar apropriado. Uma vez que a cadeia de blocos é imutável, não é possível apagar registros. Além disso, para o caso de cadeias públicas, todos os registros ficam abertos para todos os nós da cadeia.

Portanto, a legalização e regulamentação das cadeias de bloco passam não só pela maturidade da tecnologia, como também pelo forte investimento em pesquisa em mecanismos que permitam garantir a privacidade e a deleção de dados sem, com isso, ter que alterar a estrutura da cadeia [Sahmim et al., 2019].

2.6.2. Projetos de pesquisa e tendências futuras

2.6.2.1. Tendências de pesquisa

Uma das principais tendências de pesquisa está relacionada aos mercados de carbono. Com o aumento do interesse na redução do aquecimento global, diversos fóruns mundiais passaram a propagar iniciativas de compensação de emissão de gás carbônico por créditos de carbono advindos de atividades que reduzem a emissão de carbono.

Dentro das iniciativas já existentes, tem-se a da *Universal Protocol*, que lançou um crédito de carbono negociável no varejo, permitindo que projetos certificados transformem suas reduções de gases de efeito estufa em créditos de carbono negociáveis. Esse créditos são definidos como tokens fungíveis, chamados de *Universal Carbon* (UPCO2)[52]. A empresa criou ainda o *Bitcoin Zero*, que é um *token* construído como contrato inteligente do tipo ERC-20 baseados no UPCO2. O *Bitcoin Zero*[53] é um *wrapper* que combina um Bitcoin com a retirada de 10 toneladas de carbono obtidos a partir dos projetos de floresta tropical baseados no instrumento REDD+, criado pela Convenção-Quadro das Nações Unidas sobre Mudança do Clima - *United Nations Framework Convention on Climate Change* (UNFCCC). Esses créditos de carbono são certificados por organismos internacionais, validando o que seria um Bitcoin "verde".

A *SavePlanetEarth*[54], uma empresa inglesa, está disponibilizando uma plataforma para NFTs inteligentes de crédito de carbono certificados na cadeia de blocos *Phantasma*. O ciclo de funcionamento desses tokens é apresentado na Figura 2.12.

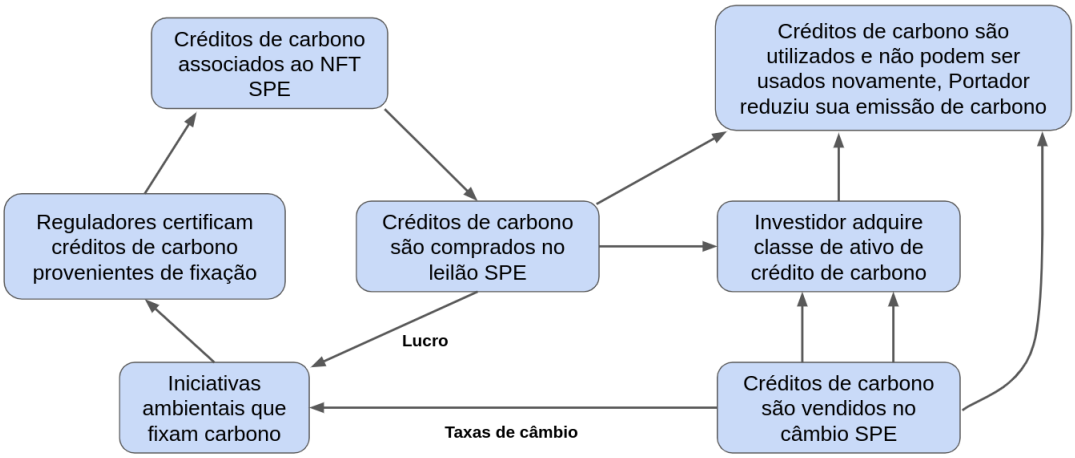


Figura 2.12. Ciclo de vida dos NFTs da organização *Save Planet Earth*, os quais são tokens de carbono negociáveis. Adaptado de [Save Planet Earth, 2021].

A *First Carbon*[55] desenvolveu uma plataforma chamada de *Mint Carbon*[56], que permite a negociação de créditos de carbono como NFTs criados na cadeia de bloco Poly-

[52]https://universalcarbon.com/
[53]https://www.universalprotocol.io/bitcoinzero
[54]https://saveplanetearth.io/
[55]https://www.firstcarbonsolutions.com/
[56]https://mintcarbon.io/

gon[57], fornecendo aos emissores de crédito de carbono acesso a cadeia de blocos e permitindo que os usuários rastreiem e negociem seus créditos.

Outros estão usando as cadeias de bloco para financiar soluções regenerativas. Por exemplo, o projeto NFTree[58], criado pela parceria entre a *Crown Platform* e a *Micorriza Association* na Espanha, está criando NFTs de pegadas de carbono na cadeia de blocos Crown[59]. A proposta é financiar o progresso ecológico por meio de certificados de carbono com validade mundial, transparentes e de fácil acesso. Já a *Carbonland Trust*[60] criou um ativo de créditos de remoção de carbono tokenizados como NFTs baseado em conservação florestal, chamado de *Carbonland Trust ESG NFTs*. Outra iniciativa é da *Cambridge Centre for Carbon Credits* (4C)[61], que promove uma solução para comprar créditos de carbono para financiar soluções "verdes" que busquem preservar biodiversidade. A *ClimateCoin Foundation*[62] incentiva a compensação de emissões de carbono, transformando registros de crédito de carbono oficiais em NFTs. Pessoas que plantam árvores ou reduzem as emissões de gás carbônico recebem tokens.

Dentro desse contexto, ficam claras as oportunidades de pesquisa dentro dos VCM, permitindo soluções mais seguras, com maior controle de mercado e que consideram novas formas de geração de crédito de carbono.

Existem ainda inciativas dentro da área de geração de energia distribuída. Nesse caso, as cadeias de blocos surgem como forma de melhorar a gerência dos mercados de energia descentralizados, aumentando a confiabilidade e a transparência dos serviços. Entre as iniciativas de mercado já disponíveis, tem-se a Powerledger[63] que permite a compra, venda ou troca do excesso de eletricidade renovável gerada por meio de uma rede par-a-par. O projeto Solstroem[64] foca na aceleração da transição energética em países em desenvolvimento e emergentes, fornecendo créditos de microcarbono pela geração em rede solar ao invés de pelo uso de combustível. Os créditos, que são georreferenciados e com carimbo de data/hora, podem ser vendidos para indivíduos ou empresas que precisem realizar compensações de crédito de carbono.

Entre os mercados de energia baseados em cadeia de blocos, destaca-se o *Grid Singularity*[65], que é uma plataforma ciente da grade energética para o mercado de energia descentralizado. Outra plataforma similar é a *TransActive Grid*[66], que usa as cadeias de bloco para criar um mercado de energia produzida em casa em uma escala local.

Com a tokenização dos RECs, gerou-se um novo modelo de negócios, simplificando o processo e evitando a possibilidade de dupla contagem e outras formas de fraude [Nova, 2021]. A transparência trazida pelos RECs tokenizados e negociados por

[57] https://polygon.technology/
[58] https://nftree.org/
[59] https://www.crownplatform.com/
[60] https://www.carbonlandtrust.com/
[61] https://4c.cst.cam.ac.uk/
[62] https://www.climateaction.org/directory/climate-coin-foundation?supplier=Climate%20Coin%20Foundation#products-and-services
[63] https://www.powerledger.io/
[64] https://www.solstroem.com/
[65] https://gridsingularity.com/
[66] http://www.solutionsandco.org/project/transactive-grid/

meio de contratos inteligentes e NFTs abre a oportunidade de criar novos tipos de REC, relacionados com variáveis como tempo, local da geração de energia, tipo de geração e até mesmo outras formas de redução de emissões de gás carbônico. Com isso, existe uma ampla gama de oportunidades de pesquisa em modelos de mercado e desenvolvimento tecnológico para viabilizar esses novos modelos de forma segura e inteligente.

É importante destacar que a principal proposta de valor dos NFTs para RECs é a garantia de não duplicidade, natural aos NFTs, e de integridade garantida pelo contrato inteligente acionado pelos próprios dispositivos geradores de energia.

2.6.2.2. Projetos de pesquisa

Existem diversos projetos de pesquisa e desenvolvimento pelo mundo relacionando as cadeias de blocos com aplicações do setor de energia. Andony *et al.*, em sua pesquisa, mostram uma classificação percentual dos casos de uso de cadeias de bloco no setor energético. A Figura 2.13 mostra esses dados, evidenciando que uma parcela significativa dos investimentos se aplica à geração de tokens e de certificados verdes [Andoni et al., 2019].

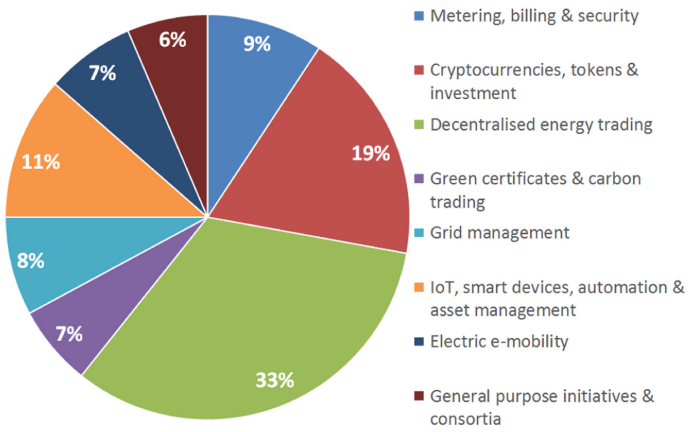


Figura 2.13. Estudo da distribuição entre áreas de 140 iniciativas de uso de cadeias de blocos no setor de energia, promovidas por empresas e instituições de pesquisa. Fonte: [Andoni et al., 2019].

Cabe destacar que projetos de pesquisa relacionadas ao mercado de energia e de carbono estão surgindo por todo mundo, em consonância com as evoluções e o surgimento de *startups*.

Entre os projetos existentes nos EUA, pode-se citar o *Brooklyn Microgrid*, que desenvolveu uma plataforma para comercialização de energia solar [67], sendo um dos primeiros programas de engenharia aplicada com cadeias de blocos utilizadas no setor de energia. Nesse projeto, que envolvia dez residências, cinco residências eram geradoras de energia, enquanto que as demais residências compravam o excedente, por meio de transações par-a-par [Zhao et al., 2019]. Outros projetos incluem o *Filament*, que usa cadeias de blocos para gerenciamento de informações de dispositivos IoT para dar suporte a soluções

[67] https://www.brooklyn.energy/

de falha na rede elétrica [Tripathy et al., 2020]. Outro projeto, desenvolvido por empresas de carregamento de baterias de carro americanas e alemãs, visou o desenvolvimento de estações de carregamento de veículos elétricos compartilhados baseado em cadeias de blocos. Com essa plataforma, é possível vender energia gerada e também carregar o carro por meio do aplicativo JuiceNet [Yaqub et al., 2020, Zhao et al., 2019].

Na Irlanda, tem-se o projeto EnerPort, promovido pelo governo irlandês, em parceria com indústrias, para criar uma rede par-a-par colaborativa, baseada em cadeias de blocos. Nesse projeto, visa-se promover o mercado de energia par-a-par entre microgrids [Verma et al., 2018].

No Chile, a *Comisión Nacional de Energía*, do Ministério das Energias, criou o portal *Energía Abierta Beta*. Esse serviço é baseado nas cadeias de blocos e visa dar maior transparência ao mercado de energia no país [Pareti e Núñez, 2021].

Na África do Sul, em 2015, foi desenvolvido o *Sun Exchange's*, um *marketplace* baseado em cadeia de blocos para microgrids. O projeto usa tokens para a energia gerada [Jackson, 2022].

Na China, o projeto ECO2 Ledger usa cadeias de blocos para controlar créditos de carbono, tornando-os mais confiáveis e rastreáveis[68]. O sistema permite ainda que pessoas consigam medir a sua redução de pegada de carbono, no aplicativo MyCarbon[69], e vender esses créditos. O serviço rapidamente alcançou mais de 500 mil usuários e acumulou mais de 100 mil toneladas de crédito de carbono.

A empresa Nori [70] propõe um sistema em que um NFT intitulado NRT é emitido na rede Polygon para um usuário que foi capaz de implementar, com sucesso, um projeto para remover uma tonelada de CO2 da atmosfera por ao menos dez anos. A mecânica requer que uma empresa independente verifique o projeto e ateste sua validade por meio da metodologia US Croplands. Uma vez em posse do NFT, seu dono pode trocá-lo na cadeia de blocos com um usuário que tenha demanda por créditos de carbono. A empresa alega que, antes desse sistema, para fazer uma transação de crédito de carbono, era necessário uma série de contratos e intermediários. Com a sua utilização, a cadeia de blocos se torna um canal único de transação entre o usuário que gerou o crédito de carbono e o usuário que deseja adquiri-lo.

No Brasil, o projeto "Implantação de um Modelo de Negócio para Compartilhamento de Veículos Elétricos Usando como Estudo de Caso o Sistema de Transporte Coletivo da UFF", em desenvolvimento na Universidade Federal Fluminense (UFF), visa utilizar uma cadeia de blocos para a tarifação no compartilhamento de veículos elétricos da UFF de forma sustentável e com incentivos sociais aos alunos. Esta proposta prevê o desenvolvimento de um sistema de compartilhamento de veículos elétricos, compreendendo toda a cadeia de produção, desde o controlador veicular presente nos veículos elétricos, até a plataforma online de monitoramento, controle e tarifação dos veículos. Ainda, este sistema será testado em um ambiente operacional, constituído pelo sistema de transporte coletivo da UFF, resultando em testes, demonstrações e avaliações que qua-

[68] https://www.eco2.cc/

[69] https://eco2.cc/MyCarbon.html

[70] https://nori.com/

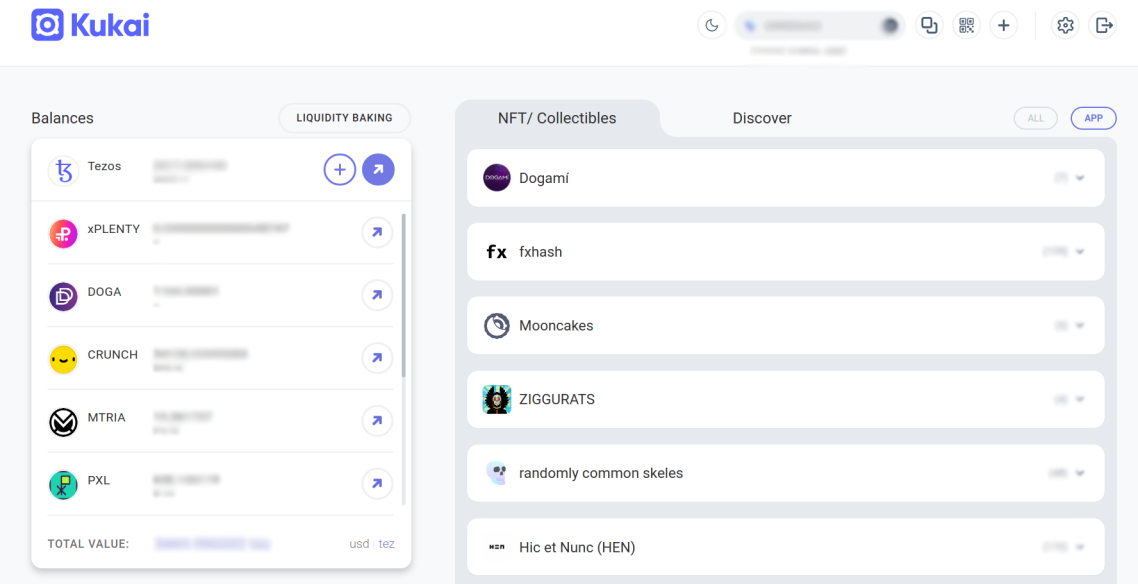


Figura 2.14. Aplicativo da carteira que habilita o acesso à rede de bloco Tezos.

lifiquem o sistema de compartilhamento com elevado grau de maturidade (TRL7). Por fim, será construído um modelo de negócios para o sistema proposto, de modo a permitir a extensão para todo o território nacional. Entre os objetivos, tem-se o uso de NFTs para gerenciamento de identidades e créditos de carbono.

A aplicação de cadeias de bloco para sistemas de energia é um tema altamente discutido e que está movimentando altíssimas quantias por todo o mundo. Os projetos apresentados visam trazer uma amostragem do que está sendo feito e a capilaridade das iniciativas.

2.7. Hands-on - Emissão e transferência de NFTs

Uma parte fundamental desta proposta de aplicação de mobilidade elétrica são as operações de emissão e transferência de propriedade de NFTs. Nesse cenário, o *hands-on* permite o contato direto com essas operações por meio de duas cadeias de blocos amplamente utilizadas no mercado: Ethereum e Tezos. A emissão permite os participantes do minicurso criarem um NFT utilizando qualquer tipo de mídia digital. A transferência de propriedade permite que os participantes vislumbrem a troca ou venda de um NFT com outros pares. Essas operações, em conjunto, viabilizam o livre mercado de NFTs que motivam o seu uso em diversas esferas.

Em primeiro lugar, é demonstrado como criar uma carteira na cadeia de blocos Tezos. A carteira é um aplicativo descentralizado (ou DApp), neste caso, web, que cria uma interface amigável para acessar as informações da carteira que está armazenada na cadeia de blocos. Além disso, a carteira também permite a interação com a cadeia de blocos, como submeter uma transação à cadeia. É adotada a plataforma Kukai[71] para a demonstração e criação da carteira. A carteira pode ser observada pela Figura 2.14.

[71] `https://wallet.kukai.app/`

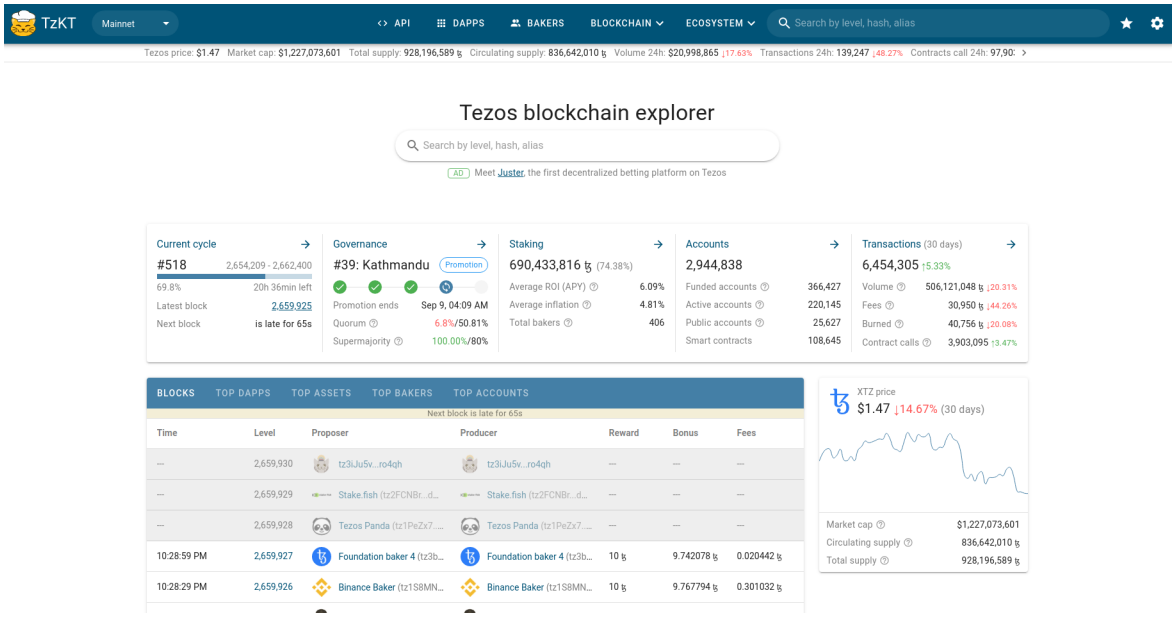


Figura 2.15. Indexador da cadeia de blocos Tezos.

Ao lado esquerdo da Figura, se encontram as moedas fungíveis, onde aparecem suas respectivas logos e a quantidade que a carteira possui. Ao lado direito da Figura, são as coleções e quantos NFTs a carteira possui para cada coleção. Como exemplo, Ziggurats é uma coleção criada pelo Mike Shinoda, o cantor da banda Linkin Park. Essa carteira da figura possui ao menos um NFT de cada coleção listada.

Em seguida, é demonstrado um indexador e explorador de cadeia de blocos — Tezos e Ethereum. Cada cadeia de blocos possui um meio de consultar informações históricas de maneira eficiente. Trabalhar diretamente com a cadeia de blocos para buscar informações históricas seria demorado. Portanto, a plataforma TzKT[72], ou o Etherscan[73] no Ethereum permitem diversas consultas, como exemplo: (1) quais foram todas as transações efetuadas para um NFT específico?; (2) qual o saldo de $XTZ de uma determinada carteira?; (3) quem criou determinado NFT? Além disso, o indexador também permite identificar outras informações gerais como quais as transações foram processadas em determinado bloco, ou a quantidade de gás uma determinada transação consumiu, ou ainda o total de gás consumido pelo bloco. É possível acompanhar as recompensas históricas obtidas por meio do consenso de Prova de Participação, ver qual foi o nó que processou o bloco, ou ainda buscar por carteiras, contratos inteligentes, ou até por aplicações. É possível observar pela Figura 2.15 a interface do indexador TzKT da cadeia de blocos Tezos.

Com efeito, a plataforma Objkt[74], que é a maior plataforma de negociação de NFTs de propósitos gerais na cadeia de blocos Tezos, será demonstrada. Neste momento, serão identificadas as informações associadas ao NFT na plataforma, tal como royalties, colecionadores, coleções, transações, volume, etc. Em seguida, será demonstrado como

[72] https://tzkt.io/
[73] https://etherscan.io/
[74] https://objkt.com/

negociar NFTs por meio da compra de um NFT. O NFT comprado será demonstrado via carteira web ou pelo explorador da cadeia de blocos por meio da transação ou pelo NFT em si. Será avaliado como o conteúdo digital do NFT está armazenado: se ocorre por meio do IPFS, e quais metadados estão associados e armazenados fora da cadeia.

Em seguida, será demonstrado como criar um NFT. O NFT criado será distribuído para todos os participantes. Pode-se dizer que o NFT distribuído funcionará como um mecanismo de POAP (*Proof of Attendance Protocol*), isto é, um NFT que irá representar a experiência física do minicurso de maneira digitalizada. Ou ainda, representará a prova de comparecimento ao minicurso. Os participantes serão encorajados a criarem suas carteiras durante o minicurso para poderem receber o POAP.

Com efeito, todas as tarefas de exploração do indexador da cadeia de blocos, emissão de NFT e busca por informações na plataforma de negociação de NFTs serão repetidas para a cadeia de blocos Ethereum — exceto a parte de emitir e comprar NFTs em função do alto custo das transações no Ethereum. Para esta cadeia de blocos, serão adotadas as plataformas de negociação OpenSea[75] e Foundation[76] durante o hands-on.

2.8. Considerações finais e perspectivas futuras

O padrão NFT tem sido utilizado em vários domínios, tais como cadeias de suprimentos, rastreabilidade de dados e manufatura, para representar bens físicos na forma de tokens digitais por conta de suas características de unicidade e rastreabilidade durante todo clico de vida do ativo. Entretanto, cada abordagem tem uma metodologia diferente para representar e sincronizar as informações entre o mundo físico e o digital.

O NFT é uma tecnologia ainda recente e com diversos desafios de pesquisa. Um dos principais desafios é sobre segurança e privacidade. No estágio atual, o anonimato e a privacidade dos NFTs ainda são pouco estudados. A maioria das transações NFT depende da plataforma Ethereum, que fornece apenas pseudo-anonimato. Os usuários podem ocultar parcialmente suas identidades se os links entre suas identidades reais e os endereços correspondentes forem desconhecidos pelo público. Caso contrário, todas as atividades dos usuários sob o endereço exposto são visíveis. As soluções existentes de preservação de privacidade, como, por exemplo, criptografia homomórfica [Wang et al., 2020], prova de conhecimento zero [Wang e Kogan, 2018], assinatura em anel [Noether et al., 2016], computação multipartidária [Neto et al., 2020], ainda não foram aplicadas aos esquemas relacionados a NFT devido à sua complexidade. Semelhante a outros tipos de sistemas baseados em cadeia de blocos, diminuir os custos computacionais torna-se característica chave para implementação de esquemas de privacidade do usuário.

Nos principais projetos de NFT, um *hash* criptográfico como identificador será marcado com o *token*, em vez de uma cópia do arquivo, e depois registrado na cadeia de blocos para economizar consumo de energia. Isso faz com que o usuário perca a confiança no NFT porque o arquivo original pode ser perdido ou danificado. Vários projetos de NFT integram um sistema de armazenamento de arquivos especializado, como o IPFS [Benet, 2014], no qual os endereços IPFS permitem que os usuários encontrem um conteúdo

[75] `https://opensea.io/`
[76] `https://foundation.app/`

desde que algum cliente da rede IPFS o hospede. Inevitavelmente, tais sistemas têm falhas. Quando os usuários carregam metadados NFT para nós IPFS, não há garantia de que seus dados serão replicados entre todos os nós. Os dados podem ficar indisponíveis se o ativo estiver armazenado no IPFS e o único nó que o armazena estiver desconectado da rede. Este problema foi relatado por `decrypto.io` [77] e `checkmynft.com`. Além disso, um NFT pode apontar para um endereço de arquivo incorreto. Se for esse o caso, um usuário não pode provar que ele realmente possui o NFT. Portanto, contar com um sistema externo como componente central de armazenamento para um sistema NFT é vulnerável.

De toda a forma, apesar de ainda existirem riscos associados, os NFTs já mostraram de forma concreta o seu potencial como instrumento para revolucionar mercados digitais. As vantagens trazidas são muito grandes e reduzem significativamente a burocracia e os custos no registro e negociação de bens não-fungíveis. Os NFTs representam ainda uma forma de tornar o processo mais seguro e transparente para os usuários, tornando toda a cadeia de processo mais segura.

As oportunidades em termos de aplicações trazidas pelos NFTs ainda estão longe de serem exauridas, existindo diversas possibilidades de criação de novos negócios com potencial para revolucionar no mercado. As oportunidades de pesquisa, em especial na área de segurança, para essas aplicações também são inúmeras, representando um tema que ainda será abordado na academia pelos próximos anos.

Referências

[Andoni et al., 2019] Andoni, M., Robu, V., Flynn, D., Abram, S., Geach, D., Jenkins, D., McCallum, P. e Peacock, A. (2019). Blockchain technology in the energy sector: A systematic review of challenges and opportunities. *Renewable and Sustainable Energy Reviews*, 100:143–174.

[Ante, 2022] Ante, L. (2022). The Non-Fungible Token (NFT) market and its relationship with Bitcoin and Ethereum. *FinTech*, 1(3):216–224.

[Apolinário, 2019] Apolinário, W. d. A. (2019). Implementação de smart contracts e tokens não-fungíveis na geração e aquisição de certificados de energia renovável no brasil. B.S. thesis, Universidade Federal do Rio Grande do Norte.

[Arcenegui et al., 2021] Arcenegui, J., Arjona, R., Román, R. e Baturone, I. (2021). Secure combination of IoT and blockchain by physically binding IoT devices to smart non-fungible tokens using PUFs. *Sensors*, 21(9):3119.

[Beeple, 2021] Beeple, C. (2021). Everydays: The first 5000 days. `https://onlineonly.christies.com/s/beeplefirst-5000-days/beeple-b-1981-1/112924?ldp\_breadcrumb=back`. Acessado em 24 de agosto de 2022.

[Benet, 2014] Benet, J. (2014). IPFS-content addressed, versioned, P2P file system - Draft 3. *arXiv preprint arXiv:1407.3561*, p. 1–11.

[Bloomberg, 2022] Bloomberg (2022). Nvidia game card prices fall along with crypto mining demand. `https://www.bloomberg.com/news/articles/2022-06-30/nvidia-game-card-prices-plunge-along-with-crypto-mining-demand`. Acessado em 26 de agosto de 2022.

[77] Disponível em `https://decrypt.co/62037/missing-or-stolen-nfts-how-to-protect`. Acessado em 01/09/2022

[Browne, 2021] Browne, R. (2021). Visa jumps into the NFT craze, buying a 'CryptoPunk' for $150,000. https://www.cnbc.com/2021/08/23/visa-buys-cryptopunk-nft-for-150000.html. Acessado em 24 de agosto de 2022.

[Carbon Credits, 2021] Carbon Credits (2021). The ultimate guide to understanding carbon credits. https://carboncredits.com/the-ultimate-guide-to-understanding-carbon-credits/. Acessado em 31 de agosto de 2022.

[Chainlink, 2022a] Chainlink (2022a). Chainlink. https://chain.link/. Acessado em 27 de agosto de 2022.

[Chainlink, 2022b] Chainlink (2022b). New report highlights how blockchains and oracles are redefining the energy industry. https://blog.chain.link/blockchains-and-oracles-are-redefining-the-energy-industry/. Acessado em 31 de agosto de 2022.

[Che, 2021] Che, T. (2021). The world's 1st destroyed diamond NFT: Who, what, when, where, why & how. https://taschalabs.com/the-worlds-1st-destroyed-diamond-nft-who-what-when-where-why-how/. Acessado em 23 de agosto de 2022.

[CoinTelegraph Research, 2021] CoinTelegraph Research (2021). Blockchains vie for NFT market, but Ethereum still dominates. https://cointelegraph.com/news/blockchains-vie-for-nft-market-but-ethereum-still-dominates-report. Acessado em 24 de agosto de 2022.

[Digiconomist, 2022] Digiconomist (2022). Ethereum energy consumption index. https://digiconomist.net/ethereum-energy-consumption. Acessado em 26 de agosto de 2022.

[Dowling, 2022] Dowling, M. (2022). Is non-fungible token pricing driven by cryptocurrencies? *Finance Research Letters*, 44:102097.

[Ethereum.org, 2022] Ethereum.org (2022). Ethereum energy consumption. https://ethereum.org/en/energy-consumption/. Acessado em 26 de agosto de 2022.

[Etherscan, 2021a] Etherscan (2021a). Transação de compra do diamante no Ethereum. https://etherscan.io/tx/0xb3eb6a51c740f3a9532d9f38ea874be377d08cf28a73135602d3e134cffa410e. Acessado em 23 de agosto de 2022.

[Etherscan, 2021b] Etherscan (2021b). Transação de compra do Nyan Cat no Ethereum. https://etherscan.io/tx/0xa1042c7dac0750c48049a5556d42553cec6f90d9ff1ec9bfe3b4574265d9ac2f. Acessado em 24 de agosto de 2022.

[Filecoin, 2022] Filecoin (2022). Filecoin. ipns://filecoin.io/. Acessado em 26 de agosto de 2022.

[Flow, 2022a] Flow (2022a). Flow. https://flow.com/. Acessado em 26 de agosto de 2022.

[Flow, 2022b] Flow (2022b). Sustainability built to be green: Leading web3 to a more eco-friendly future. https://flow.com/sustainability. Acessado em 26 de agosto de 2022.

[Gao et al., 2014] Gao, S., Chau, K. T., Liu, C., Wu, D. e Chan, C. C. (2014). Integrated energy management of plug-in electric vehicles in power grid with renewables. *IEEE Transactions on Vehicular Technology*, 63(7):3019–3027.

[Gao et al., 2020] Gao, Y., Al-Sarawi, S. F. e Abbott, D. (2020). Physical unclonable functions. *Nature Electronics*, 3(2):81–91.

[Gourisetti et al., 2021] Gourisetti, S. N. G., Ümit Cali, Choo, K.-K. R., Escobar, E., Gorog, C., Lee, A., Lima, C., Mylrea, M., Pasetti, M., Rahimi, F., Reddi, R. e Sani, A. S. (2021). Standardization of the distributed ledger technology cybersecurity stack for power and energy applications. *Sustainable Energy, Grids and Networks*, 28:100553.

[IEC TC 57, 2022] IEC TC 57 (2003-2022). IEC 61850 - communication networks and systems in substations. Relatório técnico, IEC - TC 57 (Technical Committee on Power systems management and associated information exchange).

[Jackson, 2022] Jackson, J. (2022). The blockchain projects making renewable energy a reality. `https://cointelegraph.com/magazine/2022/03/18/blockchain-projects-making-renewable-energy-reality`. Acessado em 23 de agosto de 2022.

[Kang et al., 2017] Kang, J., Yu, R., Huang, X., Maharjan, S., Zhang, Y. e Hossain, E. (2017). Enabling localized peer-to-peer electricity trading among plug-in hybrid electric vehicles using consortium blockchains. *IEEE Transactions on Industrial Informatics*, 13(6):3154–3164.

[Karame, 2012] Karame, G. O. (2012). Two Bitcoins at the price of one? Double-spending attacks on fast payments in Bitcoin. Em *In Proc. of Conference on Computer and Communication Security*.

[Karandikar et al., 2021] Karandikar, N., Chakravorty, A. e Rong, C. (2021). Blockchain based transaction system with fungible and non-fungible tokens for a community-based energy infrastructure. *Sensors*, 21(11):3822.

[Karger et al., 2021] Karger, E., Jagals, M. e Ahlemann, F. (2021). Blockchain for smart mobility—literature review and future research agenda. *Sustainability*, 13(23):13268.

[Kiong e Xiang, 2021] Kiong, L. e Xiang, L. (2021). *Investing in DeFi and NFT projects on alternative blockchains: A Beginner's Guide to Investing in DeFi and NFT Projects on Alternative Blockchains other than Ethereum*. Liew Voon Kiong.

[Kirli et al., 2022] Kirli, D., Couraud, B., Robu, V., Salgado-Bravo, M., Norbu, S., Andoni, M., Antonopoulos, I., Negrete-Pincetic, M., Flynn, D. e Kiprakis, A. (2022). Smart contracts in energy systems: A systematic review of fundamental approaches and implementations. *Renewable and Sustainable Energy Reviews*, 158(112013).

[Krause e Tolaymat, 2018] Krause, M. J. e Tolaymat, T. (2018). Quantification of energy and carbon costs for mining cryptocurrencies. *Nature Sustainability*, 1(11):711–718.

[Li et al., 2020] Li, W., Rentemeister, M., Badeda, J., Jöst, D., Schulte, D. e Sauer, D. U. (2020). Digital twin for battery systems: Cloud battery management system with online state-of-charge and state-of-health estimation. *Journal of Energy Storage*, 30:101557.

[Liu et al., 2021] Liu, C., Zhang, X., Chai, K., Loo, J. e Chen, Y. (2021). A survey on blockchain-enabled smart grids: Advances, applications and challenges. *IET Smart Cities*, 3:56–78.

[Lopes et al., 2015] Lopes, Y., Fernandes, N. C. e Muchaluat-Saade, D. C. (2015). Geração Distribuída de Energia: Desafios e Perspectivas em Redes de Comunicação. Em *Minicursos do XXXIII SBRC*, p. 55–109. SBC.

[Medeiros et al., 2019] Medeiros, D. S. V., Fernandes, N. C. e Mattos, D. M. F. (2019). Smart Contracts and the Power Grid: A Survey. Em *1st Blockchain, Robotics and AI for Networking Security Conference (BRAINS)*, p. 140–194.

[Miraz et al., 2021] Miraz, M. H., Excell, P. S. e Sobayel, K. (2021). The Non-Fungible Token (NFT) market and its relationship with Bitcoin and Ethereum. *Annals of Emerging Technologies in Computing (AETiC)*, 5(4):54–59.

[Morte et al., 2020] Morte, A. B., Meira, A., Costa, R. e Mariz, D. (2020). Uma análise sobre o uso de DLTs no tratamento de dados pessoais: Aderência aos princípios e direitos elencados na LGPD. Em *Anais do III Workshop em Blockchain: Teoria, Tecnologia e Aplicações*, p. 74–87, Porto Alegre, RS, Brasil. SBC.

[Musamih et al., 2022] Musamih, A., Salah, K., Jayaraman, R., Yaqoob, I., Puthal, D. e Ellahham, S. (2022). NFTs in healthcare: Vision, opportunities, and challenges. *IEEE Consumer Electronics Magazine*, p. 1–14.

[Neto et al., 2020] Neto, H. N. C., Mattos, D. M. F. e Fernandes, N. C. (2020). Privacidade do usuário em aprendizado colaborativo: Federated learning, da teoria à prática. Em *Livro de Minicursos do SBSEG 2020*. Sociedade Brasileira de Computação.

[Nita et al., 2018] Nita, S. L., Mihailescu, M. I. e Pau, V. C. (2018). Security and cryptographic challenges for authentication based on biometrics data. *Cryptography*, 2(4).

[Noether et al., 2016] Noether, S., Mackenzie, A. et al. (2016). Ring confidential transactions. *Ledger*, 1:1–18.

[Nofer et al., 2017] Nofer, M., Gomber, P., Hinz, O. e Schiereck, D. (2017). Blockchain. *Business & Information Systems Engineering*, 59(3):183–187.

[Nova, 2021] Nova, A. (2021). A token improvement: The rise of non-fungible tokens. `https://www.powerledger.io/media/a-token-improvement`. Acessado em 29 de agosto de 2022.

[Oliveira et al., 2020] Oliveira, M. T., Reis, L. H., Medeiros, D. S., Carrano, R. C., Olabarriaga, S. D. e Mattos, D. M. (2020). Blockchain reputation-based consensus: A scalable and resilient mechanism for distributed mistrusting applications. *Computer Networks*, 179:107367.

[Pareti e Núñez, 2021] Pareti, S. e Núñez, I. (2021). Blockchain as an information system in Chile: The case of Open Energy Project - Chilean's Ministry of Energy. *Revista Ibérica de Sistemas e Tecnologias de Informação*, 1(E39):554–568.

[Platt et al., 2021] Platt, M., Sedlmeir, J., Platt, D., Xu, J., Tasca, P., Vadgama, N. e Ibañez, J. I. (2021). The energy footprint of blockchain consensus mechanisms beyond proof-of-work. Em *2021 IEEE 21st International Conference on Software Quality, Reliability and Security Companion (QRS-C)*, p. 1135–1144.

[Poon e Buterin, 2017] Poon, J. e Buterin, V. (2017). Plasma: Scalable autonomous smart contracts. Relatório técnico, Plasma.io, `https://plasma.io/plasma.pdf`.

[Poon e Dryja, 2016] Poon, J. e Dryja, T. (2016). The Bitcoin lightning network: Scalable off-chain instant payments. Relatório Técnico Draft 0.5.9.2, Bitcoin Lightning, https://www.bitcoinlightning.com/wp-content/uploads/2018/03/lightning-network-paper.pdf.

[Rebello et al., 2019] Rebello, G., Camilo, G., Silva, L., Souza, L., Guimarães, L., Alchieri, E., Greve, F. e Duarte, O. (2019). Correntes de blocos: Algoritmos de consenso e implementaçao na plataforma Hyperledger Fabric. Em *Jornada de Atualização em Informática (JAI) - XXXIX Congresso da Sociedade Brasileira de Computação*.

[RED, 2021] RED (2021). Restart energy democracy platform - user guide. Relatório Técnico Version 1.0, RED.

[Rosenfeld et al., 2012] Rosenfeld, M. et al. (2012). Overview of colored coins. *White paper, bitcoil. co. il*, 41:94.

[Rossi, 2022] Rossi, E. (2022). NFTs for renewable energy certification - a solid use case. https://www.flexidao.com/post/nfts-for-renewable-energy-certification-a-solid-use-case. Acessado em 31 de agosto de 2022.

[Ruggieri et al., 2021] Ruggieri, R., Ruggeri, M., Vinci, G. e Poponi, S. (2021). Electric mobility in a smart city: European overview. *Energies*, 14(2).

[Sahmim et al., 2019] Sahmim, S., Gharsellaoui, H. e Bouamama, S. (2019). Edge computing: Smart identity wallet based architecture and user centric. *Procedia Computer Science*, 159:1246–1257.

[Saingre et al., 2022] Saingre, D., Ledoux, T. e Menaud, J.-M. (2022). Measuring performances and footprint of blockchains with BCTMark: a case study on Ethereum smart contracts energy consumption. *Cluster Computing*, 25(4):2819–2837.

[Save Planet Earth, 2021] Save Planet Earth (2021). Save planet earth official white paper - the cryptocurrency response to reversing the effects of climate change. Relatório Técnico v4, Save Planet Earth.

[Singh et al., 2021] Singh, S., Hosen, A. S. M. S. e Yoon, B. (2021). Blockchain security attacks, challenges, and solutions for the future distributed IoT network. *IEEE Access*, 9:13938–13959.

[Solana, 2022] Solana (2022). Solana's energy use report: March 2022. https://solana.com/news/solanas-energy-use-report-march-2022. Acessado em 26 de agosto de 2022.

[Spells of Genesis, 2022] Spells of Genesis (2022). Spells of genesis. https://spellsofgenesis.com/. Acessado em 23 de agosto de 2022.

[Stanwell, 2022] Stanwell (2022). How blockchain can impact the energy market. https://whatswatt.com.au/how-blockchain-can-impact-the-energy-market/. Acessado em 23 de agosto de 2022.

[Tezos Foundation, 2022] Tezos Foundation (2022). Sustainability through innovation. https://tezos.com/carbon/. Acessado em 26 de agosto de 2022.

[Thomason, 2022] Thomason, J. (2022). Blockchain and GreenFi - tools for climate action. https://thepaypers.com/expert-opinion/blockchain-and-greenfi-tools-for-climate-action--1257628. Acessado em 23 de agosto de 2022.

[Tripathy et al., 2020] Tripathy, R. P., Mishra, M. R. e Dash, S. R. (2020). Next generation warehouse through disruptive IoT blockchain. Em *2020 International Conference on Computer Science, Engineering and Applications (ICCSEA)*, p. 1–6. IEEE.

[Valeonti et al., 2021] Valeonti, F., Bikakis, A., Terras, M., Speed, C., Hudson-Smith, A. e Chalkias, K. (2021). Crypto collectibles, museum funding and OpenGLAM: challenges, opportunities and the potential of non-fungible tokens (NFTs). *Applied Sciences*, 11(21):9931.

[Verma et al., 2018] Verma, P., O'Regan, B., Hayes, B., Thakur, S. e Breslin, J. G. (2018). Enerport: Irish blockchain project for peer- to-peer energy trading. *Energy Informatics*, 1(14).

[VisaNews, 2021] VisaNews (2021). #newprofilepic. https://twitter.com/VisaNews/status/1430185056098820105. Acessado em 26 de agosto de 2022.

[Wang et al., 2020] Wang, Q., Qin, B., Hu, J. e Xiao, F. (2020). Preserving transaction privacy in Bitcoin. *Future Generation Computer Systems*, 107:793–804.

[Wang e Kogan, 2018] Wang, Y. e Kogan, A. (2018). Designing confidentiality-preserving blockchain-based transaction processing systems. *International Journal of Accounting Information Systems*, 30:1–18.

[World Economic Forum, 2018] World Economic Forum (2018). Fourth industrial revolution for the earth series - building block(chain)s for a better planet. Relatório técnico, World Economic Forum.

[Yahoo, 2021] Yahoo (2021). The burned picasso nft to digitally preserve artistic legacy. https://finance.yahoo.com/news/burned-picasso-nft-digitally-preserve-173000064.html. Acessado em 23 de agosto de 2022.

[Yaqub et al., 2020] Yaqub, R., Ahmad, S., Ali, H. e Asar, A. u. (2020). AI and blockchain integrated billing architecture for charging the roaming electric vehicles. *IoT*, 1(2):382–397.

[Ycharts.com, 2022a] Ycharts.com (2022a). Bitcoin average confirmation time. https://ycharts.com/indicators/bitcoin_average_confirmation_time. Acessado em 26 de agosto de 2022.

[Ycharts.com, 2022b] Ycharts.com (2022b). Ethereum average confirmation time. https://ycharts.com/indicators/ethereum_average_block_time. Acessado em 26 de agosto de 2022.

[Yoon et al., 2010] Yoon, J. W., Kim, H. e Huh, J. H. (2010). Hybrid spam filtering for mobile communication. *Computers & Security*, 29(4):446–459.

[Yuan e Wang, 2016] Yuan, Y. e Wang, F.-Y. (2016). Towards blockchain-based intelligent transportation systems. Em *2016 IEEE 19th International Conference on Intelligent Transportation Systems (ITSC)*, p. 2663–2668. IEEE.

[Zemler e Westner, 2019] Zemler, F. e Westner, M. (2019). Blockchain and GDPR: Application scenarios and compliance requirements. Em *Portland International Conference on Management of Engineering and Technology (PICMET)*, p. 1–8.

[Zhang e Wang, 2019] Zhang, X. e Wang, D. (2019). Adaptive traffic signal control mechanism for intelligent transportation based on a consortium blockchain. *IEEE Access*, 7:97281–97295.

[Zhao et al., 2017] Zhao, X., Chen, Z., Chen, X., Wang, Y. e Tang, C. (2017). The DAO attack paradoxes in propositional logic. Em *2017 4th International Conference on Systems and Informatics (ICSAI)*, p. 1743–1746.

[Zhao et al., 2019] Zhao, Y., Peng, K., Xu, B., Liu, Y., Xiong, W. e Han, Y. (2019). Applied engineering programs of energy blockchain in US. *Energy Procedia*, 158:2787–2793.

Capítulo 3

Introdução à Análise de Códigos Maliciosos para ambiente Windows

Renato Marinho[1,2], Mateus Santos[1], Raimir Holanda[1,2], Antonio Horta[1,3]

[1]Morphus Segurança da Informação
[2]Universidade de Fortaleza (UNIFOR)
[3]Instituto Militar de Engenharia (IME)

Abstract

Malicious code analysis aims at a deep understanding of how malware works, what tactics, techniques and procedures are used, what tools are applied, if there are network operations, file exfiltration, capture of user information or the system, among other activities. In this mini-course, participants will be trained in malicious code analysis techniques for the Windows environment. The course applies a methodology that seeks, incrementally, to aggregate new information about the malware at each stage, comprising the following stages: binary profiling (OSINT), fully automated analysis, binary properties analysis, behavior analysis and manual reverse analysis.

Resumo

A análise de código malicioso visa o entendimento profundo do funcionamento de um malware, como ele atua, quais as táticas, técnicas e procedimentos são utilizados, quais ferramentas são empregadas, se há operações de rede, exfiltração de arquivos, captura de informações do usuário ou do sistema, entre outras atividades. Neste minicurso os participantes serão capacitados nas técnicas de análise de códigos maliciosos para o ambiente Windows. O curso aplica uma metodologia, que busca, de forma incremental, agregar novas informações sobre o malware em cada estágio, compreendendo os seguintes estágios: profiling do binário (OSINT), análise totalmente automatizada, análise de propriedades do binário, análise de comportamento e análise reversa manual.

3.1. Introdução

O número de ataques cibernéticos está, sem dúvida, em ascensão, tendo como alvos setores público e privado e tendo como um dos principais vetores de ataque códigos maliciosos (*malicious software* ou *malware*), que são arquivos executáveis que apresentam deliberadamente intenções prejudiciais à sistemas computacionais [Skoudis and Zeltser 2003], o que pode incluir do vazamento de informações sensíveis do usuário ao controle total do sistema infectado.

Esses ataques cibernéticos se concentram em indivíduos ou organizações com um esforço para extrair informações valiosas. Em alguns casos, esses ataques cibernéticos estão supostamente vinculados a crimes cibernéticos ou grupos patrocinados por países, mas também podem ser realizados por grupos individuais para atingir objetivos financeiros. Apesar do crescente investimento em tecnologias de segurança, ainda estamos vendo sérias violações de segurança acontecendo em grandes corporações e governos [Kim 2018].

Em sua quase totalidade, esses ataques cibernéticos usam software malicioso (também chamado de malware) para infectar seus alvos. Mais especificamente, um malware é um código que executa ações maliciosas, podendo assumir a forma de um executável, script, código ou qualquer outro software. Apesar do avanço no desenvolvimento de antimalware, o número de ataques de malware está em tendência de alta [Sahay et al. 2020].

Em sua essência, o objetivo da análise de malware é geralmente fornecer as informações que você precisa para responder a uma intrusão de rede. Os objetivos são tipicamente determinar exatamente o que aconteceu, e garantir que todas as máquinas e arquivos infectados foram localizados [Sikorski 2012]. Ao analisar um malware suspeito, o objetivo normalmente será determinar, por exemplo, exatamente o que um binário suspeito em particular pode fazer, como detectá-lo na rede e como medir e conter seus danos.

Existem quatro abordagens fundamentais para a análise de malware: estática, dinâmica, código e memória [Monnappa 2018], [Kleymenov and Thabet 2019]. A análise estática consiste em um processo de analisar um binário sem executá-lo. Este tipo de análise pode confirmar se um arquivo é malicioso, fornecer informações sobre sua funcionalidade e, às vezes, fornecer informações que lhe permitirão produzir assinaturas de rede simples. A análise estática é direta e pode ser rápida, mas pode se demonstrar ineficaz contra malware sofisticado pois pode perder comportamentos importantes.

A análise dinâmica é o processo de execução do binário suspeito em um ambiente isolado e monitorando seu comportamento. Esta técnica de análise é fácil de executar e fornece informações valiosas sobre a atividade do binário durante sua execução. Entretanto, antes de executar o malware com segurança, deve-se configurar um ambiente cuidadosamente, de maneira que lhe permita estudar a execução do malware sem risco de danos ao sistema ou rede.

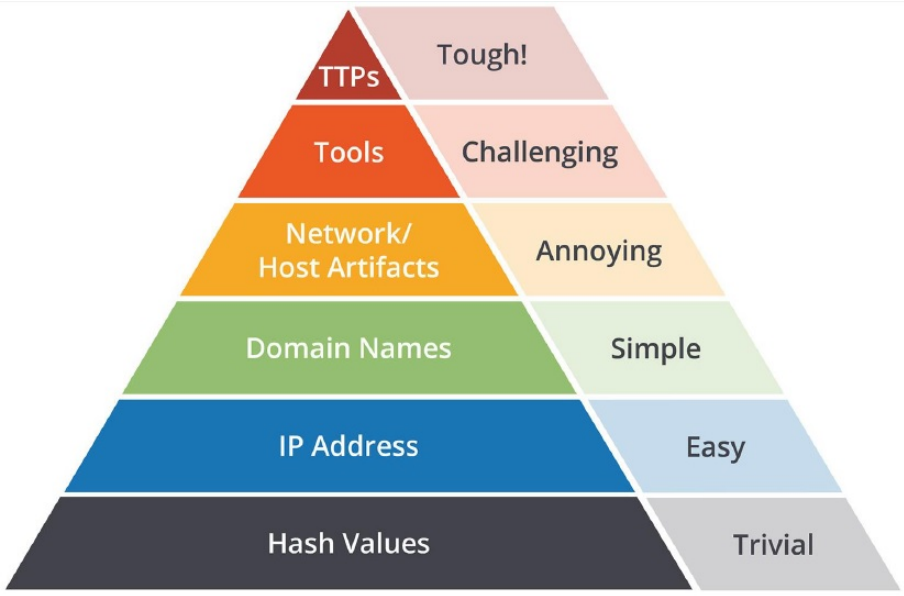
Análise de código é uma técnica que se concentra em analisar o código para entender o funcionamento interno do binário. Esta técnica revela informações que não são possíveis de determinar apenas a partir de análises estáticas ou dinâmicas. Esta análise é dividida em análise de código estática e análise de código dinâmica. A análise de código estática envolve desmontar o binário suspeito e olhar para o código para entender

o comportamento do programa, enquanto a análise de código dinâmica envolve depurar o binário suspeito de maneira controlada para entender sua funcionalidade. Nesta classe de análise é necessária uma compreensão de conceitos de linguagem de programação e sistema operacional.

A técnica de análise de memória consiste em analisar a memória RAM do computador para artefatos forenses. É tipicamente uma técnica forense, mas integrá-lo em uma análise de malware ajudará a obter uma compreensão do comportamento do malware após a infecção. A análise de memória é especialmente útil para determinar os recursos furtivos e evasivos do malware.

À medida que o malware se ofusca cada vez mais e aplica técnicas de anti-análise para impedir nossa análise, precisamos de métodos mais sofisticados que nos permitam levantar a cortina escura projetada para nos manter fora [Andriesse 2018]. De acordo com o tipo de análise realizada em um malware pode-se causar diferentes impactos para o atacante, o que costumamos chamar de Pirâmide da Dor [Bianco 2022]. Na Figura 3.1 observamos que análises triviais como a descoberta do *hash* de um malware causa um impacto mínimo, entretanto a descoberta das táticas, técnicas e procedimentos utilizados pelo malware tem um potencial de gerar um grande impacto para o atacante.

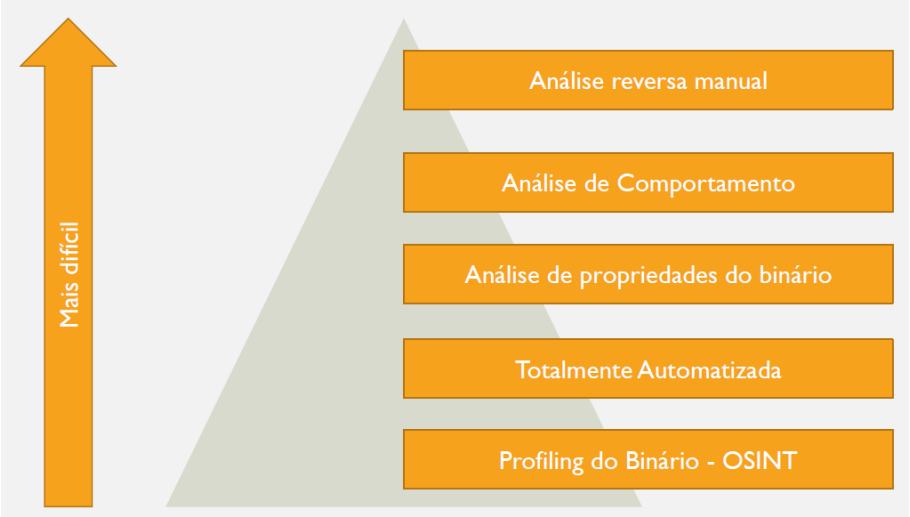
Figura 3.1. Pirâmide da Dor



Source: David J. Bianco, personal blog

Para o processo de análise de malware, os autores utilizaram uma metodologia, que busca, de forma incremental, agregar novas informações sobre o malware em cada estágio, conforme descrito na Figura 3.2. Como pode ser observado, a metodologia compreende os seguintes estágios: *profiling* do binário (OSINT), Análise totalmente automatizada, análise de propriedades do binário, análise de comportamento e análise reversa manual. Na aplicação desta metodologia e com o propósito de melhor exemplificar o uso de técnicas de análise, foi utilizado o malware TextInputDocsx.exe. Este artefato é classificado como um malware bancário e será utilizado na evolução das análises.

Figura 3.2. Metodologia do Processo de Análise de Malware. Fonte:Alissa Torres



Para fazer frente a este cenário atual de contínuas ameaças, precisamos de profissionais de cibersegurança que sejam pelo menos igualmente conhecedores daqueles que cometem esses crimes online. Ser capaz de desenvolver mecanismos adequados de detecção depende da compreensão como esses criminosos operam, como eles entram, o que procuram e como eles exfiltram o que encontraram da infraestrutura [Carvey 2014]. Nas demais seções deste capítulo são abordadas em detalhe a preparação de um laboratório e os estágios da metodologia do processo de análise de malware proposta pelos autores.

3.2. Preparação de um laboratório de análise

A utilização de um ambiente de laboratório devidamente preparado e isolado é um pré-requisito fundamental para análise de artefatos maliciosos. Este tipo de infraestrutura permite a investigação do artefato sem a exposição desnecessária do host do analista e da rede na qual se encontra a riscos de contaminação e prejuízos imprevisíveis. Ao analisar um código de um ransomware no seu próprio host, por exemplo, o analista corre o risco de executá-lo por acidente e ter sua estação de trabalho criptografada - sem contar com o risco de contaminação ou criptografia de outros hosts ou diretórios compartilhados conectados ao host.

Nesta seção, trataremos dos tipos de ambientes de laboratório, suas diferenças, vantagens e desvantagens e detalharemos como os alunos do minicurso devem proceder para a utilização de ambiente virtualizado para a realização das práticas.

3.2.1. Tipos de ambientes de análise

Ambientes de laboratório para análise de malware podem ser compostos por hosts físicos e/ou virtuais desconectados da rede e, de preferência, desconectados da internet. A recomendação de estarem desconectados da rede está ligado a prevenção de contaminação do ambiente e o isolamento da internet tem a ver com o maior controle do processo de análise e do entendimento do comportamento do artefato, conforme veremos mais adiante.

Quanto a questão do sistema ser físico ou virtual, há alguns pontos a serem con-

siderados. Alguns artefatos maliciosos podem fazer verificações do sistema operacional onde está sendo executado. Caso identifique que está em um ambiente virtual, podem não se manifestar ou se manifestar de maneira diferente para despistar o analista. Por outro lado, o ambiente físico apresenta desafios. Um deles é a remoção do malware após a infecção, que pode ser bastante trabalhosa e demorada de ser feita manualmente.Uma alternativa seria a utilização de ferramentas de recuperação da imagem completa do sistema após a infecção. Apesar de facilitar o processo, pode se tornar onerosa do ponto de vista de tempo.

Por estes motivos, é mais comum que o laboratório de análise seja composto por hosts virtuais. Nestes, é possível utilizar o recurso de *snapshot*, o que facilita o processo de execução do malware e de recuperação do estado do sistema de forma bastante rápida. Pesa contra este tipo de ambiente, no entanto, a questão de os malwares poderem "perceber"que estão sendo analisados em um ambiente controlado. Caberá ao analista identificar tais tentativas de identificação e contorná-las de alguma forma.

Por exemplo, o malware poderá tentar identificar hosts virtualizados por meio da leitura de chaves de registro do Windows que estão presentes somente nesta classe de hosts. As chaves de registro 'HKCU\SOFTWARE\VMware, Inc.\VMware Tools' e 'HARDWARE\ACPI \FADT\VBOX__', por exemplo, só estarão presentes em sistemas virtualizados com o VMWare e VirtualBox, respectivamente. Caso o analista perceba a leitura destas chaves e logo após uma decisão de interromper a execução do código, o analista poderá intervir no código invertendo a lógica de uma condicional 'jmp', por exemplo. A Figura 3.3 mostra um exemplo de código realizando a checagem da existência da chave do VirtualBox. Este e outros exemplos de técnicas de evasão podem ser encontrados nos projetos Evasion Techniques[1] do Checkpoint Research e Pafish[2] do pesquisador Alberto Ortega.

3.2.2. Ambiente do minicurso

Para este minicurso, utilizaremos um ambiente composto por dois hosts virtuais, sendo um Windows 10 e outro Linux. No sistema Windows, serão utilizadas ferramentas de análise estática e dinâmica de códigos maliciosos. O objetivo é utilizar o sistema, sobretudo, para a execução e o monitoramento do comportamento do artefato suspeito utilizando ferramentas tais como Process Hacker e Process Monitor. Este monitoramento pode propiciar a identificação, por exemplo, da criação de arquivos ou de mecanismos de persistência.

Já o sistema Linux será utilizado para a simulação de um ambiente de rede com o objetivo de analisar o comportamento de comunicação do artefato com hosts externos. É bastante comum que os códigos maliciosos estabeleçam conexões com hosts externos para download de novos componentes, para a exfiltração de dados ou somente para avisar o atacante que mais uma vítima foi contaminada.

Tanto a criação de arquivos como a tentativa de comunicação externa pelo malware podem ser úteis para o estabelecimento dos indicadores de comprometimento (IOCs) do artefato analisado. Estes IOCs são de grande valia na identificação do escopo de um incidente pois viabilizam a identificação de outros hosts contaminados.

[1]https://evasions.checkpoint.com/

[2]https://github.com/a0rtega/pafish

Figura 3.3. Exemplo de código malicioso buscando pela chave de registro do VirtualBox

```
/* sample of usage: see detection of VirtualBox in the table below to check registry path */
int vbox_reg_key7() {
    return pafish_exists_regkey(HKEY_LOCAL_MACHINE, "HARDWARE\\ACPI\\FADT\\VBOX__");
}

/* code is taken from "pafish" project, see references on the parent page */
int pafish_exists_regkey(HKEY hKey, char * regkey_s) {
    HKEY regkey;
    LONG ret;

    /* regkey_s == "HARDWARE\\ACPI\\FADT\\VBOX__"; */
    if (pafish_iswow64()) {
        ret = RegOpenKeyEx(hKey, regkey_s, 0, KEY_READ | KEY_WOW64_64KEY, &regkey);
    }
    else {
        ret = RegOpenKeyEx(hKey, regkey_s, 0, KEY_READ, &regkey);
    }

    if (ret == ERROR_SUCCESS) {
        RegCloseKey(regkey);
        return TRUE;
    }
    else
        return FALSE;
}
```

A seguir, o detalhamento de ferramentas e máquinas virtuais necessárias para o minicurso:

- **Sistema de virtualização**: Recomendamos a utilização do VMware Workstation Pro 14 (ou superior) ou VMware Fusion Pro 10 (ou superior) dependendo do sistema operacional Windows ou OS X, respectivamente. Ambos são versões comerciais. Caso o computador não possua a licença comercial do VMWare, é possível fazer o download de versões de avaliação por 30 dias. As versões Pro são necessárias uma vez que suportam a criação de snapshots - recurso essencial durante as práticas do mini-curso.

  Versões de avaliação do VMWare Pro estão disponíveis em:

  – https://www.vmware.com/products/workstation-pro.html

  – https://www.vmware.com/my/products/fusion-pro.html

  Observação importante: Computadores Apple com o processador M1 não suportam os recursos de virtualização necessários e não devem ser utilizados neste mini-curso.

- **Máquina Virtual Linux**: Para a máquina virtual Linux, recomendamos o uso da distribuição REMnux© [3]. O REMnux© é um kit de ferramentas para engenharia reversa e análise de códigos maliciosos e provê uma lista de ferramentas gratuitas criadas pela comunidade. O uso desta distribuição, portanto, facilita o trabalho do

[3]https://remnux.org/

analista uma vez que não precisa se preocupar em instalar as ferramentas de forma independente.

As instruções de instalação e o link para download do REMnux© estão disponíveis em:

  - https://docs.remnux.org/install-distro/get-virtual-appliance.

- **Máquina Virtual Windows**:

  Para máquina virtual Windows, disponibilizamos um projeto chamado Win Reverse VM[4] cujo objetivo é facilitar a preparação de uma VM Windows para análise de malware.

  O projeto utiliza como base uma VM Windows 10 disponibilizada pela Microsoft por um período de avaliação. O projeto Win Reverse VM instala na VM ferramentas comumente utilizadas na análise estática e dinâmica de artefatos maliciosos. Estão na lista: Process Monitor, Process Hacker, API Monitor, x64dbg, Detect it easy, dentre outras.

  Siga as instruções de instalação do projeto para ter a sua VM Windows:

  - https://github.com/rrmarinho/winreversevm#installation

3.2.3. Configurações de isolamento

Uma vez que as VMs Linux e Windows tenham sido instaladas, certifique-se de ajustar as configurações de rede e internet de forma que:

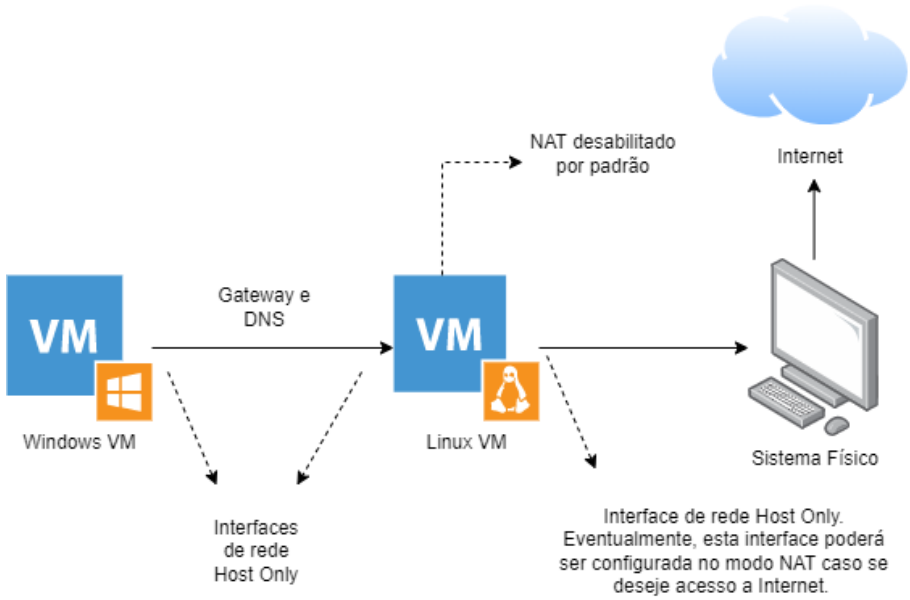
- A VM Linux tenha duas interfaces de rede - ambas em modo 'Host Only'. Uma das interfaces será utilizada para a comunicação com o host Windows e a outra para comunicação com o host hospedeiro (seu sistema operacional). A interface de comunicação com o host hospedeiro poderá ser alterada para o modo NAT caso precise de internet no host Linux ou, até mesmo, no host Windows. Para o segundo caso, será necessário o uso de NAT no Linux para habilitar a saída do Windows;
- A VM Windows tenha uma interface de rede em modo 'Host Only';
- A VM Windows tenha as configurações de Gateway e DNS apontando para o endereço IP de uma das interfaces 'Host Only' da VM Linux;
- As duas VMs não estejam saindo para a Internet.

A Figura 3.4 demonstra como deve ser montado o ambiente do laboratório.

Ao final do processo de criação das VMs Windows e Linux, crie um *snapshot* de cada das VMs para preservar o estado inicial.

[4]https://github.com/rrmarinho/winreversevm

Figura 3.4. Diagrama do Laboratório de Análise de Malware



3.3. Profiling do artefato suspeito (OSINT)

Na metodologia de análise de códigos suspeitos, a busca por informações sobre um artefato em repositórios públicos utilizando OSINT (*Open Source Intelligence*) pode fornecer desde indícios de que estejamos realmente diante de um código malicioso a análises completas, incluindo comportamento, análise reversa e indicadores de comprometimento (IOCs). Em situações de resposta a incidentes, por exemplo, onde informações tempestivas são cruciais, este atalho na análise pode fazer toda a diferença na tomada de decisões, como o isolamento de um sistema infectado ou a troca de credenciais de um usuário vítima de um *phishing*, para citar dois exemplos. Desta forma, exigindo pouco esforço e tempo do analista, a etapa de *profiling* via OSINT é, muitas vezes, a mais indicada para o início de uma análise.

Por outro lado, é importante notar que a fase de *profiling* pode não ser suficientemente rica para eliminar a necessidade de aprofundamento da análise do artefato. Alguns *samples*, como chamamos uma amostra de um *malware*, podem não ter sido analisados anteriormente ou as informações disponíveis a seu respeito podem ser muito resumidas.

Nesta seção, abordaremos o uso de técnicas de OSINT para *profiling* de artefatos suspeitos com base nos resultados do VirusTotal [5] utilizando o binário 'TextInputDocsx.exe' como exemplo. Existem muitas outras ferramentas que podem ser utilizadas, sobretudo em conjunto, neste processo, tais como Malpedia [6], Open Threat Exchange (OTX) [7], Malware Hash Registry (MHR) [8], Hybrid Analysis [9], Any.run [10] e Intezer [11].

[5] https://virustotal.com
[6] https://malpedia.caad.fkie.fraunhofer.de/
[7] https://otx.alienvault.com/
[8] https://team-cymru.com/community-services/mhr/
[9] https://www.hybrid-analysis.com/
[10] https://any.run/
[11] https://www.intezer.com/

No entanto, decidimos focar no VirusTotal por ser uma das mais abrangentes e importantes para esta finalidade. Este mesmo processo pode ser replicado para outros samples desconhecidos de seu interesse.

3.3.1. Identificação do tipo de ameaça

Uma das primeiras perguntas a serem respondidas a respeito de uma ameaça é a sua finalidade. Por exemplo, o código teria sido criado para roubar informações (*malware* do tipo *information stealer* ou simplesmente *infostealer*), seria um *backdoor* (código que oferece uma porta de entrada alternativa ao atacante) ou um ransomware? Esta pergunta está diretamente ligada ao entendimento das capacidades da ameaça e, consequentemente, às ações que devem ser tomadas em resposta a sua presença no sistema.

Nem sempre as ameaças podem ser classificadas em um único tipo. Na verdade, é comum que ameaças somem características de múltiplos tipos. Por exemplo, podem somar características de códigos que roubam informações com códigos que realizam a criptografia de disco. Desta forma, dependendo da profundidade de informações disponíveis sobre a ameaça, deveremos ter em mente que o tipo de malware apontado pode refletir somente a ação principal da ameaça. O artefato pode tomar outras ações que só podem ser descobertas com análises mais aprofundadas.

Conforme mencionado anteriormente, utilizaremos o VirusTotal na tentativa de identificação do tipo da ameaça. O VirusTotal é uma ferramenta que inspeciona o arquivo submetido com múltiplos motores de análise de malware ou *engines* (72 no momento) de anti-vírus de diferentes fabricantes. Caso o item submetido seja uma URL ou domínio de internet, a ferramenta buscará pelo item em *blocklists*. Cada *engine* analisará o artefato e, caso o identifique como malicioso, indicará um rótulo para a ameaça detectada (ex: Trojan.BitCoinMiner). Além dos rótulos, o VirusTotal poderá oferecer análises adicionais com base na execução do artefato em *sandboxes*.

O VirusTotal fornece três opções de uso da ferramenta no seu *website*:

1. FILE: utilizando a opção 'FILE', o usuário da ferramenta submete o arquivo para análise pelas múltiplas *engines*. É muito importante que o usuário esteja atento ao aviso do VirusTotal de que, ao fazer isso, estará compartilhando o arquivo com os assinantes da plataforma VirusTotal. Isso significa que o usuário poderá correr o risco de ter dados sigilosos expostos a terceiros. Por isso, esta não seria a opção inicial mais adequada;

2. URL: esta opção permite que o usuário forneça uma URL que será avaliada por múltiplas *engines* e *blocklists*. Esta consulta também será compartilhada com os assinantes da plataforma;

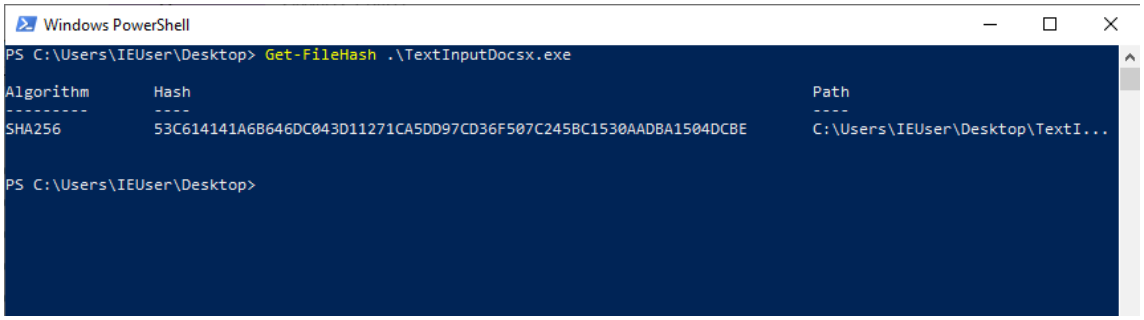
3. SEARCH: Nesta opção, o usuário tem a possibilidade de fazer uma busca na plataforma por análises prévias de URLs/domínios, endereços IP e *hashes* de arquivos. Observe que, ao invés de submeter o arquivo em si, podemos submeter o identificador *hash* do arquivo. Caso o arquivo já tenha sido submetido por outro usuário, você terá acesso ao resultado da análise.

Como pode ser observado, a opção SEARCH é a mais adequada para o início da etapa de *profiling* do artefato utilizando o VirusTotal.

Para ilustrar como o VirusTotal pode auxiliar na identificação do tipo da ameaça, vamos submeter à ferramenta o *hash* MD5 do binário 'TextInputDocsx.exe' à plataforma. Para calcular o *hash* do binário, há diferentes opções. Na VM Windows, pode utilizar o utilitário DIE ou simplesmente um PowerShell *Get-FileHash*. A seguir, como o comando PowerShell pode ser utilizado:

1. Abrir um prompt de comando PowerShell;
2. Mudar o diretório para o local onde o binário 'TextInputDocsx.exe' encontra-se no disco;
3. Executar o comando: GetFileHash .\TextInputDocsx.exe. Por padrão, o algoritmo *hash* SHA256 será utilizado. O SHA256 pode ser utilizado na busca do VirusTotal. No entanto, caso queira calcular com o algoritmo MD5, basta passar o parâmetro -a MD5. Assim, teríamos: GetFileHash -a MD5 .\TextInputDocsx.exe. Na Figura 3.5, temos o exemplo de uso do comando.

Figura 3.5. Cálculo do hash do binário TextInputDocsx.exe usando o comando Get-FileHash

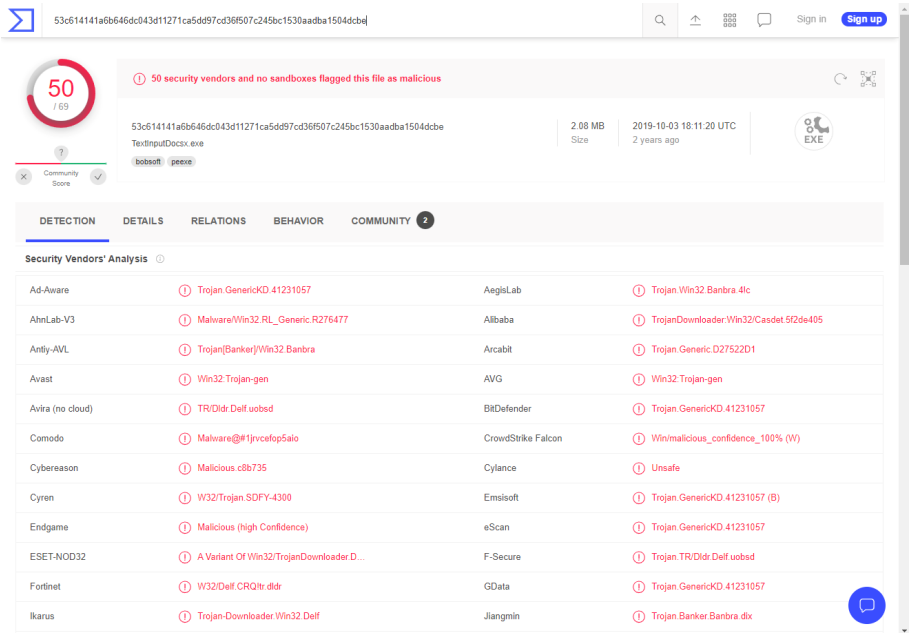


O *hash* SHA256 resultante é o: 53C614141A6B646DC043D11271CA5DD97CD36F507C245BC1530AADBA1504DCBE. Utilizando este código na busca do VirusTotal, tivemos o resultado apresentado na Figura 3.6.

A partir desta resultado, podemos observar:

- O VirusTotal nos informa que 50 de 69 *engines* consideraram o arquivo malicioso. Quando um número muito baixo de *engines* consideram o arquivo malicioso, há uma chance de entendermos que o arquivo seja malicioso mesmo que ele não seja. No entanto, quando temos um grande número de *engines* apontando que o arquivo é malicioso, como neste caso, a chance de cairmos num falso positivo é bem menor;
- A análise mais recente do VirusTotal para este arquivo foi no dia 03/10/2019 às 18:11:20 UTC. Este indicador pode nos dizer que esta ameaça não é algo novo,

Figura 3.6. Resultados do VirusTotal para o hash do arquivo TextInputDocsx.exe



ou, pelo menos, que não há uma campanha abrangente em curso utilizando esta ameaça. Para ameaças utilizadas em campanhas abrangentes ou genéricas, é mais comum termos uma análise recente da ameaça no VirusTotal;

- Analisando os rótulos associados pelas *engines* de antivírus, encontramos múltiplos deles indicando que o binário trata-se de um Trojan. Adicionalmente, 5 dos rótulos indicam que o artefato seria um *Banker*, ou seja, um malware bancário cuja finalidade principal é o roubo de dados financeiros das vítimas. Por fim, 8 das engines indicam no rótulo a palavra *Banbra*[12]. Este rótulo indica que este é um malware projetado para roubar informações pessoais de clientes de bancos brasileiros.

3.3.2. Identificação de capacidades da ameaça

Em suma, até aqui, com a simples consulta do *hash* do binário no VirusTotal, pudemos entender, a partir de um repositório compartilhado de dados, que o binário suspeito trata-se de um **trojan bancário com alvo em clientes de bancos brasileiros**.

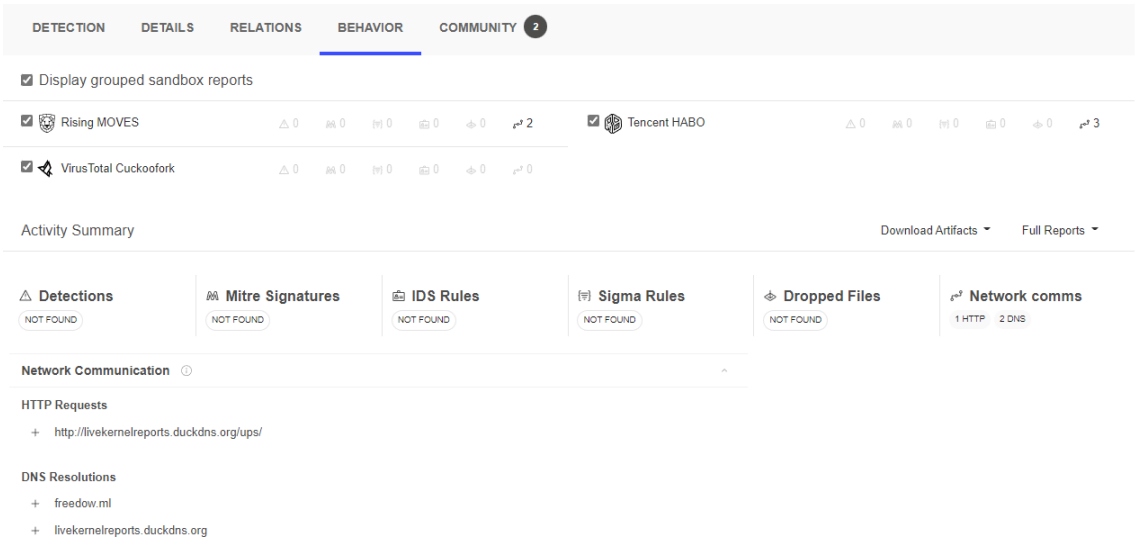
Mas há ainda mais possibilidades de descobertas a partir do VirusTotal. Uma delas é a possível descoberta de indicadores de comprometimento (IOCs). IOC, em cibersegurança, é um artefato observado na rede ou em um sistema que, com alta confiança, indica o comprometimento por uma ameaça em particular [Gragido 2012]. Caso estejam disponíveis, estarão principalmente nas abas 'BEHAVIOR' e 'RELATIONS'.

Em 'BEHAVIOR', teremos acesso aos resultados da execução do binário em uma ou mais *sandboxes*. Na Figura 3.7, podemos observar a aba 'BEHAVIOR' para o nosso artefato 'TextInputDocsx.exe' na qual há indicadores de comunicações de rede: uma conexão HTTP e 2 conexões DNS. Voltando ao tema de identificar IOCs importantes para

[12]https://threats.kaspersky.com/br/threat/Trojan-Banker.Win32.Banbra/

o nosso artefato, certamente estes são de grande valia. Alguém que queira buscar por computadores que tenham executado a ameaça, pode buscar por computadores que tenham tentado resolver os nomes freedow.ml e livekernelreports.duckdns.org a partir dos registros de log do DNS Server e pode buscar por computadores que tenham enviado a comunicação HTTP para a URL http://livekernelreports.duckdns.org/ups/.

Figura 3.7. Aba BEHAVIOR do VirusTotal para o binário TextInputDocsx.exe

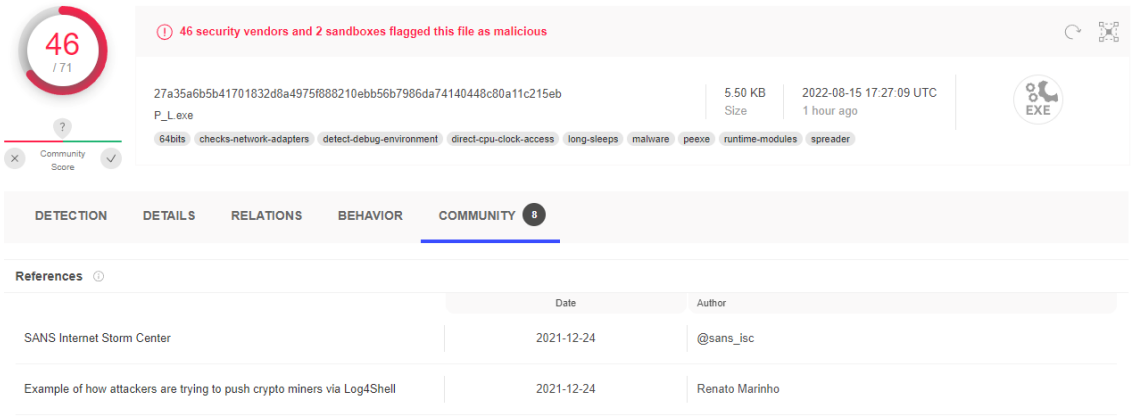


Na aba 'RELATIONS', você poderá ter acesso a análise dos IOCs desta ameaça pelo VirusTotal de forma individualizada. Para o nosso exemplo, teremos acesso nesta aba às detecções dos domínios resolvidos nas consultas DNS e da URL acessada via HTTP. Adicionalmente, teremos acesso aos endereços IP para os quais as consultas DNS foram resolvidas no momento da execução do malware na sandbox. A informação dos endereços IP agregam aos IOCs de domínio e URL já observados.

Por fim o VirusTotal oferece dados que podem ser igualmente relevantes para a nosso objetivo de *profiling*. Na aba 'COMMUNITY', o VirusTotal apresenta possíveis comentários e referências externas citando a ameaça. Para ter acesso a informações mais completas desta aba, crie uma conta gratuita no VirusTotal. Para o exemplo do binário 'TextInputDocsx.exe', não há referências externas contendo análises. No entanto, para ilustrar as possibilidades, vamos considerar a análise de um outro binário cujo *hash* MD5 é e74ec7381bf7020ec707bd722afe6d38. A Figura 3.8 mostra a aba 'COMMUNITY' com links para uma análise da ameaça realizada por Renato Marinho cujo título é *Example of how attackers are trying to push crypto miners via Log4Shell* [13]. Ao acessar referências externas, poderá ter ainda mais indicadores ou mesmo um entendimento mais abrangente das capacidades da ameaça, como neste caso.

[13]https://morphuslabs.com/example-of-how-attackers-are-trying-to-push-crypto-miners-via-log4shell-294ec2a5cf43

Figura 3.8. Exemplo de referências externas para uma ameaça no VirusTotal



3.4. Análise automatizada

Neste seção, serão abordados conceitos e ferramentas para a realização de análise automatizada de artefatos suspeitos. Ferramentas de análise automatizada podem ser bastante úteis uma vez que, sem muito esforço por parte do analista, informações valiosas de comportamentos maliciosos, como, por exemplo, a criação de uma chave de registro que fará com que o ameaça se torne persistente no sistema operacional (autoexecutável no *reboot*), possam ser obtidas em um curto espaço de tempo. Entretanto, diferentes plataformas apresentam especificidades quanto ao alvo da execução em sandbox. Como exemplo, enquanto na plataforma Windows exemplares de malware afetam o subsistema de chaves de registro [Botacin et al. 2018], na plataforma Linux, exemplares de malware afetam o subsistema de arquivo /proc [Galante et al. 2018].

No entanto, os ambientes automatizados também apresentam limitações. Algumas ameaças implementam técnicas *anti-sandbox* e não manifestam suas ações maliciosas ao perceberem que estão em um ambiente de análise. Adicionalmente, o ambiente de *sandbox* pode não apresentar todos os recursos ou estímulos necessários para que o artefato malicioso manifeste seus comportamentos. Por exemplo, a ameaça pode fazer um teste da linguagem do teclado do usuário. Caso não esteja em ptBR, não executará o estágio seguinte. Outros comportamentos mais avançados podem exigir a comunicação com um servidor remoto de comando e controle (C2), o que poderá não estar disponível em um ambiente de *sandbox*. Apesar destas limitações, dada a boa relação esforço/benefício, ou seja, baixo esforço para o potencial benefício, esta etapa pode ser considerada nas etapas iniciais da análise de um artefato suspeito.

3.4.1. Funcionamento da análise automatizada

A análise automatizada consiste na execução de arquivos suspeitos em ambientes controlados (*sandboxes*) com o objetivo de identificar comportamentos maliciosos. Enquanto o arquivo suspeito é executado, o ambiente da *sandbox* monitora cada evento gerado pelo artefato no sistema e busca identificar comportamentos potencialmente maliciosos, como a mudança de um registro do Windows que faz com que algo seja executado na iniciali-

zação do sistema (técnica de persistência).

Finalizada a execução do artefato, o sistema que provê a *sandbox* emitirá um relatório das ações observadas com destaque especial para aquelas potencialmente maliciosas. Adicionalmente, os relatórios apresentam uma lista de indicadores de comprometimento (IOCs), tais como endereços IP ou nomes de *hosts* consultados durante a execução, arquivos ou registros criados/acessados no sistema, etc. No geral, os relatórios trazem também um percentual indicando um grau de 'certeza' de que aquele artefato seria malicioso com base nos seus comportamentos.

Caso o resultado indique que o artefato é malicioso e aponte suas características, o analista já terá informações valiosas para a tomada de decisões sobre os próximos passos na estratégia de defesa. Caso o resultado aponte que o artefato não é malicioso, é importante ter em mente que este resultado pode não refletir a realidade. Alguns artefatos maliciosos podem identificar que estão em um ambiente controlado e podem não manifestar suas ações ou podem reagir com ações que parecem legítimas para despistar os resultados da *sandbox*, conhecidas como técnicas *anti-sandbox*. Neste caso, o mais indicado é que o analista busque avançar em análises mais aprofundadas para descobrir se o artefato é malicioso ou não.

Há esforços da comunidade acadêmica para minimizar as chances de detecção do ambiente controlado. Em [Liu et al. 2022], os autores propõem a emulação de comportamentos de usuários reais no sistema. Já a pesquisa [Mills and Legg 2020] investiga as técnicas *anti-evasion* por meio de reconfiguração da *sandbox*.

3.4.2. Tipos de sandboxes

Há uma grande variedade de ferramentas para análise automatizada de malware com o uso de *sandboxes*. Essas ferramentas podem ser divididas em dois grupos principais: *sandboxes* públicas e *sandboxes* privadas.

As *sandboxes* públicas são aquelas fornecidas por provedores de serviços em cibersegurança que disponibilizam suas *sandboxes* para usuários via internet. A maior parte destas plataformas aplicam alguma limitação na versão gratuita da *sandbox*, como, por exemplo, o tempo que o binário suspeito passará em execução.

As *sandboxes* públicas apresentam vantagens como a facilidade de uso e a disponibilidade imediata. Como desvantagens, podemos citar as limitações para as versões gratuitas e, principalmente, a necessidade da submissão do *sample* a ser analisado. Nestes casos, é importante observar se o arquivo terá que ficar disponível para terceiros.

A seguir, uma lista das principais ferramentas de *sandbox* disponíveis online:

- Any.run (https://any.run)
- Intezer Analyze (https://analyze.intezer.com/)
- Hybrid Analysis (https://www.hybrid-analysis.com/)
- Joe Sandbox Cloud (https://www.joesandbox.com/)
- FileScan.IO (https://www.filescan.io/)

Do outro lado estão as *sandboxes* privadas. Existem as soluções corporativas e soluções *open-source*. As soluções corporativas, geralmente fornecidas por provedores de soluções de anti-malware, tem a vantagem da facilidade de implementação e do suporte ao usuário, mas pode ser mais limitada quanto a customizações. Já a solução open-source, pode apresentar desafios de implantação e pode ser mais limitada em recursos, mas tem a grande vantagem da customização.

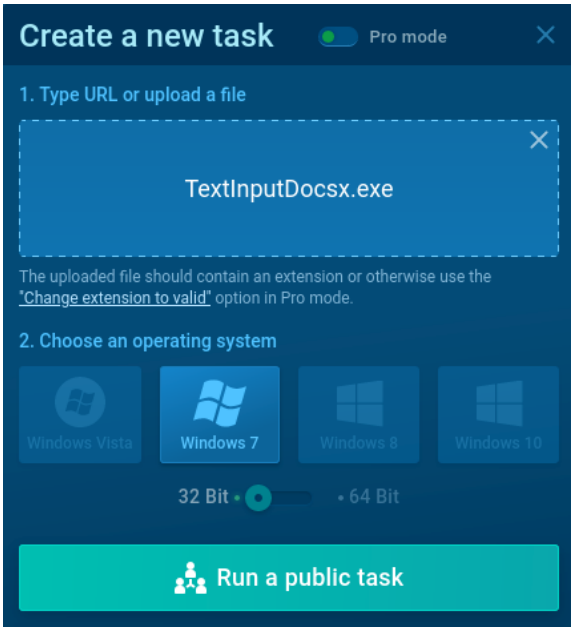
Uma ferramenta de *sandbox* open-source bastante popular é a Cuckoo Sandbox[14]. A ferramenta pode analisar diferentes tipos de arquivos (executáveis, documentos office, arquivos PDF, e-mails, etc), bem como websites maliciosos nas plataformas Windows, Linux, macOS e Android por meio de ambientes virtualizados.

3.4.3. Analisando nosso artefato utilizando sandboxes

Utilizaremos a *sandbox* Any.run para analisar o nosso binário de exemplo (TextInputDocsx.exe). Para isso, tivemos que criar uma conta gratuita na plataforma para, então submeter o arquivo. Seguimos estes passos utilizando a VM Linux, uma vez que é mais fácil ter acesso à internet por meio desta plataforma. Lembramos que não é recomendável descompactar o binário malicioso no seu sistema real por dois principais motivos: primeiro que o seu antivírus poderá reconhecê-lo como malicioso e apagá-lo. Segundo porque você pode clicar no binário por acidente descompactado e executar o malware no seu host.

Ao submetermos o binário ao Any.run, verificamos as limitações da versão gratuita: execução somente no Windows 7 e na arquitetura 32bits e a disponibilização dos dados para acesso público, conforme Figuras 3.9 e 3.10.

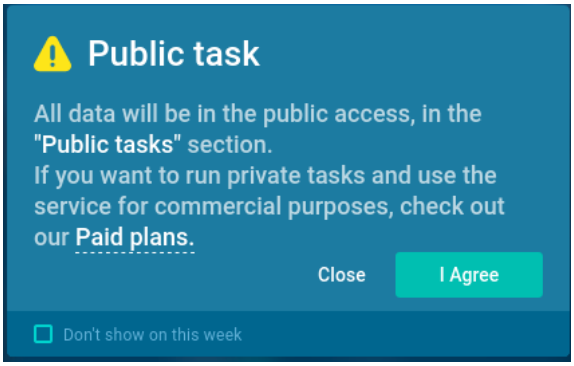
Figura 3.9. Limitações de execução da versão grátis da Sandbox Any.run



Como resultados para a análise, podemos destacar:

[14]https://cuckoosandbox.org/

Figura 3.10. Disponibilização dos dados submetidos para acesso público



- As consultas DNS para os domínios livekernelreports.duckdns.org e freedow.ml;
- Tentativas de conexão ao endereço IP 188.114.97.3 (resolvido para a consulta do domínio freedow.ml) na porta TCP/26457;
- Aviso de ameaça em potencial devido a tentativa de resolução de um domínio suspeito (.ml) e um domínio hospedado em um serviço de DNS dinâmico (*.duckdns);
- Há a mudança/criação do arquivo 'C:\Users\admin\AppData\Roaming\Progrestime.html';
- O artefato faz a leitura de chaves de registro para verificar os idiomas suportados bem como para obter o nome do computador;

Como resultado final, apesar de alguns indicadores potencialmente maliciosos, não há um veredito da plataforma. A análise do *sample* 'TextInputDocsx.exe' atingiu uma pontuação de 30 de um total de 100, o que não foi suficiente para apontar que o binário é malicioso.

Analisamos também o artefato utilizando a *sandbox* Hybrid Analysis e nesta o resultado apontou que o binário é malicioso. A pontuação desta *sandbox*, chamada de *Threat Score* atingiu 100 de 100 possíveis. Como destaques para o relatório gerado, temos:

- O *sample* foi considerado malicioso por uma grande quantidade de *engines* de antivírus;
- O processo submete arquivos a um *web server*;
- O processo aloca memória em processos remotos;
- Utiliza URLs identificadas como maliciosas por pelo menos uma *engine* de reputação;
- Por fim e não menos importante, que o processo tem a habilidade de capturar as teclas digitadas no teclado. Esta é uma característica comum de códigos maliciosos que visam o roubo de informações.

Diferentemente da Any.run, a Hybrid Analysis chegou a veredito de que o artefato é malicioso e apresentou pelo menos um IOC a mais, que foi a tentativa de conexão HTTP a URL 'livekernelreports.duckdns.org/ups/'.

Com isso, observamos que a análise do nosso binário com o uso destas duas *sandboxes* gratuitas agregou algumas novas informações e indicadores ao que já havíamos conseguido de informações na fase de *profiling*. Sabemos agora que o processo busca a alocação de memória em outros processos numa possível tentativa de persistência e evasão de defesas e que tem a habilidade de capturar teclas digitadas pelo usuário.

3.5. Análise estática

A análise estática é fundamental para uma triagem bem sucedida de um artefato desconhecido. A partir dela é possível identificar indicadores estáticos e propriedades suspeitas do artefato e fornecer um direcionamento para as próximas fases de análise: a comportamental e a reversa. Essa fase pode ter etapas prejudicadas por ofuscações existentes no artefato, tais como o uso de *packers*, que são softwares que ofuscam o conteúdo através do uso de criptografia ou compressão.

Essa fase é essencialmente importante em cenários de incidentes no qual o sample a ser analisado não consta em bases já conhecidas de malwares, podendo ser uma ameaça completamente nova ou apenas a modificação ou variante de uma já existente. Nas seções que seguem, será explicitada cada etapa de análise estática a ser realizada em um artefato.

3.5.1. Identificação do tipo de arquivo

Uma das primeiras análises estáticas que deve ser feita em um artefato desconhecido, é identificar seu tipo de arquivo. Todos os arquivos são compostos de zeros e uns e sua diferenciação está no seu formato, ou padrão de construção. Um programa que aceita um tipo de arquivo, como um editor de imagens ou reprodutor de áudio, realiza um processo de *parsing* no arquivo para ler seu conteúdo.

Arquivos executáveis são lidos e interpretados por um programa que reside no sistema operacional, chamado de *loader*. O loader irá preparar os requisitos de endereçamento virtual, memória, registradores e dependências para a criação de um processo, ou *imagem*, do executável.

Arquivos *EXE* e *DLL*, por exemplo, utilizam o formato **Portable Executable (PE)**, enquanto que sistemas Linux e BSD utilizam **Executable and Linking Format (ELF)** e o sistema macOS faz uso do **Mach object file format (Mach-O)**.

Na Figura 3.11, está exemplificado as estruturas que compõem um binario PE. Nela, é possível perceber que

Abaixo, na Figura 3.12, podemos identificar a composição dos bytes de um arquivo PE em disco. Algumas estruturas importantes são o *DOS Header*, que funciona como uma camada de compatibilidade com o *MS-DOS*, o DOS Stub, que é um pequeno programa em MS-DOS executado apenas quando o programa está rodando no MS-DOS e o PE Header, que conterá metadados necessários para instruir o loader a como carregar o binário.

Figura 3.11. Estrutura geral de um *Portable Executable*

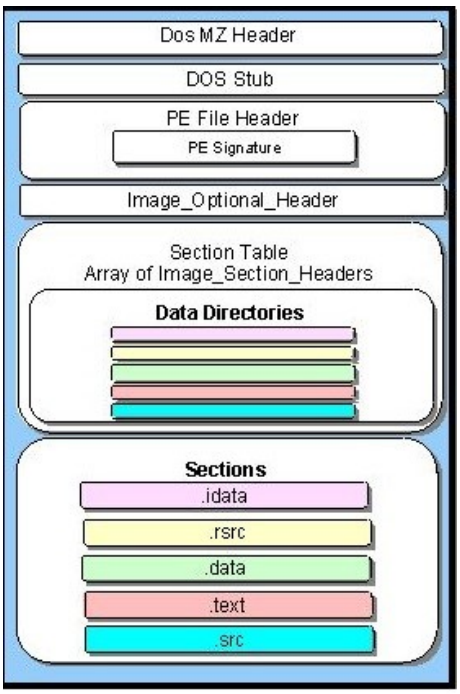
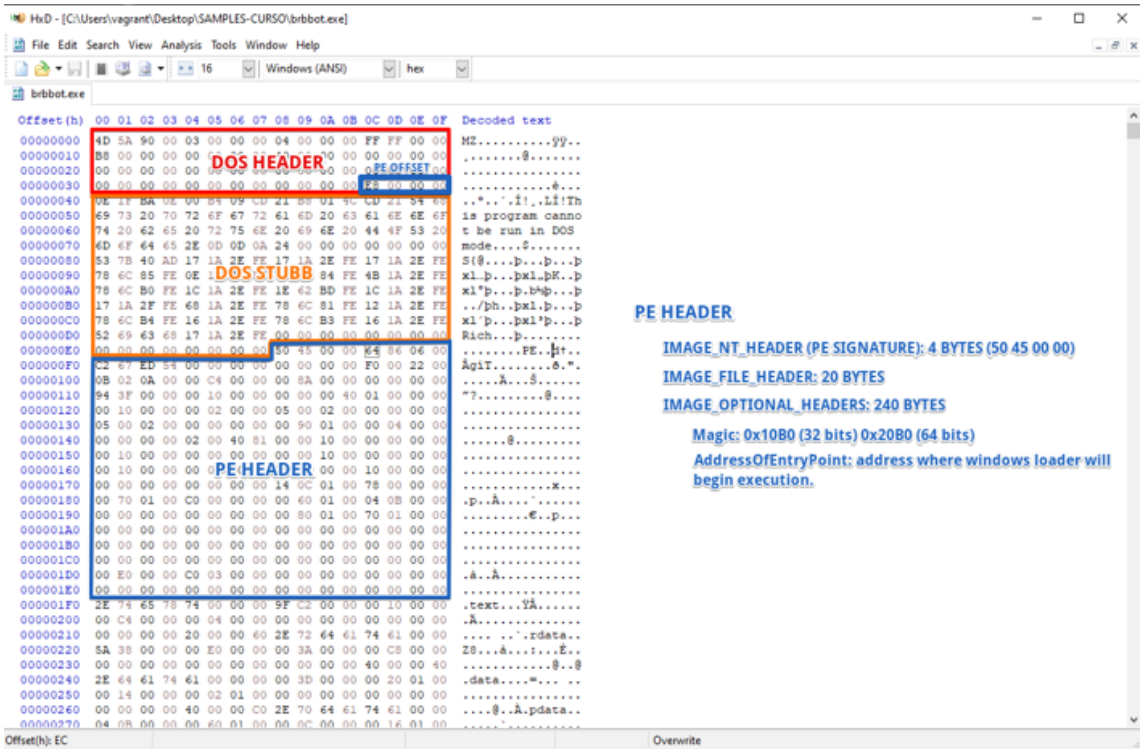


Figura 3.12. Estrutura geral de um *Portable Executable*



Todo arquivo PE começa com os bytes "MZ", e nos quatro bytes (também chamados de *DWORD*) começando no *offset* 0x3C, há o campo *e_lfanew* do DOS Header. Esse

valor, armazenado em *little endian*, é a localização em disco para a *PE Signature*, que são os bytes "PE\0\0", e identificam aquele arquivo como um *Portable Executable*. A PE Signature está contida no *IMAGE_NT _HEADERS*, um cabeçalho que abrange todos os demais cabeçalhos referentes ao formato.

Imediatamente após a assinatura PE, há o início do cabeçalho *IMAGE_FILE _HEADERS*[15], que contém 20 bytes de tamanho, responsável por armazenar informações da arquitetura alvo, o número de seções, o timestamp de compilação, além de informações que auxiliam a construção dos demais headers.

```
typedef struct _IMAGE_FILE_HEADER {
  WORD  Machine;
  WORD  NumberOfSections;
  DWORD TimeDateStamp;
  DWORD PointerToSymbolTable;
  DWORD NumberOfSymbols;
  WORD  SizeOfOptionalHeader;
  WORD  Characteristics;
} IMAGE_FILE_HEADER, *PIMAGE_FILE_HEADER;
```

Após o *IMAGE_FILE_HEADERS*, há o *IMAGE_OPTIONAL_HEADER*[16], de tamanho variável, definido pelo campo *SizeOfOptionalHeader* do cabeçalho anterior. Os campos mais úteis uma análise de malware estão na listagem abaixo.

```
typedef struct _IMAGE_OPTIONAL_HEADER {
...
  DWORD                AddressOfEntryPoint;
...
  DWORD                ImageBase;
...
  WORD                 Subsystem;
  WORD                 DllCharacteristics;
...
  IMAGE_DATA_DIRECTORY DataDirectory
  [IMAGE_NUMBEROF_DIRECTORY_ENTRIES];
} IMAGE_OPTIONAL_HEADER32, *PIMAGE_OPTIONAL_HEADER32;
```

O campo *AddressOfEntryPoint* conterá o *offset*, que será somado ao valor do *ImageBase* o, e essa soma irá resultar no endereço virtual do primeiro byte a ser executado durante o carregamento do EXE. Em DLLs, não é necessário um endereço de entrada, visto que elas são carregadas de maneira diferente que EXEs.

Os campos *Subsystem* e *DllCharacteristics* definem, respectivamente, qual o subsistema de execução, por exemplo, se a aplicação roda com gráficos ou apenas modo

[15]https://docs.microsoft.com/en-us/windows/win32/api/winnt/nswinntimage_file_header#syntax
[16]https://docs.microsoft.com/en-us/windows/win32/api/winnt/nswinntimage_optional_header32#syntax

texto, e informações sobre proteções de carregamento de códigos em endereços de dados (*Data Prevention Execution - DEP*) e randomização de endereços de carregamento (*Address Space Layout Randomization - ASLR*). Por fim, o campo *DataDirectory* definirá um array fixo de diretórios de dados, que são tabelas que contém metadados de importação, exportação, relocação, entre outros. É nessa estrutura que serão descritas os endereços e tamanhos de cada um dos diretórios de dados.

As seções são regiões de bytes bem definidos que têm um propósito específico quando carregadas, como execução de códigos e armazenamento de dados inicializados, não inicializados e somente leitura, para citar algumas. Os nomes e funcionamentos das seções podem variar de compilador para compilador, mas cada uma delas terá informações de tamanho em disco e virtual e permissões para as páginas de memória (leitura, escrita e execução). Na listagem abaixo, segue o nome das principais seções presentes na maioria dos binários PE e uma descrição sobre sua funcionalidade.

- **.text**: contém instruções codificadas para uma determinada arquitetura, geralmente com permissões de leitura e execução;
- **.rdata**: armazena dados inicializados que são somente leitura;
- **.data**: armazena dados inicializados, como variáveis globais, que permitem leitura e sobrescrita;
- **.idata**: quando presente, conterá informações de funções importadas;
- **.edata**: guarda informações sobre funções exportadas pelo binário;
- **.pdata**: apenas presente em binários 64-bits, guarda informações sobre tratamento de exceções;
- **.rsrc**: armazena informações sobre imagens, strings, ícones e outros recursos embutidos em um binário.

3.5.2. Packers

Um *packer* é um software que irá modificar o conteúdo de um executável, comprimindo ou ofuscando seções de código e de dados. Ao realizar esse procedimento, o packer também colocará um *stub*, código a ser executado para a descompressão ou desofuscação do código original em runtime, e irá alterar o entry point do binário para o endereço desse procedimento. Quando o binário é executado, há a desofuscação do código original através da execução do stub, e após esse procedimento, o stub irá alterar o fluxo de código para o código original.

Packers conhecidos, como o *Ultimate Packer for eXecutables (UPX)*[17], contém assinaturas catalogadas por soluções antivírus e identificadores de tipos de arquivos. Através da detecção do packer, é possível aplicar algumas técnicas para retirá-lo do binário, que vão desde buscar por softwares *unpacker*, que realizam a desofuscação do código original

[17] https://github.com/upx/upx

e retiram o *stub*, até o *unpacking* manual do código, buscando executar e encontrar o fim do *stub* e o entry point original, através de análise reversa.

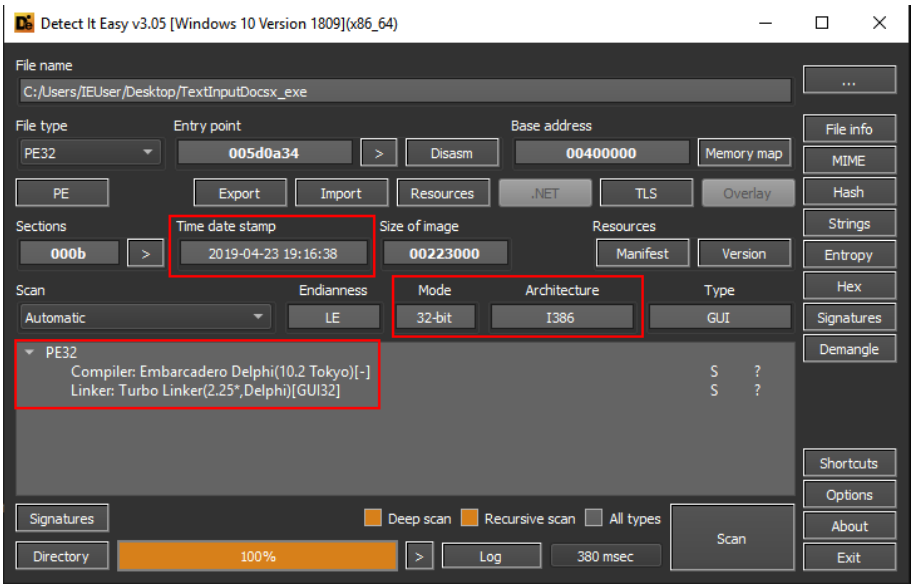
Além de assinaturas, a entropia, que é a medida de desorganização de um determinado sistema, também pode ser útil em cenários de análise de malware. Ela pode servir para identificar códigos criptografados ou comprimidos, visto que os bytes que compõem as instruções de um código estão dispostos de uma forma mais organizada, e com diversos padrões repetidos, do que um código ofuscado de alguma forma. Isso permite inferir que determinado binário utiliza um packer, mesmo quando esse não tem assinatura catalogada. Não há um valor de entropia que identifique que um binário utiliza um packer com certeza, logo, o analista deverá levar em consideração outros fatores que corroborem para essa hipótese, como o baixo nível de strings legíveis por humanos e poucas funções utilizadas.

3.5.3. Prática de identificação e parsing de arquivo PE

Para a tarefa de identificação e *parsing* de um artefato, é possível utilizar o detector de tipos e parser PE *Detect It Easy (DIE)*. Na ferramenta, é possível identificar o tipo de arquivo, navegar por sua estrutura PE (caso seja um binário PE), buscar informações de seções de código e de dados, strings, entropia e diversos outros conceitos que serão abordados nas próximas seções.

Na identificação do artefato 'TextInputDocsx.exe' a partir da ferramenta DIE, é possível verificar o modo de operação (32-bit ou 64-bit), a arquitetura, identificação do compilador (Delphi, nesse caso) e a data e o horário em que possivelmente o binário foi compilado, conforme Figura 3.13.

Figura 3.13. Informações gerais do binário ao ser aberto no identificador DIE

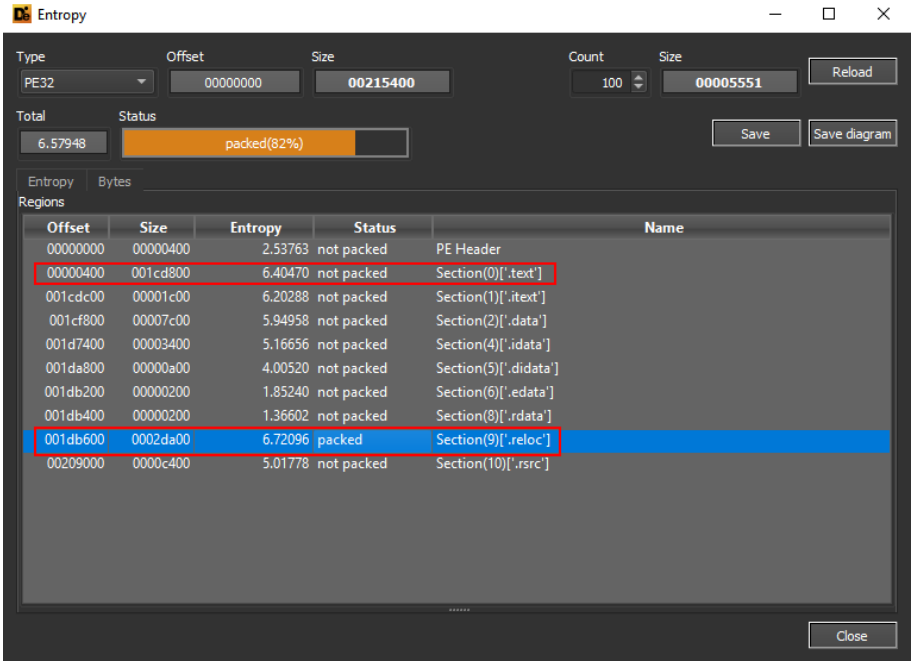


Foi citado data e horário que possivelmente o binário foi compilado pois na compilação de um código-fonte, o compilador irá produzir um binário correspondente e escreverá um tempo de compilação nos metadados do arquivo. Porém esse valor pode ter sido alterado por um ator malicioso para falsificar a data original. Além disso, certos

compiladores utilizam datas padrão para todos os binários, que não reflete o período real de compilação.

Ao verificar a entropia, é possível identificar que o binário contém seções com baixa entropia e uma, a *.reloc*, que está acima da média para uma seção comum. Devido a isso, a ferramenta informa que o binário, como um todo, está possivelmente com *packer*. Entretanto, como apenas uma seção está com uma alta entropia, e ela não é a de código (*.text*) ou de dados (*.data*), podemos inferir que o binário não está usando um packer.

Figura 3.14. Entropia das seções do binário



3.5.4. Identificação de strings maliciosas

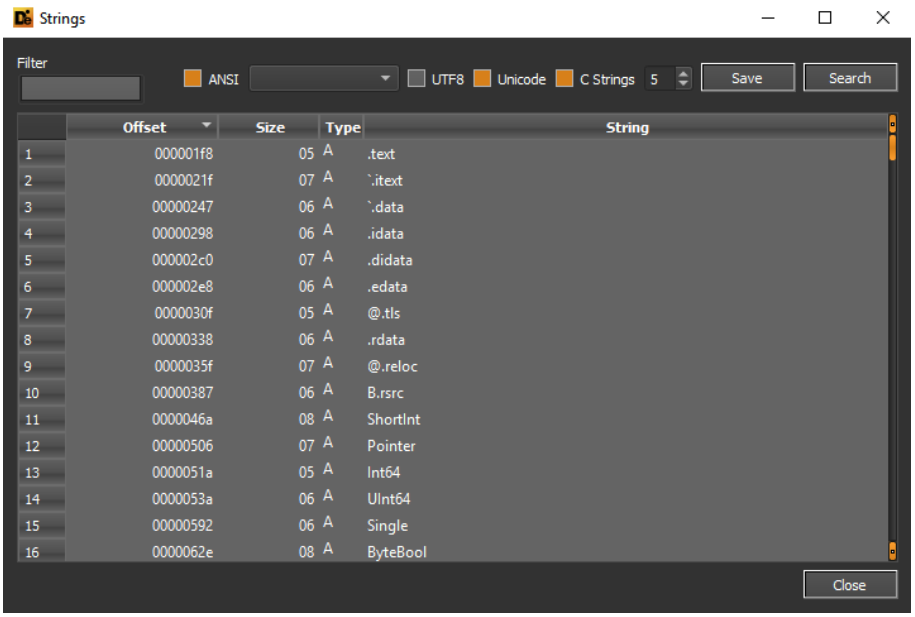
As ferramentas *DIE* e *PEStudio* permitem a busca de cadeias de caracteres incluídas em arquivos binários. Dentro de um binário podem ser encontradas URLs, IPs, domínios, chaves de registro, entre outros possíveis indicadores maliciosos. Também é possível identificar certos tipos de encoding e criptografia através de certas cadeias de caracteres padrão, como no caso da codificação *base64*, que pode ser identificada por uma cadeia de caracteres que contém os intervalos de letras, símbolos e números A–Z, a–z, 0–9, +, / e =.

Além disso, é importante notar a distinção entre cadeias de caracteres *ASCII* e *UTF-16*. Strings ASCII, mais comuns em programas Linux, contém apenas um byte por caractere e terminam com o valor \0, também chamado de *NULL*. Strings *UTF-16*, por sua vez, são prevalentes na programação para a plataforma Windows, contém dois bytes por caractere, sendo o segundo byte o valor \0, e terminam pelo caractere \0\0. Dependendo da ferramenta utilizada para busca das strings, será necessário indicar a codificação na qual deseja-se buscar as strings.

Na Figura 3.15 está uma imagem da ferramenta *DIE*, na janela de busca de strings do artefato 'TextInputDocsx.exe'. É possível buscar por strings através do botão "Strings",

no qual uma janela irá se abrir, mostrando strings ASCII e UTF-16 por padrão. Nesse artefato em específico, não foi possível encontrar strings que fossem interessantes para *insights* sobre o funcionamento dele.

Figura 3.15. Strings encontradas através do DIE



3.5.5. Imports e funções

Todo binário *PE* tem uma estrutura chamada *import table*, localizada nos *Data Directories* e representada pela seção *.idata*. Nela vão estar informações de quais funções externas contidas em DLLs o loader deverá carregar para o funcionamento correto do executável.

Todo EXE deverá carregar a *kernel32.dll* e algumas outras DLLs, que dependerão das funcionalidades escritas pelo programador. Logo, a partir da import table, é possível inferir quais são as funcionalidades de um artefato. Por exemplo, caso o binário crie subprocessos, na sua import table provavelmente constará a função *ShellExecute* da *Shell32.dll*, ou caso faça conexão com um host externo, o uso da função *gethostbyname* da *Winsock2.dll* poderá indicar a resolução *Domain Name System (DNS)* de um dado nome de host. Esse processo de importar funções externas ao binário e consequentemente armazenar o nome das bibliotecas e funções importadas na import table é chamado de *dynamic linking*.

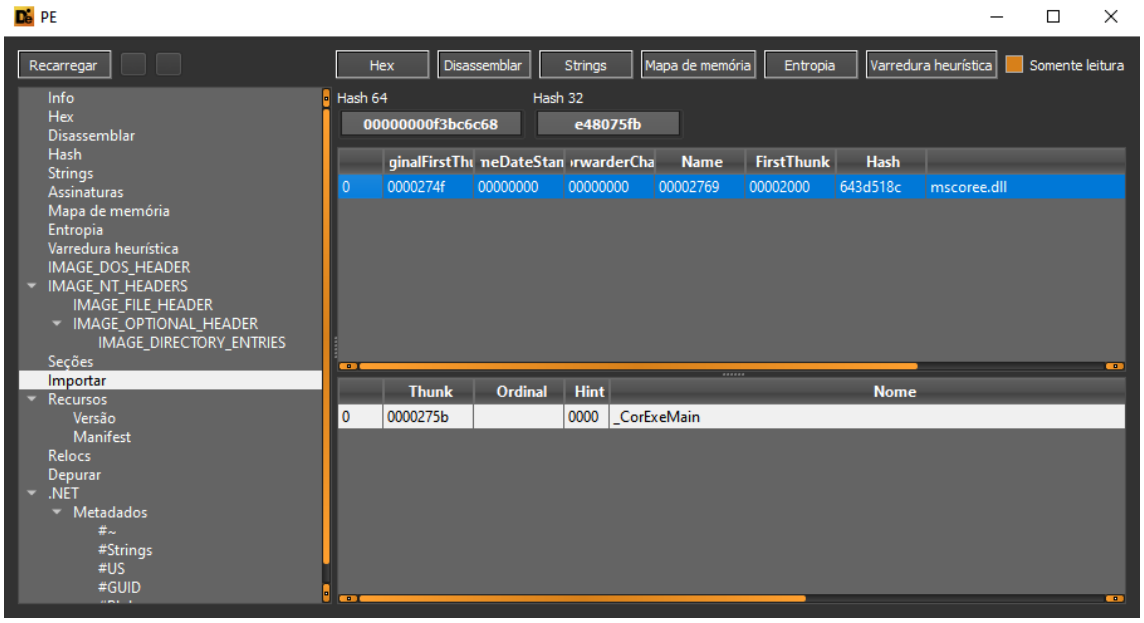
Há algumas abordagens para evitar o uso *dynamic linking*. Uma delas é o *static linking*, onde o código das funções externas utilizadas são copiadas para o programa, durante o processo de geração do binário, não gerando nenhuma dependência externa de bibliotecas e por esse motivo não tendo tais funções descritas na import table. Outra forma é o *runtime linking*, onde serão utilizadas as funções *GetProcAddress* e *LoadLibrary* da *kernel32.dll* para resolução dos endereços das funções durante a execução do programa, evitando guardar seus nomes na importa table.

Em um binário simples, como uma aplicação que apenas escreve na tela *Hello World!*, há uma série de imports implícitos, feitos pelo compilador no processo de ge-

ração do executável. Tais imports são necessários para o carregamento correto de até mesmo binários simples, e uma import table que contenha poucos imports, especialmente as que contém *GetProcAddress* e *LoadLibrary*, usadas no *runtime linking*, indicam que determinado binário provavelmente está ofuscado de alguma forma.

Algumas linguagens de programação implementam parte da funcionalidade de *runtime* dos binários em DLL, e a presença de tais DLLs ou funções pode indicar o uso de determinada linguagem na construção do binário. Por exemplo, é possível identificar o uso da linguagem C# em um binário pela presença da DLL *mscoree.dll*, com a função importada *_CorExeMain*, como exemplificado na Figura diefig:dotnet_die, através de um simples programa que escreve 'Hello World!' na tela.

Figura 3.16. Exemplo da import table de um binário escrito em C#



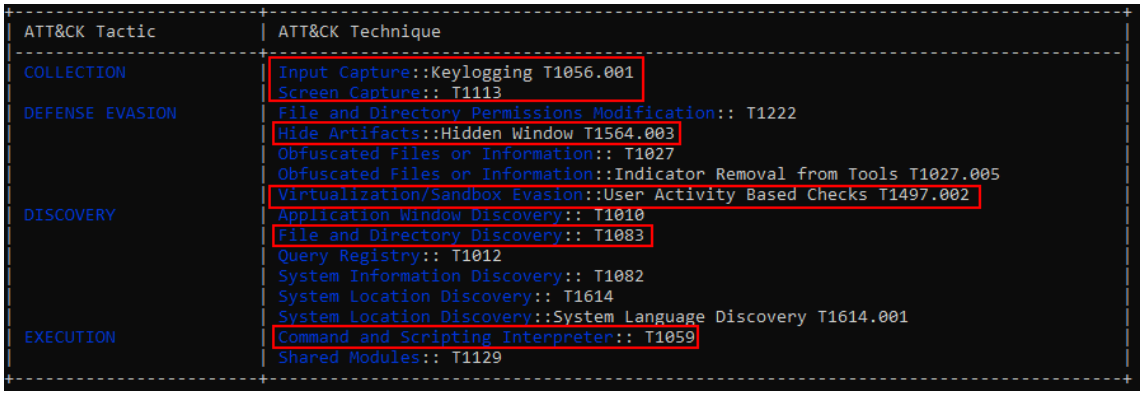
3.5.6. Capacidades do binário com CAPA

A ferramenta CAPA, desenvolvida e distribuída pela Fireeye, permite a análise de capacidades suspeitas ou maliciosas de EXEs ou DLLs. Ela funciona a partir de uma base de arquivos de assinatura, escritos no formato *YAML Ain't Markup Language (YAML)*, que buscam por padrões já identificados e catalogados de comportamentos suspeitos ou maliciosos. Com ela, é possível identificar se um binário realiza *dynamic linking*, tem funções de keylogging ou conecta-se com um servidor de *command and control (C2)*, entre outras capacidades.

Ao executar *C:\Tools\Reverse\capa\capa.exe TextInputDocsx.exe*, há o processamento do binário e suas funções pelo capa e são retornadas três tabelas, que constroem diferentes visões das capacidades do artefato. A primeira tabela, conforme Figura 3.17, mostra uma visão do artefato dividida em táticas e técnicas do framework *MITRE ATT&CK*. É possível notar que a ferramenta identificou capacidades de coleta de entradas do usuário, da tela, além de capacidades relacionadas a evasão de virtualização/sandbox e descoberta

de arquivos da máquina.

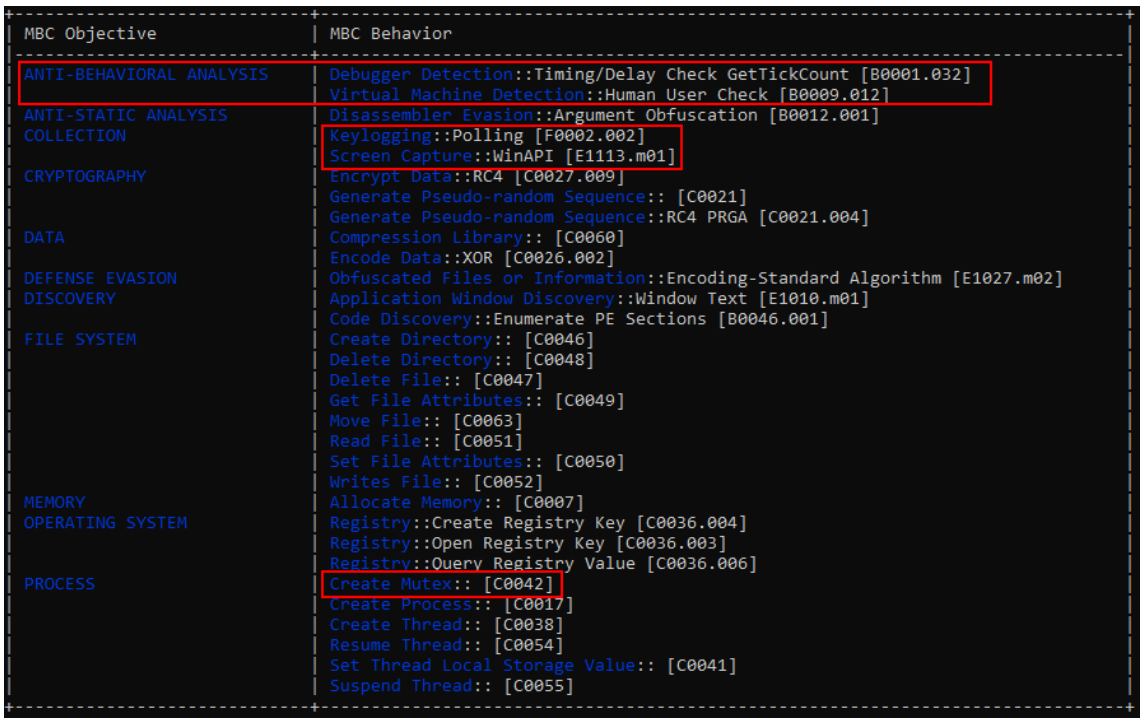
Figura 3.17. Correlações *MITRE ATT&CK* retornadas pela ferramenta *capa*



ATT&CK Tactic	ATT&CK Technique
COLLECTION	Input Capture::Keylogging T1056.001
	Screen Capture:: T1113
DEFENSE EVASION	File and Directory Permissions Modification:: T1222
	Hide Artifacts::Hidden Window T1564.003
	Obfuscated Files or Information:: T1027
	Obfuscated Files or Information::Indicator Removal from Tools T1027.005
	Virtualization/Sandbox Evasion::User Activity Based Checks T1497.002
DISCOVERY	Application Window Discovery:: T1010
	File and Directory Discovery:: T1083
	Query Registry:: T1012
	System Information Discovery:: T1082
	System Location Discovery:: T1614
	System Location Discovery::System Language Discovery T1614.001
EXECUTION	Command and Scripting Interpreter:: T1059
	Shared Modules:: T1129

A segunda tabela traz informações sobre *Malware Behavior Catalog (MBC)*, catalógo de comportamentos de malwares baseado no *MITRE ATT&CK*. No caso do artefato, é possível verificar a utilização de técnicas para detecção de debuggers via API *GetTickCount*, de virtualização, criação de *mutexes*, e corroborar com a capacidade de *keylogging/screenlogging*.

Figura 3.18. Correlações *MBC* retornadas pela ferramenta *capa*



MBC Objective	MBC Behavior
ANTI-BEHAVIORAL ANALYSIS	Debugger Detection::Timing/Delay Check GetTickCount [B0001.032]
	Virtual Machine Detection::Human User Check [B0009.012]
ANTI-STATIC ANALYSIS	Disassembler Evasion::Argument Obfuscation [B0012.001]
COLLECTION	Keylogging::Polling [F0002.002]
	Screen Capture::WinAPI [E1113.m01]
CRYPTOGRAPHY	Encrypt Data::RC4 [C0027.009]
	Generate Pseudo-random Sequence:: [C0021]
	Generate Pseudo-random Sequence::RC4 PRGA [C0021.004]
DATA	Compression Library:: [C0060]
	Encode Data::XOR [C0026.002]
DEFENSE EVASION	Obfuscated Files or Information::Encoding-Standard Algorithm [E1027.m02]
DISCOVERY	Application Window Discovery::Window Text [E1010.m01]
	Code Discovery::Enumerate PE Sections [B0046.001]
FILE SYSTEM	Create Directory:: [C0046]
	Delete Directory:: [C0048]
	Delete File:: [C0047]
	Get File Attributes:: [C0049]
	Move File:: [C0063]
	Read File:: [C0051]
	Set File Attributes:: [C0050]
	Writes File:: [C0052]
MEMORY	Allocate Memory:: [C0007]
OPERATING SYSTEM	Registry::Create Registry Key [C0036.004]
	Registry::Open Registry Key [C0036.003]
	Registry::Query Registry Value [C0036.006]
PROCESS	Create Mutex:: [C0042]
	Create Process:: [C0017]
	Create Thread:: [C0038]
	Resume Thread:: [C0054]
	Set Thread Local Storage Value:: [C0041]
	Suspend Thread:: [C0055]

A terceira e última tabela cita diretamente as capacidades encontradas de acordo com o *namespace* (categoria para um arquivo de regras YAML) que foi utilizado para

detectá-las. Como mostrado na Figura 3.19, é possível identificar capacidades conhecidas de bankers/infostealers.

Figura 3.19. Tabela de capacidades suspeitas e seu namespace de detecção

CAPABILITY	NAMESPACE
inspect load icon resource	anti-analysis
check for time delay via GetTickCount (3 matches)	anti-analysis/anti-debugging/debugger-detection
check for unmoving mouse cursor	anti-analysis/anti-vm/vm-detection
contain obfuscated stackstrings (2 matches)	anti-analysis/obfuscation/string/stackstring
get geographical location (4 matches)	collection
log keystrokes	collection/keylog
log keystrokes via polling (11 matches)	collection/keylog
capture screenshot	collection/screenshot
encode data using XOR (13 matches)	data-manipulation/encoding/xor
encrypt data using RC4 PRGA (3 matches)	data-manipulation/encryption/rc4
generate random numbers using the Delphi LCG	data-manipulation/prng/lcg

Utilizando a opção '-v' do CAPA, é possível identificar o endereço das funções suspeitas encontradas, conforme Figura 3.20:

Figura 3.20. Endereçamento de funções suspeitas identificadas pelo capa

```
log keystrokes
namespace  collection/keylog
scope      function
matches    0x55AF30

log keystrokes via polling (11 matches)
namespace  collection/keylog
scope      function
matches    0x4E6700
           0x4EA890
           0x4EEE40
           0x515DF8
           0x55AF30
           0x560898
           0x5608D4
           0x56BAA4
           0x56BAF8
           0x572AD4
           0x57E998

capture screenshot
namespace  collection/screenshot
scope      function
matches    0x4C4F14
```

Munido dessas informações, o analista poderá guiar melhor seus próximos passos nas análises posteriores. É possível até mesmo utilizar esse conhecimento efetuar operações, como alteração das chamadas de funções para evasão de virtualização e debugging, que podem atrapalhar as análises comportamentais e reversa.

3.6. Análise de Comportamento

Nesta seção, serão abordados os conceitos e as ferramentas necessárias para a análise manual de comportamento de artefatos suspeitos. Para tanto, serão utilizadas ferramentas de monitoramento de processos, de rede e de ações no sistema de arquivos e no registro do Windows.

O objetivo desta etapa da metodologia é examinar as ações realizadas pelo artefato suspeito quando executado. Diferentemente da análise estática, quando são observadas,

por exemplo, *strings* suspeitas apontando para um *hostname* desconhecido que podem indicar uma conexão maliciosa, na análise dinâmica ou de comportamento, poderemos observar de fato as funcionalidades do artefato analisado. No caso do exemplo da *string* suspeita, pode ser possível observar a tentativa de resolução do endereço do host suspeito, eventuais tentativas de conexão ao IP, qual porta foi utilizada, e assim por diante.

Como se pode observar, o objetivo é semelhante ao da análise automatizada com o uso de *Sandboxes*. As *Sandboxes* podem realmente ajudar na descoberta de comportametnos do artefato malicioso, no entanto, podem apresentar algumas limitações que poderão ser superadas na análise manual. Considere o seguinte exemplo: durante a análise de um artefato suspeito numa *Sandbox*, você identificou que o programa tenta realizar uma conexão na porta 80 de um determinado endereço IP. Como a funcionalidade da *Sandbox* é mais limitada e a tentativa de acesso falharia, você não conseguiria descobrir dados importantes que seriam transacionados na conexão.

Por outro lado, na análise manual de comportamento, dada a maior flexibilidade, esta limitação poderia ser superada. Para tanto, para o exemplo em questão, ao identificar que o artefato analisado estaria tentando uma conexão a um host da internet, você poderia simular que aquele endereço estaria disponível, permitindo que o malware estabelecesse uma conexão com um falso serviço enquanto você monitora os pacotes enviados.

Observe ainda neste exemplo que o acesso ao host desejado pelo malware foi simulado, ou seja, o malware não acessaria o host na internet, mas uma outra VM (no caso poderia ser a VM Linux), que seria preparada para aguardar a conexão do malware enquanto registraria todo o fluxo de dados.

Uma pergunta que você pode estar se fazendo é: por que não deixar o malware se comunicar com o host desejado na internet enquanto os pacotes são coletados? A resposta é que muito provavelmente você perderia o controle do processo de análise além de sinalizar ao atacante que o encontrou ou expondo a sua análise de alguma forma. Imagine se, neste caso, a conexão fosse para baixar e executar um novo conteúdo malicioso, que resultaria em novas conexões a outros endereços e novas ações no sistema operacional. Muito facilmente você poderia perder a trilha da análise em meio a uma grande quantidade de dados e ações.

Desta forma, o ideal é conduzir a análise de comportamento passo a passo, num ciclo contínuo de descoberta de recursos demandados pelo malware e fornecimento monitorado do recurso.

Nas subseçoes a seguir, apresentaremos a análise de comportamento do PE 'TextInputDocxs.exe'. Para isso, utilizaremos uma série de ferramentas que monitorarão o processo na busca por indicadores maliciosos tais como de alterações em arquivos, Registros do Windows e comunicação com endereços suspeitos (tráfego de comando e controle ou C2).

3.6.1. Ferramentas para análise de comportamento

A seguir a lista das ferramentas que serão utilizadas nesta análise:

- **Process Monitor**

O Proccess Monitor, também conhecido por 'procmon', é uma ferramenta avançada de monitoramento para o Windows que mostra em tempo real as atividades em arquivos, Registro do Windows e processos. Ela combina as funcionalidades de dois utilitários legados do Sysinternals, Filemon e Regmon, e adiciona uma extensiva lista de melhorias, incluindo filtragem de eventos, lista completa de propriedades de eventos tais como IDs de sessões e nomes de usuários, dados de processos, dados completos de threads com integração de suporte a símbolos para cada operação, registro de log simultâneo para um arquivo e outras mais. Por estes motivos, torna-se uma ferramenta fundamental na atividade de análise de comportamento de malwares.

O Proccess Monitor monitora todas as chamadas de sistema. Como há muitas chamadas de sistema no Windows (na casa de dezenas de milhares por minuto), executá-lo por muito tempo pode levar ao esgotamento dos recursos de RAM da máquina de análise. Desta forma, o ideal é executar a captura de eventos na ferramenta por um período curto de tempo - somente enquanto monitora as atividades do malware.

- **Process Hacker**

  Process Hacker é uma ferramenta gratuita e poderosa para monitoramento de processos e seus recursos, podendo ser bastante útil na análise de códigos maliciosos. A sua funcionalidade é semelhante ao gerenciador de tarefas do próprio Windows, com incrementos importantes. Como exemplo, podemos citar a possibilidade de descoberta de *handles* associados a um processo. *Handles* são apontadores para itens que foram abertos ou criados no sistema operacional, tais como janelas, processos, DLLs e arquivos. Esta funcionalidade pode ser útil, por exemplo, na análise de arquivos utilizados por um código malicioso. Uma outra funcionalidade que pode ser útil para a análise de comportamentos maliciosos é o monitoramento de rede. Com o Process Hacker é possível acompanhar conexões ativas por processo em tempo real.

- **Wireshark**

  Wireshark é uma das ferramentas mais utilizadas para análise de protocolos e tráfego de rede. A ferramenta, que é grátis e multiplataforma, permite a captura e a análise de centenas de protocolos. Do ponto de vista de análise de comportamento de um artefato suspeito, podemos dizer que é uma ferramenta fundamental uma vez que pode nos ajudar a entender como o malware está realizando comunicações de rede. Para o monitoramento do tráfego, utilizaremos o Wireshark na VM Linux capturando o tráfego gerado pelo artefato analisado na VM Windows.

- **FakeDNS**

  FakeDNS é uma ferramenta que simula um servidor DNS e responde um determinado endereço IP para qualquer consulta. Desenvolvida inicialmente por Francisco Santos [18], está disponível em uma versão modificada na distribuição REMnux. Do ponto de vista da análise de comportamento, esta ferramenta nos ajuda tanto na

[18] https://code.activestate.com/recipes/491264-mini-fake-dns-server

identificação de consultas DNS realizadas por um artefato suspeito quanto no direcionamento de eventuais conexões realizadas para um endereço IP de interesse do analista, onde a captura do tráfego estará sendo realizada. No nosso ambiente de análise, apontaremos o IP da VM Linux para qualquer consulta DNS realizada no na VM Windwows.

- **Noriben**

  Noriben [19] é um script em Python que funciona em conjunto com o Proccess Monitor para, automaticamente, coletar, analisar, e reportar indicadores de códigos maliciosos em tempo de execução. Em resumo, o Noriben faz com que o ambiente de execução do artefato suspeito, no nosso caso a VM Windodws, funcione de forma similar a uma Sandbox. Seu funcionamento, basicamente, consiste na sequinte sequencia de passos: executar o Noriben, executar o código malicioso e aguardar alguns instantes e depois interromper o Noriben. Ao final deste processo, estarão disponíveis um arquivo de texto com o report dos eventos identificados no período do monitoramento e um arquivo com extensão 'pml' da coleta realizada pelo Proccess Monitor.

- **ProcDOT**

  O ProcDOT [20] é uma ferramenta que contribui para a análise dos resultados obtidos pelo Proccess Monitor e pelo Wireshark (opcionalmente). Desenvolvida por Christian Wojner, do CERT da Austria, a ferramenta facilita a interpretação dos eventos coletados ao apresentá-los na forma de um grafo interativo. Desta forma, fica mais fácil de entender a relação entre o processo analisado e as ações realizadas ao longo do tempo, tais como a leitura de um registro do Windows e a gravação de um arquivo.

3.6.2. Análise de comportamento do binário TextInputDocxs.exe

Nesta seção apresentaremos a análise do binário TextInputDocxs.exe utilizando as ferramentas descritas no tópico anterior. Estamos utilizando este binário apenas como um exemplo. As mesmas técncias e ferramentas podem ser utilizadas na análise de outros artefatos.

3.6.2.1. Preparando o ambiente

Antes de dispararmos a execução do binário na VM Windows, precisaremos preparar o ambiente de monitoramento.

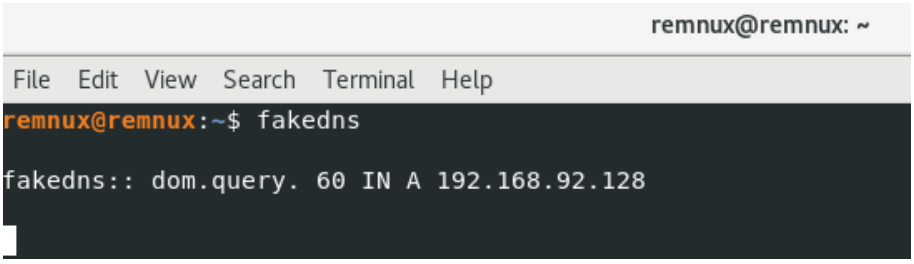
Para facilitar a condução da preparação e do processo de monitoramento, tome nota das interfaces de rede e dos endereços IP das interfaces de rede que conectam as VMs Windows e Linux. Estas interfaces estarão no modo 'Host Only'.

Preparando a VM Linux:

[19] https://github.com/Rurik/Noriben
[20] https://procdot.com/index.htm

- FakeDNS: no REMnux, abra um terminal de comando (menu Activities e depois Terminal) e digite o comando 'fakedns'. Isso fará com que um serviço falso de DNS passe a funcionar na VM Linux, conforme Figura 3.21.

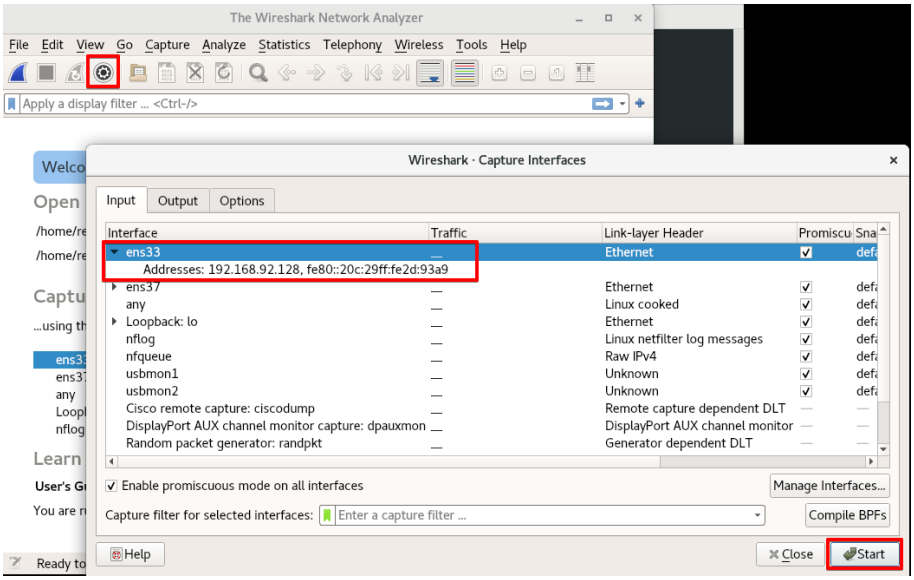
Figura 3.21. Fakedns em execução



- Wireshark: na sequencia, execute o Wireshark. Para isso, volte a clicar no botão 'Activities' e, na barra de busca qua aparecerá, digite 'Wireshark'. Clique no aplicativo 'Wireshark' resultante da busca. No Wireshark, clique no botão de monitoramento, selecione a interface que conecta a VM Linux a VM Windows e depois clique no botão 'Start', conforme Figura 3.22. Observe que, não necessariamente, o endereço IP da imagem corresponderá ao endereço IP utilizado na sua VM Linux. O importnate é que o monitoramento seja feito na interface que conecta as duas VMs entre si.

  Por enquanto, deixe o Wireshark monitorando os pacotes. No entanto, antes de executarmos o malware, é recomendável que a captura seja reiniciada para termos um menor número de pacotes a analisar. O botão para reiniciar a captura está ao lado esquerdo do botão de início da captura na Figura 3.22.

Figura 3.22. Executando o Wireshark



Preparando a VM Windows:

- Certifique-se de que a VM não esteja com acesso à Internet e que as configurações de Gateway e DNS da VM estejam apontando para o endereço IP da VM Linux. Para mais detalhes de como preparar o ambiente de laboratório, leia a seção 3.2.2.

- Execute o Process Hacker e o deixe em execução. Esta ferramenta será utilizada para monitorar o funcionamento do binário e para interrompê-lo quando for o momento certo;

- Binário suspeito: deixe o binário 'TextInputDocsx.exe' [21] em ponto de execução na VM Windows. O binário está no formato ZIP protegido por senha. A senha é 'infected'. É muito importante que só descompacte o arquivo dentro da VM Windows. Isso evitará que o antivirus do seu host possa removê-lo ou que você execute por acidente o binário no seu host;

- Noriben: localize um diretório de nome 'Noriben' na área de trabalho e abra-o. Neste diretório tem o script Noriben e o diretório de saída (output) dos resultados. Por enquanto, não execute ainda o script.

- Snapshot: com o ambiente preparado, crie um *snapshot* da VM Windows. Isso facilitará o retorno da VM exatamente ao estado em que está caso sejam necessárias outras execuções do malware - o que é muito comum de acontecer na análise de artefatos maliciosos.

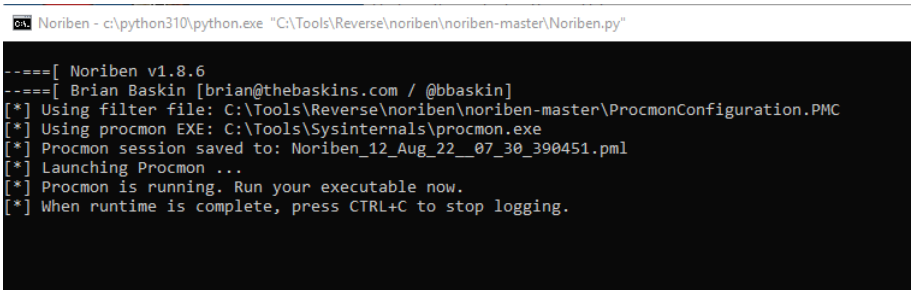
3.6.2.2. Executando o código suspeito

Com o ambiente preparado, é hora de executar o código suspeito. O objetivo aqui é o monitoramento das ações do malware enquanto é executado. Para isso, siga os passos a seguir:

1. Volte a VM Linux e reinicie a captura do Wireshark;

2. Na VM Windows, execute o Noriben. Uma janela de Prompt de Comando será aberta neste momento, conforme 3.23. Conforme informado na tela, para finalizar o processo de monitoramento, deve-se digitar CTRL+C na tela do Noriben.

Figura 3.23. Noriben em execução



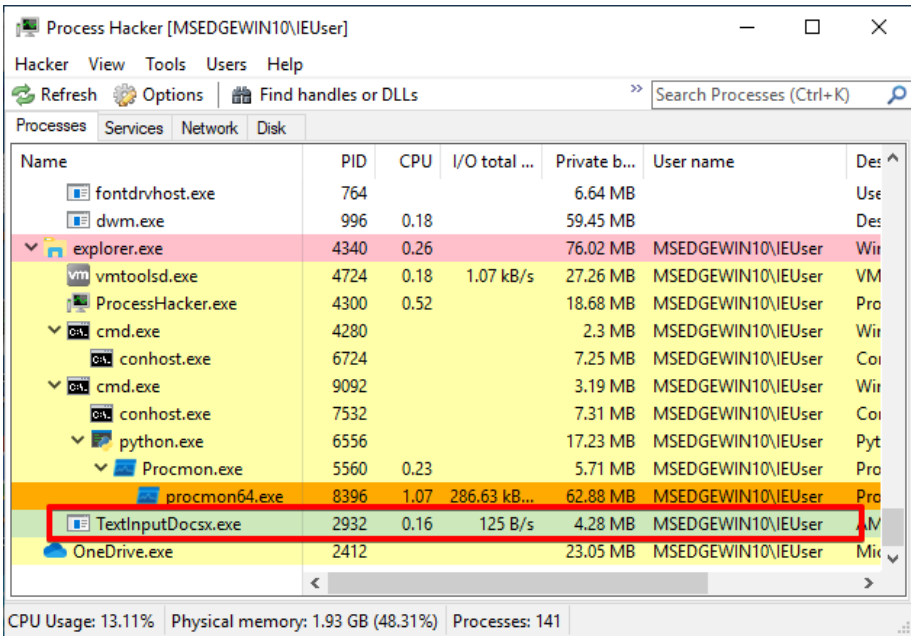
[21] https://github.com/rrmarinho/sbseg2022

3. Execute o binário suspeito: conforme orientações da seção anterior, o binário suspeito estará na VM Windows pronto para ser executado. Execute-o neste momento. A execução do binário suspeito não resultará em nenhuma nova janela. Será silenciosa.

4. Process Hacker: No Proccess Hacker, que já estará em funcionamento, localize o processo do binário suspeito, conforme Figura 3.24. Aguarde cerca de 30 a 40 segundos e interrompa o processo. Para isso, clique no processo com o botão direito e depois em 'Terminate'.

   O tempo a ser aguardado para interromper o processo não segue um padrão. Como esta é uma primeira execução, escolhemos interromper o processo com pouto tempo de execução para ver se algo já ocorreu. No entanto, pode acontecer de o malware ter sido programado para iniciar alguma ação somente após 1 min após ser executado ou até mais do que isso. O fato de não sabermos o tempo ideal a se aguardar no monitoramento é uma limitação da análise de comportamento que pode ser superada com a análises mais avançadas, com o uso de debuggers e disassemblers.

Figura 3.24. Processo TextInputDocsx.exe no Process Hacker

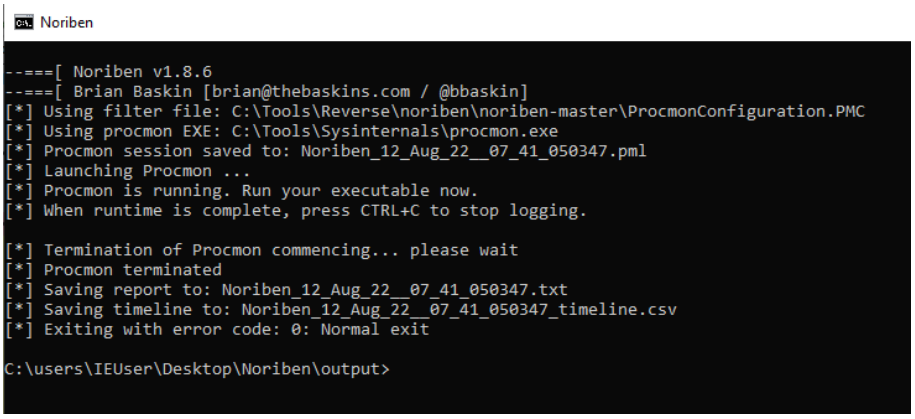


5. Encerrando o monitoramento no Noriben: volte a tela do Noriben, clique na tela preta do terminal para dar o foco e depois pressione CTRL+C. Com isso, o monitoramento será encerrado, conforme Figura 3.25.

   Ao mesmo tempo, uma janela de Bloco de Notas com o relatório de execução será aberta. Por enquanto, deixe o relatório aberto somente. Vamos tratar da análise no próximo tópico.

6. Wireshark: volte a VM Linux e interrompa o monitoramento do Wireshark usando o botão '*Stop Capture*' - o segundo da barra de ícones;

Figura 3.25. Encerrando o monitoramento do Noriben



7. FakeDNS: ainda na VM Linux, volte a janela do FakeDNS e interrompa o processo pressionando CTRL+C.

Uma vez encerrada a etapa de monitoramento, iniciaremos as análises.

3.6.2.3. Analisando os resultados

Até aqui, preparamos o ambiente, executamos o artefato suspeito enquanto o monitorávamos e agora chegou a etapa de análise dos resultados. Confira a seguir os resultados em cada uma das ferramentas:

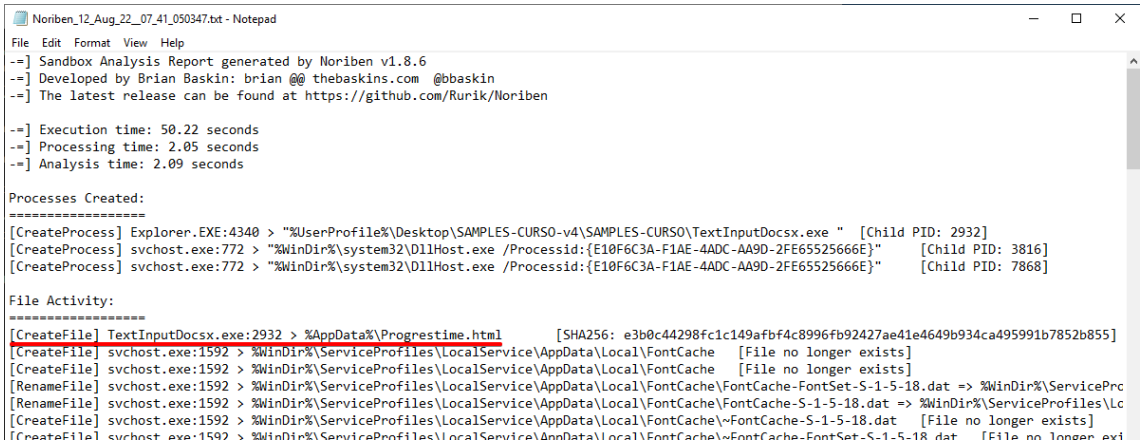
1. **Noriben**

   O relatório do Noriben é dividido em 5 seções principais: *Process Created*, *File Activity*, *Registry Activity*, *Network Activity* e *Unique Hosts*. Em cada uma destas seções, estarão presentes todos os eventos capturados pelo Proccess Monitor enquanto esteve em execução. Desta forma, é importante que tenha em mente que os eventos apresentados no relatório incluem os eventos causados pelo binário executado, mas não somente. Vão ser apresentados eventos diversos que ocorreram no sistema operacional no período da coleta. É exatamente por isso que o quanto mais restrito for o período da coleta, menor será a quantidade de eventos a serem analisados.

   Após uma análise do relatório do Noriben, observamos que o processo 'TextInputDocsx.exe', cujo PID (Process ID) era 2932, criou o arquivo '%AppData% \Processtime.html', conforme Figura 3.26.

   Um próximo passo da análise seria a abertura do arquivo para verificar seu conteúdo, por exemplo. Neste caso, o arquvio está vazio e, por isso, não acrescenta muito. No entanto, de qualquer forma, a criação do arquivo em si pode ser considerada um indicador de comprometimento (IOC) deste artefato. Isso quer dizer que, se você tiver que identificar computadores onde este mesmo artefato tenha sido executado, a identificação da presença deste arquivo poderia ser um excelente indício.

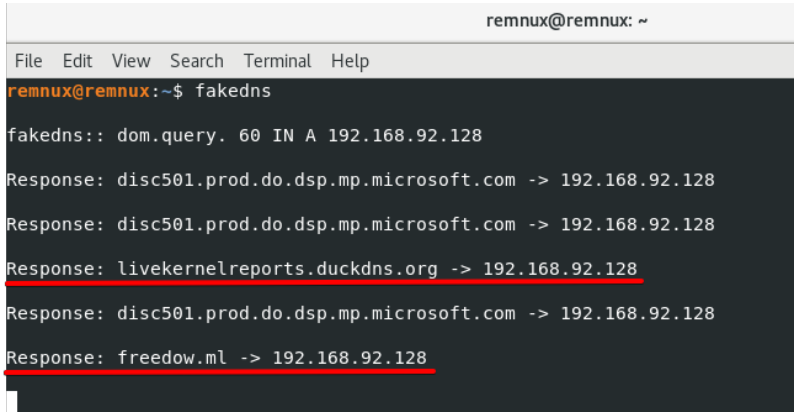
Figura 3.26. Relatório do Noriben



2. **FakeDNS**

   Observando as resoluções de DNS realizadas pelo FakeDNS, identificamos duas que potencialmente estão ligadas à execução do código suspeito, conforme 3.27. Análises posteriores nos ajudarão a identificar se foram ou não originadas pelo binário. Por enquanto há um bom indício pois são as únicas que diferem das outras que foram consultas DNS a domínios Microsoft - provavelmente realizadas por ferramentas do SO.

Figura 3.27. Análise do FakeDNS



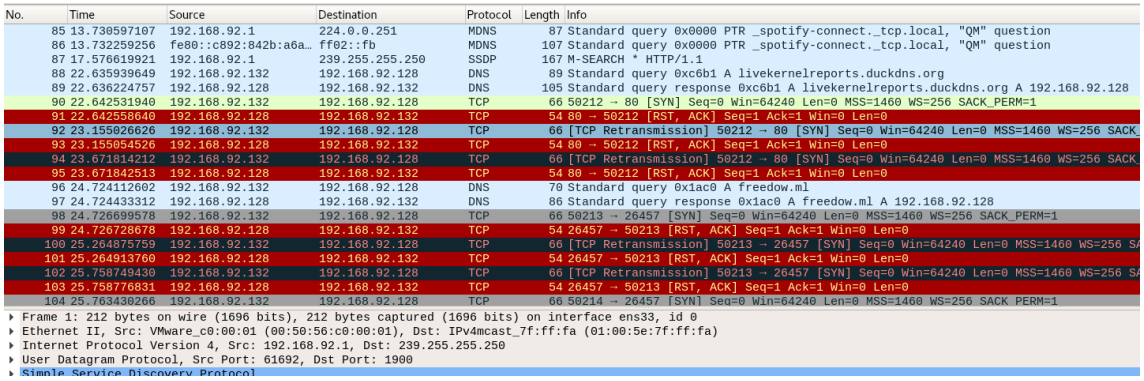
3. **Wireshark**

   De volta ao Wireshark, identificamos uma sequencia de conexões bastante interessante, conforme pode ser visto na Figura 3.28. Acompanhe os itens a seguir para entender os detalhes nas referidas linhas indicados na figura.

   - Linhas 88 e 96: requisições DNS para resoluções dos nomes livekernelreports.duckdns.org e freedow.ml - os mesmos indicados no FakeDNS;
   - Linhas 89 e 97: respostas do FakeDNS devolvendo o IP 192.168.92.128 à consulta DNS, o que é esperado;

- Linhas 90 e 98: pacotes de início de conexão (SYN) para as portas TCP/80 e TCP/26457 do IP 192.168.92.128. Esta conexão, originalmente, seria destinada aos endereços IP resultantes das consultas DNS acima. Isso indica que o malware poderia estar tentando conexão com o atacante. Observe que as conexões não tiveram sucesso uma vez que estas portas não estão abertas no host 192.168.92.128 - que, neste caso é a VM Linux.

Figura 3.28. Análise do Wireshark



4. **ProcDOT**

   De volta à VM Windows, execute o software ProcDOT. O ícone deve estar na área de trabalho.

   Na primeira vez que é executado, o ProcDOT precisa ser configurado. Na janela de opções, informe os caminhos das ferramentas solicitas:

   - Path to windump/tcpdump: C:\textbackslash Tools \Reverse\windump \WinDump.exe
   - Path to dot (Graphviz): C:\Program Files\Graphviz\bin\dot.exe

   Uma vez configurado, no campo 'Procmon' da tela, informe o path do arquivo no formato 'csv' gerado pelo Noriben e no campo 'Launcher', busque pelo processo 'TextInputDocsx.exe'. Siga os passos de 1 a 4 na Figura 3.29.

   Uma vez selecionados os parâmetros (não esqueça de clicar em Refresh), um grafo será apresentado pelo ProcDOT representando os eventos relacionados ao processo selecionado, conforme Figura 3.30.

   Observe que este resultado resume bem o comportamento do artefato até aqui: a criação do arquivo Progresstime.html e as tentativas de conexão nas portas TCP/80 e TCP/26457 ao endereço IP da VM Linux. Aqui comprovamos a suspeita que foi levantada lá na análise do FakeDNS e do Wireshark - as conexões às portas suspeitas foram realmente feitas pelo binário analisado. Uma outra forma de comprovar isso seria através do Process Hacker.

Figura 3.29. Seleção de parâmetros no ProcDOT

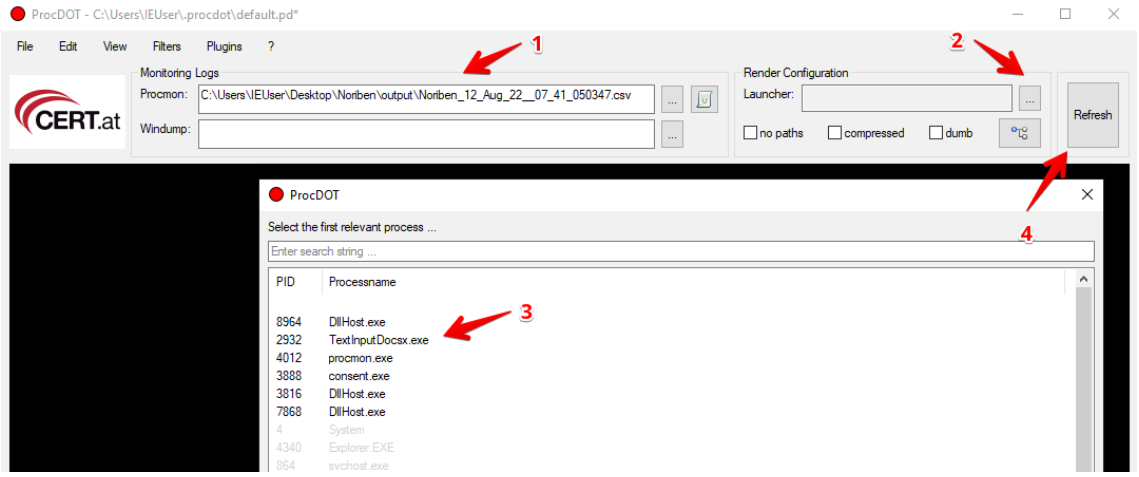
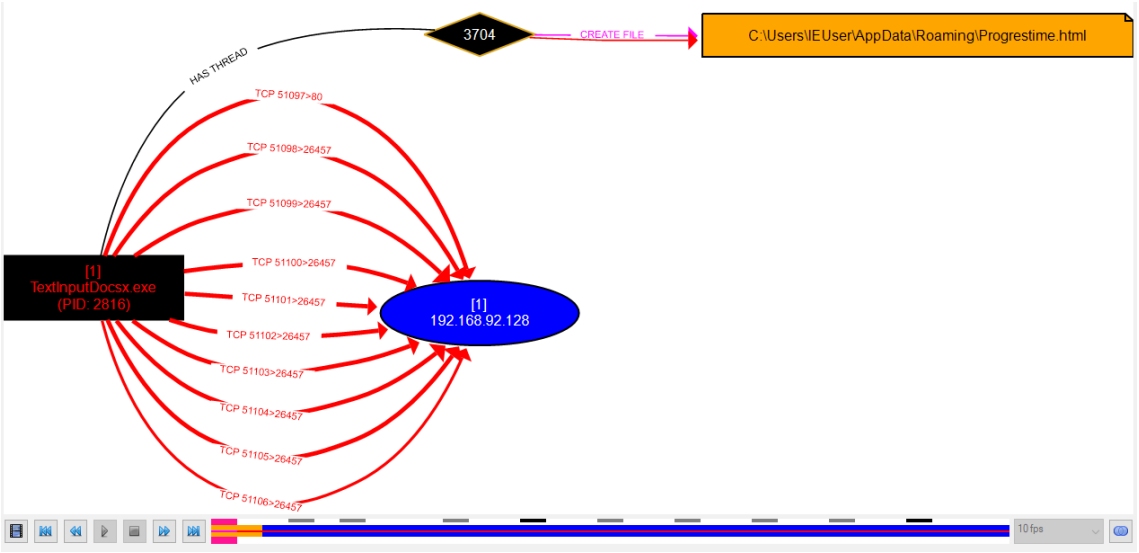


Figura 3.30. Resultado ProcDOT



3.6.2.4. Próximos passos

Até aqui, observamos alguns indicadores de comprometimento ligados ao binário analisado:

1. Criação do arquivo '%AppData% \Processtime.html'.
2. Consultas DNS aos nomes: livekernelreports.duckdns.org e freedow.ml;
3. Tentativas deconexões TCP nas portas 80 e 23457;

No entanto, podemos tentar avançar mais na análise de comportamento. Conforme

explicado na introdução desta seção, na análise de comportamento podemos fornecer de forma gradativa os recursos demandados pelo artefato.

Um próximo passo natural para esta análise seria tentar descobrir o que o malware estaria buscando nas portas às quais tentou se conectar. Para tanto, abriremos as portas demandadas na VM Linux e aguardaremos a conexão enquanto monitoramos o tráfego via Wireshark.

Para fornecer as portas 80 e 23457, proceda da seguinte forma na VM Linux:

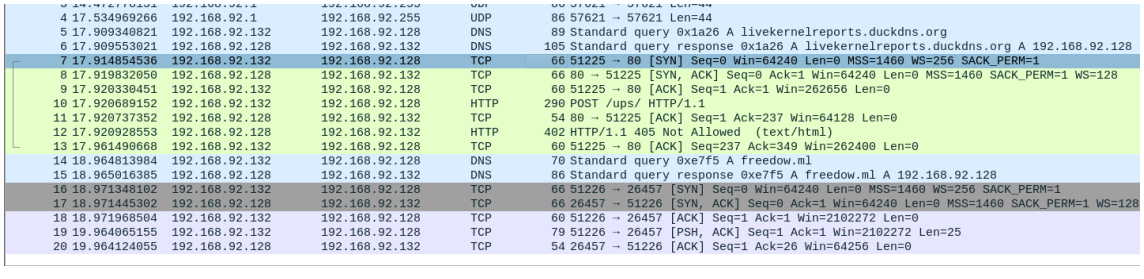
- Porta 80: no prompt de comando da VM Linux, inicie o serviço http. Na tela de comando, digite 'httpd start' e <ENTER>. Com isso, um serviço HTTP server será iniciado na VM Linux;
- Porta 23457: para este serviço, iniciaremos um socket TCP utilizando o Netcat. Para isso, execute no terminal o comando 'nc -l -p 26457'.

Agora, refaça o experimento seguindo os seguintes passos:

1. Retorne o *snapshot* da VM Windows. Isso vai garantir que observaremos a primiera infecção do malware na máquina. Opcionalmente, para este caso em específico, basta apagar o arquivo '%AppData% \Processtime.html' que o malware realizará todo o comportamento;
2. Na VM Linux, certifique-se de deixar o FakeDNS e Wireshark ativos;
3. Inicie o 'TextInputDocsx.exe' na VM Linux enquanto monitora o tráfego no Wireshark.

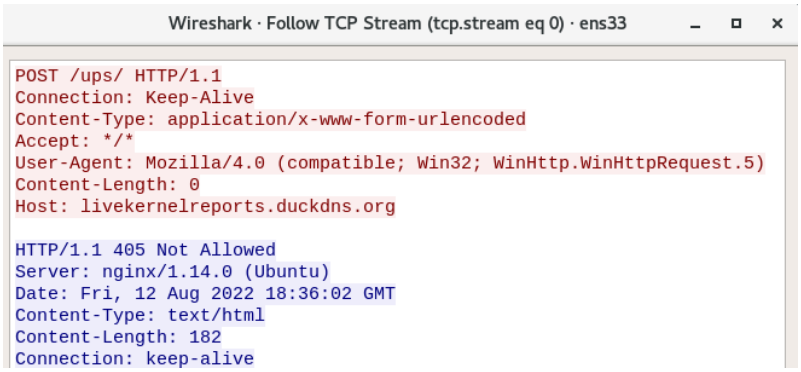
Como resultado, você deve observar algo semelhante ao que temos na Figura 3.31.

Figura 3.31. Resultado Wireshark



Observe agora que as conexões nas portas 80 e 26457 foram estabelecidas com sucesso. Adicionalmente, descobrimos que na porta TCP/80 o malware enviou um HTTP request do tipo POST na URL '/ups/'. Podemos, inclusive descobrir mais indicadores, como, por exemplo o 'user-agent' utilizado pelo malware na requisição HTTP. Para isso, basta inspecionar a conexão na porta 80 no Wireshark com o botão direito, depois Follow e depois TCP Stream. O resultado será algo semelhante ao da Figura 3.32.

Figura 3.32. Resultado Wireshark - HTTP Request



Para a conexão TCP/26457, a mesma análise pode ser feita. Adicionalmente, na própria tela onde executamos o Netcat, veremos o resultado. Observe que o malware realizou uma conexão na porta aberta com o Netcat e enviou uma sequencia de dados 'LOGIN TestUser password'. Muito provavelmente, está enviando informações para o atacante ou tentando autenticar a nova vítima. Quando estiver com tempo, tente interagir com o malware por meio do Netcat e veja se ele responde.

3.7. Análise reversa

A análise reversa, também chamada de engenharia reversa, é a última etapa de análise, sendo a mais custosa em questão de tempo e complexa em nível de conhecimento. Nela, o analista utilizará ferramentas do tipo *debugger* ou *disassembler*, que permitirão realizar procedimentos de alteração de fluxo de código, modificação de regiões da memória RAM e de registradores em um dado processo [Wong 2018],[Dang et al. 2014].

Alguns procedimentos, como descriptografia de conteúdo e identificação e bypass de técnicas anti-sandbox só podem ser realizadas, depedendo do cenário, com uma análise dinâmica profunda. É nela também que será possível entender exatamente o funcionamento de cada thread ou função importante executando no processo.

3.7.1. Arquitetura de computadores

A arquitetura de computadores pode ser pensada como uma organização dos componentes físicos e virtuais que fornece abstrações para o sistema operacional. As arquiteturas de computadores modernos são baseados na arquitetura de Von Neumann, que definiu componentes como a *Control Processing Unit (CPU)*, *Control Unit*, *Instruction Register*, e a memória principal (RAM) para armazenamento de dados e instruções. Seu funcionamento utiliza alguns registradores, que são memórias extremamente voláteis embutidas no processador, para realizar o ciclo de *fetch-decode-execute*, que busca uma instrução na RAM para ser executada, a decodifica e executa.

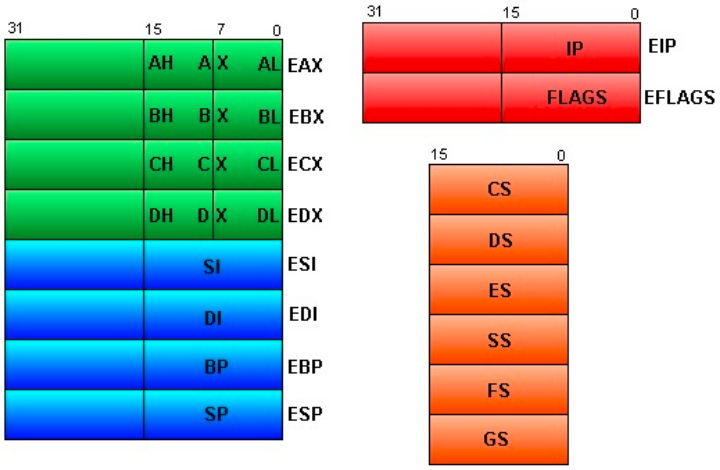
As arquiteturas são categorizadas de acordo com o tamanho da palavra, ou seja, quantos bits o processador consegue processar em um ciclo de execução. Isso também limita a quantidade de memória principal máxima que um processador pode gerenciar, visto que um registrador específico é utilizado nas arquiteturas modernas para apontar o próximo endereço da RAM que deverá ser decodificado e executado.

As arquiteturas modernas geralmente tem o tamanho de uma palavra de 32-bit ou 64-bit e contém uma série de registradores para uso geral, que auxiliam na execução de um programa. Exemplos de arquiteturas são a x86, amd64 e ARM, e cada um tem definido uma *Instruction Set Architecture (ISA)*, que é o conjunto de instruções que são disponibilizadas para o sistema operacional, além de um conjunto de registradores para propósitos gerais, segmentação e controle de fluxo. Entre os registradores mais importantes na arquitetura x86, estão:

- **Propósito geral:** EAX, EBX, ECX, EDX, EDI, ESI, EBP, ESP
- **Controle de fluxo:** EIP, EFLAGS
- **Segmentos**: CS, DS, ES, SS, FS, GS

Na Figura 3.33, é possível ter uma visão desses registradores com relação a quantidade de bits máxima que pode ser armazenada. Note que alguns registradores de 32-bits, como EAX, EBX, ECX e EDX podem ser acessados pelas suas representações de 16-bits (AX, BX, CX, DX), a parte mais significativa dos 16-bits (AH, BH, CH, DH) e a parte menos significativa dos 16-bits (AL, BL, CL, DL). Esse acesso é possível devido a retrocompatibilidade com programas escritos para arquiteturas antigas da Intel, que contavam com palavras de 16-bits e 8-bits.

Figura 3.33. Registradores x86 e a quantidade de bits que armazenam



Os registradores EIP e EFLAGS são registradores especiais, pois o EIP irá indicar o endereço da próxima instrução a ser executada, enquanto que o EFLAGS é uma máscara de bits que cada bit irá representar uma flag de comparação. Essas flags são configuradas

de acordo com o resultado da última instrução executada, como, por exemplo, se a instrução retornou que o valor de dois registradores é igual. Na listagem abaixo, estão algumas flags que podem ser configuradas nesse registrador:

- **ZF**: a Zero Flag é ativada quando o resultado de uma operação é igual a zero. Caso contrário, é desativada;
- **CF**: a Carry Flag é ativada quando o resultado de uma operação é muito grande ou muito pequeno para o operando de destino. Caso contrário, é desativada;
- **SF**: a Sign Flag é ativada quando o resultado de uma operação é negativo. Caso contrário, é desativada;
- **TF**: a Trap Flag é usada para debug. O processador x86 executará uma instrução por vez se esta flag estiver ativada;

O sistema operacional utiliza essas abstrações para criar mais abstrações de código, como *system calls* e *APIs*, e de estruturas de dados para gerenciamento de memória. Entre elas, estão:

- **stack**: área utilizada para variáveis locais e parâmetros para funções;
- **heap**: área utilizada para alocação dinâmica de memória durante a execução do programa;

Durante o carregamento de um código, a stack é uma estrutura de dados que irá crescer dos endereços mais altos para os endereços mais baixos. Isso quer dizer que ao diminuirmos um endereço na stack, estamos alocando regiões novas de memória para ela. A heap, por sua vez, cresce de endereços menores para os endereços maiores, em direção à stack. Além disso, a stack é gerenciada por dois registradores: EBP e ESP, responsáveis por armazenar o endereço da base da pilha e do topo dela, respectivamente, e por operações de empilhamento e desempilhamento.

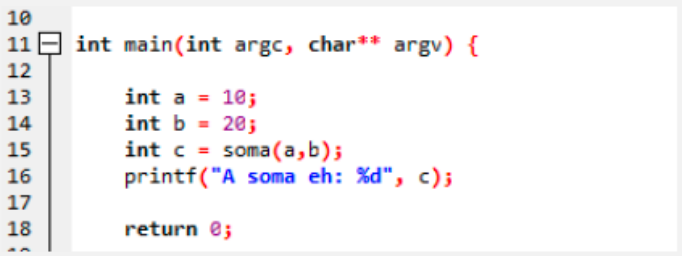
3.7.2. Assembly x86

Nesta seção serão abordados aspectos teóricos que servirão de base para a engenharia reversa de binários PE.

Antes do tratamento do assunto Assembly, é importante relembrar o processo de produção de um executável. Para a construção de um arquivo executável pelo sistema operacional, é geralmente utilizada uma linguagem de alto nível, como C/C++, para a escrita do código-fonte. Esse código-fonte é tratado por um compilador, que, depois de diversas etapas, como análise léxica, sintática e montagem, gerará um código de máquina que poderá ser lido pelo processador. Um exemplo de código-fonte escrito na linguagem C está exemplificado na Figura 3.34.

O código de máquina gerado é composto de zeros e uns, em uma ordem lógica de opcodes e argumentos. Os opcodes são números inteiros que podem ser entendidos

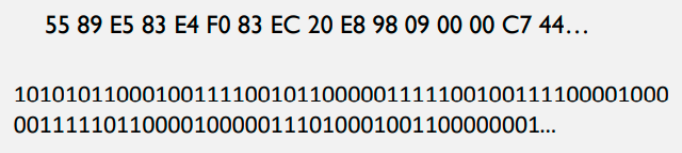
Figura 3.34. Exemplo de código-fonte escrito na linguagem C



```
10
11  int main(int argc, char** argv) {
12
13      int a = 10;
14      int b = 20;
15      int c = soma(a,b);
16      printf("A soma eh: %d", c);
17
18      return 0;
```

como o endereço de instruções dentro do *ISA*, e são geralmente representados pelo seu valor em hexadecimal. Alguns opcodes aceitam operandos, que são bytes que deverão estar imediatamente após eles, e o conjunto opcode e operando forma uma **instrução**. Tais operandos podem ser registradores, endereços de memória ou valores imediatos (constantes inteiras). Os primeiros 16 bytes gerados pela compilação do programa mostrado na Figura 3.34 estão demonstrados em sua forma hexadecimal e binária na Figura 3.35.

Figura 3.35. Exemplo de código de máquina gerado por um programa escrito em C



```
55 89 E5 83 E4 F0 83 EC 20 E8 98 09 00 00 C7 44…

10101011000100111100101100000111110010011110000100000111110110000100000111010001001100000001…
```

Programar utilizando apenas código de máquina é uma tarefa desafiadora, propensa a erros. Para isso, foi criado o que é chamado de linguagem Assembly, ou linguagem de montagem, que é uma tradução das instruções do código de máquina para palavras, também chamados de *mnemônicos* e seus operandos, que é mais legível para seres humanos. A partir de um arquivo com instruções escritas em Assembly, é possível gerar o código de máquina por um processo de montagem, ao se utilizar um software *assembler*.

Como Assembly é uma tradução direta dos opcodes e argumentos escritos em código de máquina, é possível construir duas sintaxes de linguagens de montagem para um mesmo conjunto de instruções. Para a arquitetura x86, existem duas muito utilizadas: a notação Intel e a ATT. A notação Intel é comumente utilizada em contexto de engenharia reversa em Windows, enquanto que a ATT é mais utilizada no contexto Linux.

Segue abaixo um exemplo da sintaxe Intel. Nela, a ordem dos operandos é ao contrário: 'MOV ECX, AABBCCDDh' significa colocar o valor imediato 'AABBCCDD' no registrador 'ECX', não é necessário sufixar o mnemônico da operação e a derreferência de um ponteiro é feita com o uso de colchetes.

```
mov ecx, AABBCCDDh
mov ecx, [eax]
```

Em contraste com a sintaxe Intel, abaixo está um exemplo de uso da sintaxe ATT. Nela, a ordem dos operandos é da esquerda para direita: 'movl $0xAABBCCDD, %ecx'

significa colocar o valor imediato 'AABBCCDD' no registrador ECX. Note que os registradores são prefixados com o símbolo '%', o mnemônico da operação é sufixado para indicar o tamanho do operando ('movl' indica operandos de 32-bits) e a derreferência de um ponteiro é feita com o uso de parênteses.

```
movl $0xAABBCCDD, %ecx
movl (%eax), %ecx
```

Na listagem abaixo, temos algumas das principais operações que um analista de malware irá se deparar no cotidiano. Para uma lista completa, é recomendado checar o site da Intel[22] para um melhor direcionamento na documentação.

- MOV, LEA: utilizados para movimentação de valores e passagem de ponteiros, respectivamente;
- PUSH, POP: operações de empilhamento e desempilhamento, respectivamente, realizadas na stack;
- ADD, SUB, MUL, DIV: operações aritméticas de soma, subtração, multiplicação e divisão;
- OR, AND, XOR, NEG, NOT: operações lógicas bit-a-bit de ou, e, xor, complemento de 1 e complemento de 2;
- JMP, CALL, RET: mudança do fluxo de código através de mudança incondicional, chamada e retorno de funções/procedimentos;
- CMP, TEST: utilizados para comparação, configuram o valor do registrador EFLAGS de acordo com a comparação realizada.

O processo de converter o código de máquina para Assembly, conhecido como disassembly, também é possível e é um dos recursos que facilita a inspeção a baixo nível de um código compilado. Esse procedimento é realizado através de ferramentas *disassembler*, e permitirá uma análise estática a nível de código de máquina, mais aprofundada do que a análise estática básica realizada nas seções anteriores. A Figura 3.36 representa o processo de disassembly, com o código de máquina à esquerda e sua representação Assembly à direita.

A Figura 3.37 representa, em alto nível, o processo de compilação e disassembly.

É importante destacar que binários escritos em linguagens que geram códigos gerenciados (não nativos) em *intermediate languages*, como Python e Java, que usam máquinas virtuais para interpretar seus códigos, ou que contenham um compilador *Just In Time (JIT)* embutido, como C#, terão instruções que não são específicas para uma arquitetura real de computadores. Ou seja, utilizam instruções que não serão interpretadas por um processador real, e sim um virtual, que irá realizar o procedimento de tradução dessas instruções para as instruções da arquitetura na qual o programa está sendo executado. Como exemplo desse tipo de instrução, na Figura 3.38 está exemplificado as instruções necessárias para chamar a função *print* em Python:

[22]https://www.intel.com/content/www/us/en/developer/articles/technical/intel-sdm.html

Figura 3.36. Exemplo do disassembly de um programa escrito em C

```
55                  push ebp
89 E5               mov ebp,esp
83 E4 F0            and esp,FFFFFFF0
83 EC 20            sub esp,20
E8 98 09 00 00      call exemplo1.401EC0
C7 44 24 1C 0A 00 00 mov dword ptr ss:[esp+1C],A
C7 44 24 18 14 00 00 mov dword ptr ss:[esp+18],14
8B 44 24 18         mov eax,dword ptr ss:[esp+18]
89 44 24 04         mov dword ptr ss:[esp+4],eax
8B 44 24 1C         mov eax,dword ptr ss:[esp+1C]
89 04 24            mov dword ptr ss:[esp],eax
E8 B4 FF FF FF      call exemplo1.401500
```

Figura 3.37. Processo de compilação e disassembly

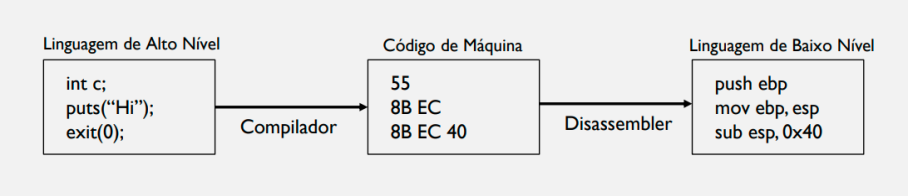


Figura 3.38. Exemplo do código interpretado pela máquina virtual python

```
1  print('Hello World')

1          0 LOAD_NAME                0 (print)
           2 LOAD_CONST               0 ('Hello World')
           4 CALL_FUNCTION            1
           6 POP_TOP
           8 LOAD_CONST               1 (None)
          10 RETURN_VALUE
```

3.7.3. Descompilação

Outra forma de realizar análise estática de código é através do processo de descompilação, realizada por um *decompiler*, no qual há a tentativa de reconstrução do código-fonte original a partir do código de máquina. Esse processo gera dados muito importantes para uma rápida análise do fluxo de código de um artefato, porém essa reconstrução não retornará o código-fonte original de um programa nativo, devido a perdas de dados resultantes do processo de compilação.

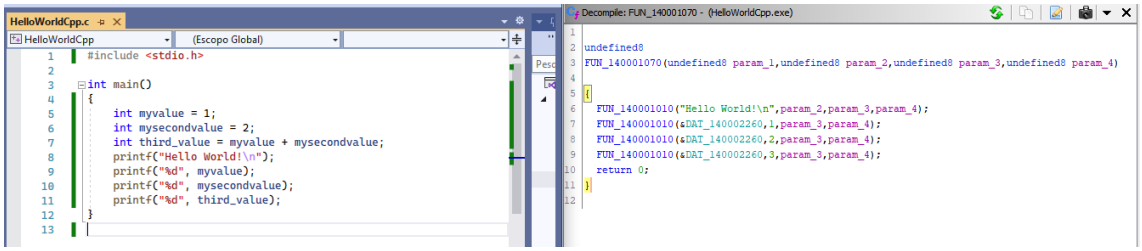
Nomes de variáveis locais são um exemplo de um desses dados perdidos e que não poderão ser retornados pelo processo de descompilação. Um exemplo de perda está demonstrado na Figura 3.39, em que o programa original escrito em C, à esquerda, contém os nomes das variáveis locais, enquanto que o mesmo programa descompilado pela ferramenta *Ghidra*[23] não possui, além dos nomes das funções que não estão mais presentes na versão descompilada.

Uma exceção à essas perdas significativas de informação são binários com código gerenciado. Devido a necessidade de um maior controle sobre variáveis e outros componentes de um código, é necessário que o binário gerado tenha metadados que um programa nativo não dispõe. Através desses metadados, é posssível gerar um descompilador mais efetivo que consegue encontrar informações de classes, namespaces e até nome de variáveis.

Para exemplificar esse conceito, na Figura 3.40 está explicitado, à esquerda, o

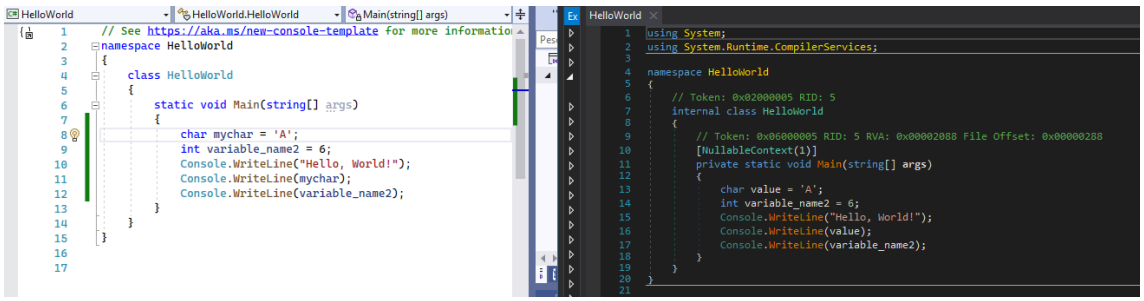
[23]https://ghidra-sre.org/

Figura 3.39. Processo de descompilação de um código nativo



código-fonte original de um programa simples escrito em C#, e na direita o seu código descompilado a partir da ferramenta *dnSpy*[24]. A diferença para um código nativo é notável, visto que o nome do namespace, da classe e até das variáveis locais foram restaurados com sucesso.

Figura 3.40. Processo de descompilação de um código gerenciado



3.7.4. Debugging

Essa etapa compreenderá a execução de um EXE em um processo debugger. Ela será importante para entender o passo-a-passo da execução de uma ou mais threads durante a execução do processo.

Na listagem abaixo, estão descritas algumas funções implementadas nos principais debuggers atuais:

- **step into**: prossegue a execução para a próxima instrução, inclusive para instruções dentro de funções, e pausa o processo;
- **step over**: prossegue a execução para a próxima instrução, executando todas as instruções dentro de uma determinada função, retornando para a função que chamou e pausando o processo;
- **step out**: caso o registrador de instrução esteja dentro de uma função, ele irá executar todas as instruções daquela instrução até o fim dela.

[24]https://github.com/dnSpy/dnSpy

Outra funcionalidade de debuggers importante para engenharia reversa são os *breakpoints*. Eles são tipos de interrupção que permitem a um *debugger* pausar a execução de um processo. Existem três tipos de breakpoints: de software, de memória e de hardware.

O breakpoint de software é representado na arquitetura x86 pela instrução *INT 3*, ou interrupção 3, que irá sinalizar para o *debugger* que o processo deverá ser parado e analisado. Quando o processador encontra essa instrução, ele irá sinalizar para o sistema operacional, que por sua vez irá sinalizar para o *debugger* que o processo foi pausado, permitindo realizar análises de memória e de código.

O breakpoint de hardware é dependente da arquitetura e do processador, pois são registradores físicos que identificam a condição para a parada do processo, e não uma instrução por si só, e por isso não são sobrescritos por operações, como os de software. Podem ser configurados breakpoints de hardware para leitura, escrita e execução. Seu principal uso é para identificar o momento de execução de um determinado byte que tem seu valor alterado constantemente, ou reside em uma região de memória alocada dinamicamente.

O breakpoint de memória é utilizado quando se sabe o intervalo de páginas de memória em que determinada execução de código poderá acontecer, mas não um endereço em específico. Ele é configurado a partir da mudança das permissões de páginas de memória para que qualquer leitura, escrita ou execução gere uma exceção a ser tratada pelo debugger. Ele, por sua vez, irá parar a execução para que o analista consiga dar prosseguimento ao processo de análise.

Existem diversas ferramentas que dão acesso a funcionalidades de debugging, sendo um dos mais conhecidos e usados em análise de malware a ferramenta x64dbg, ou x32dbg no caso de binários 32-bits. Na Figura 3.41, é possível perceber a tela principal do programa quando um binário é carregado a partir do menu **File -> Open**.

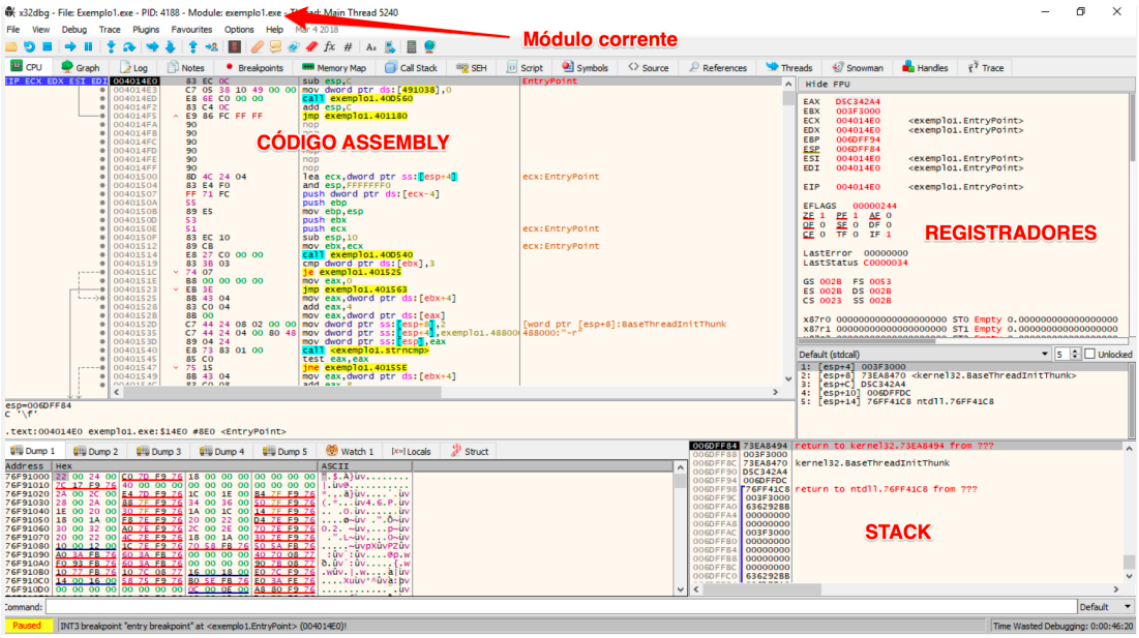
Alguns pontos importantes sobre essa janela:

- módulo corrente: informa para o analista qual módulo atual ele está analisando;
- código assembly: disassembly do módulo ou de outras partes;
- registradores: valores atuais dos registradores;
- stack: valores atualmente armazenados na stack de execução.

Utilizando os conhecimentos de análise reversa, vamos buscar dar um passo a mais na análise do nosso binário 'TextInputDocsx.exe'. A análise reversa, como visto nesta seção, é a etapa mais complexa e a que pode levar mais tempo do analista. O mais indicado, portanto, é mirar pontos específicos de interesse do código ao invés de fazer uma análise passo a passo de todo o binário.

Neste sentido, para esta análise reversa, escolhemos buscar um entendimento melhor do ponto onde paramos na análise de comportamento. Vimos que há uma comunicação do malware com a porta TCP/26457 na qual verificamos o envio de uma mensagem

Figura 3.41. Layout de tela principal do x64dbg



'LOGIN TestUser password'. Para simular o lado do servidor, utilizamos a ferramenta NetCat (comando: nc -l -p 26457).

Com o mesmo ambiente utilizado na análise de comportamento (incluindo o NetCat), executando o binário na ferramenta x32dbg, siga os passos listados a seguir. Eles servirão de guia na busca de um melhor entendimento da comunicação do malware com o servidor de comando e controle (C2) e de exemplo de como a ferramenta pode ser utilizada em futuras análises.

1. Execute o binário no x32dbg. Logo que iniciar o *debugging*, você perceberá que estará no módulo 'ntdll.dll'. Clique no botão 'run' para avançar para o 'EntryPoint' do código do binário a ser analisado;

2. Com o NetCat aguardando a conexão, execute o 'run' mais uma vez. Você deverá perceber que a mensagem 'LOGIN TestUser password' chegou no NetCat e que o código no x32dbg está no modo 'Running' (informação no canto inferior esquerdo);

3. Agora teremos que identificar a parte do código que está executando a comunicação com a porta 26457 e aguardando uma resposta. Para isso, mude para a aba 'Threads' e identifique a *thread* 'Main'. Ela estará com o valor 'UserRequest' no campo 'Wait Reason'. Clique 2x na *thread* e depois mude para a aba 'Call Stack'. Na aba 'Call Stack' é apresentada a lista de instruções chamadas com as mais recentes primeiro. Observe que a segunda instrução chama uma função da biblioteca mswsock, uma biblioteca responsável por comunicação de rede. Como o nosso interesse é monitorar o código que chama a função de comunicação, vamos inserir um *breakpoint* na mswsock. Para isso, selecione a linha e depois pressione F2. Você perceberá que um *breakpoint* foi inserido na aba 'Breakpoints';

4. Agora, na tela que está executando o NetCat, digite um conteúdo qualquer e depois digite <ENTER>;

5. Neste momento, o x32dbg deve estar parado na instrução imediatamente após a chamada à instrução 'mswsock.7464A460'. Clique em 'run' novamente;

6. De volta a tela do NetCat, você poderá observar que um novo conteúdo foi enviado pelo malware (<|ARQUIVO1|>AMIGO1<|>Windows 10 (Version 10.0, Build 17763, 64-bit Edition)<|>MSEDGEWIN10<|>);

7. Agora, experimente enviar um novo conteúdo na tela do NetCat em resposta ao malware. (Ex: novoconteudo). Até aqui, praticamente repetimos o que fizemos na análise de comportamento utilizando uma ferramenta de *debugging*, o que nos dá mais controle;

8. Agora, volte ao x32dbg para dar continuidade na análise do que será feito com a novo conteúdo enviado ao malware. Um atalho importante neste momento é o botão 'Run to user code' que fica na barra de ferramentas. Com ele, você avançará no restante das instruções da biblioteca mswsock, que não nos interessam, e vamos até a próxima instrução no código do nosso binário. O debugger avançará até a instrução '00598B8C' do nosso binário;

9. Agora, utilizando a função 'Step Over' ou F8, avance nas instruções até chegar na de endereço '005C1A69'. A partir deste endereço, teremos instruções que compararão o conteúdo digitado com uma string decodificada em tempo de execução. Observação importante: tive que seguir as instruções uma a uma até chegar em algo que me chamasse a atenção. Caso queira avançar diretamente para a instrução '005C1A69', pode optar por configurar um *breakpoint* nela e depois mandar executar o código com o 'run'. Para isso, utilizando a barra inferior de comandos do x32dbg digite o comando: setbpx 0x005C1A69 e depois <ENTER>;

10. Continue a execução das instruções com 'Step Over' até chegar na instrução '005C1AAD'. Neste ponto, observe que a execução da instrução anterior decodificou o conteúdo 'DOWNLOAD1';

11. Analisando as instruções seguintes, verificamos que a função chamada na instrução '005C1AB3' faz uma comparação do 'novoconteudo' com 'DOWNLOAD1'. Este pode ser um sinal importante para a nossa análise. Execute novamente os passos até aqui e forneça o conteúdo 'DOWNLOAD1' ao malware ao invés de 'novoconteudo';

12. Dando continuidade a análise, a comparação feita na função em '005C1AB3' agora será verdadeira. Na sequencia, na instrução '005C1AD2' será solicitada uma nova entrada no NetCat. Para facilitar o retorno à análise depois do fornecimento do novo conteúdo, inclua um *breakpoint* na instrução '005C1ADA';

13. No NetCat, digite o comando '1'. OBS: experimentei outros conteúdos de texto e uma exceção foi gerada porque a entrada não foi numérica;

14. Ao retornar ao x32dbg, execute o 'run' para retomar a partir do *breakpoint*. Na sequencia, na instrução '005C1B1B' um novo input será solicitado. Informe '1' também e, da mesma forma, configure um *breakpoint* para retornar ao código com facilitade. Neste caso, pode ser configurado para a instrução '005C1B28';

15. Nas instruções subsequentes, observe que há uma nova decodificação de string. Neste caso, será decodificada a string '.zip' após a execução da instrução '005C1B96';

16. Na sequencia, ao chegar na instrução '005C1BBB' utilizando 'Step Over', você poderá verificar que um path de arquivo estará na pilha (*stack*). No meu caso, o path é C:\Users\IEUser\AppData\Roaming\Microsoft\p1E8u4t5r2.zip. No entando, a cada execução, o nome do arquivo muda e o path se mantém. Inclusive, você poderá observar que o arquivo 'zip' foi criado;

17. Avançando nas instruções com 'Step Over', notei uma execção ao passar pela chamada no endereço '005C1C93'. Notei que isso ocorreu pelo fato de o arquvio 'zip' estar vazio. É possível que o arquivo seja criado por algum outro caminho que não o que seguimos. Para evitar a exceção e continuar a análise, preparei um zip com um conteúdo qualquer para sobrescrever o zip solicitado pelo malware. Faça o mesmo e reinicie o processo até aqui substituindo o seu zip pelo criado pelo malware antes da execução da instrução '005C1C93';

18. Na nova execução, você passará pela instrução '005C1C93' sem a exceção. Avance até a instrução '005C1D5A' e observe que um novo conteúdo foi decodificado: 'RtkNGUI.exe';

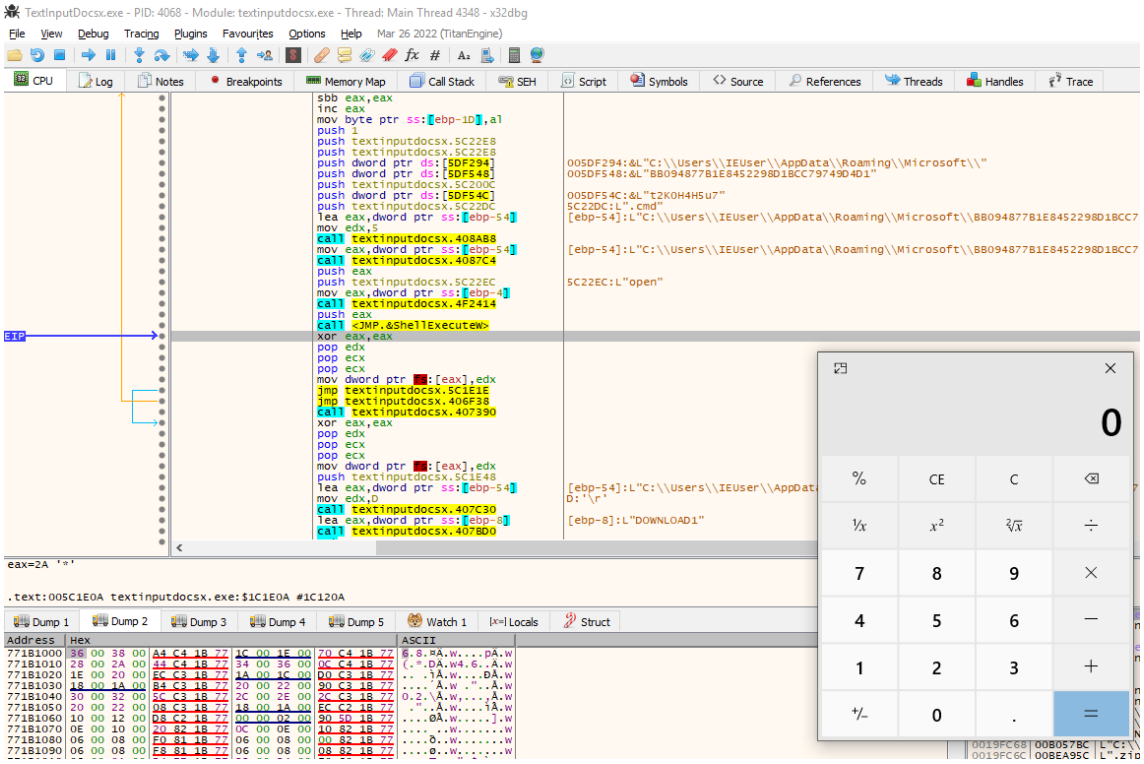
19. Avançando nas instruções com 'Step Over', pare na de endereço '005C1DAB', onde uma chamada para a função 'MoveFileW' é executada. Observe que, neste momento, o código renomeará o arquivo 'RtkNGUI.exe' para 'RtkNGUI.cmd'. Ou seja, ele espera que dentro do zip exista um arquivo de nome 'RtkNGUI.exe'. Neste momento, recriei o zip contendo o binário da calculadora do Windows renomeada para 'RtkNGUI.exe' e refiz o processo. Faça o mesmo;

20. Avance com 'Step Over' até a instrução '005C1E05', onde será executada uma chamada para a função 'ShellExecuteW'. Observe nos parâmetros para a função que o arquivo 'RtkNGUI.cmd' será executado;

21. Avance mais um passo com o 'Step Over' e veja que a calculadora será executada, como na Figura 3.42.

    Esta sequencia exemplifica como uma análise reversa pode ser realizada para buscar um maior entendimento do comportamento de artefatos maliciosos com o uso de um *debugger*. Até aqui, verificamos que o código executa um conteúdo adicional contido em um arquivo zip. Outras descobertas podem ser feitas com análises semelhantes.

Figura 3.42. Execução de comando pelo malware



3.8. Conclusão

Nesse minicurso apresentamos os conceitos básicos de análise de códigos maliciosos e introduzimos o leitor nas características particulares das soluções de análise para o ambiente *Windows*.

Acreditamos que a partir dos conhecimentos obtidos neste curso, os leitores estarão aptos a realizar procedimentos básicos de análise de malware, em especial nas seguintes áreas: *Profiling* do artefato suspeito (OSINT), análise automatizada, análise estática, análise comportamento e análise reversa. Esperamos, assim, contribuir para o desenvolvimento das pesquisas dos leitores em análise de códigos maliciosos de maneira geral.

Por fim, com o propósito de obter uma melhor compreensão dos temas abordados ao longo deste capítulo e apresentados no minicurso, recomendamos ao leitor aprofundar os conhecimentos por meio da leitura dos trabalhos referenciados.

Agradecimentos. Os autores agradecem à empresa Morphus Segurança da Informação por meio do Morphus Labs e ao Conselho Nacional de Desenvolvimento Científico e Tecnológico (CNPq), em especial via Programa de Bolsas de Produtividade em Pesquisa (processo número 306389/2020-7).

Referências

[Andriesse 2018] Andriesse, D. (2018). *Practical Binary Analysis*. No Starch Press.

[Bianco 2022] Bianco, D. (2022). Pirâmide da dor. http://www.sans.org/tools/the-pyramid-of-pain/.

[Botacin et al. 2018] Botacin, M. F., Geus, P. L., and Grégio, A. R. A. (2018). The other guys: automated analysis of marginalized malware. *Journal of Computer Virology and Hacking Techniques*, 14(1):87–98.

[Carvey 2014] Carvey, H. (2014). *Windows Forensic Analysis Toolkit*. Syngress.

[Dang et al. 2014] Dang, B., Gazet, A., Bachaalany, E., and Josse, S. (2014). *Practical Reverse Engineering*. Wiley.

[Galante et al. 2018] Galante, L., Botacin, M., Grégio, A., and de Geus, P. L. (2018). Malicious linux binaries: A landscape. *Anais Estendidos do XVIII Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*, pages 213–222.

[Gragido 2012] Gragido, W. (2012). Understanding indicators of compromise (ioc) part i. *Dostupné z http://blogs. rsa. com/understanding-indicators-of-compromise-ioc-part-i/Ověřeno ke dni*, 18(5):2016.

[Kim 2018] Kim, P. (2018). *The Hacker Playbook 3: Practical Guide To Penetration Testing*. Independently published.

[Kleymenov and Thabet 2019] Kleymenov, A. and Thabet, A. (2019). *Mastering Malware Analysis*. Packt Publishing Ltd.

[Liu et al. 2022] Liu, S., Feng, P., Wang, S., Sun, K., and Cao, J. (2022). Enhancing malware analysis sandboxes with emulated user behavior. *Computers & Security*, 115:102613.

[Mills and Legg 2020] Mills, A. and Legg, P. (2020). Investigating anti-evasion malware triggers using automated sandbox reconfiguration techniques. *Journal of Cybersecurity and Privacy*, 1(1):19–39.

[Monnappa 2018] Monnappa, K. A. (2018). *Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware*. Packt Publishing Ltd.

[Sahay et al. 2020] Sahay, S. K., Sharma, A., and Rathore, H. (2020). Evolution of malware and its detection techniques. In Tuba, M., Akashe, S., and Joshi, A., editors, *Information and Communication Technology for Sustainable Development*, pages 139–150, Singapore. Springer Singapore.

[Sikorski 2012] Sikorski, Michael; Honig, A. (2012). *Practical malware analysis: the hands on guide to dissecting malicious software*. no starch press.

[Skoudis and Zeltser 2003] Skoudis, E. and Zeltser, L. (2003). *Malware: Fighting Malicious Code*. Prentice Hall PTR, Upper Saddle River, NJ, USA.

[Wong 2018] Wong, R. (2018). *Mastering Reverse Engineering: Re-engineer your ethical hacking skills*. Packt Publishing Ltd.

Capítulo 4

Protegendo Redes de Computadores na era do Plano de Dados Programáveis

Arthur Selle Jacobs[1], Antônio João Gonçalves de Azambuja[2], Alberto Egon Schaeffer-Filho[3], Jéferson Campos Nobre[4], Juliano Araújo Wickboldt[5], Lisandro Zambenedetti Granville[6], Luciano Paschoal Gaspary[7], Weverton Luis da Costa Cordeiro[8]

Instituto de Informática (INF)
Universidade Federal do Rio Grande do Sul (UFRGS)

Abstract

Recent advances in Software Defined Networking (SDN) have expanded our ability to program the network towards the data plane. Through domain-specific languages like P4, network operators can quickly implement new protocols in forwarding devices, customize their functionality and develop innovative services. This flexibility comes, however, at a cost: security and correctness properties of the entire network (e.g., isolation and accessibility) become much more difficult to guarantee, because the behavior of the network is now determined by a combination of the configuration maintained by the control plane and data plane programs residing on forwarding devices. In this context, existing tools for network verification, which depend on a fixed and invariant model of the data plane, are inadequate for programmable data planes. This chapter aims to foster discussion on this subject, by presenting (i) how the concept of data plane programmability can be used to make computer networks more secure, and (ii) what major security challenges emerge along with the concept.

[1]E-mail: asjacobs@inf.ufrgs.br
[2]E-mail: azambuja@inf.ufrgs.br
[3]E-mail: alberto@inf.ufrgs.br
[4]E-mail: jeferson.nobre@inf.ufrgs.br
[5]E-mail: jwickboldt@inf.ufrgs.br
[6]E-mail: granville@inf.ufrgs.br
[7]E-mail: paschoal@inf.ufrgs.br
[8]E-mail: weverton.cordeiro@inf.ufrgs.br

Resumo

Os avanços recentes em Redes Definidas por Software (Software Defined Networking, SDN) expandiram nossa capacidade de programar a rede em direção ao plano de dados. Através de linguagens específicas de domínio como o P4, os operadores de rede podem rapidamente implementar novos protocolos em dispositivos de encaminhamento, personalizar suas funcionalidades e desenvolver serviços inovadores. Essa flexibilidade vem, no entanto, com um custo: as propriedades de segurança e de corretude em toda a rede (e.g., isolamento e acessibilidade) tornam-se muito mais difíceis de garantir, porque o comportamento da rede agora é determinado por uma combinação da configuração mantida pelo plano de controle e os programas do plano de dados que residem nos dispositivos de encaminhamento. Neste contexto, as ferramentas existentes para verificação de redes, as quais dependem de um modelo fixo e invariante do plano de dados, são inadequadas para planos de dados programáveis. Este capítulo visa fomentar discussão sobre esse assunto, ao apresentar (i) como o conceito de programabilidade do plano de dados pode ser usado para tornar as redes de computadores mais seguras, e (ii) quais os principais desafios de segurança que emergem juntamente com o conceito.

4.1. Introdução

Os avanços recentes em Redes Definidas por *Software* (*Software Defined Networking*, SDN) expandiram a capacidade de programar a rede em direção ao plano de dados. Segundo a *Open Networking Foundation* (OFN), as SDN são arquiteturas dinâmicas, gerenciáveis com adaptabilidade, que permitem a separação entre o controle da rede e as funções de encaminhamento. Com SDN, as infraestruturas passaram a ser mais flexíveis, logicamente centralizadas, e com uma *interface* aberta e bem definida. Através de linguagens específicas de domínio, os operadores de rede podem rapidamente implementar novos protocolos em dispositivos de encaminhamento, personalizar suas funcionalidades e desenvolver serviços inovadores.

O protocolo *OpenFlow* (OF) tem papel fundamental na habilitação das arquiteturas SDN, estabelecendo uma camada de abstração da rede física para o elemento de controle, permitindo a configuração e/ou manipulação do plano de dados da rede por meio de uma programação via *software* [McKeown et al. 2008, Garcia et al. 2018, Feferman et al. 2018]. Esse protocolo foi desenvolvido em um projeto *Open Source* no contexto de pesquisas realizadas na *Stanford University* e *University of California - Berkeley*. O protocolo OF permite que dispositivos de fabricantes diferentes suportem novas funções e protocolos, habilitando uma separação do plano de controle e dados. O plano de controle externo e centralizado permite a programação de encaminhamento unificado, políticas e suportes de segurança na criação de redes flexíveis e reconfiguráveis. A Figura 4.1 apresenta uma visão geral de SDN implementado com *OpenFlow* [Fernandes and Rothenberg 2014, McKeown et al. 2008].

A adoção do SDN abriu espaço para avanços no cenário de rede, permitindo a reorganização lógica da rede em um plano de aplicação, considerando a inteligência da rede; um plano de controle; e um plano de dados com a manipulação do fluxo [Kreutz et al. 2014]. A Figura 4.2 apresenta uma visão geral da arquitetura SDN, com os três planos conceituais e interfaces de comunicação: 1) Plano de aplicação (*Applica-*

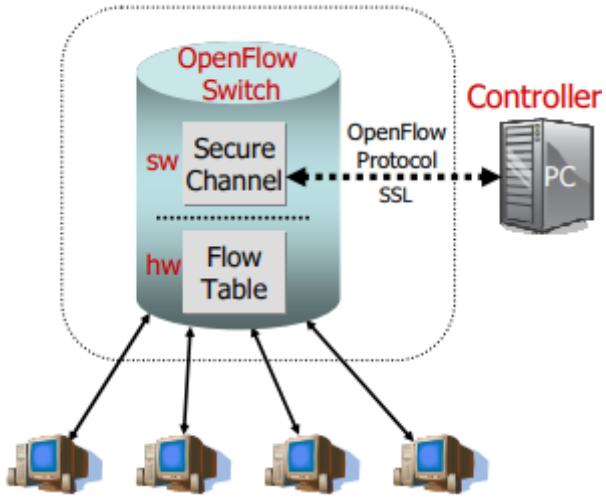


Figura 4.1: SDN implementado com OpenFlow (Fonte: [McKeown et al. 2008]).

tion plane): responsável por orquestrar serviços na malha de rede definida pelo plano de dados; 2) Plano de controle (*Control plane*): implementa abstrações de alto nível como gráfico de rede, objetivos de fluxo e intenções para aplicativos visando a configuração e gerenciamento de elementos de rede. São exemplos de controladores SDN: *ONOS*, *NOX*, *POX*, *Floodlight* e *Ryu*; e 3) Plano de dados (*Date plane*): considera os dispositivos, roteadores e/ou *switches*, que realizam encaminhamento de pacotes e processamento pela rede, como por *middleboxes* pela rede.

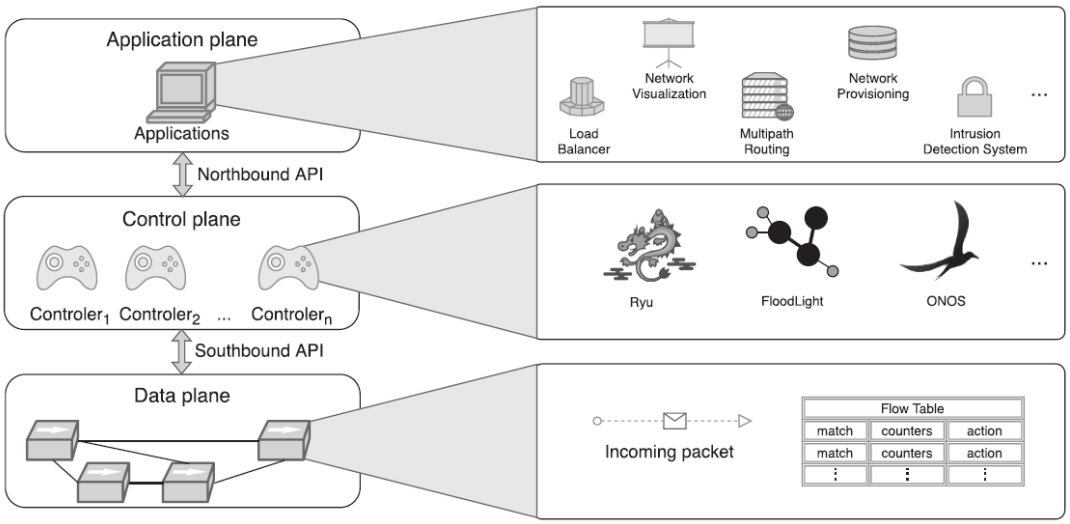


Figura 4.2: Planos conceituais SDN e interfaces de comunicação com planos de controle, aplicação e de dados (Fonte: [da Silva et al. 2015]).

Aplicações que usam o *software* do plano de encaminhamento permitem que sejam definidos quais pacotes devem ser processados e consequentemente ações devem ser tomadas. No entanto, a possibilidade de programar o plano de dados traz importantes desafios: as propriedades de segurança e de corretude em toda a rede (e.g., isolamento

e acessibilidade) tornam-se muito mais difíceis de garantir, porque o comportamento da rede agora é determinado por uma combinação da configuração mantida pelo plano de controle e os programas do plano de dados que residem nos dispositivos de encaminhamento. As ferramentas existentes para verificação de redes que dependem de um modelo fixo e invariante do plano de dados, são inadequadas para planos de dados programáveis.

O presente capítulo apresenta os principais fundamentos de programabilidade do plano de dados e P4. Tais fundamentos permitem explorar desafios de segurança relacionados essa programabilidade, assim como abordagens que a usem em mecanismos de segurança. Além disso, são discutidos exemplos e códigos-fonte para auxiliar na descrição de propriedades da utilização de programabilidade do plano de dados.

O presente capítulo está organizado da seguinte forma. Na Seção 4.2, são revisados os principais fundamentos em relação à Programabilidade do Plano de Dados e P4. Na Seção 4.3, a *Protocol-Independent Switch Architecture* (PISA) e o *pipeline* de processamento de pacotes são apresentados. Na Seção 4.4, propriedades de segurança de rede frente a aspectos de programabilidade são discutidos. Na Seção 4.5, abordagens para a verificação de políticas de segurança são apresentadas. Na Seção 4.6, mecanismos para a imposição de políticas de segurança são discutidos. Na Seção 4.7, cobre-se um conjunto de tópicos avançados sobre segurança em planos de dados programáveis. Finalmente, considerações finais e perspectivas de trabalhos futuros são apresentados na Seção 4.8.

4.2. P4 - *Programming Protocol-Independent Packet Processors*

Linguagens de programação de domínio específico, como o P4, foram propostas para especificar a lógica de processamento de pacotes de dispositivos no plano de dados programáveis por meio de uma arquitetura de alto nível independente de abstrações. Os operadores de rede podem rapidamente implementar novos protocolos em dispositivos de encaminhamento, personalizar suas funcionalidades e desenvolver serviços inovadores.

A flexibilidade advinda do plano de dados programável traz consigo um custo relacionado com as propriedade de segurança e corretude em toda a rede. As ferramentas existentes para verificação de redes, as quais dependem de um modelo fixo e invariante do plano de dados, são inadequadas para planos de dados programáveis. Diante das limitações inerentes nos ASICs *(Application Specific Integrated Circuits)* dos dispositivos de rede, que tornavam as alterações e implementações de novas funcionalidade um processo rígido e/ou inflexível, foram necessárias mudanças na forma de processamento dos pacotes.

As mudanças na forma de processamento dos pacotes têm foco para o plano de dados considerando o referido contexto em uma abordagem para descrever o gerenciamento do fluxo de dados e como fazer. Com este cenário emerge a linguagem de processamento de pacotes P4 em 2014, quando foi publicado *Programming Protocol-Independent Packet Processors* [Bosshart et al. 2014a], com uma uma gestão realizada pela organização *P4 Language Consortium* [9].

P4 [Bosshart et al. 2014b] é um DSL de alto nível para programar como os *switches* processam os pacotes. P4 é independente de destino, ou seja, adequado para descre-

[9]https://p4.org/

ver o comportamento de vários tipos de *switch* (por exemplo, ASICs de função xed, NPUs, comutadores de *software*, FPGAs). A linguagem abstrai a análise e o processamento de pacotes, fornecendo um encaminhamento generalizado. O código P4 é organizado logicamente da seguinte forma (veja Figura 4.3):

1. Declaração de dados: é uma seção que define o formato do cabeçalho do pacote e as informações de metadados que podem ser usadas para sua análise. Esta seção é mapeada em um cabeçalho e um barramento de metadados que transporta essas informações por todos os estágios de processamento. Os tipos de cabeçalho são declarados de forma semelhante às estruturas em C, ou seja, os campos são definidos em uma ordem específica e com um tamanho pré-determinado. As Figuras 4.4(a) e (b) ilustram a declaração do cabeçalho *Ethernet* padrão e de um protocolo arbitrário que utiliza *tags* como identificadores de origem e destino, respectivamente;

2. *Parser logic*: é uma seção que especifica como, quando e em que ordem cada um dos cabeçalhos deve ser analisado. Esta seção de um programa P4 é mapeada para os elementos *parser* e *deparser* do modelo de encaminhamento (veja a Figura 4.3). Esses elementos são então responsáveis por extrair o campo de cabeçalho dos pacotes em seu ingresso (*parser*) e no processo de tradução de estruturas de dados em um formato que possa ser armazenado e reconstruído na sequência no mesmo ou em um ambiente computacional diferente; e

3. *Match-action tables and control flows*: é uma seção que especifica tabelas do tipo *OpenFlow* capazes de corresponder em campos de cabeçalho arbitrários e modificação de cabeçalhos de pacotes (e metadados) por meio de ações personalizadas. Também expressa controle funções sem a ordem e circunstâncias que cada tabela deve ser executada. Esta seção é mapeada no contexto de configuração dos elementos compatíveis com *match-action* nos *pipelines* de entrada e saída. O *pipeline* de entrada executa o pacote (agnóstico de saída) e também estipula as intenções de saída (por exemplo, qual porta encaminhar um pacote). O *pipeline* de saída realiza ainda mais modificações de pacote necessárias, reescrevendo um endereço de origem de cabeçalho *Ethernet* com o endereço MAC da porta de saída.

Para [Bosshart et al. 2014b], três desafios foram estabelecidos: 1) Processamento de pacotes configuráveis: busca realizar a análise do programa para suportar a declaração de cabeçalhos, fazendo com que essa análise não fique dependente a um formato de pacote específico ou cabeçalho, tornando possível a inserção de novos cabeçalhos ao longo do tempo; 2) Flexibilização nas tabela de pacotes: busca configurar as tabelas de correspondência permitindo estruturas em série, paralelas ou até híbridas, para possibilitar inserção de novas funcionalidades no programa; e 3) Primitivas genéricas de processamento de pacotes: busca disponibilizar primitivas genéricas (por exemplo, *copy*, *add*, *remove* e *modify*) que permitam a alteração dos metadados e dados, podendo ser facilmente utilizadas.

Esses desafios têm fomentado pesquisas pela indústria e academia no contexto das tecnologias programáveis, dentre elas: 1) Switches Application Specific Integrated Circuits (ASIC): Barefoot Tofino, Cisco Doppler, Cavium (Xpliant), Intel Flexpipe; 2) Field Programmable Gate Arrays (FPGA): Altera, Xilinx; 3) Central Processing Unit (CPU):

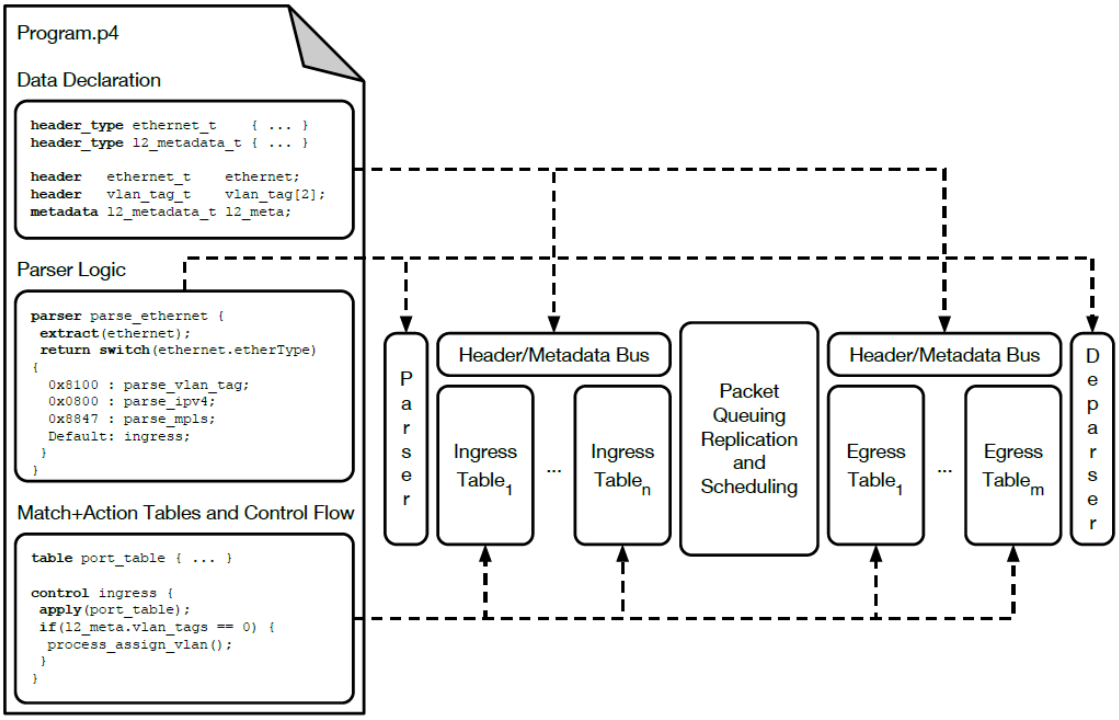


Figura 4.3: Seções de código P4 e mapeamento para o modelo de encaminhamento abstrato (Adaptado de [Kim et al. 2006]).

```
header ethernet {
  fields {
    dst_addr : 48;
    src_addr : 48;
    ether_type : 16;
  }
}
```

((a)) Ethernet header

```
header customl2proto {
  fields {
    src_tag : 16;
    dst_tag : 16;
    ethertype : 16;
  }
}
```

((b)) Custom L2 protocol header

Figura 4.4: Exemplos de declaração de cabeçalho P4.

DPDK, Open Vswitch, VPP, BPF; e 4) Network Processor Unit: Netronome, EZchip. A programação desses equipamentos tem uma complexidade gerada pelo fato de terem *interfaces* proprietárias com programação de baixo nível.

A linguagem P4, baseada na arquitetura de *switch* programável *Protocol-Independent Switch Architecture* (PISA), possibilita abstrair o plano de dados fazendo com que a complexidade da programação do comportamento dos equipamentos seja superada com uma programação de alto nível. Com o PISA insere-se um novo paradigma com *hardware* baseado em um *pipeline* de configuração e reconfiguração utilizando o modelo de *match-action*. Com isso, essa arquitetura permite maior flexibilidade sem impactar a perfomance e permite que o programador recupere o controle do plano de dados com funções aperfeiçoadas para uma rede específica.

Para os autores Garcia et al. [Garcia et al. 2018] a programabilidade do plano de dados apresenta os seguintes benefícios: 1) Controle: permite que o dispositivo de rede seja controlado da forma como planejado; 2) Capacidade: possibilita adicionar no-

vas funções aos dispositivos, já que possuem a flexibilidade na sua programação; 3) Exclusividade: implementa protocolos personalizados; 4) Eficiência: otimiza a capacidade para o uso dos recursos presentes nos dispositivos; 5) Confiabilidade: cria uma camada de proteção para não utilizar funções implementadas por terceiros; e 6) Monitoramento: acompanha o processamento e exporta dados para auferir o desempenho.

São três os objetivos definidos na linguagem P4 no contexto da programação de redes [Feferman et al. 2018]: 1) Independência de protocolos: o *switch* deve ter a capacidade para analisar qualquer tipo de protocolo, sendo que deve ser declarado inicialmente no programa como analisar o cabeçalho e as ações aplicando o modelo de *Match-action*; 2) Independência de arquitetura: o programa P4 busca uma abstração na camada física, fazendo com que a gestão das particularidades dos equipamentos seja realizada pelo próprio compilador; e 3) Reconfiguração em campo: o processamento de pacotes pode ser alterado durante a sua execução.

4.3. PISA e Pipeline de Processamento de Pacotes

A PISA (Protocol-Independent Switch Architecture), tem três partes: 1) *Parser*; 2) *Pipeline match-action*; e 3) *Deparser*. Baseado nessa arquitetura um programa P4 compreende declarações de cabeçalho (*header*), máquina de estado do analisador de pacotes (*parser*) e as tabelas de ação de correspondência (*pipeline match-action*). A linguagem P4 arca com a implementação de *parser*, o qual percorre os *headers* do começo ao fim, obtendo os valores de campo considerando o percurso. Esses valores são encaminhados para o processamento das tabelas *match-action*, identificando os campos de pacotes e metadados que serão lidos com as possíveis ações executadas como respostas. As declarações *header* determinam os nomes e as larguras dos campos para os cabeçalhos de protocolo para estabelecer o destinto de processamento do programa P4.

São os seguintes os componentes do P4 associados à arquitetura PISA: 1) *Parser* programável: busca identificar e especificar o formato para processamento, em uma cadeia de *bits*, dos cabeçalhos que estão presentes no pacote. *Parser* em sua definição determina a forma para identificar e reconhecer cabeçalhos ou sequências de cabeçalhos válidos nos pacotes; 2) *Header*: visa especificar a largura, restrições de tipos, valores e estrutura de uma série de campos de *bits* implementados pelo programador; 3) *Pipeline match-action*: tem como propósito a comparação de um ou mais campos dos cabeçalhos obtidos no *parser* com uma tabela programada pelo controlador, realizando uma correspondência de um campo e um valor que apresentam uma associação nas tabelas; 4) *Tables*: são mecanismos para realizar os processamentos dos pacotes. As tabelas são ações associadas aos pacotes conforme o *matching* executado nos *parsers*; e 5) *Actions*: conjunto de primitivas usados para uma certa função customizada, podendo incluir primitivas como por exemplo: *copy-header*, *remove-header*, *add*.

No modelo abstrato de encaminhamento PISA, apresentado na Figura 4.5, os *switches* encaminham pacotes por meio de uma analisador programável seguido por estágios de *match+action*, organizados em série. No primeiro, os pacotes entram na fase de análise ou de *parser* em que seus cabeçalhos são obtidos. No segundo os cabeçalhos são encaminhados para a etapa de *match-action*. Os *pipelines* de ingresso e egresso são utilizados para realizar essa atividade. Nos dois *pipelines* o funcionamento é semelhante,

em ambos o conteúdo dos cabeçalhos obtido é comparado com as tabelas *match-action* visando identificar uma ação que será realizada nesses cabeçalhos. No terceiro, antes do pacote sair do *switch*, o pacote é refeito com os cabeçalhos modificados. Com isso, o processo de encaminhamento é concluído [Bosshart et al. 2014b, Brum 2022].

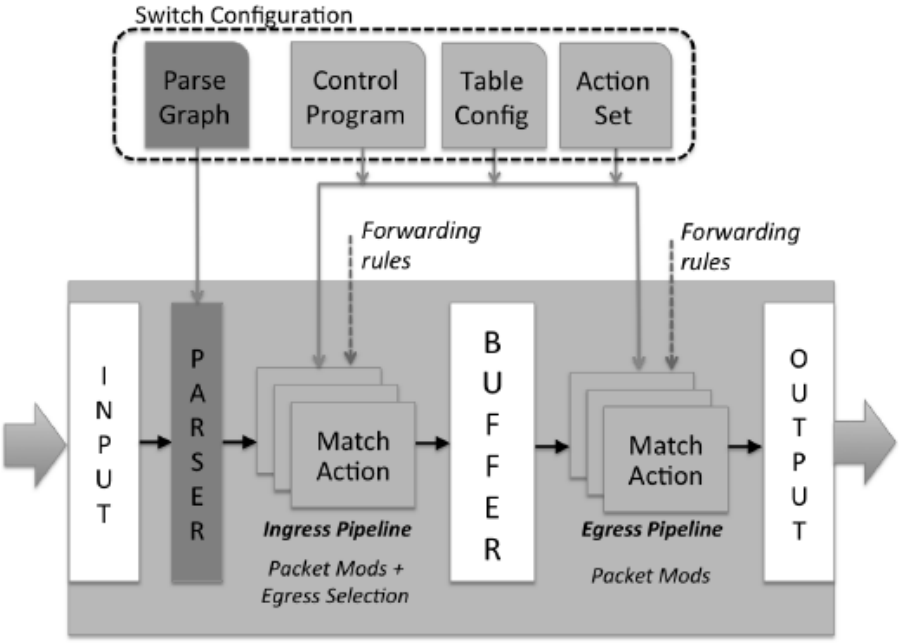


Figura 4.5: Abstração do modelo de encaminhamento PISA de dispositivos de encaminhamento programáveis (Fonte:[Bosshart et al. 2014b]).

4.4. Propriedades de Segurança de Rede

A capacidade de programar o plano de dados significa que é possível não apenas remodelar o comportamento da rede, mas também tornar a rede mais segura e confiável, melhorando sua confiabilidade, disponibilidade e integridade [Avizienis et al. 2004]. Isso pode ser feito por meio de um fluxo de serviços de segurança e confiabilidade, desenvolvidos a partir de blocos de construção provisionados diretamente nos dispositivos. Exemplos de blocos de construção incluem monitoramento e classificação de fluxo, bem como recursos de plano de dados de aplicação de políticas. Essa abordagem de provisionamento de serviços pode trazer várias vantagens exclusivas. Por exemplo, conformidade com a política pode ser garantida mesmo se o plano de controle e/ou um subconjunto de dispositivos de encaminhamento estiverem com defeito/comprometidos. Sendo assim, a medição da rede e a detecção de anomalias podem ocorrer de maneira verdadeiramente distribuída, com os *switches* acionando prontamente ações de contramedidas, se for necessário.

Além dos requisitos de desempenho, as redes modernas podem ter políticas de segurança (explícitas ou implícitas) que definem o fluxo de informação entre *hosts*. Em uma rede *multi-tenant*, por exemplo, o operador pode querer garantir que os *tenants* estejam completamente isolados uns dos outros ou que um *tenant* não possa negar ao outro acesso à rede. Várias classes de propriedades foram consideradas pela comunidade de pesquisa: independentes de contexto (propriedades agnósticas de sessões de fluxo), dependente de

contexto (referem-se aos fluxos de dados, por exemplo, iniciação da sessão), quantitativas (que são asseguradas com base em contadores, por exemplo, largura de banda garantida) e híbridas. À medida que os planos de controle e de dados se tornam mais complexos, torna-se mais difícil garantir que eles funcionem sempre corretamente. Para garantir que certas propriedades críticas sejam sempre satisfeitas, é vantajoso ter um mecanismo separado que seja apenas responsável por garantir que essas propriedades sejam respeitadas.

4.4.1. Modelagem e Análise de Políticas de Segurança

Uma maneira de expressar os requisitos que um sistema em rede deve atingir ou satisfazer é por meio de políticas de rede. A literatura é rica em soluções para especificação de políticas, verificação e aplicação. Boubata e Aib [Boutaba and Aib 2007], apresentam uma perspectiva histórica sobre a gestão de rede baseada em políticas. Os requisitos muitas vezes confiam em protocolos padrão para definir o que pode ser observado e executado (como endereços IP, portas TCP/UDP e outros campos de cabeçalho de protocolos padrão). A agenda de pesquisa de modelagem e análise de políticas para planos de dados programáveis deve se concentrar em três grandes questões: 1) como modelar e expressar políticas, 2) como traduzir/refinar políticas e 3) como lidar com conflitos entre elas.

4.4.1.1. Propriedade baseadas em políticas específicas

As soluções baseadas em políticas para o plano de dados programável deve considerar classes de propriedades para expressar requisitos de nível superior/inferior que um sistema necessita satisfazer. A questão é, quais são essas classes e propriedades? Trabalhos anteriores consideraram isolamento, acessibilidade e equivalência em SDN, mas sem fornecer uma discussão conceitual de nível superior [Khurshid et al. 2013, Lopes et al. 2015].

Uma propriedade é dita independente de contexto se for agnóstica de fluxo de sessões, ou seja, pode ser definida por pacote, sem recorrer ao estado das informações. Exemplos incluem isolamento e conectividade. Por outro lado, uma propriedade é dita dependente do contexto se aborda o fluxo de pacotes fluxos dependendo de sua semântica na rede. Um exemplo é o início da sessão, que expressa em que direção as conexões podem ser iniciadas na rede (por exemplo, um *host* pode enviar uma consulta de resolução de nomes, mas não receber um). Neste caso, diz-se que algum *host* tem permissão para iniciar uma sessão com outro. Outra classe agrega propriedades quantificáveis. Os exemplos incluem largura de banda garantida, limite de largura de banda e k-redundância. A primeira expressa uma taxa mínima que um *host* tem garantido para enviar pacotes para outro. O segundo expressa uma taxa máxima permitida para o fluxo de informações entre esses *hosts*. A terceira propriedade, k-redundância (k interpretada como uma métrica de redundância), é definida para um determinado *link* lógico e especifica a existência de k outros *links* lógicos conectando o mesmo conjunto de *hosts*. Esta propriedade pode ser útil para expressar canais de *backup* e/ou melhorar a robustez contra Ataques de negação de serviço distribuído (DDoS).

Por fim, as propriedades híbridas apresentam aquelas com características de mais de uma das classes acima. Um exemplo é o *link* equivalência, que expressa que os *links* lógicos conectando quaisquer duas entidades têm o mesmo isolamento, conectividade, lar-

gura de banda, configurações, etc. Uma noção estendida da propriedade de equivalência é a redundância k-equivalente. Um *link* é dito k-equivalente redundante se houver k outros *links* conectando o mesmo conjunto de *hosts* e com propriedades equivalentes. A oportunidade de pesquisa envolve a proposta de linguagens políticas expressivas que apoiem o nível de especificação de políticas, e que simultaneamente se aproximem mutuamente de metas conflitantes. Por exemplo, essas linguagens devem ser agnósticas do formato do cabeçalho do pacote ou da semântica de análise, mas também permitem a expressão de políticas de uma maneira que corresponda ao atual comportamento do *switch*.

4.4.1.2. Tradução de políticas de nível superior para nível inferior

Como o *hardware* de rede é personalizado sob demanda e sua semântica de análise de pacotes muda com o tempo, as soluções de especificação de políticas de segurança precisam ter uma dinâmica de revisão e atualização [Udupi et al. 2007, Craven et al. 2011]. Essas políticas em um contexto de planos de dados programáveis despertam oportunidades de pesquisas. Por exemplo: 1) Como garantir a consistência entre políticas de nível superior e inferior [Verma 2002, Westerinen et al. 2001] à medida que o comportamento do *switch* muda?; 2) Como pode-se expressar políticas baseadas em propriedades genéricas de segurança e confiabilidade, de uma forma que as torne verificáveis e aplicáveis em qualquer configuração de plano de dados? Neste contexto, é importante definir quais classes de propriedade são de interesse, bem como entender as suas implicações no projeto de mecanismos de tradução de políticas de segurança.

Em uma rede definida por *software*, cabe ao controlador garantir que as políticas de nível superior sejam mantidas [Kreutz et al. 2013]. No entanto, à medida que os aplicativos do plano de controle e os programas de comutação do plano de dados evoluem de forma independente e se tornam mais complexos, torna-se mais difícil garantir a consistência das políticas intra e internível. Esse cenário dinâmico exige soluções que vão além da tradução de políticas e também verificam inconsistências. Um exemplo é uma política declarando que duas redes A e B devem ser isoladas (um cenário de *datacenter* multilocatário) e uma permitindo pacotes do *host* $a_i \in A$ para $b_j \in B$. Outro caso é uma política que expressa que dois *hosts* estão simultaneamente isolados e conectados.

Pesquisas anteriores consideraram casos como conflitos entre diferentes tipos de políticas de nível superior [Lupu and Sloman 1999] e análise de conflito baseada em regras [Hamed and Al-Shaer 2006]. No entanto, eles são limitados, pois consideram linguagens de especificação de políticas de nível mais alto ou são fortemente acoplados a protocolos de rede tradicionais. Sendo assim, a criação de soluções que possam garantir consistência de políticas de nível superior a inferior, considerando a especificação abstrata de programas de comutação, apresenta-se como uma avenida de pesquisa promissora a ser explorada pela comunidade de pesquisa.

4.5. Verificação de Políticas de Segurança

A imposição e a verificação são abordagens complementares que podem ser aplicadas como solução para garantir que políticas de segurança sejam respeitadas. Usando a imposição, o plano de dados pode ser monitorado durante a execução para buscar e bloquear

ações que resultem em violações das políticas. A verificação (em conjunto com validação) se concentra em encontrar os *bugs* antes que os programas sejam implantados. Ela atua assegurando que o programa atenda às propriedades declaradas por seus requisitos.

Em um mundo onde os gerentes e operadores de rede podem redefinir o comportamento de dispositivos de encaminhamento, escrevendo seus próprios códigos para implementar alguma especificação de protocolo, a verificação e validação adequada (V&V) do código dos dispositivos torna-se crítica para o gerenciamento adequado das operações de rede e, portanto, a continuidade dos negócios. Em 2016, um roteador com defeito forçou a *Southwest Airlines* a cancelar 2.300 voos em quatro dias, resultando em uma perda de $ 74 milhões [Carey 2017]. Alguns anos depois (julho de 2020), uma configuração de roteador defeituosa na *Cloudflare* causou uma interrupção de rede que durou apenas 27 minutos, mas levou a uma grande interrupção dos serviços de Internet em todo o mundo por mais de uma hora [Winder 2020]. A comunidade de redes tem pesquisado soluções para lidar com defeitos de *software* antes que eles causem tais danos. Abordagens como metadados sintáticos, execução simbólica, asserções e testes funcionais têm sido aplicadas ao teste de *software* de plano de dados. Nesta seção são abordadas algumas das técnicas utilizadas para verificação e validação para *software* de plano de dados programáveis.

Tome como exemplo o trecho de código na Figura 4.6 de um programa de *switch* que faz NAT e ACL. Mesmo uma simples linha ausente como `ck.update(hdr.ipv4)` (no controle `TopDeparser`) não pode ser capturada durante o tempo de desenvolvimento sem informações extras fornecidas pelo programador (neste caso, os pacotes IPv4 de saída devem ter uma soma de verificação válida). Há casos mais complexos, no entanto. Liu et al. [Liu et al. 2018] descreveram um caso hipotético (inspirado em uma situação real de uma atualização de recurso de produto Cisco [Kazemian 2017]) em que um roteador defeituoso não estava mais aplicando as regras de ACL corretamente após uma atualização de *software*, devido a uma alteração na ordem em que o ACL e o NAT foram aplicados da versão anterior para a versão mais recente. Este caso também é mostrado na Figura 4.6, na seção `apply` do controle `TopPipe`. Outros casos, como verificar se um determinado cabeçalho é válido, podem evitar acessos para campos de cabeçalho indefinidos (como `hdr.ipv4.src_addr: lpm;` e `hdr.ipv4.dst_addr: lpm;`).

Exemplos como o da Figura 4.6 ilustram casos de defeitos de *software* por *omissão* (quando uma especificação de protocolo não está totalmente implementada no código do *switch*) e *fato incorreto* (quando o *switch* comportamento do código não está de acordo com sua especificação) [Travassos et al. 1999], respectivamente. Diversas abordagens foram desenvolvidas para verificar se um dado plano de dados satisfaz um conjunto de propriedades pretendidas [Son et al. 2013, Dobrescu and Argyraki 2014, Lopes et al. 2015, Panda et al. 2017]. No entanto, aqueles que são capazes de modelar programas P4 não podem raciocinar sobre propriedades específicas do programa. No restante desta seção, o estado da arte sobre verificação e validação de *software* de plano de encaminhamento (V&V) será revisado a partir de duas técnicas: verificação de planos de dados com asserções e via análise de fluxo de dados.


```
#include <core.p4>
#include "vss.p4"

header ethernet_t {
  bit<48> dst_addr;
  bit<48> src_addr;
  bit<16> ether_type;
}
header ipv4_t {
  bit<4> version;
  bit<4> ihl;
  bit<8> diffserv;
  bit<16> totalLen;
  bit<16> id;
  bit<3> flags;
  bit<13> fragOffset;
  bit<8> ttl;
  bit<8> protocol;
  bit<16> checksum;
  bit<32> src_addr;
  bit<32> dst_addr;
}
struct headers {
  ethernet_t eth;
  ipv4_t ipv4;
}
```

```
parser TopParser(packet_in pkt, out headers hdr) {
  state start {
    hdr.extract(pkt.eth);
    transition select(pkt.eth.ether_type) {
      0x800: parse_ipv4;
      default: accept;
    }
  }
  state parse_ipv4 {
    hdr.extract(pkt.ipv4);
    transition accept;
  }
}

control TopDeparser(inout headers hdr, packet_out pkt) {
  Ck16() ck;
  apply {
    pkt.emit(hdr.eth);
    if (hdr.ipv4.isValid()) {
      ck.clear();
      hdr.ipv4.checksum = 16w0;
      hdr.ipv4.checksum = ck.get();
    }
    pkt.emit(hdr.ipv4);
  }
}
```

```
control TopPipe(inout headers hdr, in error parseError,
    ↪ in InControl inCtrl, out OutControl outCtrl) {
  action allow() { }
  action deny() { outCtrl.outputPort = DROP_PORT; }
  action rewrite(bit<32> src_addr, PortId port) {
    hdr.ipv4.src_addr = src_addr;
    outCtrl.outputPort = port;
  }
  table acl {
    actions = { allow; deny; }
    key = { hdr.ipv4.src_addr: lpm; }
  }
  table nat {
    actions = { rewrite; }
    key = { hdr.ipv4.dst_addr: lpm; }
  }
  apply {
    if (hdr.ipv4.isValid()) {
      acl.apply();
      nat.apply();
    }
  }
}

VSS(TopParser(), TopPipe(), TopDeparser()) main;
```

Figura 4.6: Código simples NAT e ACL P4 para o modelo de switch VSS (código adaptado de [Liu et al. 2018]).

4.5.1. Verificação de Segurança usando Data Flow Analysis

Análise de fluxo de dados [Hecht 1977] tem sido amplamente utilizado para otimização de código por compiladores e detecção de anomalias de programa por meio de análise estática de programa. Em geral, classifica cada ocorrência de uma variável no programa como **definição** ou como **uso**. Essas técnicas, portanto, usam informações de fluxo de dados do programa para derivar requisitos de teste.

Para detectar o uso indevido de valores de dados devido a erros de codificação, o teste de fluxo de dados pode ser usado como critério de teste estrutural [Vergilio et al. 1997]. Rapps e Weyuker propuseram o *Def-Use Graph*, uma extensão do *Control Flow Graph* (CFG) [Rapps and Weyuker 1985]. Em sua proposta, são adicionadas informações ao CFG sobre o fluxo de dados do programa, que identifica as associações em que um valor é atribuído a uma variável (*definição de variável*) e onde esse valor é lido (*uso de variável*). Os testes de fluxo de dados são gerados a partir dessas associações. De acordo com o modelo de fluxo de dados [Vergilio et al. 1997], sempre que um valor é armazenado em um local de memória ocorre a definição da variável. É o caso quando a variável está no lado esquerdo de uma atribuição de comando, comando de entrada ou chamadas de procedimento como um parâmetro de saída.

Para gerar os testes de fluxo de dados, todos os subcaminhos são mapeados entre a atribuição de uma variável (definição) aos pontos em que a variável é utilizada (uso). Há duas maneiras de usar uma variável: computando a variável *(c-uses)*, onde um valor é usado em um cálculo ou declaração de saída; ou usando predicados *(p-uses)* que ocorrem sempre que um valor é usado em uma instrução de predicado. A notação para representar esses padrões é baseada em Rapps e Weyuker [Rapps and Weyuker 1985]:

- d – definido, inicializado
- u – usado

Existem três possibilidades para a primeira ocorrência de uma variável através de um caminho de programa. O símbolo $\sim$ é usado para denotar que antes disso a variável não existia [Rapps and Weyuker 1985]:

Tabela 4.1: Teste de anomalias (Fonte: [Rapps and Weyuker 1985]).

	Anomalia	**Descrição**
dd	Definido e definido novamente	Não inválido, mas suspeito. Possível *bug*.
du	Definido e usado	Permitido. Caso normal.
ud	Usado e definido	Permitido.
uu	Usado e usado novamente	Permitido.

1. ∼d – a variável não existe, então está definida (d)
2. ∼u – a variável não existe, então é usada (u)
3. ∼k – a variável não existe, então ela é destruída (k)

Dessas três possibilidades, apenas a primeira está correta, onde a variável não existia e então ela é definida. A segunda está incorreta porque você não pode fazer um uso seguro de uma variável a menos que ela tenha sido definida antes e a terceira provavelmente também está incorreta, porque uma variável está sendo destruída antes de ser criada. Na linguagem P4, matar ou destruir variáveis não é uma construção da linguagem.

Def-use paths (também chamado de *du-paths*) é um par ordenado (d, u), onde uma instrução chamada *d* contém uma definição de uma variável *v*, que é usada em uma instrução *u* em um programa [Rapps and Weyuker 1985]. Table 4.1 lista as combinações de uso e as consequências correspondentes.

A identificação de uma anomalia por meio do teste de fluxo de dados nem sempre representa um resultado incorreto na execução da aplicação. Embora possa ser uma anomalia inofensiva, vale a pena investigar porque muitas vezes representa um sinal de erro do programador ou más práticas de codificação.

Um método para detectar as anomalias do fluxo de dados foi desenvolvido por Fosdick e Osterweil [Fosdick and Osterweil 2011]. A ideia básica é calcular as chamadas expressões de caminho em um gráfico de fluxo usando algoritmos de análise de fluxo de dados. Uma expressão de caminho descreve todas as ações executadas em uma variável ao longo dos caminhos mapeados. Anomalias no fluxo de dados podem ser detectadas pela sequência de definições e usos que ocorrem com cada variável ao longo do caminho.

A Figura 4.7 descreve parte da análise de fluxo de dados para o código mostrado na Figura 4.6. O gráfico de fluxo de dados para este código é mostrado na parte inferior da figura, ou seja, os caminhos possíveis são mapeados e cada nó tem a anotação de uso (leitura) e definição (escrita) para cada variável. Em seguida, é realizada a análise do fluxo de dados para cada caminho. Na parte superior da figura, a análise da definição e sequência de uso para cada variável identificada para os fluxos #1 e #2. Para este código, pode-se ver um *bug* e um *bug* em potencial, de acordo com as possíveis primeiras ocorrências das variáveis e as possibilidades de pares mostradas na Tabela 4.1.

O teste de fluxo de dados é comumente aplicado para testes de unidade, ou seja, testes de uma unidade de programa. No entanto, existem investigações que estendem o teste de fluxo de dados para testes de integração [Horgan and London 1991]. Para diferentes níveis de teste (unitário, integração, sistema) são aplicados diferentes critérios de

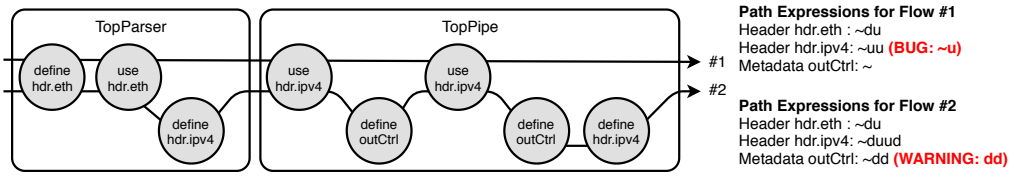


Figura 4.7: Análise parcial de fluxo de dados para o trecho de código na Figura 4.6.

teste de fluxo de dados, o que limita de alguma forma o número de caminhos explorados para identificar defeitos. Além disso, também existem trabalhos que estendem a aplicação dessa técnica de teste para testar *software* e componentes orientados a objetos.

4.5.1.1. Visão Geral

A presente seção apresenta uma visão geral de como usar a análise de fluxo de dados para descobrir problemas e vulnerabilidades de segurança em programas de *switch* P4, usando a Figura 4.8 como base. A abordagem usa uma especificação JSON do código do *switch*. Portanto, a primeira etapa do processo de verificação é a geração de uma especificação JSON a partir de um programa P4. O arquivo JSON utilizado é o mesmo esperado pelo switch BMv2, *software switch* utilizado popularmente para avaliar as funcionalidades de uma especificação de programa P4. O lado esquerdo da Figura 4.9 mostra um trecho de código basic.p4, enquanto no lado direito é mostrado o mesmo trecho de código em JSON.

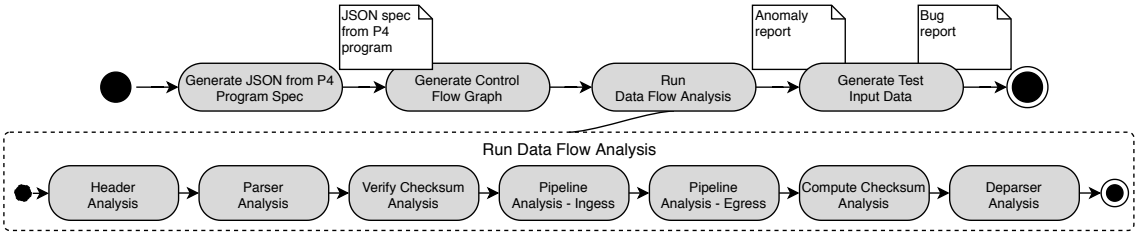


Figura 4.8: Visão geral da análise do fluxo de dados do código do switch P4.

Com base na especificação JSON, gera-se o gráfico de fluxo de controle do programa P4. Este gráfico contém todos os caminhos de execução possíveis dentro da especificação P4. Para ilustrar, o gráfico de fluxo de controle representado na Figura 4.10, do código do *switch* basic.p4[10], mostra oito caminhos de execução possíveis.

Para cada caminho possível, executa-se a análise de fluxo de dados. Este processo (detalhado na parte inferior da Figura 4.8) gera um relatório indicando expressões de caminho em cada variável e campos de cabeçalho, bem como anomalias identificadas em expressões de caminho de acordo com a teoria da análise de fluxo de dados.

[10]O código fonte do *switch* basic.p4 foi obtido dos tutoriais disponíveis na linguagem oficial do P4 7repositório do github. Uma cópia do arquivo pode ser encontrada em https://github.com/ComputerNetworks-UFRGS/p4-data-flow/blob/master/examples/basic.p4

```
/* -- P4_16 -- */
#include <core.p4>
#include <v1model.p4>

const bit<16> TYPE_IPV4 = 0x800;

/*************************
***** H E A D E R S *****
*************************/

typedef bit<9> egressSpec_t;
typedef bit<48> macAddr_t;
typedef bit<32> ip4Addr_t;

header ethernet_t {
   macAddr_t dstAddr;
   macAddr_t srcAddr;
   bit<16> etherType;
}
```

```
{
"program" : "basic.p4",
 "_meta_" : {
  "version" : [2, 7],
  "compiler" : "https://github.com/p4lang/
      ↪ p4c"
 },
 "header_types" : [ {
   "name" : "scalars_0",
   "id" : 0,
   "fields" : []
  }, {
   "name" : "ethernet_t",
   "id" : 1,
   "fields" : [
    ["dstAddr", 48, false],
    ["srcAddr", 48, false],
    ["etherType", 16, false]
   ] },
```

Figura 4.9: Trecho de código do programa basic.p4.

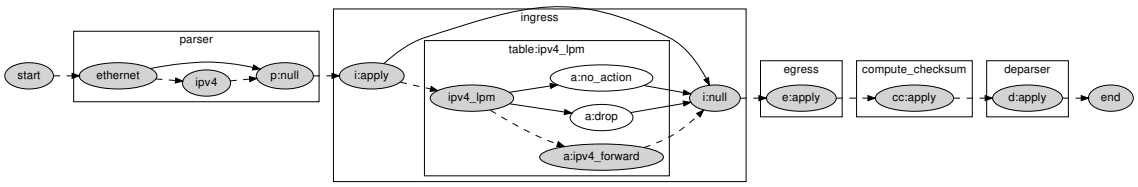


Figura 4.10: Gráfico de fluxo de controle para o programa de switch basic.p4.

A Figura 4.11 mostra as expressões de caminho obtidas para o caminho de execução destacadas nas linhas pontilhadas e elipses cinzas na Figura 4.10. Para ilustrar, considere a expressão de caminho do campo `ipv4.ttl:` DUDUU. A primeira definição (D) ocorre durante a extração do cabeçalho `packet.extract(hdr.ipv4)`. Então, um use (U) seguido por definição ocorre em `hdr.ipv4.ttl = hdr.ipv4.ttl-1`. Finalmente, dois usos ocorrem para cálculo do *checksum* e *deparsing* dos pacotes.

Existem casos de variáveis usadas mas nunca definidas, que são reportadas como **bugs**. Para outros *bugs* potenciais relatados, eles são usados como entrada para gerar pacotes de teste. Nesta etapa, implanta-se a especificação JSON no switch BMv2 e injeta-se pacotes de teste que tentam explorar as expressões de caminho anômalo. Os pacotes de teste são cuidadosamente projetados para explorar o fluxo de execução que causa a expressão do caminho anômalo e exercitá-lo. Caso ocorra um comportamento anormal do *switch* (por exemplo, um pacote que deveria ter sido descartado é encaminhado ou um pacote é descartado silenciosamente), então um *bug* é revelado.

Tendo fornecido uma visão geral da solução, a seguir apresenta-se com mais detalhes o processo de análise de fluxo de dados e a geração automatizada de pacotes de teste. Sendo assim, assume-se sem perda de generalidade o uso do modelo *V1Switch*, que é composto por blocos *parser*, *checksum*, *ingress*, *egress*, *compute checksum* e *deparser*. Observe que com esta mesma metodologia, é possível implementar uma solução funcional para outras arquiteturas existentes.

```
Execution Path:
   0 -> parsers/start -> parsers/parse_ipv4 -> parsers/null -> ingress/node_2 ->
       ↪ ingress/MyIngress.ipv4_lpm -> ingress/MyIngress.ipv4_lpm/MyIngress.
       ↪ ipv4_forward -> ingress/null -> egress/null -> compute_checksum -> deparsers
DF_Table:
   standard_metadata
      egress_spec: D
   ethernet
      dstAddr: DUDU
      srcAddr: DDU
      etherType: DUU
   ipv4
      version: DUU
      ihl: DUU
      diffserv: DUU
      totalLen: DUU
      identification: DUU
      flags: DUU
      fragOffset: DUU
      ttl: DUDUU
      protocol: DUU
      hdrChecksum: DDU
      srcAddr: DUU
      dstAddr: DUUU
      $valid$: UU
   MyIngress.ipv4_forward
      dstAddr: PU
      port: PU
```

Figura 4.11: Exemplo de expressões de caminho para o código do switch basic.p4. P significa passagem de parâmetro. O caminho segue o JSON gerado pelo compilador p4c para o modelo de switch simples bmv2.

A análise de cada caminho de execução dentro de um código de *switch* P4 compreende o processamento de cada um dos componentes do *switch* nesse caminho: definições de cabeçalho, *parser* e controles, conforme mostrado nas tarefas descritas na parte inferior da Figura 4.8. O processo realizado em cada tarefa é descrito em mais detalhes na literatura [Birnfeld et al. 2020].

A saída da etapa de análise de fluxo de dados é uma lista de possíveis *bugs*. *Bug* potencial significa que ele pode realmente ser classificado como um *bug* caso de teste possa levar a uma situação defeituosa. Assim, o último passo da proposta é verificar esses possíveis *bugs* para confirmar se realmente são *bugs* no programa.

Para exercitar esses casos, faz-se necessária uma abordagem adicional, como por exemplo um gerador de pacotes para exercitar os caminhos com potenciais *bugs*. Com isso, pode-se gerar pacotes para confirmar os possíveis *bugs* identificados em um código de *switch* executando no BMv2. Sendo assim, faz-se necessário preencher as tabelas do *switch* com um conjunto mínimo de regras que exercitam o caminho a ser explorado, e então enviar um conjunto de pacotes de teste, fixando em zero/indefinido os casos suspeitos. Caso o pacote não seja processado de acordo com o resultado esperado (*e.g.*, encaminhado quando deveria ter sido descartado), então o *bug* é confirmado.

Apenas um pequeno subconjunto de *bugs* relatados precisa dessa inspeção mais detalhada usando essa técnica de teste estrutural, como por exemplo o caso de variáveis que são definidas duas vezes sem uso no meio (*i.e.*, sua expressão de caminho tem uma

subsequência DD). A lógica é que a ocorrência de uma variável escrita duas vezes pode indicar um problema na lógica de programação do *switch*. Entre as etapas de um processo de análise de fluxo de dados de código de *switch* P4, a geração de pacotes de teste é a única que pode exigir esforço manual para gerar pacotes usando um gerador de pacotes.

4.5.1.2. Estudo de Caso de Vulnerabilidades: NAT

A aplicação de análise de fluxo de dados permitiu identificar um pequeno problema de segurança no programa simple_nat, que implementa um NAT com suporte a IPv4. A técnica revelou três *bugs* em seu código. A Figura 4.12 mostra os caminhos de execução (e respectivas expressões de caminho dos campos de cabeçalho) relacionados a dois desses *bugs*. A primeira refere-se à possibilidade de pacotes sem cabeçalho IPv4 serem processados pela tabela ipv4_lpm. Observe no primeiro caminho de execução que após analisar o cabeçalho *Ethernet (parsers/ethernet)*, o analisador vai para o estado de saída *(parsers/null)*. Este é o caso do caminho `default: accept;` no `transition select(hdr.ethernet.etherType)`, como pode ser visto no código simple_nat-16.p4 disponível no repositório. Mais tarde no caminho de execução (ingress/node_4 -> ingress/ipv4_lpm), uma condicional com erros permite que o pacote sem um cabeçalho IPv4 válido seja processado pela tabela ipv4_lpm.

O código defeituoso neste caso é `if (meta.meta.do_forward == 1w1 && hdr.ipv4.ttl > 8w0)`. Da especificação P4_16[11], campos de cabeçalho que não são definidos antes do uso podem ter um valor indefinido e, portanto, levar a mudança a um comportamento anormal. De fato, após exercitar o *bug* usando PTF, descobriu-se que um pacote sem cabeçalho IPv4 que deveria ter sido descartado na verdade foi encaminhado pelo *switch* (defeito de *software* devido a fato incorreto). Para evitar isso, os autores do código do *switch* simple_nat deveriam ter verificado a validade do cabeçalho IPv4, usando o método `hdr.ipv4.isValid()`.

O segundo *bug* está relacionado aos campos de cabeçalho TCP não extraídos, mas mesmo assim traduzidos. Observe no segundo caminho na Figura 4.12 que o cabeçalho IPv4 é analisado, mas não o cabeçalho TCP (`parsers/parse_ethernet -> parsers/parse_ipv4 -> parsers/parse_null`). Então, ao chegar em `egress/send_frame`, a ação `do_rewrites` é acionada sem verificar se o cabeçalho TCP é válido (um defeito de *software* por omissão). Por fim, o terceiro *bug* encontrado está relacionado ao uso do campo IPv4 TTL sem uma definição prévia (fato incorreto), causando uma aplicação incorreta da tabela `ipv4_lpm`.

4.5.2. Verificação de Segurança em Programas P4 com Asserções

Asserções têm sido popularmente usadas como uma abordagem de verificação de rede capaz de modelar e verificar (na compilação) propriedades gerais de segurança e correção de programas P4. Uma das soluções nesta linha é o ASSERT-P4 [Neves et al. 2018], que fornece uma linguagem de asserção expressiva permitindo aos programadores especificar suas propriedades pretendidas simplesmente anotando seus programas P4.

[11] P4_16 Specification: https://p4.org/p4-spec/docs/P4-16-v1.0.0-spec.html

```
0 -> parsers/start -> parsers/parse_cpu_header -> parsers/parse_ethernet -> parsers/
    ↪ null -> ingress/if_info -> ingress/if_info/set_if_info -> ingress/nat ->
    ↪ ingress/nat/nat_hit_ext_to_int -> ingress/node_4 -> [continua]
[continua] -> ingress/ipv4_lpm -> ingress/ipv4_lpm/set_nhop -> ingress/forward ->
    ↪ ingress/forward/set_dmac -> ingress/null -> egress/node_9 -> egress/send_to_cpu
    ↪  -> egress/send_to_cpu/do_cpu_encap -> egress/null -> compute_checksum ->
    ↪ deparsers
   ethernet
      dstAddr: DDU
   ipv4
      version: UU
      ihl: UU
      diffserv: UU
      totalLen: UU
      identification: UU
      flags: UU
      fragOffset: UU
      ttl: UUDUU
      protocol: UUU
      srcAddr: UUU
      dstAddr: UUU
   tcp
      srcPort: UU
      dstPort: UU
      seqNo: UU
      ackNo: UU
      dataOffset: UU
      res: UU
      flags: UU
      window: UU
      urgentPtr: UU

0 -> parsers/start -> parsers/parse_cpu_header -> parsers/parse_ethernet -> parsers/
    ↪ parse_ipv4 -> parsers/null -> ingress/if_info -> ingress/if_info/set_if_info ->
    ↪  ingress/nat -> ingress/nat/nat_hit_int_to_ext -> ingress/node_4 -> [continua]
[continua] -> ingress/ipv4_lpm -> ingress/ipv4_lpm/set_nhop -> ingress/forward ->
    ↪ ingress/forward/set_dmac -> ingress/null -> egress/node_9 -> egress/send_frame
    ↪ -> egress/send_frame/do_rewrites -> egress/null -> compute_checksum ->
    ↪ deparsers
   ethernet
      dstAddr: DDU
   ipv4
      hdrChecksum: DDU
   tcp
      seqNo: UU
      ackNo: UU
      dataOffset: UU
      res: UU
      flags: UU
      window: UU
      urgentPtr: UU
```

Figura 4.12: Execution paths with faulty behavior for simple_nat.

Uma vez anotado, um programa é executado simbolicamente, com asserções sendo verificadas enquanto todos os seus caminhos são percorridos. Dado que o tempo gasto para realizar a execução simbólica cresce exponencialmente com a complexidade do programa, também apresenta-se uma variedade de técnicas de otimização que podem ser empregadas para reduzir o tempo de verificação e o número de instruções executadas. As técnicas consistem em usar paralelização da execução simbólica, sinalizadores de otimização do compilador, anotações de código para restringir os pacotes e o fluxo de controle, uma estratégia de relatório de erros para otimizar as operações de E/S e descoberta de violação de asserção e fatiar o programa para reduzir a complexidade do modelo.

Os desenvolvedores usam asserções para expressar propriedades de programas P4. Uma *linguagem de asserções* é necessária para capturar comportamentos de processamento de pacotes e facilitar a tarefa de especificar propriedades de rede complexas. Isso inclui o raciocínio sobre a formação de pacotes, encaminhamento e propriedades de fluxo de controle, cujo comportamento pode depender não apenas do estado das variáveis do programa em um local específico, mas também de como o programa manipula os pacotes em outros pontos do código. Para atingir esse objetivo, o ASSERT-P4 introduz uma linguagem de asserção usando o mecanismo de anotação de código disponível em P4. A solução define uma anotação chamada `assert`, permitindo que o desenvolvedor e/ou terceiros expressem/interpretem propriedades de forma intuitiva.

4.5.3. Visão Geral

A Figura 4.13 resume a gramática da linguagem de asserção. Assemelha-se a asserções do estilo C encontradas em linguagens de programação tradicionais, diminuindo a barreira para a adoção. No entanto, o conceito de asserção no escopo de programabilidade do plano de dados geralmente é mais amplo e, inclui elementos como *location-restricted* e *location-unrestricted*. Um elemento com restrição de localização é aquele que testa o valor de uma variável de programa onde a asserção é especificada, como em linguagens de programação tradicionais como C ou Java. Os de localização irrestrita, em contraste, aplicam-se a todo o programa de plano de dados. Eles podem ser usados, por exemplo, para garantir propriedades de nível superior que se espera que o programa satisfaça, como isolamento – afirmar que certos pacotes nunca seriam encaminhados para certas portas, ou para garantir que algumas ações sejam tomadas em certos cabeçalhos.

Sintaticamente, cada assertiva é composta por uma expressão *booleana*, que pode incluir métodos primitivos. Os métodos permitidos são `forward`, `traverse_path`, `constant`, `if`, `extract_header` e `emit_header`. Tanto as expressões quanto os métodos podem operar sobre um ou mais valores, campos de cabeçalho ou cabeçalhos. Não há diferença de sintaxe entre elementos com restrição de localização e sem restrição de localização. Semanticamente, cada asserção representa um *booleano* que deve ser avaliado como verdadeiro ou falso, onde valores e campos de cabeçalho são avaliados como verdadeiro se forem diferentes de zero e falsos caso contrário. As expressões podem ser inteiras ou *booleanas* e, em ambos os casos, com a mesma semântica que suas contrapartes na linguagem P4.

Os métodos funcionam da seguinte forma: `if(`b_1`,` b_2`, [`b_3`])` é semelhante às declarações condicionais tradicionais: se a expressão b_1 for verdadeira, então a expressão

```
b ::= v                 m ::= forward()
    | f                     | traverse_path()
    | m                     | constant(f)
    | !b                    | if(b, b, [b])
    | b || b                | extract_header(h)
    | b && b                | emit_header(h)
    | b == b            i ::= v
    | b != b                | f
    | i >= i                | i * i
    | i <= i                | i / i
    | i < i                 | i % i
    | i > i                 | i + i
    | i == i                | i - i
    | i != i
```

Figura 4.13: Gramática da linguagem de asserção.

b_2 será avaliada, caso contrário a alternativa b_3 será avaliada. Este é o único método com restrição de localização, com todos os outros sendo irrestritos. `traverse_path()` indica se uma determinada estrutura no programa (por exemplo, uma ação) será percorrida antes que a execução do programa termine. `constant(f)` é verdadeiro se o campo `f` não mudar a partir do local de asserção, ou seja, até que o programa termine. `forward()` retorna *true* quando o pacote não for descartado no final do programa. `extract_header(h)` é verdadeiro se um cabeçalho `h` foi ou será extraído do pacote. Por fim, `emit_header(h)` retorna *true* se o pacote for transmitido com o cabeçalho `h`.

Os métodos apresentados na linguagem permitem a especificação de tipos de propriedades que seriam difíceis ou impossíveis de expressar usando apenas asserções tradicionais. A adição de `forward()` permite a expressão de propriedades de encaminhamento, que são essenciais para programas de plano de dados. `traverse_path()` permite raciocinar sobre o fluxo de controle do código fonte. `constant()` facilita a verificação da integridade das variáveis em todo o programa. Tanto `extract_header()` quanto `emit_header()` permitem a expressão das propriedades de formação de pacotes no nível do analisador e do analisador, respectivamente. Finalmente, `if()` auxilia o processo de combinação de métodos e expressões em uma expressão condicional.

A Figura 4.14 mostra um exemplo de um programa P4 anotado, com asserções em negrito. Por clareza, apenas as partes mais relevantes do programa são exibidas. Este programa descreve um *pipeline* de processamento de pacotes com uma única tabela (`dmac`), que é instanciada dentro do bloco de controle `TopPipe`. Cada entrada da tabela pode invocar uma das duas ações (`Drop()` ou `Set_dmac()`). As assertivas visam verificar se: (i) pacotes marcados para *drop* nunca são encaminhados (linha 7), e (ii) somente pacotes com TTL maior que zero são encaminhados (linha 21). As duas asserções contêm elementos sem restrição de localização (por exemplo, `forward()` captura o estado do programa no final de sua execução) e elementos com restrição de localização (por exemplo, a expressão `headers.ip.ttl >0` testa o valor de `headers.ip.ttl` no ponto em que a asserção é encontrada).


```
1  ...
2 control TopPipe(inout Parsed_packet headers,
3                   out OutControl outCtrl) {
4  ...
5  action Drop() {
6    outCtrl.outputPort = DROP_PORT;
7    @assert("if(traverse_path(), !forward())");
8 }
9  action Set_dmac(EthernetAddress dmac) {
10   headers.ethernet.dstAddr = dmac;
11 }
12 table dmac {
13   key = { nextHop : exact; }
14   actions = { Drop; Set_dmac; }
15   default_action = Drop;
16 }
17 apply {
18   ...
19   dmac.apply();
20   ...
21   @assert("if(forward(), headers.ip.ttl > 0)");
22 }
23}
```

Figura 4.14: Exemplo de um programa P4 anotado.

4.5.4. Estudo de Caso: Verificação de Programas P4

Primeiro, demonstra-se a eficácia da proposta para encontrar *bugs* e violações de políticas em planos de dados programáveis. Usa-se asserções para encontrar *bugs* em quatro aplicativos P4 recentes e confirma-se examinando manualmente os códigos-fonte.

Dapper [Ghasemi et al. 2017]: Dapper é uma ferramenta de diagnóstico de desempenho de plano de dados que infere gargalos TCP analisando pacotes em tempo real. Coloca-se um conjunto de asserções básicas no início do bloco de controle de ingresso, e através da asserção `if(ipv4.ttl == 0, !forward())`, descobre-se que o Dapper pode encaminhar pacotes IPv4 mesmo quando o campo TTL é zero. Por inspeção manual, nota-se que mesmo que o campo TTL seja decrementado conforme o esperado, seu valor nunca é verificado antes do encaminhamento. Por causa desse *bug*, ao depurar uma rede com um *loop*, os dispositivos baseados em Dapper podem continuar encaminhando pacotes para sempre. O protótipo encontrou esse *bug* em menos de um segundo.

NetPaxos [Dang et al. 2016]: NetPaxos é uma implementação baseada em rede do protocolo de consenso Paxos. Existem dois tipos diferentes de programas P4 nesta aplicação, um para Líderes/Coordenadores e outro para Aceitadores. Todos os outros atores são considerados totalmente implementados em *hosts* finais. Examina-se a versão atual da implementação e o conjunto de regras de encaminhamento disponibilizado pelos autores, acrescentando assertivas ao seu código. Como parte da implementação, um *Acceptor* vota adicionando informações de votação aos pacotes recebidos antes de encaminhá-los. Especificamente, a execução violou a asserção `if(traverse_path(), forward())`, localizada dentro da ação que realiza o voto. Isso indica que há pacotes

válidos (contendo informações de votação) sendo descartados. Ao inspecionar manualmente o código, descobre-se que o problema ocorre porque os pacotes são marcados primeiro para serem descartados por outra ação e não desmarcados pelas ações de votação. Esse *bug* pode ser corrigido marcando os pacotes a serem encaminhados dentro das ações que realizam a votação. De acordo com o *feedback* dos autores sobre esse *bug*, o código foi portado para $P4_{16}$, deixando a base de código antiga sem manutenção e exposta a *bugs*. O protótipo encontrou essa violação de afirmação em menos de um segundo.

DC.p4 [Sivaraman et al. 2015]: DC.p4 implementa o comportamento de um *switch* de *data center*. Ele contém várias funcionalidades, como encaminhamento L2/L3, ECMP, VLAN, espelhamento de pacotes, encapsulamento e várias ACLs (ou seja, L2, L3 ou com base em cabeçalhos mais específicos). Este programa contém mais de 2500 linhas de código distribuídas em 37 tabelas. As tabelas, por sua vez, são organizadas assumindo dois *pipelines* sequenciais de processamento de pacotes, um para pacotes de entrada/entrada e outro para pacotes de saída/saída, intercalados por um sistema de filas.

A configuração da tabela L3 ACL é verificada para descartar o tráfego com um endereço IP de destino específico para filtrar adequadamente esse tipo de pacote. Usa-se a asserção `if(ipv4.dstAddr == FILTER_ADDR, !forward())` para expressar que pacotes com endereços de destino IPv4 iguais a `FILTER_ADDR` devem ser descartados. Descobre-se que apenas configurar a L3 ACL não é suficiente para descartar pacotes IPv4, independentemente da política que está sendo aplicada. De fato, verifica-se que a L3 ACL apenas sinaliza pacotes para serem filtrados por outro módulo do sistema, que também deve ser configurado adequadamente. Embora este não seja um *bug* real, ainda é uma configuração incorreta no programa.

Switch [P4.org 2018]: Desde a introdução do documento DC.p4, sua base de código evoluiu para o programa Switch.p4, que é mantido ativamente. Usa-se abordagem de verificação para reproduzir dois *bugs* conhecidos, relatados em seu repositório. A primeira é a modificação de um campo de um cabeçalho inválido.[12] O *bug* é replicado testando com uma afirmação se o cabeçalho é válido antes de definir seu Campos. O segundo *bug* está relacionado ao encapsulamento de túnel[13], onde os cabeçalhos encapsulados são substituídos sempre que vários níveis aninhados estão presentes. Incluí-se uma declaração para testar se os cabeçalhos internos não são válidos antes de realizar o encapsulamento. A asserção falhou, confirmando que os cabeçalhos encapsulados podem ser substituídos e seu conteúdo original, descartado.

4.6. Imposição (*Enforcement*) de Políticas de Segurança

Uma alternativa à verificação é a imposição (enforcement). Em vez de verificar se uma configuração de rede está correta, um *kernel* de segurança logicamente separado evita ações que violem a política de segurança. O *kernel* de segurança deve mediar todas as ações de manipulação de pacotes no plano de dados. Ao contrário do modo de verificação, onde verifica-se as violações da política antes de uma configuração ser enviada para a rede, no modo de imposição, verifica-se as violações da política, uma vez que estão prestes a ocorrer. Tanto a verificação como a imposição têm suas vantagens e desvantagens.

[12] https://github.com/p4lang/switch/pull/102

[13] https://github.com/p4lang/switch/issues/97

Por um lado, a verificação capta problemas precocemente; um verificador pode fornecer informações de diagnóstico detalhadas sobre por que uma configuração viola uma política durante a fase de verificação. No regime de imposição, os problemas são detectados à medida que ocorrem. A imposição pode ser mais atrativa do que a verificação, porque não depende da complexidade do programa de controle ou do plano de dados.

Sendo assim, emergem benefícios para imposição (*enforcement* da política de segurança em plano de dados programáveis. Os planos permitem que os operadores de rede modifiquem o *pipeline* de processamento de pacotes dos dispositivos de rede para implementar novos protocolos, personalizar o comportamento da rede e estabelecer serviços de rede avançados. No que pese a sua simplicidade da programação, os programas P4 demonstraram ser propensos a uma variedade de *bugs* e erros de configuração [Stoenescu et al. 2016, Freire et al. 2018]. Como resultado, os operadores de rede precisam de estruturas para garantir que os programas que produzem tenham um comportamento correto para obter os benefícios de um ecossistema de *software* de plano de dados. Ferramentas de verificação de rede de última geração podem obter um modelo da rede, sua configuração e um conjunto de propriedades específicas usando formalismos tradicionais (por exemplo, lógica temporal ou regras de *Datalog*) e verificar automaticamente se essas propriedades são válidas para qualquer pacote [Beckett et al. 2017, Lopes et al. 2015].

Embora essas ferramentas ajudem os operadores de rede a identificar *bugs* antes que eles se manifestem, deve-se considerar: (i) Primeiro que a maioria dessas ferramentas exige que os programadores modelem manualmente os planos de dados programáveis, atividade complexa e propensa a erros [Lopes et al. 2015]; (ii) Em segundo lugar, essas ferramentas são geralmente restritas em termos de propriedades de acessibilidade para reduzir os tempos de verificação [Lopes et al. 2016]; (iii) Terceiro, ferramentas mais expressivas capazes de verificar múltiplas propriedades frequentemente enfrentam problemas graves de escalabilidade (por exemplo, verificar a conformidade com uma especificação de protocolo pode levar dias, mesmo para um único plano de dados programáveis; e (iv) Por fim, os programadores precisam ter habilidades técnicas formais de verificação para especificar corretamente suas propriedades.

Neves et al. [Neves et al. 2021] apresentam uma nova abordagem baseada na aplicação dinâmica (ou em tempo de execução) em vez de verificação estática. Essa abordagem tem várias vantagens práticas. Já que não é necessário esperar pelo resultado de um longo processo de verificação para enviar uma nova configuração para os *switches* de rede. Sendo assim, a aplicação do tempo de execução pode intervir prontamente se situações problemáticas realmente ocorrerem, possibilitando: obter informações úteis do código com *bugs* quando ele tem um comportamento correto e reparar problemas sem interferir em qualquer serviço de rede.

Em contraposição com a verificação estática, a aplicação do tempo de execução também permite ao programador expressar a política e o mecanismo usando o mesmo ambiente de programação que o resto do programa. Esse valor deve ser considerado, não só porque facilita a vida do programador, como evita também erros de tradução entre a implementação e as políticas. Sendo assim, para perceber os benefícios de uma aplicação dinâmica, Neves et al. [Neves et al. 2021] desenvolveram o P4box, um sistema para implantação de monitores de tempo de execução em planos de dados programáveis.

Usando P4box os programadores podem anexar monitores antes e depois dos blocos de controle, transições de estado do analisador e chamadas para funções externas de um programa P4. Cada monitor pode modificar a entrada e saída do bloco de código ou função que monitora, permitindo a verificação de pré e pós-condições a serem utilizadas para impor propriedades específicas ou modificar o comportamento do bloco monitorado.

Um monitor de tempo de execução insere-se na interação de um bloco de controle P4 ou analisador com o restante do ambiente de execução, como apresentado na Figura 4.15, permitindo que o programador do monitor modifique o comportamento do bloco P4 incluso com o restante do ambiente. Um bloco programável P4 faz a *interface* com o restante do ambiente de execução P4 na entrada no bloco, retornar do bloco as chamadas para funções externas fornecidas pela arquitetura. Na programação do modelo P4box, quando um bloco programável é invocado, o controle passa primeiro para um monitor, também escrito em P4, antes de passar para o bloco programável pretendido. Da mesma forma, quando um bloco programável completa o processamento, o controle passa primeiro para o monitor antes de retornar ao dispositivo, permitindo que um monitor modifique o comportamento de blocos programáveis de maneira bem definida.

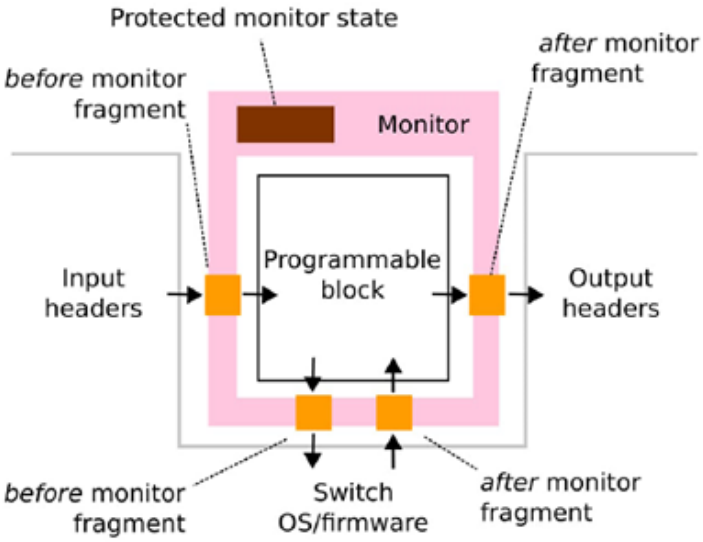


Figura 4.15: Modelo de programação P4box.

4.6.1. Monitores de Segurança para Switches P4

Um monitor de programa é uma construção de linguagem que foi desenvolvida (como uma extensão para P4) inspirada no paradigma *Aspect-Oriented Programming* (AOP) (Aspect-Oriented Programming) [Kiczales et al. 1997], o qual fornece construções em nível de linguagem para anexar código a pontos designados em um programa existente sem modificar o programa em si. Os programadores podem usar monitores para modificar ou verificar o comportamento de blocos de controle, analisadores e funções externas de programas P4 e, assim, garantir que eles respeitem um conjunto de propriedades desejadas. Os monitores são particularmente adequados ao contexto em que os programas de plano de dados são montados a partir dos módulos mantidos externamente, onde pode ser desejável alterar ou verificar o comportamento desses módulos sem modificar seu código.

Os monitores também podem interpor as chamadas para funções externas: quando um bloco programável invoca uma função externa, o controle passa primeiro para o monitor, depois para a função e depois volta para o monitor novamente, antes de retornar ao bloco programável. Um monitor pode, assim, modificar o comportamento aparente de uma função externa. Os monitores são declarados e definidos no nível superior de um programa P4, juntamente com os blocos de controle, blocos do analisador e outras declarações de nível superior. A Figura 4.16 apresenta a sintaxe para um monitor.

```
monitor <name> ( [param-list] ) on <object> {
    [local-declarations]
    (before | after) { <p4-statements> }
}
```

Figura 4.16: Sintaxe para um monitor.

Cada monitor é identificado por um único <nome> e pode receber parâmetros adicionais (<param-list>) contendo cabeçalhos e metadados, além dos parâmetros do objeto monitorado. Cada monitor deve estar associado a um plano de dados <object>, que pode ser um analisador, bloco de controle ou função externa. O tipo de recurso define o conjunto de elementos <p4-statements> que o monitor suporta. Os monitores podem ter dois tipos de métodos, a saber: antes e depois, que especificam o código fragmentos que são executados antes e depois do recurso monitorado, respectivamente. Finalmente, eles também podem conter declarações locais (por exemplo, ações, tabelas) visíveis dentro do monitor, mas não no bloco monitorado.

O P4box instrumenta um programa P4 com monitores em tempo de compilação de tal forma que o primeiro não pode contornar ou interferir neste último. O programa P4 original e os monitores de tempo de execução de definição de fonte P4 são fornecidos ao P4box, que combina o programa original com os monitores no nível intermediário para produzir um novo programa adequado para posterior compilação. No final, o código em nível de máquina contendo todos os monitores é gerado para uma variedade de destinos. Durante o processo de instrumentação, o P4box aproveita os recursos de linguagem fornecidos pelo P4, como escopos e *namespaces* separados, além da análise estática, para fornecer as seguintes garantias para cada monitor: 1) Mediação completa: o fluxo de execução do programa de plano de dados original sempre passará por um monitor (quando for definido pelo programador). Isso significa que não é possível para o programa original contornar um monitor; e 2) Não interferência: o programa original não pode interferir na operação de um monitor modificando suas variáveis locais ou cabeçalhos, o que significa que os monitores são completamente isolados do programa do plano de dados.

Juntas, as propriedades de mediação completa e não interferência permitem que os monitores restrinjam o que o programa P4 original permite realizar mesmo quando o último não é confiável (por exemplo, um programa de terceiros). Os monitores são, portanto, não apenas um mecanismo de estruturação de programa P4 orientado a aspectos, mas também uma caixa de proteção de *software* que pode ser usada para encapsular código P4 não confiável ou com erros. A Figura 4.17 mostra o fluxo de trabalho do P4box.

Os monitores podem ser combinados para impor propriedade mais complexas,

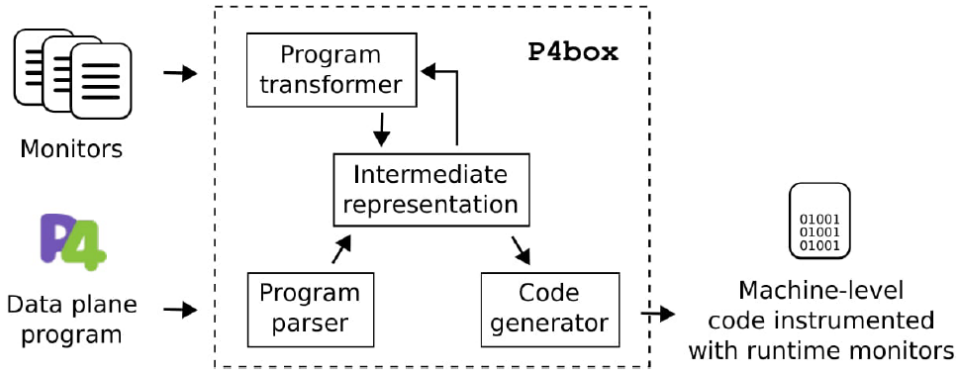


Figura 4.17: Fluxo de trabalho do P4box.

como as que envolvem extração e emissão de rótulos em pacotes. O P4box pode impor as propriedades de *(waypointing)* verificando e atualizando os rótulos sempre que esses pacotes cruzarem um dispositivo na cadeia. Como exemplo, a Figura 4.18 mostra um cenário em que os pacotes vindos de uma rede externa (ou seja, através de roteador R) deve primeiro ser inspecionado por um sistema IDS antes de chegar a um servidor web (*hosts* H1-H3). Neste caso, um monitor P4box em R introduz rótulos em cada pacote para impor o *(waypointing)*. Esses rótulos são então atualizados por outro monitor no *switch* S1 e um terceiro monitor os verifica no *switch* S2 para descartar pacotes destinados aos servidores web e não contém a *tag* atualizada (L1).

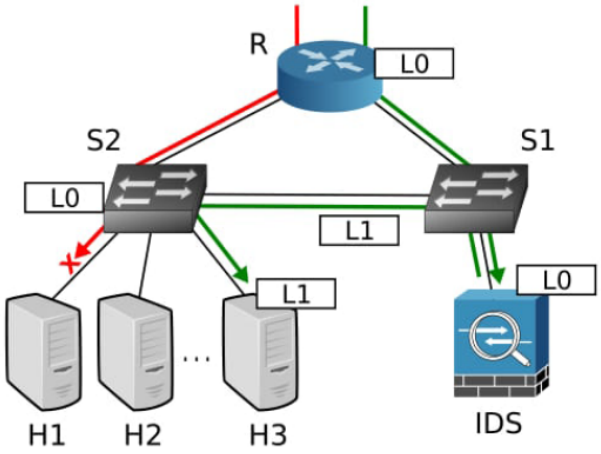


Figura 4.18: Exemplo de topologia para o *waypointing*.

A Figura 4.19 mostra como o P4box interage com o programa P4 para impor o *(waypointing)*, onde as setas verticais representam o fluxo de execução. Observe que o P4box prende o programa em três pontos: (i) primeiro: entre o análise dos cabeçalhos *Ethernet* e IPv4, para verificar se o pacote contém um rótulo e extrair o último; (ii) segundo: logo antes do início do pipeline de ação de correspondência, para operar no rótulo (por exemplo, verificar, atualizar ou remover) dependendo de como o dispositivo está conectado na topologia; e (iii) terceiro: para emitir o rótulo durante a fase de análise.

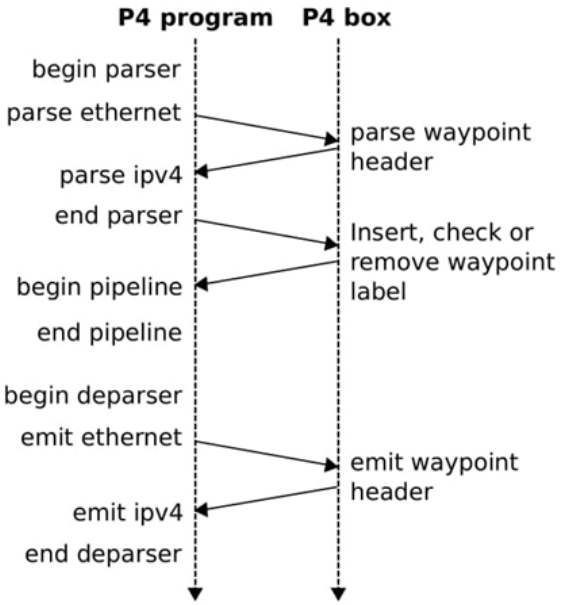


Figura 4.19: Interação entre o P4box e o programa P4 para aplicar o *waypointing*.

O P4Box suporta três tipos de monitores, a saber: 1) Monitores de bloco de controle: são responsáveis por garantir que um cabeçalho não seja modificado erroneamente pelo programa de plano de dados. O monitor é anexado ao *pipeline* de processamento e possui dois elementos: (i) antes do bloco programável, ele obtêm o estado do pacote original assim que é analisado (l.5-8); e (ii) após a bloco, ele testa se os cabeçalhos monitorados foram modificados (l.10-17). A Figura 4.20 mostra um exemplo de monitor de bloco de controle, que pode ser usado para detectar e processar informações sobrescrevendo *bugs*; 2) Monitor *parser*: podem ser anexados aos analisadores de nível superior. Como tal, antes e depois podem conter máquinas de estado finito e ambas devem ter um estado inicial e de aceitação. É possível especializar um monitor *parser* para um estado de *parser* específico, caso em que antes e depois são associados apenas a este último. Um exemplo de monitor analisador é mostrado na Figura 4.21 nas linhas 6 a 17, onde o monitor é anexado ao estado *ethernet* analisador e usado para extrair uma imposição cabeçalho. Esses monitores são particularmente úteis para pular a extração de *bits* de pacote que por algum motivo (por exemplo, confidencialidade) não devem ser visíveis para o programa de plano de dados; e 3) Monitores externos: são anexados as chamadas externas. Seus recursos são restritos ao que as ações podem fazer em P4 devido às limitações que o último tem em relação as chamadas externas (por exemplo, não é possível fazer declarações locais ou invocar uma tabela de dentro de uma ação). Semelhante aos monitores *parser*, os monitores externos também podem ser especializados em subgrupos de um recurso. Nesse caso, uma assinatura é usada para aplicar um monitor apenas a um subconjunto das chamadas externas. Um exemplo é apresentado na Figura 4.21 nas linhas 20 a 24, onde o monitor externo é aplicado apenas para chamadas para emissão de cabeçalhos do tipo *ethernet-t*. Os monitores externos são úteis para acompanhar como o programa de plano de dados interage com a plataforma subjacente.


```
1  monitor hdrInvMonitor() on Pipeline {
2    ipv4_t protec_ipv4;
3    udp_t  protec_udp;
4
5    before {
6      protec_ipv4 = hdr.inner_ipv4;
7      protec_udp = hdr.inner_udp;
8    }
9
10   after {
11     if( protec_ipv4 != hdr.inner_ipv4 ||
12         protec_udp != hdr.inner_udp ){
13       /*Run enforcement action
14         (e.g., restore original header
15         value, notify the control plane,
16         write log) */
17   }}
15 }
```

Figura 4.20: Exemplo para o monitor bloco de controle.

4.7. Tópicos Avançados

A maioria das funções de segurança de rede baseia-se em duas abordagens básicas de perfil de tráfego de plano de dados: inspeção de pacotes e análise de fluxo. A sua utilização no plano de dados tem o potencial de servir para vários fins, apoiando novos mecanismos de detecção de intrusão e serviços de verificação de políticas de segurança. Portanto, é crítico fornecer essas duas funções no plano de dados programável.

Existem vários desafios relacionados à gerência de segurança que serão brevemente cobertos usando o conceito de Programabilidade do Plano de Dados. A flexibilidade encontrada nas infraestruturas programáveis em conjunto com interfaces abertas e bem definidas permitem o desenvolvimento de ferramentas de segurança adequadas a planos de dados programáveis.

Esta seção inicial é organizada da seguinte forma. Inicialmente é apresentada a exploração de planos de dados programáveis para detectar ataques distribuídos de negação de serviço (*Distributed Denial of Service*- DDoS). Finalmente, depuração e rastreabilidade de aplicações em planos de dados programáveis é discutida.

4.7.1. Explorando Planos de Dados Programáveis para Detectar Ataques DDoS

Ataques de negação de serviço distribuído (DDoS) fazem uso dos limites de capacidade específicos aplicados a todos os recursos da rede. Esses ataques dependem de *botnet* para esgotar recursos computacionais e interromper aplicações na Internet [Hoque et al. 2015]. Buscam encaminhar um grande número de solicitações para o recurso tecnológico invadido, visando exceder a sua capacidade, interrompendo o seu funcionamento.

À medida que os *botnets* aumentam a sua aplicabilidade para explorar os dispositivos IoT (Internet das Coisas) vulneráveis, a frequência, a capacidade e o volume dos ataques DDoS amplia o seu alcance drasticamente. A detecção dessa ameaça é o pri-


```
1  struct p4boxState {
2    waypoint_t wp_header;
3  }
4
5  //Parser monitor to extract enforcement header
6  monitor wpParser(inout p4boxState pstate) on ParserImpl {
7    after parse_ethernet {
8      state start {
9        transition select(packet.lookahead<bit<32>>()){
10         16w0xFFFF : parse_wp_header;
11         default : accept;
12       }
13     }
14     state parse_wp_header {
15       packet.extract(pstate.wp_header);
16       transition accept;
17 }}}
18
19 //Extern monitor to emit enforcement header
20 monitor wpExtern(inout p4boxState pstate)
21                            on emit<ethernet_t>{
22   after {
23     packet.emit(pstate.wp_header);
24 }}
25
26 monitor wpControl(inout p4boxState pstate) on Pipeline {
27   ...
28   table check_waypoint {...}
29   ...
30
31   before {
32     //Enforce waypointing property
33     insert_label.apply();
34     check_waypoint.apply();
35     remove_label.apply();
36 }}
```

Figura 4.21: Exemplo para o monitor *parser* e monitor externo

meiro passo para minimizar as perdas por meio do desencadeamento das medidas defensivas, no entanto, representa um desafio para a pesquisa em rede [Antonakakis et al. 2017, Anstee et al. 2017, Zargar et al. 2013].

Preferencialmente, a detecção e o bloqueio de ataques DDoS devem ocorrer nas fontes para economizar esforços de deslocamento e processamento sobre o tráfego indesejado [Gil and Poletto 2001, Mirkovic et al. 2002, Peng et al. 2004]. No entanto, isso é impedido pela disseminação da atividade maliciosa, que é construída a partir da sincronização de solicitações aparentemente legítimas. Além disso, essas fontes normalmente pertencem a diferentes domínios administrativos, nos quais as políticas de segurança são definidas de forma independente. Mais adiante, nas proximidades da vítima, apesar do tráfego de ataque ser mais proeminente para detecção [Kim et al. 2006, Hoque et al. 2015], ele pode já ter saturado recursos *in-path*. A alternativa é implantar medidas defensivas em Provedores de Serviços de Internet (ISPs), que gerenciam a comunicação [Haq et al. 2015, Kang et al. 2016]. Os ISPs se beneficiam de uma visão privilegiada do tráfego e contam com *links* de alta taxa de transferência, permitindo que eles descubram e impeçam as ameaças em tempo hábil.

Ao contrário dos *datacenters*, onde o monitoramento de rede sofisticado pode ser realizado em *hosts* finais [Moshref et al. 2016, Yu et al. 2011], os ISPs dependem de *switch primitive* como amostragem de pacotes [CiscoNetworks 2017, Sflow 2017] e contagem baseada em fluxo [McKeown et al. 2008]. Os dados resultantes são então normalmente montados em servidores fora de banda para inspeção. Enquanto essas primitivas apresentam compensações entre granularidade de visibilidade, utilização de largura de banda, espaço de memória e a comunicação com servidores externos incorre em uma latência adicional para detectar eventos de rede [Moshref et al. 2013]. A fim de manter a utilização razoável da largura de banda e a carga de processamento, a amostragem de pacotes é geralmente empregada em taxas agressivamente baixas [Phaal 2009], apenas transmitindo informações de um conjunto limitado de pacotes. Diferentemente, a contagem baseada em fluxo, como em *switches* OF [McKeown et al. 2008], fornece valores exatos para métricas volumétricas com um alto custo de entradas nas tabelas.

Como alternativa promissora para este problema, o conceito emergente de programabilidade do plano de dados oferece flexibilidade para a execução de algoritmos nos *switches* de rede [Bosshart et al. 2014b]. Assumindo um fluxo de pacotes como entrada, esses algoritmos são modelados como um *pipeline* de primitivas elementares, acessos à memória e pesquisas em tabelas. Sendo assim, os operadores podem definir funções de monitoramento e delegá-las a dispositivos de plano de dados em toda rede. Essa arquitetura pode ser explorada para realizar a inspeção em cada pacote sem incorrer em sobrecarga de comunicação. No entanto, buscando executar a taxa de linha com custos razoáveis, o processamento de pacotes é restrito a um pequeno orçamento de tempo e uma quantidade limitada de memória por estágio de *pipeline* [Bosshart et al. 2013].

Lapolli et. al [Lapolli et al. 2019] desenvolveram uma arquitetura de sistema para detecção de DDoS, na qual o plano de dados responsável pela coleta do fluxo das métricas e sua inspeção. Isso é apresentado na forma de uma detecção de ataque DDoS em banda sistema totalmente implementável em uma chave programável através de P4. O trabalho compreende um *pipeline* de processamento para estimar as entropias dos endereços IP de origem e destino. Esses valores são usados para caracterizar o tráfego supostamente legítimo em tempo real. Os resultados desta caracterização servem para calcular a detecção limiares considerando um coeficiente de sensibilidade parametrizável. A fim de respeitar o rigoroso orçamento de tempo e restrições de memória para o cálculo da entropia, a frequência de endereços IP distintos é aproximada por esboços de contagem aprimorados [Charikar et al. 2002]. Outras funções aritméticas de computação intensiva são resolvidas com a ajuda de uma tabela de pesquisa *Longest Match Routing Rule* (LPM) otimizada para memória.

O sistema de detecção de DDoS é composto por um único *switch* programável que executa de forma independente a análise estatística em banda para detectar ataques DDoS. Como esses ataques são caracterizados por uma grande quantidade de *hosts* convergindo o tráfego para uma ou poucas vítimas [Haq et al. 2015], as distribuições de endereços IP de origem e destino tendem a se desviar de seu padrão normal na presença de tráfego malicioso. A entropia de Shannon [Shannon 1948] é frequentemente utilizada como forma de identificar esse desvio, apresentando alta precisão para este objetivo [Lakhina et al. 2005, Bhuyan et al. 2015]. Com isso, a abordagem sugerida busca estimar as entropias dos endereços IP e caracterizar os endereços supostamente legítimos o

tráfego para definir limites de detecção.

A Figura 4.22 apresenta uma visão da arquitetura do sistema. Para representar os endereços de tráfego recentes de distribuição, os blocos de estimativa de entropia operam em partições consecutivas do fluxo de pacotes de entrada. Essas partições, denominadas janelas de observação, contêm um número predeterminado de pacotes de interesse de reduzir os requisitos de processamento. A estimativa é construída como um algoritmo de *streaming* a ser implementado em um *pipeline* de processamento. Ao final de cada janela de observação, os blocos de caracterização do tráfego obtêm os valores de entropia para atualizar o modelo de tráfego legítimo. Por sua vez, o bloco de detecção de anomalias calcula os limites de detecção em função deste modelo e emite um alarme de ataque quando são superados pelas últimas estimativas de entropia. Esse alarme é realimentado para a detecção da anomalia nos blocos de caracterização de tráfego para que seus modelos considerem apenas tráfego supostamente legítimo.

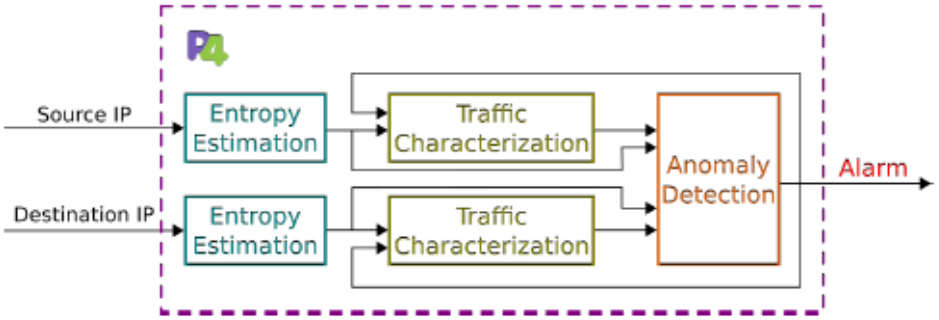


Figura 4.22: Arquitetura do sistema de detecção de ataques DDoS.

4.7.2. Depuração e Rastreabilidade de Aplicações em Planos de Dados Programáveis

Planos de dados programáveis permitem que a execução de aplicações cruzem a fronteira entre servidores x86 tradicionais e a rede de computadores, habilitando o descarregamento (ou seja, *offloading*) de partes da computação para PDPs. Esse paradigma tem sido chamado de *in-network computing* [Benson 2019]. À luz desse desenvolvimento, tanto a indústria quanto pesquisadores começaram a investigar ativamente novos projetos para aplicações distribuídas a fim de melhorar o desempenho, a escalabilidade ou a confiabilidade dessas, transferindo parte de sua funcionalidade para a rede. Dessa forma, uma vasta gama de problemas tem explorado essa possibilidade de descarregar parte da computação para a rede: *Caching*: NetCache [Jin et al. 2017] armazena em cache pares de chave-valor em switches, evitando potencialmente longos RTTs para acessar um servidor de armazenamento de chave-valor remoto; *Agregação de Dados*: DAIET [Sapio et al. 2017] realiza agregação de dados na rede para maior escalabilidade; *Machine Learning*: machine learning dentro de switches pode mitigar gargalos existentes durante o treinamento distribuído de modelos [Sanvito et al. 2018, Xiong and Zilberman 2019]; *Pattern Matching*: a correspondência de padrões eficiente pode ser alcançada através da realização de parte da computação na rede [Jepsen et al. 2019].

À medida que essas abordagens recém-descobertas se aproximam da implanta-

ção, surgem preocupações práticas sobre seu gerenciamento em tempo de execução, porque as aplicações distribuídas agora podem executar parcialmente no plano de dados. Especificamente, a incorporação de lógica em PDPs adicionou outra camada de complexidade para rastrear e solucionar problemas dessas aplicações, e esforços tradicionais de rastreabilidade e observabilidade de aplicações em servidores x86 tradicionais não se traduzem diretamente para *in-network computing* [Benson 2019]. Em particular, switches programáveis atuais não fornecem uma abstração rica o suficiente para suportar técnicas de rastreamento tradicionais [Sigelman et al. 2010, Mace and Fonseca 2018, Chow et al. 2014], e essa falta de primitivas de rastreamento força os programadores a criarem suas próprias soluções exclusivas. Isso leva à criação de ferramentas de rastreamento muito específicas e não reutilizáveis para depurar a computação na rede. Mais importante, rastros produzidos por soluções específicas para o PDP provavelmente não serão interoperáveis com estruturas de diagnóstico de rastreamento existentes, por exemplo, Dapper[Sigelman et al. 2010] do Google. Ortogonalmente, as estruturas de rastreamento existentes não fornecem primitivas para gerar ou capturar dados de rastreamento em planos de dados programáveis.

Um desafio de pesquisa atual visa preencher a lacuna entre técnicas tradicionais para telemetria de redes e *frameworks* de rastreamento distribuído. Isso requer abordar execuções que cruzem a fronteira da aplicação distribuída para o plano de dados programável, capturando dados de rastreamento de PDPs e apresentando-os ao plano de aplicação por meio de uma abstração flexível e bem definida. Um dos primeiros esforços nessa direção é o P4-Intel [Castanheira et al. 2019], que (i) aproveita a telemetria de rede para instrumentar PDPs no monitoramento de dados de rastreamento arbitrários definidos pelo usuário e (ii) coordena o armazenamento, coleta e formatação desses dados de rastreamento internamente, fornecendo apenas dados de contexto bem formados para qualquer ferramenta de depuração do plano de aplicação.

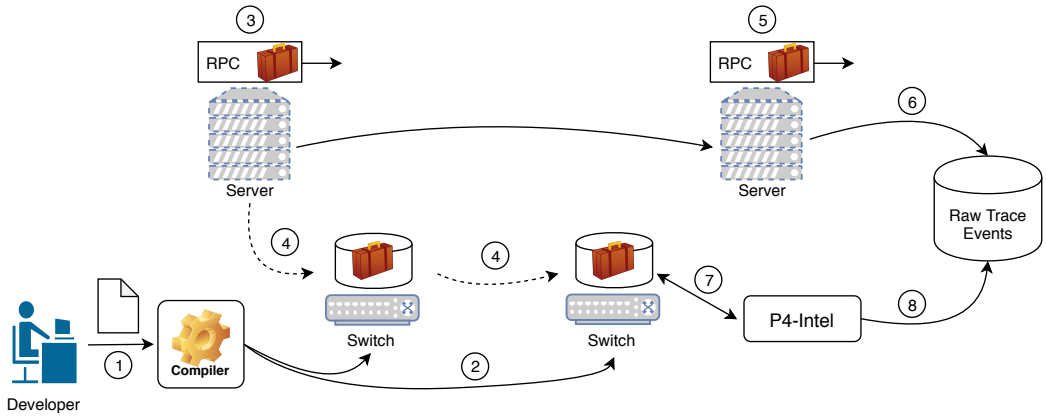


Figura 4.23: *Workflow* de Diagnóstico do P4-Intel. Malas vermelhas representam os contextos (*bagagem*), ou seja, os *logs* de rastreamento de RPCs. Os servidores armazenam o contexto dentro dos RPCs e propagam o contexto *in-band* dentro das mensagens RPC. Os *switches* armazenam os contextos localmente e os propagam *out-of-band*, em mensagens separadas, para o armazenamento externo.

A Figura 4.23 apresenta o *workflow* de alto nível do P4-Intel. Nele, de forma offline, desenvolvedores de aplicações (etapa 1) irão fazer anotações em programas de

in-network computing (escritos na linguagem P4) com o conjunto de variáveis definidas pelo usuário para exportar e implantar esses programas na rede (etapa 2). Essas variáveis que compõem o conjunto de dados que podem ser capturados e exportados usando uma abstração de *bagagem*. Em tempo de execução, o *framework* de rastreamento incluirá tags (ou cabeçalhos) em pacotes contendo chamadas RPC (etapa 3) e os propagará para outros servidores (etapa 5) via RPCs e, finalmente, os armazenará em um banco de dados externo (etapa 6) assim que o rastreamento de RPC for concluído. Além disso, à medida que esses rastreamentos de RPCs se propagam pela rede, o *framework* do plano de dados captura os dados apropriados e os armazena localmente no switch (etapa 4) ou, se necessário, os anexa aos pacotes. Periodicamente, o P4-Intel irá interagir com o plano de dados para exportar esses dados para uma entidade centralizada (passos 7-8) que combinará os dados capturados no plano de dados com os dados coletados nos RPCs.

Sistemas como o P4-Intel não apenas simplificam o rastreamento de programas PDP, mas também, dada a interoperabilidade com *frameworks* de sistemas distribuídos emergentes, também tem o potencial de simplificar o gerenciamento e facilitar o processo de depuração feito pelos programadores.

4.8. Considerações Finais

Os avanços em SDN expandiram a capacidade de programar a rede em direção ao plano de dados, o que possibilita a separação entre o controle da rede e as funções de encaminhamento. Tal separação está mudando radicalmente o cenário de rede, ajudando a enfrentar a ossificação de rede. Neste contexto, o operador de rede pode determinar a lógica para processamento de pacotes e assim habilitar arquiteturas dinâmicas, e gerenciáveis. No entanto, acentua-se a dificuldade e serem garantidas as propriedades de segurança e de corretude em toda rede, considerando uma combinação da configuração mantida pelo plano de controle e os programas do plano de dados.

O presente capítulo discute abordagens de segurança para redes de computadores na era dos planos de dados programáveis. Inicialmente, foi apresentada uma revisão dos principais fundamentos de programabilidade do plano de dados, especialmente no que tange à P4. Em seguida, é descrito como o conceito de programabilidade do plano de dados pode ser usado para enfrentar desafios de segurança em redes de computadores. Por fim, tópicos avançados e propostas já desenvolvidas neste contexto, são apresentados.

Apesar da discussão apresentada no capítulo, novos tópicos podem ser discutidos em trabalhos futuros. A relação entre a segurança através da programabilidade do plano de dados e a restrição de recursos de rede e/ou dispositivos pode ser explorada. A ampliação de funcionalidade de segurança em tal plano pode contribuir para assegurar propriedades de segurança em ambientes desafiadores. Finalmente, a relação de aspectos de segurança em redes 5G/6G e a programabilidade do plano de dados também pode ser abordada. A evolução das redes de comunicação móvel apresenta diversos desafios de segurança os quais poderiam ser enfrentados com dinamicidade no plano de dados.

Referências

[Anstee et al. 2017] Anstee, D., Bussiere, D., Sockrider, G., and Morales, C. (2017). Worldwide infrastructure security report. *Arbor Networks Inc., Westford, MA, USA*.

[Antonakakis et al. 2017] Antonakakis, M., April, T., Bailey, M., Bernhard, M., Bursztein, E., Cochran, J., Durumeric, Z., Halderman, J. A., Invernizzi, L., Kallitsis, M., et al. (2017). Understanding the mirai botnet. In *26th USENIX security symposium (USENIX Security 17)*, pages 1093–1110.

[Avizienis et al. 2004] Avizienis, A., Laprie, J.-C., Randell, B., and Landwehr, C. (2004). Basic concepts and taxonomy of dependable and secure computing. *IEEE transactions on dependable and secure computing*, 1(1):11–33.

[Beckett et al. 2017] Beckett, R., Gupta, A., Mahajan, R., and Walker, D. (2017). A general approach to network configuration verification. In *Proceedings of the Conference of the ACM Special Interest Group on Data Communication*, pages 155–168.

[Benson 2019] Benson, T. A. (2019). In-network compute: Considered armed and dangerous. In *Proceedings of the Workshop on Hot Topics in Operating Systems*, HotOS '19, pages 216–224, New York, NY, USA. ACM.

[Bhuyan et al. 2015] Bhuyan, M. H., Bhattacharyya, D., and Kalita, J. K. (2015). An empirical evaluation of information metrics for low-rate and high-rate ddos attack detection. *Pattern Recognition Letters*, 51:1–7.

[Birnfeld et al. 2020] Birnfeld, K., da Silva, D. C., Cordeiro, W., and de França, B. B. N. (2020). P4 switch code data flow analysis: Towards stronger verification of forwarding plane software. In *NOMS 2020 - 2020 IEEE/IFIP Network Operations and Management Symposium*, pages 1–8.

[Bosshart et al. 2014a] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., et al. (2014a). P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, 44(3):87–95.

[Bosshart et al. 2014b] Bosshart, P., Daly, D., Gibb, G., Izzard, M., McKeown, N., Rexford, J., Schlesinger, C., Talayco, D., Vahdat, A., Varghese, G., et al. (2014b). P4: Programming protocol-independent packet processors. 44 (3): 87–95, july 2014.

[Bosshart et al. 2013] Bosshart, P., Gibb, G., Kim, H.-S., Varghese, G., McKeown, N., Izzard, M., Mujica, F., and Horowitz, M. (2013). Forwarding metamorphosis: Fast programmable match-action processing in hardware for sdn. *ACM SIGCOMM Computer Communication Review*, 43(4):99–110.

[Boutaba and Aib 2007] Boutaba, R. and Aib, I. (2007). Policy-based management: A historical perspective. *Journal of Network and Systems Management*, 15(4):447–480.

[Brum 2022] Brum, H. B. (2022). Um método para coleta dinâmica e eficiente de estatísticas em redes programáveis.

[Carey 2017] Carey, S. (2017). Why a single failed router can ground a thousand flights. *The Wall Street Journal*.

[Castanheira et al. 2019] Castanheira, L., Schaeffer-Filho, A., and Benson, T. A. (2019). P4-intel: Bridging the gap between icf diagnosis and functionality. In *Proceedings of the 1st ACM CoNEXT Workshop on Emerging In-Network Computing Paradigms*, ENCP '19, page 21–26, New York, NY, USA. Association for Computing Machinery.

[Charikar et al. 2002] Charikar, M., Chen, K., and Farach-Colton, M. (2002). Finding frequent items in data streams. In *International Colloquium on Automata, Languages, and Programming*, pages 693–703. Springer.

[Chow et al. 2014] Chow, M., Meisner, D., Flinn, J., Peek, D., and Wenisch, T. F. (2014). The mystery machine: End-to-end performance analysis of large-scale internet services. In *11th USENIX Symposium on Operating Systems Design and Implementation (OSDI 14)*, pages 217–231, Broomfield, CO. USENIX Association.

[CiscoNetworks 2017] CiscoNetworks (2017). Cisco ios netflow. In *Accessed on June, 29 2017*.

[Craven et al. 2011] Craven, R., Lobo, J., Lupu, E., Russo, A., and Sloman, M. (2011). Policy refinement: Decomposition and operationalization for dynamic domains. In *2011 7th International Conference on Network and Service Management*, pages 1–9. IEEE.

[da Silva et al. 2015] da Silva, A. S., Smith, P., Mauthe, A., and Schaeffer-Filho, A. (2015). Resilience support in software-defined networking: A survey. *Computer Networks*, 92:189–207.

[Dang et al. 2016] Dang, H. T., Canini, M., Pedone, F., and Soulé, R. (2016). Paxos made switch-y. *SIGCOMM Comput. Commun. Rev.*, 46(2):18–24.

[Dobrescu and Argyraki 2014] Dobrescu, M. and Argyraki, K. (2014). Software dataplane verification. In *11th USENIX Symposium on Networked Systems Design and Implementation (NSDI 14)*, pages 101–114, Seattle, WA. USENIX Association.

[Feferman et al. 2018] Feferman, D. L., Mejia, J. S., Saraiva, N., and Rothenberg, C. E. (2018). Uma nova revolucao em redes: Programacao do plano de dados com p4. *Escola Regional de Informátiva do Piauí (ERIPI), Teresina, Brazil*.

[Fernandes and Rothenberg 2014] Fernandes, E. L. and Rothenberg, C. E. (2014). Openflow 1.3 software switch. *Salao de Ferramentas do XXXII Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuıdos SBRC*, pages 1021–1028.

[Fosdick and Osterweil 2011] Fosdick, L. D. and Osterweil, L. J. (2011). Data flow analysis in software reliability. In *Engineering of Software*, pages 49–85. Springer.

[Freire et al. 2018] Freire, L., Neves, M., Leal, L., Levchenko, K., Schaeffer-Filho, A., and Barcellos, M. (2018). Uncovering bugs in p4 programs with assertion-based verification. In *SOSR*, page 4. ACM.

[Garcia et al. 2018] Garcia, L. F. U., Villaça, R. S., Ribeiro, M. R., Martins, R. F. T., Verdi, F. L., and Marcondes, C. (2018). Introdução à linguagem p4-teoria e prática. *Simpósio Brasileiro de Redes de Computadores e Sistemas Distribuídos (SBRC)-Minicursos*.

[Ghasemi et al. 2017] Ghasemi, M., Benson, T., and Rexford, J. (2017). Dapper: Data plane performance diagnosis of tcp. In *Proceedings of the Symposium on SDN Research*, SOSR '17, pages 61–74, New York, NY, USA. ACM.

[Gil and Poletto 2001] Gil, T. M. and Poletto, M. (2001). {MULTOPS}: A {Data-Structure} for bandwidth attack detection. In *10th USENIX Security Symposium (USENIX Security 01)*.

[Hamed and Al-Shaer 2006] Hamed, H. and Al-Shaer, E. (2006). Taxonomy of conflicts in network security policies. *IEEE Communications Magazine*, 44(3):134–141.

[Haq et al. 2015] Haq, O., Abaid, Z., Bhatti, N., Ahmed, Z., and Syed, A. (2015). Sdn-inspired, real-time botnet detection and flow-blocking at isp and enterprise-level. In *2015 IEEE International Conference on Communications (ICC)*, pages 5278–5283. IEEE.

[Hecht 1977] Hecht, M. S. (1977). *Flow Analysis of Computer Programs*. Elsevier Science Inc.

[Hoque et al. 2015] Hoque, N., Bhattacharyya, D. K., and Kalita, J. K. (2015). Botnet in ddos attacks: trends and challenges. *IEEE Communications Surveys & Tutorials*, 17(4):2242–2270.

[Horgan and London 1991] Horgan, J. R. and London, S. (1991). Data flow coverage and the c language. In *Proceedings of the symposium on Testing, analysis, and verification*, pages 87–97.

[Jepsen et al. 2019] Jepsen, T., Alvarez, D., Foster, N., Kim, C., Lee, J., Moshref, M., and Soulé, R. (2019). Fast string searching on pisa. In *Proceedings of the 2019 ACM Symposium on SDN Research*, SOSR '19, page 21â28, New York, NY, USA. Association for Computing Machinery.

[Jin et al. 2017] Jin, X., Li, X., Zhang, H., Soulé, R., Lee, J., Foster, N., Kim, C., and Stoica, I. (2017). Netcache: Balancing key-value stores with fast in-network caching. SOSP '17.

[Kang et al. 2016] Kang, M. S., Gligor, V. D., Sekar, V., et al. (2016). Spiffy: Inducing cost-detectability tradeoffs for persistent link-flooding attacks. In *NDSS*, volume 1, pages 53–55.

[Kazemian 2017] Kazemian, P. (2017). Network path not found? *Forward Networks Blog*.

[Khurshid et al. 2013] Khurshid, A., Zou, X., Zhou, W., Caesar, M., and Godfrey, P. B. (2013). {VeriFlow}: Verifying {Network-Wide} invariants in real time. In *10th USENIX Symposium on Networked Systems Design and Implementation (NSDI 13)*, pages 15–27.

[Kiczales et al. 1997] Kiczales, G., Lamping, J., Mendhekar, A., Maeda, C., Lopes, C., Loingtier, J.-M., and Irwin, J. (1997). Aspect-oriented programming. In *European conference on object-oriented programming*, pages 220–242. Springer.

[Kim et al. 2006] Kim, Y., Lau, W. C., Chuah, M. C., and Chao, H. J. (2006). Packetscore: a statistics-based packet filtering scheme against distributed denial-of-service attacks. *IEEE transactions on dependable and secure computing*, 3(2):141–155.

[Kreutz et al. 2013] Kreutz, D., Ramos, F. M., and Verissimo, P. (2013). Towards secure and dependable software-defined networks. In *Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking*, HotSDN '13, pages 55–60, New York, NY, USA. ACM.

[Kreutz et al. 2014] Kreutz, D., Ramos, F. M., Verissimo, P. E., Rothenberg, C. E., Azodolmolky, S., and Uhlig, S. (2014). Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, 103(1):14–76.

[Lakhina et al. 2005] Lakhina, A., Crovella, M., and Diot, C. (2005). Mining anomalies using traffic feature distributions. *ACM SIGCOMM computer communication review*, 35(4):217–228.

[Lapolli et al. 2019] Lapolli, C., Adilson Marques, J., and Gaspary, L. P. (2019). Offloading real-time ddos attack detection to programmable data planes. In *2019 IFIP/IEEE Symposium on Integrated Network and Service Management (IM)*, pages 19–27.

[Liu et al. 2018] Liu, J., Hallahan, W., Schlesinger, C., Sharif, M., Lee, J., Soulé, R., Wang, H., Caşcaval, C., McKeown, N., and Foster, N. (2018). P4v: Practical verification for programmable data planes. In *ACM SIGCOMM 2018*, pages 490–503, New York, NY, USA. ACM.

[Lopes et al. 2016] Lopes, N., Bjørner, N., McKeown, N., Rybalchenko, A., Talayco, D., and Varghese, G. (2016). Automatically verifying reachability and well-formedness in p4 networks. *Technical Report, Tech. Rep*.

[Lopes et al. 2015] Lopes, N. P., Bjørner, N., Godefroid, P., Jayaraman, K., and Varghese, G. (2015). Checking beliefs in dynamic networks. In *12th USENIX Symposium on Networked Systems Design and Implementation (NSDI 15)*, pages 499–512, Oakland, CA. USENIX Association.

[Lupu and Sloman 1999] Lupu, E. C. and Sloman, M. (1999). Conflicts in policy-based distributed systems management. *IEEE Trans. Softw. Eng.*, 25(6):852–869.

[Mace and Fonseca 2018] Mace, J. and Fonseca, R. (2018). Universal context propagation for distributed system instrumentation. In *Proceedings of the Thirteenth EuroSys Conference*, EuroSys '18, pages 8:1–8:18, New York, NY, USA. ACM.

[McKeown et al. 2008] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. (2008). Openflow: enabling innovation in campus networks. *ACM SIGCOMM computer communication review*, 38(2):69–74.

[Mirkovic et al. 2002] Mirkovic, J., Prier, G., and Reiher, P. (2002). Attacking ddos at the source. In *10th IEEE International Conference on Network Protocols, 2002. Proceedings.*, pages 312–321. IEEE.

[Moshref et al. 2013] Moshref, M., Yu, M., and Govindan, R. (2013). Resource/accuracy tradeoffs in software-defined measurement. In *Proceedings of the second ACM SIGCOMM workshop on Hot topics in software defined networking*, pages 73–78.

[Moshref et al. 2016] Moshref, M., Yu, M., Govindan, R., and Vahdat, A. (2016). Trumpet: Timely and precise triggers in data centers. In *Proceedings of the 2016 ACM SIGCOMM Conference*, pages 129–143.

[Neves et al. 2018] Neves, M., Freire, L., Schaeffer-Filho, A., and Barcellos, M. (2018). Verification of p4 programs in feasible time using assertions. In *Proceedings of the 14th International Conference on Emerging Networking EXperiments and Technologies*, CoNEXT '18, page 73–85, New York, NY, USA. Association for Computing Machinery.

[Neves et al. 2021] Neves, M., Huffaker, B., Levchenko, K., and Barcellos, M. (2021). Dynamic property enforcement in programmable data planes. *IEEE/ACM Transactions on Networking*, 29(4):1540–1552.

[P4.org 2018] P4.org (2018). Switch. https://github.com/p4lang/switch.

[Panda et al. 2017] Panda, A., Lahav, O., Argyraki, K., Sagiv, M., and Shenker, S. (2017). Verifying reachability in networks with mutable datapaths. In *14th USENIX Symposium on Networked Systems Design and Implementation (NSDI 17)*, pages 699–718, Boston, MA. USENIX Association.

[Peng et al. 2004] Peng, T., Leckie, C., and Ramamohanarao, K. (2004). Proactively detecting distributed denial of service attacks using source ip address monitoring. In *International conference on research in networking*, pages 771–782. Springer.

[Phaal 2009] Phaal, P. (2009). sflow: Sampling rates. In *June 2009*.

[Rapps and Weyuker 1985] Rapps, S. and Weyuker, E. J. (1985). Selecting software test data using data flow information. *IEEE Trans. Softw. Eng.*, 11(4):367–375.

[Sanvito et al. 2018] Sanvito, D., Siracusano, G., and Bifulco, R. (2018). Can the network be the ai accelerator? In *Proceedings of the 2018 Morning Workshop on In-Network Computing*, NetCompute '18, page 20â25, New York, NY, USA. Association for Computing Machinery.

[Sapio et al. 2017] Sapio, A., Abdelaziz, I., Aldilaijan, A., Canini, M., and Kalnis, P. (2017). In-network computation is a dumb idea whose time has come. In *Proceedings of the 16th ACM Workshop on Hot Topics in Networks*, HotNets-XVI, pages 150–156, New York, NY, USA. ACM.

[Sflow 2017] Sflow (2017). sflow.org - making the network visible. In *Accessed on June, 29 2017*.

[Shannon 1948] Shannon, C. E. (1948). A mathematical theory of communication. *The Bell system technical journal*, 27(3):379–423.

[Sigelman et al. 2010] Sigelman, B. H., Barroso, L. A., Burrows, M., Stephenson, P., Plakal, M., Beaver, D., Jaspan, S., and Shanbhag, C. (2010). Dapper, a large-scale distributed systems tracing infrastructure. Technical report, Google, Inc.

[Sivaraman et al. 2015] Sivaraman, A., Kim, C., Krishnamoorthy, R., Dixit, A., and Budiu, M. (2015). Dc.p4: Programming the forwarding plane of a data-center switch. In *Proceedings of the 1st ACM SIGCOMM Symposium on Software Defined Networking Research*, SOSR '15, pages 2:1–2:8, New York, NY, USA. ACM.

[Son et al. 2013] Son, S., Shin, S., Yegneswaran, V., Porras, P., and Gu, G. (2013). Model checking invariant security properties in openflow. In *2013 IEEE International Conference on Communications (ICC)*, pages 1974–1979. IEEE.

[Stoenescu et al. 2016] Stoenescu, R., Popovici, M., Negreanu, L., and Raiciu, C. (2016). Symnet: Scalable symbolic execution for modern networks. In *ACM SIGCOMM 2016*, pages 314–327. ACM.

[Travassos et al. 1999] Travassos, G., Shull, F., Fredericks, M., and Basili, V. R. (1999). Detecting defects in object-oriented designs: Using reading techniques to increase software quality. *SIGPLAN Not.*, 34(10):47–56.

[Udupi et al. 2007] Udupi, Y. B., Sahai, A., and Singhal, S. (2007). A classification-based approach to policy refinement. In *2007 10th IFIP/IEEE International Symposium on Integrated Network Management*, pages 785–788. IEEE.

[Vergilio et al. 1997] Vergilio, S. R., Maldonado, J. C., and Jino, M. (1997). Constraint based selection of test sets to satisfy structural software testing criteria. In *Proceedings 17th International Conference of the Chilean Computer Science Society*, pages 256–263.

[Verma 2002] Verma, D. C. (2002). Simplifying network administration using policy-based management. *IEEE network*, 16(2):20–26.

[Westerinen et al. 2001] Westerinen, A., Schnizlein, J., Strassner, J., Scherling, M., Quinn, B., Herzog, S., Huynh, A., Carlson, M., Perry, J., and Waldbusser, S. (2001). Terminology for policy-based management. Technical report.

[Winder 2020] Winder, D. (2020). Much of the internet went down yesterday: Here's the reason why. *Forbes*.

[Xiong and Zilberman 2019] Xiong, Z. and Zilberman, N. (2019). Do switches dream of machine learning? toward in-network classification. In *Proceedings of the 18th ACM Workshop on Hot Topics in Networks*, HotNets '19, page 25â33, New York, NY, USA. Association for Computing Machinery.

[Yu et al. 2011] Yu, M., Greenberg, A., Maltz, D., Rexford, J., Yuan, L., Kandula, S., and Kim, C. (2011). Profiling network performance for multi-tier data center applications. In *8th USENIX Symposium on Networked Systems Design and Implementation (NSDI 11)*.

[Zargar et al. 2013] Zargar, S. T., Joshi, J., and Tipper, D. (2013). A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks. *IEEE communications surveys & tutorials*, 15(4):2046–2069.

Sobre os organizadores

Carlos Raniery P. dos Santos é professor adjunto na UFSM. Possui doutorado em Ciência da Computação pela UFRGS (2013). Foi pesquisador visitante no centro de pesquisas IBM T.J. Watson - Hawthorne (2010-2011), onde participou em projetos de Gerência de Serviços de TI e Gerência de Segurança. Atuou como secretário do IEEE Technical Committee on Network Operations and Management (2017-2023).

Walter Priesnitz Filho é doutor em Segurança de Informação pela Universidade de Lisboa (UL) - Instituto Superior Técnico (IST) em Portugal, com período sanduíche no Institute of Applied Information Processing and Communications (IAIK) da Graz University of Technology (Áustria) - 2018. Mestre em Ciência da Computação pela UFSC - 2003. Atualmente é Professor Adjunto da Universidade Federal de Santa Maria - UFSM/CTISM.

Paulo André da S. Gonçalves possui doutorado pela Université Pierre et Marie Curie (Sorbonne Université) (2004). Atualmente, é professor da UFPE. Foi conselheiro da RNP (2019-2021). Foi membro da diretoria do Laboratório Nacional de Computação (LARC) (2012-2021), sendo Diretor do Conselho Técnico-Científico de 2017 a 2019. Coordenou projeto de cooperação internacional entre Brasil e França para a formação conjunta de Engenheiros (2018-2021).

Marcia Henke é Professora Adjunta da Universidade Federal de Santa Maria (UFSM). Possui doutorado em Informática com ênfase em Ciência da Computação, mestrado em Ciência do Gerenciamento com foco em Sistemas de Informação, especialização em Redes e Sistemas Distribuídos e graduação em Ciência da Computação. Participa de quatro projetos, atualmente.

Patrocinadores

nic.br cgi.br

CAPES

CNPq

XLabs security

TEMPEST

AVATO

ripple

FACULDADE IBPTECH