UNIVERSIDADE FEDERAL DE SANTA CATARINA
CENTRO TECNOLÓGICO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

Henrique Fell Lautert

Modelo de detecção de ataques de negação de serviço em redes IoT: Uma abordagem de baixo custo computacional e de operação online na borda da rede.

Florianópolis
2023

Henrique Fell Lautert

Modelo de detecção de ataques de negação de serviço em redes IoT: Uma abordagem de baixo custo computacional e de operação online na borda da rede.

Dissertação submetida ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Santa Catarina para a obtenção do título de mestre em Ciência da Computação.
Orientador: Prof. Douglas Dyllon Jeronimo de Macedo, Dr.

Florianópolis
2023

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

Fell Lautert, Henrique
Modelo de detecção de ataques de negação de serviço em redes IoT : Uma abordagem de baixo custo computacional e de operação on-line na borda da rede / Henrique Fell Lautert ; orientador, Douglas Dyllon Jeronimo de Macedo, 2023.
101 p.

Dissertação (mestrado) - Universidade Federal de Santa Catarina, Centro Tecnológico, Programa de Pós-Graduação em Ciência da Computação, Florianópolis, 2023.

Inclui referências.

1. Ciência da Computação. 2. Ataques de Negação de Serviço. 3. Internet das Coisas. 4. Detecção de Anomalias. I. Macedo, Douglas Dyllon Jeronimo de. II. Universidade Federal de Santa Catarina. Programa de Pós-Graduação em Ciência da Computação. III. Título.

Henrique Fell Lautert

Modelo de detecção de ataques de negação de serviço em redes IoT: Uma abordagem de baixo custo computacional e de operação online na borda da rede.

O presente trabalho em nível de mestrado foi avaliado e aprovado por banca examinadora composta pelos seguintes membros:

Prof. Odorico Machado Mendizabal, Dr.
Universidade Federal de Santa Catarina (UFSC)

Prof. Roger Kreutz Immich, Dr.
Universidade Federal do Rio Grande do Norte (UFRN)

Prof. Alex Sandro Roschildt Pinto, Dr.
Universidade Federal de Santa Catarina (UFSC)

Certificamos que esta é a **versão original e final** do trabalho de conclusão que foi julgado adequado para obtenção do título de mestre em Ciência da Computação.

ICP-Edu
Documento assinado digitalmente
Marcio Bastos Castro
Data: 13/12/2023 17:08:39-0300
CPF: ***.707.720-**
Verifique as assinaturas em https://v.ufsc.br

Coordenação do Programa de
Pós-Graduação

ICP-Edu
Documento assinado digitalmente
Douglas Dyllon Jeronimo de Macedo
Data: 13/12/2023 16:33:06-0300
CPF: ***.987.372-**
Verifique as assinaturas em https://v.ufsc.br

Prof. Douglas Dyllon Jeronimo de Macedo,
Dr.
Orientador

Florianópolis, 2023.

RESUMO

O crescimento no número de dispositivos e aplicações IoT, bem como a heterogeneidade e limitação de hardware dos mesmos, dificultam a aplicação de mecanismos tradicionais de segurança. Dessa forma, a camada de IoT se tornou uma parte altamente vulnerável da rede. Neste contexto, é proposto um sistema de detecção de intrusão de baixa complexidade computacional, para reconhecimento *online* dos ataques de negação de serviço. Uma característica comum de ataques de negação de serviço é o aumento repentino no número de pacotes ou requisições. Para localizar este aumento repentino, o tráfego de rede é primeiramente filtrado por protocolo ou tipo de pacote, e então reduzido ao número de pacotes em relação ao tempo. Sobre esta série de dados, são aplicadas as técnicas de janela deslizante para formação de médias móveis, posteriormente são realizadas comparações entre as médias móveis capturadas para identificar as anomalias. Por fim é realizado uma busca pelo destino de maior recorrência somente no momento em que as anomalias são reconhecidas. Testes realizados em dados extraídos de arquivo pcap, contendo ataques realizados em dispositivos reais, demonstram a precisão no reconhecimento dos ataques. Além disso, são descritas as ferramentas e técnicas para implementação do modelo proposto em ambiente realista.

Palavras-chave: Ataques de Negação de Serviço. Detecção de Anomalia. Internet das Coisas.

ABSTRACT

The growth in the number of IoT devices and applications, as well as their heterogeneity and hardware limitations, make it difficult to apply traditional security mechanisms. In this way, the IoT layer has become a highly vulnerable part of the network. In this context, a low computational complexity intrusion detection system is proposed for *online* recognition of denial of service attacks. A common characteristic of denial of service attacks is a sudden increase in the number of packets or requests. To localize this surge, network traffic is first filtered by protocol or packet type, and then reduced to the number of packets over time. On this data series, sliding window techniques are applied to form moving averages, subsequently comparisons are made between the captured moving averages to identify anomalies. Finally, a search for the destination with the highest recurrence is carried out only when the anomalies are recognized. Tests carried out on data extracted from a pcap file, containing attacks carried out on real devices, demonstrate the accuracy in recognizing attacks. Furthermore, the tools and techniques for implementing the proposed model in a realistic environment are described.

Keywords: Denial of Service Attacks. Anomaly Detection. Internet of Things.

RESUMO EXPANDIDO

Introdução

A Internet das Coisas (IoT) já é realidade em nosso dia-a-dia. Relógios, televisores, câmeras de segurança, caixas de som com assistente virtual, são alguns dos dispositivos conectados à Internet cada vez mais comuns em nossas vidas. Além disso, temos dispositivos conectados auxiliando na automação industrial, coletando dados para administração de cidades inteligentes e aperfeiçoamento da produção agrícola. Seja pela facilidade, conforto, entretenimento, diferencial tecnológico ou produtivo, os dispositivos IoT estão em plena ascensão. O rápido crescimento combinado com a heterogeneidade e limitação de hardware dos dispositivos IoT trazem consigo problemas de segurança para as redes onde estão conectados. É desafiante pensar soluções para redes seguras, uma vez que a maioria de seus componentes não suportam mecanismos tradicionais de segurança. Diante disso, podemos observar várias pesquisas neste tema, mas poucas próximas de serem implementadas em ambientes reais. Enquanto isso, nos deparamos com ataques de impacto devastador como os ataques de negação de serviços (DoS) e sua variação de negação de serviço distribuída (DDoS). Este tipo de ataque tira proveito das grandes de dispositivos em redes IoT, utilizando estes dispositivos como robôs para atacar determinada vítima. Os ataques de DoS são difíceis de prevenir pelo fato de utilizarem pacotes que inicialmente são apenas uma parte do estabelecimento natural de conexões dos protocolos, portanto não podem ser simplesmente bloqueados. O ataque do tipo TCP-SYN, por exemplo, utiliza-se do mecanismo de estabelecimento de conexão para sobrecarregar o alvo. Bloquear este tipo de tráfego impediria o estabelecimento de conexões TCP. A definição de um limite fixo também pode impedir o tráfego benigno e prejudicar o funcionamento da rede.

Objetivos

Esta pesquisa visa a proposição de um modelo de detecção de ataques de negação de serviço para redes IoT. Já como objetivos específicos, pretende compreender o estado da arte por meio de um mapeamento sistemático da literatura, bem como propor um modelo para identificação de ataques de negação de serviço em redes IoT com baixo custo computacional, e que utilize como referência uma janela de tempo para alimentar o mecanismo de reconhecimento dos ataques. Dessa forma, se tornando capaz de operar em ambientes reais, com hardwares disponíveis na borda da rede e sem a necessidade de ler um *dataset* inteiro para diferenciar o que é um ataque e do que é tráfego benigno.

Metodologia

A pesquisa foi realizada com abordagem predominantemente quantitativa, embasada em números, mais precisamente, o número de pacotes segmentados por protocolos em relação ao tempo, para análise estatística de relações de causa e efeito. De forma complementar, foi analisado com abordagem qualitativa os desafios para implementação das propostas em ambiente real. O referencial teórico foi obtido mediante um mapeamento sistemático da literatura, com objetivo de conhecer as propostas existentes e responder às perguntas de pesquisa. Foram analisados os resultados alcançados pelos trabalhos existentes na área de análise de anomalias, buscando dados como precisão, tempo de resposta, complexidade computacional, proporção de

redução de dados e *datasets* utilizados. Por fim, a análise dos dados foi realizada com auxílio da ferramenta *WireShark*, para segmentação de protocolos e contabilização de estatísticas, linguagem *Python* para elaboração e validação do modelo, e *Jupyter* notebook para apresentação dos códigos fontes e gráficos produzidos. A eficiência do modelo de reconhecimento de anomalias proposto foi obtida através da aplicação da proposta em 3 *datasets* de acesso público, com 7 diferentes tipos de ataque e protocolos: *TCP SYN flood, Smurf(ICMP), UDP(Fraggle), SNMP reflection, SSDP reflection, HTTP flood(Mirai) e MQTT-Malária*. Somados todos os experimentos, em diferentes intervalos de tempo, de 1m, 10s, 1s, 500ms e 100ms, chegou-se a um total de 423 amostras de ataques analisadas.

Resultados e Discussão

Nesta pesquisa foi desenvolvido um modelo experimental para reconhecimento de ataques de DoS em redes IoT, por meio do método de análise de anomalias. Observando a característica comum ao diversos tipos de ataques DoS, o aumento no número de determinados tipos de pacotes em relação ao tempo, foi proposto um modelo que, após a aplicação de um filtro BPF, é alimentado por uma série de dados de um único recurso, reduzindo em até 99,72% a quantidade de dados a serem analisados para o reconhecimento de uma anomalia. O algoritmo de reconhecimento adota a técnica de janela deslizante para a formação de médias móveis e comparação de crescimento proporcional, reduzindo a complexidade computacional a escala linear, a mais baixa da literatura para este propósito, alcançando uma precisão de 97,67% com tempo de resposta de 20 segundos, compondo assim um conjunto de resultados não encontrados nas demais propostas.

Considerações finais

A crescente demanda e o ritmo acelerado de adoção de dispositivos IoT, acarreta inevitavelmente em mais dispositivos a serem explorados pelos ataques de negação de serviço. A maioria dos dispositivos IoT estão situados em redes locais, contudo o maior crescimento é observado nas redes 5G. O volume de tráfego gerado pelos ataques de negação de serviço também vem crescendo, ultrapassando atualmente a gradeza de *Terabits* por segundo, derrubando grandes provedores de serviços digitais. Como trabalhos futuros sugere-se o desenvolvimento de um extrator de estatísticas de arquivos pcap para o propósito deste modelo. Este recurso permitirá a expansão dos testes, bem como a medição de leitura e escrita de cada tempo aplicado no protótipo. Outra oportunidade de pesquisa é o aprofundamento da etapa de análise após o reconhecimento da anomalia. Nesta pesquisa foi proposto uma busca seletiva de destino, para neutralizar os efeitos do ataque apenas enquanto ao seu alvo. Como através da análise de anomalia já foi realizado um recorte no momento do ataque, pode ser realizada uma análise mais profunda, combinando possivelmente as técnicas de análise de anomalia com análise de assinaturas. Ainda que custo computacional seja mais alto, será aplicado somente em um recorte do tráfego de rede, e não de forma contínua.

Palavras-chave: Ataques de Negação de Serviço. Detecção de Anomalia. Internet das Coisas.

LISTA DE ABREVIATURAS E SIGLAS

AE	*Auto Encoder*
ANN	*Artificial Neural Network*
ARIMA	*Autoregressive Integrated Moving Average*
BPF	*Berkeley Packet Filters*
COAP	*Constrained Application Protocol*
DDoS	*Distributed Denial of Service*
DLL	*Dynamic Link Library*
DNN	*Deep Neural Network*
DNS	*Domain Name System*
DoS	*Denial Of Service*
EWMA	*Exponentially Weighted Moving Average*
ICMP	*Internet Control Message Protocol*
IDS	*Intrusion Detection System*
IoT	*Internet of Things*
IP	*Internet Protocol*
MITM	*Man in The Middle*
ML	*Machine Learning*
MQTT	*Message Queuing Telemetry Transport*
MSL	Mapeamento Sistemático da Literatura
PCA	*Principal Component Analysis*
pcap	*Packet Capture*
POSIX	*Portable Operating System Interface*
SDN	*Software Defined Network*
SNMP	*Simple Network Management Protocol*
SQL	*Structured Query Language*
SSDP	*Simple Service Discovery Protocol*
SYN	*Synchronize*
TCP	*Transmission Control Protocol*
UDP	*User Datagram Protocol*

SUMÁRIO

1 INTRODUÇÃO

A Internet das Coisas (IoT) já é realidade em nosso dia-a-dia. Relógios, televisores, câmeras de segurança, caixas de som com assistente virtual, são alguns dos dispositivos conectados à Internet cada vez mais comuns em nossas vidas. Além disso, temos dispositivos conectados auxiliando na automação industrial, coletando dados para administração de cidades inteligentes e aperfeiçoamento da produção agrícola. Seja pela facilidade, conforto, entretenimento, diferencial tecnológico ou produtivo, os dispositivos IoT estão em plena ascensão. Um relatório elaborado pela IoT Analytics indica aceleração na adoção de dispositivos IoT nos próximos anos, apesar da crise de semicondutores, é esperado um crescimento de 16% em 2023, atingindo 30 bilhões de dispositivos IoT conectados em 2027 (SATYAJIT, 2023).

1.1 CONTEXTUALIZAÇÃO DO PROBLEMA

O rápido crescimento combinado com a heterogeneidade e limitação de hardware dos dispositivos IoT trazem consigo problemas de segurança para as redes onde estão conectados. É desafiante pensar soluções para redes seguras, uma vez que a maioria de seus componentes não suportam mecanismos tradicionais de segurança (AL-FUQAHA *et al.*, 2015). Diante disso, podemos observar várias pesquisas neste tema, mas poucas próximas de serem implementadas em ambientes reais (COOK; MISIRLI; FAN, 2019).

Enquanto isso, nos deparamos com ataques de impacto devastador como os ataques de negação de serviços (DoS) e sua variação de negação de serviço distribuída (DDoS). Este tipo de ataque tira proveito das grandes quantidades de dispositivos em redes IoT, utilizando estes dispositivos como robôs para atacar determinada vítima. Em outubro de 2016, foi registrado um terabit de tráfego por segundo contra a Dyn, um importante servidor DNS (Domain Name System), impactando diretamente na disponibilidade dos serviços Twitter, Netflix, Spotify, Airbnb, Reddit, Etsy, SoundCloud and The New York Times (PERLROTH, 2016). Já em 2020, outro ataque DDoS atingiu um 2,3 Tbps contra a Amazon Web Services (PORTER, 2020). Neste mesmo ano, a bolsa de valores da Nova Zelândia foi alvo de um ataque DDoS, que resultou na interrupção do serviço por 2 dias consecutivos (BBC, 2020). Os ataques DDoS impulsionados pela IoT aumentaram 300% apenas no primeiro semestre de 2023, causando uma perda financeira global estimada em 2,5 mil milhões de dólares. Em 2023, 90% dos ataques DDoS complexos basearam-se em *botnets* (NOKIA, 2023).

Os ataques de (DoS) são difíceis de prevenir pelo fato de utilizarem pacotes que inicialmente são apenas uma parte do estabelecimento natural de conexões dos protocolos, portanto não podem ser simplesmente bloqueados (BALABAN *et al.*, 2021). O ataque do tipo TCP SYN, por exemplo, utiliza-se do mecanismo de estabelecimento

de conexão para sobrecarregar o alvo. Bloquear este tipo de tráfego impediria o estabelecimento de conexões TCP. A definição de um limite fixo também pode impedir o tráfego benigno e prejudicar o funcionamento da rede (MERGENDAHL; LI, J., 2020).

Existem diversas pesquisas utilizando técnicas de aprendizado de máquina (*Machine Learning* - ML) em *datasets* (SAGHEZCHI *et al.*, 2022) (SHARMA *et al.*, 2021) que demonstram alta precisão no reconhecimento de ataques de negação de serviço. Contudo, poucos abordam o uso de uma janela de tempo e aprendizagem *online* na leitura do *dataset* ou tráfego de rede. No mundo real, utilizar um *dataset* inteiro para realizar treinamento e só depois disso iniciar o reconhecimento dos ataques abre um espaço de vulnerabilidade e também dificulta a generalização para aplicação em diferentes redes IoT (COOK; MISIRLI; FAN, 2019).

O número de recursos extraídos e utilizados para realizar o treinamento também merece destaque. Além do alto custo computacional envolvido (LI, F. *et al.*, 2019), reunir todos os recursos e simplesmente delegá-los ao processo de *Principal Component Analysis* (PCA) para a seleção dos principais recursos pode comprometer a precisão no reconhecimento de ataques (SAGHEZCHI *et al.*, 2022). Ao avaliar processos de seleção de recursos é possível visualizar que existe um pequeno número de recursos, como endereço IP de destino, endereço IP de origem, protocolo e número de pacotes em relação ao tempo, onde há maior variação e, portanto, um ganho no potencial de reconhecimento do ataque (SAHOO; PUTHAL, 2020).

Devido a padrões de transmissões mais regulares em redes IoT, novas estratégias de reconhecimento de anomalias na borda da rede com menor custo computacional podem ser exploradas (WAN *et al.*, 2020). Em uma pesquisa específica sobre o reconhecimento de anomalias em redes IoT, (COOK; MISIRLI; FAN, 2019) aponta desafios como: redução de dados para análise e reconhecimento dos ataques com aprendizado online, utilizando apenas uma janela de tempo para identificação do ataque.

Diante deste contexto, foi definida a seguinte pergunta de pesquisa: É possível reconhecer ataques de negação de serviço e reduzir seus impactos a partir de um modelo que opere na borda da rede?

1.2 JUSTIFICATIVA

Os dispositivos IoT são geralmente concebidos para funções bem específicas, como, por exemplo, um monitor de pressão sanguínea, conectado à rede, permitindo monitoramento remoto de pacientes. Outros exemplos semelhantes são sensores de um processo industrial, câmeras IP, lampadas e tomadas inteligentes, dentre outros dispositivos presentes em nossas vidas. O fato de terem funções bem definidas reduz seu custo de produção (OJO *et al.*, 2018) e permite a interação digital ou extração de dados em áreas que antes estavam totalmente desconectadas dos sistemas e Internet.

Por outro lado, temos um número crescente de dispositivos a serem explorados pelos ataques cibernéticos.

Os ataques de negação de serviço (DoS) têm ocupado *sites* de notícias alertando para a potência, cada vez mais alta, atingindo taxas de 3,47 Tbps e até 340 milhões de pacotes por segundo (PRAMATAROV, 2022). Neste contexto, as redes e dispositivos IoT têm servido como base para amplificar estes tipos de ataques (BONASERA; CHOWDHURY; LATIF, 2021). Existem algumas razões já mapeadas na literatura para que isso aconteça, como uso de senhas padrão, vulnerabilidades de software e própria massificação que permite que uma vulnerabilidade descoberta seja expandida a dispositivos idênticos (TAWALBEH *et al.*, 2020), levam estes dispositivos a funcionarem como amplificadores para ataques DoS.

Diante disso, observamos diversas pesquisas nesta área, na sua maioria com o uso de ML. Apesar da alta precisão atingida nos modelos de ML, ainda existe uma lacuna diante da aplicabilidade, visto que os modelos necessitam de treinamento, fazem uso intenso de recursos computacionais. E, em sua maioria, são baseadas na leitura de *datasets* inteiros, sem considerar uma janela de tempo ou aprendizado *online* (COOK; MISIRLI; FAN, 2019). Neste cenário, esta pesquisa realiza um mapeamento sistemático da literatura sobre o tema e propõe um modelo de reconhecimento de ataques com segmentação de protocolos e comparação de médias móveis para identificar os ataques DoS com operação online já na borda da rede, prevenindo, dessa forma, que os ataques DoS assumam proporções devastadoras.

1.3 OBJETIVOS

Nas seções seguintes estão descritos o objetivo geral e os objetivos específicos.

1.3.1 Objetivo Geral

Esta pesquisa tem como objetivo geral a proposição de um modelo de detecção de ataques de negação de serviço para redes IoT.

1.3.2 Objetivos Específicos

- Analisar propostas existentes para identificar ataques de negação de serviço em redes IoT;
- Avaliar sistemas que operam na borda da rede e trazem dados sobre o experimento;
- Propor um modelo para identificação de ataques de negação de serviço em redes IoT com baixo custo computacional, que opere de forma online na borda da rede;

- Realizar experimentos a partir de bases de dados que contenham tráfego e ataques de rede de dispositivos reais.

1.4 CONTRIBUIÇÕES

Ao analisar como os ataques de negação de serviço acontecem e seus sintomas por meio de estatísticas do tráfego de rede, abre-se um leque de modelos para o reconhecimento dos mesmos. Diferente das redes tradicionais, o perfil dos dispositivos IoT tem seu comportamento na rede melhor definido, portanto mais viável para ser mapeado por modelos matemáticos, como nesta proposta, onde após a inicialização, o aprendizado acontece de forma *online*, sem necessidade de ler todo *dataset* para o modelo ser treinado e finalmente se tornar capaz de reconhecer os ataques.

Nesta pesquisa foi proposto um modelo denominado micro-IDS. Seu funcionamento básico consiste na preparação dos dados, onde os pacotes são segmentados por protocolo ou tipo de pacote, então contabilizados em relação ao tempo, formando assim a série de dados que alimenta o modelo. Posteriormente são formadas 3 médias móveis, e então comparada a atual, com a anterior e antepenúltima para reconhecer uma anomalia. Por fim é realizado uma busca pelo destino de maior recorrência somente no momento da anomalia. O experimento do micro-IDS obteve sucesso ao reconhecer ataques de negação de serviço priorizando a eficiência e viabilidade de implementação. A comparação entre as médias móveis é uma métrica que se demonstrou generalizável, com potencial para reconhecer novos tipos de ataques de negação de serviço, independente do protocolo. Com complexidade do $O(n)$ foi possível identificar ataques de *TCP SYN flood, Smurf(ICMP), UDP(Fraggle), SNMP reflection, SSDP reflection, HTTP flood(Mirai) e MQTT-Malária* em 3 *datasets*, em diferentes unidades de tempo, com tendência de precisão de 100% na unidade de tempo de 1 minuto, 97,67% em 10 segundos, 90% em 1 segundo, 88,89% em 500 milissegundos e 89,66% em 100 milissegundos. A neutralização do ataque na borda da rede, impede que ataques DoS e DDoS tomem proporções massivas. Sendo parte importante na redução de danos deste tipo de ataque.

A comparação das múltiplas médias móveis contribui com a capacidade de generalização. Como o limite é adaptativo, dinâmico, ele é acionado apenas quando existe um crescimento abrupto, e não quando atinge um determinado número específico, como acontece com limites fixos. O modelo contém um mecanismo de reconhecimento pré-carregado, dessa forma não necessita de centenas de épocas de aprendizado. Seu referencial é construído a partir da média atual (n) e das janelas anteriores (n-1) e (n-2). Além disso, as variáveis ΔG, o número de amostras para a formação das médias móveis (k) e taxa mínima para comparação de médias possibilitam seu funcionamento em redes IoT de diferentes proporções.

O código-fonte do experimento, bem como as estatísticas extraídas dos arquivos

contendo tráfego de rede, foram carregados para a plataforma *GitHub* (LAUTERT, H., 2023), e disponibilizados de forma pública. O uso da linguagem *Jupyter Notebook* facilita a reprodução e implementação melhorias no modelo, ou até mesmo a proposição de novos modelos utilizando as estatísticas já extraídas.

1.5 ESTRUTURA DO TRABALHO

O trabalho está segmentado por capítulos, sendo o Capítulo 2, composto pela fundamentação teórica, no Capítulo 3 é apresentado um mapeamento sistemático da literatura e a discussão sobre estado da arte. Já no Capítulo 4 é apresentada a metodologia, em seguida, no Capítulo 5 é apresentada a proposta. Por fim, no Capítulo 6 são apresentados os resultados experimentais e no nos Capítulos 7 e 8, a discussão e as considerações finais.

2 FUNDAMENTAÇÃO TEÓRICA

2.1 INTERNET DAS COISAS

O termo “Internet das Coisas” (do inglês “*Internet of Things*”), doravante designado como IoT, foi citado em 1999, em uma apresentação pelo cientista da computação Ashton *et al.* (2009). Enquanto trabalhava na Procter & Gamble, ele propôs colocar chips de identificação por radiofrequência (RFID) em produtos da cadeia de suprimentos para torná-los rastreáveis. Dez anos depois, o conceito foi publicado e explicado, descrevendo a limitação de tempo, atenção e precisão dos seres humanos em capturar dados. Em contrapartida, indicando o potencial do uso de computadores para coleta de dados de forma automatizada, reduzindo o desperdício, perda e custo das operações (ASHTON *et al.*, 2009).

Desde então, diversos autores vem refletindo sobre este novo paradigma da computação. Whitmore, Agarwal e Da Xu (2015) definem IoT como “um paradigma onde objetos do cotidiano podem ser equipados com recursos de identificação, detecção, rede e processamento que permitirá que eles se comuniquem entre si e com outros dispositivos e serviços pela Internet para atingir algum objetivo”.

Basicamente temos coisas, objetos (rastreadores de ônibus, placas informativas, sistemas de controle de temperatura, irrigação, sensores de estacionamento e de centros de distribuição, dentre outros) conectados à Internet. Estas coisas, na verdade, são dispositivos que alimentam sistemas com dados, e dessa forma podemos otimizar o uso de recursos, tempo e energia. McKinsey (2022) descreve o conceito de IoT como objetos físicos incorporados com sensores e atuadores que se comunicam com sistemas de computação por meio de redes com ou sem fio, permitindo que o mundo físico seja monitorado digitalmente ou mesmo controlado.

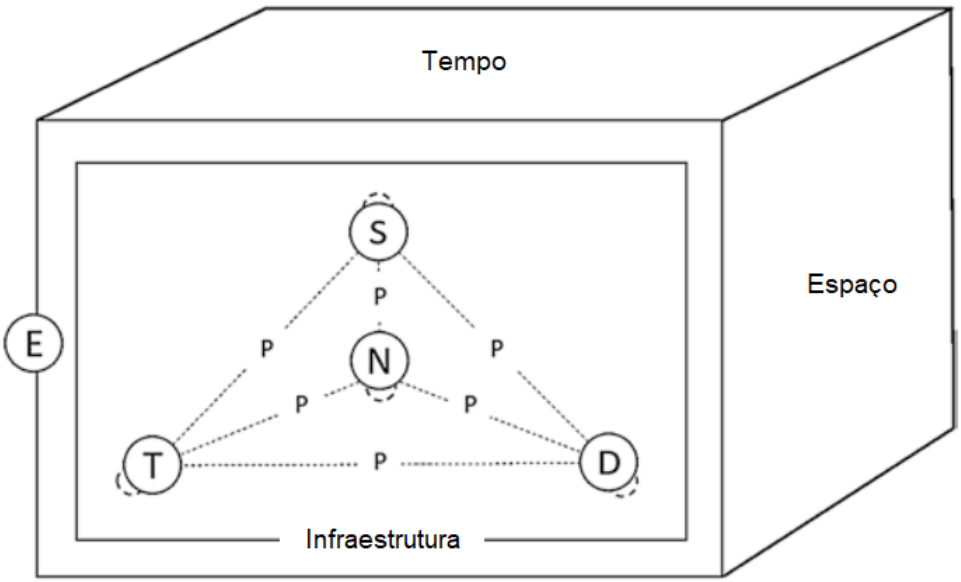
Composição de sistemas IoT

Lynn *et al.* (2020) desenvolveu um framework conceitual que engloba os componentes dos sistemas IoT, que pode ser visualizado na Figura 1. Por meio dele podemos compreender problemas relacionados à IoT e questões de pesquisa em conjunto com níveis amplamente aceitos de generalização em ambas as ciências sociais (nano, micro, macro) e ciências da computação (computação, algorítmica, física/implementação). Além disso, fornece um quadro suficientemente geral para abstração da IoT na medida em que facilita a criação de sentido sem entrar em um nível não generalizável.

Neste *framework*, cinco entidades principais são identificadas e definidas - Atores (S - *Social Actors*), coisas (T - *Things*), dados (D - *Data*), redes (N - *Network*) e eventos (E - *Events*). Cada uma dessas entidades tem uma miríade de características que podem mudar e evoluir ao longo do tempo e flexionar nossa compreensão de

como o valor pode ser gerado e capturado em diferentes unidades de análise (LYNN *et al.*, 2020).

Figura 1 – *Framework* conceitual que engloba os componentes dos sistemas IoT



Fonte: (LYNN et al., 2020)

- Atores Sociais (S): Tipicamente humanos, mas não necessariamente; O *framework* é flexível para acomodar conceitos emergentes de computadores como atores sociais;
- Coisas (T) são principalmente físicos, mas também podem ser virtuais e existem em realidade aumentada e/ou virtual. Dois requisitos funcionais chave das Coisas (T) são a coleta de dados e a conectividade da rede;
- Dados (D) aqui são artefatos discretos que podem se conectar a outras entidades, incluindo outros dados;
- Redes (N) são sistemas de entidades interconectadas;
- Eventos (E) são ocorrências de interesse em determinado momento e/ou física ou espaço virtual;
- Os processos (P) são obviamente críticos para como as entidades interagem em sistemas IoT. Eles são essenciais para como o valor é criado, capturado e entregue em sistemas IoT.

Podemos observar que os eventos são relacionados primordialmente ao tempo e espaço, estes compõem a parte mais externa, a ligação com o mundo físico nos sistemas IoT. Já na parte da infraestrutura, temos as coisas (T), atores sociais (S), rede

(N) e dados (D). Onde cada uma dessas entidades podem acionar as demais através dos processos (P). Este conceito abrange praticamente qualquer sistema IoT.

Para elucidar o conceito com um caso, tomamos como exemplo um sistema de irrigação inteligente. Os eventos (E) a serem lidos são relacionados ao tempo e espaço, ou mundo físico. Uma coisa ou sensor físico reconhece a umidade do solo, se a umidade está baixa, dispara um processo para outra coisa, ou sensor virtual, então é verificado a previsão do tempo por meio da internet, sem a necessidade de outro sensor físico compondo a solução. O sistema então poderá projetar a irrigação conforme a previsão do tempo, ou melhor, milímetros de chuva previstos, evitando desperdícios e preservando o reservatório disponível. Vejamos que os processos podem acionar direta ou indiretamente as demais entidades. Dessa forma observa-se a amplitude de possibilidades na composição dos sistemas IoT.

2.2 COMPUTAÇÃO DE BORDA

Por fazer parte desta interação do mundo físico com o mundo digital, grande parte desses dispositivos está localizada na borda da rede, compondo a computação de borda ou *Edge Computing*.

A computação de borda é a implantação de recursos de computação e armazenamento no local onde os dados são produzidos. Idealmente, isso coloca a computação e o armazenamento no mesmo ponto que a fonte de dados na borda da rede (BIGELOW, 2021). Por exemplo, um *switch* programável pode coletar e processar dados produzidos por sensores já na primeira etapa de conexão com a rede (WAN *et al.*, 2020). Os resultados de qualquer processamento desse tipo ainda podem ser enviados de volta para outro *data center*, então serem arquivados e mesclados com outros resultados de dados para análises mais amplas.

Os maiores benefícios de processar os dados próximos da origem estão relacionados com largura de banda, latência e congestão (YU *et al.*, 2017). Processar os dados na borda da rede possibilita que sejam enviados para as camadas mais altas somente os dados de maior relevância, reduzindo a quantidade total de dados trafegada (PIOLI *et al.*, 2022). Além disso, algumas decisões podem ser tomadas mais próximas do tempo real, eliminando atrasos de comunicação. Resolver parte do processo na borda da rede também evita que em caso de falha na conexão, muitos dados precisem ser transmitidos ou retransmitidos após o restabelecimento da conexão levando a uma congestão de dados.

Aspectos de segurança

Zhao *et al.* (2018) demonstra avanços nas questões de segurança e privacidade ao processar dados na borda da rede. Os dados podem ser protegidos antes de serem

enviados, ou até mesmo serem processados localmente sem necessidade de envio as camadas mais altas da rede, reduzindo as ameaças à privacidade através do processamento na borda da rede. Há ainda pesquisas recentes que apontam oportunidade de reconhecimento de anomalias na borda da rede, contribuindo para segurança e privacidade.

> Segurança e privacidade são dois dos principais desafios enfrentados atualmente pela ampla implantação de dispositivos IoT em redes de borda devido a recursos de segurança inadequados, autorização defeituosa e processos de autenticação e gerenciamento de senha fraca. Com ataques cibernéticos explorando redes IoT fracamente protegidas, dispositivos muitas vezes deixam rastros de tráfego substanciais em redes de borda, é intuitivo explorar comportamentos multidimensionais para detectar tráfego de anomalias e ameaças de segurança (WAN *et al.*, 2020).

Há alguns pontos que merecem atenção no aspecto de ameaças de segurança na camada da borda da rede. Xiao *et al.* (2019) aponta quatro aspectos a serem observados:

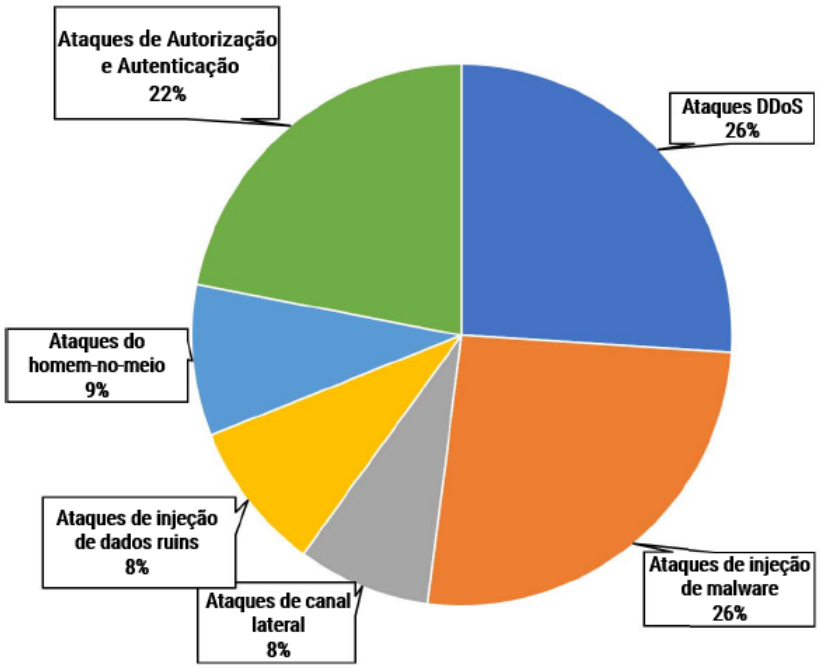
1. Poder de computação limitado: comparado a servidores na nuvem, o poder de computação de um servidor de borda é relativamente mais fraco. Portanto, um servidor de borda é mais vulnerável a ataques existentes que podem não ser mais eficazes contra um servidor em nuvem. Da mesma forma, comparado com computadores de uso geral, os dispositivos de borda têm sistemas de defesa mais frágeis; como consequência, muitos ataques que podem ser ineficazes contra computadores podem representar sérias ameaças aos dispositivos periféricos.
2. Falta de percepção do ataque: Ao contrário dos computadores de uso geral, a maioria dos dispositivos IoT não tem interfaces de usuário, embora alguns podem ter telas brutas de diodo emissor de luz (LED). Portanto, um usuário pode ter conhecimento limitado sobre a condição de execução de um dispositivo, por exemplo, se ele foi desligado ou comprometido. Assim, mesmo que um ataque esteja ocorrendo em um dispositivo de borda, a maioria dos usuários pode não ser capaz de discerni-lo.
3. Heterogeneidade de sistema operacional e protocolo: ao contrário de computadores de uso geral que tendem a usar sistemas operacionais padrão e protocolos de comunicação como *Portable Operating System Interface* (POSIX) (STALLMAN, 2008), a maioria dos dispositivos de borda tem sistemas operacionais e protocolos diferentes sem regulamentação padronizada. Este problema leva diretamente às dificuldades de projetar um sistema unificado de proteção para computação de borda.
4. Controle de acesso de granulação grossa: o modelo de controle de acesso projetados para os computadores de uso geral e computação em nuvem

consistem principalmente em quatro tipos de permissões: sem leitura e gravação, somente leitura, somente gravação, leitura e gravação. Tal modelo nunca seria satisfatório em computação de borda devido a sistemas mais complicados e seus aplicativos habilitados, que exigem controle de acesso refinado, que deve lidar com questões como “quem pode acessar quais sensores fazendo o quê, quando e como”. Infelizmente, os modelos atuais de controle de acesso são principalmente de granulação grossa.

Tipos de ataques na borda da rede

A maioria dos ataques que acontecem na borda da rede, podem ser classificados em 6 categorias, ataques DDoS, ataques de canal lateral (*side-channel*), ataques de injeções de *malware* (*malware injection*), ataques de autenticação e autorização (*authentication and authorization*), ataques do homem-no-meio (*man-in-the-middle*), e ataques de injeção de dados mal-formados (*bad-data injection*). Dentre eles, o ataque do tipo de DDoS é mais comum, atingindo 26%, ao lado de injeções de malware, também com 26%. A proporção pode ser visualizada na Figura 2 (XIAO *et al.*, 2019). Nas subseções a seguir serão descritos estes tipos de ataques, com um detalhamento maior nos ataques de negação de serviço, na Seção 2.3, o qual é o foco da proposta nesta pesquisa.

Figura 2 – Porcentagens dos tipos de ataques direcionados a infraestruturas de computação de borda.



Fonte: (XIAO *et al.*, 2019).

Ataques de canal lateral (*side-channel*)

Os ataques de canal lateral estão intimamente relacionados à existência de fenômenos fisicamente observáveis, causados pela execução de tarefas computacionais em dispositivos microeletrônicos (STANDAERT, 2010). Um ataque de canal lateral é baseado em informações extras que podem ser coletadas devido à maneira como um protocolo ou algoritmo é implementado. Informações de tempo, consumo de energia, vazamentos eletromagnéticos e som são exemplos de informações extras que podem ser exploradas para facilitar ataques de canal lateral (SPREITZER *et al.*, 2017). Em vez de atacar diretamente o próprio algoritmo, os ataques de canal lateral visam a implementação ou as características físicas do sistema, aproveitando o vazamento não intencional de informações durante o processo de computação.

- Ataques de temporização: Esses ataques exploram variações no tempo de execução ou atrasos em um sistema para inferir informações sobre as operações criptográficas que estão sendo executadas. Ao medir o tempo necessário para executar determinadas operações, um invasor pode deduzir informações sobre a chave usada na computação (JIANG; FEI; KAELI, 2017).
- Ataques de análise de energia: os ataques de análise de energia analisam os padrões de consumo de energia de um dispositivo durante operações criptográficas. Ao monitorar o consumo de energia, um invasor pode deduzir informações sobre as operações internas e extrair chaves secretas ou outros dados confidenciais (CHARI *et al.*, 1999).
- Ataques de radiação eletromagnética: Esses ataques envolvem a captura e análise de emissões eletromagnéticas de um dispositivo durante sua operação. Ao estudar a radiação eletromagnética, um invasor pode obter informações sobre os cálculos internos e recuperar informações. (STANDAERT, 2010).

Ataques de injeção de *malware* (*malware injection*)

A injeção de *malware* refere-se ao processo de introdução de código ou *software* malicioso em um sistema ou programa com a intenção de comprometer sua segurança e obter acesso ou controle não autorizado. Normalmente envolve a inserção ou modificação de código existente, ou a adição de um novo código que executa ações maliciosas (MONNAPPA, 2018). A injeção de *malware* pode assumir várias formas:

- Injeção de código: envolve a inserção de código malicioso em uma seção vulnerável de um programa, geralmente explorando vulnerabilidades de *software* ou manipulação de entrada insegura (NOMAN; ABU-SHARKH, 2023).

O código injetado pode executar ações não autorizadas, como roubar dados, manipular funcionalidades ou fornecer entrada para acesso remoto.

- Injeção de (*Structured Query Language*) SQL: Este é um tipo específico de injeção de código que visa aplicativos da Web com conexões de banco de dados vulneráveis. Ao inserir instruções SQL maliciosas nas entradas do usuário, um invasor pode manipular consultas de banco de dados, recuperando informações confidenciais ou modificando dados (GOWTHAM; PRAMOD, 2021).
- Injeção de *Dynamic Link Library* (DLL): a injeção de DLL é uma técnica em que um código mal-intencionado é injetado em um processo em execução, forçando-o a carregar um arquivo DLL mal-intencionado. Isso permite que o código injetado seja executado dentro do contexto do processo de destino, permitindo ações como capturar pressionamentos de tecla, monitorar atividades do sistema ou obter acesso não autorizado (KIM; SHIN; SEO, 2022).
- *Cross-site scripting* (XSS): Esta forma de injeção ocorre em aplicativos da web e envolve a injeção de *scripts* maliciosos em páginas da web visualizadas por outros usuários (RODRIGUEZ *et al.*, 2020). Quando esses *scripts* injetados são executados no navegador da vítima, eles podem roubar informações, sequestrar sessões do usuário ou realizar ações não autorizadas em nome do usuário.

O impacto da injeção de *malware* pode ser grave, variando de acesso não autorizado a informações confidenciais, interrupção do sistema, perda financeira ou até mesmo controle completo de sistemas comprometidos. A proteção contra a injeção de *malware* envolve o emprego de práticas seguras de codificação, atualizações regulares de *software* e validação de entrada de dados robusta (NOMAN; ABU-SHARKH, 2023).

Ataques de autenticação e autorização (*authentication and authorization*)

Os ataques de autenticação e autorização visam obter acesso a recursos sem as credenciais corretas. Autenticação refere-se especificamente a como uma aplicação determina quem você é, e autorização refere-se a aplicação limitando seu acesso apenas ao que você deve acessar (FAIRCLOTH, 2016). No contexto de IoT, além da limitação de hardware para trabalhar com mecanismos tradicionais de segurança, observa-se um problema de negligência da segurança. Em uma pesquisa realizada por (TANKARD, 2015), constatou-se que 80% dos dispositivos permitem o uso de senhas fracas, e 70% não criptografam os dados trafegados na rede. Cenário que permite até mesmo a falsificação de dispositivos IoT, onde os invasores podem tentar representar dispositivos IoT autorizados manipulando identificadores ou credenciais

de dispositivos, contornando as verificações de autorização e obtendo acesso não autorizado (RAJASHREE; SOMAN; SHAH, 2018).

Ataques do homem-no-meio (*man-in-the-middle*)

Um ataque *man-in-the-middle* (MITM) é um termo geral para quando o atacante se posiciona em uma conversa entre um usuário e uma aplicação, seja para capturar informações ou para se passar por uma das partes, fazendo parecer uma troca normal de informações. O atacante MITM pode interceptar, modificar, alterar ou substituir o tráfego de comunicação das vítimas alvo (isso distingue um MITM de um simples vazamento de informações). Além disso, as vítimas desconhecem o intruso, acreditando assim que o canal de comunicação é protegido (CONTI; DRAGONI; LESYK, 2016).

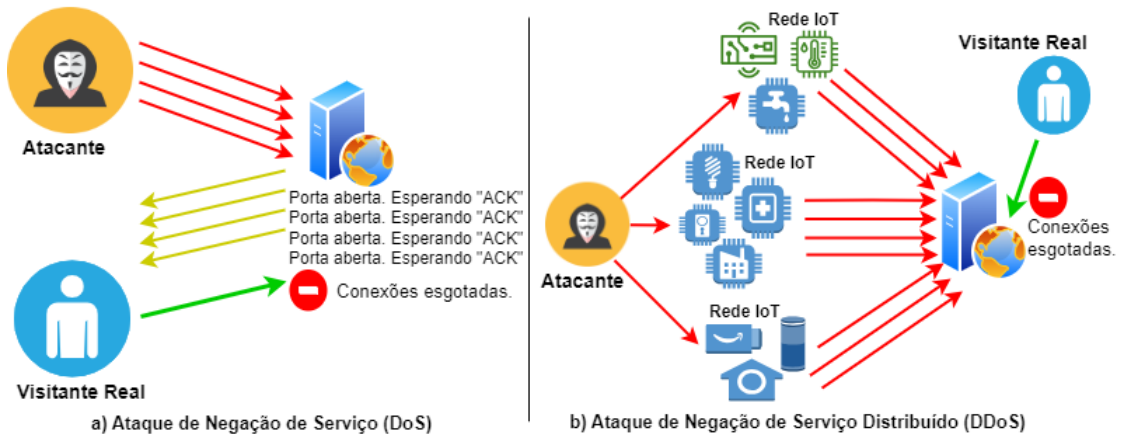
2.3 ATAQUES DE NEGAÇÃO DE SERVIÇO

Tecnicamente o objetivo de um ataque de negação de serviço é sobrecarregar o alvo, tornando-o indisponível, caracterizando dessa forma a negação do serviço. Para isto, o atacante dispara um altíssimo número de requisições fazendo o alvo responder lentamente, até se tornar totalmente incapaz de responder às requisições legítimas. Uma variação desta técnica são os ataques de negação de serviço distribuídos (DDoS), onde vários dispositivos são utilizados para amplificar o efeito do ataque (SHARAFALDIN *et al.*, 2019).

O crescimento acelerado no número de dispositivos conectados por meio da Internet das Coisas tem formado um ambiente propício para ataques DDoS massivos. Os dispositivos IoT geralmente usam senhas padrão e não possuem mecanismos tradicionais de segurança, tornando-os vulneráveis à exploração. A infecção de dispositivos IoT passa geralmente despercebida pelos usuários, e um invasor pode facilmente comprometer centenas de milhares desses dispositivos para realizar um ataque em alta escala sem o conhecimento dos proprietários do dispositivo (CETINKAYA; ISHII; HAYAKAWA, 2019).

Utilizar pequenos dispositivos de forma distribuída, além de potencializar o ataque, aumenta a dificuldade de reconhecimento, pois a verdadeira fonte do ataque torna-se mais difícil de identificar. Na Figura 3 (a), podemos visualizar um ataque DoS, onde o atacante esgota os recursos do servidor, sobrecarregando o servidor e tornando o serviço indisponível. Já na Figura 3 (b) esta mesma técnica realizada de forma distribuída, gerando uma sobrecarga ainda maior ao alvo do ataque. Os tipos de ataques de negação de serviço abordados nesta pesquisa serão descritos nas subseções seguintes.

Figura 3 – Ataques de negação de serviço



Fonte: Elaborado pelo autor.

Aperto de mão de três vias e o ataque de inundação TCP SYN

Quando um sistema (chamado de cliente) tenta estabelecer uma conexão TCP com um sistema que fornece um serviço (o servidor), o cliente e o servidor trocam uma sequência definida de mensagens, também conhecido como *Three-way Handshake*, que pode ser traduzido como um “aperto de mão de três vias” (CONRAD; MISENAR; FELDMAN, 2014).

O cliente começa enviando uma mensagem de sincronização (SYN) para o servidor. O servidor então reconhece a mensagem SYN enviando uma mensagem de sincronização reconhecida SYN-ACK para o cliente. O cliente então termina de estabelecer a conexão respondendo com uma mensagem de reconhecimento ACK (HSU *et al.*, 2016). A conexão entre o cliente e o servidor é então aberta e os dados específicos do serviço podem ser trocados entre o cliente e o servidor.

O potencial de abuso surge no ponto em que o sistema do servidor envia uma confirmação (SYN-ACK) de volta ao cliente, mas ainda não recebeu a mensagem ACK, caracterizando uma ligação semi-aberta. O servidor construiu em sua memória uma estrutura de dados que descreve todos as conexões pendentes. Essa estrutura de dados é de tamanho finito e pode transbordar criando intencionalmente muitas conexões parcialmente abertas (LONG; THOMAS, 2001).

A criação de conexões semi-abertas é facilmente realizada com falsificação de IP. O sistema de ataque envia mensagens SYN para o sistema servidor da vítima; estes parecem ser legítimos, mas, na verdade, fazem referência a um sistema cliente que é incapaz de responder às mensagens SYN-ACK. Isso significa que a mensagem ACK final nunca será enviada ao sistema do servidor vítima, em alguns casos, o sistema pode esgotar a memória, travar ou ficar inoperante.

Smurf (ICMP)

No ataque do tipo *Smurf*, uma requisição ICMP (*Internet Control Message Protocol*) é enviada com o IP de origem alterado para o IP da vítima. Para amplificar o efeito, a requisição é enviada em *broadcast*, que faz a requisição ser enviada para todos os endereços conectados na rede (HADDADI; BEGHDAD, 2018). Dessa forma, todos os hospedeiros irão retornar a resposta desta requisição para o IP da vítima, sobrecarregando ou até mesmo tornando o serviço indisponível.

UDP (Fraggle)

Esses ataques são variantes do ataque Smurf para o protocolo UDP (*User Datagram Protocol*). A diferença é que os pacotes UDP, após utilizado o IP *Spoofing* (IP de origem alterado para o IP da vítima), são enviados para a porta 7 do protocolo (porta de eco). Ao receber esse pacote, o comutador da rede local retransmite o pacote de requisição para todos os nós da rede e as respostas são enviadas para a vítima (GOMES; ARAUJO, M. S. d. A.; CAMPOS, 2015).

Se inúmeros pacotes forem enviados simultaneamente, ocorre uma inundação de pacotes UPD, afetando o funcionamento normal dos dispositivos de rede. Este problema ocorreu em dispositivos Cisco. Muitos fornecedores habilitam pequenos servidores por padrão para diagnóstico de rede ou gerenciamento de dispositivos, o que resulta em possíveis ataques. Por exemplo, durante um ataque da Pepsi, os invasores enviaram um grande número de pacotes para uma porta de diagnóstico de um dispositivo Cisco, causando a negação de serviço do dispositivo (HUAWEIDOCS, 2023).

SSDP Reflection

Os ataques do tipo *Simple Service Discovery Protocol* (SSDP) *reflection*, utilizam a mensagem do tipo *M-SEARCH* como requisição. O objetivo original do protocolo é buscar serviços de outros dispositivos, como resposta pode receber várias mensagens com informações dos serviços existentes em cada dispositivo. Para compor o ataque, o atacante altera o IP de origem para o IP da vítima, dessa forma faz que com as mensagens de resposta de todos os dispositivos sejam refletidas para um único alvo para sobre carregar o IP da vítima (LEE, Y.-j.; CHAE; LEE, K.-w., 2021).

SNMP Reflection

O protocolo *Simple Network Management Protocol* (SNMP) é utilizado para gerenciamento de dispositivos em rede, nele é definido como os dispositivos gerenciados se comunicam e trocam informações. Os dispositivos gerenciados executam agentes e mantêm um banco de dados, o *Management Information Base* (MIB), que contém variáveis que representam os dados de gerenciamento. Essas variáveis podem ser

usadas para monitoramento do dispositivo ou rede e para invocar ações. Os pedidos de consulta são feitos por mensagens enviadas via porta UDP 161 (GONDIM; OLIVEIRA ALBUQUERQUE; OROZCO, 2020).

Na composição do ataque, além da alteração do IP de origem com o IP da vítima, o atacante pode aumentar o volume de informações requisitadas com a mensagem do tipo *GetBulkRequest*, que retorna dados de várias variáveis (ARUKONDA; SINHA, 2015). Deste modo, a vítima receberá diversas informações do SNMP, podendo sobrecarregar, ou até mesmo tornar o serviço indisponível.

MQTT-Malaria

O *Message Queuing Telemetry Transport* (MQTT) é um protocolo de comunicação com foco em IoT, que funciona em cima do protocolo TCP/IP. Um sistema MQTT se baseia na comunicação entre cliente e servidor, em que o primeiro pode realizar tanto postagens, quanto captação de informação e o segundo administra os dados a serem recebidos e enviados. Para isso, é utilizado o paradigma chamado *Publish/Subscribe*(Publicar/Assinar). O protocolo se popularizou pela simplicidade, baixo consumo de dados e pela possibilidade comunicação bilateral (NERI; LOMBA; BULHÕES, 2019).

Com objetivo de testar a escalabilidade e carga de sistemas que utilizam o protocolo MQTT, (PALSSON, 2019) desenvolveu a ferramenta MQTT-Malaria, capaz de gerar um altíssimo número de mensagens, com opção de definir tamanhos conforme o sistema a ser testado. Caso a configuração seja definida com parâmetros excedam a capacidade do sistema alvo, a ferramenta poderá causar a indisponibilidade do serviço.

HTTP Flood

Inundação HTTP refere-se a um tipo de ataque de negação de serviço distribuído (DDoS), em que vários computadores ou dispositivos comprometidos inundam um servidor de destino com um excesso de solicitações HTTP em um curto período. Essas solicitações são normalmente projetadas para sobrecarregar os recursos do servidor de destino, como capacidade de processamento, memória e largura de banda da rede, dificultando a resposta do servidor às solicitações legítimas do usuário e fazendo com que ele fique lento ou sem resposta (MUNIVARA PRASAD; RAMA MOHAN REDDY; VENUGOPAL RAO, 2017).

Em um ataque de inundação HTTP, o invasor visa esgotar a capacidade do servidor alvo de lidar com conexões de entrada e processar solicitações HTTP, levando, em última análise, à interrupção do serviço. Isto pode afetar a disponibilidade de *websites*, aplicações *web* e serviços *online*, podendo causar perdas financeiras, danos à reputação e insatisfação do usuário (RAZUMOV *et al.*, 2020).

Mirai Botnet

Mirai é um tipo de malware usado para criar uma *botnet*, que é uma rede de dispositivos comprometidos controlados por uma única entidade ou invasor. A Mirai ganhou atenção significativa no final de 2016, quando foi responsável pelo lançamento de ataques massivos de negação de serviço distribuído (DDoS) que interromperam vários sites e serviços *online* (WOOLF, 2016).

Mirai visa especificamente dispositivos de Internet das Coisas (IoT), como câmeras conectadas à Internet, roteadores, DVRs e outros dispositivos inteligentes que muitas vezes carecem de medidas de segurança adequadas. Esses dispositivos são comprometidos pela exploração de nomes de usuário e senhas padrão ou vulnerabilidades conhecidas, transformando-os em *"bots"* que podem ser controlados remotamente pelo invasor (KOLIAS *et al.*, 2017).

Depois que um dispositivo é infectado pelo Mirai, ele se torna parte da *botnet* Mirai. O controlador da *botnet*, muitas vezes referido como *"botmaster"* ou *"bot herder"*, pode usar os dispositivos comprometidos para lançar ataques DDoS em grande escala contra alvos específicos. Esses ataques envolvem sobrecarregar os servidores do alvo com uma inundação de tráfego da *botnet*, tornando os serviços do alvo inacessíveis (PRADO *et al.*, 2018).

2.4 SISTEMAS DE DETECÇÃO DE INTRUSÃO

Os sistemas de detecção de intrusão (IDS) são sistemas de software ou hardware que automatizam o processo de monitoramento dos eventos que ocorrem em um sistema de computador ou rede, analisando-os em busca de sinais de problemas de segurança (BACE; MELL *et al.*, 2001). Atualmente os principais métodos de detecção são: Reconhecimento baseado em assinaturas (*Signature-based*), em análise de protocolo (*Stateful protocol analysis*) e análise de anomalias (*Anomaly-based*) (LIAO *et al.*, 2013).

Reconhecimento baseado em assinaturas (*Signature-based*)

Na detecção pelo reconhecimento de assinaturas, o IDS trabalha com um banco de dados de ataques conhecidos, chamados também de assinaturas. Então compara o conteúdo da rede, com as assinaturas e caso reconheça, classifica o evento como um ataque. Embora seja simples e efetivo em reconhecer ataques já conhecidos, é ineficaz para detectar ataques desconhecidos e variantes de ataques conhecidos, outro ponto negativo é dificuldade de manter as assinaturas/padrões atualizados (HAMZA *et al.*, 2019).

> (...)é muito difícil (se não impossível, conforme a disponibilidade atual no mercado) implantar com eficiência um IDS baseado em assinatura em um

> dispositivo de ponta típico de um sistema IoT. De fato, um IDS baseado em assinatura não pode detectar ataques desconhecidos, pois pode não ter essas definições em seu vocabulário de ataques. Pior ainda, em um ambiente IoT, um IDS é implantado remotamente em gateways IoT de baixo custo, se não em dispositivos finais menores. Isso torna mais difícil atualizar regularmente as definições de ataques, conforme exigido em uma abordagem baseada em assinatura (ESKANDARI *et al.*, 2020).

Análise de protocolo (*Stateful protocol analysis*)

A análise de protocolo estável depende de perfis universais desenvolvidos pelo fornecedor que especificam como determinados protocolos devem ou não ser usados. Na análise de protocolo o IDS é capaz de entender e rastrear o estado da rede, transporte e aplicação. É um método que parte de um conhecimento detalhado do protocolo, porém demanda um maior consumo de recurso para desempenhar a análise, e também pode não reconhecer ataques que utilizam de etapas de estabelecimento de conexão classificadas como benignas (YANG *et al.*, 2014).

Análise de anomalias (*Anomaly-based*)

Na detecção de ataques por análise de anomalias, os modelos de reconhecimento primeiramente apreendem ou consideram padrões normais e esperados da rede, e se algum evento diferir é classificado como uma anomalia, podendo ser reportado para o administrador da rede, ou até mesmo aplicar uma regra contra o ataque (DOSHI; MOZAFFARI; YILMAZ, 2019). É o melhor método para detectar ataques ainda não conhecidos, porém pode gerar falsos positivos, além disso, fica indisponível durante o tempo de aprendizado, sendo assim um desafio a ser superado. É muito usado no contexto de ataques de negação de serviço, uma vez que a característica principal destes ataques é um crescimento abruto em determinados tipos de pacotes (COOK; MISIRLI; FAN, 2019).

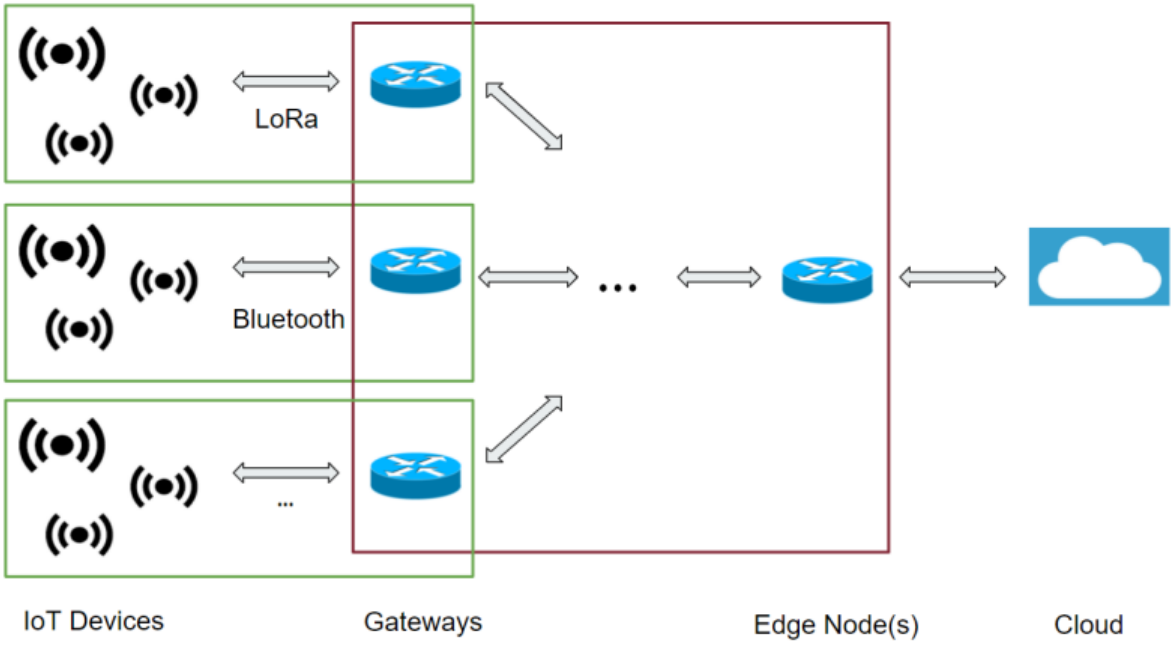
IDS no contexto de IoT

A literatura recente é rica em trabalhos de pesquisa que lidam com IDSs inteligentes operando em redes tradicionais. Contudo, menos foi feito especificamente para IDSs em ambientes IoT. Historicamente, o maior obstáculo ao lidar com IDSs implantados em *gateways* IoT é representado pelo poder computacional limitado de tais dispositivos, o que dificulta a execução de um IDS completo (ESKANDARI *et al.*, 2020).

Acima dos *gateways* IoT, na topologia de rede, estão localizados os roteadores de borda, que possuem capacidade de lidar com as demandas de um IDS projetados para esta área da rede, conforme a Figura 4. Dentro desta concepção, uma recomendação é que não seja demasiado específico para tecnologias de comunicação IoT, para

poder suportar dispositivos heterogêneos, com diferentes tecnologias de comunicação. Ainda na Figura 4, é destacada a rede do dispositivo, em verde, e a rede de borda ou *Edge Network* em vermelho.

Figura 4 – Arquitetura de rede de um sistema IoT habilitado para borda.



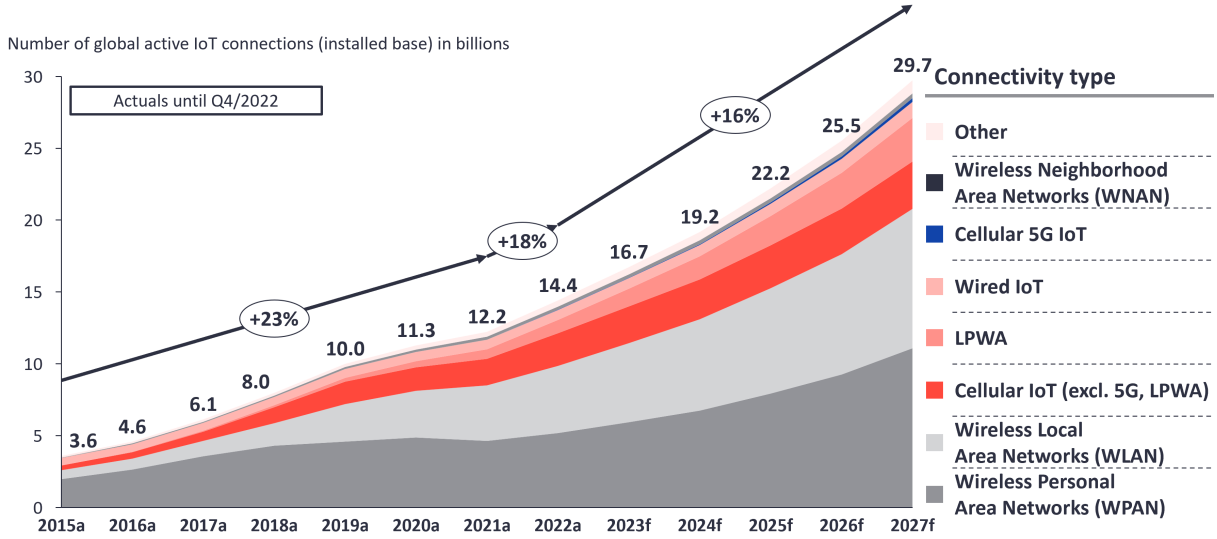
Fonte: (SPADACCINO; CUOMO *et al.*, 2022).

> Um IDS específico de IoT visa dispositivos usando uma tecnologia de comunicação específica, como 6LoWPAN, BLE, LoRaWAN, etc. Esta classe de IDS deve ser implantada na mesma rede do dispositivo. Eles geralmente realizam suas previsões com base em mensagens enviadas pelos dispositivos IoT aproveitando as informações de controle da tecnologia específica, como verificação do cumprimento do protocolo. Por outro lado, os IDSs agnósticos não dependem de uma tecnologia específica. Eles utilizam informações disponíveis independentemente de qual tecnologia é usada atualmente pelos dispositivos, como tráfego TCP/IP. Esta classe de IDS é adequada para ser usado na borda da rede, uma vez que pode lidar com tráfego gerado por dispositivos heterogêneos que utilizam diferentes tecnologias de comunicação (SPADACCINO; CUOMO *et al.*, 2022).

2.5 REDUÇÃO DE DANOS NA BORDA DA REDE

Conforme o estudo realizado pela IoT Analytics (SATYAJIT, 2023), a maior concentração de dispositivos IoT estão conectados à Internet por meio de redes *Wireless Area Network* (WAN) e *Wireless Personal Area Network* (WPAN). Na Figura 5, é possível visualizar o crescimento dos dispositivos IoT, onde as áreas WAN e WPAN respondem pela maior parte dos dispositivos.

Figura 5 – IoT Analytics - State of IoT 2023



Fonte: (SATYAJIT, 2023).

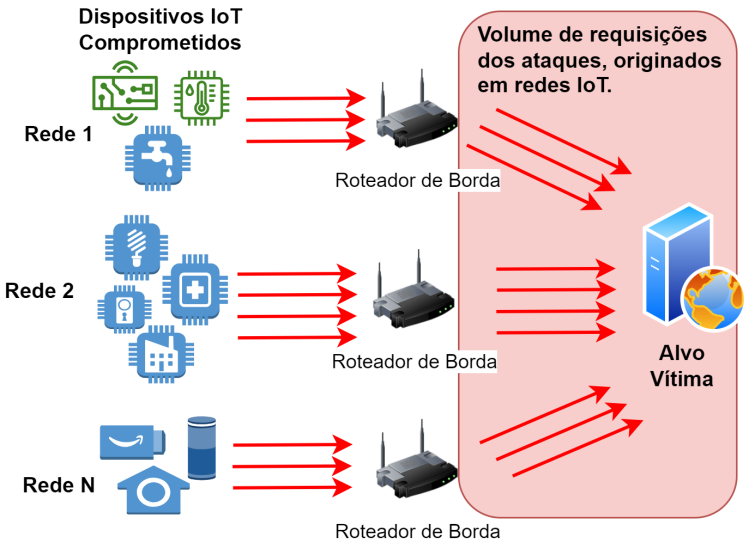
Neste contexto, a proposta desta pesquisa está posicionada na borda da rede, mais precisamente em roteadores de borda, os primeiros concentradores das conexões à Internet. Wan *et al.* (2020) aponta para padrões de transmissões segmentando protocolos nesta área da rede, o que permite previsibilidade e novas formas de reconhecimento de anomalias.

> (...) a plataforma baseada em roteador de borda tem uma visão mais detalhada e abrangente do tráfego IoT. Mais especificamente, o roteador de borda pode ver o tráfego de entrada e saída não traduzido de e para um dispositivo IoT. O tráfego interno da LAN na rede de borda também é visível para a interface interna do roteador de borda. Além disso, as soluções baseadas em roteadores de borda são transparentes para dispositivos IoT e, portanto, não há necessidade de os usuários instalarem ou atualizarem pacotes e aplicativos adicionais em dispositivos IoT. O roteador de borda é um ponto de verificação de segurança central em um local ideal para controle e aplicação de políticas. Esses pontos fortes exclusivos nos motivam a desenvolver plataformas de medição de tráfego baseadas em roteador para capturar, armazenar, caracterizar e minerar padrões de tráfego de IoT dispositivos em redes de borda (WAN *et al.*, 2020).

Reduzir os danos de ataques DDoS já na borda da rede, permite que os pacotes maliciosos sejam barrados no roteador de borda, o que impede que o ataque ganhe volume para alvos externos. Nas Figuras 6 e 7 é possível visualizar o posicionamento da proposta em pesquisa. Na Figura 6, observamos os ataques originados de diferentes redes IoT, atingindo grandes volumes. Já na Figura 7, os ataques sendo controlados na borda da rede, impedindo de criar um grande volume até o alvo ou vítima.

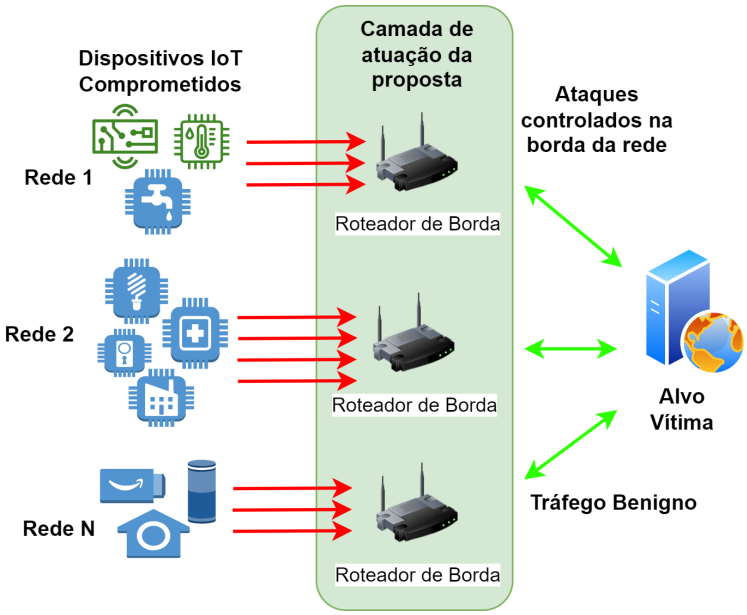
Os ataques de negação de serviço, são relatados nas 3 camadas de computação, Computação de Borda ou *Edge Computing*, Computação de Névoa ou *Fog*

Figura 6 – Concentração de requisições originadas de redes IoT



Fonte: Elaborado pelo autor.

Figura 7 – Posicionamento da Proposta



Fonte: Elaborado pelo autor.

Computing e Computação de Nuvem ou *Cloud Computing* (PARIKH *et al.*, 2019). A camada da borda, contendo dispositivos IoT, serve como amplificador dos ataques, sendo a base para o ataque ganhar força. Uma vez que o atacante consegue comprometer dispositivos na borda da rede, o ataque é direcionado geralmente a um alvo externo (RAFIQUE *et al.*, 2019). Cada dispositivo IoT comprometido, envia uma requisição ao alvo. Então, na visão do alvo, não há propriamente um ataque, há apenas

requisições que fazem parte dos protocolos, vindo de diferentes lugares ou dispositivos. O que derruba o alvo é a sobrecarga, o volume de requisições atingidas, por ele estar acontecendo de forma distribuída. Reduzir este problema na borda, evita justamente esse avolumamento de requisições, além de reduzir os danos do ataque, evita que várias requisições subam para camadas mais altas, realizando assim um uso eficiente dos recursos (SANTOS; MONTEIRO; ENDO, 2020).

2.6 ANÁLISE DE TRÁFEGO COM WIRESHARK

O *Wireshark* é o principal analisador de protocolo de rede da atualidade. Ele permite que você veja o que está acontecendo em sua rede em um nível granularizado. É o padrão de fato em muitos setores e instituições educacionais (COMBS, 2023). Os ataques de rede podem ser majoritariamente identificados observando o tráfego de entrada e saída, porque um comportamento não usual é resultado de padrões suspeitos de tráfego. Por exemplo, os seguintes eventos de ataques sempre deixarão rastros nos pacotes capturados (NDATINYA *et al.*, 2015):

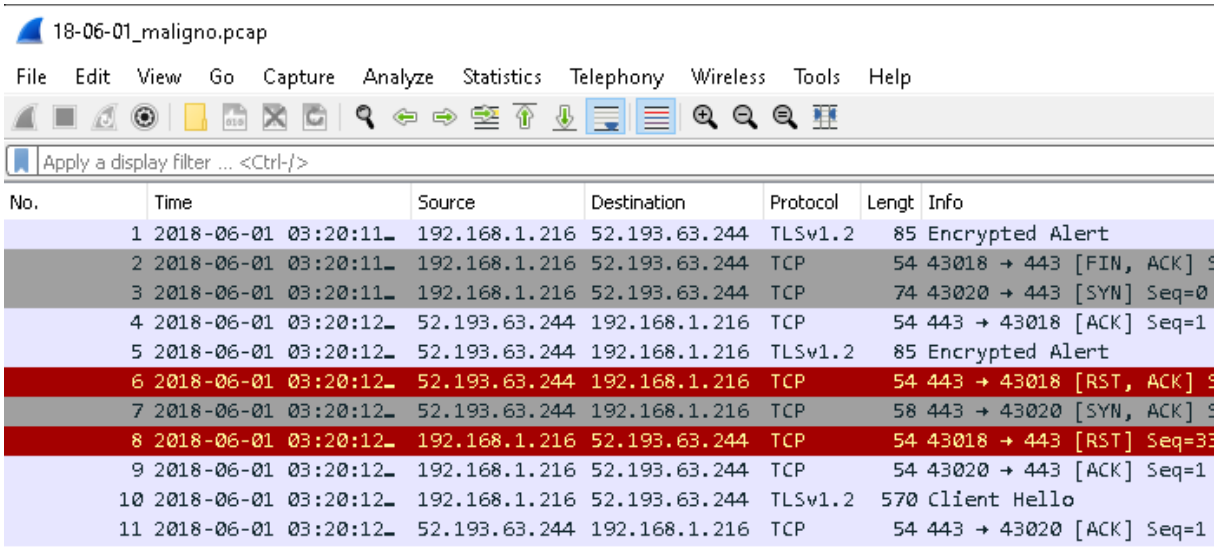
- um host sendo escaneado (escaneamento TCP/SYN/UDP/ACK/ICMP);
- um host sofrendo um ataque DoS devido aos níveis de inundação SYN/ICMP ou até mesmo da aplicação;
- tráfego de rede mediante portas não usuais, dentre outros.

Na Figura 8, foi carregado o arquivo 18-06-01.pcap (HAMZA *et al.*, 2019) que contem o tráfego integral de uma rede com dispositivos IoT, contendo diversos ataques de negação de serviço, este exemplo foca no ataque DoS do tipo TCP SYN. No quadro superior os pacotes são visualizados em lista, contendo o número (*No.*), tempo (*Time*), origem (*Source*), destino (*Destination*), protocolo (*Protocol*), tamanho (*Lenght*) e informações (*Info*).

- *No.*: representa a posição numérica do pacote capturado. O colchete indica que este pacote faz parte de uma conversa.
- Time: mostra quanto tempo levou do início da captura até a captura do pacote. É possível mudar o formato de exibição desses números, caso você queira algo diferente.
- *Source*: é o endereço do sistema que enviou o pacote.
- *Destination*: é o endereço de destino do pacote.
- *Protocol*: indica o tipo de pacote, por exemplo, TCP, DNS, DHCPv6 ou ARP.
- *Length*: exibe o tamanho do pacote em bytes.
- *Info*: apresenta informações sobre o conteúdo do pacote.

Na Figura 9 visualizamos o quadro do meio e o quadro inferior, onde as informações são agregadas em camadas de rede. Ao expandir cada camada, são visualizados

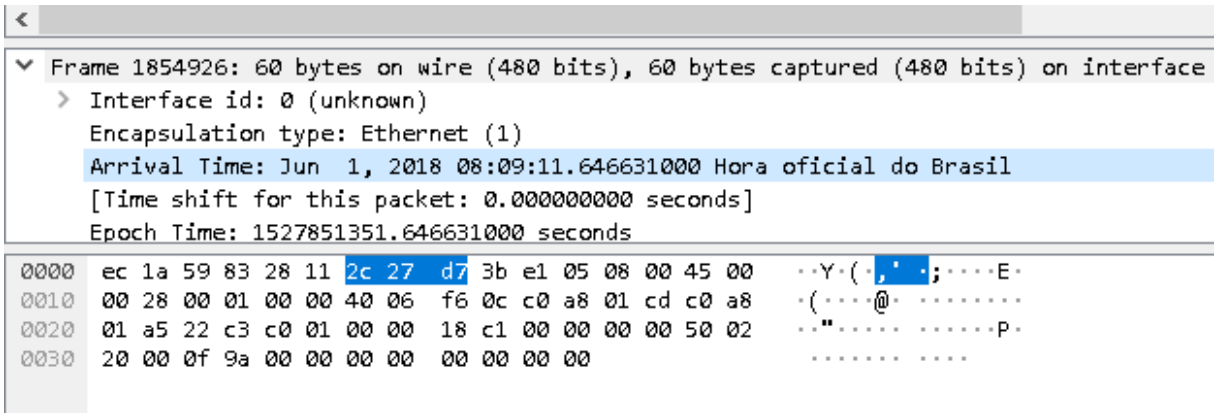
Figura 8 – Tráfego de rede visualizado com WireShark



Fonte: Elaborado pelo autor.

os detalhes de cada campo. Neste mesmo exemplo, ainda na Figura 9, a hora de transmissão do pacote é colocada em evidência. No quadro inferior é visualizado os bytes que compõe cada parte do pacote de rede.

Figura 9 – Quadros inferiores do WireShark



Fonte: Elaborado pelo autor.

Filtros e Recursos Avançados

O *WireShark* captura o conteúdo de cada pacote transmitido, devido à diversidade de padrões e protocolos de rede, é gerado uma grande quantidade de informação, o que dificulta análises específicas à primeira vista. Contudo, o *WireShark* possui uma funcionalidade de filtros que permite separar, destacar ou esconder tráfego usando

diversos critérios, como, por exemplo, endereço IP, destino, origem, protocolo, tipo de pacote, conteúdo de requisições HTTP, dentre outros disponíveis na documentação do *WireShark*[1]. Seguindo o exemplo do ataque TCP SYN, é primeiramente aplicado o filtro *tcp*, para exibir somente pacotes do protocolo TCP. Em seguida é aplicado o filtro *tcp.flags.syn == 1 and tcp.flags.ack == 0*, neste caso são obtidos somente os pacotes que requisitam o estabelecimento de uma conexão, ainda sem o reconhecimento (ACK) da outra parte. Por fim, é gerado um gráfico de entrada e saída, através do menu *Statistics->I/O Graphs*, visualizado na Figura 10.

Neste gráfico é possível visualizar os 2 filtros aplicados, em diferentes cores. Primeiramente, em preto, é apresentado o número total de pacotes por minuto, sem filtros. Esta linha possui um padrão mais aleatório, que dificulta o reconhecimento de variações. Na linha amarela, é aplicado o filtro *tcp*, é um observado um número menor de pacotes, mas ainda com um nível de aleatoriedade. Como vários protocolos de rede rodam sobre o protocolo de TCP, por este atuar na camada de transporte (TANENBAUM; WETHERALL, 2011), essa primeira segmentação (filtro *tcp*) não destaca o momento dos ataques. Na linha verde é aplicado o filtro *tcp.flags.syn == 1 and tcp.flags.ack == 0*, neste caso o momento dos ataques de inundação TCP SYN, de 1, 10 e 100 pacotes por segundo fica mais evidente. Os números ainda podem ser exportados em formato csv, no botão *Save As...* para aplicação de modelos de reconhecimento de anomalias.

[1] https://wiki.wireshark.org/DisplayFilters

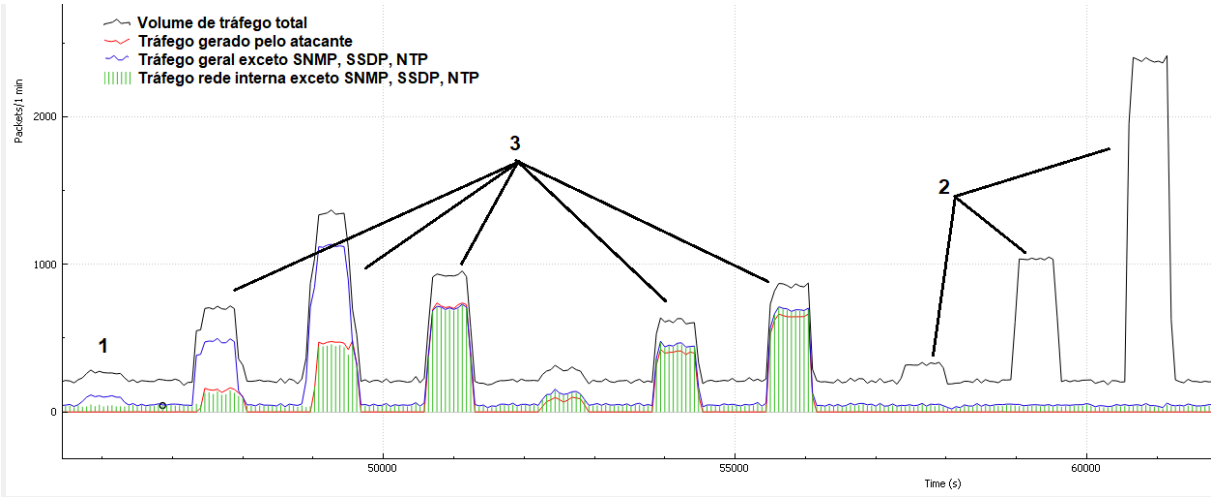
Figura 10 – Filtros e Statistics->I/O Graphs do WireShark - TCP

Fonte: Elaborado pelo autor.

A mesma situação ocorre para o protocolo UDP, que também atua na camada de transporte, portanto é comum abrigar outros protocolos, como, por exemplo, SNMP, SSDP, DNS, dentre outros. Na Figura 11 é visualizado o tráfego UDP na linha preta, obtido com o filtro *udp*. Na linha vermelha, foi isolado o tráfego do atacante, com o filtro textitip.src==192.168.1.205. Neste ponto percebe-se que há outras barras, variações volumosas, além do tráfego do ataque. Para analisar o que há nesses pontos, basta clicar na parte da linha desejada que o software marca o pacote, podendo ser visualizado com riqueza de detalhes como nas Figuras 8 e 9. Dessa forma, foram analisados os pacotes em cada variação da Figura 11, que não fazia parte do tráfego gerado pelo atacante. O tráfego indicado pelo número 1, era referente a tráfego DNS com a rede externa, as barras indicadas pelo número 2 eram de ataque SNMP. Os volumes adicionais no momento do ataque, indicados pelo número 3 eram referentes a pacotes SSDP. Então foi aplicado o filtro *udp && ip.src==192.168.0.0/16 && (not snmp) && (not icmp) && (not ssdp)* para captar somente tráfego UDP da rede interna, exceto SNMP

e SSDP.

Figura 11 – Filtros e Statistics->I/O Graphs do WireShark - UDP



Fonte: Elaborado pelo autor.

Através deste filtro, destacado pelas barras verdes da Figura 11, chega-se a uma aproximação do tráfego do atacante, representado pela linha vermelha. Mesmo nos protocolos TCP e UDP, que abrigam outros protocolos, é possível aplicar filtros para reconhecimento de ataques. Estes exemplos demonstram algumas possibilidades de uso, comprovando o potencial deste software para análises sobre o tráfego de rede.

3 MAPEAMENTO SISTEMÁTICO DA LITERATURA

Para analisar as propostas baseadas na análise de anomalias para identificação de ataques de negação de serviço em redes IoT, foi realizado um Mapeamento Sistemático da Literatura (MSL). O protocolo de pesquisa foi elaborado com acompanhamento de especialista na área de estudos sistemáticos da literatura (BENITTI, 2012) e analisado e aprovado pelo orientador desta pesquisa. Mediante uma busca realizada em 4 bases de pesquisa, foram analisados 903 artigos, após aplicação de critérios de inclusão e exclusão, 33 artigos foram incluídos para extração dos dados relativos a 8 questões de pesquisa. Considerando os resultados, observou-se que há espaço para otimização dos modelos de reconhecimento, com uso de algoritmos e dados de *dataset* atualizados, além de experimentação e prototipagem em ambientes realistas.

Inicialmente foram selecionadas 2 revisões sistemáticas semelhantes ao tema escolhido. A mais recente de (NASCIMENTO; ARAUJO, J.; RIBEIRO, 2022), possui o título altamente alinhado com o tema, porém na seleção de palavras-chave, limita aos protocolos COAP e MQTT. Em relação a esta estratégia e objetivo, a revisão proposta pretende ser mais ampla, visando soluções multiprotocolos. Em outra revisão, executada há 2 anos, (AL-HADHRAMI; HUSSAIN, 2021) utiliza alguns termos de ataques específicos, “Blackhole Attacks”, “Jamming IoT”, “6lowpan Attack”, “Flooding Attack”, o que também restringe e exclui alguns trabalhos importantes sobre ataques de negação de serviço. Em relação a estas duas revisões, o mapeamento proposto buscou estudos primários que tratam o problema de reconhecimento de ataques através da análise de anomalias. Uma vez que os ataques de negação de serviços, em algumas vezes referenciados também como volumétricos, aumentam radicalmente a quantidade de pacotes em relação ao tempo, gerando anomalias na rede (HAMZA *et al.*, 2019).

Um sistema de reconhecimento de anomalias no contexto de ataques cibernéticos é geralmente avaliado por 2 métricas (TAVALLAEE, 2011): (i) Eficiência; Quantos recursos envolvidos para atingir o objetivo. (ii) Precisão ou eficácia; Capacidade de distinguir tráfego benigno de maligno. Diante disso, foi realizada uma comparação entre os grupos de soluções encontradas, analisando resultados como: precisão no reconhecimento de ataques, tempo de resposta, proporção de redução de dados e complexidade computacional envolvida no processo.

3.1 QUESTÕES DE PESQUISA

As questões de pesquisa auxiliam na definição do escopo do trabalho, portanto devem ser claras e objetivas (SAMPAIO; MANCINI, 2007). Diante disso, foram elaboradas as seguintes perguntas:

Q1. Quais são as propostas para identificar ataques de negação de serviço em redes IoT baseadas na análise de anomalias?

Q2. Qual a precisão alcançada pelas soluções propostas?

Q3. Qual o tempo de resposta para o ataque ser identificado?

Q4. Qual é a complexidade computacional da solução proposta?

Q5. Os dados usados para identificação do ataque são reduzidos? Qual a proporção?

Q6. Os testes são realizados em qual ambiente ou dataset?

Q7. Quais os desafios para implementação em ambiente real?

Q8. Qual o ano de publicação dos artigos selecionados?

3.2 PROCESSO DE SELEÇÃO

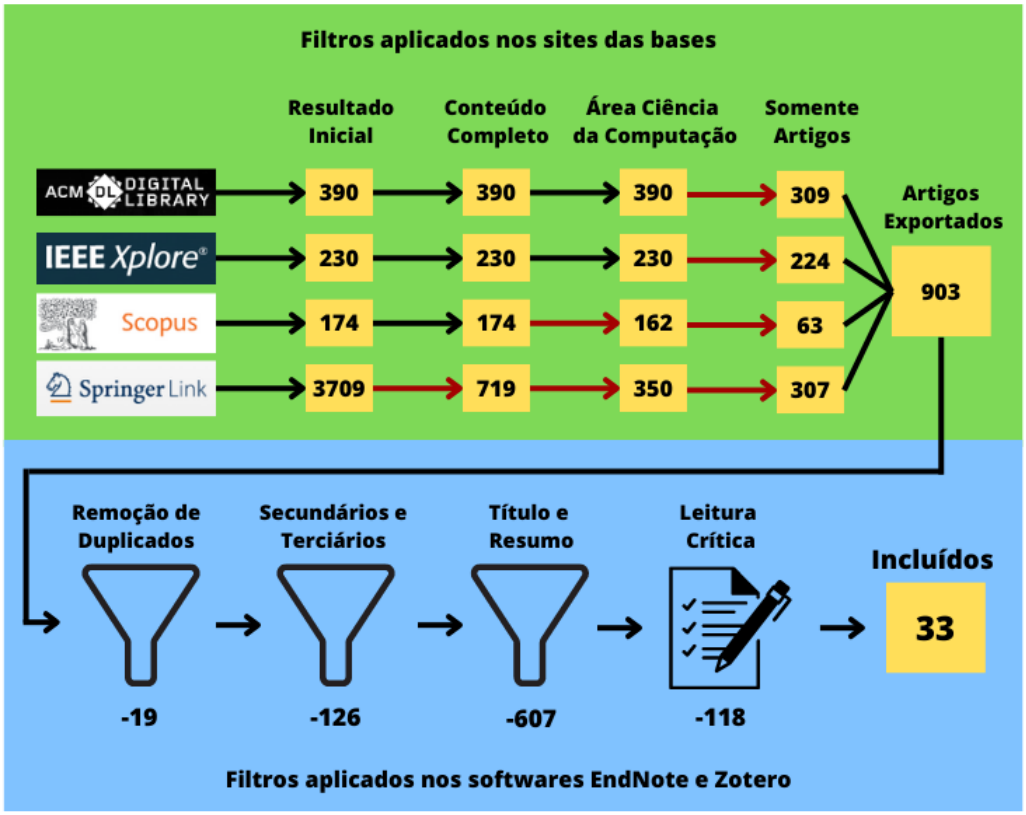
Para obter os trabalhos relacionados foi executada a seguinte *string* de busca: (“DOS” OR “Denial of service” OR “volumetric”) AND “attacks” AND (“IOT” OR “Internet of Things”) AND “Anomaly”. A busca foi realizada no mês de Outubro de 2022 nas bases ACM, IEEEXplore, Scopus e Springer utilizando as configurações detalhadas no Apêndice A e os critérios descritos no Apêndice B.

Os artigos exportados pelas bases foram centralizados na ferramenta EndNote. O Zotero foi usado na fase de busca de PDFs devido ao maior sucesso nesta etapa. O processo de seleção pode ser visualizado com maiores detalhes pela Figura 12.

3.3 TRABALHOS SELECIONADOS

Os trabalhos selecionados e seus respectivos referenciais podem ser visualizados na Tabela 10 - Apêndice B. Para permitir uma melhor apresentação das referências foi criado um índice, dessa forma, em caso de gráficos e imagens, a referência será apresentada pelo índice da Tabela 10 - Apêndice B.

Figura 12 – Processo de seleção de artigos



Fonte: Elaborado pelo autor.

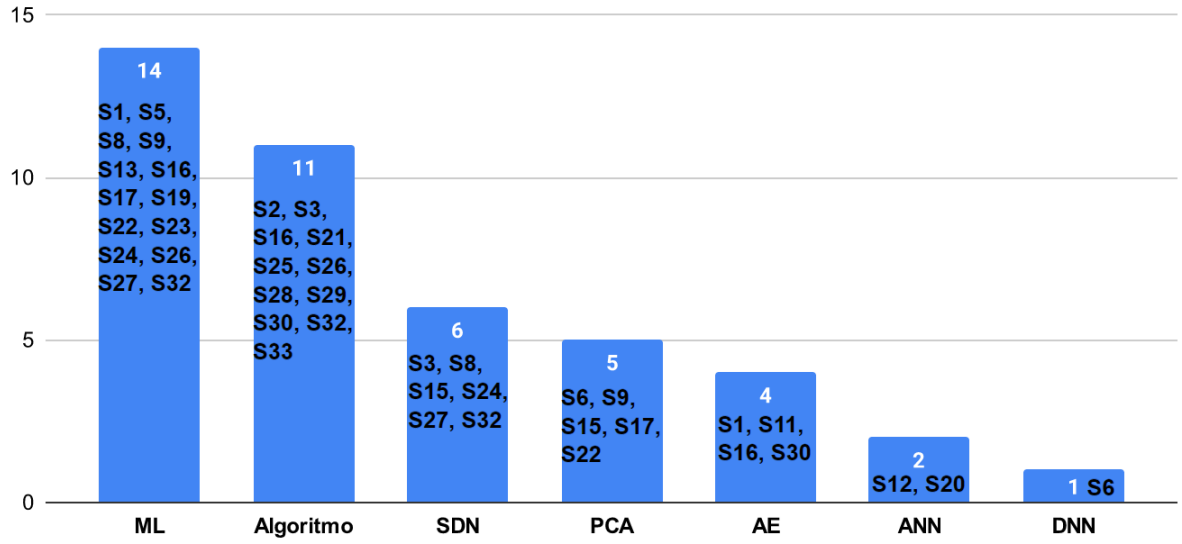
3.4 RESULTADOS

A extração dos resultados foi dividida em cada questão de pesquisa, sendo apresentada a seguir.

Q1. Quais são as propostas para identificar ataques de negação de serviço em redes IoT baseadas na análise de anomalias?

As propostas dos trabalhos selecionados foram categorizadas em *Machine Learning* (ML), algoritmo, *Software Defined Network* (SDN), *Principal Component Analysis* (PCA), *Auto Encoder* (AE), *Artificial Neural Network* (ANN) e *Deep Neural Network* (DNN). Na Figura 13 podemos observar uma predominância no uso de ML, com 14 artigos, seguido de algoritmo com 11, SDN 6, PCA 5, AE 6, e ANN e DNN com 2 e 1, respectivamente.

Figura 13 – Métodos utilizados na identificação de anomalias



Fonte: Elaborado pelo autor.

Qual a precisão alcançada pelas soluções propostas? (Q2)

Sobre a precisão alcançada, observou-se que 26 dos artigos apresentam dados sobre a precisão, sendo que 15 apresentam precisão maior ou igual a 95%. Na Figura 14, é possível visualizar os artigos selecionados de forma ordenada, conforme a precisão, sendo a mais alta no topo e a mais baixa na parte inferior. Ao lado da taxa de precisão é apresentada a sigla dos métodos e tecnologias utilizadas, que pode ser visualizada de forma estendida através da lista de abreviaturas e siglas.

Qual o tempo de resposta para o ataque ser identificado? (Q3)

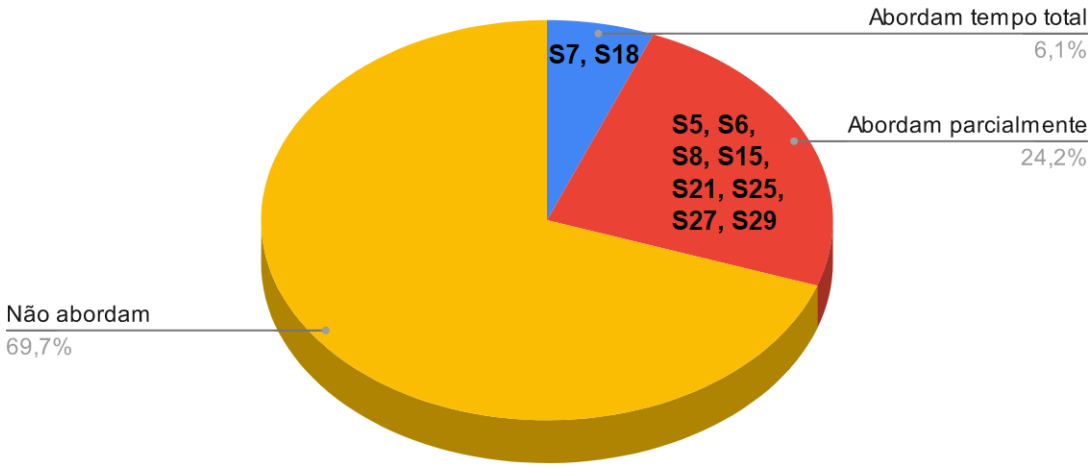
Neste sentido, 69,7% dos artigos não abordam o tempo de resposta. Apenas 2 abordam o tempo total de resposta, e 8 abordam parcialmente o tempo de resposta. A abordagem parcial é considerada nos casos onde é apresentado somente o tempo de resposta, sem as etapas de pré-processamento como o tempo de treinamento dos algoritmos de ML. Na Figura 15 é possível visualizar a proporção dos trabalhos que abordam tempo de resposta, já na Figura 16 é apresentado um comparativo, onde os trabalhos S7, S18, S25, S29 apresentam tempo de resposta menor que um segundo.

Figura 14 – Trabalhos ordenados por precisão

Precisão atingida			Tecnologias e Métodos
>=98%	S30	100	VAE, cyclic queuing algorithm
	S6	99,9	PCA,GMO,DNN
	S14	99,9	Kullback-Leibler (KLD)
	S22	99,9	ML (Decision Tree)
	S19	99	ML, GBM
	S32	98,88	SDN, ML, RF-GMM algorithm
	S24	98	SDN, ML
>=95%	S8	97,3	SDN,ML,SVM
	S3	97,26	SDN,ARIMA algorithm
	S4	97,03	CCTAV,Mahalanobis
	S5	97	Correlation,Gain Ratio,ML
	S27	96,05	SDN, ML
	S11	95,86	AE
	S20	95	ANN, Lyapunov exponent
	S21	95	algorithm
<95%	S12	94,96	ANN, SVM
	S15	94,9	SDN, PCA, Clustering (X-means)
	S17	94,8	PCA, ML, (Random Tree)
	S18	94	EWMA
	S13	92	ML, SVM
	S16	91,7	ML, AE, Bayes algorithm
	S1	90,99	DAE,ML
	S23	90	ML (one-class classification)
	S26	90	ML, LSTM, ARIMA algorithm
	S28	90	AOT algorithm
	S2	86,7	KOAD algorithm,Mahalanobis

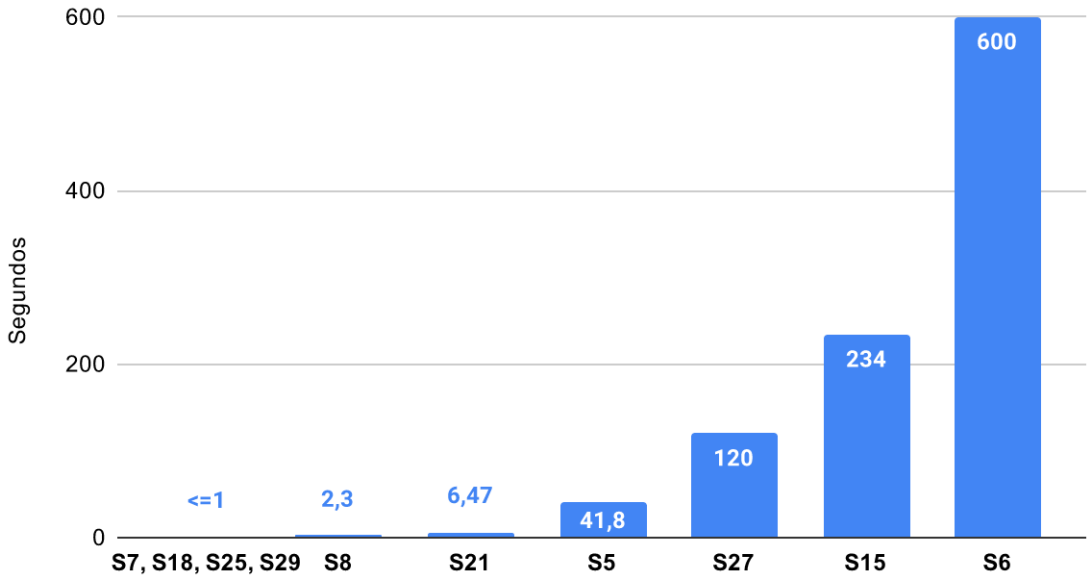
Fonte: Elaborado pelo autor.

Figura 15 – Trabalhos que abordam tempo de resposta



Fonte: Elaborado pelo autor.

Figura 16 – Tempo de resposta das soluções



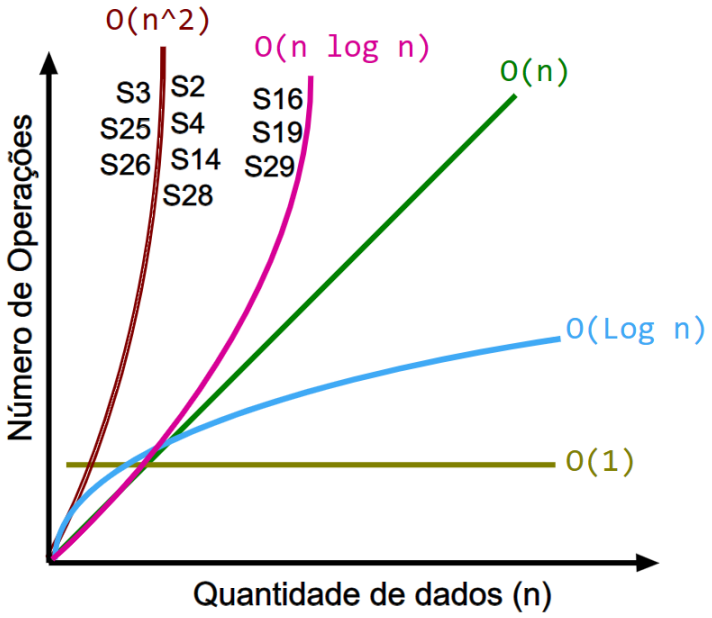
Fonte: Elaborado pelo autor.

Qual é a complexidade computacional da solução proposta? (Q4)

Como na borda da rede temos recursos computacionais limitados, a complexidade computacional é uma métrica importante de ser observada. Ela informa quanto recurso necessitamos para atingir o objetivo. Além do aspecto da viabilidade de implementação de modelos em redes IoT, o aperfeiçoamento dos modelos e algoritmos trazem ganhos operacionais e de eficiência energética.

Neste contexto, 10 artigos apresentaram análise e declaração da complexidade computacional. Na Figura 17, os artigos foram posicionados conforme a complexidade computacional. A complexidade não traz um número exato, mas a noção de quantidade de operações necessárias conforme a quantidade de dados. Dessa forma, os artigos S16, S19 e S29 são mais eficientes em relação aos demais, que apresentam complexidade quadrática.

Figura 17 – Complexidade computacional das soluções propostas

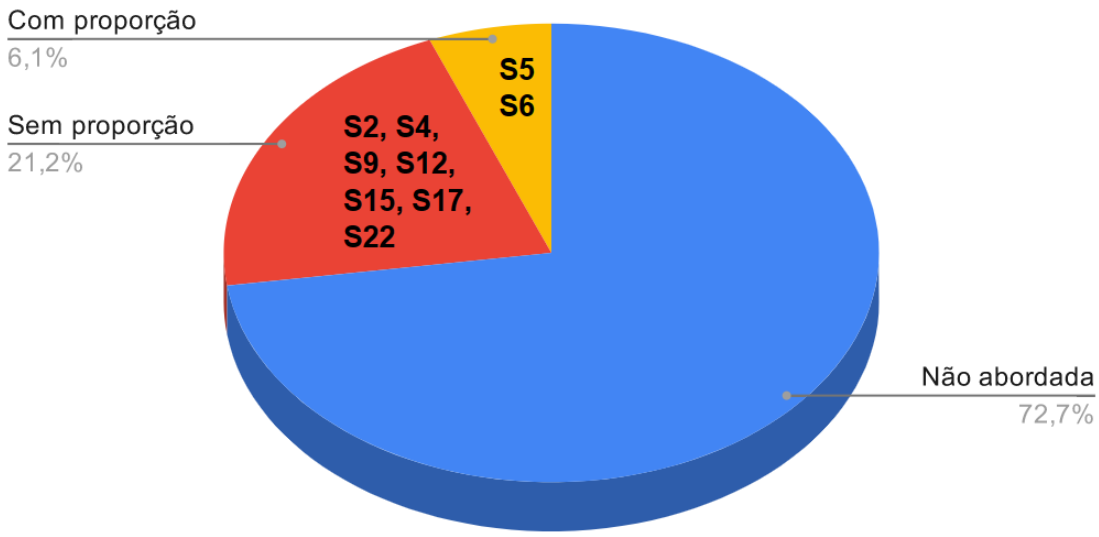


Fonte: Elaborado pelo autor.

Os dados usados para identificação do ataque são reduzidos? Qual a proporção? (Q5)

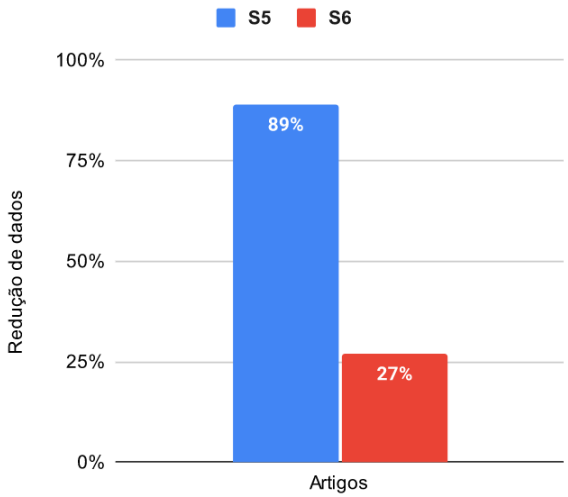
Neste sentido, 72,7% dos artigos incluídos não abordaram a questão da redução de dados. 21,2% citaram a redução de dados, mas sem quantificar a proporção. Apenas os artigos S5 e S6 apresentaram a proporção da redução de dados, atingindo taxas de 89% e 27% respectivamente. A abordagem geral pode ser visualizada na Figura 18, já a proporção de redução de dados é apresentada na Figura 19.

Figura 18 – Abordagem geral sobre a redução de dados



Fonte: Elaborado pelo autor.

Figura 19 – Proporção na redução de dados



Fonte: Elaborado pelo autor.

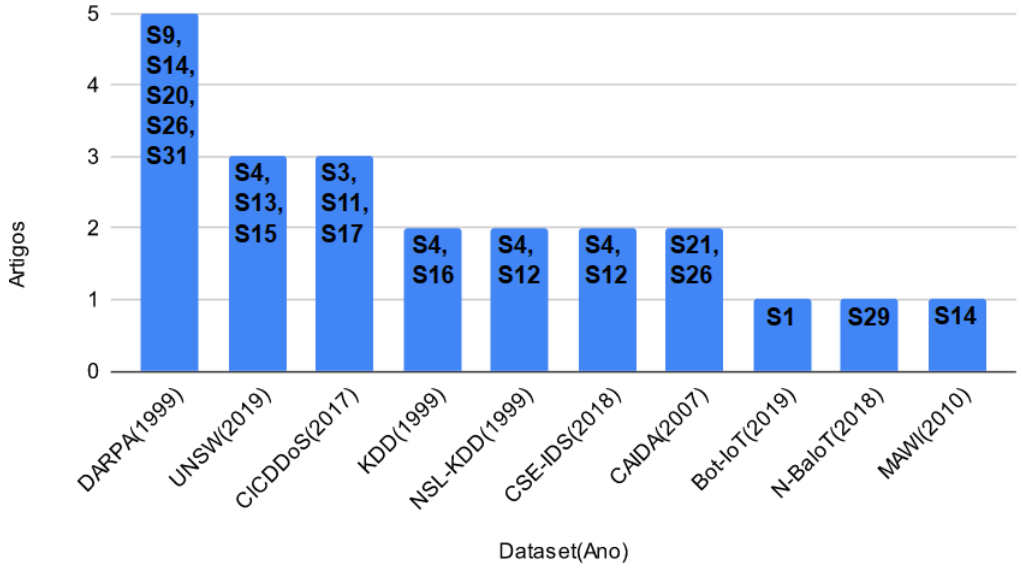
Os testes são realizados em qual ambiente ou dataset? (Q6)

O *dataset* mais usado nos artigos selecionados foi o DARP (1999), utilizado em 5 artigos, seguido por UNSW (2019) que foi utilizado em 3 artigos, e CICDDoS (2017) também 3. Na Figura 20 é possível visualizar um panorama dos *datasets* e os artigos em que eles foram utilizados.

Sobre o ano de geração dos dados, na Figura 21 observa-se que 40,9% são de 1999, seguidos por 18,2% com dados de 2019, 13,6% nos anos de 2017 e 2018, 9,1%

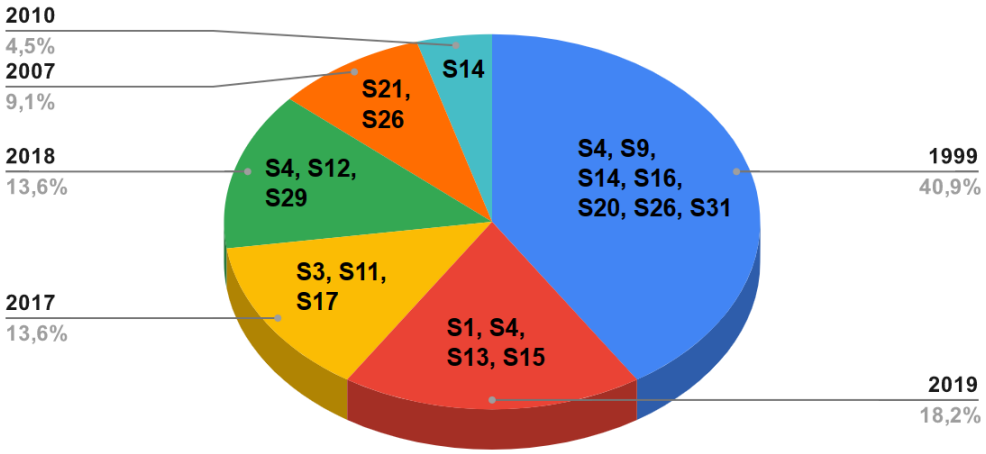
no ano de 2007 e 4,5% no ano de 2010.

Figura 20 – Datasets utilizados pelos soluções



Fonte: Elaborado pelo autor.

Figura 21 – Ano de geração (idade) dos Datasets



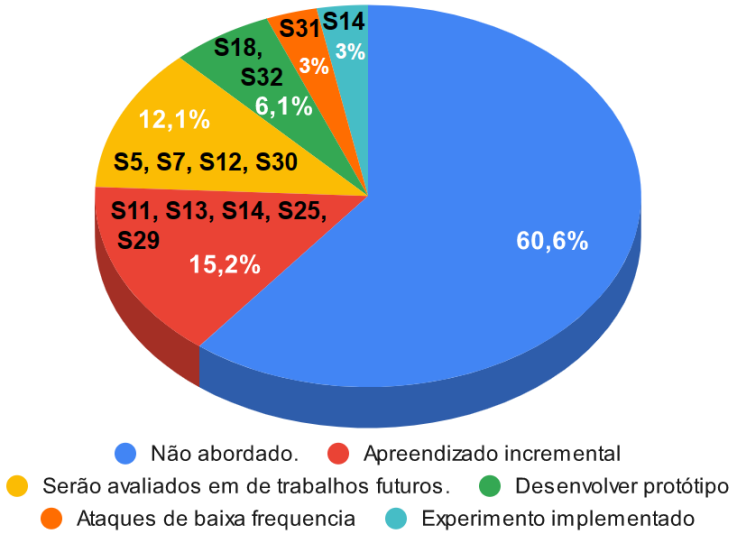
Fonte: Elaborado pelo autor.

Quais os desafios para implementação em ambiente real? (Q7)

Na Figura 22 nota-se que o desafio de maior recorrência para implementação em ambiente real é o de aprendizado incremental, citado em S11, S13, S14 e S25. Seguido por desenvolvimento de protótipos, S18 e S32, e ataques de baixa frequência S31. Os artigos S5, S7, S12 e S30, indicaram a implementação como trabalhos futuros.

Apenas S14 realizou a implementação do experimento e 60,6% não apresentaram discussão sobre os desafios de implementação em ambiente real.

Figura 22 – Desafios para implementação em ambiente real



Fonte: Elaborado pelo autor.

Qual o ano de publicação dos artigos? (Q8)

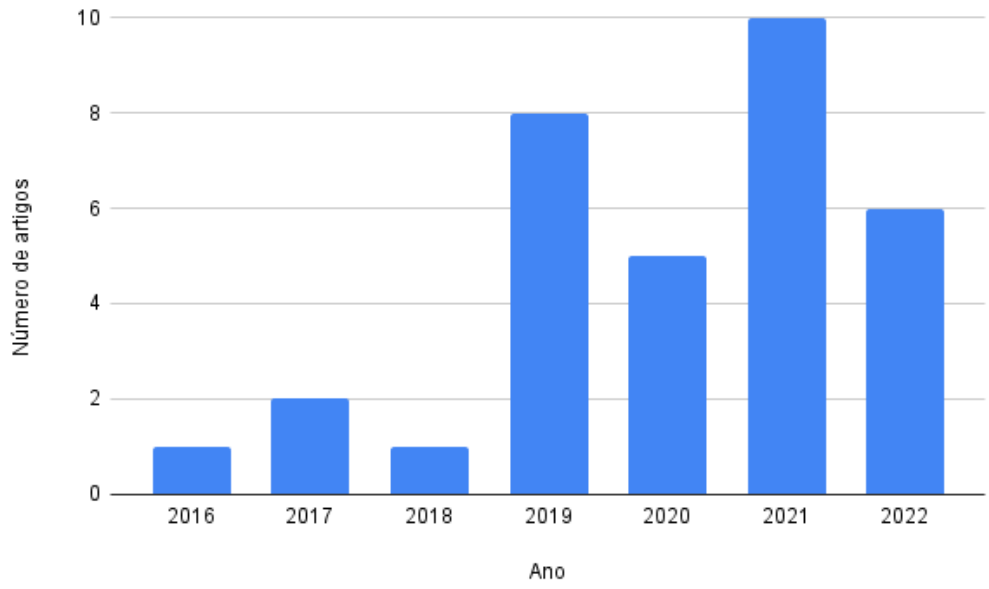
Observa-se que o reconhecimento de ataques de negação de serviço em redes IoT é um tema emergente, com maior número de publicações nos últimos 4 anos, conforme demonstrado pela Figura 23. É preciso pontuar que o ano de 2022 foi limitado a outubro de 2022. Portanto, uma nova execução deste mapeamento sistemático poderá conter um número ainda mais elevado no ano de 2022.

3.5 DISCUSSÃO SOBRE TRABALHOS RELACIONADOS

A Q1 revela uma predominância no uso de ML para identificar e classificar os ataques de negação de serviço através da leitura e processamento das anomalias na rede. Em seguida vemos taxas de precisão com ML de aproximadamente 99% em S19, S22, S24, S32 na Q2. Unindo esses 2 resultados, pode-se constatar o alto potencial no uso ML para identificação de ataques DoS. Contudo, mais da metade destas soluções realiza o processamento do *dataset* de forma offline S1, S8, S9, S13, S17, S19, S27.

O processamento offline do *dataset* inviabiliza a implementação em ambientes reais. Não podemos ler um *dataset* inteiro para então diferenciar o que é um ataque do que é tráfego benigno. Isto seria como aceitar uma série de ataques até que o sistema de defesa entrasse em operação, o que poderia ocasionar a indisponibilidade da rede.

Figura 23 – Número de artigos publicados ao longo dos anos



Fonte: Elaborado pelo autor.

Dessa forma as pesquisas S5, S16, S23, S24, S26, S32 estão mais avançadas em termos de aplicabilidade.

O segundo método de maior recorrência nos resultados da Q1, é o uso de algoritmos, algumas vezes combinados com ML S16, S26, S32, atingindo uma precisão de até 98%. Dentro das soluções que utilizam algoritmos, observam-se 4 referências à média móvel, duas S3 e S26 trazem um modelo auto-regressivo, *Autoregressive Integrated Moving Average* (ARIMA), S18 utiliza a média móvel ponderada exponencial (EWMA) e S22 aplica a suavização dos valores com média móvel para melhorar a classificação dos modelos. Os experimentos com algoritmos em S28 revelam que modelos relativamente simples de ML são mais adequados para o reconhecimento de anomalias na borda da rede devido aos custos computacionais de treinamento destes modelos.

Os resultados da Q2 demonstram alta precisão de reconhecimento na grande maioria dos trabalhos, dessa forma novas pesquisas nessa área devem e precisam ir além desta métrica. Na Q3, que trata sobre o tempo de resposta da solução, 70% dos trabalhos não abordam este fator, crucial para um sistema de reconhecimento de anomalias relacionadas a ataques de negação de serviço. Atrasos no reconhecimento de anomalias podem causar indisponibilidade do sistema. Dos 30% que abordam o tempo de serviço, apenas S7 e S18 estão próximos de um reconhecimento em tempo real. S25 e S29 apesar de levarem menos de 1 segundo na etapa de reconhecimento, não consideram o tempo de treinamento do sistema. Nos artigos de ML que trazem dados

mais completos sobre o tempo de resposta, observa-se 234 segundos (treinamento + reconhecimento) em S15 e 600 segundos somente para treinamento em S6.

Além do tempo de resposta, a complexidade computacional da solução, abordada na Q4 também revela-se um campo aberto de pesquisa. A maioria das soluções apresenta complexidade quadrática ou superior S2, S3, S4, S14, S25, S26, S28. A redução da complexidade computacional é um avanço em qualquer camada, especialmente na borda da rede, onde temos recursos computacionais limitados (COOK; MISIRLI; FAN, 2019). Nesse sentido, S3 e S25 segmentam o processo em 2 etapas, uma para reconhecer a anomalia na rede, com menor custo computacional, e outra para mitigar com ataque, com maior custo, porém aplicada somente no recorte temporal da anomalia. Esta estratégia demonstra-se eficiente por trabalhar com um menor número de variáveis e comparações na etapa de reconhecimento da anomalia.

Na Q5, que trata da redução de dados, observa-se que 73% dos trabalhos não abordam esta questão. S2, S4, S9, S12, S15, S17, S22 abordam de forma genérica, indicando que o número de dados ou recursos foi reduzido. Apenas 2 apresentam a proporção que os dados foram reduzidos, S5 em 89% e S6 em 27%. Uma técnica relacionada diretamente com este tema, que aparece nos resultados da Q1 é a análise dos principais componentes, *Principal Component Analysis* (PCA). O PCA recebe diversos dados de um *dataset* e indica quais são os de maior relevância, o que diminui o tempo de processamento e treinamento dos algoritmos de ML. Além do PCA, também observou-se o uso de correlação em S5 e S12, e (*Grey Wolf Optimizer*) GWO em S6. Estes modelos pré-definidos de redução e seleção de dados podem ser otimizados atuando de forma mais agressiva quando segmentado as etapas do processo de reconhecimento em identificação de anomalia e mitigação do ataque como em S3 e S25. Combinar técnicas de redução de dados para auxiliar a identificar qual seria a característica mais comum dos ataques de negação serviço eleva o potencial da redução. Ao trabalhar com 8 variáveis, S12 demonstra uma precisão de 95% e após reduzir para apenas uma variável, ainda mantém uma precisão de 92%, provando o potencial da redução de dados.

A validação de modelos de redução de dados e reconhecimento de anomalias é frequentemente realizada com uso de datasets. Na Q6 verificou-se que o dataset mais usado é de 1999, porém temos datasets mais atualizados que refletem com mais precisão as características das redes IoT. Ao comparar com os dados da Q8, que trata do ano de publicação dos artigos, observa-se que a maioria dos trabalhos foi publicada nos últimos 4 anos, 2019, 2020, 2021 e 2022, em contraponto a maioria dos artigos usa dados de 1999.

Por fim, trazemos a discussão da Q7, sobre os desafios de implementação das propostas em ambientes reais. O principal desafio mapeado foi o de aprendizado incremental, seguido por desenvolvimento de protótipo ou parte da solução, e ataques de

Tabela 1 – Trabalhos Relacionados

Autor e Ano	Rec. Online	Complexidade	Param. Dinâmica	Precisão	Multi Protocolo
(HE *et al.*, 2020)	Sim	$O(i * 3n)$	Sim	91,7%	Sim
(DANESHGADEH *et al.*, 2018)	Não	$O(n^2)$	Não	86,7%	Sim
(BHUVANESWARI AMMA; SELVAKUMAR, 2021)	Não	$O(n^3)$	Não	97,03%	Sim
(BOUYEDDOU *et al.*, 2021)	Não	$O(n^2)$	Não	99,9%	Não
(LI, F. *et al.*, 2019)	Não	$O(n^2) + O(n)$	Sim	94,7%	Sim
(JIA *et al.*, 2022)	Não	$O(n^2)$	Sim	97,26%	Sim
(MERGENDAHL; LI, J., 2020)	Sim	$O(n^2)$	Não	97%	Sim
(ANJUM *et al.*, 2022)	Não	$O(th) + O(n)$	Sim	99%	Sim
Proposta (LAUTERT, H. F.; MACEDO, D. de; PIOLI, 2023a)	Sim	$O(n)$	Sim	97,67%	Sim

Fonte: Elaborado pelo autor.

baixa frequência. Ao ligar com os resultados da Q1, onde se observa a predominância no uso de ML, fica nítido o desafio de como fazer o modelo aprender de forma *online*, sem usar um *dataset* inteiro e tornar o modelo de reconhecimento demasiado específico para o *dataset*, prejudicando a generalização. O desenvolvimento de protótipo ou parte da solução, também demonstra complexidade, uma vez que para analisar um modelo por meio de estatísticas de rede é preciso ler o tráfego de rede, converter em estatísticas, e no caso de ML ainda treinar o modelo para finalmente entrar em operação. A leitura das estatísticas muitas vezes ocorre via *switches* SDN, mas deve-se considerar o custo e a capilaridade destes na borda rede. Outro desafio são os ataques de baixa frequência que muitas vezes passam despercebidos pelos modelos de reconhecimento de anomalias.

Na Tabela 1 pode-se observar que a maioria dos trabalhos não descreve como realizar o reconhecimento *online* das ameaças. Apesar do alto nível de precisão do atingido pelos trabalhos de ML, em ambientes reais não podemos ler um *dataset* inteiro para então diferenciar o que é um ataque do que é tráfego benigno. Isto seria como aceitar uma série de ataques até que o sistema de defesa entrasse em operação, o que poderia ocasionar a indisponibilidade da rede. Dentro dessas constatações, nesta pesquisa é proposto um sistema de detecção de intrusão de baixa complexidade computacional, com viabilidade de operação *online* e parametrização dinâmica através do histórico de médias móveis.

4 METODOLOGIA

A pesquisa foi realizada com abordagem predominantemente quantitativa, embasada em números, mais precisamente, o número de pacotes segmentados por protocolos em relação ao tempo, para análise estatística de relações de causa e efeito (WAZLAWICK, 2009). Foram analisados os resultados alcançados pelos trabalhos existentes na área de análise de anomalias, buscando dados como precisão, tempo de resposta, complexidade computacional, proporção de redução de dados. De forma complementar será analisado com abordagem qualitativa os desafios para implementação das propostas em ambiente real.

A coleta de dados foi inicialmente realizada mediante um mapeamento sistemático da literatura, com objetivo de conhecer as propostas existentes e responder às perguntas de pesquisa. O procedimento foi orientado e revisado por especialista na área (BENITTI, 2012) e revisado e aprovado pelo orientador da pesquisa (MACEDO, J. de; DYLLON; FAGUNDES, 2023).

Por fim, a análise dos dados foi realizada com auxílio da ferramenta *WireShark* (COMBS, 2023), para contabilização de estatísticas, linguagem *python* para elaboração e validação do modelo, e *jupyter* notebook para apresentação dos códigos fontes e gráficos produzidos. A eficiência do modelo de reconhecimento de anomalias proposto foi obtida através da aplicação da proposta nos *datasets* de (HAMZA *et al.*, 2019), (KANG *et al.*, 2019) e (VACCARI *et al.*, 2020). Já os procedimentos metodológicos serão descritos na seção seguinte.

4.1 PROCEDIMENTOS METODOLÓGICOS

Para facilitar a reprodutibilidade dos experimentos desta pesquisa, visto que cada *dataset* é gerado com caraterísticas distintas, os procedimentos metodológicos foram separados por *datasets* nas seções seguintes. Dentro de cada seção foram descritas as etapas e referências para verificação do sucesso ou falha no reconhecimento de cada ataque. Além da aplicação do modelo para reconhecimento do ataque, também foram usados arquivos de tráfego benigno, para verificação de falsos positivos.

DATASET UWSN

Devido à recorrência do *dataset UWSN IoT traces* (HAMZA *et al.*, 2019) nos trabalhos relacionados durante a etapa de mapeamento sistemático da literatura, bem como o ano de geração dos dados, este mesmo foi o escolhido para iniciar a aplicação dos experimentos. Inicialmente foi acessado o *site*[1] onde o *dataset* está hospedado. Após verificação no arquivo attackinfo.xlsx[2], o qual possui informações detalhadas

[1] https://iotanalytics.unsw.edu.au/attack-data.html
[2] https://iotanalytics.unsw.edu.au/anomaly-data/attackinfo.xlsx

dos ataques, como arquivo, tempo, origem e destino, foi constatado que apenas os arquivos 18-10-24.pcap, 18-10-23.pcap, 18-10-22.pcap, 18-06-03.pcap, 18-06-02.pcap, 18-06-01.pcap possuem as informações descritas sobre o ataque. Portanto, para possibilitar a verificação da variação no número de pacotes gerada no momento do ataque, conforme a marca temporal (*timestamp*) de cada ataque, somente estes arquivos foram incluídos. Ao carregar o arquivo 18-10-24.pcap, o *WireShark* retornou erro de integridade, indicando interrupção no meio de um pacote, portando este arquivo foi removido dos testes. A seguir são descritos os passos realizados para aplicação do modelo.

Ataques TCP SYN

1. Verificação no arquivo attackinfo.xlsx[2] sobre qual arquivo pcap foi realizado o ataque.
2. Carregamento do arquivo 18-06-01.pcap[3] no software *WireShark*.
3. Aplicação do filtro *tcp.flags.syn == 1 and tcp.flags.ack == 0*.
4. Extração de estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 1 minuto, 10 segundos, e 1 segundo, 500 milissegundos e 100 milissegundos. No formato *csv*, no botão *Save As*.
5. Verificação de ataques conforme a marca temporal (*timestamp*) e variação no número de pacotes gerada no momento do ataque.
6. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo AnomaliaTCPSYN.ipynb[4]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente[5], dessa forma é facilitada a reprodução e continuidade dos testes.
7. O procedimento foi repetido, voltando a etapa 2 com o arquivo 18-10-23.pcap, e publicado no arquivo AnomaliaTCPSYN-2.ipynb[6]

Ataques SSDP

1. Verificação no arquivo attackinfo.xlsx[2] sobre qual arquivo pcap foi realizado o ataque.
2. Carregamento do arquivo 18-06-03.pcap[7] no software *WireShark*.
3. Aplicação do filtro *ssdp*.

[3] https://iotanalytics.unsw.edu.au/anomaly-data/pcap/AttackAndBenign/18-06-01.pcap
[4] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaTCPSYN.ipynb
[5] https://github.com/hflautert/AnomalyDetection/tree/main/Stats
[6] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaTCPSYN-2.ipynb
[7] https://iotanalytics.unsw.edu.au/anomaly-data/pcap/AttackAndBenign/18-06-03.pcap

4. Extração de estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 1 minuto, 10 segundos, e 1 segundo, 500 milissegundos e 100 milissegundos. No formato *csv*, no botão *Save As*.
5. Verificação de ataques conforme a marca temporal (*timestamp*) e variação no número de pacotes gerada no momento do ataque.
6. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo AnomaliaSSDP.ipynb[8]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente[9], dessa foram é facilitada a reprodução e continuidade dos testes.
7. O procedimento foi repetido, voltando a etapa 2 com o arquivo 18-10-23.pcap, e publicado no arquivo AnomaliaSSDP-2.ipynb[10]

Ataques Smurf(ICMP)

1. Verificação no arquivo attackinfo.xlsx[11] sobre qual arquivo pcap foi realizado o ataque.
2. Carregamento do arquivo 18-06-02.pcap[12] no software *WireShark*.
3. Aplicação do filtro *icmp*.
4. Extração de estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 1 minuto, 10 segundos, e 1 segundo, 500 milissegundos e 100 milissegundos. No formato *csv*, no botão *Save As*.
5. Verificação de ataques conforme a marca temporal (*timestamp*) e variação no número de pacotes gerada no momento do ataque.
6. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo AnomaliaICMP.ipynb[13]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.
7. O procedimento foi repetido, voltando a etapa 2 com o arquivo 18-10-23.pcap, e publicado no arquivo AnomaliaICMP-2.ipynb[14]

Ataques UDP

1. Verificação no arquivo attackinfo.xlsx[15] sobre qual arquivo pcap foi realizado o ataque.

[8] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaSSDP.ipynb
[9] https://github.com/hflautert/AnomalyDetection/tree/main/Stats
[10] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaSSDP-2.ipynb
[11] https://iotanalytics.unsw.edu.au/anomaly-data/attackinfo.xlsx
[12] https://iotanalytics.unsw.edu.au/anomaly-data/pcap/AttackAndBenign/18-06-02.pcap
[13] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaICMP.ipynb
[14] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaICMP-2.ipynb
[15] https://iotanalytics.unsw.edu.au/anomaly-data/attackinfo.xlsx

2. Carregamento do arquivo 18-06-02.pcap[16] no software *WireShark*.
3. Aplicação do filtro *udp*.
4. Extração de estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 1 minuto, 10 segundos, e 1 segundo, 500 milissegundos e 100 milissegundos. No formato *csv*, no botão *Save As*.
5. Verificação de ataques conforme a marca temporal (*timestamp*) e variação no número de pacotes gerada no momento do ataque.
6. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo AnomaliaUDP.ipynb[17]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.

IoT *network intrusion dataset*

Para complementar os testes foi selecionado o *IoT network intrusion dataset* (KANG *et al.*, 2019), que também é recente, com última atualização em 2019. Em relação aos demais *datasets*, se destaca por disponibilizar informações detalhadas de como isolar os pacotes de ataques por meio de filtros. Dessa forma facilita a aplicação de modelos experimentais, que utilizam como origem conteúdo integral da rede, com arquivos do tipo *pcap*, como na proposta desta pesquisa.

Inicialmente foi visualizado o conteúdo do arquivo *dataset-description.xlsx* e realizado o *download* do arquivo iot-intrusion-dataset.zip, ambos disponíveis no site[18] do respectivo *dataset*. Após o *download* o arquivo *iot-intrusion-dataset.zip* foi descompactado, gerando os arquivos descritos nas próximas subseções.

Como neste *dataset* os ataques foram fatiados em pequenos arquivos, usando o recorte no momento do ataque. Os arquivos de tráfego maligno possuem menos de um minuto de duração, portando os tempos de teste do modelo foram reduzidos para 100ms, 500ms e 1s. A seguir são descritos os passos realizados para aplicação do modelo.

Extração de estatísticas - tráfego normal

1. Carregado o arquivo *benign-dec.pcap* software *WireShark*.
2. Aplicado o filtro *tcp.flags.syn == 1 and tcp.flags.ack == 0*.
3. Extraídas as estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 100 milissegundos, 500 milissegundos, e 1 segundo. No formato *csv*, no botão *Save As*.
4. A etapa anterior foi repetida aplicando os filtros *udp* e *http*.

[16] https://iotanalytics.unsw.edu.au/anomaly-data/pcap/AttackAndBenign/18-06-02.pcap
[17] https://github.com/hflautert/AnomalyDetection/blob/main/AnomaliaUDP.ipynb
[18] https://ieee-dataport.org/open-access/iot-network-intrusion-dataset

Ataques TCPSYN

1. Carregados os arquivos *dos-synflooding-1a6-dec.pcap* software *WireShark*.
2. Aplicado o filtro *tcp.flags.syn == 1 and tcp.flags.ack == 0*.
3. Extraídas as estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 100 milissegundos, 500 milissegundos, e 1 segundo. No formato *csv*, no botão *Save As*.
4. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo IOTIN_TCPSYN.ipynb[19]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.

Ataques UDP

1. Carregados os arquivos *mirai-udpflooding-(1a4)-dec.pcap* software *WireShark*.
2. Aplicado o filtro *udp*.
3. Extraídas as estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 100 milissegundos, 500 milissegundos, e 1 segundo. No formato *csv*, no botão *Save As*.
4. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo IOTIN_UDP.ipynb[20]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.

Ataques HTTP

1. Carregados os arquivos *mirai-httpflooding-1a4-dec* software *WireShark*.
2. Aplicado o filtro *http*.
3. Extraídas as estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 100 milissegundos, 500 milissegundos, e 1 segundo. No formato *csv*, no botão *Save As*.
4. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo IOTIN_HTTP.ipynb[21]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.

[19] https://github.com/hflautert/AnomalyDetection/blob/main/IOTIN_TCPSYN.ipynb
[20] https://github.com/hflautert/AnomalyDetection/blob/main/IOTIN_UDP.ipynb
[21] https://github.com/hflautert/AnomalyDetection/blob/main/IOTIN_HTTP.ipynb

DATASET MQTTset

Para testar o protocolo MQTT foi selecionado o MQTTset. Através dele foi testado o reconhecimento de ataques do tipo DoS MQTT, realizados com a ferramenta *MQTT malaria tool*.

Como o ataque MQTT foi isolado em um pequeno arquivo, usando o recorte no momento do ataque. O arquivo de tráfego maligno possui menos de 3 minutos de duração, portanto inviabiliza a aplicação do modelo na frequência de 1 minuto. Os tempos de teste do modelo realizados nos tempos de 1s, 500ms e 100ms. A seguir são descritos os passos realizados para aplicação do modelo.

1. Carregados os arquivos *capture-1w.pcap* e *capture-malariaDoS.pcap* no software *WireShark*.
2. Aplicado o filtro *mqtt*.
3. Extraídas as estatísticas no menu *Statistics->I/O Graphs*. Exportado nas frequências de 100 milissegundos, 500 milissegundos, e 1 segundo. No formato *csv*, no botão *Save As*.
4. Publicação do modelo de reconhecimento de anomalias na plataforma *GitHub*, por meio do arquivo MQTTset-Malaria.ipynb[22]. Os arquivos *csv*, que alimentam o modelo, foram disponibilizados publicamente no mesmo repositório.

Verificação de Destinos e Portas

Após a verificação da anomalia, para obter os destinos nos arquivos contendo os ataques, foi utilizada a função *Statistics->IPV4 Statistics->Destinations and Ports* no software *WireShark*. Estas informações também podem ser obtidas utilizado o *pyshark*, que traz os dissecadores do *WireShark* para a linguagem *python*.

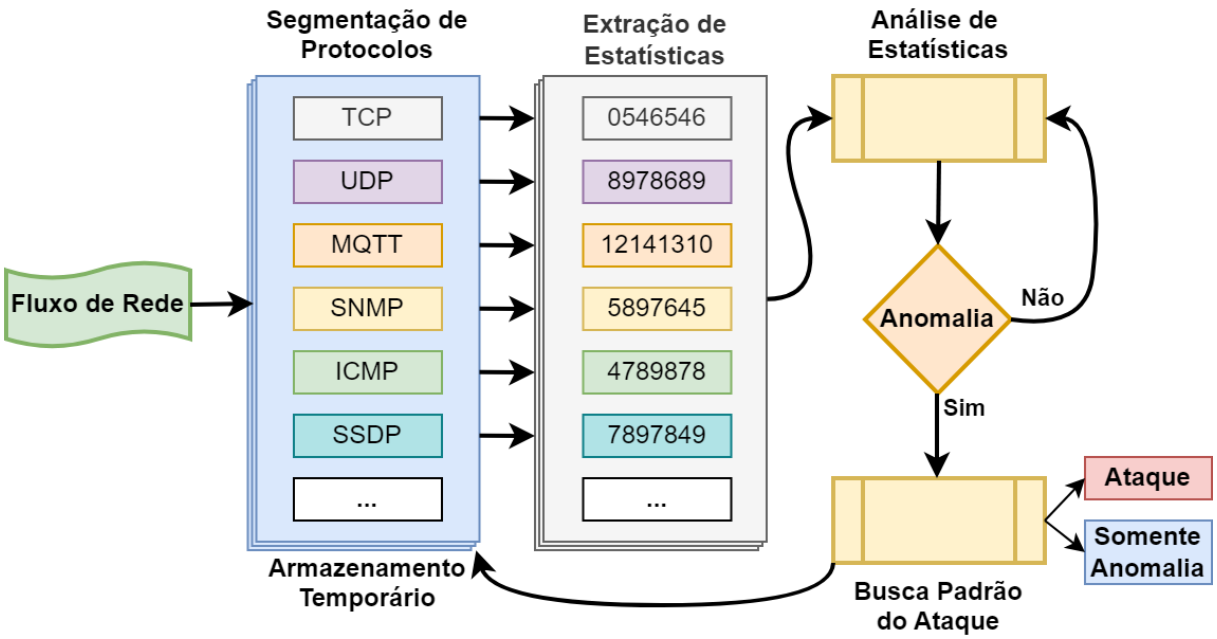
[22] https://github.com/hflautert/AnomalyDetection/blob/main/MQTTset-Malaria.ipynb

5 PROPOSTA

Nesta pesquisa é proposto o uso de janela deslizante e médias móveis para reconhecer as anomalias causadas pelos ataques de negação de serviço com complexidade de $O(n)$ (LAUTERT, H. F.; MACEDO, D. de; PIOLI, 2023b), a menor dentre os trabalhos encontrados na literatura. Além disso, é priorizado o reconhecimento *online* e a parametrização dinâmica para detecção dos ataques de negação de serviço. O modelo foi validado nos *datasets* de (HAMZA *et al.*, 2019), (KANG *et al.*, 2019) e (VACCARI *et al.*, 2020) sendo testado em ataques *TCP SYN flood, Smurf (ICMP), SNMP reflection, SSDP reflection, UDP reflection, HTTP flood e MQTT-Malaria*.

A arquitetura do modelo é apresentada pela Figura 24. Inicialmente é realizada a segmentação por protocolo, ou por tipo de pacote, através da aplicação de filtros *Berkeley Packet Filters* (BPF). Com a segmentação é realizado um armazenamento temporário delimitado por uma fatia de tempo. Nesta pesquisa foram experimentados tempos de 1 minuto, 10 segundos, 1 segundo, 500 milissegundos e 100 milissegundos. Este armazenamento temporário será usado na última etapa do modelo, para buscar o padrão do ataque somente no recorte de tempo com o momento em que a anomalia foi reconhecida.

Figura 24 – Proposta



Fonte: Elaborado pelo autor.

Após o processo de segmentação de protocolos é realizada a extração de estatísticas. Nesta etapa são obtidos o número de pacotes em relação ao tempo de cada protocolo ou tipo de pacote. Uma característica comum dos ataques de negação

de serviço volumétricos, é o aumento repentino de um determinado tipo de pacote ou requisição. Então, ao isolar os protocolos é possível visualizar as variações no número de pacotes em relação ao tempo de cada protocolo ou tipo de pacote, sendo generalizável aos novos tipos de ataques DoS e protocolos. A extração do número de pacotes em relação ao tempo, já segmentados por protocolo, compõe a série de dados que será analisada pelo mecanismo de reconhecimento de anomalias.

Nesta série de dados, é aplicado o algoritmo de janela deslizante, por meio dele são obtidos os valores do intervalo de (k) que será usado para a formação das médias móveis. A execução desta etapa foi realizada por meio de uma função da biblioteca *numpy*, sendo o primeiro passo para obter a média móvel (MM) na Equação (1).

Após obter a média das k últimas medições, é verificado se há um crescimento sucessivo na proporção de ΔG, onde a média atual é ΔG maior que a penúltima, e a penúltima é ΔG maior que a antepenúltima Equação (2). Caso for, o evento é classificado como anomalia, é reconhecido um crescimento anormal de um tipo de protocolo ou tipo de pacote na rede.

$$\begin{aligned} MM_k &= \frac{p_{n-k+1} + p_{n-k+2} \cdots + p_n}{k} \\ &= \frac{1}{k} \sum_{i=n-k+1}^{n} p_i \end{aligned} \tag{1}$$

Limite atingido SE

$$MM(n) > MM(n-1) + \Delta G \wedge MM(n-1) > MM(n-2) + \Delta G \tag{2}$$

Após o reconhecimento de uma anomalia, é realizado uma busca pelo padrão do ataque. Nesta pesquisa foi experimentado o destino e protocolo de maior recorrência no momento da anomalia. Esta estratégia de busca é eficaz nos casos dos ataques que cooptam dispositivos IoT para realizar ataques a um único alvo. Como o posicionamento da proposta é no primeiro concentrador ou roteador da borda da rede, com esta informação é possível reduzir os danos destes tipos ataques. Para ataques com múltiplos alvos ou ataques de baixa frequência outas técnicas de reconhecimento do ataque devem ser experimentadas, incluindo operações de maior complexidade, aproveitando o fato que esta operação acontecerá somente no momento da anomalia.

Mesmo dentro do recorte de redes IoT, cada rede possui características distintas, de tamanho e funcionamento, gerando tráfegos também com características distintas, isto já é percebido na diferença entre os 3 *datasets* (HAMZA *et al.*, 2019), (KANG *et al.*, 2019) e (VACCARI *et al.*, 2020) usados para validação do modelo. Em (HAMZA *et al.*, 2019) observa-se uma linha de pacotes de estabelecimento de conexão (TCPSYN *tcp.flags.syn == 1 and tcp.flags.ack == 0*) mais estável, já em (KANG *et al.*, 2019) esta mesma linha apresenta maior variação. O volume do tráfego MQTT no ataque

MQTT-Malária de (VACCARI *et al.*, 2020) chega a ser 166 vezes maior que tráfego MQTT benigno. Diante desta realidade o modelo proposto conta com 3 variáveis de configuração, apresentadas nas próximas seções.

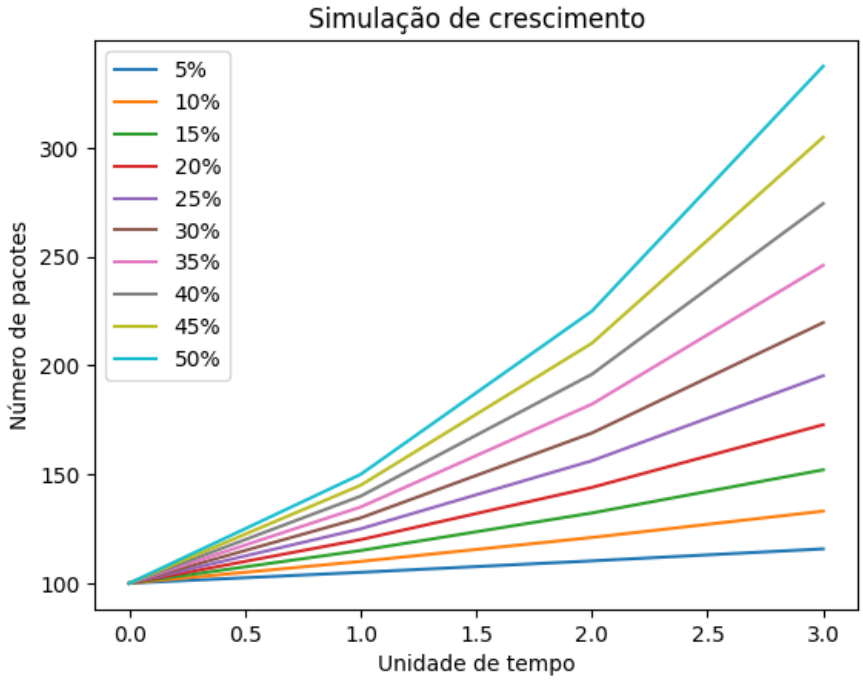
5.1 VARIÁVEIS DO MODELO

Nesta seção será apresentado o efeito de cada variável do sistema. No capítulo seguinte, os resultados em cada unidade de tempo, seguido pelos resultados gerais e configurações das variáveis.

Proporção de crescimento - Delta G

Para diferenciar o tráfego normal de um ataque, o modelo verifica se existe um crescimento sucessivo e consistente de um determinado tipo de protocolo ou pacote. É realizada a comparação entre as 3 últimas médias móveis, sendo que a taxa de crescimento é configurável por porcentagem. Neste ponto é preciso analisar qual proporção a média móvel atual deve ser maior que penúltima, e a penúltima maior que a antepenúltima. Na Figura 25 é visualizado um crescimento hipotético em relação a 100 pacotes por unidade de tempo, sem definição justamente pela possibilidade do uso de qualquer unidade de tempo. Enquanto maior a taxa de crescimento, mais abrupta é a curva.

Figura 25 – Simulação de crescimento no número de pacotes

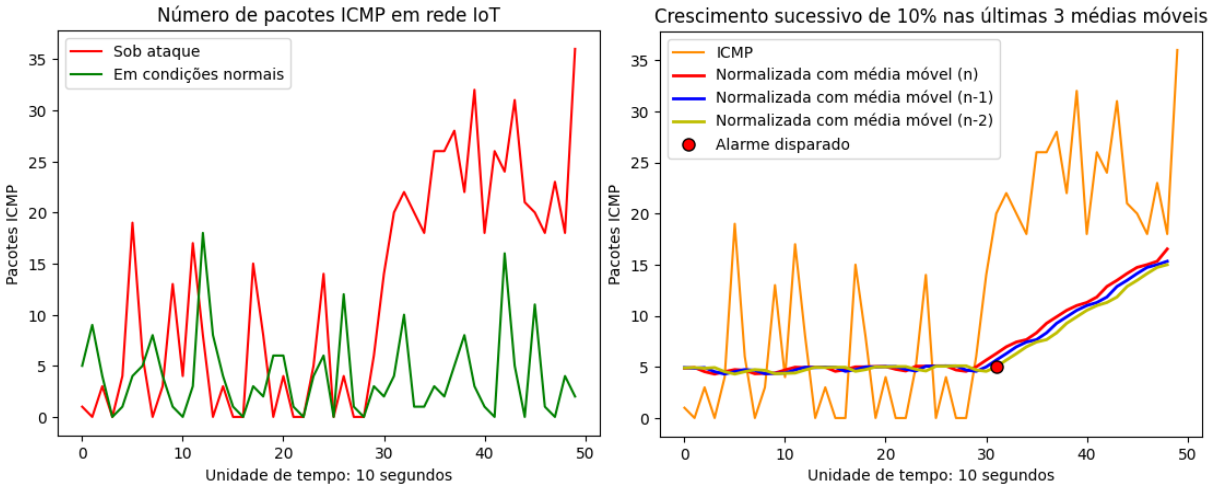


Fonte: Elaborado pelo autor.

Uma taxa mais baixa torna o modelo mais sensível, reconhecendo ataques de menor frequência. Porém, a sensibilidade também gera falsos positivos, o modelo passa reconhecer variações benignas geradas pelos protocolos de rede como um ataque. Uma taxa alta demais, pode deixar ataques passarem sem serem reconhecidos. Portanto, essa taxa deve ser configurada após análise das características de cada rede, como a quantidade de dispositivos, as funções dos dispositivos, frequência dos dados gerados ou monitorados, protocolo de transmissão, dentre outros fatores que influenciam na atividade da rede. Devido à diversidade de dispositivos IoT é difícil predeterminar ou recomendar até mesmo uma faixa de configuração. As configurações experimentadas nesta pesquisa estão relacionadas com os 3 *datasets*, sendo que em um mesmo *dataset* ou ambiente, um protocolo pode ter a configuração de variáveis diferente do outro.

Na Figura 26 a), é visualizado o tráfego ICMP em situação normal, na linha verde, e em situação de ataque, na linha vermelha. Há variações tanto na rede em condições normais, quanto na rede sob ataque, mas no momento do ataque é gerado um tráfego de maior volume, na parte mais à direita da linha vermelha. Já na Figura 26 b), é aplicado com sucesso uma taxa de crescimento de 10% entre últimas médias móveis, ativando o alarme durante o crescimento sucessivo e consistente do tráfego ICMP. As médias móveis se distanciam, de forma que a média móvel atual é 10% maior que a penúltima, e a penúltima 10% maior que a antepenúltima (Equação (2)), o que faz ativar o limite, assim o ataque é reconhecido.

Figura 26 – a) Tráfego ICMP | b) Reconhecimento do ataque

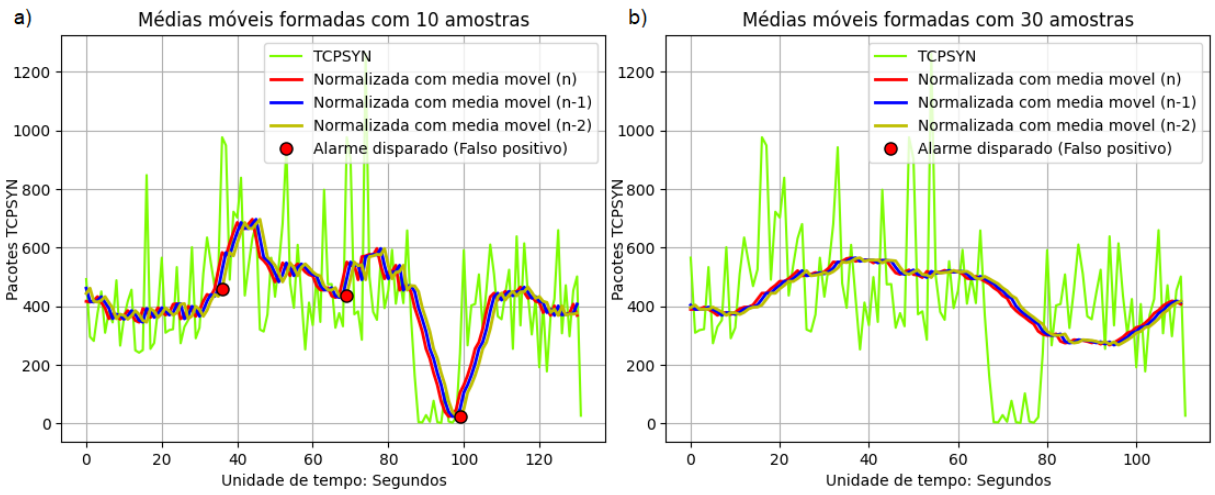


Fonte: Elaborado pelo autor.

Número de amostras para formação da média móvel - k

A segunda variável define o número de amostras para a formação da média móvel, dado por *k* na Equação (1). Um número de amostras maior torna a linha da média móvel mais estável, pois conta com um amplo número de amostras para formar a média. Por outro lado, enquanto menor o número de amostras, mais a média móvel se aproxima do valor unitário de cada amostra, se tornando mais volátil. Na Figura 27 a), é demonstrada a formação da média móvel com o valor de 10 amostras, tornando o modelo demasiado sensível, acarretando falsos positivos. Já na Figura 27 b) o número de amostras é elevado para 30, suavizando as médias formadas e eliminando os falsos positivos.

Figura 27 – Médias móveis formadas com 10 e 30 amostras



Fonte: Elaborado pelo autor.

Como a série de dados pode ser de qualquer tempo, por exemplo, número de pacotes por segundo, ou a cada 100 milissegundos. Esta variável é de extrema importância para ajustar o modelo dentre os diferentes tempos para contabilização dos pacotes. Teoricamente, ao obter sucesso no reconhecimento de anomalias na frequência de 1 segundo, basta alongar o número de amostras para formação da média móvel em 10 vezes, para captar as mesmas anomalias em intervalos de 100ms. Uma vez que 100ms é a décima parte de um segundo.

Como consequência deste alongamento no número de amostras, tem-se a necessidade de um tempo maior para o modelo iniciar a comparação das médias móveis. Considerando a frequência de 1 segundo, ao usar 30 amostras de para formação da média móvel. O modelo levará 30 segundos para iniciar a comparação entre as médias móveis. Passado este tempo de carregamento, a cada segundo serão comparadas as médias móveis, atual com a anterior e a penúltima, levando até 2 segundos

para reconhecimento do ataque. Se reduzir esse intervalo para 20, o modelo levaria apenas 20 segundos para iniciar o carregamento e os mesmos 2 segundos para o reconhecimento do ataque.

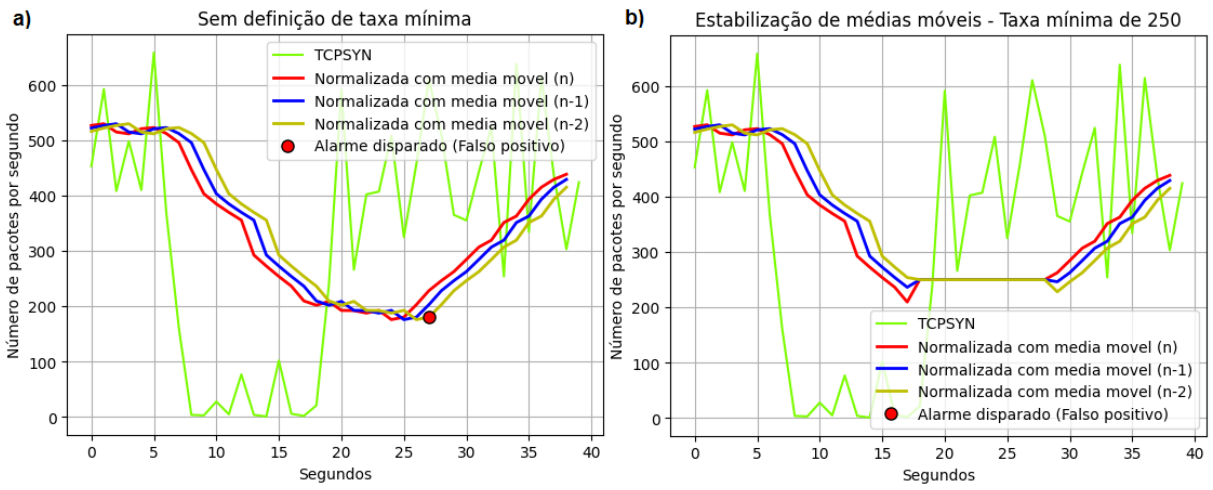
Ao trabalhar com 100ms, mesmo alongando o intervalo de formação de média móvel para 300, resulta nos mesmos 30 segundos para carregamento. Porém, em apenas 200ms para o reconhecimento do ataque. Dentro dessa possibilidade, os limites mínimos para a frequência de tempo, passam a se tornar mais uma questão de custo de processamento do que uma limitação do modelo para o reconhecimento de anomalias próximo do tempo real.

Taxa mínima para considerar a variação das médias móveis

Existem situações onde um protocolo pode não transmitir nada até ser inicializado. Ou seja, passa de 0 pacotes por unidade de tempo para algum valor maior que 0. Esta variação foi reconhecida como ataque pelo modelo proposto inicialmente. Outro caso é a intermitência, por hora transmite, e posteriormente passa um determinado tempo sem transmissão. Ao sair do estado ocioso, de não transmissão para um estado ativo, as médias móveis alertavam um falso positivo.

Para contornar estas situações foi criada esta taxa mínima para considerar a variação das médias móveis. Na prática, as médias móveis só são comparadas caso sejam superiores a esta taxa mínima. Na Figura 28 a), é demonstrado um falso positivo no caso de uma intermitência, já na Figura 28 b), é demonstrado o efeito da taxa mínima, eliminando esse mesmo falso positivo.

Figura 28 – Efeito da taxa mínima em casos de intermitência



Fonte: Elaborado pelo autor.

Um situação indesejável causada pela taxa mínima é tornar o modelo incapaz de reconhecer ataques de baixa frequência. Por esse motivo é recomendável que a

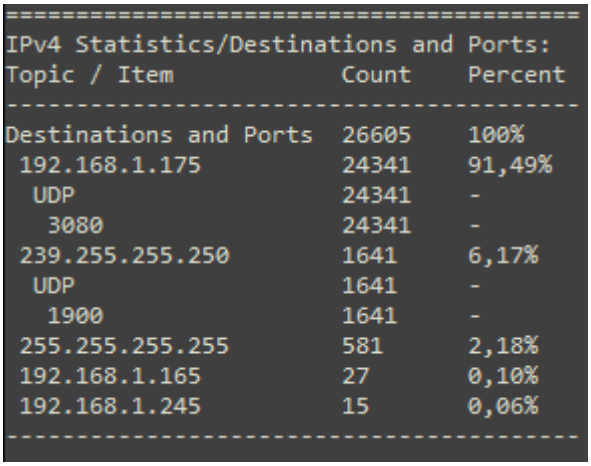
taxa mínima seja usada com cautela, enquanto menor for o valor, maior a capacidade do modelo de reconhecer ataques de baixa frequência. Contudo, também é recomendável que o valor seja maior que 0 (zero), para evitar falsos positivos gerados por intermitência. Aumentar o número de amostras para formação das médias móveis permite o uso de valores menores para taxa mínima, uma vez que a linha da média se torna mais estável.

Busca seletiva de destino

Após o reconhecimento dos ataques, o modelo proposto busca pelo destino, apenas nos últimos arquivos (apresentado anteriormente na Figura 24), onde houve o crescimento abruto no número de pacotes por protocolo. A estratégia de buscar o alvo do ataque após o reconhecimento da anomalia reduz o volume de dados a ser processado, o que contribui para a eficiência do modelo. No caso do ataque SSDP *reflection*, é considerado o volume do tráfego parcial de rede de 1,36GB, quando reduzido ao período em que aconteceu o ataque, e ao protocolo SSDP o volume é reduzido para 14,6MB. A busca do alvo é realizada lendo aproximadamente 1% do tráfego total.

Na Figura 29, gerada por meio da função *Statistics->IPV4 Statistics->Destinations and Ports* no software *WireShark*, observa-se que 91,49% dos pacotes são destinados ao endereço IP 192.168.1.175, que no caso é o alvo do ataque. Abaixo do IP, visualiza-se o protocolo de transporte, neste caso é o UDP e por fim a porta de destino 3080. Nesta etapa pode ser tanto aplicado uma regra para contenção do ataque via *iptables*, quanto enviado o tráfego anômalo para análise profunda em camadas mais altas da rede.

Figura 29 – Busca seletiva de destino - Ataque SSDP reflection



```
==========================================
IPv4 Statistics/Destinations and Ports:
Topic / Item            Count    Percent
------------------------------------------
Destinations and Ports  26605    100%
 192.168.1.175          24341    91,49%
  UDP                   24341    -
   3080                 24341    -
 239.255.255.250        1641     6,17%
  UDP                   1641     -
   1900                 1641     -
 255.255.255.255        581      2,18%
 192.168.1.165          27       0,10%
 192.168.1.245          15       0,06%
------------------------------------------
```

Fonte: Elaborado pelo autor.

5.2 COMPLEXIDADE COMPUTACIONAL

A complexidade computacional é uma métrica importante de ser observada, especialmente na borda da rede, onde temos recursos computacionais limitados. É uma maneira de medir quantos recursos necessitamos para atingir o objetivo. Além do aspecto da viabilidade de implementação de modelos em redes IoT, o aperfeiçoamento dos modelos e algoritmos trazem ganhos operacionais e de eficiência energética (DEAN, 2015).

A complexidade computacional da média móvel é $O(w)$, onde w é o tamanho da janela de dados, que no modelo proposto é k. Então é realizada a soma de todas as amostras e por fim divididos por k (Equação(1)). A operação *online*, necessita de 3 médias móveis $O(3w)$ para o reconhecimento do ataque. Considerando o tráfego de rede ou os arquivos do *dataset* como n, o modelo obtém continuamente a média móvel a partir destes dados, elevando a complexidade de operação para $O(n)$.

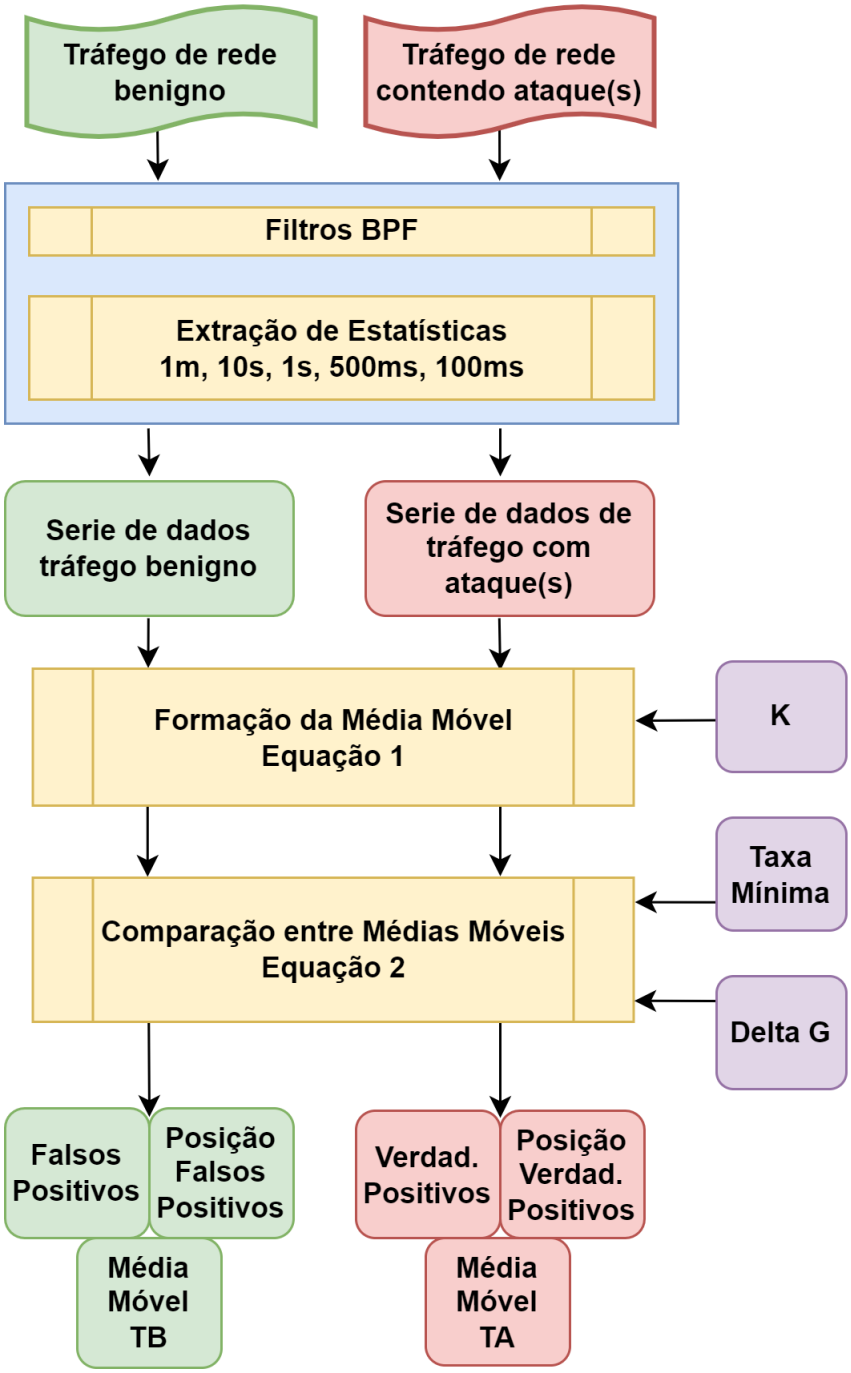
5.3 ETAPAS DE EXPERIMENTAÇÃO E ALGORITMO

A Figura 30 apresenta um fluxograma que descreve as etapas do experimento e a função do algoritmo utilizada para obter os resultados do modelo proposto. Na primeira etapa, na parte superior da figura, é realizada a extração de estatísticas, tanto do tráfego benigno, quanto do tráfego contendo ataques. Dentro desta etapa há dois processos. O primeiro, de aplicação do fitro BPF, que separa somente o tráfego de um determinado protocolo, ou tipo de pacote. Após a aplicação do filtro, os pacotes são contabilizados em relação ao tempo. Nesta pesquisa, foram experimentados os tempos, de 1 minuto, 10 segundos, e 1 segundo, 500 milissegundos e 100 milissegundos. Como resultado desta primeira etapa, obtêm-se duas séries de dados. A série de dados contendo os pacotes em relação ao tempo, do tráfego de rede benigno, e do tráfego de rede contendo um ou mais ataques.

Concluída a primeira etapa, o modelo contém 5 dados de entrada. Duas séries de dados, e as 3 variáveis que calibram o modelo. Então é formada a média móvel, conforme a variável k, e posteriormente aplicada a regra de comparação entre a média móvel atual (n), com a anterior (n-1) e a antepenúltima (n-2). A comparação só acontece se for maior que a taxa mínima. Já a proporção em que é comparada, é ajustada pelo valor de ΔG.

Enfim, o modelo retorna 6 dados de saída: os falsos positivos, a posição dos falsos positivos, a média móvel tráfego benigno (TB), os verdadeiros positivos, sua posição, e a média móvel do tráfego com ataques (TA). Nesta fase experimental, estes dados de saída, auxiliam na compreensão de funcionamento geral do modelo, para entender em que momento o ataque ou falso positivo está sendo reconhecido, bem como verificar os efeitos dos ajustes nas variáveis de configuração.

Figura 30 – Fluxograma do algoritmo



Fonte: Elaborado pelo autor.

5.4 COMPONENTES PARA PROTÓTIPO

Atualmente existem diversos experimentos de aprendizado de máquina para reconhecimento de anomalias em redes IoT, apesar da alta precisão, o desafio é viabilizar a implementação na borda da rede, com aprendizado *online*, usando apenas uma janela de tempo para alimentar o sistema de reconhecimento (COOK; MISIRLI; FAN, 2019). O modelo proposto nesta pesquisa foi desenvolvimento com foco em baixa complexidade computacional para maior viabilidade de implementação, bem como eficiência energética. Nesta seção é apresentado um conjunto de ferramentas de código aberto que permitem a implementação do modelo em *hardwares* de roteadores de borda acessíveis.

Para executar esse conjunto de componentes, é necessário um sistema operacional, para tal tarefa foi escolhido o *OpenWrt*. Um sistema operacional *Linux* voltado para dispositivos embarcados, com gerenciamento de pacotes, ampla compatibilidade de hardware e altamente personalizável. O projeto foi criado em 2004, mantendo desenvolvimento contínuo, tendo a versão estável mais recente de abril de 2023, e a experimental de agosto de 2023. Devido à versatilidade quanto ao seu uso, vem sendo utilizado em pesquisas e experimentos de segurança na borda da rede (PALMESE; REDONDI; CESANA, 2022) e sistemas de posicionamento (ESTRADA *et al.*, 2023).

Conforme observado anteriormente, na Figura 24, a primeira etapa do processo é a captura do tráfego de rede e armazenamento em uma área temporária, denominada *buffer*. Para esta etapa é designado o *daemonlogger*, um registrador de pacotes rápido projetado especificamente para uso em ambientes de gerenciamento de segurança de rede (ROESCH, 2023). Além da compatibilidade com o software *OpenWrt*[1], esta ferramenta oferece recursos fundamentais para a prototipagem do modelo. O comando a seguir será descrito em detalhes no próximo parágrafo.

```
$ daemonlogger eth0 -t 60 -M 75 \
"tcp.flags.syn == 1 and tcp.flags.ack == 0"
```

Após a chamada da ferramenta *daemonlogger*, é descrita a interface de coleta, *eth0*. Selecionada a interface, é definido o intervalo de tempo da captura, neste exemplo *-t 60*, indica 60 segundos. O parâmetro "-M 75" grava arquivos no disco até que ele seja 75% utilizado, então exclui o arquivo mais antigo no diretório de registro. Isto evita que em casos de ataques de inundação intensivos, o disco lote. Por fim é definido o tipo de tráfego a ser capturado, tendo como padrão os filtros BPF.

A próxima etapa consiste na extração de estatísticas do tráfego capturado. Esta etapa é realizada com a ferramenta *scapy*[2], um poderoso manipulador de pacotes

[1] https://openwrt.org/packages/pkgdata/daemonlogger
[2] https://github.com/secdev/scapy

baseado em *python*, também compatível com o *OpenWrt*[3]. A extração é realizada por meio de um *script*, desenvolvido por (DI VITA, 2019). Feito isso, a aplicação do modelo é realizada puramente em *pyhton* [4], suportado pelo *OpenWrt*[5]. Para envio de alerta, é utilizado o *mailsend*, compatível e disponível via gerenciador de pacotes.

Todas as ferramentas descritas podem ser vizualizadas na Tabela 2. Observa-se que a soma do espaço ocupado por todas as ferramentas, atinge não mais que 40 MB, o que viabiliza a implementação em diversos hardwares que operam na de borda da rede compatíveis com este sistema operacional.

Tabela 2 – Ferramentas para protótipo

Software	Versão	Tamanho
OpenWrt	21.02.2	30 MB
iptables	1.8.7	Incluso
python3	3.10.12-1	8,5 MB
scapy	2.4.5-1	1,5 MB
daemonlogger	1.2.1-1	10 KB

Fonte: Elaborado pelo autor.

[3] https://openwrt.org/packages/pkgdata/scapy
[4] https://github.com/hflautert/AnomalyDetection
[5] https://openwrt.org/packages/pkgdata/python3

6 RESULTADOS EXPERIMENTAIS

Este modelo foi testado em 3 *datasets* (HAMZA *et al.*, 2019), (KANG *et al.*, 2019) e (VACCARI *et al.*, 2020), com 7 diferentes tipos de ataque e protocolos: *TCP SYN flood, Smurf(ICMP), UDP(Fraggle), SNMP reflection, SSDP reflection, HTTP flood(Mirai) e MQTT-Malária*. Somados todos os experimentos, em diferentes intervalos de tempo, de 1m, 10s, 1s, 500ms e 100ms, chegou-se a um total de 423 amostras de ataques analisadas, onde 398 foram reconhecidos com sucesso. Ocorreram 4 casos de falsos positivos. A precisão geral do modelo ficou em 93,21%. Todos os experimentos realizados, bem como os arquivos que alimentam o modelo, estão disponíveis na plataforma GitHub[1], em Jupyter Notebook. (LAUTERT, H., 2023).

6.1 REDUÇÃO DE DADOS

A proposta do modelo é reconhecer ataques com filtro de protocolo, para a obter o número de pacotes segmentados por protocolo. Em relação ao conteúdo integral do tráfego de rede, há uma grande redução de dados. Somando os arquivos pcaps dos *datasets* (HAMZA *et al.*, 2019), (KANG *et al.*, 2019) e (VACCARI *et al.*, 2020) analisados pelo modelo, chegou-se a um total de 16,6GB. Já a soma de todos os arquivos csv, com a extração dos pacotes segmentados por protocolos em relação aos tempos de 1m, 10s, 1s, 500ms e 100ms, resulta em apenas 47MB. Portanto, o modelo necessitou apenas 0,28% do conteúdo integral do tráfego de rede para realizar o reconhecimento das anomalias. Representando uma redução de dados de 99,72%, uma prática recomendada para sistemas de IoT (PIOLI *et al.*, 2022).

Adicionalmente foi realizada uma comparação por amostragem entre as estatísticas extraídas do conteúdo integral da rede e as estatísticas necessárias para o reconhecimento da anomalia. A amostra do conteúdo integral da rede totalizou 2,68GB, já as estatísticas extraídas somaram 750MB. Por fim, as estatísticas contendo somente o número de pacotes em relação aos 5 diferentes tempo de cada protocolo em que o modelo foi testado, somaram 722KB. Neste método, utilizando não o conteúdo integral da rede, mas as estatísticas já extraídas como referência, a redução de dados ficou em 96,71%. Dessa forma, o modelo necessitou apenas 3,29% dos dados em relação a extração completa das estatísticas.

Tamanha redução só foi possível pelo fato deste modelo ser alimentado pelo número de pacotes segmentados por protocolos. Após a aplicação do filtro de protocolo, o modelo busca reconhecer o ataque basicamente pela variação no número de pacotes, o que resulta em único dado ou recurso (*feature*) a ser processado. Diferente dos modelos pré-concebidos de aprendizado de máquina necessitam de diversos de recursos, aumentando quantidade de dados a serem processados, o que dificulta o

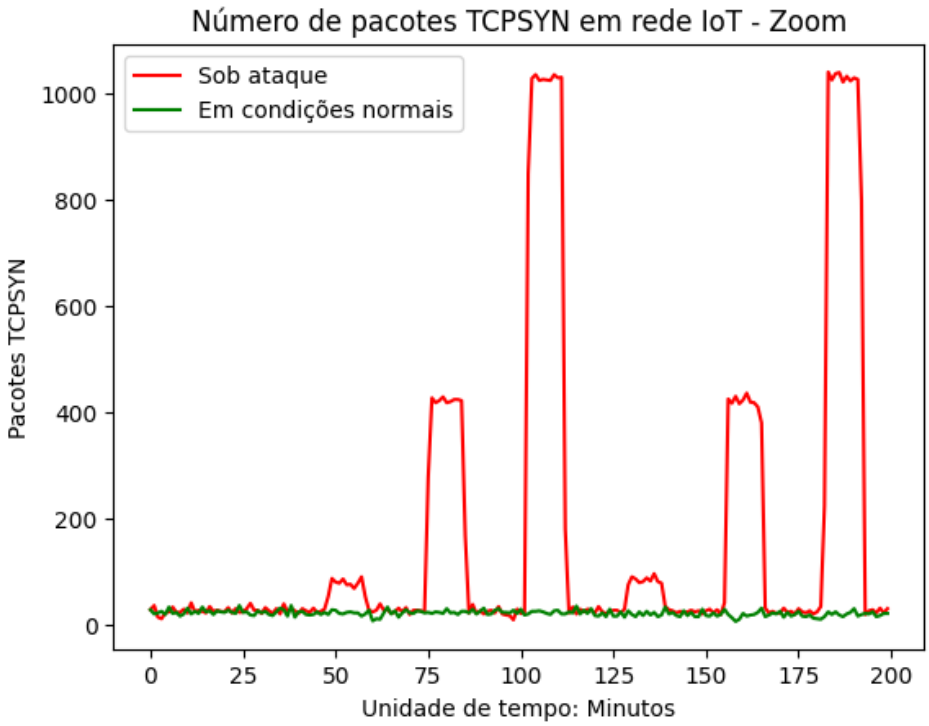
[1] https://github.com/hflautert/AnomalyDetection

processamento em pequenos dispositivos na borda da rede. Reunir todos os recursos e simplesmente delegá-los ao processo de Principal Component Analysis (PCA) para a seleção dos principais recursos pode até mesmo comprometer a precisão no reconhecimento de ataques (SAGHEZCHI *et al.*, 2022).

6.2 PACOTES SEGMENTADOS POR PROTOCOLOS A CADA MINUTO

Nos testes realizados com a frequência de um minuto, foi atingida a maior precisão, onde os resultados tendem a 100% de precisão no reconhecimento dos ataques. Obter o número de pacotes, segmentados por protocolo, a cada minuto resulta em uma linha mais definida, com menos amostras discrepantes e os momentos dos ataques bem definidos e condensados, antes mesmo de aplicar as médias móveis. A Figura 31 traz a linha do tráfego normal em verde, e a linha do tráfego com ataques em vermelho. Nela é possível visualizar com nitidez a diferença entre os dois tipos de tráfego, na linha verde pequenas variações, já na linha em vermelho, formam-se barras crescentes, geradas pelos ataques de 1, 10 e 100 pacotes por segundo.

Figura 31 – Tráfego normal e ataque TCPSYN

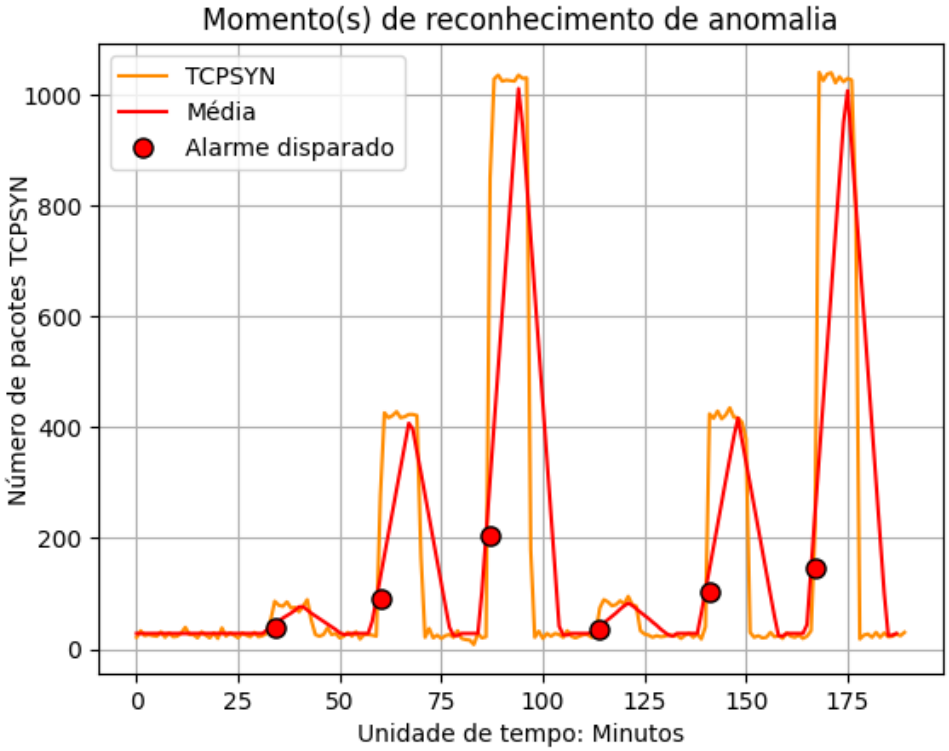


Fonte: Elaborado pelo autor.

Com a frequência de 1 minuto, a média móvel é aplicada em cima de linhas já estabilizadas, ressaltando o aumento no volume de pacotes gerados no momento do ataque. A Figura 32 exibe o tráfego gerado pelos ataques, a média móvel, e momento

de identificação da anomalia, quando há o crescimento sucessivo e consistente no número de pacotes. Nos demais protocolos foi observado o mesmo padrão, com linhas estáveis e tráfego de ataque consolidado.

Figura 32 – Identificação de anomalias



Fonte: Elaborado pelo autor.

A variável ΔG não necessitou ajustes, foi de 10% em todos os protocolos. Assim como o número de amostras para formação da média móvel, que foi de 10 para todos os casos. A única variável que precisou ajustes foi a da taxa mínima, com destaque para o protocolo SSDP, onde a inicialização, parte que passa de transmitindo zero para aproximadamente 450 pacotes por minuto, foi estabilizada através da taxa mínima. Os resultados de todos os protocolos podem ser visualizados na Tabela 3. Alguns ataques do mesmo tipo, estão separados por conter tráfegos de diferentes arquivo do *dataset*, especificados anteriormente no Capítulo 4.

Apesar de ser muito preciso, nesta configuração o modelo pode leva 2 minutos para reconhecer o ataque. O que em alguns casos pode levar a indisponibilidade do roteador de borda, e do próprio sistema de reconhecimento de anomalias. O modelo faz a comparação das 3 últimas médias e cada nova média móvel leva um minuto para ser formada e então comparada com a anterior, considerando que a média atual já está formada, é necessário mais um minuto para comparar a atual com anterior, e outro minuto para comparar a anterior com a antepenúltima. Nas seções seguintes,

foram realizados testes em frequências de tempo menores, de forma gradual.

Tabela 3 – Resultados com frequência de 1 minuto

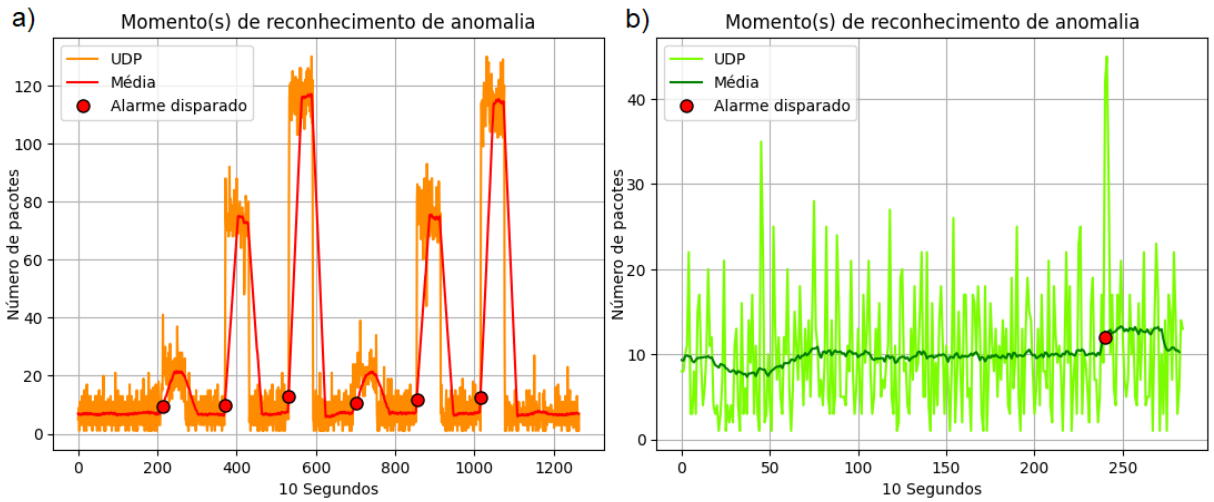
Dataset	Tipo de Ataque	Lançado	Detectado	F.Positivo	Precisão
UNSW-2019 (HAMZA *et al.*, 2019)	TCP SYN	30	30	0	100%
	TCP SYN	7	7	0	100%
	UDP (Fraggle)	6	6	0	100%
	ICMP (Smurf)	22	22	0	100%
	ICMP (Smurf)	7	7	0	100%
	SNMP reflection	6	6	0	100%
	SSDP reflection	3	3	0	100%
	SSDP reflection	3	3	0	100%
Geral		84	84	0	100%

Fonte: Elaborado pelo autor.

6.3 PACOTES SEGMENTADOS POR PROTOCOLOS A CADA 10 SEGUNDOS

Na frequência de 10 pacotes por segundo, a precisão recuou para 97,67%. No protocolo UDP foi encontrado o primeiro falso positivo. Na Figura 33 a) é demonstrado o reconhecimento de 6 ataques UDP (Fraggle), na Figura 33 b) o tráfego normal com um falso positivo, ambos a configuração da taxa mínima em 5. A Figura 34 demonstra a mesma situação de ataque, com a taxa mínima elevada de 5 para 10, é possível perceber que um ataque, circulo vermelho mais a esquerda, já deixa de ser reconhecido, com o falso positivo ainda existindo.

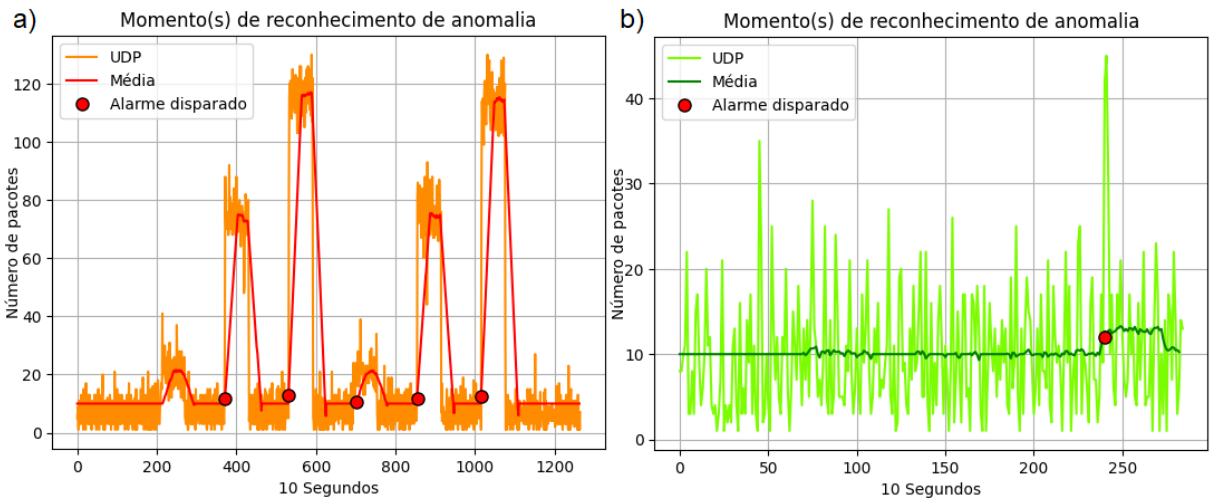
Figura 33 – Reconhecimento de ataques com taxa mínima de 5



Fonte: Elaborado pelo autor.

Foi realizado tentativa de aumentar o número de amostras, mas ocorreu o mesmo efeito, perda de reconhecimento de ataque real, sem desarmar o falso po-

Figura 34 – Reconhecimento de ataques com taxa mínima de 10



Fonte: Elaborado pelo autor.

sitivo. A única maneira eliminar o falso positivo, foi aumentando o número de amostras de 35 para 60, dessa forma seriam reconhecidos 4 de 6 ataques reais, sem falso positivo, mas assim a precisão seria reduzida. Então foi mantido a configuração para o reconhecimento dos 6 ataques reais, com somente um falso positivo. A mesma situação foi observada no protocolo ICMP, onde foi encontrado outro falso positivo. Nos demais casos a precisão foi de 100%.

Na redução da frequência de um minuto para 10 segundos, percebem-se algumas amostras de número de pacotes apresentando maior variação, como na Figura 34 b), onde existem picos, em verde, distantes da média. Mesmo com os ataques ainda bem definidos, como se observa na Figura 34 a), percebe-se que as amostras representadas pelas linhas laranjas apresentam maior volatilidade que na frequência de 1 minuto (Figura 32). A maioria das situações pode ser contornada com ajuste da taxa mínima, número de amostras para formação da média móvel e ΔG. Porém coletar amostras em tempos menores, pode gerar este efeito de pontos discrepantes, com potencial de afetar a precisão do modelo proposto. Os resultados de todos os protocolos podem ser visualizados na Tabela 4.

6.4 PACOTES SEGMENTADOS POR PROTOCOLOS A CADA SEGUNDO

Na frequência de pacotes segmentados por protocolos por segundo, a precisão recuou para 90%. O maior desafio são os ataques de baixa frequência, tanto no protocolo TCP, em ataques do tipo TCPSYN, quanto no protocolo ICMP, em ataques ICMP(Smurf). Em ambos os casos, o modelo não pode reconhecer parte dos ataques na frequência de 1 pacote por segundo. Aumentar a sensibilidade do modelo, reduzindo

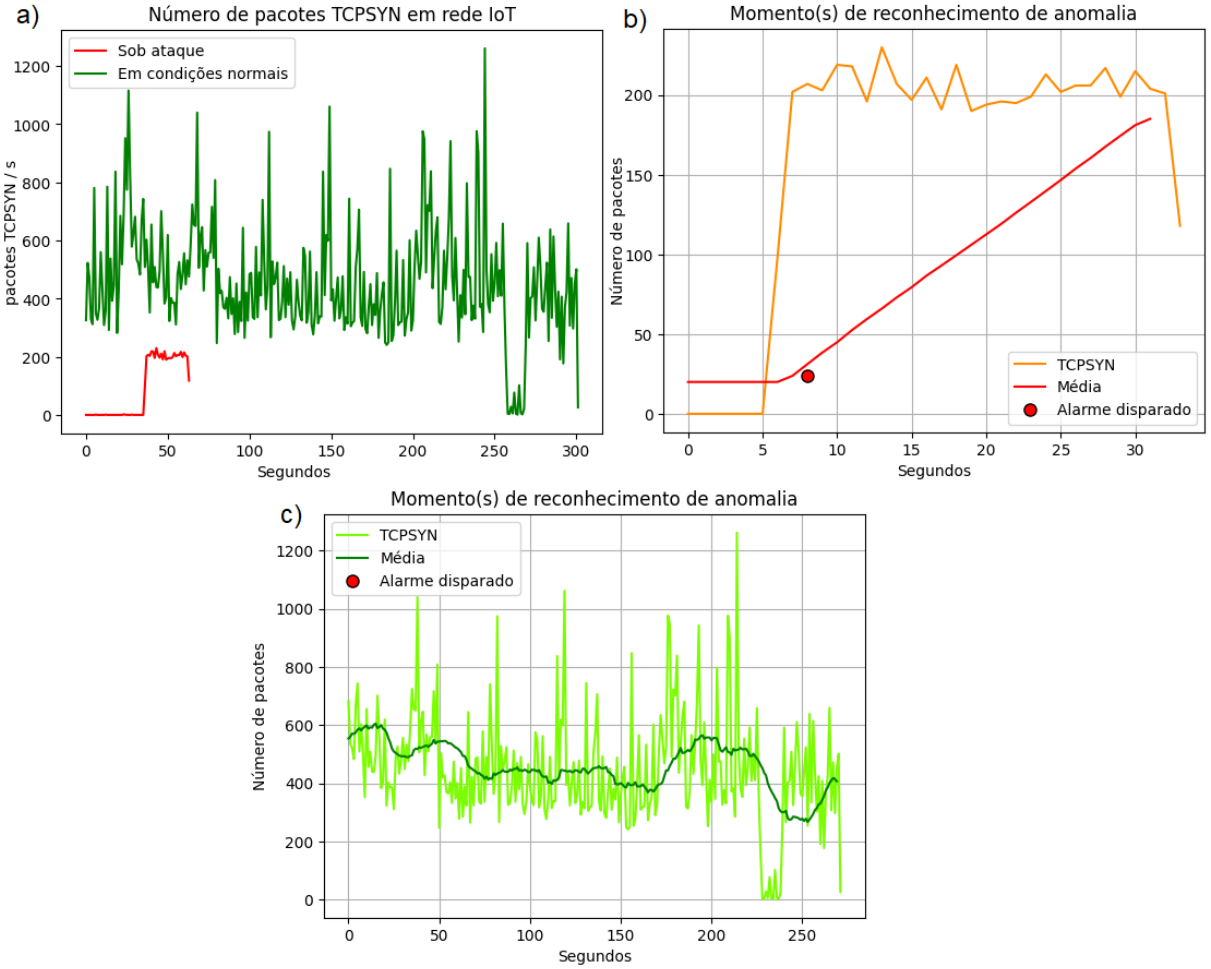
Tabela 4 – Resultados com frequência de 10 segundos

Dataset	Tipo de Ataque	Lançado	Detectado	F.Positivo	Precisão
UNSW-2019 (HAMZA *et al.*, 2019)	TCP SYN	30	30	0	100%
	TCP SYN	7	7	0	100%
	UDP (Fraggle)	6	6	1	85,71%
	ICMP (Smurf)	22	22	1	95,65%
	ICMP (Smurf)	7	7	0	100%
	SNMP reflection	6	6	0	100%
	SSDP reflection	3	3	0	100%
	SSDP reflection	3	3	0	100%
Geral		84	82	2	97,62%

Fonte: Elaborado pelo autor.

a variável ΔG, o número de amostras para formação da média móvel e taxa mínima, neste caso gerou falsos positivos em maior proporção que o reconhecimento dos ataques reais, reduzindo ainda mais a precisão do modelo nesta frequência de 1 segundo.

Figura 35 – Reconhecimento de variações em pacotes TCPSYN



Fonte: Elaborado pelo autor.

Por outro lado, o modelo demonstrou boa capacidade de reconhecer variações geradas pelo ataque, mesmo quando o pico no número de pacotes é menor do que o volume do tráfego benigno. Devido à concepção do modelo, o que é fundamental para o sucesso no reconhecimento, é que exista uma variação com proporção significativa. Na Figura 35 a), originada do *dataset* de (KANG *et al.*, 2019), vemos a linha em verde, representando o tráfego benigno, e a linha vermelha representando o tráfego de um ataque TCPSYN. Apesar da linha vermelha ter valores absolutos menores que a linha verde, a proporção de sua variação é maior. Na Figura 35 b) vemos o ataque sendo identificado. Por fim, na Figura 35 c) vemos o tráfego benigno sem falsos positivos, uma vez que a proporção da sua variação é menor que a gerada pelo ataque. Os resultados de todos os protocolos podem ser visualizados na Tabela 5.

Tabela 5 – Resultados com frequência de 1 segundo

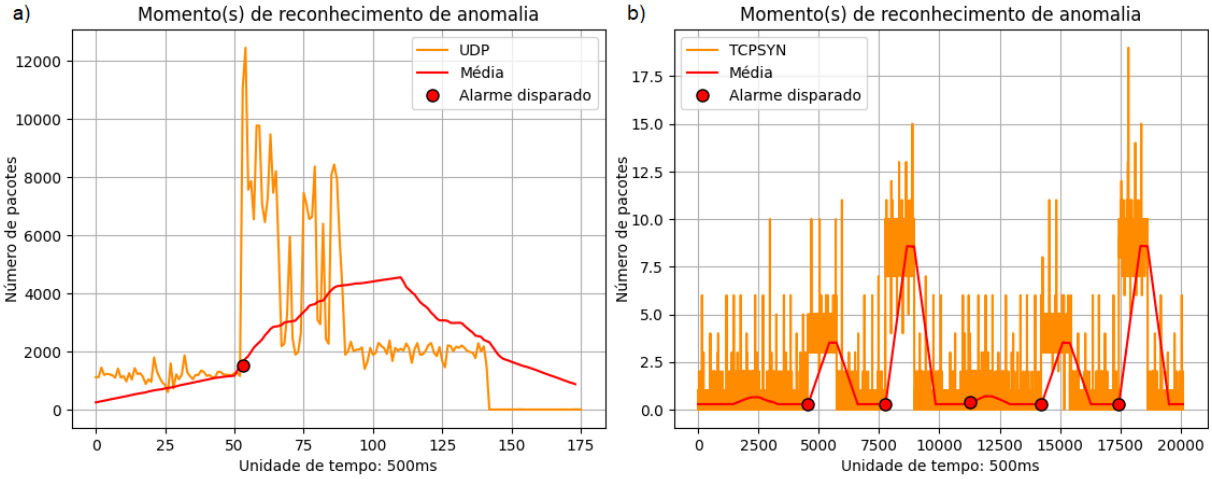
Dataset	Tipo de Ataque	Lançado	Detectado	F.Positivo	Precisão
UNSW-2019 (HAMZA *et al.*, 2019)	TCP SYN	30	26	0	86,67%
	TCP SYN	7	7	0	100%
	UDP (Fraggle)	6	6	0	100%
	ICMP (Smurf)	22	18	1	78,26%
	ICMP (Smurf)	7	6	0	85,71%
	SNMP reflection	6	6	0	100%
	SSDP reflection	3	3	0	100%
	SSDP reflection	3	3	0	100%
IOTnInt-2019 (KANG *et al.*, 2019)	TCP SYN	6	6	0	100%
	UDP (Mirai)	4	4	0	100%
	HTTP (Mirai)	4	4	0	100%
MQTTset-2020 (VAC-CARI *et al.*, 2020)	MQTT(Malaria)	1	1	0	100%
Geral		99	90	1	90%

Fonte: Elaborado pelo autor.

6.5 PACOTES SEGMENTADOS POR PROTOCOLOS A CADA 500 MILISSEGUNDOS

Na frequência de 500 milissegundos, os resultados foram semelhantes ao de 1 segundo, ressaltando a dificuldade dos ataques de baixa frequência, que dessa vez também afetaram o reconhecimento dos ataques TCPSYN, UDP (Fraggle) e ICMP (Smurf). Na Figura 36 a) observa-se o reconhecimento parcial dos ataques do tipo TCPSYN do *datasets* de (HAMZA *et al.*, 2019). Em ataques que geram uma variação maior, como no UDP(Mirai) de (KANG *et al.*, 2019), o modelo reconhece o ataque com sucesso Figura 36 b). Portanto, há potencial para funcionamento do modelo mesmo em frequências de tempos menores que um segundo. Os resultados de todos os protocolos podem ser visualizados na Tabela 6.

Figura 36 – Reconhecimento de ataques TCPSYN e UDP(Mirai) a cada 500ms



Fonte: Elaborado pelo autor.

Tabela 6 – Resultados com frequência de 500 milissegundos

Dataset	Tipo de Ataque	Lançado	Detectado	F.Positivo	Precisão
UNSW-2019 (HAMZA *et al.*, 2019)	TCP SYN	30	26	0	86,67%
	TCP SYN	7	7	0	100%
	UDP (Fraggle)	6	5	0	83,33%
	ICMP (Smurf)	22	18	0	81,82%
	ICMP (Smurf)	7	6	0	85,71%
	SNMP reflection	6	6	0	83,33%
	SSDP reflection	3	3	0	100%
	SSDP reflection	3	3	0	100%
IOTnInt-2019 (KANG *et al.*, 2019)	TCP SYN	6	6	0	100%
	UDP (Mirai)	4	4	0	100%
	HTTP (Mirai)	4	4	0	100%
MQTTset-2020 (VACCARI *et al.*, 2020)	MQTT(Malaria)	1	1	0	100%
Geral		99	88	0	88,89%

Fonte: Elaborado pelo autor.

6.6 PACOTES SEGMENTADOS POR PROTOCOLOS A CADA 100 MILISSEGUNDOS

Nesta frequência algumas amostras precisaram ser recortadas em partes menores, mais próximas da janela do ataque. Ao extrair o número de pacotes nessa frequência em arquivos maiores como no *dataset* de (HAMZA *et al.*, 2019), a extração não era gerada de forma completa pelo software *WireShark*. Dessa forma, o número de amostras foi um pouco menor, ficando em 57 no total, das quais 52 foram reconhecidas com sucesso, tendo um falso positivo. Novamente o modelo encontrou limitações

Tabela 7 – Resultados com frequência de 100 milissegundos

Dataset	Tipo de Ataque	Lançado	Detectado	F.Positivo	Precisão
UNSW-2019(HAMZA *et al.*, 2019)	TCP SYN	9	7	1	70%
	TCP SYN	6	4	0	66,67%
	UDP (Fraggle)	6	6	0	100%
	ICMP (Smurf)	6	6	0	100%
	ICMP (Smurf)	3	3	0	100%
	SNMP reflection	6	5	0	83,33%
	SSDP reflection	3	3	0	100%
	SSDP reflection	3	3	0	100%
IOTnInt-2019 (KANG *et al.*, 2019)	TCP SYN	6	6	0	100%
	UDP (Mirai)	4	4	0	100%
	HTTP (Mirai)	4	4	0	100%
MQTTset-2020 (VAC-CARI *et al.*, 2020)	MQTT(Malaria)	1	1	0	100%
Geral		57	52	1	89,66%

Fonte: Elaborado pelo autor.

para reconhecer ataques de baixa frequência, onde a proporção da variação não é suficiente para ativar seu mecanismo de reconhecimento. Contudo, a precisão ficou em 89,66% para o reconhecimento de ataques em até 200 milissegundos a uma complexidade de $O(n)$. Os resultados de todos os protocolos podem ser visualizados na Tabela 7.

6.7 PRECISÃO E CONFIGURAÇÕES GERAIS

A precisão geral em cada tempo, juntamente com a configuração aproximada do ΔG, número de amostras para a formação das médias móveis, taxa mínima e inicialização (em minutos) pode ser visualizado na Tabela 8. Nela observa-se a queda na precisão enquanto menor a frequência de tempo, com uma leve quebra de tendência no tempo de 100ms. Possivelmente devido ao recorte mais próximo das amostras de ataque, porque em uma série de dados mais longa, provavelmente a precisão apresentará uma queda, visto que há maior probabilidade de falsos positivos com tempo maior de tráfego de rede.

Tabela 8 – Precisão e Configurações Gerais

Frequência de Tempo	Precisão (%)	ΔG	Número de Amostras	Taxa Mínima	Inicialização (Minutos)	Tempo de Resposta
1 Minuto	100	10%	10	75,38	10	2m
10 Segundos	97,67	10%	41	15,00	6,77	20s
1 Segundos	90	5%	300	7,50	4,98	2s
500 Milissegundos	88,89	4%	448	3,68	3,73	1s
100 Milissegundos	89,69	4%	2213	1,13	3,69	200ms

Fonte: Elaborado pelo autor.

O ΔG tem tendência inversa ao número de amostras, como nas unidades de tempo menores há um número maior de amostras para formar a média, as linhas se tornam mais suaves, então a margem, ou porcentagem para reconhecer a anomalia acaba sendo menor. Já a taxa mínima, tende a ser sempre menor nas frequências tempo menores, porque o número de pacotes também é menor. Por fim, chega-se ao tempo de inicialização necessária em cada uma das faixas de tempo. O modelo necessita do número de amostras multiplicada pela frequência de tempo para formar base para as comparações. Após a inicialização, a anomalia é detectada com 2 unidades de tempo, como a média móvel (n) já está formada, é necessária uma unidade de tempo para comparar com média anterior (n-1), o outra para comparar com a antepenúltima (n-2).

7 DISCUSSÃO

Nesta pesquisa foi desenvolvido um modelo experimental para reconhecimento de ataques de DoS em redes IoT, por meio do método de análise de anomalias. Observando a característica comum dos ataques DoS volumétricos, o aumento no número de determinados tipos de pacotes em relação ao tempo, foi proposto um modelo que após a aplicação de um filtro BPF, é alimentado por uma série de dados de um único recurso, reduzindo em até 99,72% a quantidade de dados a serem analisados para o reconhecimento de uma anomalia. O mecanismo de reconhecimento adota a técnica de janela deslizante para a formação de médias móveis e comparação de crescimento proporcional, mantendo a complexidade de $O(n)$ para o reconhecimento de ataques DoS.

Durante todo o processo de desenvolvimento, foi dado enfase a baixa complexidade, com objetivo de viabilizar a implementação, bem como reduzir o custo energético e computacional deste tipo de desafio na borda da rede. Visto que por meio do mapeamento sistemático da literatura (MSL), foi identificado uma lacuna quanto a aplicabilidade das soluções de modelos de aprendizado de máquina, que na maioria das vezes necessitam ler estatísticas de todo *dataset* para tornarem-se capazes de identificar o que é do que não é um ataque DoS.

A análise dos resultados do MSL permitiram uma aproximação do estado da arte sobre o reconhecimento de ataques de DoS em redes IoT, por meio do método de análise de anomalias. As bases para a proposição do modelo foram a segmentação do processo de reconhecimento em mais de uma etapa, demonstradas por (YAEGASHI; TAKESHITA; NAKAYAMA, 2022) e (SALAHUDDIN *et al.*, 2021), onde é primeiramente reconhecida a anomalia e posteriormente submetido a análise de maior profundidade, contribuindo para eficiência no processo de reconhecimento do ataque. Outra inspiração foi o uso de médias móveis em modelos de aprendizado de máquina, demonstradas por (JIA *et al.*, 2022) e (SAGHEZCHI *et al.*, 2022).

Na Tabela 9 é possível observar que a complexidade declarada pelos trabalhos é maior que linear, devido ao custo computacional gerado pelos processos de aprendizado de máquina na leitura do tráfego de rede. De todos os trabalhos que abordaram a complexidade, nenhum abordou o tempo de resposta, nos campos identificado como não abordado (NA). Para situar a proposta quanto ao tempo de resposta, foram adicionados os melhores tempos de resposta dentro do escopo da MSL, contudo, apesar de tempo de resposta baixíssimo, como alcançado por (SANTOYO-GONZÁLEZ; CERVELLÓ-PASTOR; PEZAROS, 2019), nenhum trabalhado que abordou o tempo de resposta, abordou a complexidade computacional, que é crucial devido à limitação de hardware na borda da rede, e sempre importante devido ao custo energético. Dessa forma, o modelo de Micro-IDS proposto nesta pesquisa, se destaca pelo con-

Tabela 9 – Comparação com trabalhos relacionados

Autor e Ano	Rec. Online	Complexidade	Precisão	Tempo de Resposta
(HE *et al.*, 2020)	Sim	$O(i * 3n)$	91,7%	NA
(DANESHGADEH *et al.*, 2018)	Não	$O(n^2)$	86,7%	NA
(BHUVANESWARI AMMA; SELVAKUMAR, 2021)	Não	$O(n^3)$	97,03%	NA
(BOUYEDDOU *et al.*, 2021)	Não	$O(n^2)$	99,9%	NA
(LI, F. *et al.*, 2019)	Não	$O(n^2) + O(n)$	94,7%	NA
(JIA *et al.*, 2022)	Não	$O(n^2)$	97,26%	NA
(MERGENDAHL; LI, J., 2020)	Sim	$O(n^2)$	97%	NA
(ANJUM *et al.*, 2022)	Não	$O(th) + O(n)$	99%	NA
(SANTOYO-GONZÁLEZ; CERVELLÓ-PASTOR; PEZAROS, 2019)	Sim	NA	95%	0,005s
(DOSHI; YILMAZ; ULUDAG, 2021)	Sim	NA	NA	0,2s
(BINU; MOHAN; HARIDAS, 2021)	Sim	NA	97,3%	2,3s
(BHALE; BISWAS; NANDI, 2019)	Sim	NA	95%	6,47s
(ANTHI *et al.*, 2019)	Não	NA	97%	41,8s
(SAHOO; PUTHAL, 2020)	Sim	NA	96,05%	120s
(LAUTERT, H. F.; MACEDO, D. de; PIOLI, 2023a) (a cada 10s)	Sim	$O(n)$	97,67%	20s
(LAUTERT, H. F.; MACEDO, D. de; PIOLI, 2023a) (a cada 100ms)	Sim	$O(n)$	89,66%	0,2s

Fonte: Elaborado pelo autor.

junto de resultados, ao trabalhar com uma janela de tempo e reconhecimento *online* das anomalias, com complexidade linear, a mais baixa da literatura para este propósito, alcançando uma precisão de 97,67% com tempo de resposta de 20 segundos. Compondo um conjunto de resultados não encontrados nas demais propostas.

O mecanismo pré-carregado, configurável com 3 variáveis para ajuste em diferentes dimensões e características de redes de IoT, permite a implementação baseada em roteadores acessíveis na borda da rede através do sistema operacional de código aberto *OpenWrt* a uma complexidade computacional de O(n), com apenas 40MB de aplicações para operar o modelo no sistema operacional *OpenWrt*. No aspecto do aprendizado, o modelo tem como referência as médias (n), (n-1) e (n-2), sendo assim, seu aprendizado acontece de forma *online*, sem a necessidade da leitura do *dataset* inteiro, apenas as últimas médias móveis são usadas para diferenciar uma anomalia de uma situação normal. Características que credenciam o modelo para operar na borda da rede, reduzindo os danos de ataques DoS.

8 CONSIDERAÇÕES FINAIS

A crescente demanda e o ritmo acelerado de adoção de dispositivos IoT, acarreta inevitavelmente em mais dispositivos a serem explorados pelos ataques de negação de serviço. A maioria dos dispositivos IoT estão situados em redes locais, contudo o maior crescimento é observado nas redes 5G. O volume de tráfego gerado pelos ataques de negação de serviço também vem crescendo, ultrapassando atualmente a gradeza de *Terabits* por segundo, derrubando grandes provedores de serviços digitais.

A literatura atual possui um grande volume de pesquisas relacionando dispositivos IoT com ataques de negação de serviço, em sua maioria fazem o uso de aprendizado de máquina para diferenciar ataques do tráfego benigno. Apesar da alta precisão atingida nos modelos de aprendizado de máquina, ainda existe uma lacuna diante da aplicabilidade, visto que os modelos necessitam de treinamento, fazem uso intenso de recursos computacionais e a maioria são baseadas na leitura de *datasets* inteiros. O número de informações a serem analisadas, independentemente do modelo, também merece destaque. Extrair todas as informações possíveis do tráfego de rede, e delegá-las a um modelo de aprendizado de máquina, além de ser um processo custoso, não resulta diretamente em maior precisão. O modelo de ML proposto por Hamza *et al.* (2019), no mesmo *dataset* utilizado nesta pesquisa, levou quase 7 minutos para concluir o treinamento, atingindo uma precisão de 94,9%, já o tempo de resposta ficou em apenas 13,4ms. Lembrando que, para novos testes em diferentes *datasets* ou em casos de novos ataques, o sistema precisa ser treinado novamente. Em ambientes reais não é comum termos o tráfego de rede com os ataques marcados para que um modelo de ML possa ser treinado. Já o modelo do micro-IDS, obteve precisão de 97,67%, com tempo de resposta 20 segundos, sem necessidade de ser treinado, pelo fato de trabalhar com um mecanismo pré-carregado de comparação de médias móveis.

Diante deste contexto, o objetivo desta pesquisa foi propor um modelo para detecção de ataques de negação de serviço com viabilidade de operação na borda da rede. Tanto o objetivo geral, quanto os objetivos específicos, que nortearam as etapas da pesquisa, foram atingidos. A experiência de pesquisa e desenvolvimento geraram considerações a serem compartilhadas.

Com a realização de um mapeamento sistemático da literatura, descrito na íntegra no Capítulo 3, foi verificado 3 menções ao uso de médias móveis. Outra descoberta foi que alguns trabalhos analisaram e declararam a complexidade de sua proposta, todos acima da complexidade linear, ou seja, maior que $O(n)$. Alguns trabalhos também segmentavam a tarefa em partes, para aumentar a eficiência da solução. Sobre estas bases, o modelo foi criado considerando o aspecto mais comum dos ataques de negação de serviço volumétricos, o crescimento em um determinado tipo de pacote. Então

foi inicialmente executado um filtro por protocolo no tráfego geral da rede, posteriormente extraídas as estatísticas de número de pacotes em relação ao tempo, e por fim aplicado um modelo de verificação de crescimento baseado no histórico de médias móveis.

Os primeiros experimentos foram realizados com estatísticas extraídas por minuto. Apesar do sucesso na precisão, com tendência de 100% na maioria dos protocolos, bem como a baixa complexidade atingida O(n), nessa fase foram constatadas duas limitações. A primeira foi o tempo de resposta, como as novas médias eram formadas apenas após um minuto, e o modelo realiza a comparação das 3 últimas médias, obtida a primeira média, o sistema, precisa de mais duas, resultando em 2 minutos para o ataque ser reconhecido. A segunda limitação foi a referência de comparação entre as médias, que apesar de não ser um valor absoluto, eram de proporção absoluta na primeira versão do algoritmo. Definida em 10%, pela observação de crescimento em apenas um *dataset*, em uma unidade de tempo, apenas por minuto. Dessa forma, o modelo não poderia ser aplicado em diferentes unidades de tempo, ou até mesmo em redes IoT de diferentes proporções.

Constatadas estas limitações, foi desenvolvida uma nova versão onde a proporção de crescimento passou a se tornar uma variável, configurável. Assim como o número de amostras para a formação de média móvel. Essas alterações permitiram a expansão dos testes em diferentes unidades de tempo e *datasets*. Por outro lado, o modelo passou a exigir uma configuração de acordo com cada ambiente. A variável de taxa mínima ainda pode ser implementada como uma taxa flutuante. Como o modelo já possui o histórico de médias, pode ser aplicado um fator limitante quando as médias móveis apresentam uma redução abruta. Dessa forma esta variável será configurada de forma automática. Além das médias móveis há espaço para uso de outros métodos, como o percentil, por exemplo.

Um desafio que não foi superado pelo modelo proposto foi o de reconhecimento de ataques de baixa frequência. A fórmula do algoritmo é baseada no crescimento de pacotes de um determinado tipo. Os ataques de baixa frequência produzem uma variação muito baixa, que passa despercebido pelo mecanismo de reconhecimento. A possibilidade de ajustar as configurações para tornar o modelo mais sensível não são satisfatórias, neste caso, a sensibilidade fica demasiada alta, o que acaba por produzir produzem mais falsos positivos, que verdadeiros, não contribuindo para o aumento da precisão. Este tipo de ataque necessita de maior combinação de informações a serem comparadas.

Como trabalhos futuros sugere-se o desenvolvimento de um extrator de estatísticas de arquivos pcap para o propósito deste modelo. Este recurso permitirá a expansão dos testes, bem como a medição de leitura e escrita de cada tempo aplicado no protótipo. Apesar da precisão se manter em 89,66% para o reconhecimento do ataque em

até 200ms, é prudente avaliar o impacto de leitura e escrita no sistema operacional. Visto que a cada 100ms seria gerado um arquivo para extração de estatísticas. Por outro lado, reduzir a frequência da formação e comparação das médias móveis para 10 segundos, permite maior precisão e 100 vezes menos processos de leitura e escrita.

Outra oportunidade de pesquisa é o aprofundamento da etapa de análise após o reconhecimento da anomalia. Nesta pesquisa foi proposto uma busca seletiva de destino, para neutralizar os efeitos do ataque apenas enquanto ao seu alvo. Como através da análise de anomalia já foi realizado um recorte no momento do ataque, pode ser realizada uma análise mais profunda, combinando possivelmente as técnicas de análise de anomalia com análise de assinaturas. Ainda que custo computacional seja mais alto, será aplicado somente em um recorte do tráfego de rede, e não de forma contínua.

REFERÊNCIAS

ALEKSANDROVA, Elena B; LAVROVA, Daria S; YARMAK, AV. Benford's law in the detection of DoS attacks on industrial systems. **Automatic Control and Computer Sciences**, Springer, v. 53, p. 954–962, 2019.

ALJUHANI, Ahamed; ALHARBI, Talal; LIU, Hang. Xfirewall: A dynamic and additional mitigation against ddos storm. *In*: PROCEEDINGS of the International Conference on Compute and Data Analysis. [*S.l.*: *s.n.*], 2017. P. 1–5.

ALSHARIF, Nizar. Ensembling PCA-based Feature Selection with Random Tree Classifier for Intrusion Detection on IoT Network. *In*: IEEE. 2021 8th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI). [*S.l.*: *s.n.*], 2021. P. 317–321.

ANJUM, Afia; AGBAJE, Paul; HOUNSINOU, Sena; OLUFOWOBI, Habeeb. In-Vehicle Network Anomaly Detection Using Extreme Gradient Boosting Machine. *In*: IEEE. 2022 11th Mediterranean Conference on Embedded Computing (MECO). [*S.l.*: *s.n.*], 2022. P. 1–6.

ANTHI, Eirini; WILLIAMS, Lowri; SŁOWIŃSKA, Małgorzata; THEODORAKOPOULOS, George; BURNAP, Pete. A supervised intrusion detection system for smart home IoT devices. **IEEE Internet of Things Journal**, IEEE, v. 6, n. 5, p. 9042–9053, 2019.

ARUKONDA, Srinivas; SINHA, Samta. The innocent perpetrators: reflectors and reflection attacks. **Advanced Computer Science**, v. 4, p. 94–98, 2015.

ASHTON, Kevin *et al.* That 'internet of things' thing. **RFID journal**, Hauppauge, New York, v. 22, n. 7, p. 97–114, 2009.

BACE, Rebecca Gurley; MELL, Peter *et al.* Intrusion detection systems. US Department of Commerce, Technology Administration, National Institute of . . ., 2001.

BALABAN, Ioana *et al.* Denial-of-Service Attack. **International Journal of Information Security and Cybercrime (IJISC)**, Asociatia Romana pentru Asigurarea Securitatii Informatiei, v. 10, n. 1, p. 59–64, 2021.

BBC. **New Zealand stock exchange halted by cyber-attack - BBC News**. [*S.l.*: *s.n.*], 2020. https://www.bbc.com/news/53918580. (Acessado em 02/06/2023).

BENITTI, Fabiane Barreto Vavassori. Exploring the educational potential of robotics in schools: A systematic review. **Computers & Education**, Elsevier, v. 58, n. 3, p. 978–988, 2012.

BHALE, Pradeepkumar; BISWAS, Santosh; NANDI, Sukumar. LORD: LOw Rate DDoS Attack Detection and Mitigation Using Lightweight Distributed Packet Inspection Agent in IoT Ecosystem. *In*: IEEE. 2019 IEEE International Conference on Advanced Networks and Telecommunications Systems (ANTS). [*S.l.*: *s.n.*], 2019. P. 1–6.

BHUVANESWARI AMMA, NG; SELVAKUMAR, S. A statistical class center based triangle area vector method for detection of denial of service attacks. **Cluster Computing**, Springer, v. 24, p. 393–415, 2021.

BIGELOW, Stephen J. **What Is Edge Computing? Everything You Need to Know**. [*S.l.*: *s.n.*], 2021. https://www.techtarget.com/searchdatacenter/definition/edge-computing. (Acessado em 22/03/2023).

BINU, PK; MOHAN, Deepak; HARIDAS, EM Sreerag. An sdn-based prototype for dynamic detection and mitigation of dos attacks in iot. *In*: IEEE. 2021 Third International Conference on Inventive Research in Computing Applications (ICIRCA). [*S.l.*: *s.n.*], 2021. P. 5–10.

BONASERA, Will; CHOWDHURY, Md Minhaz; LATIF, Shadman. Denial of service: A growing underrated threat. *In*: IEEE. 2021 International Conference on Electrical, Computer, Communications and Mechatronics Engineering (ICECCME). [*S.l.*: *s.n.*], 2021. P. 1–6.

BOUYEDDOU, Benamar; HARROU, Fouzi; KADRI, Benamar; SUN, Ying. Detecting network cyber-attacks using an integrated statistical approach. **Cluster Computing**, Springer, v. 24, p. 1435–1453, 2021.

BOVENZI, Giampaolo; ACETO, Giuseppe; CIUONZO, Domenico; PERSICO, Valerio; PESCAPÉ, Antonio. A hierarchical hybrid intrusion detection approach in IoT scenarios. *In*: IEEE. GLOBECOM 2020-2020 IEEE Global Communications Conference. [*S.l.*: *s.n.*], 2020. P. 1–7.

CETINKAYA, Ahmet; ISHII, Hideaki; HAYAKAWA, Tomohisa. An overview on denial-of-service attacks in control systems: Attack models and security analyses. **Entropy**, MDPI, v. 21, n. 2, p. 210, 2019.

CHARI, Suresh; JUTLA, Charanjit S; RAO, Josyula R; ROHATGI, Pankaj. Towards sound approaches to counteract power-analysis attacks. *In*: SPRINGER. ADVANCES in Cryptology—CRYPTO'99: 19th Annual International Cryptology Conference Santa Barbara, California, USA, August 15–19, 1999 Proceedings 19. [*S.l.*: *s.n.*], 1999. P. 398–412.

COMBS, Gerald. **Wireshark · Go Deep**. [*S.l.*: *s.n.*], 2023. `https://www.wireshark.org/`. (Acessado em 05/06/2023).

CONRAD, E; MISENAR, S; FELDMAN, J. Chapter 2—Domain 2: Telecommunications and Network Security. **Eleventh Hour CISSP, 2nd ed.; Conrad, E., Misenar, S., Feldman, J., Eds**, p. 23–43, 2014.

CONTI, Mauro; DRAGONI, Nicola; LESYK, Viktor. A survey of man in the middle attacks. **IEEE communications surveys & tutorials**, IEEE, v. 18, n. 3, p. 2027–2051, 2016.

COOK, Andrew A; MISIRLI, Göksel; FAN, Zhong. Anomaly detection for IoT time-series data: A survey. **IEEE Internet of Things Journal**, IEEE, v. 7, n. 7, p. 6481–6494, 2019.

DANESHGADEH, Salva; KEMMERICH, Thomas; AHMED, Tarem; BAYKAL, Nazife. A hybrid approach to detect DDoS attacks using KOAD and the Mahalanobis distance. *In*: IEEE. 2018 IEEE 17th International Symposium on Network Computing and Applications (NCA). [*S.l.*: *s.n.*], 2018. P. 1–5.

DEAN, Walter. Computational complexity theory, 2015.

DI VITA, Luca. **Protocols Counter**. [*S.l.*: *s.n.*], 2019. `https://github.com/lucadivit/Protocols_Counter`. [Acessado em 31/08/2023].

DOSHI, Keval; MOZAFFARI, Mahsa; YILMAZ, Yasin. RAPID: Real-time anomaly-based preventive intrusion detection. *In*: PROCEEDINGS of the ACM Workshop on Wireless Security and Machine Learning. [*S.l.*: *s.n.*], 2019. P. 49–54.

DOSHI, Keval; YILMAZ, Yasin; ULUDAG, Suleyman. Timely detection and mitigation of stealthy DDoS attacks via IoT networks. **IEEE Transactions on Dependable and Secure Computing**, IEEE, v. 18, n. 5, p. 2164–2176, 2021.

DYBÅ, Tore; DINGSØYR, Torgeir. Strength of evidence in systematic reviews in software engineering. *In*: PROCEEDINGS of the Second ACM-IEEE international symposium on Empirical software engineering and measurement. [*S.l.*: *s.n.*], 2008. P. 178–187.

ESKANDARI, Mojtaba; JANJUA, Zaffar Haider; VECCHIO, Massimo; ANTONELLI, Fabio. Passban IDS: An intelligent anomaly-based intrusion detection system for IoT edge devices. **IEEE Internet of Things Journal**, IEEE, v. 7, n. 8, p. 6882–6897, 2020.

ESTRADA, Rebeca; VALERIANO, Irving; AIZAGA, Xavier; VARGAS, Lourdes; VERA, Nelson; ZAMBRANO, Diego. Wifi indoor positioning system based on openwrt. *In*: IEEE. IEEE EUROCON 2023-20th International Conference on Smart Technologies. [*S.l.*: *s.n.*], 2023. P. 728–733.

FAIRCLOTH, Jeremy. **Penetration tester's open source toolkit**. [*S.l.*]: Syngress, 2016.

AL-FUQAHA, Ala; GUIZANI, Mohsen; MOHAMMADI, Mehdi; ALEDHARI, Mohammed; AYYASH, Moussa. Internet of things: A survey on enabling technologies, protocols, and applications. **IEEE communications surveys & tutorials**, IEEE, v. 17, n. 4, p. 2347–2376, 2015.

GOMES, Lucas de Carvalho; ARAUJO, Marcos Seefelder de Assis; CAMPOS, Vinícius Silva. **DoS, DDoS e Botnets**. [*S.l.*: *s.n.*], 2015. `https://www.gta.ufrj.br/grad/15_1/dos/index.html`. (Acessado em 25/08/2023).

GONDIM, João JC; OLIVEIRA ALBUQUERQUE, Robson de; OROZCO, Ana Lucila Sandoval. Mirror saturation in amplified reflection Distributed Denial of Service: A case of study using SNMP, SSDP, NTP and DNS protocols. **Future Generation Computer Systems**, Elsevier, v. 108, p. 68–81, 2020.

GOWTHAM, M; PRAMOD, HB. Semantic query-featured ensemble learning model for SQL-injection attack detection in IoT-ecosystems. **IEEE Transactions on Reliability**, IEEE, v. 71, n. 2, p. 1057–1074, 2021.

HADDADI, Mohamed; BEGHDAD, Rachid. DoS-DDoS: taxonomies of attacks, countermeasures, and well-known defense mechanisms in cloud environment. **Edpacs**, Taylor & Francis, v. 57, n. 5, p. 1–26, 2018.

AL-HADHRAMI, Yahya; HUSSAIN, Farookh Khadeer. DDoS attacks in IoT networks: a comprehensive systematic literature review. **World Wide Web**, Springer, v. 24, n. 3, p. 971–1001, 2021.

HAMZA, Ayyoob; GHARAKHEILI, Hassan Habibi; BENSON, Theophilus A; SIVARAMAN, Vijay. Detecting volumetric attacks on Iot devices via sdn-based monitoring of mud activity. *In*: PROCEEDINGS of the 2019 ACM Symposium on SDN Research. [*S.l.*: *s.n.*], 2019. P. 36–48.

HE, Wenji; LIU, Yifeng; YAO, Haipeng; MAI, Tianle; ZHANG, Ni; YU, F Richard. Distributed variational bayes-based in-network security for the Internet of Things. **IEEE Internet of Things Journal**, IEEE, v. 8, n. 8, p. 6293–6304, 2020.

HSU, Fu-Hau; HWANG, Yan-Ling; TSAI, Cheng-Yu; CAI, Wei-Tai; LEE, Chia-Hao; CHANG, KaiWei. TRAP: A three-way handshake server for TCP connection establishment. **Applied Sciences**, MDPI, v. 6, n. 11, p. 358, 2016.

HUAWEIDOCS. **Defense Against UDP Flood Attacks - CX916, CX916L, and CX930 Switch Modules V100R001 Security Maintenance Guide 04 - Huawei**. [*S.l.*: *s.n.*], 2023. `https://support.huawei.com/enterprise/es/doc/EDOC1100015135/942e81fc/defense-against-udp-flood-attacks`. (Acessado em 25/08/2023).

JIA, Kun; LIU, Chaoge; LIU, Qixu; WANG, Junnan; LIU, Jiazhi; LIU, Feng. A lightweight DDoS detection scheme under SDN context. **Cybersecurity**, Springer, v. 5, n. 1, p. 27, 2022.

JIANG, Zhen Hang; FEI, Yunsi; KAELI, David. A novel side-channel timing attack on GPUs. *In*: PROCEEDINGS of the on Great Lakes Symposium on VLSI 2017. [*S.l.*: *s.n.*], 2017. P. 167–172.

KANG, Hyunjae; AHN, Dong Hyun; LEE, Gyung Min; YOO, Jeong Do; PARK, Kyung Ho; KIM, Huy Kang. **IoT network intrusion dataset**. [*S.l.*]: IEEE Dataport, 2019.

KIM, Junwon; SHIN, Jiho; SEO, Jung Taek. Detection and Blocking Method against DLL Injection Attack Using PEB-LDR of ICS EWS in Smart IoT Environments. **Journal of Internet Technology**, v. 23, n. 4, p. 875–888, 2022.

KOLIAS, Constantinos; KAMBOURAKIS, Georgios; STAVROU, Angelos; VOAS, Jeffrey. DDoS in the IoT: Mirai and other botnets. **Computer**, IEEE, v. 50, n. 7, p. 80–84, 2017.

LAUTERT, Henrique. **GitHub - hflautert/AnomalyDetection — github.com**. [*S.l.*: *s.n.*], 2023. `https://github.com/hflautert/AnomalyDetection`. [Acessado em 22/08/2023].

LAUTERT, Henrique Fell; MACEDO, DDJ de; PIOLI, Laércio. Micro IDS: On-Line Recognition of Denial-of-Service Attacks on IoT Networks. *In*: SPRINGER. ADVANCED Information Networking and Applications: Proceedings of the 37th International Conference on Advanced Information Networking and Applications (AINA-2023), Volume 1. [*S.l.*: *s.n.*], 2023. P. 446–459.

LAUTERT, Henrique Fell; MACEDO, DDJ de; PIOLI, Laércio. Recognition of Denial-of-Service Attacks in IoT Networks with Linear Complexity Model. *In*: ACM/IEEE. THE 16th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2023). [*S.l.*: *s.n.*], 2023.

LEE, Yong-joon; CHAE, Hwa-sung; LEE, Keun-wang. Countermeasures against large-scale reflection DDoS attacks using exploit IoT devices. **Automatika: časopis za automatiku, mjerenje, elektroniku, računarstvo i komunikacije**, KoREMA-Hrvatsko društvo za komunikacije, računarstvo, elektroniku, mjerenja . . ., v. 62, n. 1, p. 127–136, 2021.

LI, Fangyu; SHINDE, Aditya; SHI, Yang; YE, Jin; LI, Xiang-Yang; SONG, Wenzhan. System statistics learning-based IoT security: Feasibility and suitability. **IEEE Internet of Things Journal**, IEEE, v. 6, n. 4, p. 6396–6403, 2019.

LIAO, Hung-Jen; LIN, Chun-Hung Richard; LIN, Ying-Chih; TUNG, Kuang-Yuan. Intrusion detection system: A comprehensive review. **Journal of Network and Computer Applications**, Elsevier, v. 36, n. 1, p. 16–24, 2013.

LONG, Neil; THOMAS, Rob. Trends in denial of service attack technology. **CERT Coordination Center**, v. 648, p. 651, 2001.

LYNN, Theo; ENDO, Patricia Takako; RIBEIRO, Andrea Maria NC; BARBOSA, Gibson BN; ROSATI, Pierangelo. The internet of things: definitions, key concepts, and reference architectures. **The Cloud-To-Thing Continuum: Opportunities and Challenges in Cloud, Fog and Edge Computing**, Springer International Publishing, p. 1–22, 2020.

MACEDO, Jeronimo de; DYLLON, Douglas; FAGUNDES, Priscila Basto. An Analysis on the Use of Knowledge Organization Systems in the Process of Requirements Engineering, 2023.

MACHAKA, Pheeha; BAGULA, Antoine; NELWAMONDO, Fulufhelo. Using exponentially weighted moving average algorithm to defend against DDoS attacks. *In*: IEEE. 2016 Pattern recognition association of South Africa and robotics and mechatronics international conference (PRASA-RobMech). [*S.l.*: *s.n.*], 2016. P. 1–6.

MAZHAR, Noman; SALLEH, Rosli; ZEESHAN, Muhammad; HAMEED, M Muzaffar; KHAN, Nauman. R-IDPS: Real time SDN based IDPS system for IoT security. *In*: IEEE. 2021 IEEE 18th International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI (HONET). [*S.l.*: *s.n.*], 2021. P. 71–76.

MCKINSEY. **What is the Internet of Things?** [*S.l.*: *s.n.*], 2022. `https://www.mckinsey.com/featured-insights/mckinsey-explainers/what-is-the-internet-of-things#/`. (Acessado em 26/06/2023).

MERGENDAHL, Samuel; LI, Jun. Rapid: Robust and adaptive detection of distributed denial-of-service traffic from the internet of things. *In*: IEEE. 2020 IEEE Conference on Communications and Network Security (CNS). [*S.l.*: *s.n.*], 2020. P. 1–9.

MONNAPPA, KA. **Learning Malware Analysis: Explore the concepts, tools, and techniques to analyze and investigate Windows malware**. [*S.l.*]: Packt Publishing Ltd, 2018.

MUNIVARA PRASAD, K; RAMA MOHAN REDDY, A; VENUGOPAL RAO, K. BIFAD: Bio-inspired anomaly based HTTP-flood attack detection. **Wireless Personal Communications**, Springer, v. 97, p. 281–308, 2017.

NASCIMENTO, Márcio; ARAUJO, Jean; RIBEIRO, Admilson. Systematic review on mitigating and preventing DDoS attacks on IoT networks. *In*: IEEE. 2022 17th Iberian Conference on Information Systems and Technologies (CISTI). [*S.l.*: *s.n.*], 2022. P. 1–9.

NDATINYA, Vivens; XIAO, Zhifeng; MANEPALLI, Vasudeva Rao; MENG, Ke; XIAO, Yang. Network forensics analysis using Wireshark. **International Journal of Security and Networks**, Inderscience Publishers (IEL), v. 10, n. 2, p. 91–106, 2015.

NERI, Renan; LOMBA, Matheus; BULHÕES, Gabriel. **Protocolo MQTT - Redes 1**. [*S.l.*: *s.n.*], 2019. `https://www.gta.ufrj.br/ensino/eel878/redes1-2019-1/vf/mqtt/`. (Acessado em 19/06/2023).

NOKIA. **Nokia Threat Intelligence Report finds malicious IoT botnet activity has sharply increased | Nokia**. [*S.l.*: *s.n.*], 2023. `https://www.nokia.com/about-us/news/releases/2023/06/07/nokia-threat-intelligence-report-finds-malicious-iot-botnet-activity-has-sharply-increased/`. (Acessado em 29/11/2023).

NOMAN, Haitham Ameen; ABU-SHARKH, Osama MF. Code Injection Attacks in Wireless-based Internet of Things (IoT): A Comprehensive Review and Practical Implementations. **Sensors**, MDPI, v. 23, n. 13, p. 6067, 2023.

OJO, Mike O; GIORDANO, Stefano; PROCISSI, Gregorio; SEITANIDIS, Ilias N. A review of low-end, middle-end, and high-end iot devices. **IEEE Access**, IEEE, v. 6, p. 70528–70554, 2018.

PALMESE, Fabio; REDONDI, Alessandro EC; CESANA, Matteo. Feature-Sniffer: Enabling IoT Forensics in OpenWrt based Wi-Fi Access Points. *In*: IEEE. 2022 IEEE 8th World Forum on Internet of Things (WF-IoT). [*S.l.*: *s.n.*], 2022. P. 1–6.

PALSSON, Karl. Malaria Toolkit. **Retrieved May**, v. 31, p. 2020, 2019.

PARIKH, Shalin; DAVE, Dharmin; PATEL, Reema; DOSHI, Nishant. Security and privacy issues in cloud, fog and edge computing. **Procedia Computer Science**, Elsevier, v. 160, p. 734–739, 2019.

PAUDEL, Ramesh; MUNCY, Timothy; EBERLE, William. Detecting dos attack in smart home iot devices using a graph-based approach. *In*: IEEE. 2019 IEEE international conference on big data (big data). [*S.l.*: *s.n.*], 2019. P. 5249–5258.

PERLROTH, Nicole. **Hackers Used New Weapons to Disrupt Major Websites Across U.S. - The New York Times**. [*S.l.*: *s.n.*], out. 2016.

https://www.nytimes.com/2016/10/22/business/internet-problems-attack.html. (Acessado em 02/02/2023).

PIOLI, Laércio; DORNELES, Carina F; MACEDO, Douglas DJ de; DANTAS, Mario AR. An overview of data reduction solutions at the edge of IoT systems: a systematic mapping of the literature. **Computing**, Springer, v. 104, n. 8, p. 1867–1889, 2022.

PORTER, Jon. **Amazon says it mitigated the largest DDoS attack ever recorded — theverge.com**. [*S.l.*: *s.n.*], jun. 2020. https://www.theverge.com/2020/6/18/21295337/amazon-aws-biggest-ddos-attack-ever-2-3-tbps-shield-github-netscout-arbor. [Acessado em 02/02/2023].

PRADO, Marcelo Alves *et al.* Análise experimental da botnet IoT Mirai. Universidade Federal de Uberlândia, 2018.

PRAMATAROV, Martin. **Largest DDoS Attacks: It is getting out of hand! Q4 2022 UPDATED - Neterra.cloud Blog**. [*S.l.*: *s.n.*], nov. 2022. https://blog.neterra.cloud/en/largest-ddos-attacks-are-we-safe/. (Acessado em 16/03/2023).

RAFIQUE, Wajid; HE, Xin; LIU, Zifan; SUN, Yuhu; DOU, Wanchun. CFADefense: A security solution to detect and mitigate crossfire attacks in software-defined IoT-edge infrastructure. *In*: IEEE. 2019 IEEE 21st International Conference on High Performance Computing and Communications; IEEE 17th International Conference on Smart City; IEEE 5th International Conference on Data Science and Systems (HPCC/SmartCity/DSS). [*S.l.*: *s.n.*], 2019. P. 500–509.

RAJASHREE, S; SOMAN, KS; SHAH, Pritam Gajkumar. Security with IP address assignment and spoofing for smart IOT devices. *In*: IEEE. 2018 international conference on advances in computing, communications and informatics (ICACCI). [*S.l.*: *s.n.*], 2018. P. 1914–1918.

RAZUMOV, Pavel V; SAFARYAN, Olga A; SMIRNOV, Ivan A; PORKSHEYAN, Vitaliy M; BOLDYRIKHIN, Nickolay V; KOROCHENTSEV, Denis A; CHERCKESOVA, Larissa V; OSIKOV, Sergey A. Developing of algorithm of HTTP FLOOD DDoS protection. *In*: IEEE. 2020 3rd International Conference on Computer Applications & Information Security (ICCAIS). [*S.l.*: *s.n.*], 2020. P. 1–6.

RM, Swarna Priya; MADDIKUNTA, Praveen Kumar Reddy; PARIMALA, M; KOPPU, Srinivas; GADEKALLU, Thippa Reddy; CHOWDHARY, Chiranji Lal; ALAZAB, Mamoun. An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture. **Computer Communications**, Elsevier, v. 160, p. 139–149, 2020.

RODRIGUEZ, German E; TORRES, Jenny G; FLORES, Pamela; BENAVIDES, Diego E. Cross-site scripting (XSS) attacks and mitigation: A survey. **Computer Networks**, Elsevier, v. 166, p. 106960, 2020.

ROESCH, Martin. **DaemonLogger is a fast packet logger designed specifically for use in NSM environments**. [*S.l.*: *s.n.*], 2023. `https://www.talosintelligence.com/daemon`. [Acessado em 31/08/2023].

SAGHEZCHI, Firooz B; MANTAS, Georgios; VIOLAS, Manuel A; OLIVEIRA DUARTE, A Manuel de; RODRIGUEZ, Jonathan. Machine learning for DDoS attack detection in industry 4.0 CPPSs. **Electronics**, MDPI, v. 11, n. 4, p. 602, 2022.

SAHOO, Kshira Sagar; PUTHAL, Deepak. SDN-assisted DDoS defense framework for the internet of multimedia things. **ACM Transactions on Multimedia Computing, Communications, and Applications (TOMM)**, ACM New York, NY, USA, v. 16, 3s, p. 1–18, 2020.

SALAHUDDIN, Mohammad A; POURAHMADI, Vahid; ALAMEDDINE, Hyame Assem; BARI, Md Faizul; BOUTABA, Raouf. Chronos: Ddos attack detection using time-based autoencoder. **IEEE Transactions on Network and Service Management**, IEEE, v. 19, n. 1, p. 627–641, 2021.

SALEMI, Hossein; ROSTAMI, Habib; TALATIAN-AZAD, Saeed; KHOSRAVI, Mohammad Reza. LEAESN: Predicting DDoS attack in healthcare systems based on lyapunov exponent analysis and echo state neural networks. **Multimedia Tools and Applications**, Springer, p. 1–22, 2021.

SAMPAIO, Rosana Ferreira; MANCINI, Marisa Cotta. Estudos de revisão sistemática: um guia para síntese criteriosa da evidência científica. **Brazilian Journal of Physical Therapy**, SciELO Brasil, v. 11, p. 83–89, 2007.

SANTOS, Guto L; MONTEIRO, Kayo H de C; ENDO, Patricia Takako. Living at the Edge? Optimizing availability in IoT. **The Cloud-to-Thing Continuum: Opportunities and Challenges in Cloud, Fog and Edge Computing**, Springer International Publishing, p. 79–94, 2020.

SANTOYO-GONZÁLEZ, Alejandro; CERVELLÓ-PASTOR, Cristina; PEZAROS, Dimitrios P. High-performance, platform-independent DDoS detection for IoT ecosystems. *In*: IEEE. 2019 IEEE 44th Conference on Local Computer Networks (LCN). [*S.l.*: *s.n.*], 2019. P. 69–75.

SATYAJIT, Sinha. **Number of connected IoT devices growing 16% to 16.7 billion globally**. [*S.l.*: *s.n.*], mai. 2023. `https://iot-analytics.com/number-connected-iot-devices/`. (Acessado em 22/06/2023).

SHARAFALDIN, Iman; LASHKARI, Arash Habibi; HAKAK, Saqib; GHORBANI, Ali A. Developing realistic distributed denial of service (DDoS) attack dataset and taxonomy. *In*: IEEE. 2019 International Carnahan Conference on Security Technology (ICCST). [*S.l.*: *s.n.*], 2019. P. 1–8.

SHARMA, Deepak Kumar; DHANKHAR, Tarun; AGRAWAL, Gaurav; SINGH, Satish Kumar; GUPTA, Deepak; NEBHEN, Jamel; RAZZAK, Imran. Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks. **Ad Hoc Networks**, Elsevier, v. 121, p. 102603, 2021.

SOUSA, Breno Fabrício Lira Melo; ABDELOUAHAB, Zair; LOPES, Denivaldo Cicero Pavão; SOEIRO, Natalia Costa; RIBEIRO, Willian França. An intrusion detection system for denial of service attack detection in internet of things. *In*: PROCEEDINGS of the second international conference on internet of things, data and cloud computing. [*S.l.*: *s.n.*], 2017. P. 1–8.

SPADACCINO, Pietro; CUOMO, Francesca *et al.* Intrusion detection systems for iot: opportunities and challenges offered by edge computing. **Journal on Future and Evolving Technologies**, 2022.

SPREITZER, Raphael; MOONSAMY, Veelasha; KORAK, Thomas; MANGARD, Stefan. Systematic classification of side-channel attacks: A case study for mobile devices. **IEEE Communications Surveys & Tutorials**, IEEE, v. 20, n. 1, p. 465–488, 2017.

STALLMAN, R. IEEE Standard for Information Technology-Portable Operating System Interface (POSIX (R)). **IEEE Std**, p. 1003–1, 2008.

STANDAERT, François-Xavier. Introduction to side-channel attacks. **Secure integrated circuits and systems**, Springer, p. 27–42, 2010.

TANENBAUM, Andrew S; WETHERALL, D. The network layer. **Computer Networks**, Prentice Hall, p. 343–423, 2011.

TANG, Dan; ZHANG, Dongshuo; ZHAO, Huan; LIU, Dashun; YAN, Yudong; CHEN, Jingwen. Work in Progress: Network Attack Detection Towards Smart Factory. *In*: IEEE. 2021 IEEE 27th Real-Time and Embedded Technology and Applications Symposium (RTAS). [*S.l.*: *s.n.*], 2021. P. 485–488.

TANKARD, Colin. The security issues of the Internet of Things. **Computer Fraud & Security**, Elsevier, v. 2015, n. 9, p. 11–14, 2015.

TAVALLAEE, Mahbod. An adaptive hybrid intrusion detection system. **The University of New Brunswick**, 2011.

TAWALBEH, Lo'ai; MUHEIDAT, Fadi; TAWALBEH, Mais; QUWAIDER, Muhannad. IoT Privacy and security: Challenges and solutions. **Applied Sciences**, Mdpi, v. 10, n. 12, p. 4102, 2020.

VACCARI, Ivan; CHIOLA, Giovanni; AIELLO, Maurizio; MONGELLI, Maurizio; CAMBIASO, Enrico. MQTTset, a new dataset for machine learning techniques on MQTT. **Sensors**, MDPI, v. 20, n. 22, p. 6578, 2020.

WAN, Yinxin; XU, Kuai; WANG, Feng; XUE, Guoliang. Characterizing and mining traffic patterns of IoT devices in edge networks. **IEEE Transactions on Network Science and Engineering**, IEEE, v. 8, n. 1, p. 89–101, 2020.

WAZLAWICK, Raul Sidnei. **Metodologia de pesquisa para ciência da computação**. [*S.l.*]: Elsevier, 2009. v. 2.

WHITMORE, Andrew; AGARWAL, Anurag; DA XU, Li. The Internet of Things—A survey of topics and trends. **Information systems frontiers**, Springer, v. 17, p. 261–274, 2015.

WOOLF, Nicky. **DDoS attack that disrupted internet was largest of its kind in history, experts say | Hacking | The Guardian**. [*S.l.*: *s.n.*], 2016. `https://www.theguardian.com/technology/2016/oct/26/ddos-attack-dyn-mirai-botnet`. (Acessado em 25/08/2023).

XIAO, Yinhao; JIA, Yizhen; LIU, Chunchi; CHENG, Xiuzhen; YU, Jiguo; LV, Weifeng. Edge computing security: State of the art and challenges. **Proceedings of the IEEE**, IEEE, v. 107, n. 8, p. 1608–1631, 2019.

YAEGASHI, Ryo; TAKESHITA, Erina; NAKAYAMA, Yu. Two-Stage DDoS Mitigation with Variational Auto-Encoder and Cyclic Queuing. *In*: IEEE. ICC 2022-IEEE International Conference on Communications. [*S.l.*: *s.n.*], 2022. P. 5421–5426.

YANG, Y; MCLAUGHLIN, K; SEZER, S; YUAN, YB; HUANG, W. Stateful intrusion detection for IEC 60870-5-104 SCADA security. *In*: IEEE. 2014 IEEE PES General Meeting| Conference & Exposition. [*S.l.*: *s.n.*], 2014. P. 1–5.

YU, Wei; LIANG, Fan; HE, Xiaofei; HATCHER, William Grant; LU, Chao; LIN, Jie; YANG, Xinyu. A survey on the edge computing for the Internet of Things. **IEEE access**, IEEE, v. 6, p. 6900–6919, 2017.

ZAIB, Muhammad Hassan; BASHIR, Faisal; QURESHI, Kashif Naseer; KAUSAR, Sumaira; RIZWAN, Muhammad; JEON, Gwanggil. Deep learning based cyber bullying early detection using distributed denial of service flow. **Multimedia Systems**, Springer, p. 1–20, 2021.

ZHAO, Ziming; LIU, Fang; CAI, Zhiping; XIAO, Nong. Edge computing: platforms, applications and challenges. **J. Comput. Res. Dev**, v. 55, n. 2, p. 327–337, 2018.

APÊNDICE A – CONFIGURAÇÃO DAS BASES DE PESQUISA

A.1 ACM

Na base ACM foi aplicado o filtro Content Type->Research Article, então realizado a exportação no formato BibTex, os arquivos baixados foram agregados na ferramenta Jabref, então gerado o arquivo unificado no formato ris, para importação no EndNote.

String de busca

[[All: "dos"] OR [All: "denial of service"] OR [All: "volumetric"]] AND [All: "attacks"] AND [[All: "iot"] OR [All: "internet of things"]] AND [All: "anomaly"]

A.2 IEEEXPLORE

Na base IEEEXplore não foram aplicados filtros extras. A exportação foi realizada diretamente no formato ris, incluindo citação e resumo (Citation & Abstract).

String de busca

(“DOS” OR “Denial of service” OR “volumetric”) AND “attacks” AND ("IOT"OR "Internet of Things") AND "Anomaly"

A.3 SCOPUS

Na base Scopus foram aplicados os filtros: Subject Area -> Computer Science e Document type -> Article. Na exportação foi usado o formato ris e selecionados todos os campos de informações sobre citação(Citation information) e Resumo e palavras chaves (Abstract and keywords).

String de busca

TITLE-ABS-KEY (("DOS"OR "Denial of service"OR "volumetric") AND "attacks"AND ("IOT"OR "Internet of Things") AND "Anomaly")

A.4 SPRINGER

Na base Springer foi desativada a inclusão de conteúdos sem acesso completo (Include Preview-Only content). Então foram aplicados os filtros: Content Type->Article e Discipline->Computer Science. Posteriormente os resultados foram exportados no formato csv. Para conversão no formato ris foi utilizada a ferramenta Zotero, aplicada a

ferramenta de busca por identificador DOI, então salvo em formato ris para importação no Endnote.

String de busca

’(“DOS” OR “Denial of service” OR “volumetric”) AND “attacks” AND ("IOT"OR "Internet of Things") AND "Anomaly"’

APÊNDICE B – PROCESSO DE SELEÇÃO

CRITÉRIOS DE INCLUSÃO

- Artigos com acesso completo.
- Artigos em inglês.
- Artigos que apresentem propostas para reconhecimento de ataques de negação de serviço baseados na análise de anomalias.
- Artigos que apresentem dados quantitativos sobre o experimento.
- Artigos que considerem os limites computacionais da borda da rede e apresentem discussão sobre os desafios de implementação da solução proposta em ambientes reais.

CRITÉRIOS DE EXCLUSÃO

- Artigos que não tenham afinidade com tema, casualmente obtidos pela combinação de palavras.
- Artigos que não apresentem qualquer dado quantitativo sobre o experimento.
- Artigos que não considerem os limites computacionais da borda da rede.
- Estudos secundários.
- Estudos terciários.

PROCEDIMENTO DE SELEÇÃO

O procedimento foi inspirado nos estudos de (DYBÅ; DINGSØYR, 2008), composto pelos seguintes passos:

1. Inserção de palavras chaves nas bases escolhidas.
2. Aplicação de filtro para conteúdo completo.
3. Aplicação de filtro área - Computer Science.
4. Aplicação de filtro somente artigos.
5. Centralização em softwares Endnote e Zotero.
6. Exclusão de duplicados.
7. Exclusão por título e resumo.
8. Leitura crítica dos artigos selecionados.
9. Exclusão após leitura e análise.

Tabela 10 – Trabalhos Selecionados.

Índice	Título	Referência
S1	A Hierarchical Hybrid Intrusion Detection Approach in IoT Scenarios	(BOVENZI *et al.*, 2020)
S2	A Hybrid Approach to Detect DDoS Attacks Using KOAD and the Mahalanobis Distance	(DANESHGADEH *et al.*, 2018)
S3	A lightweight DDoS detection scheme under SDN context	(JIA *et al.*, 2022)
S4	A statistical class center based triangle area vector method for detection of denial of service attacks	(BHUVANESWARI AMMA; SELVAKUMAR, 2021)
S5	A Supervised Intrusion Detection System for Smart Home IoT Devices	(ANTHI *et al.*, 2019)
S6	An effective feature engineering for DNN using hybrid PCA-GWO for intrusion detection in IoMT architecture	(RM *et al.*, 2020)
S7	An Intrusion Detection System for Denial of Service Attack Detection in Internet of Things	(SOUSA *et al.*, 2017)
S8	An SDN-Based Prototype for Dynamic Detection and Mitigation of DoS Attacks in IoT	(BINU; MOHAN; HARIDAS, 2021)
S9	Anomaly detection framework to prevent DDoS attack in fog empowered IoT networks	(SHARMA *et al.*, 2021)
S10	Benford's Law in the Detection of DoS Attacks on Industrial Systems	(ALEKSANDROVA; LAVROVA; YARMAK, 2019)
S11	Chronos: DDoS Attack Detection Using Time-Based Autoencoder	(SALAHUDDIN *et al.*, 2021)
S12	Deep learning based cyber bullying early detection using distributed denial of service flow	(ZAIB *et al.*, 2021)
S13	Detecting DoS Attack in Smart Home IoT Devices Using a Graph-Based Approach	(PAUDEL; MUNCY; EBERLE, 2019)
S14	Detecting network cyber-attacks using an integrated statistical approach	(BOUYEDDOU *et al.*, 2021)
S15	Detecting Volumetric Attacks on LoT Devices via SDN-Based Monitoring of MUD Activity	(HAMZA *et al.*, 2019)
S16	Distributed Variational Bayes-Based In-Network Security for the Internet of Things	(HE *et al.*, 2020)
S17	Ensembling PCA-based Feature Selection with Random Tree Classifier for Intrusion Detection on IoT Network	(ALSHARIF, 2021)
S18	High-performance, platform-independent DDoS detection for IoT ecosystems	(SANTOYO-GONZÁLEZ; CERVELLÓ-PASTOR; PEZAROS, 2019)
S19	In-Vehicle Network Anomaly Detection Using Extreme Gradient Boosting Machine	(ANJUM *et al.*, 2022)
S20	LEAESN: Predicting DDoS attack in healthcare systems based on Lyapunov Exponent Analysis and Echo State Neural Networks	(SALEMI *et al.*, 2021)
S21	LORD: LOw Rate DDoS Attack Detection and Mitigation Using Lightweight Distributed Packet Inspection Agent in IoT Ecosystem	(BHALE; BISWAS; NANDI, 2019)
S22	Machine Learning for DDoS Attack Detection in Industry 4.0 CPPSs	(SAGHEZCHI *et al.*, 2022)
S23	Passban IDS: An Intelligent Anomaly-Based Intrusion Detection System for IoT Edge Devices	(ESKANDARI *et al.*, 2020)
S24	R-IDPS: Real Time SDN-Based IDPS System for IoT Security	(MAZHAR *et al.*, 2021)
S25	RAPID: Real-Time Anomaly-Based Preventive Intrusion Detection	(DOSHI; MOZAFFARI; YILMAZ, 2019)
S26	Rapid: Robust and Adaptive Detection of Distributed Denial-of-Service Traffic from the Internet of Things	(MERGENDAHL; LI, J., 2020)
S27	SDN-Assisted DDoS Defense Framework for the Internet of Multimedia Things	(SAHOO; PUTHAL, 2020)
S28	System Statistics Learning-Based IoT Security: Feasibility and Suitability	(LI, F. *et al.*, 2019)
S29	Timely Detection and Mitigation of Stealthy DDoS Attacks Via IoT Networks	(DOSHI; YILMAZ; ULUDAG, 2021)
S30	Two-Stage DDoS Mitigation with Variational Auto-Encoder and Cyclic Queuing	(YAEGASHI; TAKESHITA; NAKAYAMA, 2022)
S31	Using exponentially weighted moving average algorithm to defend against DDoS attacks	(MACHAKA; BAGULA; NELWAMONDO, 2016)
S32	Work in Progress: Network Attack Detection Towards Smart Factory	(TANG *et al.*, 2021)
S33	XFirewall: A Dynamic and Additional Mitigation Against DDoS Storm	(ALJUHANI; ALHARBI; LIU, 2017)

Fonte: Elaborado pelo autor.

ANEXO A – PUBLICAÇÕES

Através desta pesquisa foi escrito um artigo como o título *Micro IDS: On-Line Recognition of Denial-of-Service Attacks on IoT Networks* (LAUTERT, H. F.; MACEDO, D. de; PIOLI, 2023a). O artigo foi aceito e apresentado no evento *International Conference on Advanced Information Networking and Applications (AINA-2023)*, que está atualmente classificado no extrato A3.

O modelo Micro-IDS com as variáveis de configuração foi transformado em artigo e aceito para publicação como artigo completo na conferência *The 16th IEEE/ACM International Conference on Utility and Cloud Computing (UCC 2023)*, que está atualmente classificada no extrato A3.