CENTRO ESTADUAL DE EDUCAÇÃO TECNOLÓGICA PAULA SOUZA
UNIDADE DE PÓS-GRADUAÇÃO, EXTENSÃO E PESQUISA
MESTRADO PROFISSIONAL EM GESTÃO E TECNOLOGIA EM SISTEMAS PRODUTIVOS

CRISTIAN HENRIQUE MARTINS DE SOUZA

HEIMDALL: DETECÇÃO DE ARTEFATOS MALICIOSOS UTILIZANDO *MACHINE LEARNING* EM AMBIENTES *INTERNET OF THINGS* HABILITADOS POR REDES DEFINIDAS POR *SOFTWARE*

São Paulo
2024

CRISTIAN HENRIQUE MARTINS DE SOUZA

HEIMDALL: DETECÇÃO DE ARTEFATOS MALICIOSOS UTILIZANDO *MACHINE LEARNING* EM AMBIENTES *INTERNET OF THINGS* HABILITADOS POR REDES DEFINIDAS POR *SOFTWARE*

Dissertação apresentada como exigência parcial para a obtenção do título de Mestre(a) em Gestão e Tecnologia em Sistemas Produtivos do Centro Estadual de Educação Tecnológica Paula Souza, no Programa de Mestrado Profissional em Gestão e Tecnologia em Sistemas Produtivos, sob a orientação do Prof. Dr. Carlos Hideo Arima.

Área de Concentração: Sistemas Produtivos

Linha de Pesquisa: Gestão de Sistemas e de Tecnologia de Informação

Projeto de Pesquisa: Tecnologias Digitais na Indústria e Serviços 4.0

São Paulo

2024

FICHA ELABORADA PELA BIBLIOTECA NELSON ALVES VIANA
FATEC-SP / CPS CRB8-10894

Souza, Cristian Henrique Martins de
S729h Heimdall : detecção de artefatos maliciosos utilizando *machine learning* em ambientes *internet of things* habilitados por redes definidas por software / Cristian Henrique Martins de Souza. – São Paulo: CPS, 2024.
105 f. : il.

Orientador: Prof. Dr. Carlos Hideo Arima
Dissertação (Mestrado Profissional em Gestão e Tecnologia em Sistemas Produtivos) – Centro Estadual de Educação Tecnológica Paula Souza, 2024.

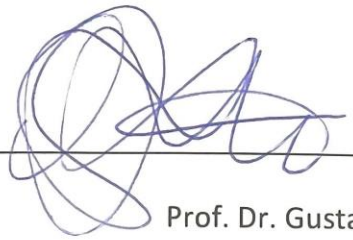
1. Detecção de malware. 2. Redes definidas por software. 3. Aprendizado de máquina. 4. Internet das coisas. I. Arima, Carlos Hideo. II. Centro Estadual de Educação Tecnológica Paula Souza. III. Título.

CRISTIAN HENRIQUE MARTINS DE SOUZA

HEIMDALL: DETECÇÃO DE ARTEFATOS MALICIOSOS UTILIZANDO *MACHINE LEARNING* EM AMBIENTES *INTERNET OF THINGS* HABILITADOS POR REDES DEFINIDAS POR *SOFTWARE*

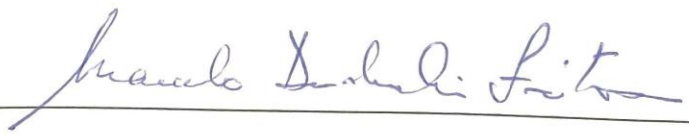
Prof. Dr. Carlos Hideo Arima

Orientador - CEETEPS



Prof. Dr. Gustavo Perri Galegale

Examinador Externo - FACULDADE DE INFORMÁTICA E ADMINISTRAÇÃO PAULISTA - FIAP



Prof. Dr. Marcelo Duduchi Feitosa

Examinador Interno - CEETEPS

São Paulo, 12 de novembro de 2024

Resumo

SOUZA, Cristian Henrique Martins de. **Heimdall: Detecção de Artefatos Maliciosos Utilizando Machine Learning em Ambientes Internet of Things Habilitados por Redes Definidas por Software**. 2024. 106 f. Dissertação (Mestrado em Gestão e Tecnologia em Sistemas Produtivos) – Centro Estadual de Educação Tecnológica Paula Souza, São Paulo, 2024.

Programas maliciosos continuam sendo um dos principais desafios para a segurança dos sistemas computacionais. O crescimento do paradigma tecnológico da Internet das Coisas (IoT, do inglês *Internet of Things*) tem gerado diversas preocupações a respeito da segurança dos dispositivos conectados à Internet, especialmente em ambientes industriais impulsionados pelas tecnologias 4.0, onde o comprometimento ou mau funcionamento de tais aparelhos pode ocasionar danos ao ambiente físico dos sistemas produtivos e colocar vidas humanas em risco. Proteger ambientes IoT é uma tarefa complexa, visto que dispositivos com características e funcionalidades distintas podem estar conectados à uma mesma infraestrutura. Logo, é essencial dispor de ferramentas agnósticas para detecção e contenção de ameaças cibernéticas. Nesse contexto, o paradigma das Redes Definidas por Software (SDN, do inglês *Software-Defined Networking*) se apresenta como um habilitador para o desenvolvimento de soluções capazes de detectar e isolar artefatos maliciosos em nível de rede. Ademais, o uso de técnicas de aprendizado de máquina eleva o potencial de detecção ao permitir a identificação rápida de artefatos desconhecidos. Diante do exposto, este trabalho de pesquisa propõe o Heimdall, uma ferramenta híbrida para detecção de artefatos maliciosos em ambientes IoT habilitados por SDN. A solução combina o uso de regras YARA e *machine learning* para classificação de artefatos maliciosos a partir da análise do tráfego da rede. As principais contribuições deste trabalho englobam a avaliação de diferentes algoritmos de aprendizado de máquina em um *dataset* robusto; a criação de uma arquitetura híbrida, genérica e resiliente para detecção de programas maliciosos em ambientes IoT habilitados por SDN; o desenvolvimento de uma prova de conceito para a arquitetura definida; e a avaliação da abordagem proposta a partir de exemplares reais de artefatos maliciosos. O algoritmo *Random Forest* implementado obteve uma acurácia de 99.33% no conjunto de dados de teste. Ao ser avaliado contra programas maliciosos reais, o Heimdall obteve uma taxa de detecção de 98.44% e um tempo de processamento médio de 0.0217s.

Palavras-chaves: Detecção de malware; Redes Definidas por Software; Aprendizado de Máquina; Internet das Coisas.

Abstract

SOUZA, Cristian Henrique Martins de. **Heimdall: Malware Detection in Software-Defined Network-enabled Internet of Things Environments Using Machine Learning**. 2024. 106 p. Dissertation (Master's Degree in Management and Technology in Productive Systems) – Centro Estadual de Educação Tecnológica Paula Souza, São Paulo, 2024.

Malicious software remains one of the main challenges for the security of computer systems. The growth of the Internet of Things (IoT) technological paradigm has raised several concerns regarding the security of devices connected to the Internet, especially in industrial environments driven by 4.0 technologies, where compromising or malfunctioning such devices can cause damage to the physical production systems environment and put human lives at risk. Protecting IoT environments is a complex task, as devices with different characteristics and functionalities may be connected to the same infrastructure. Therefore, it is essential to have agnostic tools for detecting and containing cyber threats. In this context, the Software-Defined Networking (SDN) paradigm emerges as an enabler for the development of solutions capable of detecting and isolating malicious artifacts at the network level. Furthermore, the use of machine learning techniques enhances detection potential by enabling the rapid identification of unknown artifacts. Given the above, this research proposes Heimdall, a hybrid approach for detecting malicious artifacts in SDNenabled IoT environments. The solution combines the use of YARA rules and machine learning for classifying malicious artifacts based on network traffic analysis. The main contributions of this work include: the evaluation of different machine learning algorithms on a robust dataset; the creation of a hybrid, generic, and resilient architecture for detecting malicious programs in SDN-enabled IoT environments; the development of a proof of concept for the defined architecture; and the evaluation of the proposed approach using real samples of malicious artifacts. The implemented Random Forest algorithm achieved an accuracy of 99.33% on the test dataset. When evaluated against real malicious programs, Heimdall achieved a detection rate of 98.44% and an average processing time of 0.0217s.

Keywords: Malware detection; Software-Defined Networking; Machine Learning; Internet of Things.

Lista de figuras

Lista de tabelas

Lista de abreviaturas e siglas

5G	Fifth Generation Mobile Network
API	Application Programming Interface
ARM	Advanced RISC Machine
ASIC	Application-Specific Integrated Circuit
BLE	Bluetooth Low Energy
CART	Classification and Regression Tree
CNN	Convolution Neural Network
CoAP	Constrained Application Protocol
CVE	Common Vulnerabilities and Exposures
CSV	Comma-Separated Values
DoS	Denial of Service
DDoS	Distributed Denial of Service
DNS	Domain Name System
DSL	Domain Specific Language
ELF	Executable and Linkable Format
FPGA	Field-Programmable Gate Array
FTP	File Transfer Protocol
GDPR	General Data Protection Regulation
HTTP	Hypertext Transfer Protocol
IAM	Identity and Access Management
IDS	Intrusion Detection System
IEC	International Electrotechnical Commission
IP	Internet Protocol
IPS	Intrusion Prevention System
IoMT	Internet of Medical Things
IoT	Internet of Things
ITU	International Telecommunication Union
JSON	JavaScript Object Notation
LSTM	Long Short Term Memory
LwM2M	Lightweight Machine-to-Machine

MiTM	Man-in-the-Middle
MLP	Multi-Layer Perceptron
MQTT	Message Queueing Telemetry Transport
MIPS	Microprocessor without Interlocked Pipeline Stages
MTD	Moving Target Defense
NG-SDN	Next-Generation SDN
NGFW	Next-Generation Firewall
NFV	Network Functions Virtualization
NIC	Network Interface Card
OTA	Over-The-Air
P4	Programming Protocol-independent Packet Processors
PE	Portable Executable
PFEs	Programmable Forwarding Engines
RAMI	Reference Architecture Model for Industry
REST	Representational State Transfer
RFID	Radio Frequency Identification
SANS	System Administration, Networking and Security
SDN	Software-Defined Networking
SMTP	Simple Mail Transfer Protocol
SPOF	Single Point of Failure
SQM	SDN Quarantined Network
SIEM	Security Information and Event Management
SVM	Support Vector Machine
TLS	Transport Layer Security
UDP	User Datagram Protocol
XSS	Cross-Site Scripting
WSN	Wireless Sensor Network
YARA	Yet Another Regex Analyzer

Sumário

1 INTRODUÇÃO

A quarta revolução industrial é caracterizada pela incorporação de tecnologias emergentes para automação de tarefas, o que traz ganhos significativos para os sistemas produtivos devido ao aumento da eficiência no uso de recursos e no desenvolvimento de produtos em larga escala (LASI et al., 2014). Nesse contexto, o uso de tecnologias como inteligência artificial, Internet das Coisas (IoT, do inglês *Internet of Things*), computação em nuvem e robótica se caracteriza como uma das principais tendências da Indústria 4.0, sendo essas tecnologias essenciais para impulsionar a transformação digital e a inovação nos processos industriais (PERES et al., 2020; KIM, 2017).

Com o advento da Indústria 4.0, foi necessário o desenvolvimento de normas e padrões visando orientar as etapas de projetos e protótipos. O Modelo de Referência para Arquitetura da Indústria 4.0 (do inglês, *Reference Architecture Model for Industry 4.0*) oferece uma estrutura tridimensional que facilita a padronização e interoperabilidade entre os diferentes sistemas e níveis hierárquicos da produção (HANKEL; REXROTH, 2015). Tal padronização é essencial para que dispositivos e plataformas sejam capazes de se comunicar de forma eficiente, garantindo que a produção seja flexível, descentralizada e interconectada (BASTOS et al., 2021).

Nesse cenário, a segurança é uma preocupação central do RAMI 4.0, visto que a interconexão de dispositivos e sistemas aumenta a superfície de ataque e pode amplificar o potencial dano causado por ameaças cibernéticas. Logo, a utilização de ferramentas de segurança que contribuam para a segurança do ambiente industrial é fundamental para a proteção dos dados, equipamentos e vidas humanas (ELKHAWAS; AZER, 2018).

Além disso, a quarta revolução também é marcada pelo avanço da interconectividade entre diferentes dispositivos por meio da tecnologia 5G (RAO; PRASAD, 2018). Isso permite uma maior integração entre sistemas e a coleta de dados em tempo real, resultando em tomadas de decisões mais ágeis e precisas (RATHEE et al., 2020). Como resultado, um grande volume de informações é gerado, trafegado e processado por dispositivos que nem sempre possuem medidas de segurança adequadas, principalmente em cenários de redes IoT (HASSAN et al., 2019; AL-GARADI et al., 2020).

A diversidade de fabricantes e dispositivos IoT dificulta a implementação de recursos de segurança eficazes na detecção de ameaças, visto que tais dispositivos possuem capacidades limitadas de armazenamento e processamento (KUMAR; DUBEY; PANDEY, 2021). Aliado a isso, a heterogeneidade de protocolos demanda o uso de abordagens adaptativas e com visão holística da infraestrutura para mitigação efetiva de ameaças modernas (MOHANTY et al., 2021).

Falhas de segurança em dispositivos IoT podem ser críticas à privacidade dos usuários e à segurança pessoal (ALLADI et al., 2020). O uso de protocolos inseguros pode expor informações sensíveis dos usuários a invasores por meio da interceptação do tráfego da rede (ZAMFIR et al., 2016). Ademais, um dispositivo comprometido (e.g., em uma rede industrial) pode ser utilizado para realização de ações no ambiente físico, ocasionando riscos a vidas humanas (YANG et al., 2022).

Malwares continuam sendo um dos principais desafios à segurança de sistemas computacionais. O advento do paradigma IoT foi acompanhado pelo aumento do número de programas maliciosos projetados para as arquiteturas ARM (do inglês *Advanced RISC Machine*) e MIPS (do inglês *Microprocessor without Interlocked Pipeline Stages*) (DARKI et al., 2019). Um exemplo disso é a utilização da *botnet* Mirai (KUMAR; CHANDAVARKAR, 2023), cujo objetivo é infectar e controlar dispositivos embarcados (como câmeras IP e roteadores) para execução de ataques de negação de serviço distribuídos (DDoS, do inglês *Distributed Denial of Service*) (KOLIAS et al., 2017).

Essas ameaças são responsáveis por diversos danos, como o comprometimento da integridade dos dados, roubo de informações confidenciais e prejuízos financeiros a usuários e corporações. Recentemente, *malwares* do tipo *ransomware* estão sendo amplamente utilizados por criminosos para impedir o acesso a arquivos por meio da encriptação das informações, exigindo um pagamento para resgate dos dados (RAZAULLA et al., 2023). A efetividade dos ataques por *ransomware* deu origem ao mercado criminoso de *Ransomware-as-a-Service* (RaaS), tornando a compra de ameaças avançadas acessível a todos os públicos (ALWASHALI; RAHMAN; ISMAIL, 2021).

Segundo o *ICS and OT threat predictions for 2024* (GONCHAROV, 2024), publicado pela Kaspersky Lab, empresa referência em segurança cibernética, *ransomwares* continuam sendo

uma das principais ameaças aos ambientes industriais. Já o *2023 Incident Response Analyst Report* (GERT, 2024), publicado pela mesma organização, destaca que 17.01% dos incidentes do ano de 2023 ocorreram em ambientes industriais, e que um a cada três ataques contou com o uso de *malware*. O relatório também indica que 21.82% dos casos relatados ocorreram no continente americano. Ademais, o uso generalizado de IoT em conjunto com a tecnologia 5G está gerando diversas discussões a respeito da segurança de tais dispositivos e do ambiente em que operam (SALAHDINE; HAN; ZHANG, 2023; AHMID; KAZAR, 2023; KUMAR; LYDIA; LEVRON, 2022).

Diversos avanços foram e vêm sendo feitos pela indústria e academia nas áreas de análise e detecção de artefatos maliciosos (GOPINATH; SETHURAMAN, 2023; KARA, 2023; AMIRA et al., 2023). Ferramentas de detecção devem ser capazes de identificar e conter ameaças desconhecidas (BALOGH; MOJŽIŠ; KRAMMER, 2022). Para isso, múltiplas técnicas de análise devem ser empregadas para aumentar a confiabilidade das soluções, sendo a análise comportamental a mais efetiva na detecção de *malwares* de dia zero (ABOAOJA et al., 2022).

Artefatos maliciosos podem interagir com outros dispositivos na rede de diferentes maneiras. Ações comuns incluem o envio de dados da vítima e a execução de varreduras (*scans*) na rede com o intuito de identificar outros dispositivos vulneráveis (DENG; MIR- KOVIC, 2022). Essas ações podem ser detectadas por ferramentas de IDS/IPS (do inglês *Intrusion Detection/Prevention System*). Entretanto, soluções tradicionais podem falhar na detecção de novas ameaças, visto que elas se baseiam em um conjunto predefinido de regras para tomada de decisão (CHIBA et al., 2022a). Como destacado por (CHIBA et al., 2022b), redes IoT requerem o uso de soluções de segurança adaptativas e capazes de identificar padrões maliciosos gerados por *malwares* ainda não catalogados.

Nesse contexto, o uso de *machine learning* tem se mostrado efetivo na detecção genérica de artefatos maliciosos em nível de rede (GAURAV; GUPTA; PANIGRAHI, 2023; TAYYAB et al., 2022). Para isso, um modelo é treinado com base em uma grande quantidade de dados a respeito de ameaças conhecidas. Uma vez treinado, ele é capaz de realizar previsões assertivas ao lidar com novas informações. Isso permite a detecção e classificação efetiva das ameaças com base na análise de tráfego, minimizando o risco de comprometimentos (ALQUDAH; YASEEN, 2020).

Aliado a isso, paradigmas como o das Redes Definidas por Software (SDN, do inglês *Software-Defined Networking*) e a virtualização das funções de rede (NFV, do inglês *Network Functions Virtualization*) têm viabilizado o desenvolvimento de soluções adaptativas e capazes de identificar ameaças na rede de forma holística (SILVA et al., 2023; MOZO et al., 2023; ALI; CHONG; MANICKAM, 2023). Uma das principais características das redes SDN é a alta programabilidade, viabilizada por meio da separação do plano de dados do plano de controle, o que facilita a orquestração de recursos de rede e a detecção de ameaças (LIATIFIS et al., 2023).

Pesquisas recentes elevam o potencial de SDN por meio da construção de planos de dados programáveis (HAUSER et al., 2023). Essa abordagem possibilita a reconfiguração sistemática das etapas de processamento de baixo nível aplicadas aos pacotes da rede, reduzindo a complexidade e aprimorando a utilização dos recursos dos *switches* (MICHEL et al., 2021; KFOURY; CRICHIGNO; BOU-HARB, 2021). Como destacado por (LIATIFIS et al., 2023), a próxima geração de SDN (NG-SDN, do inglês *Next-Generation SDN*) é caracterizada pelo uso de interfaces e *hardware* abertos, contrariamente às redes tradicionais, que adotam padrões proprietários e fechados. Para isso, tecnologias como a linguagem P4 estão sendo desenvolvidas para habilitar o desenvolvimento de soluções inovadoras que podem ser implantadas diretamente em *switches* programáveis (PETER; KOBO; SRIVASTAVA, 2022).

1.1 Motivação

Estudos recentes evidenciam os desafios e oportunidades para detecção eficiente de artefatos maliciosos no contexto de IoT (AYODELE; BUTTIGIEG, 2023). Tendo em vista que SDN é um elemento fundamental para a inovação e programabilidade nas redes de computadores, mecanismos que utilizam essa tecnologia têm se mostrado efetivos na mitigação de ameaças de *malware* (MADDU; RAO, 2023), especialmente considerando o atual cenário da crescente utilização da tecnologia 5G, onde dispositivos de fabricantes distintos são interconectados na mesma infraestrutura (SALAHDINE; HAN; ZHANG, 2023; KAZMI et al., 2023). A visão holística que o controlador SDN possui de toda a infraestrutura da rede permite a identificação e mitigação das ameaças de forma genérica e adaptativa (MALEH et al., 2023). Ademais, SDN

oferece capacidades que têm o potencial de aprimorar a eficácia na detecção de *malware* em comparação com abordagens tradicionais, como ferramentas de IDS ou *firewalls* de próxima geração (NGFW, do inglês *Next-Generation Firewall*). Algumas dessas vantagens incluem:

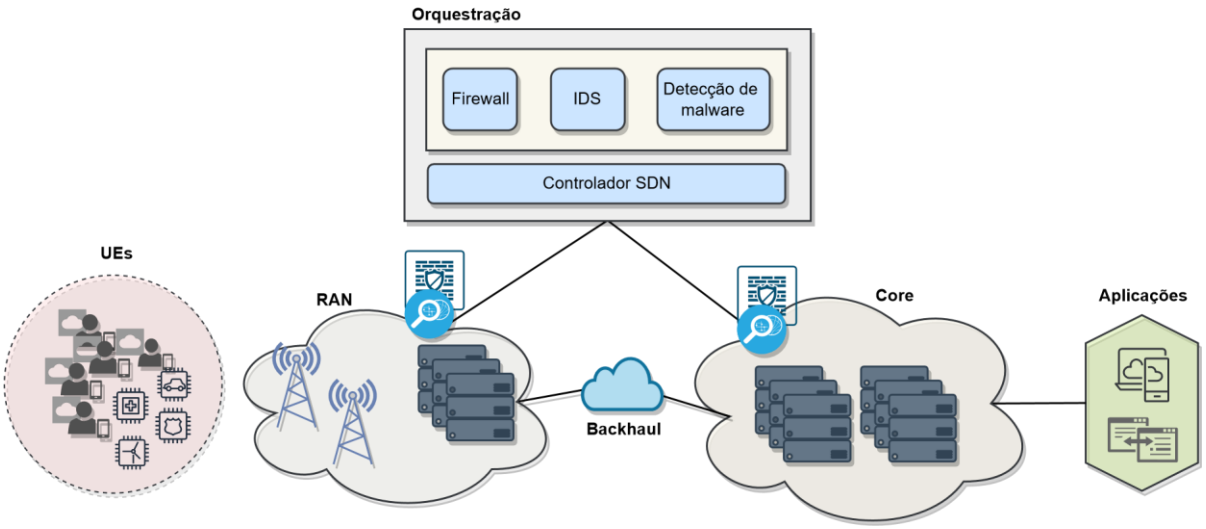
- Flexibilidade na gestão de políticas: a capacidade de configurar rapidamente as políticas da rede por meio do controlador SDN permite mitigar ameaças emergentes de forma mais precisa que as soluções tradicionais (PALADI, 2015).

- Visibilidade e controle: SDN permite uma visibilidade detalhada de todo o tráfego da rede e a capacidade de aplicar medidas de controle específicas em tempo real, o que é favorável para a detecção de atividades suspeitas (ALSAEEDI; MOHAMAD; AL-ROUBAIEY, 2019).

- Integração contextual: soluções SDN podem integrar informações contextuais e externas para aprimorar a detecção de ameaças (TAYLOR et al., 2016).

A Figura 1 ilustra um exemplo de um cenário onde a detecção e mitigação de *malware* é realizada com o suporte dos recursos oferecidos por uma infraestrutura habilitada por SDN. A visão holística do controlador SDN possibilita a implantação de módulos para defesa de toda a infraestrutura. Esses módulos podem atuar de forma agnóstica, detectando ameaças que afetam diferentes plataformas e sistemas operacionais (RAHOUTI et al., 2022). Esta abordagem possibilita a utilização de múltiplos módulos de proteção e não requer modificações substanciais na infraestrutura para operacionalizar o uso de novas ferramentas.

Figura 1 – Cenário hipotético para detecção de *malware* em SDN



Fonte: Resultado da pesquisa

Além de viabilizar uma melhor detecção de artefatos maliciosos na rede, a tecnologia SDN também pode ser utilizada como habilitadora de soluções com foco na análise de *malware*. Como destacado por (CERON; MARGI; GRANVILLE, 2016), SDN permite a estruturação de ambientes controlados e adaptativos para analisar programas maliciosos. Isso é obtido pela capacidade de rapidamente reconfigurar parâmetros da rede, permitindo a identificação de novos comportamentos do binário em análise.

O potencial das ferramentas de detecção pode ser elevado ainda mais por meio de *switches* programáveis. Nesse sentido, a utilização de dispositivos habilitados pela linguagem P4 proporciona diversos benefícios na análise dos fluxos de rede. Isso se deve à capacidade de integrar soluções de segurança diretamente ao *hardware*, eliminando a necessidade de um dispositivo central para o processamento das informações, o que, por sua vez, reduz consideravelmente a incidência de pontos únicos de falha (SPOF, do inglês *Single Point of Failure*) (KAUR; KUMAR; AGGARWAL, 2021).

A partir de uma revisão da literatura, foi possível constatar que a maior parte das soluções para detecção das ameaças de *malware* em ambientes IoT habilitados por SDN propostas pela academia estão diretamente atreladas ao controlador SDN. Dessa forma, caso ele seja alvo direto de ataques, especialmente os de negação de serviço, a segurança e operação da rede podem ser comprometidas (GKOUNTIS et al., 2017; XU et al., 2017). Logo, é

necessário o desenvolvimento de soluções mais resilientes a ataques direcionados, visando manter a estabilidade e segurança da rede.

Inspirado pelo poder e flexibilidade dos *switches* programáveis, este trabalho propõe o Heimdall, uma ferramenta para detecção de artefatos maliciosos em ambientes IoT habilitados pela tecnologia SDN. Para isso, é adotada uma abordagem híbrida, composta pela detecção de assinaturas maliciosas e pela classificação do tráfego por meio de *machine learning*. O Heimdall se apresenta como uma ferramenta de grande potencial no combate a ameaças de *malware* em IoT devido a sua capacidade de identificar artefatos que ainda estão sendo transferidos para o alvo, além de detectar fluxos maliciosos gerados por *malwares* em execução e prontamente isolar os dispositivos comprometidos. Diferentemente das abordagens tradicionais, a solução é acoplada diretamente aos *switches* da rede, o que reduz a ocorrência de pontos únicos de falha e otimiza a utilização dos recursos presentes na infraestrutura.

A ferramenta é avaliada em um ambiente controlado e emulado por meio da ferramenta Mininet, juntamente com o auxílio de *switches* virtualizados. São utilizados exemplares de *malwares* reais para validar a acurácia da solução. Ademais, aspectos como o tempo de processamento e complexidade da ferramenta também são avaliados. Os resultados obtidos evidenciam que o Heimdall obteve uma taxa de detecção real de 98.44% e tempo de processamento médio de 0.0217s, demonstrando que soluções integradas aos *switches* da rede se apresentam como abordagens rápidas, efetivas e de baixo custo para o combate a programas maliciosos.

1.2 Objetivos

Este trabalho de pesquisa tem como objetivo principal propor uma solução para detecção de artefatos maliciosos em ambientes IoT habilitados por SDN. Tendo em vista a variedade e criticidade dos dispositivos IoT que podem estar conectados a uma mesma rede, a identificação rápida e abrangente de ameaças requer o uso de soluções capazes de analisar e classificar o tráfego da rede com a menor sobrecarga possível.

Denominada Heimdall, a arquitetura funcional da ferramenta tem como objetivo aproveitar o potencial dos comutadores programáveis para viabilizar a detecção efetiva e genérica de *malwares*, bem como rapidamente isolar os dispositivos afetados do restante da rede, visando mitigar a propagação das ameaças. Para alcançar o objetivo principal, os seguintes objetivos específicos são elicitados:

1. Definição dos requisitos funcionais para detecção de programas maliciosos em ambientes IoT habilitados por SDN utilizando *machine learning* e tendo como base a adoção de *switches* programáveis;
2. Revisão bibliográfica de trabalhos relacionados que se propuseram a identificar artefatos maliciosos por meio de técnicas de *machine learning* em ambientes IoT habilitados pela tecnologia SDN;
3. Elaborar uma arquitetura híbrida para detecção genérica de *malwares* com foco em dispositivos IoT;
4. Implementar algoritmos de *machine learning* para detecção de artefatos maliciosos tendo como referência um *dataset* aprovado e validado pela academia;
5. Implementar e validar a arquitetura proposta a partir do modelo de maior acurácia e por meio de ferramentas de emulação com suporte ao P4;
6. Aferir a eficácia da solução, seus impactos na rede, discutir as lições aprendidas durante o desenvolvimento deste estudo e orientar o desenvolvimento de trabalhos futuros.

1.3 Proposições

Com base no exposto, este trabalho é fundamentado nas seguintes proposições:

- Proposição 1: a detecção de artefatos maliciosos em ambientes IoT habilitados por SDN pode ser aprimorada com o auxílio de uma abordagem híbrida baseada em assinaturas e *machine learning* acoplada diretamente em comutadores programáveis.

- Proposição 2: soluções para detecção e mitigação de programas maliciosos podem ser desenvolvidas sem a necessidade de um controlador SDN, eliminando pontos únicos de falha.

1.4 Metodologia científica

O primeiro passo adotado para o desenvolvimento desta pesquisa consistiu em uma revisão da literatura para identificar trabalhos relacionados. Foram utilizadas as bases de pesquisa ACM Digital Library, Hindawi, IEEE Xplore, MDPI, Science Direct, Springer Link e Wiley. Adicionalmente, os motores de busca Google Scholar, Scopus e Web of Science foram consultados. A revisão detalhada dos estudos é exposta na Seção 2.8.

Após a identificação das lacunas e oportunidades de pesquisa a partir da leitura dos trabalhos relacionados, a proposta de arquitetura do Heimdall foi concebida. Para sua implementação, foi utilizadas as linguagens Python e P4. A solução foi avaliada em uma plataforma de testes emulada por meio da ferramenta Mininet e com o uso do *software* BMv2 P4. Para os cenários de avaliação propostos, foi considerado que todos os dispositivos estão na mesma infraestrutura de rede. As avaliações foram realizadas em uma máquina com CPU Intel Core i7-11800H 2.30GHz (8 vCPUs), 32GB de RAM e sistema operacional Ubuntu Server 22.04.3 LTS 64-bits (Linux Kernel 6.2). Os detalhes sobre a proposta e os resultados práticos da implementação são apresentados nos Capítulos 3 e 4 respectivamente.

1.5 Contribuições

As principais contribuições deste trabalho são:

1. Avaliar diferentes algoritmos de *machine learning* em um *dataset* robusto, visando identificar o modelo mais eficaz na classificação de programas maliciosos;
2. Propor uma arquitetura híbrida, genérica e resiliente para detecção de *malwares* baseada em assinaturas maliciosas e *machine learning*;

3. Desenvolver uma solução como prova de conceito para detecção de programas maliciosos em ambientes IoT habilitados por SDN que é acoplada diretamente aos *switches* da rede;
4. Avaliar a abordagem híbrida proposta a partir de exemplares reais de artefatos maliciosos.

1.6 Produção acadêmica

A partir das atividades de pesquisa e implementação realizadas durante a execução deste estudo, as seguintes publicações foram obtidas:

- SOUZA, Cristian HM; ARIMA, Carlos H. A hybrid approach for malware detection in SDN-enabled IoT scenarios. **Internet Technology Letters**, p. e534.
- SOUZA, Cristian HM; ARIMA, Carlos H. Detecção de malware em ambientes IoT habilitados por SDN. In: **Anais do XV Workshop de Pesquisa Experimental da Internet do Futuro**. SBC, 2024.
- SOUZA, Cristian HM; ARIMA, Carlos H. Avaliação de algoritmos de machine learning para detecção de malware IoT no dataset IoT-23. In: **Anais do XXIV Simpósio Brasileiro de Segurança da Informação e de Sistemas Computacionais**. SBC, 2024.

1.7 Produção industrial

A execução deste trabalho também resultou no registro do *software* intitulado *"Heimdall: Solução para detecção de artefatos maliciosos em ambientes IoT por meio de machine learning"*, pelo Instituto Nacional da Propriedade Industrial (INPI), sob o número de registro BR512024000157-3. Ademais, a interface administrativa da ferramenta foi registrada no mesmo instituto com o título *"Heimdall-NG - Interface administrativa"* e número de registro BR512024001975-8.

1.8 Estrutura do trabalho

Este trabalho está organizado em cinco capítulos, incluindo esta introdução. O Capítulo 2 apresenta a fundamentação teórica que apoia este estudo e trabalhos relacionados; o Capítulo 3 expõe em detalhes os componentes e subcomponentes da arquitetura proposta; o Capítulo 4 apresenta os resultados obtidos com a implementação da solução; e, por fim, o Capítulo 5 apresenta as considerações finais do trabalho e aponta direções de pesquisa para o desenvolvimento de novos estudos.

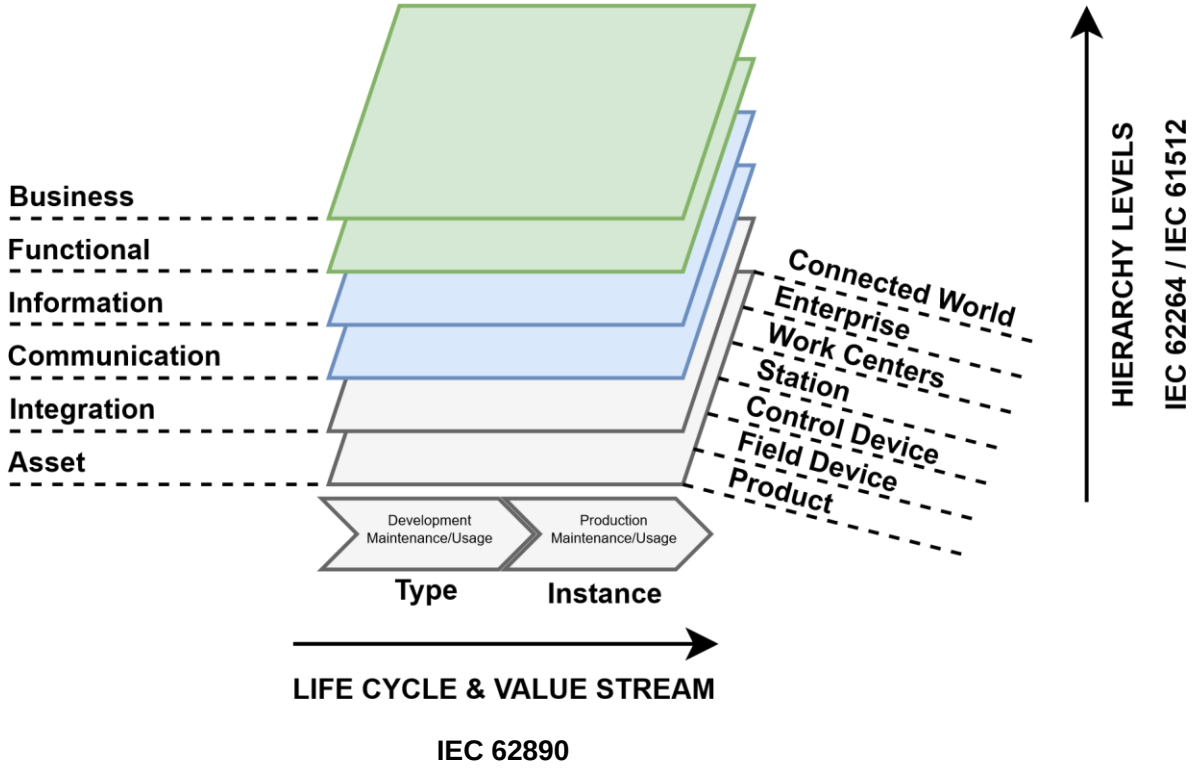
2 FUNDAMENTAÇÃO TEÓRICA

Este capítulo tem como objetivo apresentar os referenciais teóricos fundamentais para a compreensão deste trabalho, a saber: RAMI 4.0, Internet das Coisas, Redes Definidas por Software, *machine learning*, análise de *malware*, detecção de *malware* e trabalhos relacionados.

2.1 RAMI 4.0

O RAMI 4.0 consiste em uma estrutura de referência fundamental para implementação de tecnologias emergentes no contexto industrial. O modelo proporciona uma abordagem sistemática que visa assegurar a interoperabilidade entre diferentes componentes e sistemas (HANKEL; REXROTH, 2015; RIEDMANN; BINDER; NEUREITER, 2024). Sua estrutura tridimensional é sumarizada na Figura 2 e descrita nas subseções a seguir.

Figura 2 – Modelo de Referência para Arquitetura da Indústria 4.0



Fonte: Elaborado pelo autor. Adaptado de (HANKEL; REXROTH, 2015)

2.1.1 Eixo dos níveis hierárquicos

No eixo horizontal direito do modelo estão os níveis hierárquicos da IEC 62264, que contém uma série de normas internacionais para TI empresarial e sistemas de controle. Os níveis hierárquicos representam as diferentes funcionalidades dentro de fábricas ou instalações. Para representar o ambiente da Indústria 4.0, essas funcionalidades foram expandidas para incluir peças de trabalho, rotuladas como "Produto", e a conexão com a Internet das Coisas e serviços, rotulada como "Mundo Conectado".

2.1.2 Eixo do ciclo de vida e fluxo de valor

O eixo horizontal esquerdo representa o ciclo de vida das instalações e produtos, baseado na IEC 62890, utilizada na medição, controle e automação de processos industriais. Além disso, é feita uma distinção entre tipos e instâncias. Um "tipo"se torna uma "instância"quando o *design* e a prototipagem foram concluídos e o produto real está sendo fabricado. O RAMI 4.0 também combina todos os elementos e componentes de TI no modelo de camadas e ciclo de vida.

2.1.3 Eixo das camadas

As seis camadas no eixo vertical descrevem a decomposição de uma máquina em suas propriedades, estruturadas camada por camada, ou seja, o mapeamento virtual de uma máquina. Essa abordagem é comumente utilizada em sistemas complexos, onde suas propriedades são divididas em camadas. As camadas incluem aspectos como o *hardware*, as funções de controle, os dados de comunicação, os serviços e as funções de negócios. Cada camada adiciona um nível de abstração que ajuda a isolar diferentes aspectos do sistema, tornando mais fácil identificar, analisar e resolver problemas. Além disso, essa estrutura modular permite a integração de novos componentes e tecnologias de forma mais eficiente, promovendo flexibilidade e escalabilidade no desenvolvimento e implementação de soluções industriais.

2.1.4 Normas IEC de apoio ao RAMI 4.0

O RAMI 4.0 é suportado por várias normas da IEC (*International Electrotechnical Commission*), que estabelecem padrões internacionais para garantir a interoperabilidade e a segurança dos sistemas industriais. Algumas das normas mais relevantes incluem:

- IEC 62264: define modelos de integração para a automação industrial, facilitando a comunicação entre diferentes níveis hierárquicos.
- IEC 61508: aborda a segurança funcional dos sistemas elétricos, eletrônicos e programáveis eletrônicos, garantindo que tais sistemas operem corretamente nas instalações industriais.
- IEC 62443: trata a segurança cibernética para tecnologia operacional em sistemas de automação e controle industrial, fornecendo diretrizes para proteger esses sistemas contra ameaças cibernéticas.

Essas normas fornecem a base técnica necessária para implementar o RAMI 4.0 de maneira segura e eficiente, assegurando que as soluções de Indústria 4.0 atendam aos mais altos padrões de qualidade e segurança.

2.1.5 O RAMI 4.0 no contexto de segurança em IoT

A Indústria 4.0 é caracterizada pelo uso de sistemas e máquinas com funções distribuídas pela rede. A infraestrutura digital que sustenta a Indústria 4.0 é vulnerável a diferentes ameaças cibernéticas, que podem comprometer a integridade, a confidencialidade e a disponibilidade dos dados e sistemas industriais. Portanto, a segurança em IoT é fundamental para garantir a operação contínua e segura dos processos industriais.

Medidas de segurança robustas, como autenticação forte, criptografia, proteção anti-*malware*, monitoramento contínuo e atualizações regulares de *software*, são essenciais para proteger os dispositivos IoT e os sistemas industriais contra ataques cibernéticos. A falta de segurança adequada pode levar a interrupções significativas na produção, perdas financeiras e danos à reputação da organização.

2.2 Internet das Coisas

O conceito de Internet das Coisas foi originalmente definido no início dos anos 2000, emergindo a partir das pesquisas realizadas pelo Auto-ID Center[1], do Instituto de Tecnologia de Massachusetts (MIT), nas áreas de identificação por radiofrequência (RFID, do inglês *Radio Frequency Identification*) (WEINSTEIN, 2005) e redes de sensores sem fio (RSSF ou, em inglês, *Wireless Sensor Network* - WSN) (YICK; MUKHERJEE; GHOSAL, 2008). Em 2005, a União Internacional de Telecomunicações[2] (ITU, do inglês *International Telecommunication Union*) introduziu formalmente o paradigma IoT, que rapidamente ganhou a atenção da academia e indústria (RAYES et al., 2017).

Desde a sua concepção, diversas pesquisas vêm sendo conduzidas com o objetivo de interconectar dispositivos comuns à Internet (KIM; RAMOS; MOHAMMED, 2017). Estes dispositivos são capazes de coletar informações e realizar ações de forma autônoma. Isso se dá por meio de diferentes protocolos de comunicação, como o Zigbee, BLE e RFID (HOFERSCHMITZ; STOJANOVIĆ, 2020). A coleta, compartilhamento e análise de dados em tempo real abriu novas possibilidades em diferentes setores da indústria, incluindo áreas como a da saúde (VISHNU; RAMSON; JEGAN, 2020) e automobilística (RAHIM et al., 2021).

Notavelmente, IoT desempenha um papel fundamental na Indústria 4.0 devido à capacidade de interconectar máquinas e sensores em ambientes industriais, permitindo a automatização de processos de fabricação e coleta de dados abrangente sobre as operações (como equipamentos, estoques e qualidade) (ZHANG; CHEN, 2020; MUNIRATHINAM, 2020). O uso de tais informações permite a otimização da produção e melhora na eficiência, além de reduzir custos operacionais (BELLI et al., 2019). A incorporação de IoT na Indústria 4.0 não apenas promove a modernização das operações industriais, mas também ajuda a identificar e mitigar problemas (MANAVALAN; JAYAKRISHNA, 2019).

Embora o paradigma IoT tenha revolucionado a forma de comunicação entre dispositivos em redes de computadores, diversas preocupações a respeito da segurança das informações nesses ambientes continuam em aberto (HASSIJA et al., 2019; ALLADI et al., 2020). Por estarem normalmente conectados à Internet, dispositivos IoT são alvos potenciais para

[1] https://autoid.mit.edu

[2] https://www.itu.int

ataques e, ao mesmo tempo, podem servir como uma porta de entrada para invasores nas redes corporativas (BI et al., 2022). Além disso, após invadidos, eles podem ser utilizados para execução e amplificação de outros ataques, como os de negação de serviço distribuídos (SILVA et al., 2020).

Também é importante destacar que a diversidade de fabricantes e de dispositivos IoT dificulta a padronização de requisitos de segurança mínimos para a operação deles em redes modernas (LEE et al., 2021). Ademais, a limitação de recursos computacionais e de armazenamento em muitos desses dispositivos dificulta a implementação de atualizações de segurança (como as de *firmware*), frequentemente exigindo intervenção manual por parte dos administradores. Esses obstáculos realçam a necessidade de novas abordagens para a proteção de infraestruturas IoT diante dos cenários cada vez mais complexos e diversificados.

2.2.1 Arquitetura conceitual de IoT

A arquitetura conceitual de IoT pode ser compreendida por meio de três camadas (YUN; YUXIN, 2010): aplicação, rede e percepção; como exposto na Figura 3. As subseções a seguir apresentam os detalhes de cada uma das camadas.

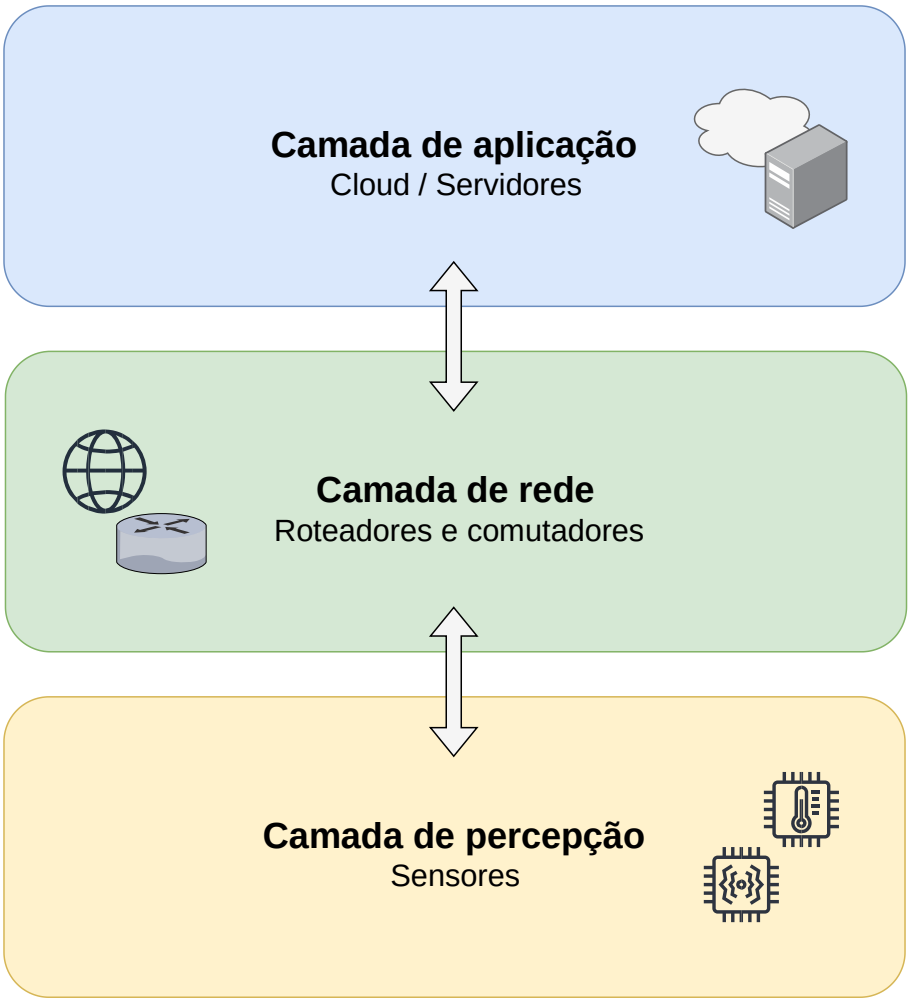
Camada de percepção

A camada de percepção também é conhecida como camada de sensores. Sua função é coletar informações do ambiente físico. Isso ocorre por meio de sensores e dispositivos (MRABET et al., 2020). Por exemplo, os sensores podem monitorar informações como temperatura, umidade e pressão. Os sensores são escolhidos de acordo com os requisitos das aplicações. Os dados coletados são convertidos para informações digitais, que são processadas e enviadas para a camada seguinte.

Segundo (BURHAN et al., 2018), o foco de atacantes nesta camada é a substituição de sensores legítimos por sensores maliciosos e a interceptação das comunicações. Logo, ameaças na camada de percepção incluem:

Eavesdropping: consiste na interceptação de comunicações privadas (como chamadas telefônicas e mensagens de texto). Essa técnica se torna efetiva quando a comunicação entre dois dispositivos é realizada sem criptografia, permitindo ao atacante a captura das informações em um formato entendível (KAPETANOVIC; ZHENG; RUSEK, 2015).

Figura 3 – Camadas da arquitetura conceitual de IoT



Fonte: Elaborado pelo autor. Adaptado de (YUN; YUXIN, 2010)

- Captura de nó: é considerado um dos ataques mais perigosos na camada de percepção de IoT e ocorre quando um atacante obtém o controle total de um nó chave da infraestrutura (como um *gateway*). Isso pode resultar na captura de todas as informações trafegadas na rede de forma insegura (BONACI; BUSHNELL; POOVENDRAN, 2010).

- Nó falso e malicioso: acontece quando um atacante adiciona um nó com dados falsos na rede e propaga dados falsos. O objetivo deste ataque é dificultar ou interromper a transmissão de dados legítimos (DUANGPHASUK; DUANGPHASUK; THAMMARAT, 2020).

- Ataques de *replay*: ocorre quando um atacante intercepta e retransmite informações válidas que trafegam em uma rede. Essas informações podem incluir dados confidenciais, como credenciais de acesso (PUTHAL et al., 2016).

- Ataques baseados no tempo de resposta: permitem que o atacante obtenha informações sobre um dispositivo por meio da análise do tempo de resposta a diferentes requisições realizadas (BUTT et al., 2019).

Camada de rede

Na camada de rede ocorrem abstrações para as tecnologias de comunicação, serviços de gerenciamento e roteamento. É a camada responsável pelo envio dos dados coletados dos sensores e dispositivos para a próxima camada. Para isso, são utilizadas tecnologias e protocolos de transmissão de dados com ou sem fio, como: Bluetooth, ZigBee, Wi-Fi, 5G e MODBUS (CHAHID; BENABDELLAH; AZIZI, 2017). A proteção da camada de rede é fundamental para garantir a conectividade entre os sensores, dispositivos IoT e interfaces com o usuário. Ademais, a infraestrutura utilizada deve ser resiliente para lidar com o aumento do número de dispositivos conectados. Segundo (BURHAN et al., 2018), ameaças comuns na camada de rede incluem:

- Negação de serviço: ataques de DoS (*Denial of Service*) têm como objetivo impedir o acesso legítimo a recursos na rede. Agentes maliciosos normalmente enviam uma grande quantidade de tráfego ao alvo no intuito de sobrecarregá-lo e evitar que ele atenda solicitações de clientes reais (VISHWAKARMA; JAIN, 2020).

- *Man-in-the-Middle* (MiTM): consiste na interceptação do tráfego entre dispositivos na rede por parte do atacante. Visto que o agente malicioso controla a comunicação, ele pode alterar informações conforme a sua necessidade, especialmente se tratando de tráfego não criptografado (LI et al., 2017).

- Ataques de desautenticação: visam interromper a conexão dos dispositivos IoT em redes *wireless*. No momento da reconexão, o atacante pode capturar a sequência de pacotes trocados com o objetivo de identificar a chave de conexão por meio de ataques de força bruta (KRISTIYANTO; ERNASTUTI, 2020).

Camada de aplicação

A camada de aplicação é responsável pela entrega de serviços ao usuário final. Nesta camada os dados são armazenados, processados e utilizados para realização de ações, como o envio de alertas, ativação de alarmes, controle de dispositivos físicos, entre outras funcionalidades. Técnicas de *big data* e de inteligência artificial podem ser utilizadas nessa camada com o objetivo de melhorar a eficiência no tratamento dos dados e na tomada de decisões (MARJANI et al., 2017; AHANGER; ALJUMAH; ATIQUZZAMAN, 2022).

Diante do baixo poder computacional dos dispositivos IoT, protocolos de comunicação específicos são empregados para troca rápida de informações. Por exemplo, o protocolo *Message Queueing Telemetry Transport* (MQTT) é ideal para aplicações que possuem uma largura de banda limitada (SONI; MAKWANA, 2017). Já o *Constrained Application Protocol* (CoAP) foi projetado para redes com baixo consumo de energia e normalmente é combinado com o *User Datagram Protocol* (UDP), o que o torna altamente eficiente (RAHMAN; SHAH, 2016). O protocolo *Lightweight Machine-to-Machine*
(LwM2M) foi desenvolvido com base no CoAP e permite uma comunicação eficiente e com baixo consumo de energia, além de adicionar recursos de segurança por padrão (como autenticação, criptografia e controle de acesso) (ALLIANCE, 2017).

É importante notar que a escolha do protocolo dependerá do contexto no qual o dispositivo será inserido e nos requisitos específicos de cada aplicação, como: latência, confiabilidade, largura de banda e compatibilidade entre dispositivos. Protocolos tradicionais, como o HTTP, projetados para lidar com grandes quantidades de dados, podem não ser apropriados para dispositivos IoT, que normalmente trafegam informações mínimas em cada pacote (WUKKADADA et al., 2018).

O estudo de (BURHAN et al., 2018) destaca que o uso de IoT em casas inteligentes introduz novas ameaças, bem como vulnerabilidades externas e internas. De acordo com os autores, ameaças comuns nesta camada incluem: ataques de *Cross-Site Scripting* (XSS) nas interfaces de controle dos dispositivos; *malwares* utilizados para causar danos ao ambiente; e perda de dados devido a problemas de conexão entre os dispositivos.

2.3 Principais desafios de IoT

A revolução tecnológica da Internet das Coisas também despertou o interesse de pesquisas relacionadas aos desafios de segurança, questões legais e aspectos éticos em tais ambientes. As subseções a seguir abordam estes tópicos em detalhes.

2.3.1 Segurança

O paradigma IoT introduz novas preocupações a respeito da segurança das informações, especialmente em um ambiente notoriamente diversificado. Dispositivos inseguros ou configurados de forma inadequada podem ser explorados e utilizados como porta de entrada para a execução de ataques mais sofisticados. Logo, garantir a confidencialidade, integridade e disponibilidade deve ser uma prioridade em redes IoT (SWESSI; IDOUDI, 2022).

Aspectos como a diversidade de dispositivos e fabricantes impõem desafios significativos na adoção de padrões de segurança. Como elencado por (JURCUT; RANAWEERA; XU, 2020), a mitigação de ameaças se torna eficaz quando a segurança é considerada como um requisito desde as fases de planejamento e *design* do dispositivo. Dessa forma, princípios de segurança devem ser incorporados durante todo o ciclo de desenvolvimento do produto, seja um *hardware* embarcado ou o *software* responsável pelo controle dele.

2.3.2 Privacidade

Preocupações a respeito da privacidade e proteção de dados pessoais também foram levantadas em diferentes estudos (SHAHID et al., 2022; LIANG; JI, 2022; RIZI; SENO, 2022).

Dispositivos IoT podem coletar uma ampla gama de informações dos seus utilizadores, como localização, informações de saúde e hábitos. O endereçamento dessas preocupações vai além do uso da criptografia para mitigar ataques de interceptação. É fundamental prover mecanismos seguros de autenticação e autorização, especialmente se os dispositivos forem compartilhados entre pessoas. Ademais, é necessário garantir que os usuários estejam cientes do uso e coleta dos seus dados, especialmente se eles forem compartilhados com terceiros.

2.3.3 Interoperabilidade

Os diferentes padrões, protocolos e fabricantes também representam um desafio para o bom funcionamento de redes IoT. Esses problemas se dão em todas as camadas da arquitetura IoT. Segundo (HATZIVASILIS et al., 2018), a interoperabilidade ocorre em quatro níveis: tecnológico, sintático, semântico e organizacional. Para endereçar problemas de interoperabilidade, soluções adaptativas e abordagens definidas por *software* vêm sendo empregadas pela academia e indústria (ALBOUQ et al., 2022; AGBAJE et al., 2022).

2.3.4 Aspectos legais

Aspectos legais e regulatórios devem ser considerados na adoção de tecnologias baseadas no paradigma de IoT (KARALE, 2021). Este cenário se apresenta complexo, visto que cada país possui suas próprias jurisdições a respeito da privacidade e uso de dados. Neste aspecto, o Regulamento Geral sobre a Proteção de Dados (GDPR, do inglês *General Data Protection Regulation*) padroniza as diretrizes para o tratamento de dados pessoais dos cidadãos da União Europeia (HOOFNAGLE; SLOOT; BORGESIUS, 2019).

O estudo conduzido por (KOUNOUDES; KAPITSAKI, 2020) destaca que IoT introduz novos requisitos para o tratamento de dados, além de apresentar as características básicas que um *framework* de privacidade para ambientes IoT deve satisfazer para possibilitar a proteção dos dados pessoais dos usuários, sendo os principais: *(i)* prevenir a inferência de informações; *(ii)* prover a transformação de dados; *(iii)* informar ao usuário sobre a utilização

dos seus dados; *(iv)* fornecer ao usuário o controle dos seus dados; e *(v)* monitorar e controlar os dispositivos que coletam dados.

2.3.5 Ética e responsabilidade

A ética é uma área da filosofia que se dedica ao estudo da conduta humana na sociedade, visando determinar o que é moralmente correto ou errado. O amplo uso de IoT levanta questões éticas importantes, principalmente no aspecto de sistemas com capacidade de tomada de decisões de forma autônoma. Desafios éticos resultantes da adoção de IoT foram identificados por (TZAFESTAS, 2018; ATLAM; WILLS, 2020; KARALE, 2021).

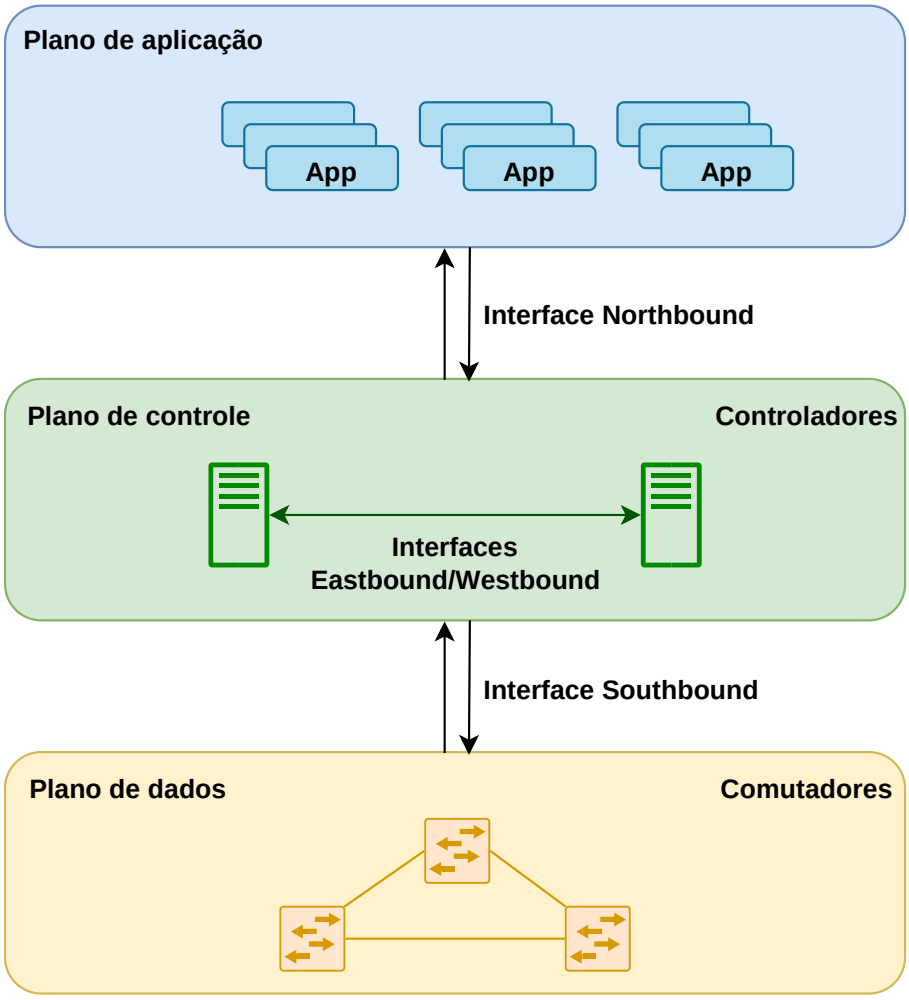
Como destacado por (POLLARD, 2015), dispositivos IoT devem adotar os seguintes padrões éticos por *design*: *(i)* capacidade de gerenciar e controlar a coleta e distribuição de dados pessoais; *(ii)* capacidade de aplicar regras e políticas independentemente do tempo e espaço; *(iii)* capacidade de suportar contextos dinâmicos (por exemplo, casa e escritório); e *(iv)* capacidade de observar, reconhecer e sustentar relações que demandem opções éticas.

2.4 Redes Definidas por Software

A abordagem das Redes Definidas por Software (SDN) tem como objetivo proporcionar uma maior flexibilidade, escalabilidade e programabilidade para as redes de computadores modernas. Isso é alcançado por meio da separação do plano de dados do plano de controle. Dessa forma, regras de encaminhamento podem ser facilmente ajustadas e configuradas com o objetivo de otimizar a performance da rede e mitigar problemas de segurança (KREUTZ et al., 2014).

Como ilustrado na Figura 4, a arquitetura SDN é composta por três planos principais: *(i)* o plano de aplicação; *(ii)* o plano de controle; e *(iii)* o plano de dados. O plano de aplicação engloba as aplicações cujo objetivo é manipular o comportamento dos dispositivos da rede. O plano de controle gerencia o comportamento dos diferentes dispositivos conectados à infraestrutura. Por fim, o plano de dados é composto pelos comutadores de pacotes (*switches*), que podem ser reconfigurados de maneira automatizada pelo controlador SDN.

Figura 4 – Arquitetura SDN



Fonte: Elaborado pelo autor. Adaptado de (KREUTZ et al., 2014)

Toda a comunicação entre o controlador central e os dispositivos da rede acontece por meio de interfaces de programação (APIs). As APIs são divididas em: *(i)* API *Northbound*; *(ii)* API *Southbound*; e *(iii)* APIs *Eastbound* e *Westbound*. A API *Northbound* é utilizada para interligar o plano de aplicação e o plano de controle, já a comunicação entre os controladores é feita por meio das interfaces *Eastbound* e *Westbound*. Por fim, a interface *Southbound* realiza o envio e coleta de informações entre o controlador e os diversos *switches* SDN que podem estar conectados na rede. No contexto de SDN, o protocolo OpenFlow (MCKEOWN et al., 2008) ganhou um grande destaque na comunidade acadêmica por oferecer tabelas de fluxo nos *switches* para armazenamento de regras de encaminhamento definidas pelo controlador.

2.4.1 Requisitos de segurança em arquiteturas IoT habilitadas por SDN

A natureza heterogênea e distribuída dos dispositivos IoT introduz desafios de segurança significativos em arquiteturas tradicionais e também nas baseadas em SDN. A integração entre IoT e SDN requer a adoção de uma abordagem de segurança multicamadas e abrangente, visando eliminar pontos únicos de falha. A seguir são apresentados os requisitos essenciais a serem considerados ao implantar uma infraestrutura IoT habilitada por SDN.

Autenticação e autorização robustas

É fundamental que dispositivos IoT sejam capazes de realizar processos de autenticação em redes SDN de maneira segura antes de estabelecer qualquer comunicação. Nesse contexto, o uso de certificados digitais, autenticação mútua e protocolos seguros devem ser implementados (KIM; LEE, 2017). Ademais, também é necessário estabelecer fortes padrões de autorização que definam claramente o que cada usuário ou dispositivo está autorizado a realizar na rede.

Gestão de identidades e acessos

A utilização de um sistema de gestão de identidades e acessos (IAM, do inglês *Identity and Access Management*) permite gerenciar contas de dispositivos IoT de maneira eficaz (ALAMRI; CROWLEY; RICHARDSON, 2022). Um sistema de IAM é capaz de registrar atividades de autenticação e autorização, o que permite criar trilhas de auditoria detalhadas no caso de violações e incidentes cibernéticos. Além disso, conforme a infraestrutura cresce, o IAM permite adicionar e gerenciar novos dispositivos e usuários sem comprometer a segurança e a eficiência da gestão (NUR; WANG, 2021).

Proteção do controlador

Por possuir uma visão holística da rede, o controlador SDN é um alvo constante de ataques. A exploração de vulnerabilidades focadas no controlador SDN pode resultar na degradação ou indisponibilidade dos serviços da rede (MALEH et al., 2023). Uma estratégica bastante empregada por administradores consiste no uso de múltiplos controladores SDN, além de estratégias de defesa baseadas na movimentação do alvo (MTD, do inglês *Moving Target Defense*) (SILVA et al., 2023).

Criptografia de dados

É fundamental garantir que todos os dados transmitidos pelos dispositivos sejam criptografados. Nesse contexto, interfaces *web* e APIs de gerenciamento devem ser protegidas por TLS (*Transport Layer Security*) (SIWAKOTI; RAWAT, 2023). Além disso, o uso de protocolos inseguros (sem criptografia) deve ser evitado em todas as camadas. Ao se adotar tais medidas, agentes maliciosos na rede são incapazes de visualizar claramente os dados trafegados, evitando o comprometimento da confidencialidade das informações (PARIS; HABAEBI; ZYOUD, 2023).

Segmentação de rede

A segmentação de rede é uma técnica que permite dividir uma rede em múltiplos segmentos (também chamados de sub-redes). Essa é uma prática comum em ambientes corporativos, onde há a necessidade de segregar as redes de funcionários e visitantes, por exemplo. Isso ajuda a limitar o impacto de violações de segurança, além de evitar que um dispositivo comprometido tenha acesso a outros segmentos da infraestrutura. A segmentação de rede pode ser implementada de forma física ou lógica (JAMES; RABBI,
2023).

Detecção de intrusão e anomalias

A detecção rápida de ameaças é importante para evitar maiores comprometimentos e prontamente bloquear as comunicações originadas de dispositivos maliciosos. Nesse contexto, a utilização de sistemas de detecção de intrusões e sistemas de prevenção de intrusões é essencial para monitorar o tráfego da rede, identificar comportamentos suspeitos e tomar as devidas providências (MOUSTAFA et al., 2023; HASSAN et al., 2023).

Atualizações de segurança

Com o objetivo de evitar a exploração de vulnerabilidades conhecidas, é crucial garantir que os dispositivos e *softwares* estejam devidamente atualizados. Esse processo inclui a atualização dos *firmwares* dos dispositivos, sistemas operacionais e aplicativos utilizados. Como destacado por (BAZZI; SHAOUT; MA, 2023), atualizações *over-the-air* (OTA) podem ser empregadas para a atualização de pacotes de segurança, correções de erros, entre outros. A vantagem dessa abordagem é não necessitar da interação com o
usuário.

Controle de tráfego

É preciso implementar políticas de controle de tráfego visando limitar tráfego desnecessário ou suspeito. Isso envolve a alocação de largura de banda de forma inteligente para os diferentes tipos de tráfego em IoT (por exemplo, priorizar o tráfego crítico para operações em tempo real, como o originado por dispositivos médicos), bem como utilizar protocolos de comunicação otimizados, como o MQTT, que é adequado para redes com largura de banda limitada. Essas práticas ajudam a reduzir a sobrecarga na rede, garantindo um desempenho eficiente e confiável das operações em redes IoT (VERGÜTZ et al., 2023).

Monitoramento contínuo

O monitoramento contínuo do tráfego e a identificação de anomalias é essencial para detectar e responder a ameaças (RAPOSO et al., 2018). Nesse contexto, soluções de SIEM (*Security Information and Event Management*) devem ser implementadas em infraestruturas corporativas para a agregação de *logs* e correlação de eventos. Um SIEM permite monitorar tentativas de acesso a recursos críticos, modificações não autorizadas em arquivos e execuções de programas maliciosos (HRISTOV et al., 2021). Além disso, é possível acompanhar os eventos em tempo real por meio de *dashboards*.

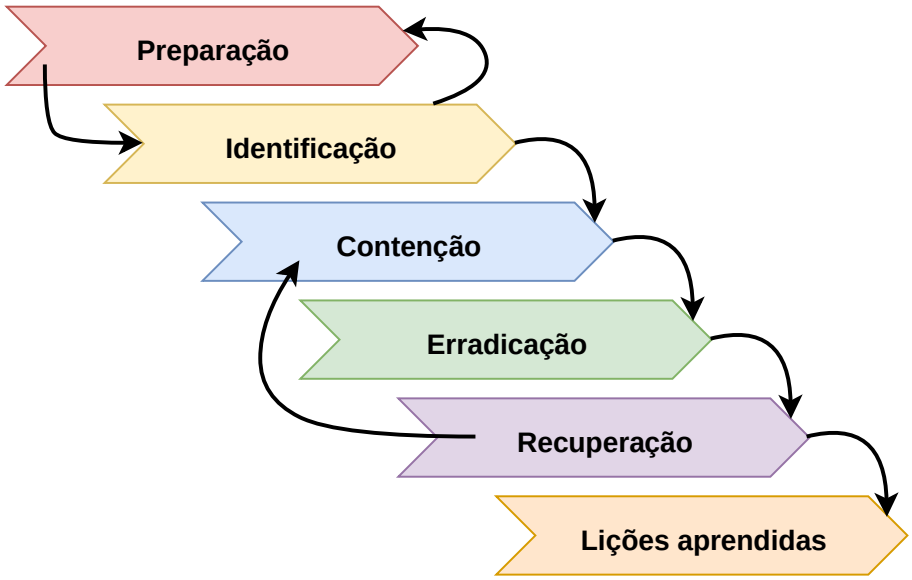
Isolamento de dispositivos comprometidos

O uso da tecnologia SDN permite o rápido isolamento de dispositivos infectados do restante da rede, mitigando ameaças como *worms* e evitando o comprometimento de outras máquinas presentes na infraestrutura (JAFARIAN et al., 2021). Isso pode ser atingido por meio de técnicas como a virtualização das funções de rede, onde dispositivos comprometidos podem ser movidos para uma quarentena virtual, com conectividade limitada, até que o impacto do incidente seja definido (RAMANI; JHAVERI, 2022).

Resposta a incidentes

As organizações devem ter um plano claro para resposta a incidentes implementado e testado para lidar eficazmente com violações de segurança quando elas ocorrem. O plano deve definir o que constitui um incidente para a empresa, além de fornecer um processo que deve ser seguido na ocorrência dele (SHAKED et al., 2023). A SANS (*System Administration, Networking and Security*) provê um guia, ilustrado na Figura 5, com melhores práticas para resposta a incidentes cibernéticos e dividido em 6 etapas: *(i)* preparação; *(ii)* identificação; *(iii)* contenção; *(iv)* erradicação; *(v)* recuperação; e *(vi)* lições aprendidas (KRAL, 2012).

Figura 5 – SANS Incident Response Plan



Fonte: Resultado da pesquisa

2.4.2 Linguagem de programação P4

P4 (*Programming Protocol-independent Packet Processors*) é uma linguagem específica de domínio (DSL, do inglês *Domain Specific Language*) voltada para a programação de dispositivos de rede, como *switches* e roteadores. Ela foi projetada para atender as necessidades de SDN e NFV (BOSSHART et al., 2014). Seu principal objetivo é habilitar o processamento de pacotes de forma independente dos protocolos de comunicação subjacentes. Em 2016, a especificação da linguagem P4 foi estendida para dispositivos como FPGAs, ASICs e interfaces de rede (NICs, do inglês *Network Interface Cards*) (KFOURY; CRICHIGNO; BOU-HARB, 2021). A especificação atual da linguagem P4 estabelece as seguintes abstrações, representadas na Figura 6:

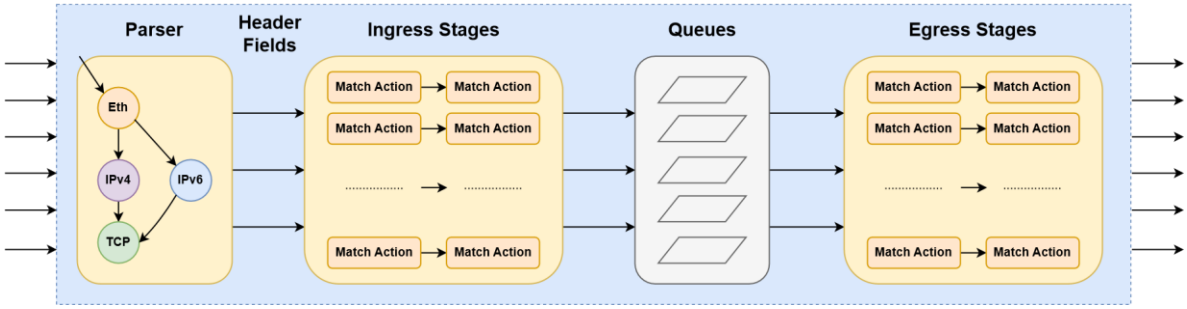
- *Headers*: definem a estrutura de cada cabeçalho em um pacote, incluindo os campos que compõem esses cabeçalhos e seus tamanhos.

- *Parsers*: especificam as sequências de cabeçalhos permitidas nos pacotes recebidos, além de indicarem quais campos devem ser extraídos dos pacotes.

- *Tables*: podem ser utilizadas para implementação de tabelas de roteamento, listas de controle de acesso, tabelas de busca e outros tipos de conjuntos de dados definidos pelo usuário.

- *Actions*: são trechos de código que determinam como os campos do cabeçalho do pacote e seus metadados associados são manipulados. Um conjunto de ações pode ser utilizado para realização de transformações específicas nos pacotes.

Figura 6 – Abstração de um *switch* P4



Fonte: Elaborado pelo autor. Adaptado de (BOSSHART et al., 2014)

Segundo (BUDIU; DODD, 2017), as principais vantagens da linguagem P4 no contexto de planos de dados programáveis são:

- Flexibilidade: a lógica de encaminhamento dos pacotes é descrita por meio de programas P4, permitindo que os desenvolvedores determinem o comportamento desejado no processamento deles.

- Independência de plataforma: programadores não precisam se preocupar com detalhes sobre o *hardware* subjacente. Logo, programas P4 podem ser executados em diferentes plataformas de *hardware*, desde que sigam as mesmas especificações de arquitetura.

- Desacoplamento: os fabricantes de *hardware* podem utilizar arquiteturas abstratas para separar componentes de baixo nível dos componentes de alto nível, permitindo que os elementos essenciais sejam projetados de forma independente e reconfigurados conforme necessário.

2.5 Machine learning

O termo *machine learning* pode ser traduzido para o português como aprendizado de máquina. Esta subárea da inteligência artificial se concentra na aplicação de algoritmos cujo objetivo é realizar previsões sobre um conjunto de dados. Modelos de *machine learning* permitem a um sistema computacional aprender e melhorar seu desempenho em tarefas específicas a partir da experiência adquirida com os dados utilizados na fase de treinamento (REBALA et al., 2019).

É importante notar que não existe um único algoritmo para resolução de problemas nessa área, sendo necessário o entendimento sobre as vantagens, desvantagens e contextos de aplicação de cada um deles. Ademais, as técnicas podem ser classificadas em aprendizado supervisionado, não-supervisionado, semi-supervisionado e por reforço (MORALES; ESCALANTE, 2022).

No aprendizado supervisionado, um conjunto de exemplos de treinamento é disponibilizado ao algoritmo. Os dados possuem o rótulo da classe associada e atributos. O objetivo é realizar previsões precisas para novos dados a partir dos exemplos utilizados no treinamento (SINGH; THAKUR; SHARMA, 2016). Já no aprendizado não-supervisionado, o algoritmo não recebe os dados rotulados, seu objetivo é identificar padrões ou estruturas nos dados (USAMA et al., 2019).

O aprendizado semi-supervisionado utiliza um conjunto de dados que contém uma combinação de dados rotulados e não rotulados. Esta abordagem é útil quando o custo para rotular dados é elevado. Em muitos cenários realistas, informações do *dataset* podem não estar rotuladas, tornando o aprendizado por reforço uma abordagem pragmática (ENGELEN; HOOS, 2020).

Já o aprendizado por reforço tem como objetivo treinar o programa para tomar decisões em busca dos melhores resultados. Para isso, um agente é treinado para tomar sequências de decisões, recebendo recompensas ou penalidades. Este método se assemelha com o aprendizado humano baseado em tentativa e erro (FRANÇOIS-LAVET et al., 2018).

As subseções a seguir exploram os algoritmos empregados neste trabalho, bem como as métricas utilizadas para avaliar os modelos gerados.

2.5.1 Árvores de decisão

As árvores de decisão são modelos que envolvem uma série de decisões em forma de uma estrutura de árvore. Cada nó interno representa uma decisão com base em um atributo e as folhas representam os resultados. Embora não sejam classificadores robustos, árvores de decisão normalmente são utilizadas como base para modelos mais sofisticados, visto que elas são computacionalmente menos complexas (CHARBUTY; ABDULAZEEZ, 2021).

2.5.2 Random Forest

O algoritmo *Random Forest* utiliza uma série de árvores de decisão para melhorar a precisão do modelo e reduzir o *overfitting* (SANTOS; PAPA, 2022), sendo um dos algoritmos mais utilizados para resolução de problemas de classificação e regressão. A partir das árvores de decisão geradas paralelamente, o algoritmo combina as suas previsões para obtenção do resultado final (LIU; WANG; ZHANG, 2012).

2.5.3 Support Vector Machine

A técnica de *Support Vector Machine* (SVM) também pode ser aplicada na resolução de problemas de classificação e regressão. Esse algoritmo de aprendizado supervisionado busca o hiperplano de separação ótima entre as classes de dados (MAMMONE; TURCHI; CRISTIANINI, 2009). No campo de *machine learning*, o SVM é considerado um dos algoritmos mais complexos, em virtude das funções de *kernel* utilizadas para mapear os dados em espaços de características de maior dimensão (PATLE; CHOUHAN, 2013).

2.5.4 Redes neurais artificiais

As redes neurais artificiais são algoritmos de *deep learning* inspirados no funcionamento do cérebro humano, imitando a maneira como os neurônios enviam sinais uns para os outros (YEGNANARAYANA, 2009). Uma rede neural é composta por diversas unidades de processamento. As unidades são conectadas por canais que estão associados a um determinado peso. A inteligência de uma rede neural vem das interações entre as unidades de processamento da rede (ZOU; HAN; SO, 2009).

Arquiteturas de redes neurais normalmente são organizadas em camadas, sendo elas: *(i)* camada de entrada; *(ii)* camadas intermediárias; e *(iii)* camada de saída. Na camada de entrada, os padrões são apresentados à rede. A maior parte do processamento se dá nas camadas intermediárias, por meio de conexões ponderadas. Por fim, o resultado é apresentado na camada de saída (GAWLIKOWSKI et al., 2023).

2.5.5 Métricas utilizadas para avaliar algoritmos de machine learning

Métricas para avaliação de modelos de *machine learning* são essenciais para medir a qualidade de um determinado modelo. Dessa forma, é possível determinar se ele se ajusta aos dados e atende aos objetivos específicos de uma tarefa (ZHOU et al., 2021). A seguir são apresentadas as métricas mais utilizadas e consideradas na avaliação dos modelos propostos neste estudo.

Matriz de confusão

Uma matriz de confusão permite visualizar de forma simples a performance de um modelo de classificação. Ela indica o número total ou porcentagem de exemplos em cada grupo: verdadeiro positivo (TP, do inglês *true positive*), verdadeiro negativo (TN, do inglês *true negative*), falso positivo (FP, do inglês *false positive*) e falso negativo (FN, do inglês *false negative*). A Tabela 1 apresenta a estrutura de uma matriz de confusão.

Tabela 1 – Matriz de confusão

Valores observados \ Valores reais	**Positivo**	**Negativo**
Positivo	TP	FP
Negativo	FN	TN

Fonte: Resultado da Pesquisa.

Acurácia

A acurácia indica a performance geral do modelo. É o número de acertos dividido pelo número total de exemplos. Por exemplo, se em um total de 100 observações, 90 delas forem classificadas corretamente, o modelo possui uma acurácia de 90%. O cálculo da acurácia é definido na Equação (1).

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (1)$$

Precisão

A precisão é calculada como a razão entre a quantidade de exemplos classificados corretamente como positivos e o total de exemplos classificados como positivos. Essa métrica dá uma ênfase maior para os erros por falsos positivos. O cálculo da precisão é exposto na Equação (2).

$$Precision = \frac{TP}{TP + FP} \quad (2)$$

Recall

O *recall*, ou revocação em português, tem como ênfase os erros por falsos negativos. Essa métrica (também conhecida como sensibilidade) é definida pela razão entre o número de exemplos classificados corretamente como positivos e a quantidade de exemplos que realmente são positivos, como exposto na Equação (3).

$$Recall = \frac{TP}{TP + FN} \quad (3)$$

F1-Score

O F1-Score é definido pela média harmônica entre a precisão e o *recall*. Um F1-Score baixo é um indicativo de que a precisão ou *recall* do modelo está baixo. Logo, essa métrica se apresenta como um indicador mais abrangente da eficácia do modelo, embora seja menos intuitiva que a acurácia. O cálculo do F1-Score é definido na Equação (4).

$$F_1 = \frac{2 * Precision * Recall}{Precision + Recall} = \frac{2 * TP}{2 * TP + FP + FN} \quad (4)$$

2.6 Análise de malware

Malwares são programas cujo objetivo é causar prejuízo a um usuário, computador ou rede. A infecção por um artefato malicioso pode ocorrer de diversas formas, sendo ataques de *phishing* e a exploração de vulnerabilidades as mais comuns (NAMANYA et al., 2018). A constante evolução dos *malwares* desenvolvidos exigiu a classificação deles com base nas suas funcionalidades. Segundo (SIKORSKI; HONIG, 2012), as principais categorias de *malware* são:

- *Backdoor*: consiste em um código malicioso que se instala em um computador para manter o acesso do atacante. Um *backdoor* permite que o atacante se conecte à vítima de forma direta para a execução de comandos.
- *Botnet*: de forma similar a um *backdoor*, essa ameaça também permite o acesso ao dispositivo infectado. Entretanto, uma estação de comando e controle é utilizada para enviar o mesmo comando a todos os dispositivos infectados, normalmente para execução de ataques de negação de serviço.

- *Downloader*: o principal objetivo dessa categoria de *malware* é baixar outros *malwares*. Comumente são executados após o acesso inicial do atacante, visando instalar artefatos maliciosos mais sofisticados sem chamar a atenção de antivírus.
- *Information-stealing malware*: artefatos maliciosos cujo objetivo é coletar informações da vítima e enviá-las ao atacante. Exemplos de *malwares* dessa categoria são *sniffers* e *keyloggers*.
- *Launcher*: programas maliciosos que fazem uso de técnicas não usuais para a execução de outros *malwares*, visando evitar a detecção por sistemas de antivírus.
- *Rootkit*: são *malwares* especializados em ocultar a sua presença e projetados para dar acesso e controle a um dispositivo. *Rootkits* podem operar dentro do próprio sistema operacional, o que os permite executar comandos com privilégios elevados. Além disso, essa característica dificulta a sua detecção por *softwares* de antivírus.
- *Scareware*: são projetados para induzir um usuário a baixar ou comprar programas falsos ou maliciosos.
- *Spam-sending malware*: após a infecção, utilizam o dispositivo infectado para o envio de *spam*.
- *Worm*: código malicioso projetado para infectar outros dispositivos na rede de forma automática.

Recentemente, *malwares* do tipo *ransomware* têm ganhado atenção da indústria e academia (OZ et al., 2022). Essas ameaças têm como principal objetivo causar a indisponibilidade das informações dos usuários por meio da encriptação dos arquivos. Após isso, é exigido um pagamento (normalmente em criptomoedas) para o resgate dos dados. *Ransomwares* podem ser classificados em duas categorias: de bloqueio e de criptografia. Na primeira, as funções básicas do dispositivo são afetadas; já na segunda, a mais comum, os arquivos do usuário são criptografados.

A análise de *malware* é de fundamental importância para a segurança da informação; pois, a partir do entendimento comportamental e do código Assembly do artefato, são

produzidas ferramentas para detecção e mitigação eficientes. Porém, o tempo que um analista leva para dissecar um binário malicioso e implantar regras na solução desenvolvida pode ser desfavorável ao usuário final.

Como especificado por (UCCI; ANIELLO; BALDONI, 2019), a análise de *malware* demanda, em geral, que a solução adotada tenha uma boa capacidade de classificar novos arquivos a partir de investigações anteriores. Os autores destacam a diversidade de ferramentas disponíveis e que a escolha errada de uma pode atrasar o trabalho de análise. Além disso, a própria tarefa de análise pode ter propósitos distintos: enquanto um analista pode estar interessado apenas em identificar se um artefato é malicioso ou não, outro analista pode se aprofundar na análise e descobrir a qual família o *malware* pertence.

A análise de um programa malicioso pode ser realizada de forma estática ou dinâmica. É importante notar que uma estratégia não substitui a outra e ambas devem ser empregadas para um completo entendimento sobre o *malware* em questão.

2.6.1 Análise estática

A análise estática é comumente a primeira etapa do processo, na qual o analista inspeciona o artefato em detalhes, mas sem executá-lo. Ela pode ser dividida entre básica e avançada.

A análise estática básica pode confirmar se um arquivo é malicioso ou não, além de prover informações sobre a sua funcionalidade e dados que podem ser utilizados para geração de assinaturas. Nesta fase é identificado o tipo de arquivo, seus *hashes*, cabeçalhos, *strings* legíveis, importações de funções em bibliotecas compartilhadas, utilização de *packers*, entre outras características (MOSER; KRUEGEL; KIRDA, 2007). Soluções como o PEStudio[3], DIE[4] e Freki (SOUZA; SILVA, 2021) podem ser utilizadas para uma avaliação inicial do binário. Entretanto, a análise básica é ineficaz no caso de ameaças sofisticadas.

[3] https://www.winitor.com

[4] https://github.com/horsicq/Detect-It-Easy

Já a análise estática avançada consiste na engenharia reversa do *malware*, visando entender o seu comportamento. Isso pode ser obtido a partir da análise das suas instruções em um *disassembler*. É importante destacar que esse tipo de análise exige um conhecimento aprofundado do código Assembly específico da arquitetura para qual o binário foi compilado, bem como do sistema operacional alvo do *malware*. Nessa categoria, ferramentas como o IDA[5], gdb[6] e radare2[7] são amplamente utilizadas.

2.6.2 Análise dinâmica

Diferentemente da análise estática, esta etapa consiste em executar o artefato malicioso em um ambiente controlado, conhecido como *sandbox*, visando monitorar o seu comportamento. A análise dinâmica também é dividida entre básica e avançada.

Na análise dinâmica básica, o analista identifica chamadas de sistema realizadas, processos em execução, consultas DNS, tráfego direcionado a serviços conhecidos (como HTTP, FTP e SMTP), arquivos criados no sistema de arquivos, entre outros eventos (KAO; HSIAO, 2018). Um ambiente de análise amplamente utilizado pela academia e indústria é o Cuckoo Sandbox (JAMALPUR et al., 2018). Porém, assim como na análise estática básica, detalhes sobre o funcionamento do *malware* podem não ser identificados pelo analista.

A análise dinâmica avançada permite a extração de informações detalhadas a respeito do programa malicioso. Nesta etapa é utilizado um *debugger*, como o x64dbg[8], para examinar as instruções do binário. Dessa forma, é possível executar suas operações passo a passo, bem como visualizar as informações utilizadas e armazenadas pelo *malware* em memória.

[5] https://hex-rays.com/ida-free/

[6] https://www.sourceware.org/gdb/

[7] https://rada.re/n/

[8] https://x64dbg.com

2.7 Detecção de malware

Diversas técnicas foram aprimoradas ao longo dos anos para identificação e contenção automatizada de artefatos maliciosos. Além disso, novas formas de identificação a partir de estudos relacionados ao processamento de imagens e visão computacional vêm sendo aprimoradas pela academia (KOSMIDIS; KALLONIATIS, 2017). As subseções a seguir exploram as principais técnicas para detecção de *malware* sem a intervenção humana.

2.7.1 Detecção baseada em assinaturas

Essa técnica se caracteriza por ser muito rápida na identificação de ameaças conhecidas. Para isso, uma base de dados é constantemente atualizada para conter assinaturas estáticas (conhecidas como *fingerprints*) previamente identificadas. A assinatura se prova mais vantajosa que a mera utilização de *hashes*, pois pode ser utilizada para rastrear diferentes *malwares* pertencentes à mesma família (SATHYANARAYAN; KOHLI; BRUHA-DESHWAR, 2008).

Quando um fabricante de soluções *anti-malware* classifica um objeto como malicioso, sua assinatura é adicionada a um banco de dados, que pode conter milhões de assinaturas para identificação de *malwares*. A solução instalada na máquina do usuário final consulta essa base de dados para classificação dos arquivos baixados.

Apesar de simples, esse método é muito utilizado por soluções comerciais de antivírus como uma primeira linha de defesa. O formato mais popular para criação de assinaturas é o YARA (LOCKETT, 2021), que permite aos analistas a definição delas com base em padrões textuais e binários.

2.7.2 Detecção baseada em heurística

Atacantes geralmente utilizam técnicas de evasão (como encriptação, *packing*, ofuscação e/ou polimorfismo) para evasão de defesas que se baseiam em assinaturas. Com o

objetivo de mitigar ameaças mais sofisticadas, antivírus modernos também empregam técnicas de detecção baseadas em heurística.

A principal vantagem desse tipo de detecção é não necessitar de uma correspondência exata de *bytes* no binário analisado para classificá-lo como malicioso (TREADWELL; ZHOU, 2009). Logo, é possível detectar novos *malwares* de maneira genérica a partir da análise estática automatizada do arquivo. Para isso, regras de classificação são construídas por analistas para determinar se um artefato é malicioso ou não.

Como especificado por (EGELE et al., 2008), as regras devem ser genéricas o suficiente para mitigar variantes de uma determinada ameaça, além de não classificar erroneamente um arquivo benigno como malicioso. Entretanto, o advento de *toolkits* como o Zeus [9] tem permitido que atacantes realizem a mutação de milhares de *malwares* diariamente, tornando as detecções por assinatura e por heurística ineficazes em muitos casos se utilizadas isoladamente.

2.7.3 Detecção baseada em comportamentos

O constante surgimento de novas famílias de *malware* demandou o desenvolvimento de soluções que não se baseiam puramente na análise estática de arquivos binários. Isso se deve ao fato de que as assinaturas só podem ser geradas e catalogadas após um exemplar de *malware* ter sido analisado.

Diante disso, a análise comportamental do artefato permite que ações suspeitas sejam identificadas em tempo real. Para isso, as soluções baseadas nessa técnica monitoram o artefato em execução, visando identificar comportamentos potencialmente perigosos, como descompactar códigos maliciosos, modificações em arquivos críticos do sistema ou tentativas de interceptar as teclas pressionadas pelo usuário. A partir desse monitoramento, soluções de antivírus são capazes de detectar ameaças não catalogadas previamente (PENG et al., 2013).

[9] https://www.kaspersky.com/resource-center/threats/zeus-virus

Vale notar que a presença de apenas uma das ações supracitadas pode não ser suficiente para classificar um programa como malicioso. Portanto, é necessário avaliar o conjunto de ações executadas pelo artefato visando reduzir a taxa de falsos positivos.

2.7.4 Detecção em nuvem

Essa abordagem é caracterizada pelo envio do artefato a ser analisado a um ambiente controlado na infraestrutura do provedor da solução de antivírus. A utilização da detecção via nuvem tem crescido devido à alta quantidade de camadas existentes em soluções tradicionais de antivírus. Tendo em vista que essas soluções mesclam as abordagens previamente abordadas, tal redundância pode causar um pequeno atraso na execução de programas benignos, degradando a experiência do usuário.

Para contornar o problema mencionado, é utilizado um agente leve na máquina do cliente. Esse agente é responsável por coletar informações relevantes sobre o artefato a ser analisado e as envia para o provedor de antivírus. Dessa forma, soluções baseadas em nuvem consomem menos recursos do sistema no qual o agente está instalado. Programas baixados mas não executados podem ser analisados assincronamente e sem a intervenção do usuário, mitigando ameaças antecipadamente (ASLAN; OZKAN-OKAY; GUPTA, 2021). Além disso, a simplicidade do agente instalado na máquina do usuário reduz a superfície de ataque contra ele.

Um ponto negativo dessa abordagem é que máquinas sem conectividade possuirão capacidades limitadas de detecção. Ademais, os provedores de antivírus podem não deixar claro quais dados estão sendo enviados para a nuvem e quais são os procedimentos para eliminação deles após o período de análise.

2.7.5 YARA

YARA é uma ferramenta de código aberto e multiplataforma desenvolvida com o objetivo de auxiliar pesquisadores e ferramentas de segurança na identificação e classificação de artefatos maliciosos (NAIK et al., 2021). Seu funcionamento é baseado na análise de padrões

textuais ou binários. Cada regra consiste em um conjunto de *strings* e expressões booleanas que determinam a sua lógica. É possível utilizar a ferramenta por meio de interfaces de linha de comando ou via *scripts* escritos em Python.

Regras YARA são compostas por duas seções: *(i)* definições de *strings*; e *(ii)* condições. A primeira contém as *strings* que farão parte da regra. Cada *string* pode ser definida em formato textual ou hexadecimal e possui um identificador, que é utilizado para se referir a ela no restante da regra. A seção de *strings* pode ser omitida se a regra não depender de nenhuma *string*. A segunda seção contém as condições que efetivamente classificarão o artefato analisado.

O Algoritmo 1 descreve uma regra que pode ser utilizada para detecção do *ransomware* Nefilim (BEAMAN et al., 2021). Nela, a *string* "NEFILIM-DECRYPT.txt" é buscada nos *bytes* do arquivo. Além disso, também é verificado se o arquivo é um executável que faz uso da função CryptEncrypt, presente na DLL ADVAPI32. Caso as duas condições sejam atendidas, o arquivo é classificado como malicioso.

Algoritmo 1 Exemplo de regra YARA para detecção do *ransomware* Nefilim

```
1: import "pe"
2:
3: rule nefilim { 4:
        strings:
5: $ransomnote = "NEFILIM-DECRYPT.txt"wide 6: condition:
7:              $ransomnote and pe.imports("ADVAPI32.dll", "CryptEncrypt")
8: }
```

2.8 Trabalhos relacionados

A detecção de *malwares* em ambientes IoT é de fundamental importância para ambientes corporativos e domésticos, visto que o comprometimento de tais dispositivos pode acarretar em danos físicos e na exposição de dados sensíveis. Diante disso, esta seção aborda os principais esforços de pesquisa propostos pela comunidade científica no âmbito da identificação de atividades maliciosas em ambientes IoT por meio de *machine learning*.

Em (WOO; KIM; CHUNG, 2017), os autores propõem um mecanismo para detecção de artefatos maliciosos que se baseia na frequência de *opcodes* (códigos de operação) em arquivos executáveis. Com o objetivo de aprimorar a segurança das redes, é proposta a utilização de uma *SDN Quarantined Network* (SQN), que possui diferentes módulos de detecção. O algoritmo *Classification and Regression Tree* (CART) é empregado para detecção de binários maliciosos. Para treinar o modelo, os autores utilizaram um *dataset* de 270.000 arquivos maliciosos e 10.000 artefatos benignos. Os experimentos evidenciam que a solução obteve uma acurácia de 94.3% na detecção de *malwares*. Entretanto, a ferramenta se limita à detecção de arquivos do tipo *Portable Executable* (PE). Ademais, a análise de *opcodes* pode não ser eficaz caso o *malware* tente mascarar o seu comportamento por meio da ofuscação deles. Por fim, os autores não avaliaram possíveis impactos do mecanismo na performance da rede.

O estudo de (LETTERI; PENNA; GASPERIS, 2018) se concentra na aplicação de técnicas de *deep learning* para identificar *botnets* em ambientes de infraestrutura baseada em SDN. Por meio da análise do tráfego de rede, a abordagem proposta utiliza uma rede neural do tipo perceptron multicamadas (MLP, do inglês *multi-layer perceptron*) para detectar padrões maliciosos. O conjunto de dados HogZilla[10], uma combinação de partes dos conjuntos de dados CTU-13 (GARCIA et al., 2014) e ISCX 2012 IDS[11], foi empregado para treinar o modelo. Os autores destacam que o modelo alcança uma taxa de acurácia de 96%. No entanto, é importante ressaltar que a eficácia dessa ferramenta pode ser afetada pelo tamanho da rede e pela quantidade de tráfego a ser monitorada, especialmente em redes muito grandes ou com um volume de tráfego elevado, o que pode requerer recursos computacionais substanciais para a detecção eficiente de *botnets*.

A solução apresentada por (CUSACK; MICHEL; KELLER, 2018) combina SDN com aprendizado de máquina para a detecção de *malwares* do tipo *ransomware*. Os autores oferecem uma estratégia baseada em *Programmable Forwarding Engines* (PFEs), que

[10] https://ids-hogzilla.org/dataset/

[11] https://www.unb.ca/cic/datasets/ids.html

habilitam a coleta de dados de cada pacote da rede de forma individual e eficaz. Para isso, são extraídas características de alto nível do tráfego gerado por dispositivos infectados (como o total de *bytes* trocados entre a máquina infectada e o servidor de comando e controle). Com base nessas características, o algoritmo *Random Forest* é utilizado para classificação do tráfego. O modelo foi treinado com 100MB de tráfego gerado por *ransomwares* e 100MB de tráfego legítimo, alcançando uma acurácia de 87%. Embora seja mencionado que a abordagem pode ser generalizada para a detecção de diferentes famílias de *ransomware*, os experimentos tiveram como foco a detecção específica do *ransomware* Cerber[12]. Além disso, o conjunto de dados de treinamento é limitado (200MB) e considerações a respeito da latência imposta pela ferramenta não são apresentadas.

O trabalho de (MAEDA et al., 2019) propõe um mecanismo baseado em *deep learning* para detecção de tráfego malicioso originado por máquinas infectadas por *botnets* em redes SDN. Para isso, os autores empregam uma rede neural do tipo MLP. Após a classificação, o mecanismo isola as máquinas infectadas e bloqueia conexões externas. O modelo foi treinado utilizando tráfego malicioso do *dataset* CTU-13, bem como tráfego legítimo do *dataset* ISOT[13]. Os autores destacam que a acurácia da solução é de 99.2%. Entretanto, o artigo destaca que não foram conduzidos experimentos em terminais que estavam efetivamente infectados com *bots*, o que pode afetar a generalização dos resultados.

Em (KHAN; AKHUNZADA, 2021), os autores oferecem um mecanismo híbrido para detecção de *malwares* que exploram vulnerabilidades em equipamentos IoT, mais especificamente em aparelhos *Internet of Medical Things* (IoMT). A solução apresentada combina *deep learning* para classificação e detecção de *malwares* e utiliza a arquitetura SDN para facilitar a aquisição e coleta do tráfego da rede. A principal contribuição deste trabalho está no aumento da eficácia na detecção de *malwares* devido ao uso de dois modelos de classificação: *Convolution Neural Network* (CNN) e *Long Short Term Memory* (LSTM). Os autores destacam que esse esquema apresenta melhores resultados quando comparado a

[12] https://www.malwarebytes.com/blog/detections/ransom-cerber

[13] https://onlineacademiccommunity.uvic.ca/isot/2022/11/27/botnet-and-ransomware-detectiondatasets/

outras estratégias de classificação. O *dataset* utilizado para treinar o modelo foi composto por 128 amostras maliciosas e 1.089 artefatos benignos, que alcançou uma acurácia de 99.83%. Embora os autores destaquem que a solução proposta tenha uma baixa complexidade, nenhuma avaliação do *overhead* causado na rede é realizada.

O estudo de (MUTHANNA et al., 2022) propõe um mecanismo para detecção de intrusões em ambientes IoT via *deep learning*. A ferramenta proposta analisa o tráfego da rede e utiliza um classificador denominado *Cuda Long Short Term Memory Gated Recurrent Unit* (cuLSTMGRU), que consiste em uma variante do algoritmo LSTM, para identificar atividades maliciosas. O modelo é treinado e avaliado utilizando o *dataset* CI-CIDS2017 (SHARAFALDIN; LASHKARI; GHORBANI, 2018), alcançando uma acurácia de 99.23%. Como trabalho futuro, os autores pretendem utilizar uma *blockchain* para melhorar a eficiência do sistema de detecção.

O trabalho de (CHANG et al., 2022) apresenta uma ferramenta para detecção de *malwares* por meio de *switches* programáveis e *deep learning*. Para isso, uma *Convolutional Neural Network* (CNN) é empregada para classificação de tráfego. Os autores afirmam que o modelo, treinado com o *dataset* IoT-23 (GARCIA; PARMISANO; ERQUIAGA, 2020), obteve uma acurácia de 99%, acompanhada por um tempo de processamento significativamente reduzido quando comparado com ferramentas tradicionais de detecção de intrusões. Entretanto, é importante observar que a solução inspeciona apenas os primeiros 40 *bytes* do *payload* de cada pacote, o que pode limitar sua eficácia, especialmente no caso de *malwares* que empregam técnicas de evasão, como a fragmentação do *payload* em múltiplos pacotes ou *junk code*. Ademais, as características utilizadas para a classificação não são especificadas no estudo.

Chaganti et al. (2023) define uma abordagem para detecção de intrusões em ambientes IoT com base na análise e classificação de tráfego. Os autores fazem uso do classificador LSTM para detecção de atividades suspeitas. O modelo foi treinado utilizando os *datasets* SDNIoT (SARICA; ANGIN, 2020) e SDN-NF-TJ (JAFARIAN, 2019), alcançando uma acurácia de 97.1%. Entretanto, a degradação imposta pela ferramenta na performance da rede não foi considerada durante as avaliações realizadas. Como trabalho futuro, os autores pretendem validar se o modelo proposto é capaz de resistir a ataques adversariais.

A Tabela 2 sumariza a análise dos trabalhos relacionados em termos de: *(i)* problema endereçado; *(ii)* estratégia utilizada; *(iii)* classificador; *(iv) dataset* utilizado para treinamento; e *(v)* acurácia de detecção. A proposta mais similar ao presente trabalho é a desenvolvida por (CHANG et al., 2022). Entretanto, a solução aqui proposta utiliza mais características do tráfego para identificação de *malwares*, melhorando a sua confiabilidade e diminuindo o número de falsos positivos. Outra distinção é a utilização de uma abordagem híbrida, composta por um módulo YARA responsável pela detecção rápida de padrões maliciosos e um classificador *Random Forest* para identificação de ameaças desconhecidas.

Além disso, a ferramenta é capaz de isolar as máquinas infectadas do restante da rede, atuando como um mecanismo de contenção para evitar a propagação de ameaças. Por fim, o Heimdall é avaliado considerando amostras de *malware* reais, que não estão presentes no conjunto de dados de treinamento ou de teste do modelo de *machine learning*.

Tabela 2 – Resumo dos trabalhos relacionados

Proposta	**Problema**	**Estratégia**	**Classificador**	**Dataset**	**Acurácia**
(WOO; KIM; CHUNG, 2017)	Detecção de arquivos PE maliciosos	Análise de tráfego	CART	Indisponível	94.3%
(LETTERI; PENNA; GAS-PERIS, 2018)	Detecção de *botnets* em redes SDN	Análise de tráfego	MLP	HogZilla	96%
(CUSACK; MICHEL; KELLER, 2018)	Detecção de *ransomwares* em nível de rede	Análise de tráfego	Random Forest	Indisponível	87%
(MAEDA et al., 2019)	Detecção e isolamento de máquinas infectadas por *botnets*	Análise de tráfego	MLP	CTU-13 e ISOT	99.2%
(KHAN; AKHUNZADA, 2021)	Detecção de *malwares* em ambientes IoMT	Análise de tráfego	CNN e LSTM	Indisponível	99.83%
(MUTHANNA et al., 2022)	Detecção de intrusões em ambientes IoT	Análise de tráfego	Cu-LSTM-GRU	CICIDS2017	99.23%
(CHANG et al., 2022)	Detecção de *malwares* em ambientes IoT	Análise de tráfego	CNN	IoT-23	99%
(CHAGANTI et al., 2023)	Detecção de intrusões em ambientes IoT	Análise de tráfego	LSTM	SDNIoT e SDNNF-TJ	97.1%

Fonte: Resultado da Pesquisa.

3 PROPOSTA

Este capítulo se dedica a discutir a proposta principal deste trabalho. A arquitetura do Heimdall tem como objetivo permitir a rápida detecção de artefatos maliciosos diretamente nos *switches* da rede por meio de uma abordagem híbrida, composta por assinaturas predefinidas e *machine learning*. A solução também deve permitir o isolamento dos dispositivos comprometidos e prover uma API para fácil gerenciamento. As seções a seguir exploram em detalhes os requisitos da solução, sua arquitetura funcional e o seu fluxo de trabalho.

3.1 Requisitos funcionais

A ferramenta proposta visa atender aos seguintes requisitos funcionais:

1. Realizar a coleta de tráfego para análise de forma agnóstica, permitindo que a solução atue independentemente dos modelos de dispositivos conectados na rede. Em cenários IoT, compostos por dispositivos que nem sempre possuem recursos de segurança embarcados, o uso de uma solução em nível de rede é essencial para mitigar ameaças de forma genérica, fornecendo uma camada adicional de proteção para tais dispositivos.

2. Identificar padrões de *malwares* na rede por meio de assinaturas maliciosas. Dessa forma, é possível detectar *malwares* que estão sendo transferidos de um dispositivo para outro. Além de rápida, essa abordagem possibilita a identificação de variantes de um artefato. Novas regras podem ser atualizadas e distribuídas aos *switches* da rede à medida que novas ameaças são identificadas e catalogadas.

3. Realizar a classificação de fluxos maliciosos por meio de *machine learning*. Isso permite identificar dispositivos infectados de forma agnóstica, fortalecendo a capacidade de resposta a ameaças não catalogadas por assinaturas.

4. Isolar dispositivos infectados do restante da rede, visando evitar a propagação de ameaças para outros dispositivos conectados. Essa medida também evita que o dispositivo comprometido envie informações sensíveis ao atacante.

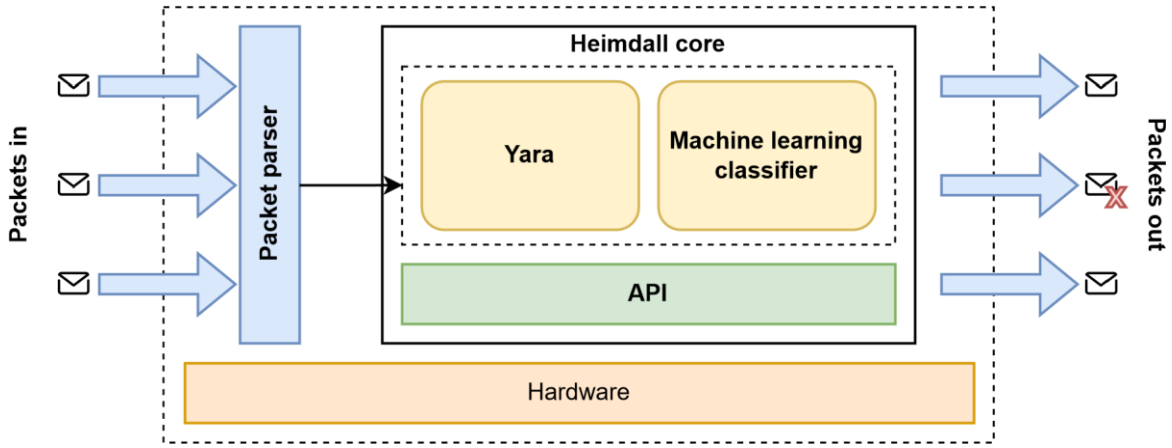
5. Prover uma API para gerenciamento e monitoramento da ferramenta. Dessa forma, é possível visualizar detalhes sobre os artefatos maliciosos detectados pelos *switches* da rede, bem como extraí-los para análises posteriores.

3.2 Arquitetura funcional do Heimdall

A arquitetura do Heimdall tem como requisito a detecção agnóstica de artefatos maliciosos em ambientes IoT. Para se atingir esse objetivo, a solução é acoplada diretamente aos comutadores presentes na rede. Isso permite que a identificação de tráfego malicioso seja realizada independentemente das características específicas dos dispositivos conectados à infraestrutura. É importante notar que essa abordagem também reduz a sobrecarga no controlador da rede e evita pontos únicos de falha.

O procedimento adotado pela solução consiste na análise do tráfego da rede. Durante essa análise, são extraídos dados que são então comparados com assinaturas de artefatos maliciosos específicos para dispositivos IoT e classificados via *machine learning*. Como exposto na Figura 7, os pacotes que chegam ao *switch* são pré-processados e enviados para detecção baseada em regras YARA. Essa abordagem possibilita a rápida detecção de artefatos que estão trafegando na rede. Em caso negativo, o módulo de classificação via *machine learning* é acionado, o que permite a identificação eficaz de ameaças desconhecidas ou que não possuem regras definidas.

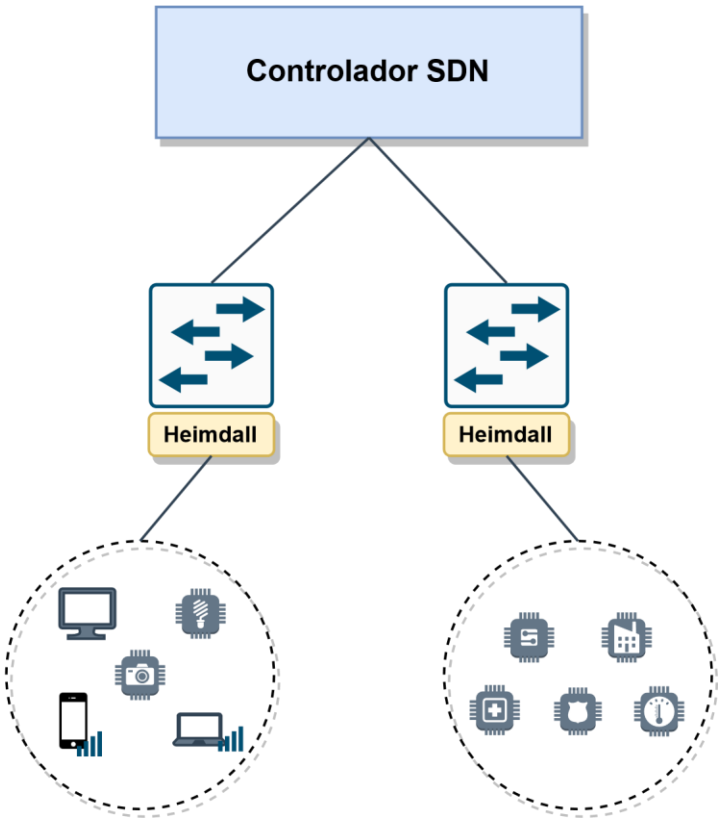
Figura 7 – Blocos funcionais do Heimdall



Fonte: Resultado da pesquisa

A Figura 8 apresenta um cenário de infraestrutura SDN após a implantação da ferramenta. Por meio da flexibilidade e programabilidade dos comutadores habilitados pela DSL P4, o Heimdall é capaz de prontamente bloquear os fluxos maliciosos diretamente no *switch* e isolar os dispositivos infectados. Ademais, a ferramenta provê uma API para que os operadores da rede consigam gerenciá-la via interface gráfica ou linha de comando, o que permite obter uma visão geral do estado da rede, artefatos maliciosos detectados e dispositivos infectados, além de possibilitar a inclusão ou remoção manual de *hosts* à lista de isolamentos. As subseções a seguir descrevem em detalhes os componentes que constituem a arquitetura proposta.

Figura 8 – Cenário de infraestrutura habilitada pelo Heimdall

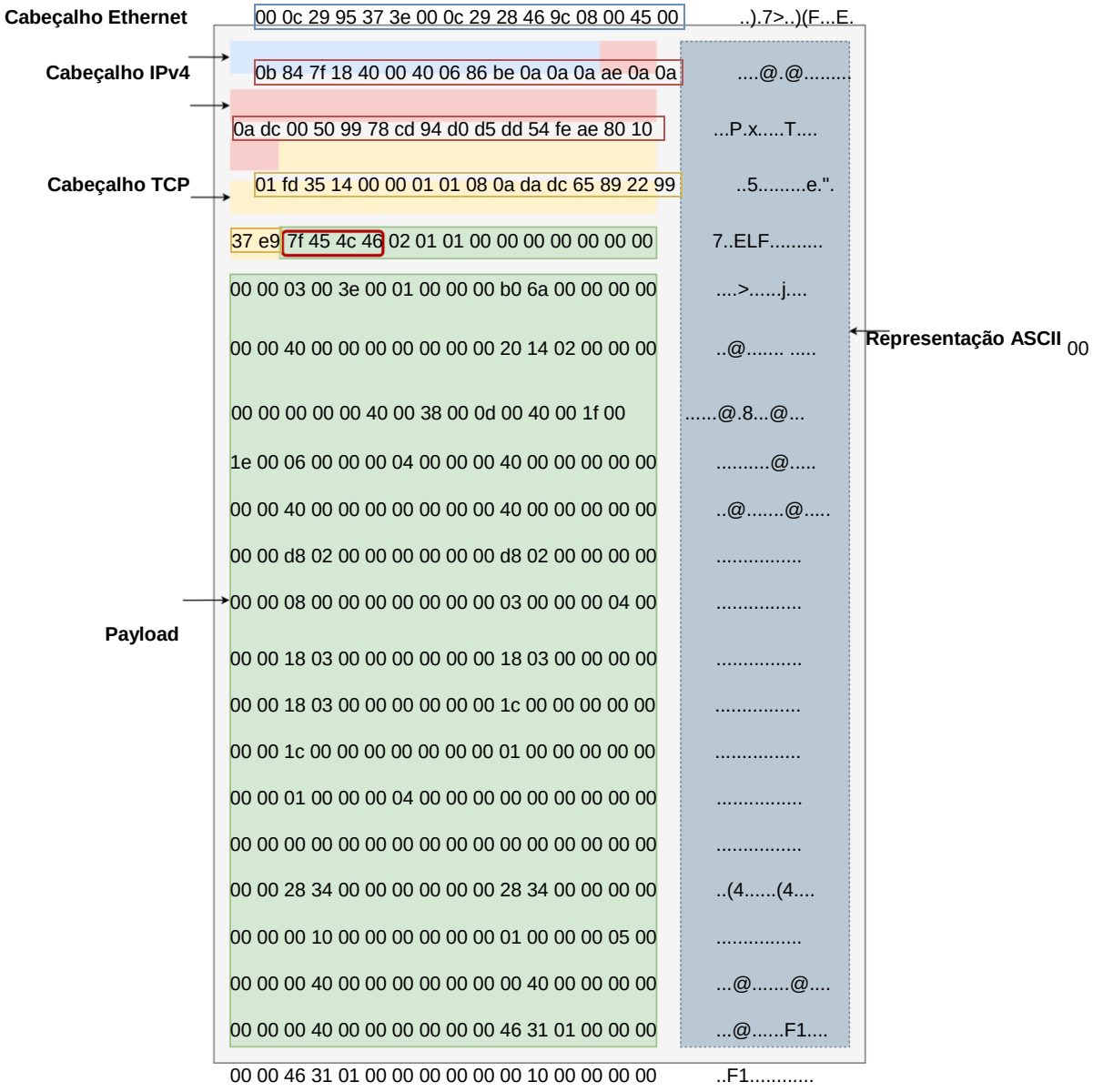


Fonte: Resultado da pesquisa

3.2.1 Packet parser

Este componente é responsável por tratar os pacotes antes do processamento por parte do núcleo da ferramenta. O módulo atua similarmente a um *sniffer* de rede, o que possibilita a visualização completa das informações presentes em cada pacote. O principal objetivo é extrair as informações relevantes para a identificação por assinaturas e classificação via *machine learning*. A Figura 9 apresenta uma visualização dos *bytes* de um pacote com conteúdo malicioso, em destaque é possível notar que se trata da transferência de um arquivo binário do tipo *Executable and Linkable Format* (ELF), caracterizado pela assinatura 0x7F454C46.

Figura 9 – Visualização dos *bytes* de um pacote



Fonte: Resultado da pesquisa

Além das informações de origem e destino da comunicação, é extraído o *payload* do pacote. Todo o seu conteúdo é considerado durante a análise, visto que características maliciosas específicas podem estar presentes em diferentes seções do binário, especialmente caso o arquivo malicioso utilize técnicas de evasão.

3.2.2 Heimdall core

O núcleo (*core*) do Heimdall é composto pelos principais componentes da arquitetura, responsáveis por efetivamente identificar atividades maliciosas na rede, além de prover uma API para gerenciamento remoto da ferramenta. A seguir, cada componente é descrito em detalhes.

Yara

As regras do Heimdall são definidas via YARA, uma solução amplamente utilizada por ferramentas de segurança. Isso possibilita a reutilização e adaptação de definições existentes, além de permitir que a ferramenta seja ajustada para detectar padrões de *malwares* com foco em diferentes sistemas operacionais.

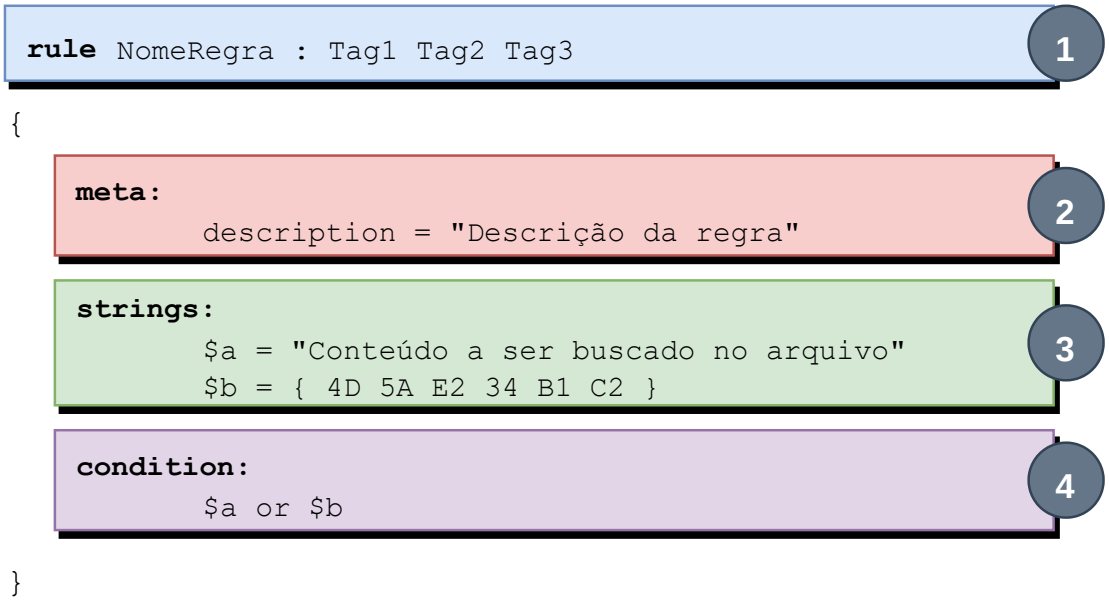
A Figura 10 apresenta a estrutura básica de uma regra para inserção na ferramenta, que é composta por um cabeçalho (1), uma meta descrição (2), uma seção de *strings* (3) e condições (4). O cabeçalho contém o nome da regra a ser definida e *tags* opcionais, que podem ser utilizadas para aplicação de filtros. A seção de meta descrição define informações úteis para o entendimento e documentação da regra. Já a seção de *strings* contém as sequências textuais e hexadecimais que serão buscadas nos *bytes* analisados. Por fim, a seção de condições estabelece quais requisitos a solução deve considerar para categorização do conteúdo analisado. É importante notar que, a depender da família de *malware* catalogada, módulos extras podem ser carregados para melhorar a precisão da regra, como os módulos PE (*Portable Executable*) e ELF (*Executable and Linkable Format*).

Machine learning classifier

O módulo de classificação via *machine learning* é responsável por analisar pacotes que não puderam ser identificados por meio das regras definidas. Para isso, um modelo de *machine learning* é treinado a partir de um *dataset* robusto, visando obter uma alta taxa de acertos.

Dessa forma, é possível mitigar ameaças que ainda não tenham sido catalogadas via regras YARA.

Figura 10 – Estrutura de uma regra YARA



Fonte: Resultado da pesquisa

As Seções 4.2 e 4.3 expõem em detalhes o *dataset* utilizado para treinamento e os algoritmos implementados neste estudo. É importante destacar que são realizados tratamentos na base de dados utilizada para eliminar informações não relevantes para o processo de treinamento. Para o desenvolvimento da prova de conceito, o algoritmo *Random Forest* é utilizado, visto que ele obteve a maior acurácia entre os modelos treinados.

API

A API do Heimdall tem a função de prover recursos para o gerenciamento remoto da solução. Para isso, é utilizado o padrão REST (*Representational State Transfer*). Dessa forma, é possível realizar consultas a respeito do estado da ferramenta, além de gerenciá-la de forma remota a partir de qualquer mecanismo capaz de interagir com o HTTP. Vale destacar que as informações são retornadas pela API em formato JSON, o que facilita o tratamento dos dados e possibilita utilizá-los como entrada para outras soluções de segurança presentes na rede, como sistemas de IDS e IPS.

A seguir são expostas as rotas de API presentes na ferramenta. Cada rota necessita de um método, que depende do tipo de operação a ser realizada, e parâmetros. Para que seja possível realizar consultas, cada *switch* possui uma chave única que deve ser especificada para autenticar o administrador.

- PUT /rule/add/{name}: adiciona uma regra YARA à base de assinaturas do Heimdall. A regra deve seguir a estrutura especificada na Figura 10 e possuir um nome único que identifica a família do *malware*.
- POST /rule/update/{name}: atualiza a regra especificada na base de dados da ferramenta.
- DELETE /rule/delete/{name}: deleta permanentemente uma regra.
- POST /host/block/{mac}: adiciona manualmente o endereço MAC especificado à lista de dispositivos isolados do restante da rede.
- POST /host/unblock/{mac}: remove manualmente o dispositivo especificado da lista de dispositivos isolados.
- POST /host/whitelist/{mac}: adiciona manualmente o endereço MAC especificado a uma *whitelist*. Todos os *hosts* presentes nessa lista não serão bloqueados mesmo se detecções positivas forem identificadas.
- PUT /whitelist/add/{hash}: adiciona um *hash* a uma *whitelist*. Dessa forma, mesmo caso o arquivo seja detectado como malicioso, nenhuma ação será tomada pela ferramenta.
- DELETE /whitelist/delete/{hash}: remove um *hash* da *whitelist*.
- GET /show/rules: retorna, em formato JSON, todas as regras YARA atualmente em uso pelo Heimdall.
- GET /show/infected: retorna, em formato JSON, informações detalhadas sobre os dispositivos infectados e presentes na lista de isolamento.
- GET /show/hashes: retorna, em formato JSON, uma lista de todos os *hashes* dos artefatos maliciosos detectados até o momento da consulta.

- GET /download/{hash}: permite realizar o *download* de um artefato a partir do seu *hash*.

A seguir é apresentado um exemplo de consulta à API do Heimdall para exposição dos dispositivos infectados. Como resultado, a ferramenta retorna detalhes sobre tais dispositivos, incluindo endereços IP, MACs, famílias das ameaças e horários de bloqueio.

Listagem 3.1 – Exemplo de consulta aos dispositivos infectados

```
$ curl --request GET --url https://10.10.10.42/api/show/infected \
--header 'x-apikey: 0c1e500ff6ea60d597ad4db206967b25'
[
    {
        "switch_uuid" : "3c26037a-3105-4ebd-a2f7-8f729d8a5461",
        "ip_addr" : "10.10.10.50",
        "mac_addr" : "B8:27:EB:45:AB:4F",
        "tag" : "Mirai",
        "time" : "2023-12-03 18:02:24"
    },
    {
        "switch_uuid" : "3c26037a-3105-4ebd-a2f7-8f729d8a5461",
        "ip_addr" : "10.10.10.51",
        "mac_addr" : "A0:AC:69:12:AD:A7",
        "tag" : "Mirai",
        "time" : "2023-12-03 20:51:12"
    }
]

$ curl --request GET --url https://10.10.10.43/api/show/infected \
--header 'x-apikey: 03a2232c772d656fec6bf71c9d65711b'
[
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
        "ip_addr" : "10.10.10.52",
        "mac_addr" : "B8:27:EB:D6:E4:C5",
        "tag" : "Mirai",
        "time" : "2023-12-04 11:35:48"
    },
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
        "ip_addr" : "10.10.10.53",
        "mac_addr" : "B8:27:EB:DC:83:DC",
        "tag" : "Unknown",
        "time" : "2023-12-06 19:15:36"
    },
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
```

```
        "ip_addr" : "10.10.10.54",
        "mac_addr" : "A0:AC:69:C6:B4:B5",
        "tag" : "Gafgyt",
        "time" : "2023-12-07 09:13:42"
    }
]
```


Já o exemplo abaixo apresenta um exemplo de consulta à API da ferramenta que retorna uma lista com os *hashes* dos artefatos detectados. Além dos *hashes* obtidos a partir das funções MD5, SHA-1 e SHA-256, a identificação do *switch* e da família do *malware* são retornadas.

Listagem 3.2 – Exemplo de consulta aos *hashes* de artefatos identificados

```
$ curl --request GET --url https://10.10.10.42/api/show/hashes \
--header 'x-apikey: 0c1e500ff6ea60d597ad4db206967b25'
[
    {
        "switch_uuid" : "3c26037a-3105-4ebd-a2f7-8f729d8a5461",
        "tag" : "Mirai",
        "md5" : "a6d1ac6b2f364761d673dded54914525",
        "sha1" : "d43030e3b49efef9382163a39f02448513eca853",
        "sha256" :
"2a9420432783500097b2c1da1a04ece0c73d7dfb20e3fe241d3cedab42f5b4ad"
    },
    {
        "switch_uuid" : "3c26037a-3105-4ebd-a2f7-8f729d8a5461",
        "tag" : "Gafgyt",
        "md5" : "1528dbfee080b4d6e45ea9ac36189b4c",
        "sha1" : "8a1ae76f51b38e9fa47bba865ff6760e99a78532",
        "sha256" :
"a02e8d85f6293dedffbae8b5a0dc6a25f44a51818f74289eae4bb37dfe096acf"
    }
]

$ curl --request GET --url https://10.10.10.43/api/show/hashes \
--header 'x-apikey: 03a2232c772d656fec6bf71c9d65711b'
[
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
```

```
        "tag" : "Mirai",
        "md5" : "5dd645591b4ca486f0b8a63119bcd9f0",
        "sha1" : "1a9ceb82f97120703ae8d9c6d46975cadcfc0f14",
        "sha256" :
"8439ae4c7b6cde315e17b2f428f33deb4b94a85b9832499cfbbbaf60b59a6293"
    },
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
        "tag" : "Mirai",
        "md5" : "b2356ec8d749914edcc18fd6647f3a4a",
        "sha1" : "46fe83c019db6dcc4a2710c575841caf5b10e1d8",
        "sha256" :
"d8c3aac41ed78942460f36d2a19341e9af7401efc30412575bd9132d58013636"
    },
    {
        "switch_uuid" : "a82857e1-40dc-4a24-a783-c94078e6b7f4",
        "tag" : "Hajime",
        "md5" : "d975ef8d6d5d89bbd12fbbd55871afd1",
        "sha1" : "0ed7fd399f93482d3cddf9d4f80ced428b527255",
        "sha256" :
"29c3e6a2e0dd0d0f091011c4dcd5568a01013018a2a1b7cd82cd6dd71876d95c"
    }
]
```


3.3 Fluxo de trabalho

O fluxo de trabalho do Heimdall, representado pelo Algoritmo 2 e pelo fluxograma da Figura 11, é composto pelos seguintes passos:

1. Inicialmente os *bytes* do pacote são extraídos pelo módulo Packet parser;
2. Os *bytes* coletados na etapa anterior são utilizados como entrada do módulo Yara para identificação rápida de artefatos conhecidos. Caso o pacote seja malicioso, o dispositivo é prontamente isolado da rede;
3. Se nenhuma regra YARA for capaz de identificar o artefato suspeito, os *bytes* são

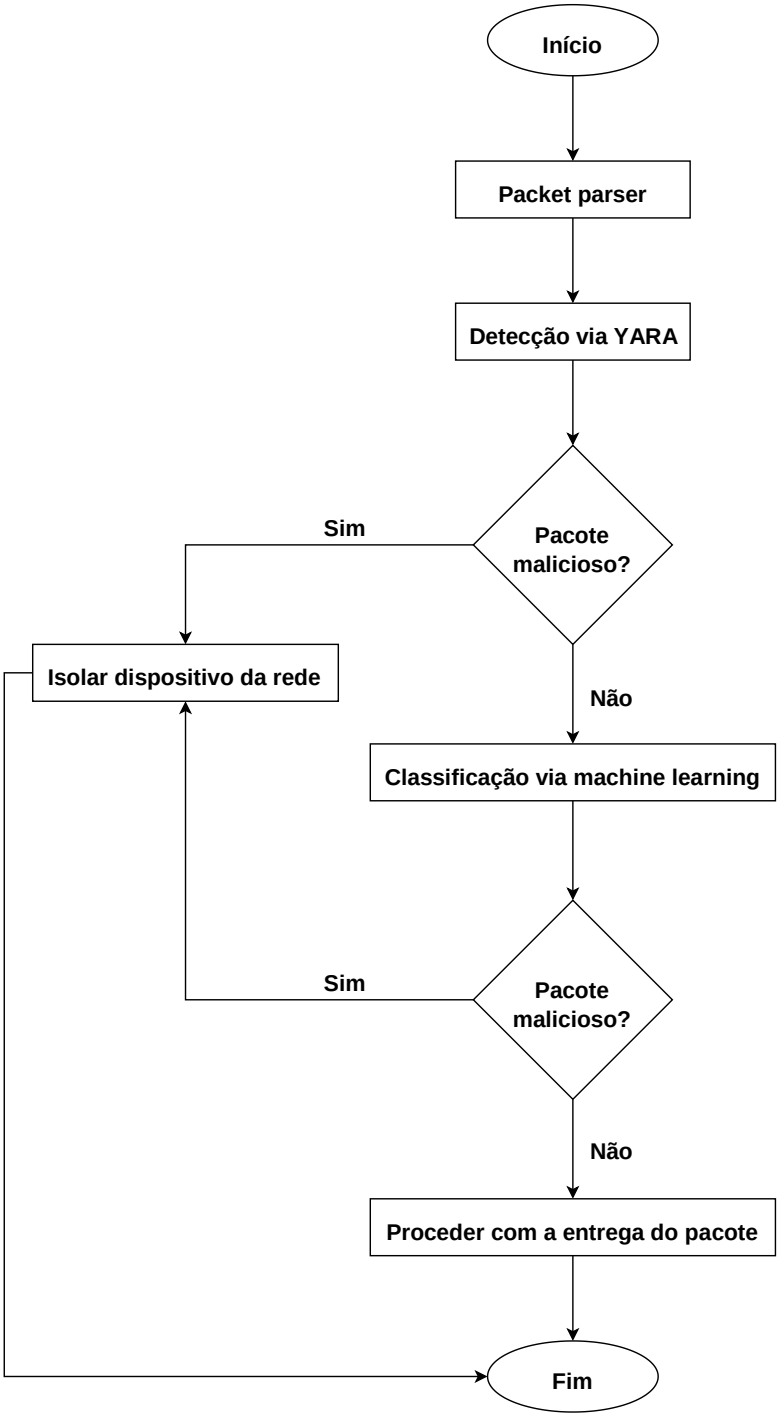
enviados ao módulo de classificação via *machine learning*. Caso o algoritmo identifique o pacote como malicioso, o dispositivo é isolado da rede;

4. Caso nenhum dos módulos identifique características maliciosas no pacote, a ferramenta procede com o encaminhamento dele ao destino.

Algoritmo 2 Lógica de operação do Heimdall

```
1: Input: packet bytes
2:
3: Begin
4:
5: payload_bytes ← Packet_Parser() 6: matches ← Y
ARA_Analysis(payload_bytes)
7:
8: if matches then
9:         Isolate the affected device
10: else
11:             isMalicious ← Machine_Learning_Classifier(payload_bytes)
12:
13:         if isMalicious then
14:             Isolate the affected device
15:         else
16:             Forward the packet
17:
18: End
```

Figura 11 – Fluxo de trabalho do Heimdall



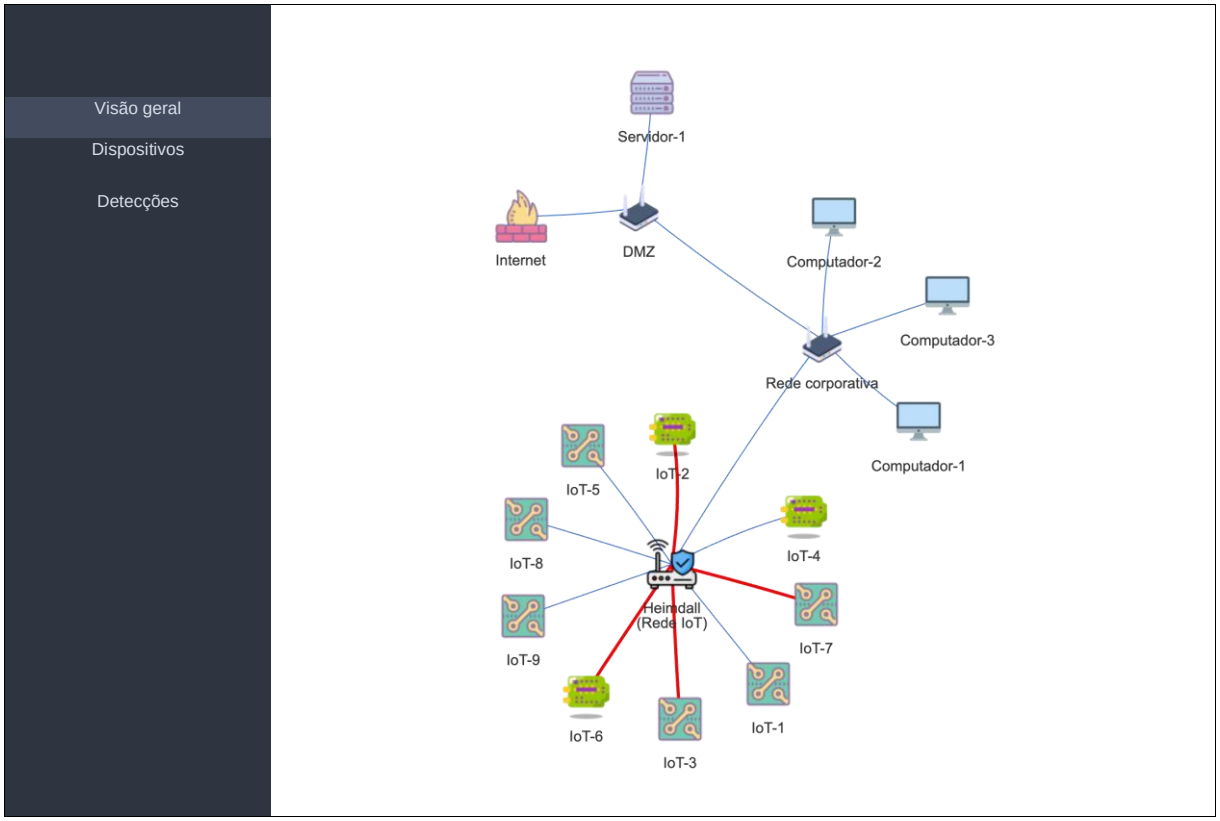
Fonte: Resultado da pesquisa

3.4 Interface gráfica

Com o intuito de facilitar a administração da solução proposta, uma interface gráfica foi desenvolvida. A Figura 12 apresenta a tela principal da ferramenta, na qual é possível visualizar a topologia de toda a infraestrutura, com destaque para a rede de dispositivos IoT

protegida. Essa tela também destaca de forma visual os dispositivos atualmente infectados, permitindo aos operadores a rápida tomada de decisão.

Figura 12 – Tela principal da aplicação



Fonte: Resultado da pesquisa

Ao clicar em um dispositivo, o operador é redirecionado para uma tela que contém a listagem de todos os equipamentos IoT conectados, com destaque para a situação atual de cada um deles (infectado ou não). Nessa tela, é possível realizar operações como visualizar detalhes, editar as propriedades do dispositivo ou isolá-lo do restante da infraestrutura. Dessa forma, a propagação de ameaças no ambiente industrial pode ser prontamente mitigada. Como pode ser observado na Figura 13, essa interface também expõe as informações de endereçamento IP e MAC, permitindo a fácil identificação dos dispositivos presentes na rede.

A Figura 14 apresenta a tela que contém informações sobre os artefatos maliciosos detectados na infraestrutura, é possível visualizar detalhes sobre as famílias dos artefatos e seus *hashes*. Tais informações podem ser utilizadas para buscar dados sobre o *malware* de interesse em bases públicas. Nessa tela, também é possível isolar os dispositivos afetados da rede ou liberá-los em caso de falsos positivos.

Figura 13 – Tela com a listagem de dispositivos IoT presentes na infraestrutura

Fonte: Resultado da pesquisa

Figura 14 – Tela com a listagem de detecções

Fonte: Resultado da pesquisa

4 VALIDAÇÃO E ANÁLISE

Este capítulo aborda a validação e análise da arquitetura proposta, bem como apresenta as lições aprendidas ao longo do desenvolvimento deste estudo. A solução é avaliada em uma plataforma de testes emulada por meio da ferramenta Mininet[14] e com a utilização do *software* BMv2 P4[15]. Para os cenários de avaliação propostos, é considerado que todos os dispositivos estão na mesma infraestrutura de rede. As avaliações foram realizadas em uma máquina com CPU Intel Core i7-11800H 2.30GHz (8 vCPUs), 32GB de RAM e sistema operacional Ubuntu Server 22.04.3 LTS 64-bits (Linux Kernel 6.2).

As seções a seguir apresentam os detalhes sobre as implementações e validações realizadas. O primeiro passo consiste na definição das regras YARA e treinamento dos modelos de *machine learning* capazes de detectar ameaças de *malware* a partir de um bom *dataset*. Em seguida, o modelo de maior acurácia é integrado à solução. A prova de conceito é feita considerando exemplares de *malware* reais. Por fim, é realizada uma análise do desempenho da ferramenta.

4.1 Regras YARA selecionadas

As regras YARA utilizadas na implementação foram obtidas a partir do repositório de código aberto Yara-Rules/rules[16]. Esse repositório contém regras elaboradas por profissionais de segurança da informação após a identificação e análise de ataques e artefatos maliciosos. As regras são divididas nas seguintes categorias:

- Anti-debug/Anti-VM: consiste em regras capazes de detectar técnicas de evasão comumente empregadas por *malwares*.

[14] http://mininet.org

[15] https://github.com/p4lang/behavioral-model

[16] https://github.com/Yara-Rules/rules

- Capabilities: regras utilizadas para detectar capacidades que não se encaixam em nenhuma das outras categorias. Embora sejam úteis para análise, podem não identificar características maliciosas isoladamente.
- CVE rules: regras especializadas na identificação de vulnerabilidades conhecidas e catalogadas pelo banco de dados CVE (do inglês, *Common Vulnerabilities and Exposures*).
- Crypto: regras utilizadas para detectar a existência de algoritmos criptográficos.
- Exploit kits: regras específicas para detecção da utilização de *kits* de exploração conhecidos.
- Malicious documents: regras YARA utilizadas para identificação de documentos maliciosos.
- Malware: regras especializadas na detecção de *malwares* conhecidos.
- Packers: regras capazes de identificar a presença de empacotadores conhecidos e utilizados para ocultar as funcionalidades do binário malicioso.
- WebShells: regras utilizadas para identificação de *web shells* conhecidos.
- E-mail: regras YARA para detecção de e-mails maliciosos, comumente utilizados em campanhas de *phishing*.
- Malware mobile: regras para identificação de *malwares* focados em dispositivos móveis.
- Deprecated: regras descontinuadas e não mantidas pela comunidade.

Com o objetivo de maximizar a performance do Heimdall, foram selecionadas regras das categorias pertinentes a ambientes IoT, sendo elas: Anti-debug/Anti-VM, CVE rules, Exploit kits, Malware, Packers e Malware mobile. As regras foram escolhidas devido ao fato de que *malwares* focados em IoT comumente empregam métodos para evasão de defesas, além de explorarem vulnerabilidades conhecidas nos dispositivos conectados.

4.2 Dataset utilizado

O *dataset* utilizado para treinamento dos modelos propostos neste estudo é o IoT-23 (GARCIA; PARMISANO; ERQUIAGA, 2020), criado pelo Avast AIC Laboratory. Esta base contém 20 capturas de *malwares* coletadas de diversos dispositivos IoT, além de 3 capturas de tráfego benigno. Seu objetivo é fornecer aos pesquisadores um grande conjunto de dados reais e rotulados de infecções e tráfego legítimo, visando auxiliar o desenvolvimento de algoritmos de *machine learning*.

O motivo da escolha desse *dataset* é a sua ampla utilização em estudos anteriores (NANTHIYA et al., 2021; JEELANI et al., 2022; OHA et al., 2021). Em números totais, o *dataset* possui 325.307.990 registros, sendo 294.449.255 deles maliciosos. Os tipos de ameaças presentes na base são descritos na Tabela 3 e suas colunas são detalhadas na Tabela 4.

Tabela 3 – Tipos de ameaças presentes no *dataset*

Tipo de ameaça	**Descrição**
Attack	Anomalias que não puderam ser identificadas e classificadas.
Benign	Tráfego benigno.
C&C	Tráfego gerado pela comunicação entre um dispositivo infectado e uma estação de comando e controle.
DDoS	Tráfego gerado por ataques de negação de serviço distribuídos.
FileDownload	Tráfego gerado pela transferência de arquivos maliciosos.
HeartBeat	Tráfego gerado pela estação de C&C para verificar a conexão com o alvo.
Mirai	Tráfego que possui características da *botnet* Mirai.
Okiru	Tráfego que possui características da *botnet* Okiru.
PartOfAHorizontalPortScan	Tráfego gerado por *scanners* de rede para coleta de informações.
Torii	Tráfego que possui características da *botnet* Torii.

Fonte: Resultado da Pesquisa.

Visto que as informações do *dataset* são distribuídas em diretórios, é necessário realizar o tratamento dos dados. Para isso, as informações de cada diretório foram combinadas em um único arquivo CSV, que é utilizado para treinar os modelos.

Tabela 4 – Colunas presentes no *dataset*

Coluna	Tipo	Descrição
ts	int	Horário da captura no formato Unix Timestamp.
uid	str	ID da captura.
id.orig_h	str	Endereço IPv4 ou IPv6 do dispositivo que originou o ataque.
id.orig_p	int	Porta utilizada pelo dispositivo que originou o ataque.
id.resp_h	str	Endereço IPv4 ou IPv6 do dispositivo onde a captura foi realizada.
id.resp_p	int	Porta utilizada pelo dispositivo alvo onde a captura foi realizada.
proto	str	Protocolo de rede utilizado.
service	str	Protocolo de aplicação utilizado.
duration	float	Duração da troca de dados entre o dispositivo infectado e o atacante.
orig_bytes	int	Quantidade de dados enviados ao dispositivo.
resp_bytes	int	Quantidade de dados enviados pelo dispositivo.
conn_state	str	Estado da conexão.
local_orig	bool	Identifica se a conexão foi originada localmente.
local_resp	bool	Identifica se a resposta foi originada localmente.
missed_bytes	int	Número de *bytes* faltantes.
history	str	Histórico do estado da conexão.
orig_pkts	int	Número de pacotes sendo enviados ao dispositivo.
orig_ip_bytes	int	Número de *bytes* sendo enviados ao dispositivo.
resp_pkts	int	Número de pacotes sendo enviados pelo dispositivo.
resp_ip_bytes	int	Número de *bytes* sendo enviados dispositivo.
tunnel_parents	str	Identificador da conexão, se ela for tunelada.
label	str	Tipo de captura (maliciosa ou benigna).
detailed_label	str	Detalhes sobre a captura, caso ela seja maliciosa.

Fonte: Resultado da Pesquisa.

Pré-processamento dos dados

A etapa de pré-processamento dos dados é fundamental para preparar e limpar os dados brutos antes de sua utilização no treinamento dos modelos, eliminando inconsistências que podem afetar a qualidade dos resultados. As seguintes operações foram realizadas após a leitura do arquivo CSV unificado:

1. **Remoção de colunas não importantes:** as colunas ts, uid e tunnel_parents foram removidas por não representarem dados relevantes para treinamento dos modelos. Já as colunas local_orig e local_resp foram eliminadas por não serem únicas.

2. ***Label encoding*:** esta técnica é utilizada para converter dados categóricos em valores numéricos, e foi aplicada para as colunas id.orig_h, id.resp_h, proto, service, conn_state e history.

3. **Substituição de valores ausentes:** linhas sem dados nas colunas duration, orig_bytes ou resp_bytes tiveram os valores ausentes configurados como a média da respectiva coluna.

4. ***Feature scaling*:** esta técnica foi utilizada para melhorar o desempenho dos modelos e reduzir o impacto de diferenças nas escalas das variáveis consideradas.
5. **Separação do conjunto de treinamento e de teste:** o *dataset* foi dividido em conjuntos de treinamento e de teste na proporção de 7:3.

4.3 Algoritmos implementados

Com o objetivo de selecionar o modelo de melhor acurácia e performance, os algoritmos *Random Forest*, SVM e uma árvore de decisão foram implementados. Ademais, uma rede neural convolucional também foi desenvolvida. A escolha de tais algoritmos se deu pelo seu amplo uso em problemas de classificação de *malware* em estudos anteriores.

Para a rede neural convolucional, é adicionada uma camada de *max pooling* 1D com pool_size=2, o que reduz a dimensionalidade dos recursos. Uma camada *flatten* é adicionada

à rede, transformando os recursos 2D resultantes da camada de *max pooling* em um vetor 1D, que é utilizado como entrada para as camadas totalmente conectadas. São adicionados 500 neurônios na camada totalmente conectada. O otimizador Adam é escolhido para ajustar os pesos da rede.

O modelo baseado em árvores de decisão é configurado com as opções padrão da biblioteca *scikit-learn*[17]. Já o algoritmo *Random Forest* é iniciado com dois *jobs* paralelos, random_state=0 e o número de árvores é configurado para 100. Isso permite que os resultados sejam facilmente reproduzidos a partir do mesmo conjunto de dados. Por fim, o modelo baseado no algoritmo SVM é configurado com parâmetro de regularização 1, visando evitar classificações incorretas. O valor de *cache* do *kernel* foi definido como 700MB.

As métricas utilizadas para determinar a efetividade dos modelos foram: acurácia, precisão, *recall* e F1-Score. Como exposto na Seção 2.5, a acurácia indica a performance geral do modelo (quantidade de classificações corretas); a precisão mede a proporção de exemplos classificados como positivos pelo modelo que são realmente positivos; o *recall* consiste na proporção de exemplos positivos que foram corretamente classificados pelo modelo; e o F1-Score representa a média harmônica entre a precisão e o *recall*. A Tabela 5 apresenta os resultados obtidos após o treinamento dos modelos.

Como exposto na Tabela 5, o modelo de melhor acurácia foi o treinado com o algoritmo *Random Forest*, atingindo a marca de 99.33%. Os algoritmos com precisões mais elevadas foram o *Random Forest* e a CNN, ambos com 0.97. O *recall* de todos os modelos atingiu um valor satisfatório. Por fim, os algoritmos de melhor F1-Score também foram o *Random Forest* e a rede neural, ambos com 0.98.

Tabela 5 – Métricas dos modelos treinados

Algoritmo	**Acurácia**	**Precisão**	***Recall***	**F1-Score**
CNN	92.83%	0.97	0.99	0.98
Decision Tree	97.33	0.93	0.99	0.96
Random Forest	99.33%	0.97	0.99	0.98
SVM	94%	0.91	0.93	0.92

Fonte: Resultado da Pesquisa.

[17] https://scikit-learn.org

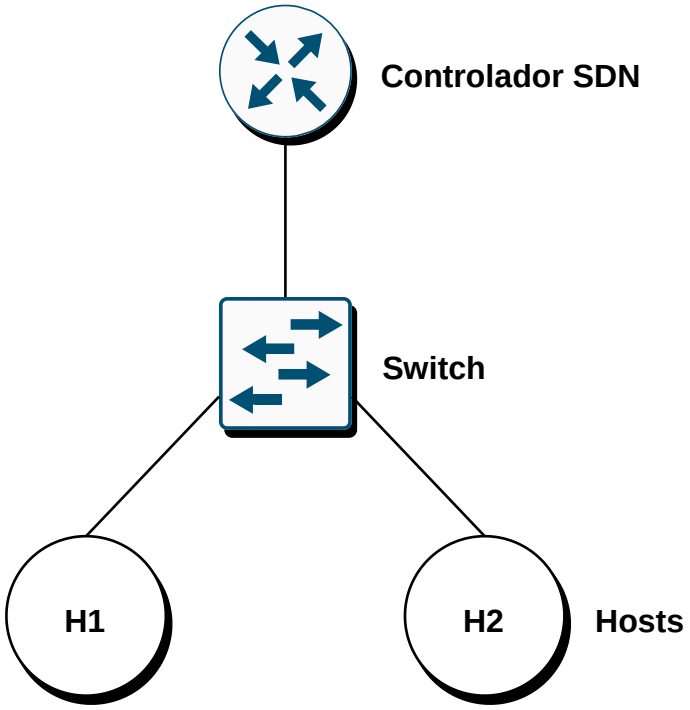
Com base nas métricas apresentadas, é evidente que todos os modelos atingiram um desempenho notável, com o *Random Forest* liderando em termos de acurácia e F1-Score. Diante disso e de sua velocidade, esse modelo foi escolhido para avaliação completa da ferramenta proposta.

4.4 Prova de conceito

Para validar a ferramenta proposta, o modelo treinado com o algoritmo *Random Forest* foi acoplado à arquitetura do Heimdall em uma infraestrutura IoT habilitada por SDN. Com o objetivo de avaliar a capacidade de generalização do modelo e a efetividade da abordagem híbrida com o YARA, foram utilizados 3.030 exemplares de *malware* reais.

A Figura 15 apresenta os detalhes da topologia utilizada para realização dos testes, sendo composta por dois dispositivos, um *switch* programável e um controlador SDN. O primeiro teste (descrito na Subseção 4.4.1) consistiu no envio dos binários do H1 para o H2. O objetivo desse teste é identificar a capacidade da ferramenta em detectar binários maliciosos trafegando na rede exclusivamente via YARA. O segundo teste, apresentado na Subseção 4.4.2, que também consistiu no envio dos artefatos a partir do H1, visa avaliar a capacidade do algoritmo *Random Forest* em detectar atividades maliciosas na rede em tempo real. Por fim, as duas abordagens são ativadas na ferramenta na Subseção 4.4.3, com o objetivo de avaliar por completo a solução, bem como a sua capacidade de prontamente isolar os dispositivos afetados.

Figura 15 – Topologia utilizada para realização das avaliações



Fonte: Resultado da pesquisa

Os artefatos maliciosos foram obtidos a partir da plataforma Malware Bazaar[18], uma solução amplamente utilizada pela comunidade de cibersegurança para análise e compartilhamento de *malware*, de forma automatizada por meio de um *script* de consulta à sua API pública. Foram utilizadas as *tags* "iot", "arm", "mirai", "gafgyt"e "hajime"para pesquisa de binários maliciosos que afetam dispositivos IoT, totalizando 3.030 artefatos. A Figura 16 apresenta a quantidade de *malwares* por família. Após o *download* deles em formato ZIP, os binários maliciosos foram extraídos e enviados ao *host* H1.

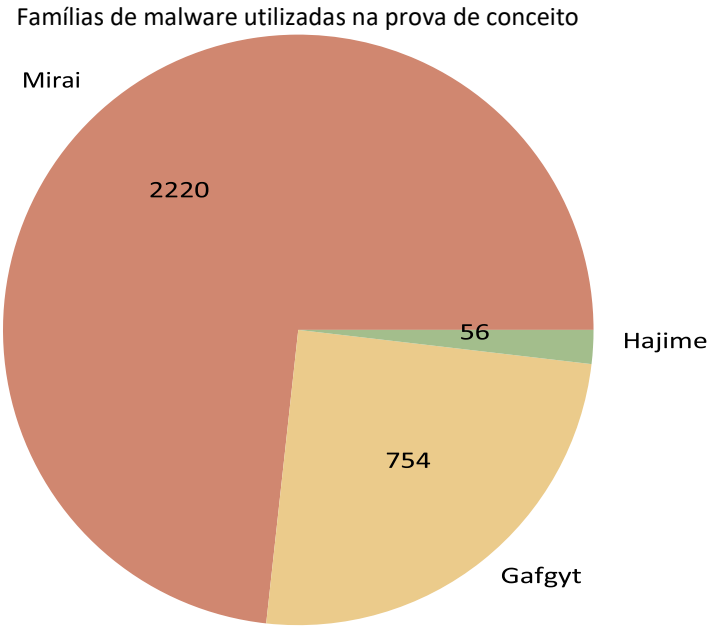
4.4.1 Taxa de detecção via YARA

No primeiro cenário avaliado, os artefatos maliciosos foram enviados de H1 para H2. Cada pacote que passa pelo *switch* é analisado a partir das regras previamente compiladas. Nesse cenário, das 3.030 amostras de artefatos maliciosos transferidas de um dispositivo para

[18] https://bazaar.abuse.ch

o outro, a solução foi capaz de identificar com sucesso 1.908, o que representa uma taxa de sucesso de 63%.

Figura 16 – Famílias de *malware* utilizadas na validação



Fonte: Resultado da pesquisa

Como exposto na Figura 17, a principal família de *malware* detectada foi a Mirai, com 1.483 detecções (77.72% do total de artefatos identificados). A família Gafgyt teve 409, representando 21.43% do total de detecções. Por fim, a família Hajime teve apenas 16 detecções, sendo 0.83% das detecções. A Tabela 6 apresenta a relação de detecções em relação ao total de artefatos utilizados na validação.

Tabela 6 – Relação de artefatos detectados via YARA

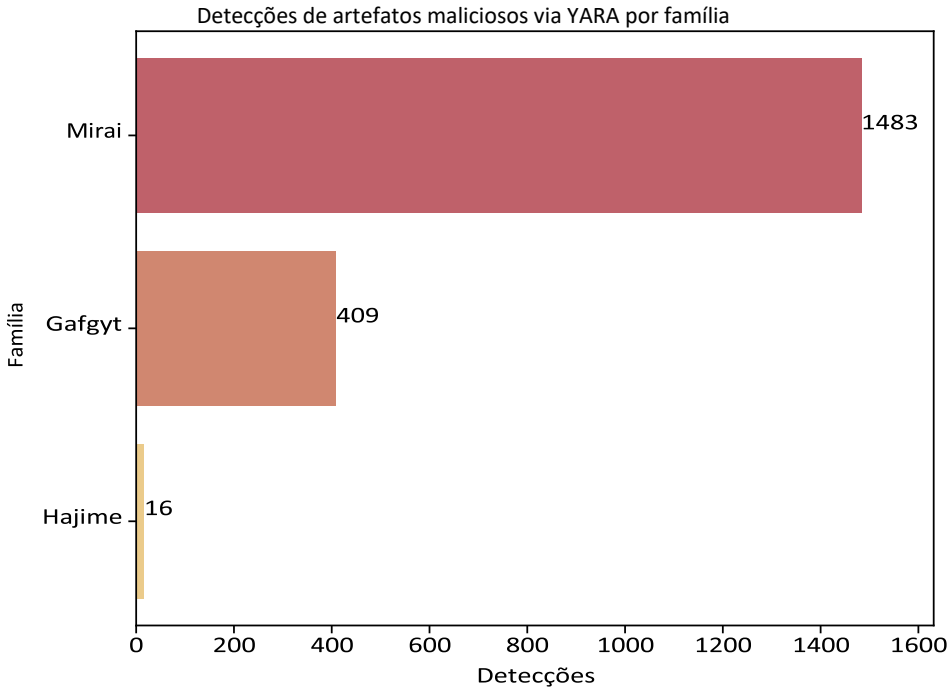
Família	**Quantidade**	**Detectados**	**Taxa de detecção**
Mirai	2.220	1.483	66.80%
Gafgyt	754	409	54.24%
Hajime	56	16	28.57%
Total	**3.030**	**1.908**	**63%**

Fonte: Resultado da Pesquisa.

Em relação às regras utilizadas, foi possível observar que as principais responsáveis pela detecção dos artefatos foram as focadas na identificação de algoritmos criptográficos, como a Big_Numbers_1 e maldoc_getEIP_method_1. A primeira busca por sequências de 32

caracteres hexadecimais em dados binários; já a segunda tem como objetivo identificar binários codificados com a função XOR. A regra Mirai_2 também teve destaque, sendo utilizada 107 vezes. Por fim, as regras Chacha_256_constant e RijnDael_AES_CHAR, também relacionadas a criptografia, foram utilizadas 69 e 22 vezes respectivamente.

Figura 17 – Detecções via YARA por família de *malware*



Fonte: Resultado da pesquisa

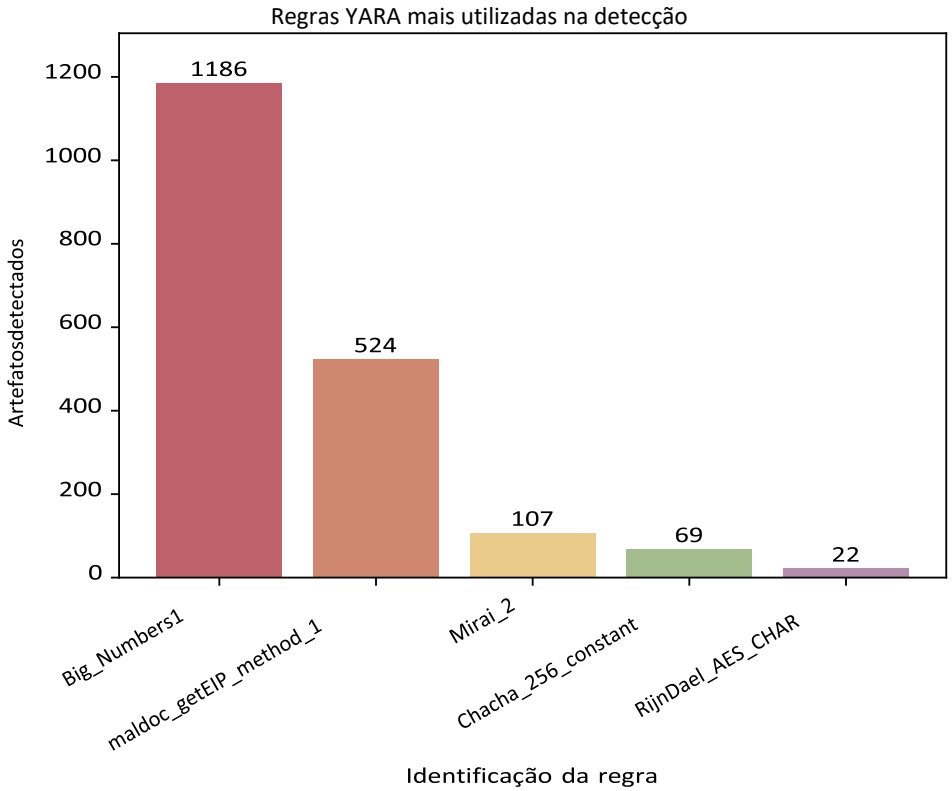
Essas estatísticas, sumarizadas na Figura 18, demonstram que *malwares* focados em IoT comumente empregam métodos de criptografia em seus algoritmos.

A Figura 19 expõe o tempo de processamento por *malware* trafegado na rede. Em média, a solução levou 0.0218s para classificação dos artefatos, o que representa um baixo atraso na entrega dos pacotes ao destino. O menor tempo registrado para classificação foi de 0.0006s e o maior de 0.0399s. Embora a taxa de detecção tenha sido de 63%, o baixo atraso na análise evidencia a viabilidade da utilização de regras YARA para detecção de artefatos conhecidos em ambientes IoT.

4.4.2 Acurácia do algoritmo Random Forest

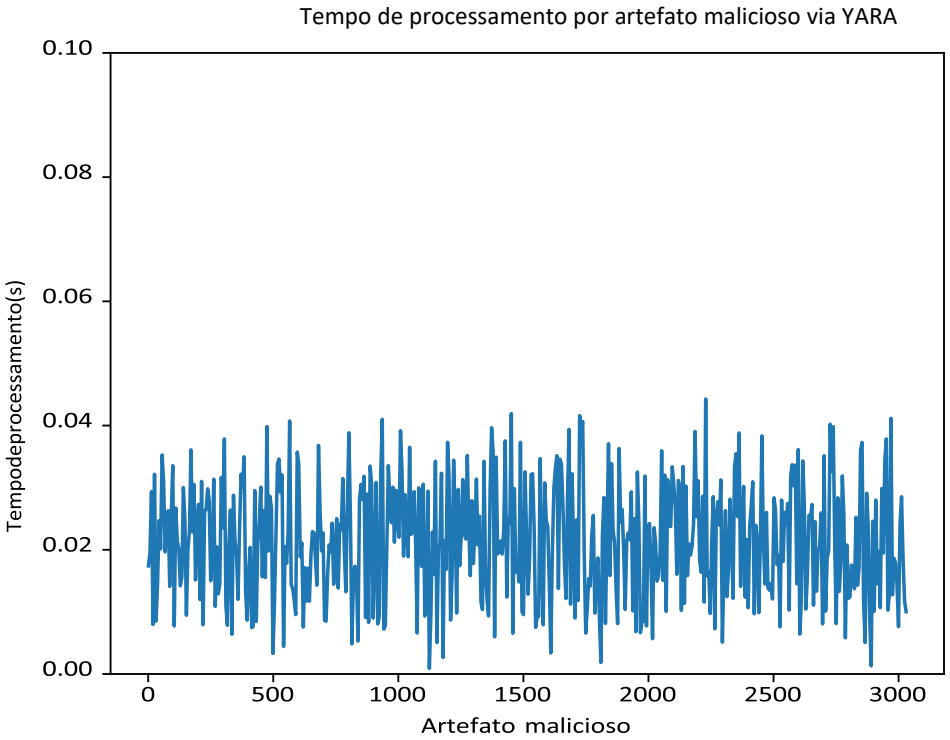
Neste cenário, os artefatos maliciosos foram enviados de H1 para H2, mas com apenas o módulo de *machine learning* habilitado no Heimdall. Os resultados evidenciam que, do total de artefatos transferidos de um dispositivo para o outro, o algoritmo obteve uma acurácia de detecção geral de 94.12%, evidenciando a capacidade de generalização do modelo. Como apresentado na Figura 20 e na Tabela 7, a família de *malware* com maior taxa de detecção também foi a Mirai, com um total de 2.139 detecções. É importante notar que o número de detecções para a família Gafgyt teve um aumento considerável (com 681 artefatos corretamente classificados), o mesmo aconteceu com artefatos da família Hajime, com o dobro de detecções.

Figura 18 – Regras YARA mais utilizadas



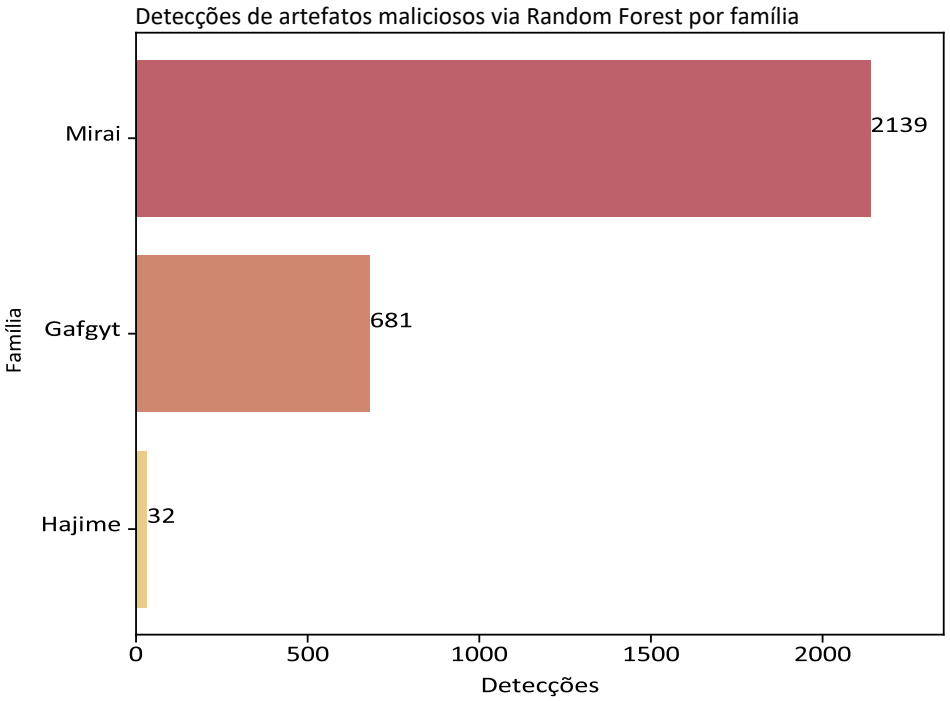
Fonte: Resultado da pesquisa

Figura 19 – Tempo para classificação dos artefatos via YARA



Fonte: Resultado da pesquisa

Figura 20 – Detecções via *Random Forest* por família de *malware*



Fonte: Resultado da pesquisa

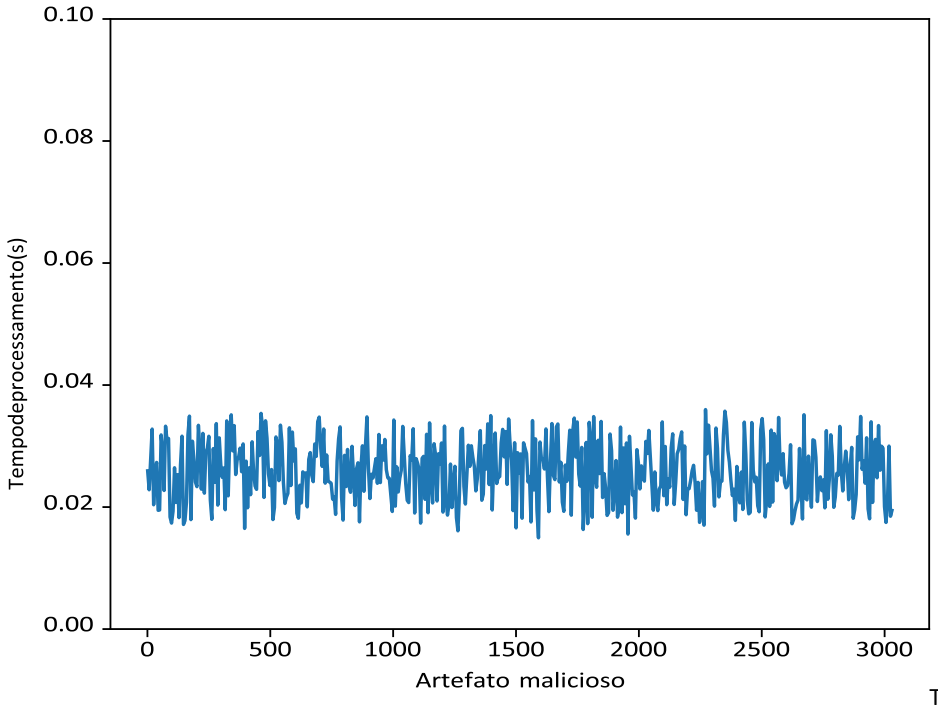
Tabela 7 – Relação de artefatos detectados via *Random Forest*

Família	Quantidade	Detectados	Taxa de detecção
Mirai	2.220	2.139	96.35%
Gafgyt	754	681	90.31%
Hajime	56	32	57.14%
Total	**3.030**	**2.852**	**94.12%**

Fonte: Resultado da pesquisa

O tempo de processamento por parte do algoritmo também se apresentou baixo, como pode ser observado na Figura 21, com uma média de 0.0259s. O menor tempo registrado para o algoritmo *Random Forest* foi de 0.0170s e o maior de 0.0349s. A elevada taxa de detecção do modelo demonstra que algoritmos de aprendizado de máquina podem ser utilizados para identificar artefatos maliciosos caso as regras falhem. Com a sofisticação dos ataques e das técnicas de evasão utilizadas pelos atacantes, o uso de modelos genéricos e adaptáveis se torna crucial para proteger eficazmente as redes IoT.

Figura 21 – Tempo para classificação dos artefatos via *Random Forest*



Tempo de processamento por artefato malicioso via Random Forest

Fonte: Resultado da pesquisa

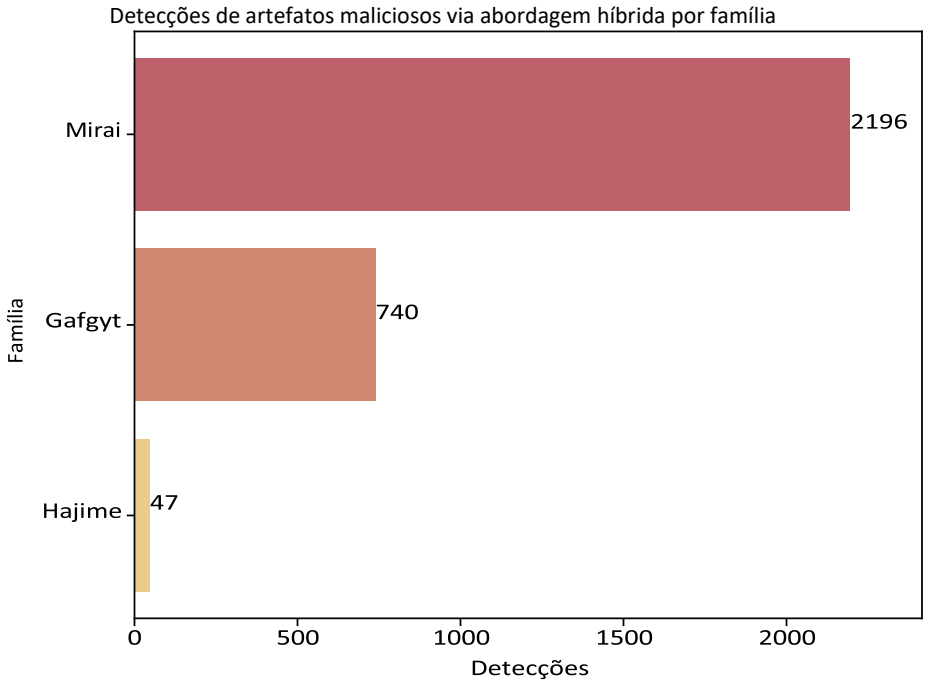
4.4.3 Abordagem híbrida

O último e principal cenário utilizado para avaliação do Heimdall consistiu no envio dos artefatos de H1 para H2 com ambos os módulos habilitados. O objetivo desse teste é validar a capacidade de detecção da ferramenta por meio de uma abordagem híbrida, composta pelo uso de regras YARA e *machine learning*.

Como exposto na Figura 22 e na Tabela 8, a acurácia da ferramenta alcançou a marca de 98.44% ao se adotar a abordagem supracitada. Isso demonstra que a combinação de assinaturas e *machine learning* potencializa a capacidade de identificação de programas maliciosos. A taxa de detecções por família seguiu o mesmo padrão dos testes anteriores, sendo a família Mirai a com maior número de detecções (2.196), seguida pela família Gafgyt (740) e Hajime (47).

Ademais, constatou-se que o tempo de processamento não teve alterações significativas, visto que o módulo de *machine learning* apenas é ativado caso as regras não sejam capazes de classificar o pacote analisado. O tempo médio para classificação dos artefatos por meio da abordagem híbrida foi de 0.0217s, como apresentado na Figura 23. O menor tempo registrado para classificação via a abordagem híbrida foi de 0.0006s e o maior de 0.0347s.

Figura 22 – Detecções por meio da abordagem híbrida



Fonte: Resultado da pesquisa

Tabela 8 – Relação de artefatos detectados via abordagem híbrida

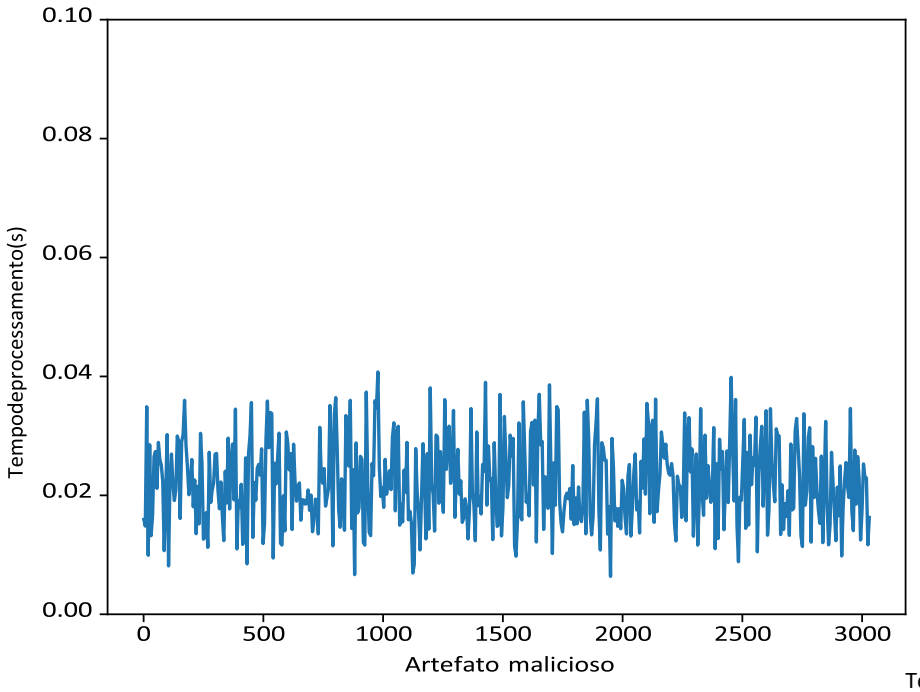
Família	Quantidade	Detectados	Taxa de detecção
Mirai	2.220	2.196	98.91%
Gafgyt	754	740	98.14%
Hajime	56	47	83.92%
Total	**3.030**	**2.983**	**98.44%**

Fonte: Resultado da Pesquisa.

4.5 Análise de desempenho

O objetivo desta seção é analisar o tempo de processamento da ferramenta, sua complexidade e suas limitações.

Figura 23 – Tempo para classificação dos artefatos via abordagem híbrida



Tempo de processamento por artefato malicioso via abordagem híbrida

Fonte: Resultado da pesquisa

4.5.1 Tempo de processamento

Como exposto na Figura 23 e na Subseção 4.4.3, a adoção de uma abordagem híbrida otimiza a taxa de detecção e tempo de processamento geral da ferramenta. A Tabela 9

apresenta as estatísticas gerais de tempo de processamento e acurácia das avaliações realizadas.

Tabela 9 – Comparação entre os cenários avaliados

Abordagem	Tempo médio	Menor registro	Maior registro	Acurácia
YARA	0.0218s	0.0006s	0.0399s	63%
Random Forest	0.0259s	0.0170s	0.0349s	94.12%
Híbrida	0.0217s	0.0006s	0.0347s	98.44%

Fonte: Resultado da Pesquisa.

Os resultados apresentam que, embora regras YARA se apresentem rápidas para detecção de artefatos conhecidos, a acurácia dessa abordagem pode ser comprometida caso o binário analisado não possua os padrões catalogados. O algoritmo *Random Forest* obteve uma boa acurácia, embora seu tempo de processamento seja ligeiramente maior. Ao combinar as duas abordagens, foi possível obter uma acurácia de 98.44%, evidenciando que o uso de um mecanismo híbrido é capaz de potencializar a taxa de detecção de artefatos maliciosos em ambientes IoT.

É importante notar que tais resultados representam o tempo de processamento médio no ambiente virtualizado utilizado para condução dos testes. Logo, alterações nos parâmetros do ambiente (como CPU, RAM e topologia da rede) podem influenciar os resultados. Ademais, o uso de um *switch* programável físico elimina a necessidade de virtualização e, consequentemente, pode reduzir ainda mais o tempo de processamento da ferramenta.

4.5.2 Análise de complexidade

A complexidade da ferramenta YARA para casamento de padrões depende de fatores como o número de regras, tamanho da sequência de *bytes* a ser analisada e padrões definidos nas regras. A complexidade de tempo do algoritmo YARA para uma única regra é linear em relação ao tamanho da sequência de *bytes*. Isso significa que, para uma regra, o tempo de execução aumentará proporcionalmente ao tamanho da sequência de *bytes*. Logo, para um conjunto de *n* regras e *m bytes* a serem analisados, a complexidade pode ser definida como

$\Theta(n \cdot m)$. Essa expressão indica que o tempo de execução do algoritmo cresce linearmente de acordo com o número de regras e quantidade de *bytes* passados como entrada do algoritmo.

O tempo de treinamento do algoritmo *Random Forest* normalmente é mais alto em comparação a algoritmos mais simples, como árvores de decisão. Isso se deve ao fato de esse algoritmo utilizar múltiplas árvores em sua implementação. A complexidade do treinamento de uma única árvore de decisão pode ser definida pela fórmula $\Theta(n \cdot m \cdot \log m)$, onde n é o número de amostras no conjunto de treinamento e m é o número de características utilizadas. Tendo em vista que o algoritmo constrói várias árvores de decisão (representado pelo parâmetro *n_estimators*, da biblioteca *scikit-learn*), a complexidade total do treinamento é proporcional ao número de árvores multiplicado pela complexidade de construção de uma árvore, resultando na expressão $\Theta(n\_estimators \cdot n \cdot m \cdot \log m)$.

A predição do algoritmo é considerada rápida e eficiente, pois cada árvore fornece uma previsão individual e a previsão final é determinada pela votação majoritária das previsões das árvores. A complexidade da predição para uma única árvore é $\Theta(\log m)$. Logo, a complexidade de predição do *Random Forest* pode ser representada por $\Theta(n\_estimators \cdot \log m)$.

O algoritmo *Random Forest* é considerado escalável para conjuntos de dados grandes, devido à sua capacidade de treinar árvores independentes de forma paralela (LIU; WANG; ZHANG, 2012). A aleatoriedade na construção das árvores e a combinação de várias delas ajudam a reduzir problemas relacionados a *overfitting* (REN et al., 2015), tornando o modelo mais generalista.

Portanto, assumindo que a identificação via regras YARA e a classificação pelo modelo treinado são executados em sequência (ou seja, o algoritmo de *machine learning* só é utilizado caso as regras não identifiquem nenhum padrão), a complexidade final, no pior cenário, para análise de um artefato consiste na soma das complexidades de cada parte, como apresentado na Equação (5).

$$\Theta\left(n_{yara} \cdot m_{yara} + n\_estimators \cdot \log m_{rf}\right) \quad (5)$$

onde n_{yara} representa o número de regras YARA utilizadas pela ferramenta; m_{yara} a quantidade de *bytes* a serem analisados; *n_estimators* o número de árvores de decisão utilizadas; e m_{rf} o número de características utilizadas.

4.6 Limitações

Embora a ferramenta proposta na presente dissertação tenha obtido uma acurácia e tempo de processamento satisfatórios, foram identificadas as seguintes limitações, que podem ser endereçadas em estudos futuros:

- **Atualização das regras YARA:** tendo em vista que as regras são acopladas diretamente ao *switch* SDN, a abordagem atual da ferramenta necessita que o operador da rede atualize as regras em cada comutador habilitado com o Heimdall, o que pode ser trabalhoso em redes complexas. Para solucionar essa limitação, uma opção viável é a utilização de um servidor central, no qual os *switches* podem consultar atualizações para as regras periodicamente e de forma automatizada.
- **Pacotes encriptados:** lidar com pacotes criptografados (e.g., via TLS) é um desafio para ferramentas de segurança, como sistemas de IDS e IPS. O mesmo se aplica ao presente trabalho caso o atacante transfira o próprio artefato ou pacotes gerados por ele de maneira encriptada. Em (MODI et al., 2020), os autores propõem a utilização de um modelo de *machine learning* especializado na detecção de *ransomwares* trafegados de forma criptografada. Já o trabalho de (FU et al., 2022) especifica uma abordagem baseada em um algoritmo de aprendizado de representação de grafos para o mesmo propósito. Uma opção viável para superar essa limitação no Heimdall é a coleta de dados relativos a tráfego criptografado de artefatos maliciosos e incluí-los ao *dataset* atual, visando avaliar a taxa de detecção da ferramenta em cenários nos quais a criptografia é habilitada.

- **Experimentos em ambiente virtualizado:** o elevado custo de *switches* programáveis dificulta o acesso a tais dispositivos e, consequentemente, a execução de experimentos

em cenários realistas. Embora os resultados obtidos no ambiente virtualizado tenham sido positivos em termos de acurácia e desempenho, uma avaliação considerando um comutador físico pode comprovar a efetividade da solução proposta em cenários reais.

5 CONSIDERAÇÕES FINAIS

A segurança da informação é de fundamental importância para a proteção dos sistemas produtivos na Indústria 4.0. Ameaças de *malware* podem comprometer a confidencialidade, disponibilidade e integridade dos dados em ambientes industriais, especialmente em cenários que envolvem dispositivos IoT, onde a aplicação de medidas de segurança se torna complexa em virtude das diferentes configurações e requisitos de cada aparelho. Nesse contexto, soluções que atuam em nível de rede se destacam como efetivas na mitigação e contenção de programas maliciosos, evitando a propagação e maiores comprometimentos ao parque tecnológico das organizações.

Este trabalho de pesquisa introduz o Heimdall, uma solução para detecção de artefatos maliciosos em ambientes IoT habilitados por SDN. A arquitetura da ferramenta proposta se diferencia do estado da arte por fazer o uso de uma abordagem híbrida, capaz de realizar a rápida identificação de programas maliciosos via regras YARA e a classificação deles por meio do algoritmo de *machine learning Random Forest*. Com o auxílio de uma
API, os operadores da rede podem facilmente alterar e gerenciar diferentes atributos da ferramenta.

Para o treinamento dos algoritmos de *machine learning* implementados neste estudo, foi utilizado o *dataset* IoT-23, que contém uma quantidade considerável de amostras benignas e maliciosas, além de ser amplamente utilizado para validações de ferramentas similares. A escolha do algoritmo supracitado se deu por sua acurácia durante o treinamento, que atingiu a marca de 99.33%. É importante notar que esse valor se refere à acurácia obtida ao se testar o algoritmo com o conjunto de teste do *dataset*. Como exposto na Subseção 4.4.2, a taxa de detecção para os 3.030 novos artefatos foi de 94.12%.

Os resultados obtidos evidenciam que a abordagem híbrida proposta foi capaz de atingir uma acurácia de 98.44% e um tempo de processamento médio de 0.0217s. Em face da constante evolução dos *malwares* na atualidade, torna-se imperativo empregar ferramentas robustas e generalistas para assegurar a proteção eficaz das redes heterogêneas modernas. Dessa forma, o Heimdall se apresenta como uma solução viável para detecção de programas maliciosos em ambientes IoT habilitados pela tecnologia SDN.

5.1 Lições aprendidas

O desenvolvimento da prova de conceito do Heimdall e a análise das suas limitações, expostas em detalhes na Seção 4.6, proporcionaram importantes constatações. Primeiramente, a atualização das regras em cada *switch* pode dificultar o gerenciamento da solução em redes complexas.

O uso de criptografia para a transferência de programas maliciosos, assim como para a comunicação entre o atacante e a vítima, dificulta a detecção precisa de ameaças, sendo necessário adaptar o modelo existente para lidar com esse tipo de padrão. Outro ponto importante é que o tempo de identificação via YARA cresce de acordo com o número de regras e a quantidade de *bytes* analisada, o que pode causar impactos no caso de regras mal otimizadas ou binários muito grandes.

5.2 Trabalhos futuros

As limitações e lições aprendidas no desenvolvimento da ferramenta guiarão trabalhos futuros, com o objetivo de preencher as lacunas identificadas. A seguir são apresentados os principais direcionamentos para a continuação desta pesquisa:

- **Avaliação da proposta em um ambiente real:** pretende-se avaliar a ferramenta em um *switch* programável físico, visando determinar a eficiência da solução ao ser executada em um *hardware* dedicado.
- **Criação de uma interface gráfica para gerenciamento holístico:** uma opção para facilitar o gerenciamento da ferramenta é o desenvolvimento de uma interface gráfica que permita uma visão holística de todas as instâncias do Heimdall presentes na infraestrutura. Dessa forma, será possível, por exemplo, atualizar as regras e o próprio modelo a partir de uma única operação.
- **Desenvolvimento de um ambiente para análise de *malware*:** fornecer um ambiente controlado para análise estática e dinâmica dos programas maliciosos detectados pode

fornecer informações valiosas aos operadores da rede, bem como permitir a identificação de falsos positivos.

- **Propor uma metodologia para avaliação de ferramentas focadas na detecção de *malware*:** realizar uma comparação justa entre diferentes ferramentas é uma tarefa difícil, visto que os *datasets* utilizados para treinamento dos modelos são diversificados. Logo, propor uma metodologia padrão para *benchmarking* de ferramentas baseadas em *machine learning* para detecção de *malwares* em ambientes IoT pode ajudar os pesquisadores na comparação das suas ferramentas em relação ao estado da arte.

- **Avaliar o comportamento da ferramenta contra artefatos especializados em se evadir de defesas:** *malwares* podem utilizar diversas técnicas de evasão para não serem detectados. Logo, avaliar o comportamento do Heimdall considerando artefatos especializados na evasão de defesas pode fornecer dados importantes para aprimorar a capacidade de detecção da solução.

- **Implementação de outros algoritmos de aprendizado de máquina:** diversos algoritmos estão disponíveis em bibliotecas de *machine learning*. A implementação e avaliação deles pode oferecer o discernimento a respeito do melhor modelo para determinados cenários (e.g., no caso de *malwares* que utilizam técnicas de evasão).

REFERÊNCIAS

ABOAOJA, F. A. et al. Malware detection issues, challenges, and future directions: A survey. *Applied Sciences*, MDPI, v. 12, n. 17, p. 8482, 2022.

AGBAJE, P. et al. Survey of interoperability challenges in the internet of vehicles. *IEEE ransactions on Intelligent Transportation Systems*, IEEE, v. 23, n. 12, p. 22838–22861, 2022.

AHANGER, T. A.; ALJUMAH, A.; ATIQUZZAMAN, M. State-of-the-art survey of artificial intelligent techniques for iot security. *Computer Networks*, Elsevier, v. 206, p.108771, 2022.

AHMID, M.; KAZAR, O. A comprehensive review of the internet of things security. *Journal of Applied Security Research*, Taylor & Francis, v. 18, n. 3, p. 289–305, 2023.

AL-GARADI, M. A. et al. A survey of machine and deep learning methods for internet of things (iot) security. *IEEE Communications Surveys & Tutorials*, IEEE, v. 22, n. 3, p. 1646–1685, 2020.

ALAMRI, B.; CROWLEY, K.; RICHARDSON, I. Blockchain-based identity management systems in health iot: A systematic review. *IEEE Access*, IEEE, v. 10, p. 59612–59629, 2022.

ALBOUQ, S. S. et al. A survey of interoperability challenges and solutions for dealing with them in iot environment. *IEEE Access*, IEEE, v. 10, p. 36416–36428, 2022.

ALI, T. E.; CHONG, Y.-W.; MANICKAM, S. Machine learning techniques to detect a ddos attack in sdn: A systematic review. *Applied Sciences*, MDPI, v. 13, n. 5, p. 3183, 2023.

ALLADI, T. et al. Consumer iot: Security vulnerability case studies and solutions. *IEEE Consumer Electronics Magazine*, IEEE, v. 9, n. 2, p. 17–25, 2020.

ALLIANCE, O. M. Lightweight machine to machine technical specification. *Approved Version*, v. 1, n. 1, 2017.

ALQUDAH, N.; YASEEN, Q. Machine learning for traffic analysis: a review. *Procedia Computer Science*, Elsevier, v. 170, p. 911–916, 2020.

ALSAEEDI, M.; MOHAMAD, M. M.; AL-ROUBAIEY, A. A. Toward adaptive and scalable openflow-sdn flow control: A survey. *IEEE Access*, IEEE, v. 7, p. 107346–107379, 2019.

ALWASHALI, A. A. M. A.; RAHMAN, N. A. A.; ISMAIL, N. A survey of ransomware as a service (raas) and methods to mitigate the attack. In: IEEE. *2021 14th International Conference on Developments in eSystems Engineering (DeSE).* [S.I.], 2021. P. 92–96.

AMIRA, A. et al. A survey of malware analysis using community detection algorithms. *ACM Computing Surveys*, ACM New York, NY, v. 56, n. 2, p. 1–29, 2023.

ASLAN, Ö.; OZKAN-OKAY, M.; GUPTA, D. A review of cloud-based malware detection system: Opportunities, advances and challenges. *European Journal of Engineering and Technology Research*, v. 6, n. 3, p. 1–8, 2021.

ATLAM, H. F.; WILLS, G. B. Iot security, privacy, safety and ethics. *Digital twin technologies and smart cities*, Springer, p. 123–149, 2020.

AYODELE, B.; BUTTIGIEG, V. Sdn as a defence mechanism: a comprehensive survey. *International Journal of Information Security*, Springer, p. 1–45, 2023.

BALOGH, Š.; MOJŽIŠ, J.; KRAMMER, P. Evaluation of system features used for malware detection. In: SPRINGER. *Proceedings of the Future Technologies Conference (FTC) 2021, Volume 3*. [S.l.], 2022. p. 46–59.

BASTOS, A. et al. Industry 4.0 readiness assessment method based on rami 4.0 standards. *IEEE Access*, IEEE, v. 9, p. 119778–119799, 2021.

BAZZI, A.; SHAOUT, A.; MA, D. Mt-sota: A merkle-tree-based approach for secure software updates over the air in automotive systems. *Applied Sciences*, MDPI, v. 13, n. 16, p. 9397, 2023.

BEAMAN, C. et al. Ransomware: Recent advances, analysis, challenges and future research directions. *Computers & security*, Elsevier, v. 111, p. 102490, 2021.

BELLI, L. et al. Toward industry 4.0 with iot: Optimizing business processes in an evolving manufacturing factory. *Frontiers in ICT*, Frontiers Media SA, v. 6, p. 17, 2019.

BI, J. et al. Defense of advanced persistent threat on industrial internet of things with lateral movement modelling. *IEEE Transactions on Industrial Informatics*, IEEE, 2022.

BONACI, T.; BUSHNELL, L.; POOVENDRAN, R. Node capture attacks in wireless sensor networks: A system theoretic approach. In: IEEE. *49th IEEE Conference on Decision and Control (CDC)*. [S.l.], 2010. p. 6765–6772.

BOSSHART, P. et al. P4: Programming protocol-independent packet processors. *ACM SIGCOMM Computer Communication Review*, ACM New York, NY, USA, v. 44, n. 3, p. 87–95, 2014.

BUDIU, M.; DODD, C. The p416 programming language. *ACM SIGOPS Operating Systems Review*, ACM New York, NY, USA, v. 51, n. 1, p. 5–14, 2017.

BURHAN, M. et al. Iot elements, layered architectures and security issues: A comprehensive survey. *sensors*, MDPI, v. 18, n. 9, p. 2796, 2018.

BUTT, S. A. et al. Iot smart health security threats. In: IEEE. *2019 19th International conference on computational science and its applications (ICCSA).* [S.I.], 2019.

CERON, J. M.; MARGI, C. B.; GRANVILLE, L. Z. Mars: An sdn-based malware analysis solution. In: IEEE. *2016 IEEE Symposium on Computers and Communication (ISCC)*. [S.l.], 2016. p. 525–530.

CHAGANTI, R. et al. Deep learning approach for sdn-enabled intrusion detection system in iot networks. *Information*, MDPI, v. 14, n. 1, p. 41, 2023.

CHAHID, Y.; BENABDELLAH, M.; AZIZI, A. Internet of things protocols comparison, architecture, vulnerabilities and security: State of the art. In: *Proceedings of the 2nd International Conference on Computing and Wireless Communication Systems*. [S.l.: s.n.], 2017. p. 1–6.

CHANG, H.-F. et al. Enabling malware detection with machine learning on programmable switch. In: IEEE. *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*. [S.l.], 2022. p. 1–5.

CHARBUTY, B.; ABDULAZEEZ, A. Classification based on decision tree algorithm for machine learning. *Journal of Applied Science and Technology Trends*, v. 2, n. 01, p. 20–28, 2021.

CHIBA, Z. et al. A deep study of novel intrusion detection systems and intrusion prevention systems for internet of things networks. *Procedia Computer Science*, Elsevier, v. 210, p. 94–103, 2022.

CHIBA, Z. et al. Review of recent intrusion detection systems and intrusion prevention systems in iot networks. In: IEEE. *2022 International Conference on Software, Telecommunications and Computer Networks (SoftCOM)*. [S.l.], 2022. p. 1–6.

CUSACK, G.; MICHEL, O.; KELLER, E. Machine learning-based detection of ransomware using sdn. In: *Proceedings of the 2018 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization*. [S.l.: s.n.], 2018. p. 1–6

DARKI, A. et al. *{*IDAPro*}* for *{*IoT*}* malware analysis? In: *12th USENIX Workshop on Cyber Security Experimentation and Test (CSET 19)*. [S.l.: s.n.], 2019.

DENG, X.; MIRKOVIC, J. Polymorphic malware behavior through network trace analysis. In: IEEE. *2022 14th International Conference on COMmunication Systems & NETworkS (COMSNETS)*. [S.l.], 2022. p. 138–146.

DUANGPHASUK, S.; DUANGPHASUK, P.; THAMMARAT, C. Review of internet of things (iot): security issue and solution. In: IEEE. *2020 17th International Conference on Electrical Engineering/Electronics, Computer, Telecommunications and Information Technology (ECTI-CON)*. [S.l.], 2020. p. 559–562.

EGELE, M. et al. A survey on automated dynamic malware-analysis techniques and tools. *ACM computing surveys (CSUR)*, ACM New York, NY, USA, v. 44, n. 2, p. 1–42, 2008.

ELKHAWAS, A. I.; AZER, M. A. Security perspective in rami 4.0. In: IEEE. *2018 13th International Conference on Computer Engineering and Systems (ICCES)*. [S.l.], 2018. p. 151–156.

ENGELEN, J. E. V.; HOOS, H. H. A survey on semi-supervised learning. *Machine learning*, Springer, v. 109, n. 2, p. 373–440, 2020.

FRANÇOIS-LAVET, V. et al. An introduction to deep reinforcement learning. *Foundations and Trends® in Machine Learning*, Now Publishers, Inc., v. 11, n. 3-4, p. 219–354, 2018.

FU, Z. et al. Encrypted malware traffic detection via graph-based network analysis. In: *Proceedings of the 25th International Symposium on Research in Attacks, Intrusions and Defenses*. [S.l.: s.n.], 2022. p. 495–509.

GARCIA, S. et al. An empirical comparison of botnet detection methods. *computers & security*, Elsevier, v. 45, p. 100–123, 2014.

GARCIA, S.; PARMISANO, A.; ERQUIAGA, M. J. *IoT-23: A labeled dataset with malicious and benign IoT network traffic*. Zenodo, 2020. More details here https://www.stratosphereips.org /datasets-iot23. Disponível em: <https: //doi.org/10.5281/zenodo.4743746>.

GAURAV, A.; GUPTA, B. B.; PANIGRAHI, P. K. A comprehensive survey on machine learning approaches for malware detection in iot-based enterprise information system. *Enterprise Information Systems*, Taylor & Francis, v. 17, n. 3, p. 2023764, 2023.

GAWLIKOWSKI, J. et al. A survey of uncertainty in deep neural networks. *Artificial Intelligence Review*, Springer, v. 56, n. Suppl 1, p. 1513–1589, 2023.

GERT, G. E. R. T. Incident response analyst report 2023. Kaspersky Lab, 2024. Disponível em: <https://media.kasperskycontenthub.com/wp-content/uploads/sites/43/ 2024/05/13125640/Kaspersky-IR_Analyst_report_2023_EN.pdf>.

GKOUNTIS, C. et al. Lightweight algorithm for protecting sdn controller against ddos attacks.

In: IEEE. *2017 10th IFIP Wireless and Mobile Networking Conference (WMNC)*. [S.l.], 2017. p. 1–6.

GONCHAROV, E. Ics and ot threat predictions for 2024. Kaspersky Lab, 2024.

GOPINATH, M.; SETHURAMAN, S. C. A comprehensive survey on deep learning based malware detection techniques. *Computer Science Review*, Elsevier, v. 47, p. 100529, 2023.

HANKEL, M.; REXROTH, B. The reference architectural model industrie 4.0 (rami 4.0). *Zvei*, April, v. 2, n. 2, p. 4–9, 2015.

HASSAN, H. A. et al. Intrusion detection systems for the internet of thing: A survey study. *Wireless Personal Communications*, Springer, v. 128, n. 4, p. 2753–2778, 2023.

HASSAN, W. H. et al. Current research on internet of things (iot) security: A survey. *Computer networks*, Elsevier, v. 148, p. 283–294, 2019.

HASSIJA, V. et al. A survey on iot security: application areas, security threats, and solution architectures. *IEEE Access*, IEEE, v. 7, p. 82721–82743, 2019.

HATZIVASILIS, G. et al. The interoperability of things: Interoperable solutions as an enabler for iot and web 3.0. In: IEEE. *2018 IEEE 23rd International Workshop on Computer Aided Modeling and Design of Communication Links and Networks (CAMAD)*. [S.l.], 2018. p. 1–7.

HAUSER, F. et al. A survey on data plane programming with p4: Fundamentals, advances, and applied research. *Journal of Network and Computer Applications*, Elsevier, v. 212, p. 103561, 2023.

HOFER-SCHMITZ, K.; STOJANOVIĆ, B. Towards formal verification of iot protocols: A review. *Computer Networks*, Elsevier, v. 174, p. 107233, 2020.

HOOFNAGLE, C. J.; SLOOT, B. V. D.; BORGESIUS, F. Z. The european union general data protection regulation: what it is and what it means. *Information & Communications Technology Law*, Taylor & Francis, v. 28, n. 1, p. 65–98, 2019.

HRISTOV, M. et al. Integration of splunk enterprise siem for ddos attack detection in iot. In: IEEE. *2021 IEEE 20th International Symposium on Network Computing and Applications (NCA)*. [S.l.], 2021. p. 1–5.

JAFARIAN, T. *SDN-NF-TJ*. IEEE Dataport, 2019. Disponível em: <https://dx.doi.org/10.21227/0s6v-p761>.

JAFARIAN, T. et al. A survey and classification of the security anomaly detection mechanisms in software defined networks. *Cluster Computing*, Springer, v. 24, p. 1235–1253, 2021.

JAMALPUR, S. et al. Dynamic malware analysis using cuckoo sandbox. In: IEEE. *2018 Second international conference on inventive communication and computational technologies (ICICCT)*. [S.l.], 2018. p. 1056–1060.

JAMES, E.; RABBI, F. Fortifying the iot landscape: Strategies to counter security risks in connected systems. *Tensorgate Journal of Sustainable Technology and Infrastructure for Developing Countries*, v. 6, n. 1, p. 32–46, 2023.

JEELANI, F. et al. The detection of iot botnet using machine learning on iot-23 dataset. In: IEEE. *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*. [S.l.], 2022. v. 2, p. 634–639.

JURCUT, A. D.; RANAWEERA, P.; XU, L. Introduction to iot security. *IoT security: advances in authentication*, Wiley Online Library, p. 27–64, 2020.

KAO, D.-Y.; HSIAO, S.-C. The dynamic analysis of wannacry ransomware. In: IEEE. *2018 20th International conference on advanced communication technology (ICACT)*. [S.l.], 2018. p. 159–166.

KAPETANOVIC, D.; ZHENG, G.; RUSEK, F. Physical layer security for massive mimo: An overview on passive eavesdropping and active attacks. *IEEE Communications Magazine*, IEEE, v. 53, n. 6, p. 21–27, 2015.

KARA, I. Fileless malware threats: Recent advances, analysis approach through memory forensics and research challenges. *Expert Systems with Applications*, Elsevier, v. 214, p. 119133, 2023.

KARALE, A. The challenges of iot addressing security, ethics, privacy, and laws. *Internet of Things*, Elsevier, v. 15, p. 100420, 2021.

KAUR, S.; KUMAR, K.; AGGARWAL, N. A review on p4-programmable data planes: Architecture, research efforts, and future directions. *Computer Communications*, Elsevier, v. 170, p. 109–129, 2021.

KAZMI, S. H. A. et al. Survey on joint paradigm of 5g and sdn emerging mobile technologies: Architecture, security, challenges and research directions. *Wireless Personal Communications*, Springer, p. 1–48, 2023.

KFOURY, E. F.; CRICHIGNO, J.; BOU-HARB, E. An exhaustive survey on p4 programmable data plane switches: Taxonomy, applications, challenges, and future trends. *IEEE access*, IEEE, v. 9, p. 87094–87155, 2021.

KHAN, S.; AKHUNZADA, A. A hybrid dl-driven intelligent sdn-enabled malware detection framework for internet of medical things (iomt). *Computer Communications*, Elsevier, v. 170, p. 209–216, 2021.

KIM, H.; LEE, E. A. Authentication and authorization for the internet of things. *IT Professional*, IEEE, v. 19, n. 5, p. 27–33, 2017.

KIM, J. H. A review of cyber-physical system research relevant to the emerging it trends: industry 4.0, iot, big data, and cloud computing. *Journal of industrial integration and management*, World Scientific, v. 2, n. 03, p. 1750011, 2017.

KIM, T.-h.; RAMOS, C.; MOHAMMED, S. *Smart city and IoT*. [S.l.]: Elsevier, 2017. 159–162 p.

KOLIAS, C. et al. Ddos in the iot: Mirai and other botnets. *Computer*, IEEE, v. 50, n. 7, p. 80–84, 2017.

KOSMIDIS, K.; KALLONIATIS, C. Machine learning and images for malware detection and classification. In: *Proceedings of the 21st Pan-Hellenic Conference on Informatics*. [S.l.: s.n.], 2017. p. 1–6.

KOUNOUDES, A. D.; KAPITSAKI, G. M. A mapping of iot user-centric privacy preserving approaches to the gdpr. *Internet of Things*, Elsevier, v. 11, p. 100179, 2020.

KRAL, P. *Incident Handlerapos;s Handbook | SANS Institute — sans.org*. 2012. <https://www.sans.org/white-papers/33901/>. Acessado em 15/10/2023.

KREUTZ, D. et al. Software-defined networking: A comprehensive survey. *Proceedings of the IEEE*, Ieee, v. 103, n. 1, p. 14–76, 2014.

KRISTIYANTO, Y.; ERNASTUTI, E. Analysis of deauthentication attack on ieee 802.11 connectivity based on iot technology using external penetration test. *CommIT (Communication and Information Technology) Journal*, v. 14, n. 1, p. 45–51, 2020.

KUMAR, G. E. P.; LYDIA, M.; LEVRON, Y. Security challenges in 5g and iot networks: A review. *Secure Communication for 5G and IoT Networks*, Springer, p. 1–13, 2022.

KUMAR, M.; DUBEY, K.; PANDEY, R. Evolution of emerging computing paradigm cloud to fog: applications, limitations and research challenges. In: IEEE. *2021 11th international conference on cloud computing, data science & engineering (Confluence)*. [S.l.], 2021. p. 257–261.

KUMAR, S.; CHANDAVARKAR, B. Analysis of mirai malware and its components. In: SPRINGER. *Machine Learning, Image Processing, Network Security and Data Sciences: Select Proceedings of 3rd International Conference on MIND 2021*. [S.l.], 2023. p. 851–861.

LASI, H. et al. Industry 4.0. *Business & information systems engineering*, Springer, v. 6, p. 239–242, 2014.

LEE, E. et al. A survey on standards for interoperability and security in the internet of things. *IEEE Communications Surveys & Tutorials*, IEEE, v. 23, n. 2, p. 1020–1047, 2021.

LETTERI, I.; PENNA, G. D.; GASPERIS, G. D. Botnet detection in software defined networks by deep learning techniques. In: SPRINGER. *Cyberspace Safety and Security: 10th International Symposium, CSS 2018, Amalfi, Italy, October 29–31, 2018, Proceedings 10*. [S.l.], 2018. p. 49–62.

LI, C. et al. Securing sdn infrastructure of iot–fog networks from mitm attacks. *IEEE Internet of Things Journal*, IEEE, v. 4, n. 5, p. 1156–1164, 2017.

LIANG, W.; JI, N. Privacy challenges of iot-based blockchain: a systematic review. *Cluster Computing*, Springer, v. 25, n. 3, p. 2203–2221, 2022.

LIATIFIS, A. et al. Advancing sdn from openflow to p4: A survey. *ACM Computing Surveys*, ACM New York, NY, v. 55, n. 9, p. 1–37, 2023.

LIU, Y.; WANG, Y.; ZHANG, J. New machine learning algorithm: Random forest. In: SPRINGER. *Information Computing and Applications: Third International Conference, ICICA 2012, Chengde, China, September 14-16, 2012. Proceedings 3*. [S.l.], 2012. p. 246–252.

LOCKETT, A. Assessing the effectiveness of yara rules for signature-based malware detection and classification. *arXiv preprint arXiv:2111.13910*, 2021.

MADDU, M.; RAO, Y. N. Network intrusion detection and mitigation in sdn using deep learning models. *International Journal of Information Security*, Springer, p. 1–14, 2023.

MAEDA, S. et al. A botnet detection method on sdn using deep learning. In: IEEE. *2019 IEEE International Conference on Consumer Electronics (ICCE)*. [S.l.], 2019. p. 1–6.

MALEH, Y. et al. A comprehensive survey on sdn security: threats, mitigations, and future directions. *Journal of Reliable Intelligent Environments*, Springer, v. 9, n. 2, p. 201–239, 2023.

MAMMONE, A.; TURCHI, M.; CRISTIANINI, N. Support vector machines. *Wiley Interdisciplinary Reviews: Computational Statistics*, Wiley Online Library, v. 1, n. 3, p. 283–289, 2009.

MANAVALAN, E.; JAYAKRISHNA, K. A review of internet of things (iot) embedded sustainable supply chain for industry 4.0 requirements. *Computers & industrial engineering*, Elsevier, v. 127, p. 925–953, 2019.

MARJANI, M. et al. Big iot data analytics: architecture, opportunities, and open research challenges. *ieee access*, IEEE, v. 5, p. 5247–5261, 2017.

MCKEOWN, N. et al. Openflow: Enabling innovation in campus networks. *SIGCOMM Comput. Commun. Rev.*, ACM, New York, NY, USA, v. 38, n. 2, p. 69–74, mar. 2008. ISSN 0146-4833. Disponível em: <http://doi.acm.org/10.1145/1355734.1355746>.

MICHEL, O. et al. The programmable data plane: Abstractions, architectures, algorithms, and applications. *ACM Computing Surveys (CSUR)*, ACM New York, NY, USA, v. 54, n. 4, p. 1–36, 2021.

MODI, J. et al. Detecting ransomware in encrypted web traffic. In: SPRINGER. *Foundations and Practice of Security: 12th International Symposium, FPS 2019, Toulouse, France, November 5–7, 2019, Revised Selected Papers 12*. [S.l.], 2020. p. 345–353.

MOHANTY, J. et al. Iot security, challenges, and solutions: a review. *Progress in Advanced Computing and Intelligent Engineering: Proceedings of ICACIE 2019, Volume 2*, Springer, p. 493–504, 2021.

MORALES, E. F.; ESCALANTE, H. J. A brief introduction to supervised, unsupervised, and reinforcement learning. In: *Biosignal processing and classification using computational learning and intelligence*. [S.l.]: Elsevier, 2022. p. 111–129.

MOSER, A.; KRUEGEL, C.; KIRDA, E. Limits of static analysis for malware detection. In: IEEE. *Twenty-Third Annual Computer Security Applications Conference (ACSAC 2007)*. [S.l.], 2007. p. 421–430.

MOUSTAFA, N. et al. Explainable intrusion detection for cyber defences in the internet of things: Opportunities and solutions. *IEEE Communications Surveys & Tutorials*, IEEE, 2023.

MOZO, A. et al. A machine-learning-based cyberattack detector for a cloud-based sdn controller. *Applied Sciences*, MDPI, v. 13, n. 8, p. 4914, 2023.

MRABET, H. et al. A survey of iot security based on a layered architecture of sensing and data analysis. *Sensors*, MDPI, v. 20, n. 13, p. 3625, 2020.

MUNIRATHINAM, S. Industry 4.0: Industrial internet of things (iiot). In: *Advances in computers*. [S.l.]: Elsevier, 2020. v. 117, n. 1, p. 129–164.

MUTHANNA, M. S. A. et al. Towards sdn-enabled, intelligent intrusion detection system for internet of things (iot). *IEEE Access*, IEEE, v. 10, p. 22756–22768, 2022.

NAIK, N. et al. Embedded yara rules: strengthening yara rules utilising fuzzy hashing and fuzzy rules for malware analysis. *Complex & Intelligent Systems*, Springer, v. 7, p. 687–702, 2021.

NAMANYA, A. P. et al. The world of malware: An overview. In: IEEE. *2018 IEEE 6th international conference on future internet of things and cloud (FiCloud)*. [S.l.], 2018. p. 420–427.

NANTHIYA, D. et al. Svm based ddos attack detection in iot using iot-23 botnet dataset. In: IEEE. *2021 Innovations in Power and Advanced Computing Technologies (i-PACT)*. [S.l.], 2021. p. 1–7.

NUR, M.; WANG, Y. An overview of identity relationship management in the internet of things. In: IEEE. *2021 IEEE International Conference on Consumer Electronics (ICCE)*. [S.l.], 2021. p. 1–5.

OHA, C. V. et al. Machine learning models for malicious traffic detection in iot networks/iot-23 dataset. In: SPRINGER. *International Conference on Machine Learning for Networking*. [S.l.], 2021. p. 69–84.

OZ, H. et al. A survey on ransomware: Evolution, taxonomy, and defense solutions. *ACM Computing Surveys (CSUR)*, ACM New York, NY, v. 54, n. 11s, p. 1–37, 2022.

PALADI, N. Towards secure sdn policy management. In: IEEE. *2015 IEEE/ACM 8th International Conference on Utility and Cloud Computing (UCC)*. [S.l.], 2015. p. 607–611.

PARIS, I. L. B. M.; HABAEBI, M. H.; ZYOUD, A. M. Implementation of ssl/tls security with mqtt protocol in iot environment. *Wireless Personal Communications*, Springer, p. 1–20, 2023.

PATLE, A.; CHOUHAN, D. S. Svm kernel functions for classification. In: IEEE. *2013 International conference on advances in technology and engineering (ICATE)*. [S.l.], 2013. p. 1–9.

PENG, W. et al. Behavioral malware detection in delay tolerant networks. *IEEE Transactions on Parallel and Distributed systems*, IEEE, v. 25, n. 1, p. 53–63, 2013.

PERES, R. S. et al. Industrial artificial intelligence in industry 4.0-systematic review, challenges and outlook. *IEEE Access*, IEEE, v. 8, p. 220121–220139, 2020.

PETER, L. S.; KOBO, H.; SRIVASTAVA, V. M. A comparative review analysis of openflow and p4 protocols based on software defined networks. *Data Intelligence and Cognitive Informatics: Proceedings of ICDICI 2022*, Springer, p. 699–711, 2022.

POLLARD, W. Iot governance, privacy and security issues. *Eur. Res. Clust. Internet Things*, v. 876, p. 23–31, 2015.

PUTHAL, D. et al. Threats to networking cloud and edge datacenters in the internet of things. *IEEE Cloud Computing*, IEEE, v. 3, n. 3, p. 64–71, 2016.

RAHIM, M. A. et al. Evolution of iot-enabled connectivity and applications in automotive industry: A review. *Vehicular Communications*, Elsevier, v. 27, p. 100285, 2021.

RAHMAN, R. A.; SHAH, B. Security analysis of iot protocols: A focus in coap. In: IEEE. *2016 3rd MEC international conference on big data and smart city (ICBDSC)*. [S.l.], 2016. p. 1–7.

RAHOUTI, M. et al. Sdn security review: Threat taxonomy, implications, and open challenges. *IEEE Access*, IEEE, v. 10, p. 45820–45854, 2022.

RAMANI, S.; JHAVERI, R. H. Ml-based delay attack detection and isolation for fault-tolerant software-defined industrial networks. *Sensors*, MDPI, v. 22, n. 18, p. 6958, 2022.

RAO, S. K.; PRASAD, R. Impact of 5g technologies on industry 4.0. *Wireless personal communications*, Springer, v. 100, p. 145–159, 2018.

RAPOSO, D. et al. Industrial iot monitoring: Technologies and architecture proposal. *Sensors*, MDPI, v. 18, n. 10, p. 3568, 2018.

RATHEE, G. et al. Decision-making model for securing iot devices in smart industries. *IEEE Transactions on Industrial Informatics*, IEEE, v. 17, n. 6, p. 4270–4278, 2020.

RAYES, A. et al. Internet of things (iot) overview. *Internet of Things From Hype to Reality: The Road to Digitization*, Springer, p. 1–34, 2017.

RAZAULLA, S. et al. The age of ransomware: A survey on the evolution, taxonomy, and research directions. *IEEE Access*, IEEE, 2023.

REBALA, G. et al. Machine learning definition and basics. *An introduction to machine learning*, Springer, p. 1–17, 2019.

REN, S. et al. Global refinement of random forest. In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. [S.l.: s.n.], 2015. p. 723–730.

RIEDMANN, S.; BINDER, C.; NEUREITER, C. Incorporating design principles in industry 4.0 system architecture using reference architectures. *Automation, Robotics & Communications for Industry 4.0/5.0*, p. 295, 2024.

RIZI, M. H. P.; SENO, S. A. H. A systematic review of technologies and solutions to improve security and privacy protection of citizens in the smart city. *Internet of Things*, Elsevier, v. 20, p. 100584, 2022.

SALAHDINE, F.; HAN, T.; ZHANG, N. Security in 5g and beyond recent advances and future challenges. *Security and Privacy*, Wiley Online Library, v. 6, n. 1, p. e271, 2023.

SANTOS, C. F. G. D.; PAPA, J. P. Avoiding overfitting: A survey on regularization methods for convolutional neural networks. *ACM Computing Surveys (CSUR)*, ACM New York, NY, v. 54, n. 10s, p. 1–25, 2022.

SARICA, A. K.; ANGIN, P. A novel sdn dataset for intrusion detection in iot networks. In: IEEE. *2020 16th International Conference on Network and Service Management (CNSM)*. [S.l.], 2020. p. 1–5.

SATHYANARAYAN, V. S.; KOHLI, P.; BRUHADESHWAR, B. Signature generation and detection of malware families. In: SPRINGER. *Australasian Conference on Information Security and Privacy*. [S.l.], 2008. p. 336–349.

SHAHID, J. et al. Data protection and privacy of the internet of healthcare things (iohts). *Applied Sciences*, MDPI, v. 12, n. 4, p. 1927, 2022.

SHAKED, A. et al. Operations-informed incident response playbooks. *Computers & Security*, Elsevier, v. 134, p. 103454, 2023.

SHARAFALDIN, I.; LASHKARI, A. H.; GHORBANI, A. A. Toward generating a new intrusion detection dataset and intrusion traffic characterization. *ICISSp*, v. 1, p. 108–116, 2018.

SIKORSKI, M.; HONIG, A. *Practical malware analysis: the hands-on guide to dissecting malicious software*. [S.l.]: no starch press, 2012.

SILVA, F. S. D. et al. Securing software-defined networks through adaptive moving target defense capabilities. *Journal of Network and Systems Management*, Springer, v. 31, n. 3, p. 61, 2023.

SILVA, F. S. D. et al. A taxonomy of ddos attack mitigation approaches featured by sdn technologies in iot scenarios. *Sensors*, MDPI, v. 20, n. 11, p. 3078, 2020.

SINGH, A.; THAKUR, N.; SHARMA, A. A review of supervised machine learning algorithms. In: IEEE. *2016 3rd international conference on computing for sustainable global development (INDIACom)*. [S.l.], 2016. p. 1310–1315.

SIWAKOTI, Y. R.; RAWAT, D. B. Investigating security vulnerability related to exposure and tls ecosystem in iot devices. In: IEEE. *2023 IEEE 24th International Conference on Information Reuse and Integration for Data Science (IRI)*. [S.l.], 2023. p. 7–12.

SONI, D.; MAKWANA, A. A survey on mqtt: a protocol of internet of things (iot). In: *International conference on telecommunication, power analysis and computing techniques (ICTPACT-2017)*. [S.l.: s.n.], 2017. v. 20, p. 173–177.

SOUZA, C. H.; SILVA, F. S. D. Freki: Uma ferramenta para análise automatizada de malware. In: SBC. *Anais Estendidos do XXI Simpósio Brasileiro em Segurança da Informação e de Sistemas Computacionais*. [S.l.], 2021. p. 58–65.

SWESSI, D.; IDOUDI, H. A survey on internet-of-things security: threats and emerging countermeasures. *Wireless Personal Communications*, Springer, v. 124, n. 2, p. 1557–1592, 2022.

TAYLOR, C. R. et al. Contextual, flow-based access control with scalable host-based sdn techniques. In: IEEE. *IEEE INFOCOM 2016-The 35th Annual IEEE International Conference on Computer Communications*. [S.l.], 2016. p. 1–9.

TAYYAB, U.-e.-H. et al. A survey of the recent trends in deep learning based malware detection. *Journal of Cybersecurity and Privacy*, MDPI, v. 2, n. 4, p. 800–829, 2022.

TREADWELL, S.; ZHOU, M. A heuristic approach for detection of obfuscated malware. In: IEEE.

2009 IEEE International Conference on Intelligence and Security Informatics. [S.l.], 2009. p. 291–299.

TZAFESTAS, S. G. Ethics and law in the internet of things world. *Smart cities*, MDPI, v. 1, n. 1, p. 98–120, 2018.

UCCI, D.; ANIELLO, L.; BALDONI, R. Survey of machine learning techniques for malware analysis. *Computers & Security*, Elsevier, v. 81, p. 123–147, 2019.

USAMA, M. et al. Unsupervised machine learning for networking: Techniques, applications and research challenges. *IEEE access*, IEEE, v. 7, p. 65579–65615, 2019.

VERGÜTZ, A. et al. Data instrumentation from iot network traffic as support for security management. *IEEE Transactions on Network and Service Management*, IEEE, 2023.

VISHNU, S.; RAMSON, S. J.; JEGAN, R. Internet of medical things (iomt)-an overview. In: IEEE.

2020 5th international conference on devices, circuits and systems (ICDCS). [S.l.], 2020. p. 101–104.

VISHWAKARMA, R.; JAIN, A. K. A survey of ddos attacking techniques and defence mechanisms in the iot network. *Telecommunication systems*, Springer, v. 73, n. 1, p. 3–25, 2020.

WEINSTEIN, R. Rfid: a technical overview and its application to the enterprise. *IT professional*, IEEE, v. 7, n. 3, p. 27–33, 2005.

WOO, S.-U.; KIM, D.-H.; CHUNG, T.-M. Method of detecting malware through analysis of opcodes frequency with machine learning technique. In: SPRINGER. *Advances in Computer Science and Ubiquitous Computing: CSA-CUTE2016 8*. [S.l.], 2017. p. 1019–1024.

WUKKADADA, B. et al. Comparison with http and mqtt in internet of things (iot). In: IEEE. *2018 International Conference on Inventive Research in Computing Applications (ICIRCA)*. [S.l.], 2018. p. 249–253.

XU, L. et al. Attacking the brain: Races in the *{*SDN*}* control plane. In: *26th USENIX Security Symposium (USENIX Security 17)*. [S.l.: s.n.], 2017. p. 451–468.

YANG, X. et al. Physical security and safety of iot equipment: A survey of recent advances and opportunities. *IEEE Transactions on Industrial Informatics*, IEEE, v. 18, n. 7, p. 4319–4330, 2022.

YEGNANARAYANA, B. *Artificial neural networks*. [S.l.]: PHI Learning Pvt. Ltd., 2009.

YICK, J.; MUKHERJEE, B.; GHOSAL, D. Wireless sensor network survey. *Computer networks*, Elsevier, v. 52, n. 12, p. 2292–2330, 2008.

YUN, M.; YUXIN, B. Research on the architecture and key technology of internet of things (iot) applied on smart grid. In: IEEE. *2010 international conference on advances in energy engineering*. [S.l.], 2010. p. 69–72.

ZAMFIR, S. et al. A security analysis on standard iot protocols. In: IEEE. *2016 international conference on applied and theoretical electricity (ICATE)*. [S.l.], 2016. p. 1–6.

ZHANG, C.; CHEN, Y. A review of research relevant to the emerging industry trends: Industry 4.0, iot, blockchain, and business analytics. *Journal of Industrial Integration and Management*, World Scientific, v. 5, n. 01, p. 165–180, 2020.

ZHOU, J. et al. Evaluating the quality of machine learning explanations: A survey on methods and metrics. *Electronics*, MDPI, v. 10, n. 5, p. 593, 2021.

ZOU, J.; HAN, Y.; SO, S.-S. Overview of artificial neural networks. *Artificial neural networks: methods and applications*, Springer, p. 14–22, 2009.