Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Desenvolvimento de um Mecanismo Integrado para a Detecção e Mitigação de Ataques DDoS em Redes Definidas por Software

Ranyelson Neres Carvalho

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Orientador
Prof. Dr. Jacir Luiz Bordim

Brasília
2025

Ficha catalográfica elaborada automaticamente,
com os dados fornecidos pelo(a) autor(a)

Neres Carvalho, Ranyelson
NR213nd Desenvolvimento de um Mecanismo Integrado para a Detecção e Mitigação de Ataques DDoS em Redes Definidas por Software / Ranyelson Neres Carvalho; orientador Jacir Luiz Bordim. Brasília, 2025.
104 p.

Tese(Doutorado em Informática) Universidade de Brasília, 2025.

1. Redes Definidas por Software. 2. Ataques de Negação de Serviço Distribuídos (DDoS). 3. Detecção e Mitigação de Ataques. 4. Mecanismo de Segurança Baseado em Confiança . 5. Plano de Dados Programável. I. Luiz Bordim, Jacir, orient. II. Título.

Universidade de Brasília
Instituto de Ciências Exatas
Departamento de Ciência da Computação

Desenvolvimento de um Mecanismo Integrado para a Detecção e Mitigação de Ataques DDoS em Redes Definidas por Software

Ranyelson Neres Carvalho

Tese apresentada como requisito parcial para
conclusão do Doutorado em Informática

Prof. Dr. Jacir Luiz Bordim (Orientador)
CIC/UnB

Prof. Dr. Antônio Jorge Gomes Abelém
UFPA

Prof. Dr. Jó Ueyama
ICMC/USP

Prof. Dr. João José Costa Gondim
ENE/UnB

Prof. Dr. Rodrigo Bonifácio de Almeida
Coordenador do Programa de Pós-graduação em Informática

Brasília, 07 de Maio de 2025

Dedicatória

Dedico este trabalho à minha família, aos meus professores e aos meus amigos, que me apoiaram e incentivaram a alcançar meus objetivos.

Agradecimentos

Primeiramente, expresso minha gratidão a Deus por ter me ajudado nessa longa caminhada e à minha família por todo apoio.

De maneira especial, agradeço à minha noiva, Míriam, pela paciência e compreensão nos momentos difíceis, que não foram poucos, e pelo apoio constante, sempre ao meu lado, me ajudando a superar os desafios e a manter o foco nos nossos objetivos. Sua presença foi fundamental para que eu pudesse seguir em frente, mesmo nas situações mais complicadas. Também sou grato aos meus fiéis companheiros de quatro patas, †Naninha, Naninho e Denguinho, por tornarem os dias mais leves com suas travessuras e carinho incondicional.

Expresso também minha gratidão ao Programa de Pós-Graduação em Informática da Universidade de Brasília, ao Departamento de Ciência da Computação por todo suporte e esclarecimentos, e aos professores da banca, cujas contribuições foram fundamentais para a melhoria deste trabalho.

Não posso deixar de reconhecer a importância dos meus amigos de laboratório e de trabalho, Marcos, Luís, Leomar, Walter, Cleypson, Lucas, Renato, Marcella, Kelly, Danielle, Charleson, Victor, Fernando, Marcelo, Hélio e Malvezzi, que tornaram essa jornada mais leve, com momentos de descontração e valiosos compartilhamentos de conhecimento.

Por fim, agradeço aos meus professores da Universidade de Brasília, em especial ao professor e orientador Jacir Luiz Bordim, pelas orientações, conselhos, confiança e paciência, que foram fundamentais para a obtenção dos resultados apresentados neste trabalho.

Resumo

As redes definidas por *software* (do inglês, *Software Defined Networking* - SDN) são arquiteturas de redes que enfatizam a separação entre o plano de controle e o plano de dados, proporcionando uma série de benefícios, como o gerenciamento centralizado, a flexibilidade e a programabilidade da infraestrutura de rede. Apesar dos benefícios oferecidos pela arquitetura SDN, do ponto de vista da segurança, essa arquitetura introduziu novas vulnerabilidades devido à comunicação necessária entre esses planos, em virtude da separação deles. Os ataques distribuídos de negação de serviço (do inglês, *Distributed Denial of Service* - DDoS), especialmente os do tipo volumétrico, representam um desafio significativo para esse tipo de rede, já que, nesta arquitetura, o controlador atua como um ponto central de controle e decisão, sendo responsável por gerenciar o tráfego e a configuração da rede. Assim, ele se torna um alvo estratégico para os ataques DDoS volumétricos, já que sua paralisação, em função do volume massivo de tráfego malicioso proveniente desses ataques, resultará na exaustão da capacidade de processamento, levando à sua indisponibilidade e comprometendo a operação de toda a rede. As soluções presentes no estado da arte apresentaram diversas estratégias para reduzir os impactos dos ataques DDoS volumétricos. Porém, ainda permanecem algumas lacunas, como a centralização das ações de detecção e mitigação no plano de controle, causando atrasos no processo para confirmar qualquer mudança no comportamento do tráfego associada a esses ataques e o aumento do volume de mensagens de controle encaminhadas ao controlador para realizar a identificação e a contenção desses ataques a rede. Além disso, muitas das estratégias desenvolvidas pelas soluções acabam penalizando uma parte significativa do tráfego legítimo, provocando bloqueios de forma indiscriminada. Este trabalho propõe um mecanismo de detecção e mitigação denominado DataControl-ML, que visa suprir essas lacunas. O mecanismo está organizado em dois procedimentos: (*i*) detecção e mitigação no plano de dados; e (*ii*) compartilhamento de informações globais por meio do plano de controle. O primeiro procedimento concentra-se em uma abordagem de detecção no plano de dados, responsável por classificar o fluxo de tráfego malicioso por meio de um modelo de classificação com base no algoritmo de aprendizagem de máquina *Random Forest*, que extrai as características dos pacotes de rede e as utiliza para identificar padrões associados

a comportamentos maliciosos. Essa abordagem é seguida por uma estratégia de mitigação que gerencia diferentes listas nos dispositivos de encaminhamento, utilizando níveis de confiabilidade, para priorizar os fluxos confiáveis (legítimos) e bloquear aqueles com baixo valor de confiança (maliciosos). O segundo procedimento visa o compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas com base no estabelecimento de uma confiança global calculada a partir de um sistema *fuzzy* no controlador, para o envio aos dispositivos de encaminhamento, promovendo uma colaboração eficiente entre o plano de dados e de controle, e maior rapidez nos processos de detecção e mitigação em diferentes pontos da rede. Os resultados experimentais obtidos mostraram que o DataControl-ML reduz cerca de 52,18% o tempo necessário para a confirmação (detecção) de uma fonte maliciosa, responsável por gerar os ataques DDoS volumétricos, quando comparado a outro mecanismo do estado da arte (o Bungee-ML). Além disso, o mecanismo proposto preserva os recursos (CPU e memória) do controlador e *switch*, diminuindo o volume de mensagens de controle enviadas ao controlador, o tempo de convergência para que os dispositivos de encaminhamento tenham informações sincronizadas e alcança uma eficácia de detecção e mitigação superior a 98%, reduzindo os impactos causados por esses ataques.

Palavras-chave: Redes Definidas por Software (SDN), Ataques de Negação de Serviço Distribuídos (DDoS), Detecção e Mitigação de Ataques, Mecanismo de Segurança Baseado em Confiança, Plano de Dados Programável.

Abstract

Development of an integrated mechanism for detecting and mitigating DDoS attacks in software-defined networks

Software-defined networking (SDN) is a network architecture that emphasizes the separation of the control plane from the data plane, providing a number of benefits such as centralized management, flexibility, and programmability of the network infrastructure. Despite the benefits offered by SDN from a security perspective, this architecture has introduced new vulnerabilities due to the communication required between these planes due to their separation. Distributed denial-of-service (DDoS) attacks, especially volumetric ones, represent a significant challenge for this type of network, since in this architecture the controller acts as a central point of control and decision, being responsible for managing network traffic and configuration. Thus, it becomes a strategic target for volumetric DDoS attacks, as the massive influx of malicious traffic can overwhelm its processing capacity, causing service disruption and compromising the operation of the entire network. State-of-the-art solutions have presented several strategies to reduce the impacts of volumetric DDoS attacks. However, some gaps still remain, such as the centralization of detection and mitigation actions in the control plane, causing delays in the process of confirming any change in traffic behavior associated with these attacks and the increase in the volume of control messages forwarded to the controller to identify and contain these network attacks. In addition, many of the mitigation strategies employed by these solutions inadvertently penalize legitimate traffic, resulting in indiscriminate blocking. This work proposes a detection and mitigation mechanism called DataControl-ML, which aims to fill these gaps. The proposed mechanism is organized into two procedures: (*i*) detection and mitigation in the data plane; and (*ii*) sharing of global information through the control plane. The first procedure adopts a detection approach in the data plane, where a machine learning-based classification model—specifically, the Random Forest algorithm—analyzes network packet features to identify patterns indicative of malicious

traffic. This approach is followed by a mitigation strategy that manages different lists in the forwarding devices, using trust levels, to prioritize the trustworthy (legitimate) flows and block those with low trust value (malicious). The second procedure focuses on sharing global information through the control plane. It defines control actions—such as blocking, allowing, or prioritizing clients—based on a global trust score calculated by a fuzzy logic system in the controller. These actions are sent to forwarding devices, enabling efficient collaboration between the data and control planes and speeding up detection and mitigation across the network. The experimental results obtained show that DataControl-ML reduces approximately 52,18% the time required for the confirmation (detection) of a malicious source, responsible for generating volumetric DDoS attacks, when compared to another state-of-the-art mechanism (Bungee-ML). In addition, the proposed mechanism helps preserve CPU and memory resources on both the controller and the switches. It reduces the number of control messages sent to the controller, shortens the convergence time for forwarding devices to synchronize information, and achieves over 98% efficiency in detecting and mitigating attacks—significantly reducing their impact.

Keywords: Software-Defined Networking (SDN), Distributed Denial-of-Service (DDoS) Attacks, Attack Detection and Mitigation, Trust-based Security Mechanism, Programmable Data Plane.

Sumário

Lista de Figuras

Lista de Tabelas

Lista de Abreviaturas e Siglas

ACK *Acknowledgment.*

ACM *Association for Computing Machinery.*

AINA *Advanced Information Networking and Applications.*

APDCM *Advances in Parallel and Distributed Computational Models.*

API *Application Programming Interface.*

ASIC *Application Specific Integrated Circuits.*

ASTESJ *Advances in Science, Technology and Engineering Systems Journal.*

CANDARW *Computing and Networking Workshop.*

CCPE *Concurrency and Computation: Practice and Experience.*

DDoS *Distributed Denial of Service.*

DNS *Domain Name System.*

HIDS *Host Intrusion Detection System.*

HTTP *HyperText Transfer Protocol.*

IAT *Inter-Arrival Time.*

ICMP *Internet Control Message Protocol.*

IDS *Intrusion Detection System.*

IEEE *Institute of Electrical and Electronics Engineers.*

IJNM *International Journal of Network Management.*

IP *Internet Protocol.*

IPS *Intrusion Prevention System.*

KNN *K-Nearest Neighbors.*

MAC *Media Access Control.*

MIB *Management Information Base.*

MTU *Maximum Transmission Unit.*

NIDS *Network Intrusion Detection System.*

ONOS *Open Network Operating System.*

OSI *Open Systems Interconnection.*

P4 *Programming Protocol Independent Packet Processors.*

PCAP *Packet Capture.*

PISA *Protocol-Idenpendent Switch Architecture.*

PRTG *Paessler Router Traffic Grapher.*

QoS *Quality of Service.*

REST *Representational State Transfer.*

RF *Random Forest.*

RPC *Remote Procedure Call.*

RTT *Round Trip Time.*

SCM *SDN Controller Manager.*

SDN *Software Defined Networking.*

SNMP *Simple Network Management Protocol.*

SVM *Support Vector Machine.*

SYN *Synchronize.*

TCP *Transmission Control Protocol.*

UDP *User Datagram Protocol.*

Capítulo 1

Introdução

As redes definidas por *software* (do inglês, *Software Defined Networking* - SDN) surgiram em resposta aos desafios enfrentados pelas redes tradicionais, como arquitetura complexa e rígida, que dificultavam a adição, remoção ou modificação dos recursos da rede de forma eficiente [2]. Os dispositivos presentes em uma rede tradicional, como *switches* e roteadores, são projetados com funções de controle e encaminhamento de maneira unificada, ou seja, em um único plano de trabalho, impondo dificuldades no gerenciamento desta rede devido à diversidade de dispositivos na infraestrutura, à variedade de protocolos a serem configurados em cada dispositivo físico e às especificidades de cada fabricante [3]. Além disso, a escalabilidade ainda pode representar um desafio, pois, embora os fabricantes geralmente disponibilizem MIB's (*Management Information Base*) que permitem a interoperabilidade ao estabelecer padrões para a comunicação entre diferentes dispositivos de rede, cada um possui configurações e características específicas [4, 5]. Dessa forma, a adição de novos dispositivos ou serviços pode demandar configurações manuais e complexas, especialmente quando não há um gerenciamento centralizado.

Diante dos desafios das redes tradicionais, as redes SDN buscaram separar as funções de controle e encaminhamento em diferentes planos de trabalho, chamados de controle e dados, para tornar a rede mais flexível, gerenciável e centralizada [6]. Assim, o plano de controle é formado pelos controladores, implementados por meio de *softwares*, responsáveis por criar e coordenar de maneira centralizada os recursos e as políticas da rede, como o encaminhamento e roteamento dos pacotes de rede. Enquanto, o plano de dados é formado pelos dispositivos de rede, como *switches* e roteadores, responsáveis pelas ações de encaminhamento e roteamento de dados conforme a política definida pelo controlador [3].

Além do plano de controle e de dados, arquitetura SDN também define um plano de aplicação que executa aplicações de rede como balanceador de carga e *firewall*, para auxiliar o controlador no gerenciamento da rede. Essa organização permite que a rede seja programada e gerenciada de forma mais dinâmica, flexível e escalável, possibilitando

a rápida implementação e modificação de serviços, como o desenvolvimento de funções de segurança e o gerenciamento de fluxos, facilitando a detecção e o diagnóstico de problemas [7].

Apesar dos benefícios oferecidos pela arquitetura SDN, a segurança ainda é motivo de preocupação, especialmente devido à separação entre o plano de controle e de dados, o que amplia a superfície de ataques [8]. Entre as ameaças mais relevantes nesse contexto estão os ataques distribuídos de negação de serviço (do inglês, *Distributed Denial of Service* - DDoS), em especial os do tipo volumétrico, que buscam sobrecarregar os recursos de rede por meio de um tráfego malicioso massivo em um espaço curto de tempo, explorando vulnerabilidades em aplicativos, serviços ou protocolos executados na rede alvo [9]. Na arquitetura SDN, o controlador torna-se um alvo estratégico, pois sua indisponibilidade pode comprometer a operação de toda a rede. Com isso, os ataques DDoS volumétricos buscam sobrecarregar o controlador para limitar a sua capacidade de lidar com o tráfego legítimo, gerando interrupções generalizadas nos serviços disponibilizadas na rede.

1.1 Motivação

Diante do cenário no qual os ataques DDoS volumétricos exploram vulnerabilidades inerentes à arquitetura SDN, especialmente em virtude da separação entre os planos (controle e dados), para sobrecarregar o controlador e comprometer a disponibilidade da rede, torna-se essencial investigar, propor e aprimorar soluções de segurança, de forma a reduzir os impactos causados por esses ataques e, ao mesmo tempo, preservar a continuidade do tráfego legítimo.

Nesse contexto, muitas soluções de segurança presentes no estado da arte têm concentrado seus esforços na elaboração de abordagens que atuam exclusivamente na camada de controle, devido à possibilidade de criar políticas de controle de tráfego de forma centralizada, permitindo um maior domínio de toda a infraestrutura de rede [10]. Todavia, esse tipo de abordagem torna o controlador ponto único de falha, e as interações frequentes entre o plano de dados e o plano de controle provocam atrasos e o aumento do volume de mensagens de controle encaminhadas ao controlador para reduzir os impactos causados pelos ataques DDoS volumétricos, aumentando o seu consumo de recursos [11]. Por exemplo, tempos mais longos para detectar e confirmar qualquer mudança no comportamento do tráfego, o que retarda o acionamento dos mecanismos de deteçcão e mitigação. Além disso, há um aumento no consumo de recursos por parte do controlador, que precisa processar continuamente um elevado volume de mensagens de controle para a tomada de decisões.

Para reduzir os atrasos e o volume de mensagens de controle direcionadas ao controlador, as soluções desenvolvidas por Shin *et al.* [12], Moreno *et al.* [13] e Lapoli *et al.* [14], propuseram o desenvolvimento de mecanismos que atuam diretamente no plano de dados, diminuindo o volume de mensagens de controle e proporcionando uma ação de detecção mais rápida por parte das soluções [15]. Porém, elas ficaram restritas devido às dificuldades de desenvolver métodos de detecção mais sofisticados e, ao mesmo tempo, garantir o processamento de pacotes em alta velocidade no plano de dados [15]. Por causa disso, soluções como Oracle [16], Cooperative-DDoS [17] e Bungee-ML [18] adotaram uma abordagem híbrida, no qual realizam parte das ações de detecção no plano de dados e a outra parte no plano de controle. Essas abordagens acabam aumentando o volume de mensagens de controle e o tempo de confirmação de um ataque, já que o dispositivo central será responsável por confirmar cada suspeita produzida no plano de dados [10].

No que tange às abordagens desenvolvidas pelas soluções para reduzir os impactos dos ataques DDoS volumétricos, elas podem ser categorizadas, segundo Imran *et al.* [19] em três tipos: (*i*) bloqueio, que busca interromper por completo o tráfego das possíveis fontes do ataque; (*ii*) controle, que atrasa de forma proposital o tráfego suspeito; e (*iii*) gerenciamento de recursos, que faz uso de recursos adicionais presentes na infraestrutura de rede. Grande parte das soluções opta pela ação de bloqueio, pois ela proporciona um alívio instantâneo aos recursos de rede e possui uma complexidade mais baixa para o seu desenvolvimento, como as soluções desenvolvidas por Tuan *et al.* [20], Kumar *et al.* [21] e Salem *et al.* [22]. Embora, esta ação vise restringir todo o tráfego suspeito, ela acaba penalizando os clientes legítimos vinculados a uma determinada porta do *switch* ou endereço IP (*Internet Protocol*) que está gerando esse tipo de tráfego, provocando um bloqueio indiscriminado.

Para evitar o bloqueio indiscriminado do tráfego, os autores em [22] e [23] buscaram adotar novos métodos, como a atribuição de confiabilidade e créditos aos clientes da rede com base em seu comportamento, para estabelecer níveis de confiança entre eles e, assim, realizar um bloqueio de maneira seletiva, restringindo apenas os clientes que estão abaixo do nível de confiabilidade. Porém, as soluções desenvolvidas atuam apenas no plano de controle, o que resulta em atrasos no processo de confirmação de qualquer atividade suspeita associada aos ataques DDoS volumétricos e ao aumento do volume de mensagens de controle para gerenciar os níveis de confiabilidade dos clientes e as demais ações de controle do tráfego de rede [15, 10].

Diante da análise das soluções presentes no estado da arte, observou-se uma variedade de abordagens para tornar essa arquitetura de rede mais segura e, assim, aproveitar os benefícios oferecidos por ela, como flexibilidade, escalabilidade e o gerenciamento de aplicações e serviços [19]. No entanto, ainda existem lacunas que podem deixar essa ar-

quitetura vulnerável, como a centralização das ações de detecção e mitigação no plano de controle, provocando atrasos adicionais para confirmar qualquer mudança no comportamento do tráfego associada aos ataques DDoS volumétricos e o aumento do volume de mensagens de controle encaminhadas ao controlador para realizar a identificação e a contenção desses ataques a rede [15]. Além disso, muitas das estratégias desenvolvidas pelas soluções para reduzir os impactos desses ataques, que, embora visem restringir o tráfego malicioso, acabam penalizando uma parte significativa do tráfego legítimo, provocando bloqueios de forma indiscriminada [20, 21, 24, 22, 25].

Considerando as lacunas observadas, identifica-se a necessidade de uma abordagem mais eficiente, que não apenas reduza os atrasos na resposta aos ataques DDoS volumétricos e diminua a dependência do controlador, mas que também avalie continuamente o comportamento dos clientes e atribua níveis de confiança com base nesse comportamento observado e utilize essas informações para, priorizar o tráfego legítimo em detrimento do tráfego malicioso, de forma seletiva e inteligente, restringindo apenas o tráfego identificado como malicioso.

1.2 Objetivo Geral

Este trabalho visa propor um mecanismo de detecção e mitigação de ataques DDoS volumétricos para redes SDN, que possibilite ações mais rápidas para conter o fluxo de tráfego de rede de clientes maliciosos, por meio de técnicas de mitigação que avaliem continuamente o comportamento dos clientes para a priorizar o tráfego legítimo e restringir o tráfego malicioso, para evitar os bloqueios indiscriminados. Para atingir esse objetivo geral, são estabelecidos os seguintes objetivos específicos:

- Adoção de uma abordagem de detecção no plano de dados que possibilite a classificação do fluxo de tráfego de rede de clientes maliciosos de forma eficiente, rápida e precisa, por meio de um modelo de classificação com base no algoritmo de aprendizagem de máquina *Random Forest*, e, assim, reduzir os atrasos no processo de confirmação de qualquer atividade suspeita associada aos ataques DDoS volumétricos e o volume de mensagens de controle encaminhadas ao controlador para realizar as ações de detecção e mitigação.

- Aplicação de uma estratégia de mitigação no plano de dados que gerencia diferentes listas nos dispositivos de encaminhamento, utilizando níveis de confiabilidade com base no comportamento dos clientes, para priorizar os fluxos de tráfego de rede de clientes confiáveis (legítimos) e bloquear o tráfego daqueles com baixo valor de

confiança (maliciosos), a fim de evitar o bloqueio indiscriminado e aprimorar a assertividade da estratégia de mitigação a ser proposta.

- Proposição de um modelo de compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas a partir do estabelecimento de uma confiança global mediante um sistema *fuzzy* no controlador, para o envio aos dispositivos de encaminhamento, criando uma visão única da rede e uma mitigação mais abrangente, permitindo a tomada de decisões locais com base em uma visão ampliada da rede.

1.3 Metodologia

A metodologia aplicada neste trabalho consiste, inicialmente, em uma revisão do estado da arte no que tange às redes SDN, aos principais ataques DDoS (volumétricos e não volumétricos), aos métodos de detecção/mitigação e às soluções relacionadas à segurança em redes SDN, com foco na redução dos impactos causados pelos ataques DDoS volumétricos. As soluções mais recentes e relevantes ao problema serão discutidas, considerando os contextos de aplicação, os planos de atuação, as estratégias de detecção e mitigação utilizadas e as principais limitações apontadas. Com base nessas analises, será definida a proposta de um mecanismo para suprir as lacunas identificadas. O mecanismo proposto será avaliado em um ambiente virtual de simulação para redes SDN. Para tal, será um utilizado simulador conhecido e amplamente aceito pela comunidade científica, de forma a permitir a comparação e validação com propostas semelhantes na literatura. Os resultados obtidos serão comparados com uma solução representativa do estado arte, com base em métricas de desempenho, como a acurácia da detecção, tempo de detecção, tempo de convergência, volume de mensagens de controle e eficácia da mitigação.

1.4 Contribuições

Em particular, as principais contribuições deste trabalho são as seguintes:

- Abordagem para a detecção de ataques DDoS volumétricos no plano de dados, que reduz o tempo de resposta para a confirmação de qualquer atividade suspeita associada a esses ataques e minimiza os atrasos causados pela comunicação frequente com o controlador;

- Estratégia de mitigação de ataques DDoS volumétricos no plano de dados com base no gerenciamento de listas, que prioriza o fluxo de tráfego de rede de clientes com

níveis de confiabilidade aceitáveis (legítimos) e bloqueia (descarta) o fluxo daqueles com baixo valor de confiança (maliciosos), obtidos por meio da análise de seus padrões de comportamento, para minimizar os efeitos causados por esses ataques;

- Modelo de compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas a partir do estabelecimento de uma confiança global, por meio do controlador para o envio aos dispositivos de encaminhamento presentes no plano de dados, aprimorando a avaliação da confiança e permitindo uma resposta mais eficiente, abrangente e adaptativa no combate aos ataques DDoS volumétricos;

- Uma abordagem que integra o plano de controle e o plano de dados para a detecção e mitigação de ataques DDoS volumétricos em toda a rede;

- Experimentos que demonstram a aplicabilidade e a eficácia do mecanismo proposto.

1.5 Publicações relacionadas a Tese

- Artigo desenvolvido em parceria e publicado no periódico *International Journal of Network Management (IJNM) - 2021*, intitulado *"A systematic review on distributed denial of service attack defense mechanisms in programmable networks"* [26] - Qualis-CC A3.

- Artigo publicado e apresentado no *35th Advanced Information Networking and Applications (AINA) - 2021*, intitulado *"DoSSec: A Reputation-Based DoS Mitigation Mechanism on SDN"* [27] - Qualis-CC A2.

- Artigo publicado e apresentado no *9th International Symposium on Computing and Networking Workshops (CANDARW) - 2021*, intitulado *"Detecting DDoS Attacks on SDN Data Plane with Machine Learning"* [28] - Qualis-CC B2.

- Artigo publicado no periódico *Concurrency and Computation: Practice and Experience (CCPE) - 2022*, intitulado *"DataPlane-ML: an integrated attack detection and mitigation solution for software defined networks"* [29] - Qualis-CC A3.

1.6 Demais publicações

- Artigo publicado e apresentado no *21st Workshop on Advances in Parallel and Distributed Computational Models (APDCM)-2019*, intitulado *"Entropy-based DoS attack identification in SDN"* [30] - Qualis-CC B4.
- Artigo publicado na revista *Advances in Science, Technology and Engineering Systems Journal (ASTESJ)-2020*, intitulado *"Enhancing an SDN architecture with DoS attack detection mechanisms"* [31] - Qualis-CC C.
- Artigo publicado e apresentado no *34th Advanced Information Networking and Applications (AINA)-2020*, intitulado *"New programmable data plane architecture based on P4 OpenFlow Agent"* [32] - Qualis-CC A2.

1.7 Estrutura do Documento

Este documento está organizado da seguinte forma:

- **Capítulo 2:** apresenta uma fundamentação teórica acerca dos conceitos relacionados ao desenvolvimento do presente trabalho, tais como redes SDN e suas definições básicas, os tipos de ataques distribuídos de negação de serviço (volumétricos e não volumétricos), métodos de detecção e mitigação desses ataques;
- **Capítulo 3:** apresenta uma revisão do estado da arte acerca dos trabalhos relacionados à segurança em redes SDN, com foco na redução dos impactos de ataques DDoS volumétricos;
- **Capítulo 4:** apresenta a proposta de desenvolvimento do trabalho, detalhando o funcionamento de cada módulo presente no mecanismo desenvolvido, com a descrição das suas funcionalidades, objetivos e integrações;
- **Capítulo 5:** apresenta os resultados obtidos da pesquisa, descrevendo o ambiente experimental utilizado, os métodos de avaliação aplicados, as métricas consideradas, os resultados observados nos experimentos e a análise dos dados obtidos, destacando as contribuições do mecanismo proposto para o estado da arte;
- **Capítulo 6:** apresenta a conclusão do trabalho, sintetizando os principais resultados alcançados, as contribuições para a área de estudo e indica os possíveis trabalhos a serem desenvolvidos no futuro.

Capítulo 2

Fundamentação Teórica

Este capítulo aborda os conceitos sobre redes SDN, os principais controladores, o plano de dados programável e as plataformas para a realização de simulações de ambientes SDN. Também são abordados os principais ataques distribuídos de negação serviço que podem comprometer a disponibilidade dos serviços oferecidos por essa arquitetura de rede, bem como as técnicas para identificá-los e combatê-los. A Seção 2.1 apresenta os conceitos relacionados às redes SDN, a Seção 2.2 expõe os principais ataques distribuídos de negação de serviço, a Seção 2.3 mostra os métodos para identificar esses ataques e a Seção 2.4, os métodos para reduzir os impactos causados por esses ataques.

2.1 Redes Definidas por Software

As redes definidas por *software* surgiram como uma resposta aos desafios impostos pela arquitetura de rede tradicional, caracterizada por sua complexidade e rigidez [3]. Essa nova arquitetura de rede busca oferecer maior flexibilidade, programabilidade e a centralização do controle da rede por meio da separação entre o plano de controle e o plano de dados.

Nesse contexto, o plano de controle é responsável por criar e coordenar as funcionalidades dos dispositivos de rede, como *switches* e roteadores, promovendo uma gestão centralizada [6]. Enquanto, o plano de dados possui a responsabilidade de encaminhamento e roteamento de pacotes conforme as regras definidas pelo plano de controle, proporcionando maior flexibilidade na gestão do tráfego da rede [8]. Esse novo paradigma de rede ganhou força com a criação do protocolo OpenFlow [2], concedendo acesso e controle de forma padronizada a tabela de consulta utilizada pelo *switch* para estabelecer o próximo movimento (ação) de cada pacote recebido, por exemplo, encaminhamento ou descarte [3].

Diferentemente do modelo presente nas redes tradicionais, onde as decisões de encaminhamento são baseadas no endereço IP (*Internet Protocol*) de destino, as redes

SDN utilizam um modelo mais flexível, baseado nos critérios de correspondência e ação (*match+action*) [2, 2]. Nesse modelo, os pacotes podem ser analisados e processados com base em múltiplos campos de cabeçalho, como endereço MAC (*Media Access Control*) de origem/destino, endereço IP de origem/destino, porta de origem/destino e tipo de protocolo [33]. Assim, quando um pacote apresenta valores que correspondem a uma regra definida nos critérios de correspondência (*match*), uma ação (*action*) correspondente é aplicada. Essa ação pode incluir o encaminhamento do pacote para uma porta específica, a modificação do seu cabeçalho ou até mesmo o seu descarte. Esse modelo proporciona um controle mais granular e dinâmico sobre o tráfego de rede, permitindo a implementação de políticas de controle de tráfego mais avançadas e flexíveis para as redes SDN.

A arquitetura SDN oferece uma organização em camadas, que permite um gerenciamento flexível, eficiente e automatizado para lidar com a complexidade na configuração e no controle da rede. As políticas de rede são administradas e implementadas nos controladores e depois encaminhadas para os *switches*. Uma rede SDN é organizada em três camadas (planos) [6], a Figura 2.1 apresenta a arquitetura SDN.

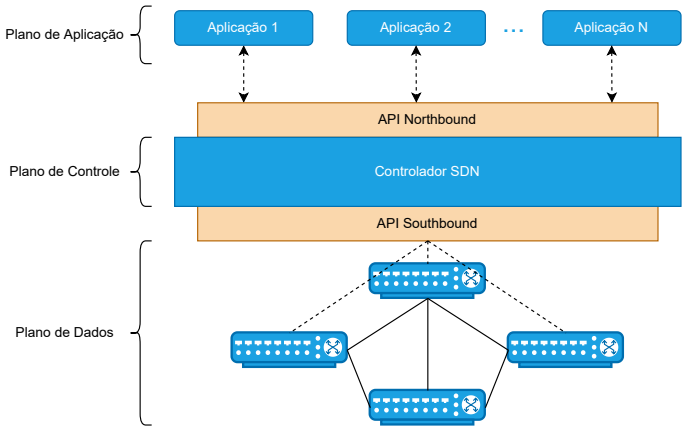


Figura 2.1: Arquitetura SDN.

O plano de dados compreende os dispositivos de rede responsáveis por ações de encaminhamento e roteamento (por exemplo, *switches* e roteadores), além de monitorar informações locais e coletar estatísticas [3]. Esses dispositivos são configurados por meio de um conjunto de regras de fluxo, usadas para redirecionar os pacotes de entrada pertencentes a um fluxo. A comunicação com o plano de controle é realizada pela API (*Application Programming Interface*) *southbound*, por exemplo, OpenFlow [2] e ForCES [34]. A implementação mais conhecida dessa API é o protocolo OpenFlow, que busca padronizar o modo como o controlador se comunica com os dispositivos de encaminhamento da rede por meio de mensagens de controle. Esse protocolo estabelece um canal de comunicação, que conecta o dispositivo de encaminhamento ao controlador, e por meio dessa interface,

o controlador configura e gerencia o dispositivo por meio do envio de comandos (mensagens) de controle para a tabela de fluxo desse dispositivo. A tabela de fluxo, indica quais fluxos o dispositivo de encaminhamento deve processar, sendo que para cada entrada dessa tabela é especificado um conjunto de ações que devem ser aplicadas a cada fluxo. Assim, é possível definir um conjunto de ações para estabelecer a conduta a ser tomada para o fluxo, como o encaminhamento ou o descarte dos pacotes. Essas ações podem ser dinamicamente modificadas, adicionadas ou removidas, sem a necessidade de reconfigurar fisicamente os dispositivos de encaminhamento da rede.

O plano de controle é formado pelos controladores, que gerenciam toda a rede, responsáveis por criar e coordenar as funcionalidades dos dispositivos de rede (*switches* e roteadores) [35]. Eles se comunicam com o plano de dados por meio de uma interface *southbound* usando protocolos, como OpenFlow [2] e ForCES [34]. Adicionalmente, o plano de controle conta com APIs horizontais à esquerda e à direita, denominadas *westbound* e *eastbound*, para o gerenciamento de múltiplos controladores na rede [3]. Essas APIs são um caso especial de interfaces requeridas por controladores distribuídos. As funções dessas interfaces incluem a importação e exportação de dados entre controladores, algoritmos para modelos de consistência de dados, e recursos de monitoramento e notificação, como verificar se um controlador está ativo ou notificar uma substituição em um conjunto de dispositivos de encaminhamento.

O plano de aplicação compreende um conjunto de aplicações de rede, como balanceadores de carga, serviços de QoS (*Quality of Service*), *firewall* e controle de acesso, que auxiliam o controlador no gerenciamento da rede [36]. Elas se comunicam com o controlador através da API *northbound*, que permite às aplicações realizar o controle e o monitoramento de funções da rede, como armazenamento e largura de banda, utilizando os serviços de rede fornecidos pelo controlador [37]. A API converte as solicitações das aplicações em instruções de baixo nível para que os controladores possam transmitir as estatísticas e/ou realizar as ações solicitadas pela aplicação em questão. Essa API pode ser implementada usando a arquitetura REST (*Representational State Transfer*), que utiliza requisições HTTP para extrair, inserir e deletar dados na rede [38].

2.1.1 Controladores SDN

O controlador é o componente principal de uma rede SDN, ele é responsável por coordenar e gerenciar os recursos de toda a rede, oferecendo uma visão unificada e centralizada [35]. Por meio dele, é possível implementar a política de encaminhamento de pacotes na rede, identificar possíveis problemas e reconfigurar a rede em tempo real para garantir o desempenho e a disponibilidade.

Existem diferentes tipos de controladores SDN, programados por meio de uma API, definindo o que será feito em cada um dos planos. Assim, cada controlador utiliza uma linguagem de programação, como C, C++, Java e Python para implementar comandos para auxiliar no controle da infraestrutura da rede. A seguir, são apresentados os principais controladores amplamente utilizados em trabalhos e pesquisas.

- **NOX**: o controlador NOX [39] foi desenvolvido em linguagem C++ e em seguida implementado em Python. Ele atua sobre o conceito de fluxo de dados, em que verifica o primeiro pacote de cada fluxo e procura uma entrada correspondente na tabela de fluxo para aplicar uma determinada ação. Ele possui um conjunto de bibliotecas, que fornecem implementações de funções comuns ao gerenciamento de rede, como módulo de roteamento e classificação rápida de pacotes.

- **POX**: o controlador POX [40] foi desenvolvido em Python, com sua arquitetura fundamentada no controlador NOX, porém com desempenho aprimorado. Por meio dele, é possível executar diferentes aplicativos, como *hub*, *switch*, balanceador de carga e *firewall*.

- **FLOODLIGHT**: o controlador FloodLight [41] foi desenvolvido em Java e possui uma arquitetura modular. Essa arquitetura permite um gerenciamento flexível dos componentes para a administração dos dispositivos. Dentre os componentes, temos o rastreamento MAC, IP, escolha de melhor caminho e acesso ao gerenciamento via interface *web*. Esse controlador incorpora um modelo de *threading*, que permite o compartilhamento de *threads* com outros módulos.

- **OPENDAYLIGHT**: o controlador OpenDaylight [42] é desenvolvido em Java e possui como características-chaves a modularidade e a flexibilidade, que permitem aos desenvolvedores selecionar os recursos mais importantes para eles e criar controladores que atendam às suas necessidades específicas. Ele possui uma arquitetura baseada em micro-serviços, o que permite aos usuários o controle por meio de aplicações e protocolos.

- **RYU**: o controlador RYU [43] é desenvolvido em Python e oferece uma arquitetura modular, permitindo que os desenvolvedores o adaptem às suas necessidades específicas. Além disso, ele também oferece uma API RESTful, que possibilita aos desenvolvedores criar aplicativos de rede baseados em chamadas HTTP, facilitando a comunicação com outros sistemas. Dentre as principais características desse controlador, podemos destacar: (*i*) suporte a múltiplos protocolos de comunicação, incluindo OpenFlow [2] e NetConf [44], permitindo a comunicação com diferentes tipos de dispositivos de rede; (*ii*) escalabilidade, permitindo lidar com grandes redes

de forma eficiente; (*iii*) modularidade, permitindo que os desenvolvedores adicionem novas funcionalidades ao controlador; e (*iv*) documentação abrangente, facilitando o aprendizado e uso da plataforma.

- **ONOS**: o controlador ONOS (*Open Network Operating System*) [45] é desenvolvido em Java e possui como característica a capacidade de operar em redes de grande escala com foco em desempenho, escalabilidade, resiliência e segurança. Esse controlador possui uma arquitetura modular e distribuída, permitindo que múltiplas instâncias do controlador trabalhem em conjunto para fornecer alta disponibilidade e tolerância a falhas. Ele oferece suporte a diversos protocolos de comunicação, como Openflow [2], NetConf [44] e P4Runtime [46], para gerenciar e configurar diferentes dispositivos de rede. Este último é um protocolo de controle do plano de dados que possibilita a comunicação entre controladores SDN e dispositivos de redes programáveis que utilizam a linguagem de programação P4 (descrita na próxima seção). Além disso, o ONOS facilita a criação de aplicações de rede ao oferecer APIs bem definidas, que permitem o desenvolvimento de soluções personalizadas para diferentes cenários de rede.

Por meio dos controladores SDN, os desenvolvedores e administradores de rede têm acesso a um ambiente que possibilita programar e configurar toda a rede, ampliando a gama de soluções a serem implementadas. Contudo, a programação da rede não se limita apenas ao plano de controle, ela pode ser expandida ao plano de dados. A seção a seguir aborda como isso pode ser realizado.

2.1.2 Plano de Dados Programável

As redes SDN trouxeram recursos adicionais que alteram a maneira como os dados são encaminhados, proporcionando flexibilidade e programabilidade à rede. Diversos protocolos foram apresentados, sendo o mais disseminado o OpenFlow [2], que padroniza a comunicação entre o controlador e os dispositivos de redes, como *switches* e roteadores.

O protocolo OpenFlow utiliza *switches* de função fixa do tipo ASICs (*Application Specific Integrated Circuits*), que admitem um conjunto pré-estabelecido de campos de cabeçalho e processam os pacotes usando um conjunto de ações pré-estabelecidas, tornando a adição de novas funcionalidades um processo complexo e dependente de novas atualizações do protocolo e dos fornecedores de *hardware* [1, 47].

Diante de tal limitação, surgiu a necessidade de mudanças na maneira como os pacotes são processados, a fim de proporcionar flexibilidade à rede. A programação da rede é, então, estendida aos dispositivos que compõem o plano de dados. Nesse cenário, surgiu

a linguagem de processamento de pacotes denominada P4 (*Programming Protocol Independent Packet Processors*), que especifica como os dispositivos do plano de dados, como *switches* e roteadores, processam os pacotes de redes [1].

A linguagem P4 é baseada na arquitetura PISA (*Protocol-Idenpendent Switch Architecture*) [1], no qual os *switches* não precisam estar vinculados a nenhum protocolo de rede específico, como o OpenFlow [2]. Essa arquitetura possui duas operações básicas: configurar e preencher. A primeira determina quais protocolos serão suportados e como os *switches* podem processar os pacotes. A segunda adiciona ou remove entradas na tabela de correspondência dos *switches*, estabelecendo as políticas aplicadas aos pacotes. A Figura 2.2 mostra o modelo de encaminhamento dos *switches* programáveis.

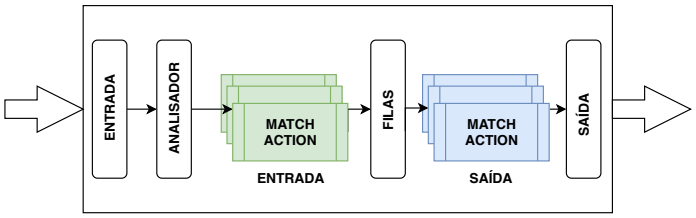


Figura 2.2: Modelo de encaminhamento dos *switches* programáveis, adaptada de [1].

Os pacotes que ingressam no *switch* são tratados pelo analisador, que, então, extrai os cabeçalhos e, assim, define os protocolos que o dispositivo suporta, além do tratamento a ser dado a eles. Os campos extraídos são encaminhados para as tabelas de correspondência do tipo (*match+action*), divididas em tabela de ingresso (entrada) e tabela de egresso (saída). As tabelas de ingresso determinam as ações a serem tomadas, por exemplo, encaminhamento, replicação, rejeição, etc. Já as tabelas de saídas realizam modificações no cabeçalho, se necessário, por exemplo, atribuir a uma fila, alterar a porta de saída, etc. Entre as tabelas, temos o enfileiramento, que processa as especificações de saída, gera as instâncias necessárias do pacote e as envia ao *pipeline* de saída do dispositivo. Após a conclusão de todo o processamento pelo *pipeline* de saída, o pacote é transmitido.

A linguagem P4 permite o uso de estruturas para manter informações sobre determinado pacote, como registradores e contadores. Além disso, possibilita que os pacotes carreguem informações adicionais entre os estágios do *pipeline*, por meio de metadados. Estes, por sua vez, contêm informações essenciais sobre o pacote de entrada, tais como porta de entrada, porta de saída, *timestamp*, prioridade atribuída, fila de transmissão ou outros dados importantes para o processamento do pacote.

Os principais componentes da linguagem P4 são:

- **Cabeçalhos:** descrevem a sequência e estrutura de uma série de campos de *bits* definidos pelo desenvolvedor. Eles incluem especificações de largura, restrições de tipos e valores.

- **Analisador:** especifica como reconhecer os cabeçalhos ou sequência de cabeçalhos válidos no pacote. Para isso, assume-se que o dispositivo de destino consegue implementar uma máquina de estado capaz de analisar o cabeçalho de cada pacote do início ao fim, extraindo seus diversos campos um por um.

- **Tabelas:** as tabelas são do tipo *match+action*, que definem os mecanismos para o processamento dos pacotes, associando ações (*actions*) aos pacotes conforme a correspondência (*match*) realizada pelo analisador.

- **Ações:** definem um bloco de ações primitivas usadas para realizar funções específicas, como copiar um campo para outro, definir um campo específico em um cabeçalho para um valor, entre outras.

- **Controle:** estabelece o controle de fluxo das tabelas, sendo responsável por especificar o caminho que o pacote deve seguir entre uma tabela e outra.

Os componentes citados acima definem um programa escrito na linguagem P4. Assim, o analisador identifica os cabeçalhos presentes em cada pacote que ingressa no dispositivo de encaminhamento. Cada tabela de *match+action* realiza uma busca em um subconjunto de campos de cabeçalho e aplica as ações correspondentes à primeira correspondência encontrada na tabela.

2.1.3 Ferramentas de emulação e simulação

Os mecanismos de emulação e simulação de ambientes SDN permitem testar soluções de rede, que posteriormente podem ser aplicadas no mundo real. Diversos fatores, como a relação custo-benefício, complexidade de gerenciamento e tempo, contribuem para a utilização desses mecanismos. Os principais mecanismos para o desenvolvimento de uma rede SDN são:

- **NS-3**: o NS-3 é um simulador de redes escrito em C++, desenvolvido para fornecer uma plataforma de simulação voltada principalmente para pesquisa e uso educacional [48]. Ele fornece modelos de funcionamento de redes e pacotes, além de disponibilizar um mecanismo de simulação para que os usuários conduzam experimentos. As principais características do simulador incluem a possibilidade de integração com bibliotecas externas e suporte a diferentes sistemas operacionais, como Linux e Windows.

- **EstiNet**: o EstiNet é um simulador e emulador de redes SDN que utiliza uma metodologia de simulação denominada *kernel re-entering*, permitindo o uso de contro-

ladores reais [49]. A ferramenta combina as vantagens das abordagens de simulação e emulação.

- **Mininet**: o Mininet é um emulador de rede desenvolvido por pesquisadores da Universidade de Stanford [50], que permite criar uma rede virtual incluindo diferentes topologias lógicas, como estrela, barramento, anel, etc. A rede é composta por *hosts*, *switches*, roteadores, controladores e enlaces, todos executados em uma única máquina através do *kernel* do Linux. Dessa forma, um *host* no Mininet se comporta exatamente como uma máquina real, oferecendo uma interface de linha de comando simples e intuitiva para criar e gerenciar a rede virtual de maneira fácil e eficiente. Uma das principais vantagens do Mininet é a capacidade de criar e testar redes complexas em um ambiente de laboratório controlado, permitindo que os usuários testem novos projetos de rede sem a necessidade de investir em *hardware* físico. A ferramenta é amplamente utilizada em ambientes acadêmicos e de pesquisa, permitindo a utilização de diferentes controladores de rede, como POX [40], OpenDayLight [42], RYU [50] e ONOS [45].

A utilização dessas ferramentas permite simular ambientes de redes complexos, interagir e personalizar protótipos de redes SDN, além de implementar soluções de monitoramento, segurança e gerenciamento de tráfego, para integrá-las com a rede física existente.

2.2 Ataques Distribuídos de Negação de Serviço - DDoS

Uma das principais ameaças de segurança em redes SDN são os ataques distribuídos de negação de serviço, que exploram a separação entre os planos (dados e controle) dessa arquitetura de rede para comprometer a disponibilidade dos serviços oferecidos por ela [36].

Os ataques DDoS consistem em tentativas coordenadas para sobrecarregar um alvo e causar a sua indisponibilidade por meio de múltiplos dispositivos comprometidos, conhecidos como *bots* ou zumbis, que podem ser utilizados, por exemplo, para gerar um volume massivo de tráfego malicioso para exaurir os recursos dos dispositivos de uma rede SDN, como o controlador e o *switch* [51, 52]. Pois, nessa arquitetura de rede, o encaminhamento de pacotes é baseado na correspondência com as entradas da tabela de fluxo do dispositivo de encaminhamento, presente no plano de dados [3]. Caso não haja correspondência na tabela de fluxo, o *switch* encapsula as informações do cabeçalho do pacote e as envia para o controlador (plano de controle), que devolve essa informação solicitando que seja adicionada uma nova entrada correspondente no *switch*, para que o pacote seja

transmitido até o destino. Assim, os atacantes exploram esse modo de operação dos dispositivos SDN para sobrecarregá-los com um grande volume de pacotes maliciosos, o que resulta na criação de inúmeras novas entradas na tabela de fluxo do *switch*, o que leva à sobrecarga de memória do dispositivo de encaminhamento e à exaustão da capacidade de processamento do controlador, causando efeitos na infraestrutura SDN e nos sistemas finais direcionados, tornando-os inacessíveis para usuários legítimos.

Os ataques DDoS são comumente associados as tentativas de sobrecarregar os recursos da vítima por meio de um alto volume de tráfego malicioso em um curto intervalo de tempo [53]. No entanto, também existem ataques que exploram vulnerabilidades em aplicações e protocolos para esgotar os recursos da vítima sem necessariamente gerar um grande volume de dados [54]. Em função disso, a literatura costuma classificar os ataques DDoS em dois grandes grupos, de acordo com sua forma de atuação: ataques volumétricos e ataques não volumétricos (de baixo volume e baixa taxa) [55].

2.2.1 Ataques Volumétricos

Os ataques volumétricos são caracterizados pelo envio de um grande volume de tráfego, visando sobrecarregar algum recurso da vítima, geralmente a banda disponível ou sua capacidade de processamento, impedindo que solicitações legítimas sejam atendidas [56]. Os ataques mais comuns dessa categoria incluem SYN-*Flood* [57], UDP-*Flood* [58], DNS-*Flood* [59] e ICMP-*Flood* [58].

O ataque SYN-*Flood* aproveita o "*three-way handshake*", que é a base para o estabelecimento de conexões usando o protocolo TCP (*Transmission Control Protocol*), a fim de esgotar os recursos da vítima [57]. Em um cenário normal, o *three-way handshake* funciona da seguinte maneira: o cliente solicita uma conexão enviando uma mensagem do tipo SYN (*Synchronize*) ao servidor, que reconhece esta solicitação enviando uma mensagem SYN-ACK (*Synchronize-Acknowledgment*) de volta ao cliente. O cliente, por sua vez, responde com um ACK (*Acknowledgment*), e a conexão é estabelecida. O ataque consiste no envio de inúmeras solicitações de conexões TCP, com endereços de origem espúrios na forma de segmentos SYN para o servidor. Consequentemente, o servidor aloca e inicializa variáveis de conexão e *buffers*, respondendo com um SYN-ACK para um endereço falso e aguarda a resposta ACK para estabelecer a conexão. Como o ACK não é recebido, a conexão permanece em estado semi-aberto (SYN-RECV) no servidor, ou seja, mantém-se na fila de conexão TCP do servidor. Com várias dessas conexões semi-abertas, o atacante pode sobrecarregar todas as filas de conexão TCP disponíveis em uma máquina servidora de destino, sobrecarregando o servidor, o que eventualmente o impede de responder a solicitações de clientes TCP legítimos.

O ataque UDP-*Flood* utiliza o protocolo UDP (*User Datagram Protocol*) para sobrecarregar uma rede ou servidor de destino com um grande volume de pacotes UDP[58]. Esse protocolo não requer o estabelecimento de uma conexão, ao contrário do TCP [60]. O atacante inunda o alvo com inúmeros pacotes UDP, especificando portas aleatórias na máquina de destino (vítima). Isso faz com que a vítima, ao receber esses pacotes, verifique repetidamente os serviços escutando em cada porta especificada. Quando nenhum serviço é encontrado, a vítima responde com um pacote ICMP (*Internet Control Message Protocol*), utilizado para comunicar problemas na transmissão de dados, notificando o endereço de origem (atacante) de que a porta de destino é inacessível (ICMP - *Destination Unreachable*). Com o volume de pacotes UDP enviados para diferentes portas do alvo, isso resulta no esgotamento de recursos por parte da vítima, que precisa responder todas as solicitações recebidas.

O ataque DNS-*Flood* utiliza o protocolo DNS (*Domain Name System*), que fornece serviço de resolução de endereço para os usuários da Internet, ou seja, mapeia nomes de domínio (como "www.exemplo.com") para os endereços IP associados, permitindo que os dispositivos se comuniquem com os servidores da rede [59, 61]. O objetivo do ataque é sobrecarregar o servidor DNS-alvo com um volume excessivo de requisições (consultas DNS), de modo que ele fique incapaz de atender às requisições de usuários legítimos. Os atacantes também podem utilizar técnicas como "reflexão/amplificação", nas quais uma consulta pequena feita pelo atacante resulta em uma resposta muito maior do servidor DNS, aumentando assim o volume de tráfego de forma exponencial [62]. Além disso, o atacante pode falsificar endereços IP de origem, fazendo com que as respostas sejam enviadas a outros servidores ou dispositivos, dificultando a identificação do atacante.

Por último, temos o ataque ICMP-*Flood*, que utiliza o protocolo ICMP, responsável pelo envio de mensagens e informações de controle entre dispositivos em uma rede, como mensagens de erro (*Destination Unreachable* e *Time Exceeded for a Datagram*) e mensagens de controle (*Echo Request* e *Echo Reply*) [63]. O atacante explora o fato de que protocolo ICMP é unidirecional e não exige autenticação para o envio remoto de inúmeros pacotes de solicitação do tipo *Echo Request* (*ping*) para a vítima, que, por sua vez, responde com um pacote *Echo Reply* para cada endereço IP solicitante. Como a vítima tentará responder a todas as solicitações, isso consome sua largura de banda, impossibilitando-a de atender aos pedidos de usuários legítimos.

2.2.2 Ataques Não Volumétricos

Os ataques não volumétricos exploram os recursos típicos dos protocolos de comunicação utilizados pela aplicação, provocando a exaustão de alguns recursos da vítima por meio de um baixo volume de tráfego. O ataque *Slow Read*, por exemplo, explora o protocolo

TCP [54], enquanto os ataques *Slowloris* e *Slow Body* exploram as características do protocolo HTTP (*HyperText Transfer Protocol*) [54]. Esse último protocolo é utilizado para a comunicação em redes de computadores, onde dois tipos de entidades, conhecidos como cliente e servidor, interagem para realizar tarefas e trocar informações, fornecendo uma maneira eficiente de enviar e receber dados de servidores usando operações como GET, POST e PUT [64].

O ataque *Slow Read* utiliza o protocolo TCP. Nesse tipo de ataque, o atacante realiza uma conexão TCP completa. Porém, quando o servidor responde à solicitação, o atacante informa uma janela TCP menor para aceitar os dados da resposta e, assim, controlar os fluxos. Isso faz com que o servidor envie os dados lentamente para o atacante, mantendo a conexão aberta. O atacante, por sua vez, tende a diminuir o tamanho da janela TCP, chegando próximo a zero, fazendo com que o servidor continue consumindo recursos para manter a conexão ativa, tornando o serviço indisponível para os demais usuários [54].

O *Slowloris* é um ataque que atua na camada de aplicação do modelo OSI (*Open Systems Interconnection*) para sobrecarregar um servidor *web*, tornando-o inacessível para usuários legítimos [65]. Em vez de inundar o servidor com um grande volume de tráfego, como ocorre em ataques DDoS volumétricos, o *Slowloris* usa recursos de servidor com solicitações HTTP que parecem mais lentas que o normal, mas que são legítimas [65]. O atacante inicia várias conexões com o servidor alvo e, em cada conexão, o atacante envia de tempos em tempos uma solicitação HTTP parcial através do método GET, mantendo as conexões abertas sem concluir as solicitações. A vítima (servidor) pressupõe que o cliente está em uma conexão lenta e, assim, mantém a conexão aberta, aguardando que cada solicitação seja concluída. À medida que mais conexões são abertas e mantidas, o servidor fica sobrecarregado com conexões pendentes, esgotando seus recursos até atingir o limite máximo em termos de conexões simultâneas. Como resultado, o servidor se torna incapaz de atender novas solicitações legítimas e fica inacessível para usuários legítimos.

Por fim, temos o ataque *Slow Body* caracterizado pelo envio de dados em campos de formulários através do método POST do protocolo HTTP, no qual inclui um cabeçalho maior do que o seu tamanho real e envia lentamente para manter o servidor ocupado [54]. O atacante envia o corpo da mensagem em pequenos pedaços (*bytes*). O servidor recebe cada fragmento e mantém a conexão aberta enquanto aguarda o restante do corpo. À medida que mais conexões são abertas e mantidas com o mesmo padrão de envio de corpo em pequenas partes, o servidor fica sobrecarregado com conexões pendentes.

2.3 Métodos de detecção de ataques DDoS

Os métodos utilizados para detectar os ataques DDoS (volumétricos e não volumétricos), possibilitam uma ação mais ágil na identificação de atividades maliciosas que podem comprometer a disponibilidade dos serviços oferecidos pela rede.

A literatura dispõe de diferentes maneiras de classificar os métodos de detecção de ataques DDoS. Segundo Lee *et al.* [66], a classificação pode ser baseada no volume de tráfego, no qual é analisada a estrutura do tráfego para tentar encontrar anomalias conforme os níveis de desvio do volume de tráfego normal. Enquanto, Kaur *et al.* [10] categoriza os métodos de detecção com base na técnica utilizada pelas soluções de segurança para analisar o comportamento do tráfego, classificando-os em três grupos: estatística, intermediação e aprendizagem de máquina.

A classificação proposta por Kaur *et al.* [10] nos permite entender de forma mais clara como as diferentes técnicas (métodos) podem ser aplicadas para detectar os mais diversos tipos de ataques DDoS em uma rede. Dessa forma, as técnicas de detecção de ataques DDoS apresentadas a seguir estão organizadas conforme essa classificação.

As técnicas estatísticas realizam a análise de diversas propriedades do tráfego entre a fase normal e a fase de ataque e, em seguida, essas propriedades são utilizadas para criar um modelo de referência do tráfego normal para identificar possíveis desvios comportamentais que caracterizem um ataque DDoS na rede [10]. Técnicas como Qui-quadrado, Entropia e ϕ-Entropia são comumente utilizadas para detectar anomalias de tráfego de redes [67]:

- **Qui-quadrado:** o qui-quadrado [56] ou χ^2, constitui uma medida que retrata a diferença entre duas distribuições consecutivas. O princípio básico deste método é comparar proporções, ou seja, as possíveis divergências entre as frequências observadas e esperadas para um certo evento. A equação é definida como:

$$\chi^2 = \sum_{i=1}^{n} [\frac{(o_i - e_i)^2}{e_i}], \tag{2.1}$$

  onde n é o número total de observações, o_i é a frequência observada para cada classe e e_i é a frequência esperada para aquela classe. Quando as frequências observadas são muito próximas das esperadas, o valor do qui-quadrado é pequeno, ou seja, existe uma alta correlação entre os conjuntos observados. Mas, quando as divergências são grandes, o valor do qui-quadrado assume valores altos, isso significa que não há um relacionamento entre o conjunto de dados observado.

- **Entropia:** a entropia de Shannon [68] mede a probabilidade de um evento acontecer com relação ao número total de eventos, por meio dela é possível descrever o grau de dispersão ou concentração de uma distribuição. A equação é definida como:

$$H = -\sum_{i=1}^{n} p_i \log_2 p_i, \tag{2.2}$$

onde n é o número de diferentes ocorrências no espaço amostral sob análise e p_i a probabilidade associada a cada ocorrência i, ou seja, a frequência de ocorrência de cada item único dividido pelo número total de itens. O resultado deste cálculo varia entre 0 e $\log_2 N$. O valor mínimo indica concentração máxima na distribuição medida, ou seja, um único valor i ocorreu durante todo o intervalo de observação. O valor máximo indica dispersão total na distribuição medida, ou seja, uma distribuição uniforme para todas as ocorrências dentro do intervalo de observação [68]. Em suma, quanto maior a aleatoriedade, maior a entropia e vice-versa. Apesar de retornar um valor negativo, esse valor é comumente transformado em valor positivo para retratar o grau de dispersão ou concentração da informação.

- **ϕ-Entropia:** a ϕ-Entropia [69] é uma medida com base na entropia de Shannon [68], que permite ajustar a sensibilidade da medição da frequência de eventos e a taxa de convergência por parte da entropia. A equação é definida como:

$$H_\phi(X) = -\frac{1}{\sinh(\phi)}(\sum_{i=1}^{n} p_i \sinh(\phi \log_2 p_i)), \tag{2.3}$$

onde p_i representa a probabilidade de ocorrência do evento X; o parâmetro ϕ é usado para ajustar a sensibilidade da medição da frequência de eventos, $\phi > 0$, que satisfaz algumas propriedades como: simetria, normalidade e monotonicidade (variação ou comportamento consistente de uma função em relação à mudança de seus parâmetros). Este último, sustenta que quando $\phi > 0$, $H_\phi(X)$ é monotonicamente crescente em relação ao parâmetro ϕ, mas estritamente falando, o aumento e diminuição de ϕ está relacionado a p_i.

As técnicas de intermediação envolvem a introdução de métodos e mecanismos que operam como intermediários na rede, proporcionando uma camada adicional de segurança entre a origem e o destino para monitorar o tráfego, inspecionar, filtrar pacotes e tomar medidas mais restritivas. Mecanismos como sistema de detecção de intrusão (do inglês, *Intrusion Detection System* - IDS) e *proxies* são comumente utilizados [70].

- **IDS:** o IDS é um mecanismo de segurança utilizado para detectar atividades maliciosas, mediante a análise de pacotes de redes, arquivos de registros (*logs*) e outras

fontes [71]. Dessa forma, ele busca identificar padrões e assinaturas de ameaças conhecidas, bem como comportamentos anormais que possam indicar um novo ataque. Ao detectar uma atividade maliciosa, ele gera alerta para que os administradores de rede possam tomar medidas adequadas para mitigar a ameaça. Os IDS podem ser classificados em dois tipos: IDS baseados em redes (do inglês, *Network Intrusion Detection System* - NIDS) que monitoram o tráfego de rede, e IDS baseados em *hosts* (do inglês, *Host Intrusion Detection System* - HIDS) que analisam atividades em *hosts* ou sistemas individuais [72].

- ***Proxy:*** o *proxy* é um mecanismo que atua como um intermediário entre um cliente (*host*) e um servidor, por exemplo, servidor *web*, permitindo que o cliente faça solicitações ao servidor indiretamente [73]. Desta forma, o cliente conecta-se ao servidor, solicitando algum serviço, página ou outro recurso disponível através do serviço de *proxy*. O mecanismo, por sua vez, recebe essa solicitação e examina para determinar qual ação deve ser tomada, como o encaminhamento ao destino ou bloqueio da requisição. O *proxy* possui diversas formas de implementação e operação, sendo elas: *web proxy* (*cache*), *proxy* reverso e *proxy* transparente [74]. O primeiro realiza o armazenamento de páginas ou arquivos de fontes externas, por exemplo, a Internet, possibilitando com que usuários internos tenham um acesso mais rápido e confiável aos conteúdos por eles requisitados, através do armazenamento em *cache* dos recursos (páginas ou arquivos). O segundo intercepta as requisições dos usuários internos com destino a uma fonte externa, promovendo benefícios como segurança e balanceamento de carga. Por último, o *proxy* transparente é aquele no qual não é necessário fazer configurações adicionais nos usuários para o seu funcionamento. Desta forma, ele intercepta de maneira automática as solicitações do usuário, sem que o mesmo esteja ciente disso [74].

Por último, temos as técnicas de aprendizagem de máquina que permitem detectar os ataques DDoS em uma rede por meio de algoritmos capazes de compreender o comportamento do tráfego de rede com base em dados históricos em larga escala. Esses algoritmos podem ser categorizados em aprendizagem supervisionado, não-supervisionado e por reforço [75, 76]. Restringimos nossa atenção às técnicas de aprendizagem supervisionado devido à sua ampla utilização e facilidade, nas quais os algoritmos são treinados com dados de entrada rotulados para uma saída específica, por exemplo, legítimo ou malicioso [76]. Assim, eles realizam o processo de treinamento até que possam detectar os padrões e relacionamentos próximos entre os dados de entrada e os rótulos de saída, permitindo que eles produzam resultados precisos quando apresentados a dados nunca vistos. Dentre

os algoritmos pertencentes a esta classe, temos o KNN (do inglês, *K-Nearest Neighbors*), SVM (do inglês, *Support Vector Machine*) e RF (do inglês, *Random Forest*):

- **KNN:** o KNN é um algoritmo de classificação que se baseia na proximidade dos seus vizinhos [77]. A ideia deste algoritmo é encontrar os k-vizinhos mais próximos, com base na distância entre os pontos presentes no espaço dimensional. Para determinar a classe de um elemento que não pertença ao conjunto de treinamento, o KNN procura k elementos do conjunto de treinamento que estejam mais próximos deste elemento desconhecido, ou seja, aquele que tenha a menor distância. Estes k elementos são chamados de k-vizinhos e a classe mais frequente será atribuída à classe do elemento desconhecido. A medida de distância para encontrar os vizinhos mais próximos são definidas em termos da distância Euclidiana, dada pela seguinte equação:

$$d\left(x,u\right)=\sqrt{\sum_{i=1}^{n}\left(x_i-u_i\right)^2}, \tag{2.4}$$

  onde d é a distância entre as instâncias x $(x_1, x_2, ..., x_n)$ e u $(u_1, u_2, ..., u_n)$ são os pontos, para um espaço n-dimensional. O objetivo é encontrar k amostras no conjunto de treinamento cuja variável x é relacionada a novas amostras em u, considerando a distância entre os dois pontos.

  A complexidade computacional do método de pesquisa pelos vizinhos mais próximo é proporcional ao tamanho do conjunto de dados de treinamento para cada amostra de teste, onde temos $O(nd)$, em que n é um número de amostras e d é um número de dimensões. Soluções mais elaboradas, como as árvores-KD permitem calcular em tempo $O(dn \log n)$ [78].

- **SVM:** o SVM é um algoritmo que busca encontrar um hiperplano que melhor separe os dados de cada classe e cuja margem de separação seja máxima [79, 80]. O SVM funciona plotando as observações como pontos em um espaço N-dimensional (onde N é o número de recursos) e gerando um hiperplano que maximiza a margem entre as classes. Este limite é então usado para categorizar novas observações. Ele desenvolve um modelo que prevê se uma nova amostra se enquadra em uma categoria ou não. Considere o seguinte conjunto de dados de treinamento $S = \{(x_i, y_i), ..., (x_n, y_n)\}$, para gerar um classificador particular $f(x)$, onde x_i representa o vetor de entrada e y_i a classe padrão de x_i. A partir das entradas, o SVM desenha um hiperplano que melhor separa os dados em classes diferentes. O hiperplano é definido como:

$$f(x) = (w \cdot x) + b = 0, \tag{2.5}$$

onde $w \cdot x$ é o produto escalar entre os vetores w e x e b é o termo independente. Com isso, o hiperplano divide o espaço em duas regiões: $+1$ se $w \cdot x + b > 0$ ou -1 se $w \cdot x + b < 0$, permitindo a separação entre as classes. Para se obter a maximização da margem, deve-se minimizar a norma de w através da seguinte equação:

$$\text{Minimizar} \frac{1}{2}||w||^2. \tag{2.6}$$

O SVM pode ser usado em problemas de classificação ou regressão [81]. Para encontrar o hiperplano que melhor separa os dados, este algoritmo possui uma complexidade em tempo $O(n^3)$, onde n é o número de amostras [82].

- **Random Forest:** o *Random Forest* é um algoritmo que consiste em uma coleção de classificadores estruturados em árvore de decisão $\{h(X, v_k), k, 1...\}$, onde v_k são vetores aleatórios, distribuídos igualmente em todas as árvores da floresta [83]. O processo de classificação consiste em avaliar um conjunto de entradas de dados contendo N elementos e indicar à qual classe a entrada está associada. Esse processo é realizado percorrendo os nós da árvore até que uma folha seja encontrada. A classe à qual a folha escolhida está associada é indicada como saída da previsão desta árvore. Tal processo é executado em todas as árvores da floresta, e o procedimento de escolha é definido por maioria de votos, ou seja, a classe que for indicada pela maioria das árvores será escolhida como resultado indicado pelo classificador.

  Para construir as árvores aleatórias, este algoritmo possui uma complexidade em tempo de $O(mkn \log n)$, onde m é o número de árvores aleatórias, n é o número de amostras e $k \ll m$ é o número de variáveis sorteadas aleatoriamente em cada nó [84].

Os métodos de detecção apresentados ajudam a identificar diferentes tipos de ataques DDoS em uma rede SDN e, assim, fornecem uma visão crítica do que está acontecendo. No entanto, é importante destacar que a detecção não impede que os ataques ocorram. Logo, é necessário conhecer e implementar as estratégias para mitigar os impactos desses ataques na rede.

2.4 Métodos de mitigação de ataques DDoS

Os métodos de mitigação de ataques compreendem um conjunto de estratégias para reduzir os impactos causados pelos ataques DDoS (volumétricos e não volumétricos) na rede, garantindo a disponibilidade dos serviços oferecidos pela rede.

Existem diversas estratégias na literatura para categorizar os métodos de mitigação empregados pelas soluções de segurança. Bawany *et al.* [85], classificam os métodos de mitigação em quatro tipos com base na forma de atuação dos sistemas de prevenção de intrusão desenvolvidos (do inglês, *Intrusion Prevention System* - IPS), cujo objetivo é identificar e bloquear o tráfego de rede malicioso: (*i*) descarte de pacotes maliciosos; (*ii*) bloqueio de portas e/ou endereço IP; (*iii*) redirecionamento de tráfego; e (*iv*) controle de largura de banda. Já Imran *et al.* [19], apresentam uma classificação similar, porém agrupam a ação de descarte e o bloqueio de pacotes em uma única categoria denominada (*i*) bloqueio; englobam as ações de redirecionamento de tráfego e controle de largura de banda na categoria (*ii*) controle (atraso); e adicionam uma nova categoria chamada de (*iii*) gerenciamento de recursos.

A classificação proposta por Imran *et al.* [19] apresenta uma abordagem mais integrada, simplificando o número de categorias e enfatizando a eficácia na combinação de ações para reduzir os impactos causados pelos ataques DDoS. Assim, as técnicas de mitigação descritas a seguir estão organizadas conforme essa classificação:

- **Bloqueio:** a técnica de bloqueio consiste em bloquear os endereços IP e/ou portas utilizadas pelos atacantes e permitir apenas endereços IP legítimos. Desta maneira, é realizada uma análise das regras definidas pelo mecanismo de prevenção, por exemplo, IPS, em que o tráfego de rede em conformidade (legítimo) é encaminhado e o tráfego considerado malicioso é descartado de forma imediata. Essa técnica proporciona um alívio instantâneo aos recursos de rede e requer um alto grau de certeza na classificação das fontes de ataque, já que uma classificação incorreta pode resultar em interrupções no serviço para endereços legítimos, impactando a continuidade dos serviços e comprometendo a sua qualidade e a confiabilidade da rede.

- **Controle (atraso):** a técnica de controle (atraso) baseia-se na ação de redirecionar o tráfego malicioso para um dispositivo especial na rede projetado para inspeção ou atrasar esse tráfego de forma intencional, atribuindo-lhe uma baixa prioridade ou valor de confiança. Ao redirecionar esse tráfego para o dispositivo de inspeção, possibilita-se que o tráfego seja analisado em um ambiente controlado, o que não causa impacto na rede, ajudando a identificar novas ameaças e a formular novas estratégias de mitigação. Ao atrasar de forma proposital o tráfego malicioso, busca-se limitar a taxa de transmissão de pacotes do atacante, a fim de conter o impacto que o ataque DDoS pode causar à rede, priorizando os pacotes de endereços que possuem um alto valor de confiança ou prioridade (legítimos). Essa técnica oferece a vantagem de não bloquear diretamente o tráfego, o que pode ser útil em cenários

onde é difícil distinguir entre tráfego legítimo e malicioso devido à complexidade do ataque.

- **Gerenciamento de Recursos:** a técnica de gerenciamento de recursos envolve o uso de recursos adicionais presentes na infraestrutura de rede, como analisadores de tráfego, mecanismos de detecção e prevenção para combater o tráfego malicioso. No qual são combinados, configurados e otimizados para garantir a eficácia do controle e da segurança da rede. Isso inclui a integração de dispositivos, como *firewalls*, IPS, IDS e outras abordagens de segurança para promover uma ação de mitigação mais eficaz por parte do mecanismo de segurança desenvolvido. Essa técnica requer uma análise cuidadosa para garantir a eficiência da alocação de recursos, pois ela pode apresentar uma complexidade e custo adicionais associados à integração e a manutenção de múltiplos dispositivos de segurança. Além disso, a integração de inúmeros dispositivos pode aumentar o risco de falhas de comunicação entre eles, criando pontos de vulnerabilidade na rede.

Os métodos de mitigação apresentados podem promover um ambiente SDN mais seguro, pois possibilitam uma resposta ativa no combate aos ataques DDoS, minimizando o impacto desses ataques nessa arquitetura de rede.

2.5 Considerações Finais

Neste capítulo, foram abordados os principais conceitos relacionados a redes SDN, bem como os principais ataques DDoS que podem comprometer a disponibilidade dos serviços oferecidos por essa arquitetura de rede, como os ataques volumétricos e não volumétricos. Também foram explorados os métodos de detecção, como os métodos estatísticos e os baseados em aprendizagem de máquina, que permitem analisar o comportamento do tráfego e identificar padrões suspeitos associados a ataques, assim como, os métodos de mitigação que incluem ações de bloqueio e de controle, que possibilitam reduzir os impactos causados por esses ataques. A seção a seguir apresenta uma revisão do estado da arte, com trabalhos relacionados à segurança em redes SDN.

Capítulo 3

Revisão do Estado da Arte

Este capítulo apresenta uma revisão do estado da arte acerca de trabalhos relacionados à segurança em redes SDN, com ênfase aos ataques DDoS volumétricos, que possuem a capacidade de afetar pontos críticos dessa arquitetura de rede em curto intervalo de tempo. Na seção 3.1 são apresentadas as soluções de segurança organizadas conforme a sua técnica de detecção para identificar esses ataques DDoS. A seção 3.2 apresenta uma discussão sobre as soluções estudadas.

3.1 Classificação das soluções de segurança em redes SDN

A seleção das soluções apresentadas a seguir concentrou-se na busca por artigos que abordaram diferentes mecanismos de defesa (detecção e mitigação) contra os ataques DDoS volumétricos, publicados nas principais conferências de redes de computadores, disponíveis na IEEE (*Institute of Electrical and Electronics Engineers*) *Xplore* [86] e ACM (*Association for Computing Machinery*) *Digital Library* [87] nos últimos 10 anos.

As soluções foram classificadas segundo a proposta elaborada por Kaur *et al.* [10], que categoriza as soluções com base na técnica utilizada para analisar o comportamento do tráfego e, assim, detectar qualquer comportamento suspeito. Essa classificação é organizada em três grupos: (*i*) estatística; (*ii*) intermediação; e (*iii*) aprendizagem de máquina. A detecção baseada na técnica estatística baseia-se na premissa de que o tráfego normal segue padrões estatísticos previsíveis, e qualquer desvio significativo desses padrões pode indicar um ataque em andamento. A intermediação envolve a introdução e a configuração de dispositivos intermediários na rede, como sistemas de detecção de intrusões (IDS) ou *proxies*, que atuam como uma camada adicional de segurança, para monitorar o tráfego de rede, inspecionar e filtrar pacotes maliciosos. Por fim, a aprendizagem de máquina

envolve o treinamento de algoritmos capazes entender o comportamento do tráfego de rede, aprendido a partir de dados históricos, para identificar padrões e comportamentos que se correlacionam com ataques DDoS.

A escolha da classificação proposta por Kaur *et al.* [10] para o presente trabalho, nos permite observar a maneira como as soluções de segurança presentes na literatura analisam o comportamento do tráfego para identificar padrões associados aos ataques DDoS volumétricos e realizar as medidas necessárias para reduzir os impactos causados por esses ataques. As subseções a seguir apresentam as soluções estudadas, categorizadas segundo essa abordagem.

3.1.1 Estatísticas

As soluções que empregam as técnicas estatísticas, partem do princípio de que o tráfego normal segue padrões estatísticos previsíveis e que qualquer desvio considerável desses padrões pode indicar a ocorrência de um ataque à rede [10]. Essa abordagem pode ser eficaz para identificar mudanças abruptas no tráfego, como as que ocorrem durante um ataque DDoS volumétrico, com base em uma comparação com o comportamento normal esperado.

Kumar *et al.* [21] propuseram uma solução que atua no plano de controle denominada Safety para detectar e mitigar o ataque DDoS SYN-*Flood*, por meio da entropia para determinar a aleatoriedade dos dados de fluxos. A análise é baseada no endereço IP de destino dos pacotes para detectar anomalias e identificar a origem do ataque. A mitigação envolve descobrir a fonte maliciosa mediante análise do volume de pacotes suspeitos e bloqueá-la na porta de origem do *switch*. Embora a solução apresente bons resultados, realizar o bloqueio de portas do *switch* pode penalizar todos os fluxos legítimos vinculados a essa porta, dependendo da infraestrutura de rede utilizada.

Os autores em [88] propuseram uma solução que atua no plano de dados para a detecção de ataques DDoS, mediante aplicação da entropia. A solução calcula as entropias dos endereços IPs de origem e destino dos pacotes que chegam ao *switch*. Esses pacotes são agrupados em fluxos de entrada, denominadas janelas de observação. Ao completar cada janela, a solução define os valores de entropia para os endereços IPs (origem e destino), visando produzir um modelo de tráfego legítimo. Em seguida, são calculados os limiares de detecção com base no modelo produzido, emitindo um alarme de ataque quando as últimas estimativas de entropia excedem os limiares de detecção. Dadas as restrições do conjunto primitivo de operações fornecidos pela linguagem P4 para o desenvolvimento da solução, como a dificuldade de implementar multiplicação e divisão [1], os autores tiveram que empregar esboços de contagem personalizados para aproximar as frequências de diferentes endereços IP para definir os valores de entropia, tornando difícil a adaptação às

características dinâmicas de um ambiente real. Por fim, a solução não realiza o processo de mitigação, emitindo apenas um alarme quando um ataque DDoS é iniciado na rede.

Kuerban *et al.* [24] propuseram uma solução chamada FlowSec, que atua no plano de controle e gerencia a largura de banda para reduzir (limitar) o número de pacotes enviados ao controlador durante um ataque DDoS. A solução coleta as estatísticas de fluxos dos dispositivos presentes no plano de dados e calcula dinamicamente a largura de banda para com o controlador, definindo um limiar. Quando esse limiar é excedido, o controlador instrui o *switch* a reduzir o número de fluxos enviados por porta para o plano de controle. Em seus experimentos, os autores conseguiram reduzir o volume de pacotes maliciosos encaminhados para o controlador em cerca de 75%. Isso reduz o impacto dos ataques DDoS na rede, mas também pode afetar o tráfego normal, já que clientes vinculados a uma porta suspeita que teve a largura de banda reduzida serão prejudicados, pois a solução não distingue com precisão o tipo de fluxo a ser atrasado.

Similar a solução desenvolvida por Kumar *et al.* [21], os autores em [89] propuseram uma abordagem estatística que analisa um conjunto de dados para a detecção e mitigação de diferentes ataques DDoS no plano de controle. A solução utiliza a entropia para detectar desvios no padrão do tráfego de rede e, assim, identificar o tráfego malicioso. O cálculo da entropia é realizado com base na análise do endereço IP de origem, *flags* TCP, número de pacotes e solicitações por segundo. Se o valor da entropia para cada janela de observação for menor que o limiar predefinido para três janelas consecutivas, a janela em questão será considerada suspeita. A estratégia de mitigação, consiste em realizar o bloqueio do endereço IP de forma permanente que está abaixo do limiar de detecção, para que ele não possa enviar solicitações adicionais ao alvo do ataque. Isso reduz o impacto dos ataques DDoS, no entanto, também pode afetar os clientes legítimos, tendo em vista que a estratégia de detecção de ataques pode gerar inúmeros alarmes falsos, provocando um bloqueio de forma generalizada.

Li *et al.* [90] propuseram uma solução que atua no plano de controle para a detecção de ataques DDoS utilizando a ϕ-Entropia, que permite ajustar os parâmetros conforme as condições da rede para facilitar o processo de descoberta do tráfego malicioso. A solução por meio do controlador extrai o endereço IP de destino de cada pacote e contabiliza o número de ocorrências de cada endereço na janela de observação, na qual a ϕ-Entropia é calculada para todos os pacotes presentes na janela. Se o valor calculado da ϕ-Entropia para a janela for menor que o limiar predefinido para cinco janelas consecutivas, isso indica a presença de um ataque DDoS. Embora a solução apresente bons resultados quando comparada ao uso da entropia de Shannon [68], a atuação exclusiva no plano de controle, pode ocasionar um aumento considerável nas interações frequentes com o plano de dados para coletar, processar e definir os valores de ϕ-Entropia, provocando tempos mais longos

para detectar um ataque DDoS. Além disso, a solução não realiza o processo de mitigação.

Procurando integrar os dois planos de trabalho de uma rede SDN (dados e controle), Yu *et al.* [17] propuseram uma solução colaborativa para a detecção de ataques DDoS, os autores projetaram um mecanismo de detecção preliminar no plano de dados, baseado no método estatístico entropia, para monitorar o tráfego de rede e reportar ao controlador caso alguma anormalidade seja encontrada. Desta maneira, o controlador recebe um alerta com base na detecção preliminar dos dispositivos de encaminhamento e, em seguida, inicia o seu segundo mecanismo de detecção para distinguir a anomalia com mais precisão através da coleta de fluxos, extração de recursos dos fluxos (número de pacotes, número da porta, IP de origem, etc.) e a aplicação do algoritmo de aprendizagem de máquina *Random Forest*. No entanto, a solução pode sobrecarregar o canal de comunicação entre os planos (dados e controle) com o envio de inúmeros fluxos para uma análise mais refinada por parte do dispositivo central, provocando um aumento no tempo de detecção para que o controlador possa confirmar cada suspeita produzida no plano de dados. Além disso, a solução não realiza o processo de mitigação.

Buscando estabelecer uma relação de confiabilidade conforme o comportamento dos clientes na rede, visando realizar um bloqueio de maneira seletiva no combate aos ataques DDoS, os autores em [23], propuseram uma abordagem no plano de controle baseada na confiança para detectar e isolar os ataques DDoS. A solução primeiro estabelece um limiar de confiança com base na média de pacotes transmitidos na rede em direção ao controlador em um ambiente normal, ou seja, aquele em que não há presença de fluxos maliciosos. O valor de confiança de cada cliente é calculado a partir do número de pacotes trafegados por ele durante um determinado período. Aquele que estiver abaixo do limiar de confiança é declarado como malicioso, ou seja, o seu número de pacotes transmitidos está acima do padrão definido como normal pela solução, consumindo mais recursos do que o habitual. Desta forma, o controlador inicia o processo de isolamento deste cliente, alertando os *switches* via mensagens de controle. Ao tomarem conhecimento, eles iniciam o processo de interromper imediatamente a comunicação com o cliente declarado como malicioso, proporcionando um alívio rápido aos recursos de rede. A solução adota uma linha promissora, no entanto, atua de forma exclusiva no plano de controle, o que pode ocasionar um aumento considerável nas interações frequentes com o plano de dados para coletar, processar e definir os valores de confiança, provocando tempos mais longos para detectar qualquer mudança no comportamento do tráfego.

Os autores em [91] propuseram uma solução chamada AEGIS, que atua no plano de controle para detectar e mitigar o ataque DDoS SYN-*Flood*. A solução, por meio de métricas como contagem de mensagens do tipo *Packet-In* não atendidas, o consumo de CPU, memória disponível e a diferença entre SYN e SYN-ACK, realiza a medição

do desempenho do controlador. Assim, se ele estiver com um desempenho muito baixo, há indícios de um possível ataque SYN-*Flood*. A mitigação é realizada por meio de um processo de análise das estatísticas de cada regra de fluxo, criando uma tabela de mitigação para a contagem do endereço MAC e IP. O controlador define regras para o descarte para o endereço MAC que possui uma contagem de IP maior que o estabelecido na tabela de mitigação. No entanto, a solução não informa a taxa de atualização para a tabela de mitigação para a inclusão e retirada de endereços suspeitos, desta forma se um endereço legítimo for classificado como ataque, permanecerá bloqueado de forma definitiva.

Similar as soluções desenvolvidas por Kumar *et al.* [21] e Sumantra *et al.* [89], os autores em [92] fizeram uso da entropia para detectar os ataques DDoS no plano de controle. A solução utiliza a entropia para detectar qualquer desvio no padrão do tráfego, mediante análise de três diferentes limiares adaptativos de detecção: taxa de fluxo de pacotes, valor de entropia e contador. Primeiramente, a taxa de fluxos de pacotes é comparada com o seu respectivo limiar no plano de dados. Após exceder esse limiar, o controlador coleta as estatísticas da tabela de fluxo dos *switches* para calcular a entropia dos endereços IP de origem e destino. Se o valor da entropia for menor que o seu limiar em questão, há um forte indício de ataque, desta forma um contador é inicializado e incrementado. Caso, exceda o limiar do contador, uma mensagem de alerta de ataque é gerada. A etapa de mitigação consiste em coletar as informações relacionadas a porta dos *switches* e o endereço IP de origem. Assim, a solução consegue rastrear o respectivo endereço e bloquear a porta em questão que está acima dos limiares, na qual a solução descarta todas as solicitações subsequentes daquele endereço IP e também remove todas as regras de fluxos da tabela do *switch*. Embora a solução apresente uma estratégia com limiares adaptativos em três níveis, realizar o bloqueio de portas do *switch* pode penalizar todos os fluxos legítimos vinculados a essa porta, dependendo da infraestrutura da rede.

Semelhante ao trabalho desenvolvido por Saini *et al.* [23], os autores em [22] propuseram uma solução no plano de controle para determinar um limiar que diferencie requisições legítimas e maliciosas, visando realizar um bloqueio de maneira seletiva no combate aos ataques DDoS. Os autores desenvolveram um algoritmo estatístico que atribui um valor de confiança e um limiar adaptativo aos clientes da rede, segundo o comportamento de cada cliente. A solução primeiro realiza a extração de alguns campos de cabeçalhos dos pacotes, como tamanho, IP de origem e destino e porta. Esses campos são comparados com limiares predefinidos (estabelecidos em um ambiente sem a presença do tráfego malicioso) para cada campo, que resultarão em acréscimo ou desconto para os seus respectivos contadores. As informações obtidas dos campos de cabeçalhos são utilizadas para calcular o valor de confiança e o limiar do cliente. Assim, em caso de ataque, os contadores dos campos de cabeçalho tendem a aumentar, provocando um impacto negativo no valor

de confiança do cliente, até que ultrapasse o limiar estabelecido pelo controlador, sendo caracterizado como malicioso. Para reduzir os impactos dos ataques, a solução então adiciona o endereço IP do cliente em uma lista de suspensão, para que os pacotes subsequentes deste endereço sejam descartados de forma imediata. Contudo, a centralização de todo o processo (detecção e mitigação) no plano de controle para gerenciar os níveis de confiabilidades dos clientes e seus respectivos limiares podem provocar um aumento no volume de mensagens de controle encaminhadas ao controlador, aumentando o tempo para a detecção e tomada de ações por parte da estratégia de mitigação.

Dawod *et al.* [93] propuseram uma solução que atua no plano de controle para reduzir os impactos do ataque DDoS SYN-*Flood*. A solução utiliza a ferramenta de análise estatística chamada *Paessler Router Traffic Grapher* (PRTG) *Enterprise Monitor*, que monitora o protocolo SNMP (*Simple Network Management Protocol*), oferecendo informações detalhadas sobre o fluxo de dados, a largura de banda, a latência e outros aspectos relacionados ao desempenho da rede. Desta forma, a detecção é realizada com base no comportamento anormal do tráfego malicioso (sobrecarga repentina da largura de banda), que após ser detectado são gerados alarmes para o controlador, que realiza o processo de extração do endereço IP do invasor e o bloqueio do respectivo tráfego por meio de regras de descarte de pacotes enviadas aos *switches*, para reduzir os impactos do ataque. A utilização de uma ferramenta adicional provoca ainda mais atrasos para detectar qualquer mudança no comportamento do tráfego, tendo em vista a atuação da solução no plano de controle. Além disso, ela não aplica nenhum método para diferenciar os endereços recorrentes e legítimos de outros endereços, assim ela pode acabar penalizando endereços legítimos.

Os autores em [94] propuseram uma solução que atua no plano de controle para reduzir os impactos causados pelo ataque DDoS SYN-*Flood*. A solução proposta utiliza o qui-quadrado para identificar discrepâncias no tráfego de rede que possam indicar a presença de tráfego malicioso, por meio da análise dos seguintes recursos: endereço MAC e as *flags* TCP (SYN, SYN-ACK e ACK). Esses recursos são utilizados pelo controlador para construir uma lista com o número de conexões semiabertas por cada *host* na rede em um determinado período. Com base nesses dados, o valor do qui-quadrado é calculado para o *host*, permitindo detectar padrões associados à atividade maliciosa, ou seja, quando o valor do qui-quadrado está abaixo do limiar estabelecido. Para mitigar os impactos do ataque, a solução realiza o bloqueio do endereço MAC do atacante identificado e realiza o descarte dos pacotes subsequentes relacionados a esse endereço. Contudo, a centralização de todo o processo (detecção e mitigação) no plano de controle pode provocar um aumento no volume de mensagens de controle encaminhadas ao controlador, aumentando o tempo para a detecção e a execução das ações de mitigação da solução proposta.

Por fim, González *et al.* [18] propuseram um mecanismo chamado Bungee-ML para combater os ataques DDoS que combina o processamento rápido do plano de dados e a alta capacidade e inteligência do plano de controle, semelhante ao trabalho desenvolvido por Yu *et al.* [17]. A solução executa, no plano de dados, um mecanismo de detecção preliminar mediante o método estatístico entropia, o qual analisa os pacotes recebidos pelo *switch*. Ao identificar fontes suspeitas no plano de dados, elas são encaminhadas ao plano de controle, juntamente com os pacotes subsequentes relacionados a essas fontes, onde o controlador extrai mais informações para classificar os endereços de origem mediante um algoritmo de aprendizagem de máquina (*Random Forest*), realizando uma análise mais profunda para decidir se as fontes suspeitas são de fatos invasores (maliciosos). Após a confirmação, o controlador notifica o plano de dados para incluir essas fontes (endereços IP) em uma lista de suspeitos confirmados, e assim os pacotes recebidos de fontes incluídas nessa lista são automaticamente descartados, visando reduzir os impactos causados por esses ataques. Todavia, a solução pode acabar aumentando o volume de mensagens de controle encaminhadas ao controlador, tendo em vista o volume de pacotes (fontes) a serem confirmados como suspeitos, além disso, ela pode levar mais tempo para confirmar (detectar) uma fonte maliciosa na rede como um ataque DDoS de fato.

3.1.2 Intermediação

As soluções que realizam a intermediação das conexões procuram adicionar uma camada adicional de segurança entre os usuários (origem) e os servidores (destino), a fim de filtrar os pacotes e tomar medidas mais restritivas [10]. Esse tipo de detecção é uma abordagem proativa que visa bloquear o tráfego malicioso antes que ele atinja os recursos da rede.

Shin *et al.* [12] propuseram uma solução que atua no plano de dados, chamada Avant-Guard, para combater o ataque DDoS SYN-*Flood*. Essa solução transforma o *switch* em um *proxy*. Dessa forma, ela intercepta todas as conexões TCP em uma sessão e encaminha apenas as solicitações completas para o controlador, ou seja, as solicitações TCP que executaram o *three-way handshake* de maneira completa. Caso contrário, a conexão é imediatamente descartada pela solução. Como efeito colateral, a solução aumenta o atraso no estabelecimento de conexões legítimas devido ao método aplicado (*proxy*). Além disso, é vulnerável ao cenário em que um atacante utilize um IP já validado pelo mecanismo para realizar um novo ataque.

Visando aprimorar a solução proposta por Shin *et al.* [12], os autores em [13] propuseram uma solução chamada Lineswitch, que também atua no plano de dados para lidar com o ataque DDoS SYN-*Flood*. A solução utiliza a concepção de intermediação probabilística, na qual realiza a intermediação das primeiras conexões de um determinado endereço IP via *proxy* no *switch*, enquanto as demais solicitações subsequentes do mesmo endereço

IP são intermediadas de acordo com uma probabilidade, reduzindo, assim, a carga de trabalho do *switch*. Os endereços IPs que não conseguem realizar o TCP *handshake* de maneira completa são adicionados a uma lista de bloqueio e, em seguida, descartados pela solução, para preservar a memória dos dispositivos de encaminhamentos. Contudo, a utilização de um *proxy*, provoca um aumento no tempo para estabelecer as conexões legítimas e deixa a solução vulnerável ao tipo de ataque em que o atacante utilize um endereço IP já validado pelo mecanismo, para causar uma saturação ao controlador.

Dridi *et al.* [95] propuseram uma solução que atua no plano de controle chamada SDN-Guard para reduzir os impactos causados pelo ataque DDoS SYN-*Flood*. A solução utiliza um IDS, que é responsável por analisar os pacotes e gerar um alerta sobre a probabilidade de ameaça de um ataque SYN-*Flood* na rede. A probabilidade é obtida com base nas estatísticas de fluxos coletadas pelo IDS, dessa maneira a solução pode tomar uma decisão sobre a ação a ser realizada para cada um dos fluxos analisados. Se o fluxo for considerado malicioso, ou seja, se a probabilidade de ameaça desse fluxo estiver acima do limiar predefinido, o IDS emite um alerta para a solução que realiza a ação de bloqueio e descarte dos fluxos para evitar que eles possam sobrecarregar a rede. Nota-se que a solução requer uma comunicação contínua com um agente externo (IDS) para obter as estatísticas da rede, o que pode provocar um aumento no tempo para detectar qualquer mudança na rede que caracterize um ataque DDoS.

A solução desenvolvida por Kim *et al.* [96] atua no plano de controle e se baseia na utilização dos mecanismos TCP *Time Out* e *Round Trip Time* (RTT) para detectar o ataque DDoS SYN-*Flood*. Ela descarta o primeiro pacote SYN de qualquer *host* e estima o tempo de espera com base no tempo entre o primeiro e o segundo pacote SYN, removendo as sessões TCP semi-abertas após o tempo de espera. Se o RTT do ACK for menor que o RTT indicado, o pacote ACK é encaminhado ao servidor. Caso contrário, é descartado e o endereço IP é bloqueado pela solução. Embora o mecanismo apresente resultados promissores, o descarte do primeiro pacote pode causar atrasos nas solicitações dos *hosts*. Além disso, ter um RTT muito curto pode penalizar *hosts* legítimos que enfrentam dificuldades para completar uma conexão, por exemplo, em redes lentas.

Por último, Fan *et al.* [97] propuseram uma solução que atua no plano de controle que faz uso de contêiner para combater os ataques DDoS. O contêiner empregado é o Kubernetes, uma plataforma para automatizar, implantar, dimensionar e gerenciar aplicativos em contêineres [98]. A plataforma é utilizada para recriar o ambiente, caso seja interrompido de forma anormal, por exemplo, em razão de um ataque DDoS. Para isso, a solução adiciona o componente chamado *SDN Controller Manager* (SCM) a estrutura do Kubernetes. Esse método fornece redundância, evitando o ponto único de falha do controlador. Para permitir as conexões legítimas, a solução realiza a intermediação das conexões

TCP entre origem e destino, mediante um sistema de intermediação de conexões (*proxy*) que valida as conexões, evitando que o controlador aloque recursos para o atacante. As conexões completas são encaminhadas ao destino, enquanto as conexões incompletas são descartadas imediatamente. Como efeito adverso, a solução aumenta o atraso causado no estabelecimento de conexões legítimas devido ao método aplicado (*proxy*) e é vulnerável ao cenário em que um atacante utilize um IP já validado pelo mecanismo para realizar um novo ataque.

3.1.3 Aprendizagem de Máquina

As soluções que utilizam técnicas de aprendizagem de máquina fazem uso de algoritmos capazes de compreender o comportamento do tráfego de rede usando dados históricos para identificar padrões e comportamentos correlacionados aos ataques DDoS [10]. Essas soluções desenvolvem modelos que podem distinguir entre o tráfego normal e o malicioso. Esses modelos são alimentados com recursos extraídos dos dados de tráfego, como características de pacotes, informações de fluxo, estatísticas de protocolo, etc [20]. Uma vez treinados, esses modelos podem ser utilizados para detectar automaticamente ataques DDoS em tempo real com base na análise contínua do tráfego de rede.

Tuan *et al.* [20] propuseram a utilização do algoritmo de aprendizagem de máquina KNN para reduzir os impactos dos ataques DDoS. O algoritmo é integrado ao controlador para detectar e descartar o tráfego de ataque. Assim, a solução extraí os seguintes recursos do tráfego para classificá-lo em malicioso ou legítimo: IP de origem, IP de destino, porta de origem e porta de destino. Na etapa de mitigação, para o fluxo considerado malicioso, o controlador impõe uma regra de bloqueio no *switch* para realizar o processo de descarte do tráfego de ataque conforme o seu endereço IP de origem. No entanto, a estratégia aumenta o volume de mensagens de controle encaminhadas ao controlador para a coleta dos dados e posterior classificação. Além disso, realizar o bloqueio de forma definitiva do endereço IP de um *host* suspeito, pode penalizar fluxos legítimos vinculados a este endereço IP, tendo em vista a capacidade dos ataques DDoS em simular o comportamento de um tráfego legítimo.

Semelhante ao trabalho desenvolvido por Yu *et al.* [17] e González *et al.* [18], Macías *et al.* [16] propuseram uma solução chamada ORACLE, que promove a coordenação do plano de controle e de dados para detectar ataques DDoS à rede. A solução realiza a coleta de informações dos fluxos, como a duração, desvio padrão do tempo entre chegadas, tamanho médio do pacote e desvio padrão do comprimento do pacote no plano de dados. Em seguida, realiza o pré-processamento dessas informações e as agrupa em janelas de observação por tempo para encaminhar ao plano de controle em intervalos regulares. O controlador é então responsável por extrair as informações de cabeçalho dos

fluxos recebidos, analisar a sua estrutura para calcular e classificar os fluxos em legítimos ou maliciosos, via algoritmo de aprendizagem de máquina (KNN ou *Random Forest*). Note que a estratégia adotada pela solução busca reduzir a sobrecarga de funções para o controlador, uma vez que as informações de fluxos já foram processadas quando chegam ao plano de controle. No entanto, isso pode acarretar um tempo mais longo para detectar qualquer atividade maliciosa, já que essa função é exclusiva do controlador. Por fim, a solução não realiza o processo de mitigação, emitindo apenas um alarme quando um ataque DDoS é iniciado na rede.

Nurwarsito *et al.* [25] propuseram uma solução para combater os ataques DDoS que atua no plano de controle chamada RYU Framework, na qual os autores desenvolveram um módulo de segurança que utiliza o controlador RYU para classificar o tráfego legítimo e malicioso mediante algoritmo de aprendizagem de máquina *Random Forest*. A solução realiza a extração de alguns atributos em intervalos de tempos regulares para o envio ao controlador, como número médio de pacotes por fluxo, média de *bytes* por fluxo, taxa de crescimento das portas de destino do fluxo e o número de endereços IPs de origem em um determinado intervalo de tempo, para o desenvolvimento do seu classificador de tráfego. Após a classificação do tráfego malicioso, é adicionada uma regra de bloqueio nos *switches*, que descarta os pacotes subsequentes deste tráfego assim que eles retornam à rede. Esta ação reduz o impacto dos ataques DDoS na rede, porém pode afetar o tráfego legítimo, tendo em vista a capacidade dos ataques DDoS em simular o comportamento de um tráfego legítimo.

Para concluir, Das *et al.* [99] propuseram uma solução que atua no plano de controle que age de forma similar ao trabalho desenvolvido por Nurwarsito *et al.* [25], que utiliza algoritmos de aprendizagem de máquina (SVM ou *Random Forest*) nas estatísticas de porta dos *switches*, geradas pelo protocolo OpenFlow para diferenciar o tráfego normal do tráfego de SYN-*Flood* (malicioso). A solução coleta as estatísticas através das mensagens do tipo OFPPortStatsRequest (requisição) e OFPPortStatsReply (resposta) que inclui informações como número de pacotes recebidos, transmitidos, taxa de transferência e *bytes* de uma porta específica, enviadas ao controlador para a aplicação do processo de treinamento do algoritmo de aprendizagem de máquina. Assim, se uma porta específica estiver acima de um limiar predefinido tendo como base as estatísticas coletadas e o processo de treinamento do classificador, isso indica um ataque em potencial. Para reduzir o impacto do ataque, os autores propuseram um identificador de ameaça que analisa as portas dos *switches* e identifica aquela com mais volume de tráfego. Em seguida, ele identifica a origem do tráfego de ataque e realiza a ação de bloqueio. Embora, a solução tenha apresentado resultados promissores, a estratégia de detecção no plano de controle aumenta o volume de mensagens de controle encaminhadas ao controlador e provoca

tempos mais longos para detectar qualquer mudança no comportamento do tráfego. Por fim, o bloqueio de portas pode acabar penalizando *hosts* legítimos ligados a elas.

3.2 Discussão

As soluções apresentadas na seção anterior retratam o estado da arte no que diz respeito às principais soluções de segurança, que procuraram detectar e minimizar os impactos causados pelos ataques DDoS volumétricos. Cada solução escolhida foi analisada sob várias perspectivas, o que levou a identificar as principais características que as distinguem. Essas características foram organizadas em grupos, que serão discutidos a seguir, para guiar nossa discussão acerca das soluções propostas:

- **Camada SDN:** refere-se à camada para a qual a solução foi projetada, considerando a arquitetura SDN, que compreende três camadas de funcionamento (aplicação, controle e dados).
- **Técnica de detecção:** está relacionada ao conjunto de técnicas utilizadas para identificar as atividades maliciosas na rede, como análise estatística, intermediação e aprendizagem de máquina.
- **Técnica de mitigação:** trata-se do conjunto de ações para minimizar os impactos causados pelos ataques DDoS, que incluem ações envolvendo o bloqueio de pacotes (descarte), o uso de equipamentos adicionais e o controle (atraso) do tráfego malicioso.
- **Volume de mensagens de controle encaminhadas ao controlador:** está associado ao aumento do volume de mensagens de controle enviadas ao controlador para lidar com as ações de detecção e mitigação pelo mecanismo de segurança desenvolvido no combate aos ataques DDoS. Foi levado em consideração, durante a análise com base na revisão do estado da arte, que as soluções apresentadas na seção anterior podem ter aumentos variados, de alto, médio ou baixo, dependendo do seu local de atuação na arquitetura SDN. Soluções que operam inteiramente no plano de dados para realizar as ações de controle reduzem a dependência do controlador, assim possuem um volume de mensagens mais baixo. Já as soluções que distribuem essas ações entre o plano de dados e o plano de controle, possuem um volume de mensagens médio. Por outro lado, soluções que realizam toda a ação exclusivamente no controlador possuem um volume alto de mensagens de controle.
- **Tempo para a detecção de uma fonte de ataque:** refere-se ao tempo exigido pela solução para a detecção (confirmação) de um ataque DDoS na rede. Foi levado

em consideração, durante a anáilse com base na revisão do estado da arte, que as soluções apresentadas na seção anterior podem apresentar tempos variáveis, classificados como alto ou baixo, dependendo da localização do mecanismo de detecção para a confirmação de um ataque em curso. Soluções que realizam a confirmação de uma fonte suspeita diretamente no plano de dados, apresentam um tempo mais baixo. Por outro lado, soluções que realizam a confirmação de uma fonte suspeita no plano de controle, apresentam um tempo mais alto.

- **Priorização de tráfego:** aborda a aplicação de mecanismos que permitem a alocação de recursos para garantir que certos tipos de tráfegos tenham prioridade sobre outros.

- **Atribuição de confiabilidade:** descreve a atribuição de confiabilidade aos clientes de uma rede para a priorização de recursos, com base em seus comportamentos ou nas interações com os demais clientes.

- **Abordagem híbrida:** envolve a integração de mais de um plano da arquitetura SDN, possibilitando aproveitar as vantagens de cada plano para otimizar o processo de detecção e/ou mitigação.

A Tabela 3.1 mostra a visão geral das soluções analisadas. Para assegurar a proteção da rede contra os ataques DDoS volumétricos, elas propuseram diferentes abordagens com a aplicação de diversas técnicas, com a possibilidade de combinar diferentes planos de ação e estratégias.

Observa-se que grande parte das soluções concentra suas ações no plano de controle, oferecendo, assim, um gerenciamento centralizado, o que possibilita o desenvolvimento de políticas de controle de tráfego de forma mais centralizada, facilitando o controle e a coordenação de toda a infraestrutura de rede [10]. No entanto, essa abordagem torna o controlador um alvo atraente (ponto único de falha) para os ataques DDoS e requer uma comunicação frequente com o plano de dados para coletar informações e gerenciar os dispositivos de encaminhamento. Essa ação provoca um aumento do volume de mensagens de controle encaminhadas ao controlador para realizar as ações de detecção e mitigação, além de causar tempos mais longos (atrasos) para detectar e confirmar qualquer mudança no comportamento do tráfego associada aos ataques DDoS, para então acionar os mecanismos de mitigação para combater esses ataques.

Para minimizar o volume de mensagens de controle encaminhadas ao controlador e os atrasos durante a detecção e a mitigação dos ataques DDoS, soluções como AvantGuard [12], Lineswitch [13] e o trabalho desenvolvido por Lapoli *et al.* [14], propuseram atuar diretamente no plano de dados, proporcionando uma ação mais célere para detectar

Tabela 3.1: Visão geral das soluções discutidas.

Solução	Ano	Camada SDN	Técnica Detecção	Técnica Mitigação	Volume Mensagens	Tempo Detecção	Priorização Tráfego	Atribuição Confiabilidade	Abordagem Híbrida
Shin *et al.* [12]	2013	dados	intermediação	bloqueio	baixo	baixo	não	não	não
Dridi *et al.* [95]	2017	controle	intermediação	gerenciamento de recursos	alto	alto	não	não	não
Ambrosin *et al.* [13]	2017	dados	intermediação	bloqueio	baixo	baixo	não	não	não
Kumar *et al.* [21]	2018	controle	estatística	bloqueio	alto	alto	não	não	não
Lapoli *et al.* [14]	2019	dados	estatística	nenhuma	baixo	baixo	não	não	não
Kim *et al.* [96]	2019	controle	intermediação	bloqueio	alto	alto	não	não	não
Tuan *et al.* [20]	2019	controle	aprendizagem de máquina	bloqueio	alto	alto	não	não	não
Kuerban *et al.* [24]	2019	controle	estatística	controle	alto	alto	não	não	não
Macías *et al.* [16]	2020	dados e controle	aprendizagem de máquina	nenhuma	médio	alto	não	não	sim
Sumantra *et al.* [89]	2020	controle	estatística	bloqueio	alto	alto	não	não	não
Li *et al.* [90]	2020	controle	estatística	nenhuma	alto	alto	não	não	não
Yu *et al.* [17]	2021	dados e controle	estatística e aprendizagem de máquina	nenhuma	médio	alto	não	não	sim
Nurwarsito *et al.* [25]	2021	controle	aprendizagem de máquina	bloqueio	alto	alto	não	não	não
Saini *et al.* [23]	2021	controle	estatística	controle	alto	alto	não	sim	não
Ravi *et al.* [91]	2021	controle	estatística	bloqueio	alto	alto	não	não	não
Mishra *et al.* [92]	2021	controle	estatística	bloqueio	alto	alto	não	não	não
Das *et al.* [99]	2022	controle	aprendizagem de máquina	bloqueio	alto	alto	não	não	não
Salem *et al.* [22]	2022	controle	estatística	bloqueio	alto	alto	não	sim	não
Dawod *et al.* [93]	2022	controle	estatística	gerenciamento de recursos	alto	alto	não	não	não
Shalini *et al.* [94]	2023	controle	estatística	bloqueio	alto	alto	não	não	não
Fan *et al.* [97]	2023	controle	intermediação	gerenciamento de recursos	alto	alto	não	não	não
González *et al.* [18]	2023	dados e controle	estatística e aprendizagem de máquina	bloqueio	médio	alto	não	não	sim
Este trabalho	2025	dados e controle	aprendizagem de máquina	bloqueio e controle	médio	baixo	sim	sim	sim

e reduzir os impactos desses ataques [15]. Todavia, elas ficaram limitadas devido às dificuldades de desenvolver mecanismos de detecção mais sofisticados e, ao mesmo tempo, garantir o processamento de pacotes em alta velocidade no plano de dados [15]. Em função disso, soluções como Oracle [16], Cooperative-DDoS [17] e Bungee-ML [18], procuraram então realizar uma abordagem híbrida, em que realizam parte das ações de detecção no plano de dados e a outra parte no plano de controle. Essas soluções, em sua maioria, acabam aumentando o volume de mensagens de controle enviadas ao controlador para a ação de detecção e mitigação e o tempo de confirmação para um ataque, já que o dispositivo central é responsável por confirmar cada suspeita produzida no plano de dados [10].

No que se refere às abordagens desenvolvidas pelas soluções para a detecção e a redução dos impactos dos ataques DDoS volumétricos. A detecção baseada no método estatístico entropia e a técnica de mitigação envolvendo a ação de bloqueio são as mais empregadas, tendo em vista a baixa complexidade ao aplicar o método estatístico e o alívio instantâneo aos recursos de rede proporcionado pela ação de bloqueio. Embora, essas ações visem detectar o tráfego suspeito e bloqueá-lo de maneira rápida, elas têm apresentado algumas adversidades, como baixa taxa de assertividade por parte dos mecanismos de detecção e a penalização do tráfego legítimo vinculado a uma determinada porta do *switch* ou

endereço IP que está gerando esse tipo de tráfego por parte dos mecanismos de mitigação, provocando um bloqueio indiscriminado [20, 21, 24, 22].

Para reduzir o bloqueio indiscriminado e aumentar a taxa de assertividade por parte dos mecanismos de mitigação e detecção, as soluções desenvolvidas por Salem *et al.* [22] e Saini *et al.* [23], buscaram adotar novos métodos. Por exemplo, a atribuição de confiabilidade para os clientes da rede com base em seu histórico de comportamento e suas interações com os demais clientes e, assim, realizar um bloqueio seletivo, restringindo apenas os clientes que estão abaixo do grau de confiabilidade. Porém, essas soluções atuam apenas no plano de controle e não aplicam nenhum mecanismo de priorização que possibilite a alocação de recursos para garantir que o tráfego legítimo tenha prioridade sobre o tráfego malicioso, provocando uma disputa desleal por recursos, tendo em vista o volume do tráfego de ataque. Além disso, gera atrasos no processo de confirmação de qualquer atividade suspeita associada aos ataques DDoS volumétricos e um aumento do volume de mensagens de controle para gerenciar o grau de confiabilidade dos clientes e as demais ações de controle do tráfego de rede [15, 10].

Observa-se a necessidade de um mecanismo de detecção e mitigação que possibilite ações mais rápidas para conter o fluxo de tráfego de rede malicioso, que evite penalizar os clientes legítimos da rede e, ao mesmo tempo, reduza o volume de mensagens de controle encaminhadas ao controlador e os atrasos na execução das ações de detecção e mitigação. Dessa forma, este trabalho busca superar as limitações apresentadas anteriormente, realizando a detecção (por exemplo, com o uso de algoritmos de aprendizagem de máquina) e a mitigação (por exemplo, a priorização do tráfego dos clientes com níveis de confiabilidade aceitáveis - legítimos, e o bloqueio (descarte) do tráfego daqueles com baixo valor de confiança - maliciosos) no plano de dados, visando reduzir os atrasos para detectar e confirmar qualquer mudança no comportamento do tráfego, o volume de mensagens de controle, e o bloqueio indiscriminado dos clientes.

Além das abordagens presentes no mecanismo proposto para reduzir os impactos dos ataques DDoS volumétricos no plano de dados, busca-se promover a integração entre os planos da arquitetura SDN (dados e controle) por meio de um modelo de compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas com base no estabelecimento de uma confiança global calculada a partir de um sistema *fuzzy* no controaldor, para o envio aos dispositivos de encaminhamento presentes no plano de dados, permitindo uma abordagem híbrida que viabilize a tomada de decisões locais com base em informações globais compartilhadas para uma resposta mais eficiente e adaptativa no combate aos ataques DDoS volumétricos.

3.3 Considerações Finais

Este capítulo apresentou uma revisão do arte sobre as soluções de segurança em redes SDN que visam reduzir os impactos causados pelos ataques DDoS volumétricos. Foi realizada uma análise comparativa dos trabalhos existentes, levando em consideração aspectos como a camada SDN de atuação, as diferentes técnicas de detecção e mitigação utilizadas, o volume de mensagens de controle encaminhadas ao controlador, o tempo de detecção (confirmação) de um ataque, a priorização de tráfego, a atribuição de confiabilidade e a adoção de uma abordagem híbrida. A análise revelou algumas lacunas significativas nas soluções atuais, como a centralização das ações de detecção e mitigação no plano de controle, que podem causar atrasos no tempo de resposta para combater os ataques DDoS volumétricos e o aumento do volume de mensagens de controle encaminhadas ao controlador para realizar as ações de identificação e contenção dos impactos causados por esses ataques. Além disso, muitas das estratégias desenvolvidas pelas soluções para reduzir os impactos desses ataques, que, embora visem restringir o tráfego malicioso, acabam penalizando uma parte significativa do tráfego legítimo, provocando bloqueios de forma indiscriminada ao tráfego dos clientes legítimos.

Capítulo 4

Mecanismo de detecção e mitigação de ataques DDoS volumétricos em redes SDN

As redes SDN têm proporcionado uma série de benefícios, como gerenciamento centralizado, maior flexibilidade e controle da infraestrutura de rede [100]. Isso possibilita adicionar novas funcionalidades conforme as necessidades dos usuários, incluindo, por exemplo, o desenvolvimento de funções de segurança, o gerenciamento de fluxos e o provisionamento de serviços [7]. Apesar dos benefícios oferecidos por essa arquitetura de rede, a segurança ainda é motivo de preocupação, pois a separação entre o plano de dados e de controle aumenta a superfícies de possíveis ataques [8]. Nesse contexto, destacam-se os ataques DDoS de natureza volumétrica, que podem sobrecarregar os recursos críticos da rede por meio do envio massivo de tráfego malicioso e/ou da exploração de falhas em protocolos e serviços que estão sendo executados na infraestrutura-alvo [9].

Na arquitetura SDN, o controlador assume o papel central da rede, sendo responsável por coordenar o funcionamento de toda a rede [101]. Essa centralização, embora traga benefícios operacionais, também o torna um ponto único de falha e, portanto, um alvo estratégico para ataques. Neste cenário, os ataques DDoS volumétricos se destacam por sua capacidade de comprometer a disponibilidade do controlador ao enviar grandes volumes de tráfego malicioso, limitando a capacidade do controlador de lidar com o tráfego legítimo da rede, levando à sua indisponibilidade e, consequentemente, afetando os demais serviços oferecidos na rede [19].

Conforme abordado no Capítulo 3, as soluções apresentaram diversas estratégias para reduzir os impactos dos ataques DDoS volumétricos e proteger o controlador. Apesar, do número de soluções desenvolvidas para combater esses ataques em uma rede SDN, ainda permanecem algumas lacunas que podem deixar essa rede exposta, como a centralização

das ações de detecção e mitigação no plano de controle. Essa centralização provoca tempos mais longos (atrasos) para detectar e confirmar qualquer mudança no comportamento do tráfego de rede associado a esses ataques, e o aumento do volume de mensagens de controle encaminhadas ao controlador para realizar as ações de identificação e contenção desses ataques na rede, aumentando o seu consumo de recursos. Além disso, muitas dessas soluções adotam o bloqueio do tráfego de forma indiscriminada, o que, embora vise restringir o acesso do tráfego malicioso ou suspeito e proporcionar um alívio instantâneo aos recursos de rede, acaba penalizando uma parte significativa do tráfego legítimo para minimizar os impactos desses ataques.

Portanto, o objetivo deste trabalho é propor um mecanismo de detecção e mitigação que possibilite respostas mais rápidas para conter o fluxo de tráfego de rede de clientes maliciosos, por meio de técnicas de priorização de tráfego que evitem penalizar os clientes legítimos e reduza o número de pacotes encaminhados ao controlador, preservando os seus recursos, evitando a sobrecarga e garantindo sua disponibilidade mesmo durante situações de ataque. Para isso, propõe-se, primeiramente uma abordagem de detecção no plano de dados mediante um modelo de classificação com base no algoritmo de aprendizagem de máquina *Random Forest* para classificar o fluxo de tráfego de rede malicioso próximo aos pontos de ingresso na rede de forma eficiente, rápida e precisa, e, assim, reduzir o atraso para identificar e confirmar esse tipo de tráfego na rede, e o volume de mensagens de controle encaminhadas ao controlador. Seguido por um método de mitigação também presente no plano de dados, por meio do gerenciamento de diferentes listas nos dispositivos de encaminhamento, utilizando níveis de confiabilidade com base no comportamento dos clientes, para priorizar o fluxo de tráfego dos clientes com níveis de confiabilidade aceitáveis (legítimos) e bloquear (descartar) o tráfego daqueles com baixo valor de confiança (maliciosos), mediante a análise de perfis de tráfego e dados de cabeçalhos de pacotes, para evitar o bloqueio indiscriminado dos clientes.

Além das abordagens presentes no mecanismo proposto para reduzir os impactos dos ataques DDoS volumétricos no plano de dados, este trabalho também propõe um modelo de compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas com base no estabelecimento de uma confiança global calculada a partir de um sistema *fuzzy* no controlador, para o envio aos dispositivos de encaminhamento presentes no plano de dados, promovendo uma colaboração eficiente entre o plano de dados e de controle, e maior rapidez nos processos de detecção e mitigação em diferentes pontos da rede.

As seções a seguir apresentarão uma visão geral do mecanismo proposto e os procedimentos para o seu desenvolvimento, detalhando as ações para atenuar as lacunas

observadas no Estado da Arte.

4.1 Visão geral do mecanismo proposto

O mecanismo proposto visa realizar ações para identificar e conter o fluxo de tráfego de rede de clientes maliciosos. Para isso, ele foi organizado em dois procedimentos: (*i*) Procedimento 1: detecção e mitigação no plano de dados; e (*ii*) Procedimento 2: compartilhamento de informações globais por meio do plano de controle.

O procedimento 1 concentra-se na implementação de ações no plano de dados para detectar e conter o fluxo de tráfego de rede de clientes maliciosos próximo aos pontos de ingresso na rede e priorizar o tráfego daqueles com níveis de confiabilidade aceitáveis (legítimos), bem como encaminhar ao plano de controle os dados necessários para a definição da confiança global desses clientes. Esse procedimento é composto por 5 módulos: Coleta de Dados; Classificação; Agregação do Valor de Confiança; Gerenciamento das Listas; e Gerenciador de Envio.

O procedimento 2 foca no desenvolvimento de ações de coordenação no plano de controle para o compartilhamento de informações globais, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas a partir do estabelecimento de uma confiança global, por meio do controlador para o envio aos *switches*, para promover uma abordagem híbrida entre os planos da arquitetura SDN (plano de dados e de controle) por parte do mecanismo proposto. Esse procedimento é composto por 3 módulos: Extração dos Dados; Calcular a Confiança Global; e Disseminação dos Dados.

A Figura 4.1 apresenta a integração entre os módulos presentes nos procedimentos do mecanismo proposto.

No instante em que os pacotes de rede chegam ao *switch*, o módulo de coleta de dados é responsável por coletar os dados desses pacotes, agrupá-los em fluxos que correspondem a uma sequência de pacotes que compartilham características comuns e são transmitidos de uma origem para um destino específico [20], armazenar as estatísticas desses fluxos e enviá-las ao módulo de classificação, mediante janelas de observação, que correspondem a intervalos de tempo ou a um número fixo de fluxos, nos quais determinados eventos, dados ou métricas são analisados, organizados e encaminhados [16]. As janelas são usadas nesse módulo para observar o comportamento do tráfego, extrair suas características e reportá-las ao módulo seguinte. O módulo de classificação é responsável por classificar os fluxos de tráfego dos clientes em legítimos ou maliciosos por meio do algoritmo de aprendizagem de máquina do tipo supervisionado *Random Forest* e produzir um valor de

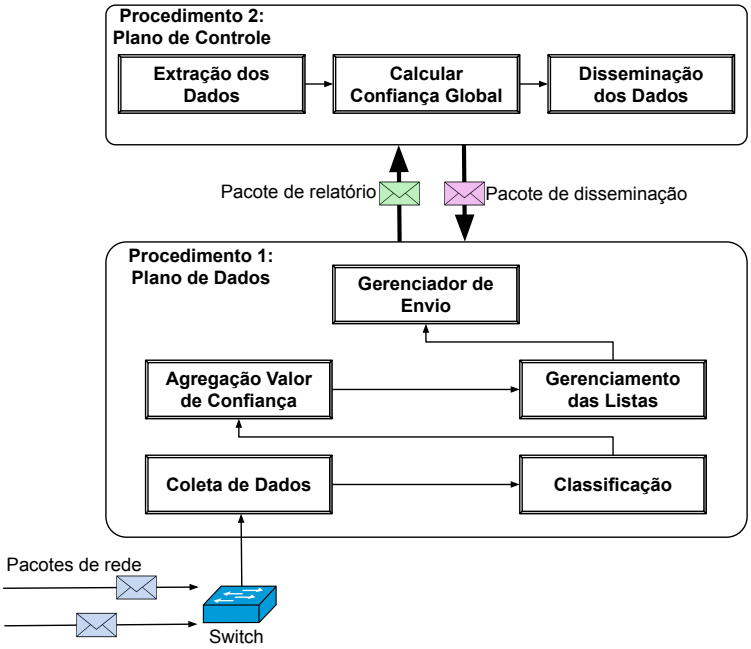


Figura 4.1: Módulos presentes nos procedimentos do mecanismo proposto.

confiança para cada classificação de fluxo desses clientes na rede, ou seja, com base no comportamento de cada um.

Os valores de confiança produzidos para cada cliente são agregados à medida que seus fluxos são classificados na rede para formar um valor de confiança local ao longo do tempo, por meio do módulo de agregação do valor de confiança. O módulo de gerenciamento das listas é responsável por atribuir o fluxo de tráfego de rede dos clientes para as listas locais do *switch* conforme o comportamento de cada cliente a partir do seu valor de confiança, para dar preferência (priorização) ao tráfego dos clientes legítimos e bloquear (descartar) o tráfego malicioso, mantendo o serviço para aqueles que atendem a um determinado nível de confiança. Por fim, o módulo (gerenciador de envio) é responsável por enviar ao plano de controle o valor mais recente de confiança local dos clientes presentes nas listas locais do *switch*, por meio de pacotes de relatórios organizados em janelas de observação. Esses pacotes consistem em pacotes normais que são clonados e tem a sua cópia modificada para incluir os dados de controle que são os valores de confiança local dos clientes anexados em um cabeçalho customizado.

No momento em que os pacotes de relatórios chegam ao plano de controle, o módulo de extração dos dados é responsável por extrair os cabeçalhos desses pacotes para obter os valores de confiança local de cada cliente computados pelos *switches* e organizá-los em uma estrutura de dados. Os dados presentes nessa estrutura são organizados e encaminhados em intervalos predefinidos ao módulo (calcular a confiança global), responsável por

determinar a confiança global de cada cliente e sua respectiva classificação (não confiável, parcialmente confiável ou confiável) por meio de um sistema *fuzzy* [102]. Esse módulo integra os diferentes valores de confiança local gerados pelos clientes da rede, proporcionando uma avaliação unificada que reflete a confiabilidade geral de cada cliente presente na rede.

O módulo de disseminação dos dados é responsável por enviar aos *switches* as ações de controle referentes à classificação da confiança global de cada cliente, por meio de pacotes de disseminação organizados em janelas de observação. As ações de controle correspondem as medidas a serem tomadas pelo mecanismo proposto para aplicar a política de restrição ou priorização do fluxo de tráfego de rede dos clientes de maneira global que podem incluir o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinada a partir do valor de confiança global do cliente estabelecido pelo sistema *fuzzy*. Esse processo cria uma visão unificada da rede, permitindo que os *switches* tomem decisões localmente, com base nas informações globais compartilhadas, garantindo uma operação mais eficiente, abrangente e coordenada por parte do mecanismo proposto.

Para o detalhamento do funcionamento de cada módulo presente nos procedimentos mencionados, é apresentada as definições adotadas relacionadas aos elementos fundamentais envolvidos para o desenvolvimento do trabalho, elaborada com base na arquitetura SDN e nos seus elementos fundamentais, tais como os dispositivos presentes nessa infraestrutura: *switches*, enlaces, controlador e clientes.

A rede SDN é representada por um grafo não direcionado $G = (S, E)$. Essa modelagem segue a representação tradicional de redes de computadores como grafos, amplamente adotada na literatura, em que dispositivos (como roteadores e *switches*) e conexões físicas ou lógicas são abstraídos como nós e arestas, respectivamente [103]. Neste trabalho, essa representação convencional é estendida para incluir, além dos dispositivos de encaminhamento, o controlador SDN, de forma a contemplar a interação entre o plano de dados e o plano de controle:

- $S = \{s_1, s_2, \ldots, s_p\}$ denota o conjunto de *switches* da rede, com $p \in \mathbb{N}^*$ representando a quantidade total de nós de comutação;
- $E \subseteq S \times S$ representa o conjunto de enlaces físicos ou lógicos estabelecidos entre os *switches*.

Cada enlace $e = (s_i, s_j) \in E$ indica uma conexão direta entre os *switches* s_i e s_j, $1 \leq i < j \leq |S|$. Considera-se que os enlaces são não direcionados, de modo que a presença de $(s_i, s_j) \in E$ implica automaticamente $(s_j, s_i) \in E$.

A infraestrutura é gerenciada por um controlador central Cr, responsável por manter uma visão global do grafo G e por implementar as políticas de encaminhamento, conforme o paradigma SDN, que dissocia o plano de controle do plano de dados.

Os clientes da rede são representados pelo conjunto:

$$N = \{n_1, n_2, \ldots, n_k\},$$

em que $k \in \mathbb{N}^*$ corresponde ao número total de clientes ativos. Cada cliente $n_j \in N$ pode estar conectado simultaneamente a um ou mais *switches* $s_i \in S$. Para formalizar essa associação, define-se a relação de conexão:

$$\mathcal{R} \subseteq S \times N,$$

de forma que $(s_i, n_j) \in \mathcal{R}$ indica a existência de uma conexão entre o *switch* s_i e o cliente n_j.

Adicionalmente, para cada par $(s_i, n_j) \in \mathcal{R}$, é atribuído um valor de confiança local $\tau(s_i, n_j)^t \in [0, 1]$, o qual representa a confiança estimada que o *switch* s_i possui em relação ao cliente n_j, no instante de tempo t. Esse valor é inferido com base nas interações observadas entre o cliente e o *switch* ao longo de uma janela temporal definida, refletindo comportamentos como tráfego anômalo, frequência de comunicação ou conformidade com políticas estabelecidas.

A Figura 4.2 apresenta uma ilustração esquemática da rede SDN, destacando os *switches*, os enlaces entre eles, a conexão dos clientes aos respectivos nós de comutação, bem como a presença do controlador central.

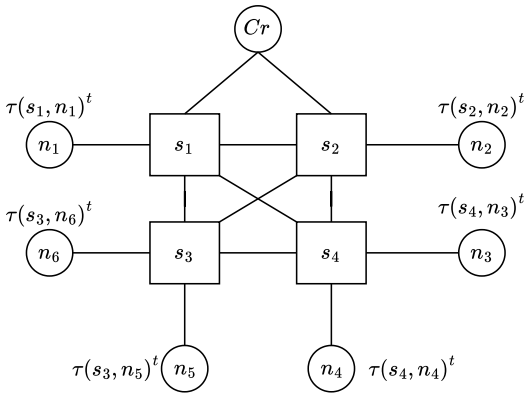


Figura 4.2: Representação esquemática da rede SDN como um grafo não direcionado.

4.2 Procedimento 1: detecção e mitigação no plano de dados

Esta seção detalha os módulos presentes no procedimento 1 do mecanismo proposto. Esse procedimento busca estabelecer meios para detectar e mitigar os ataques DDoS volumétricos diretamente no plano de dados. Para o desenvolvimento dos módulos desse procedimento, os *switches* são programados por meio da linguagem P4, que permite processar os pacotes de rede trafegados na rede em tempo de execução [1]. Assim, o *switch* habilitado para essa linguagem consegue extrair os dados de um pacote e realizar as ações de detecção e mitigação definidas pelo mecanismo proposto.

4.2.1 Coleta de Dados

O módulo de coleta de dados consiste em coletar os dados dos pacotes de rede, agrupá-los em fluxos, calcular suas estatísticas e encaminhá-las ao módulo de classificação, que classificará o tráfego de rede em legítimo ou malicioso e estabelecerá um valor de confiança para cada classificação de fluxo realizada.

Os dados coletados pelo módulo de coleta de dados podem ser agrupados em blocos w_v, $v > 0$, que chamamos de "janela de observação". As janelas podem ser definidas com um número fixo de fluxos, definidas como janelas baseadas em fluxo, ou em vários fluxos em um determinado período, caracterizando janelas baseadas em tempo. No primeiro caso, as janelas observadas possuem o mesmo número de fluxos (ou seja, $|w_v| = |w_h|$, $0 < v < h$). Nas janelas por tempo, é possível que $|w_v| \neq |w_h|$, $0 < v < h$, pois o número de fluxos pode variar ao longo do tempo. A ideia por trás das janelas de observação é agrupar os pacotes observados em fluxos com base em seus atributos, de modo a otimizar o processo de coleta e envio dos dados. Então, é realizado o processo de mapeamento de pacotes para fluxos no *switch* P4, como pode ser observado na Figura 4.3. A definição do tamanho e o tipo da janela de observação (baseada em tempo ou por fluxo) para o módulo em questão será apresentada no Capítulo 5.

Para determinar os pacotes relacionados a um fluxo, utiliza-se uma 5-tupla composta por seus endereços IP de origem/destino, porta de origem/destino e protocolo. O analisador (primeiro bloco incluído na arquitetura do *switch* P4) recebe os pacotes de entrada e os valida, extraindo as informações do cabeçalho do pacote, como protocolo, *flags*, tamanho, endereço de origem e destino, etc. A partir dessas informações, são utilizadas tabelas do tipo *match+action* que determinam as ações a serem tomadas conforme a entrada de dados [104].

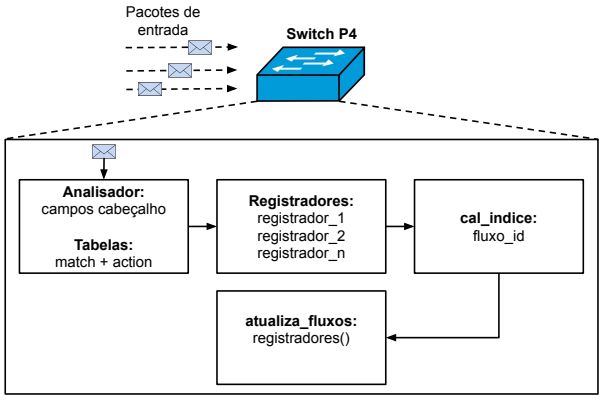


Figura 4.3: Processo de mapeamento de pacotes para fluxos.

As tabelas do tipo *match+action* serão responsáveis por processar os fluxos de rede, aplicando regras de encaminhamento ou controle com base nas características relacionadas a cada fluxo. Para isso, são criados registradores para definir uma estrutura de dados para armazenar as estatísticas de cada fluxo, que podem incluir, por exemplo, a sua duração, o seu tamanho em *bytes*, o total de pacotes encaminhados, o número de pacotes e *bytes* de fluxo por segundo, etc. Um registrador P4 é um *array* de tamanho fixo que usa um índice para apontar para os elementos armazenados [104]. Utilizou-se um conjunto desses registradores para construir uma estrutura similar a uma matriz de registros, onde cada linha armazena valores das mesmas estatísticas pertencentes a fluxos diferentes, e cada coluna armazena estatísticas diferentes do mesmo fluxo. Dessa forma, é criada a função *calc_indice* para calcular um índice único ($fluxo\_id$) permitindo que as estatísticas relacionadas a cada fluxo tenham suas informações armazenadas em seus respectivos índices de registro e possam ser recuperadas futuramente para a atualização de informações.

Por fim, a ação $atualiza\_fluxos$ é executada para atualizar as estatísticas de fluxo nos registradores correspondentes conforme os novos pacotes são recebidos. Assim, os dados armazenados de cada fluxo são organizados e encaminhados ao módulo de classificação mediante janelas de observação. A definição das estatísticas a serem coletadas para cada fluxo será apresentada no Capítulo 5.

4.2.2 Classificação

O módulo de classificação busca classificar os fluxos de tráfego de rede dos clientes em legítimos ou maliciosos e atribuir um valor de confiança a cada classificação realizada, com base no comportamento observado nos fluxos de tráfego de cada cliente presente na rede.

Para a realização das tarefas do módulo de classificação, considera-se o uso de técnicas de aprendizagem de máquina, devido ao bom desempenho apresentado por essas técnicas para a classificação de ataques DDoS, conforme observado na revisão do estado da arte. É explorado o fato de os *switches* P4 poderem ser implementados em *switches* caixa branca (*white-box switches*), permitindo a programabilidade do dispositivo sem a necessidade de estar vinculado a um fabricante de *hardware* específico, o que dá mais flexibilidade para o que necessito. Sendo assim, é utilizado um componente auxiliar acoplado ao *switch* P4, que pode acessar e interagir rapidamente com o dispositivo de encaminhamento sem afetar a velocidade de processamento exigida, através do *framework* Apache Thrift [105] para os módulos presentes no plano de dados. Esse *framework* possibilita a comunicação de baixo custo entre diferentes processos via *Remote Procedure Call* (RPC) [106]. A Figura 4.4 ilustra o funcionamento do módulo de classificação presente nos *switches*.

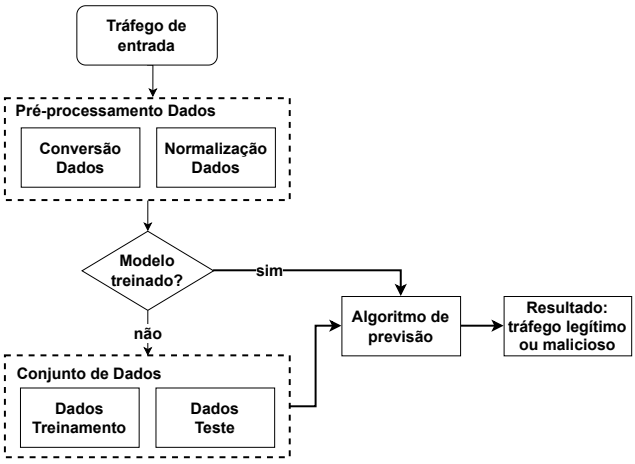


Figura 4.4: Classificação de um ataque DDoS.

Os dados de tráfego de entrada analisados pelo módulo de classificação contêm as características de tráfego brutas que são a base para o cálculo das características que melhor representam o ataque, coletadas pelo módulo de coleta de dados. Ao receber esses dados, que são uma coleção de fluxos e suas respectivas estatísticas agrupadas e organizadas em janelas de observação, eles são encaminhados para a etapa de pré-processamento dos dados. Nesta etapa, um processo de remoção de valores nulos é iniciado, em conjunto com o processo de conversão de dados, para converter dados categóricos em números, por exemplo, tipo de protocolo (TCP, UDP e ICMP), cujo objetivo é torná-los compatíveis com o algoritmo de aprendizagem de máquina a ser empregado. Tal conversão possibilita executar o processo de normalização dos dados, que consiste em ajustá-los para uma escala

ou intervalo comum, geralmente entre 0 e 1 ou -1 e 1, de modo que todos os atributos tenham impacto semelhante no modelo.

Após o pré-processamento, é verificado se existe um modelo de classificação previamente treinado e carregado nos *switches*. Se o modelo estiver disponível, é realizada a previsão (classificação) no *switch* com base em um algoritmo de aprendizagem de máquina supervisionado apresentado na Seção 2.3, por exemplo, KNN [77], SVM [79] ou *Random Forest* [83], para determinar o tipo de tráfego (legítimo ou malicioso). Caso contrário, os dados coletados pelos *switches* são encaminhados para a etapa de treinamento central localizada no plano de controle que realizará o treinamento de um novo modelo a ser disponibilizado para os *switches*. Os dados são utilizados para treinar um novo modelo, dividindo o conjunto de dados em porções de treinamento e teste (geralmente na proporção de 70:30, ou seja, 70% para treinamento e 30% para teste). Para atualizar um modelo de classificação treinado nos *switches*, o modelo anterior é removido para forçar um novo treinamento e carregamento do modelo. A definição do tipo de algoritmo de aprendizagem de máquina para o módulo de classificação será apresentado no Capítulo 5.

Vale ressaltar que a etapa de treinamento é realizada no controlador, quando não houver um modelo treinado disponível nos *switches* ou quando haver necessidade de atualização do modelo. É utilizado um algoritmo de aprendizagem de máquina do tipo supervisionado, dessa maneira, os rótulos correspondentes para o tráfego malicioso e legítimo do conjunto de dados são fornecidos durante a etapa de treinamento do modelo. Os resultados do treinamento serão usados para avaliar o comportamento do modelo (teste) nos *switches*. Para a etapa de teste, os rótulos são retirados para analisar o comportamento do modelo em dados que não foram utilizados durante o treinamento. Após a definição do algoritmo de aprendizagem de máquina, na etapa de treinamento um procedimento de otimização de hiperparâmetros é executado através da função `RandomizedSearchCV` [107], que seleciona aleatoriamente um conjunto de combinações de hiperparâmetros do algoritmo definido, avaliando o desempenho do modelo para cada combinação produzida por meio de uma validação cruzada (*cross-validation*) para encontrar o modelo com a combinação ideal desses valores e, assim, maximizar o desempenho de classificação do modelo, cujo objetivo é evitar o *overfitting* do modelo, ou seja, quando o modelo se ajusta excessivamente aos dados de treinamento, prejudicando o resultado do classificador para dados não vistos [76]. Dessa forma, por meio da função `RandomizedSearchCV` busca-se aumentar a confiabilidade e a precisão do classificador desenvolvido.

A segunda ação do módulo de classificação consiste em atribuir um valor de confiança a cada previsão realizada com base na análise do comportamento do cliente n_j conectado ao *switch* s_i, por meio do método *Platt Scaling* [108, 109]. A escolha desse método é em função da sua capacidade de ajustar as saídas das previsões dos modelos de classificação

por meio de uma regressão logística, transformando as saídas desses modelos em valores probabilísticos entre 0 e 1 que refletem o grau de confiança associado a cada previsão de modo a auxiliar na tomada de decisão. O método é definido pela seguinte equação:

$$C(y = 1|f)_{n_j} = \frac{1}{1 + exp(Af + B)}, \tag{4.1}$$

onde y representa a classe da amostra ($y = 1$: legítimo) e f é a saída não ajustada do classificador utilizado, por exemplo, KNN, SVM ou *Random Forest*. Essa saída é obtida pela combinação de recursos, por meio das variáveis independentes, que são as características dos fluxos disponíveis no conjunto de dados para cada cliente, como a sua duração, o seu tamanho em *bytes*, o total de pacotes encaminhados, etc., que o classificador utiliza para realizar a previsão em malicioso ou legítimo [110]. As variáveis A e B são os parâmetros ajustados para a regressão logística. Eles são determinados através do ajuste do modelo de regressão via máxima verossimilhança, que busca encontrar os valores adequados dos parâmetros, tornando os dados observados prováveis de terem ocorrido com base no modelo treinado [109]. Isso significa, ajustar os parâmetros (A e B) para que as probabilidades previstas pelo modelo sejam as mais próximas possíveis dos rótulos reais da classe da amostra y. Dessa maneira, esses valores são ajustados na fase de treinamento do classificador, de modo que as saídas se aproximem das probabilidades reais observadas nos dados de treino. Assim, quando aplicados na fase de teste, eles possibilitam transformar os resultados (saídas) do modelo de classificação em uma distribuição de probabilidade bem ajustada (valores de confiança) para a classe da amostra em questão, conforme o comportamento do cliente. Portanto, quanto mais próximo de 1 (valor de confiança), maior a indicação de que o fluxo de um cliente pertence à classe da amostra y (legítimo). Por outro lado, valores mais próximos de 0 indicam maior probabilidade de pertencimento à classe oposta (malicioso), caracterizando comportamento potencialmente malicioso.

4.2.3 Agregação do Valor de Confiança

O módulo de agregação do valor de confiança visa agregar os valores de confiança atribuídos aos fluxos de cada cliente durante a execução do módulo de classificação. Essa agregação permite formar um valor de confiança local para cada cliente ao longo do tempo, refletindo seu comportamento local na rede. Além disso, o módulo incorpora um fator de esquecimento, que reduz o valor de confiança local de um cliente caso ele não receba um novo valor de confiança em um determinado intervalo de tempo. Esse fator assegura que clientes inativos ou com comportamentos inconsistentes ao longo do tempo tenham sua confiança ajustada, mantendo o sistema mais dinâmico e atualizado, priorizando aqueles que são mais ativos e com bons comportamentos na rede.

O valor de confiança atribuído para cada fluxo de um cliente presente na rede é agregado pelo módulo de agregação do valor de confiança, resultando no valor de confiança local desse cliente, por meio da seguinte equação:

$$\tau(s_i, n_j)^t = \beta \cdot C_{n_j} + (1 - \beta) \cdot \tau(s_i, n_j)^{t-1}, \tag{4.2}$$

onde C_{n_j} é o valor de confiança calculado pelo método *Platt Scaling* [108] (Equação (4.1)) para o cliente n_j, $\tau(s_i, n_j)^{t-1}$ é o valor de confiança local até o tempo $t-1$ (anterior) que o *switch* s_i possui em relação ao cliente n_j, e β é o coeficiente de suavização que pode assumir valores no intervalo $(0 < \beta \leq 1)$. Um valor de β próximo a 1 dá mais peso aos valores de confiança mais recentes, tornando o modelo mais sensível às mudanças recentes no sistema. Por outro lado, um valor β próximo a 0 dá mais peso aos valores mais antigos, conferindo mais estabilidade ao modelo. Assim, o coeficiente de suavização controla o equilíbrio entre a sensibilidade a mudanças recentes e a estabilidade ao considerar o histórico, permitindo ajustar o modelo de forma eficiente para refletir as dinâmicas do sistema analisado.

O fator de esquecimento atua monitorando o valor de confiança local dos clientes presentes nas listas locais do *switch* em intervalos de tempo regulares. Dessa forma, se o cliente não obtiver um novo valor de confiança dentro do intervalo de tempo definido, o mecanismo aplicará uma redução gradativa do valor de confiança local desse cliente através da seguinte equação:

$$F(s_i, n_j) = \tau(s_i, n_j)^t \cdot \delta, \tag{4.3}$$

onde $\tau(s_i, n_j)^t$ é o valor de confiança local no instante de tempo t que o *switch* s_i possui em relação ao cliente n_j e δ é o coeficiente de redução que pode assumir valores no intervalo de $(0 < \delta \leq 1)$. Um valor de δ próximo a 0 aplica uma redução mais severa no valor de confiança quando não há atualizações recentes. Por outro lado, um valor δ próximo de 1 resulta em uma redução mais suave, permitindo que o valor de confiança diminua lentamente ao longo do tempo, garantindo mais estabilidade para os clientes. Assim, se o cliente for inativo ou tiver comportamentos inconsistentes ao longo do tempo, o seu valor de confiança local tende a cair com o tempo.

O Algoritmo 1 é executado em cada um dos *switches* $s_i \in S$ e mostra o funcionamento do gerenciamento da confiança local nos dispositivos de encaminhamento para os clientes presentes na rede SDN. O cliente n_j terá um valor de confiança C_{n_j} computado para um fluxo de rede conforme a Equação (4.1) após a conclusão da janela de observação w_v, com base nas estatísticas coletadas pelo módulo (coleta de dados) para o respectivo fluxo ao longo dessa janela. Assim, no passo 1, para cada novo valor de confiança C_{n_j} obtido pelo cliente n_j, o *switch* s_i realizará a atualização do seu valor de confiança local

$\tau(s_i, n_j)^t$ em relação ao cliente n_j no instante de tempo t por meio da Equação (4.2) para formar o valor de confiança local desse cliente ao longo do tempo, representando o seu comportamento local na rede, permitindo acompanhar a evolução de conduta do cliente em questão à medida que os seus novos fluxos são analisados e classificados pelo *switch* s_i. O passo 2 está relacionado ao fator de esquecimento. Assim, durante o intervalo de tempo de monitoramento Tm, onde Tm representa o período (intervalo) em que se verifica se o cliente n_j obteve ou não um novo valor de confiança C_{n_j} para um fluxo de rede. Em caso de ausência de um novo valor de confiança C_{n_j} durante o período de monitoramento, o *switch* s_i reduzirá o valor de confiança local $\tau(s_i, n_j)^t$ que possui em relação ao cliente n_j por meio da Equação (4.3). Essa ação reduzirá gradativamente ao longo do tempo o valor de confiança local do cliente, promovendo uma avaliação dinâmica e permitindo que o *switch* s_i ajuste de forma contínua a confiança local atribuída ao cliente conforme a sua atividade na rede, refletindo com maior clareza o comportamento atual do cliente.

Algoritmo 1: Gerenciamento da Confiança Local do *switch* $s_i \in S$

Entrada: Valor de confiança C_{n_j} calculado para o fluxo de rede do cliente n_j por meio da Equação (4.1) após a conclusão da janela de observação w_v, com base nas estatísticas coletadas para o respectivo fluxo ao longo dessa janela.
Saída: Valor de confiança local $\tau(s_i, n_j)^t$ no instante de tempo t ajustado pelo *switch* s_i para o cliente n_j.
Passo 1 - Atualização: Para cada novo valor de confiança C_{n_j} calculado para o cliente n_j, atualize o seu valor de confiança local $\tau(s_i, n_j)^t$ por meio da Equação (4.2).
Passo 2 - Redução: Na ausência de um novo valor de confiança C_{n_j} pelo cliente n_j durante o intervalo de tempo de monitoramento Tm, reduza o seu valor de confiança local $\tau(s_i, n_j)^t$ por meio da Equação (4.3).

Por fim, vale destacar que os parâmetros β, δ e Tm presentes no módulo de agregação do valor de confiança são ajustáveis, o que permite uma maior flexibilidade para o mecanismo proposto. O parâmetro β possibilita ajustar o coeficiente de suavização para a agregação do valor de confiança local, enquanto δ permite ajustar a taxa de redução para o fator de esquecimento e Tm o intervalo de tempo para monitorar se um cliente recebeu ou não um novo valor de confiança. A definição dos valores para esses parâmetros será apresentada no Capítulo 5.

4.2.4 Gerenciamento das Listas

O módulo de gerenciamento das listas é responsável por direcionar o fluxo de tráfego de rede dos clientes às listas locais do *switch*, conforme os seus comportamentos (valor de confiança) na rede, para dar preferência (priorização) ao tráfego dos clientes legítimos e bloquear (descartar) o tráfego malicioso. Essa estratégia de mitigação garante maior dis-

ponibilidade da rede para os clientes legítimos, bem como a redução do encaminhamento de tráfego malicioso ao controlador, garantindo maior disponibilidade e qualidade de serviço para os clientes legítimos. O fluxo de tráfego de rede dos clientes pode ser gerenciado em três listas locais distintas presentes no *switch* P4, nas quais possuem diferentes ações conforme o comportamento dos clientes, como mostra a Figura 4.5.

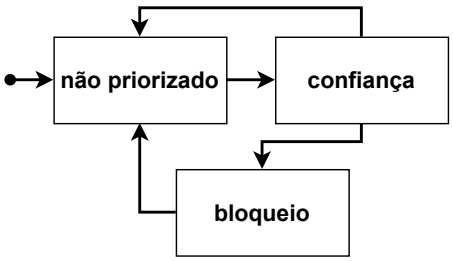


Figura 4.5: Transição dos fluxos de rede entre as listas presentes nos *switches*.

O fluxo de tráfego de rede de um cliente ainda não classificado pelo módulo de classificação, assim como o fluxo de um cliente classificado como legítimo, mas que ainda não atingiu um determinado nível de confiança, é categorizado como não priorizado. Os fluxos desses clientes são adicionados à lista denominada "não priorizado", e encaminhados aos seus destinos sem nenhuma prioridade associada quando gerenciado pelo *switch* P4. À medida que os clientes se mantêm ativos na rede e com bons níveis de confiança, os seus fluxos de tráfegos podem ser gerenciados pela lista de prioridade denominada "confiança", caso os valores de confiança dos clientes estejam acima do limiar de confiabilidade estabelecido para a lista de prioridade. Esta lista proporciona tratamento preferencial (priorização) ao fluxo de tráfego do cliente classificado como legítimo, proporcionando maior velocidade no processamento de encaminhamento dos pacotes de rede.

Conforme o comportamento dos clientes altera ao longo do tempo, os seus fluxos podem ser gerenciados novamente pela lista de não priorizado, se estiverem abaixo do limiar de confiabilidade estabelecido, perdendo assim a prioridade associada anteriormente (característica exclusiva dos clientes pertencentes a lista de confiança), ou serem bloqueados quando apresentarem comportamentos maliciosos que resultam em uma classificação de tráfego malicioso, sendo gerenciados agora pela lista denominada "bloqueio".

A lista de bloqueio trata dos fluxos de tráfegos de clientes classificados como maliciosos. Assim, os clientes gerenciados por essa lista, têm o seu tráfego descartado para prevenir possíveis ataques ou comportamentos que comprometam a disponibilidade dos recursos da rede. É adotado um bloqueio temporário para evitar a penalização indevida de clientes legítimos, garantindo que o bloqueio seja mantido para aqueles que, de fato, exercem

atividades maliciosas. Desta forma, ao ser adicionado à lista de bloqueio, o cliente tem seu fluxo de tráfego de rede descartado temporariamente e, após o término do bloqueio, pode ser realocado para a lista (não priorizado), onde seus novos fluxos serão atendidos sem qualquer prioridade associada, permitindo uma reavaliação do seu comportamento na rede.

Nota-se que a lista de prioridade (confiança) requer um limiar de confiabilidade para que os fluxos de tráfego de rede dos clientes legítimos sejam priorizados em relação ao tráfego dos demais clientes, definindo a sensibilidade do mecanismo proposto. Assim, cada *switch* $s_i \in S$ define inicialmente o seu valor de confiança médio com base nos clientes legítimos que estão sob sua gerência. Os clientes legítimos são representados pelo conjunto $L = \{n'_1, n'_2, \cdots, n'_z\}$, onde $z \leq |N|$, um subconjunto de N composto pelos clientes considerados legítimos. A confiança média de clientes legítimos é definida como:

$$Cm = \frac{1}{|L|} \sum_{j=1}^{|L|} \tau(s_i, n'_j)^t, \tag{4.4}$$

onde Cm é o valor médio de confiança local dos clientes legítimos conectados ao *switch* s_i, $|L|$ o número de clientes legítimos do conjunto L e $\tau(s_i, n'_j)^t$ o valor de confiança local atribuído pelo *switch* s_i ao cliente legítimo n'_j no instante t. Em seguida, o *switch* s_i estabelece o limiar da sua lista de confiança por meio da seguinte equação:

$$Th(s_i)^t = \alpha \cdot Cm + (1 - \alpha) \cdot Th(s_i)^{t-1}, \tag{4.5}$$

onde $Th(s_i)^t$ é o limiar, Cm é a confiança média local calculada pela Equação (4.4) e $Th(s_i)^{t-1}$ é o limiar de confiança calculado anteriormente, e α é um coeficiente de suavização. O coeficiente α assume valores no intervalo $(0 < \alpha \leq 1)$. Um valor de α mais alto implica um peso maior em valores recentes, em contraste com os valores calculados anteriores em $Th(s_i)^t$. No entanto, um valor de α mais baixo implica em um peso maior nos valores anteriores de $Th(s_i)^t$, em contraste com o valor médio atual de confiança. Assim, o limiar pode ser ajustado conforme as características da rede.

O Algoritmo 2 é executado em cada um dos *switches* $s_i \in S$ e mostra o funcionamento do gerenciamento das listas locais nos dispositivos de encaminhamento para coordenar os clientes conectados à rede SDN. Para alocar o cliente n_j a uma das listas locais do *switch* s_i, utiliza-se o valor de confiança local $\tau(s_i, n_j)^t$ atribuído a esse cliente no instante t pelo *switch* s_i, computado por meio da Equação (4.2), tendo como base os valores de confiança C_{n_j} obtidos durante a classificação dos seus fluxos de rede pelo módulo (classificação). Primeiro, é calculado o limiar de confiança $Th(s_i)^t$ para a lista de confiança do *switch* s_i por meio da Equação (4.5), considerando os valores de confiança locais dos clientes classificados como legítimos pelo *switch* s_i. Assim, se o valor de confiança local

$\tau(s_i, n_j)^t$ do cliente n_j pertencente à lista de não priorizado estiver acima do limiar $Th(s_i)^t$, ele é promovido à lista de confiança para ter o seu tráfego priorizado em relação ao tráfego dos clientes presentes na lista de não priorizado. Essa promoção indica que o cliente demonstrou um comportamento confiável, atendendo aos critérios estabelecidos para a priorização do seu tráfego na rede, como ser classificado como legítimo e manter um histórico consistente de atividades sem indícios de comportamentos maliciosos. No entanto, a permanência na lista de confiança não é definitiva, pois o valor de confiança local $\tau(s_i, n_j)^t$ do cliente n_j continua sendo atualizado a medida que os seus novos fluxos são classificados na rede pelo módulo (classificação). Caso o cliente n_j pertencente à lista de confiança apresente mudanças em seu comportamento, como variações no padrão de tráfego ou sinais de possíveis atividades suspeitas, seu valor de confiança C_{n_j} pode diminuir afetando o seu valor de confiança local $\tau(s_i, n_j)^t$, e ele pode ser movido para à lista de não priorizado se o valor de confiança local $\tau(s_i, n_j)^t$ for igual ou inferior ao limiar de confiança $Th(s_i)^t$. Por fim, se o cliente n_j for classificado como malicioso pelo módulo (classificação), ou seja, o seu fluxo de tráfego de rede apresenta características associadas a ataques, ele será adicionado à lista de bloqueio e permanecerá bloqueado por um determinado período de tempo Ψ. Assim, o cliente n_j relacionado à lista de bloqueio tem o seu tráfego de rede descartado de forma imediata, mantendo assim o serviço para aqueles que atendem ao nível de confiança estabelecido (clientes legítimos) para o mecanismo proposto.

Por último, vale ressaltar que os parâmetros α e Ψ presentes no módulo de gerenciamento das listas são ajustáveis, permitindo uma configuração mais personalizada para o mecanismo proposto. O parâmetro α permite ajustar o peso dado aos valores de confiança locais para a definição do limiar de confiança da lista de confiança, enquanto Ψ é o tempo de bloqueio que estará associado ao cliente adicionado a lista de bloqueio. A definição dos valores para esses parâmetros será apresentada no Capítulo 5.

4.2.5 Gerenciador de Envio

O módulo (gerenciador de envio) é responsável por encaminhar ao plano de controle o valor mais recente de confiança local dos clientes presentes nas listas locais do *switch*, por meio de pacotes personalizados, chamado de pacote de relatório, organizados em janelas de observação.

Os valores de confiança local dos clientes são calculados ao longo das suas participações na rede por meio da Equação (4.2), para refletir o comportamento local de cada um na rede. Esses valores são incluídos em um pacote de relatório, que consiste em um pacote normal, cuja carga útil é removida e cujo cabeçalho padrão é personalizado, para reduzir o

Algoritmo 2: Gerenciamento das listas locais do *switch* $s_i \in S$

Entrada: Valor de confiança local $\tau(s_i, n_j)^t$, que o *switch* s_i possui em relação ao cliente n_j no instante de tempo t, calculado por meio da Equação (4.2), obtido com base nos valores de confiança C_{n_j} associados aos fluxos de rede classificados desse cliente.

Saída: Alocação do cliente n_j a lista local do *switch* s_i conforme a sua classificação e valor de confiança local $\tau(s_i, n_j)^t$.

Passo 1 - Limiar de Confiança: Calcule o limiar de confiança local $Th(s_i)^t$ da lista de confiança por meio da Equação (4.5).

Passo 2 - Lista de Não Priorizado: Para o cliente n_j que pertença à lista de não priorizado, compare o seu valor de confiança local $\tau(s_i, n_j)^t$ com o limiar de confiança $Th(s_i)^t$ da lista de confiança.

- **Passo 2.1:** Se $\tau(s_i, n_j)^t > Th(s_i)^t$, então adicione o cliente n_j a lista de confiança e remova-o da lista de não priorizado.

Passo 3 - Lista de Confiança: Para o cliente n_j que pertença à lista de confiança, compare o seu valor de confiança local $\tau(s_i, n_j)^t$ com o limiar de confiança $Th(s_i)^t$.

- **Passo 3.1:** Se $\tau(s_i, n_j)^t \leq Th(s_i)^t$, então adicione o cliente n_j a lista de não priorizado e remova-o da lista de confiança.

Passo 4 - Lista de Bloqueio: Para o cliente n_j classificado como malicioso pelo *switch* s_i, adicione o cliente em questão à lista de bloqueio e defina o seu tempo de bloqueio Ψ.

seu tamanho e garantir que o foco seja nas informações de controle a serem encaminhadas ao controlador, como mostra a Figura 4.6.

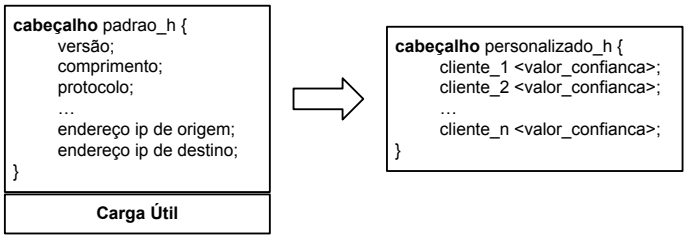


Figura 4.6: Transformação do cabeçalho de um pacote normal em um cabeçalho personalizado.

Para armazenar os valores de confiança local dos clientes presentes na rede, é realizada a cópia de um pacote normal presente no *switch*, pois devido a limitações no dispositivo de encaminhamento programável, a geração espontânea de pacotes não é possível. Para resolver essa limitação, foi utilizada a função *clone* da linguagem P4, que permite fazer uma cópia de um pacote no *switch* e, em seguida, encaminhá-lo com as alterações aplicadas [104]. Dessa maneira, o pacote original não é afetado pela clonagem, seguindo o caminho inicial, enquanto a cópia é modificada para refletir as novas atualizações e ser

enviada ao controlador. Após a cópia, o cabeçalho padrão desse novo pacote é personalizado, substituindo-o pelas novas informações, que agora incluem os valores de confiança local computados pelos *switches* para os clientes. Esse novo cabeçalho é estruturado de forma a incluir a identificação de cada cliente (endereço IP) e o valor de confiança local de cada um ($\tau(s_i, n_j)^t$), permitindo que as informações sejam processadas de maneira eficiente pelo controlador.

Foram utilizadas janelas de observação para organizar e otimizar o fluxo de coleta e envio do pacote de relatório com os valores de confiança local dos clientes, assegurando que esse pacote seja enviado com a frequência necessária e priorizada, mas sem sobrecarregar o sistema de comunicação entre o plano de dados e o plano de controle. O uso dessas janelas possibilita que as informações sejam atualizadas e enviadas regularmente, mantendo o plano de controle informado sobre o estado atual de cada dispositivo presente no plano de dados.

O Algoritmo 3 é executado em cada um dos *switches* $s_i \in S$ e mostra o funcionamento do módulo (gerenciador de envio) para que os dispositivos de encaminhamento possam encaminhar ao controlador o valor mais recente de confiança local dos clientes presentes na rede SDN. O algoritmo recebe como entrada, o valor de confiança local $\tau(s_i, n_j)^t$ de cada um dos clientes $n_j \in N$ geridos pelo *switch* s_i em suas listas locais ao longo da janela de observação w_v para o módulo (gerenciador de envio). O passo 1 consiste em realizar a cópia de um pacote normal presente no *switch* s_i, onde o cabeçalho original do pacote clonado é modificado para formar um novo cabeçalho personalizado, contendo a identificação dos clientes (endereço IP) e o valor de confiança local $\tau(s_i, n_j)^t$ de cada um deles gerido pelo *switch* s_i. Os dados referentes à confiança local dos clientes são agregados em um único pacote de relatório r, respeitando os limites do MTU (*Maximum Transmission Unit*) da rede, para evitar que ele seja fragmentado, minimizando atrasos, melhorando a eficiência da rede e otimizando a transmissão das informações. Ao alcançar esse limite, um novo pacote é clonado para continuar a inclusão dos dados restantes em um novo pacote de relatório. Assim, em um pacote de relatório r temos um grupo de clientes e seus respectivos valores de confiança local $\tau(s_i, n_j)^t$ a serem reportados. O passo 2 consiste no envio do pacote de relatório r com as estatísticas dos clientes para o controlador Cr ao final de cada janela w_v (seja ela por tempo ou por pacote). Vale ressaltar que, no caso de janelas baseadas em pacotes, ou seja, de tamanho fixo, os dados podem ser enviados ao controlador quando a janela atingir o tempo limite predefinido para o preenchimento dos dados, no qual os dados acumulados são enviados, mesmo que a janela não esteja completamente cheia. Esse processo possibilita uma transmissão estruturada e eficiente das informações relacionadas a confiança local dos clientes ao controlador para a definição da confiança global e a classificação global, que serão apresentadas na próxima seção. Em

cada janela de observação, o mecanismo deve ser capaz de continuar coletando os dados e, ao mesmo tempo, enviar os dados coletados na janela de observação anterior para o controlador.

Algoritmo 3: Gerenciador de Envio do *switch* $s_i \in S$

Entrada: Valor de confiança local $\tau(s_i, n_j)^t$, para cada um dos clientes $n_j \in N$ coletados das listas locais do *switch* $s_i \in S$ ao longo da janela de observação w_v.
Saída: Pacote de relatório r, encaminhado ao controlador Cr pelo *switch* s_i contendo os valores de confiança local $\tau(s_i, n_j)^t$ atribuído aos seus clientes.
Passo 1 - Clone e Agregação:

- **Passo 1.1:** Clona um pacote normal do *switch* s_i e personaliza o seu cabeçalho original e adiciona o valor de confiança local $\tau(s_i, n_j)^t$ dos clientes geridos pelo *switch* s_i nesse cabeçalho personalizado.
- **Passo 1.2:** Agrega os dados dos clientes em um único pacote de relatório r até o limite do MTU da rede.

Passo 2 - Envio: Envia o pacote de relatório r para o controlador Cr ao final da janela de observação w_v.

Por fim, vale salientar que o tipo de janela de observação w_v (por exemplo, baseada em tempo ou em pacotes) e o tempo limite, tempo este exclusivo para o preenchimento das janelas baseadas em pacotes, são ajustáveis, permitindo uma flexibilidade na configuração do mecanismo proposto. A definição do tipo de janela e o tempo limite para o preenchimento dos dados será apresentada no Capítulo 5.

4.3 Procedimento 2: compartilhamento de informações globais por meio do plano de controle

Esta seção detalha os módulos presentes no procedimento 2 do mecanismo proposto. Os objetivos desse procedimento consistem em definir, no plano de controle, a confiança global dos clientes presentes nas listas locais dos *switches* e realizar o compartilhamento das informações globais, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização desses clientes, determinadas a partir do estabelecimento dessa confiança global, por meio do controlador para o envio aos dispositivos de encaminhamento presentes no plano de dados, permitindo a tomada de decisão local com base nas informações globais compartilhadas, promovendo uma abordagem híbrida e abrangente para o mecanismo.

4.3.1 Extração dos Dados

O módulo de extração dos dados busca extrair os cabeçalhos dos pacotes de relatórios para obter o valor mais recente de confiança local de cada cliente computados pelos *switches* e organizá-los em uma estrutura de dados.

Para a realização das ações do módulo de extração dos dados, é importante compreender como os valores de confiança local dos clientes são extraídos e organizados para análise. A Figura 4.7 ilustra o funcionamento do módulo, destacando as etapas de extração, organização e envio dos dados, permitindo uma integração eficiente com os demais módulos do procedimento 2.

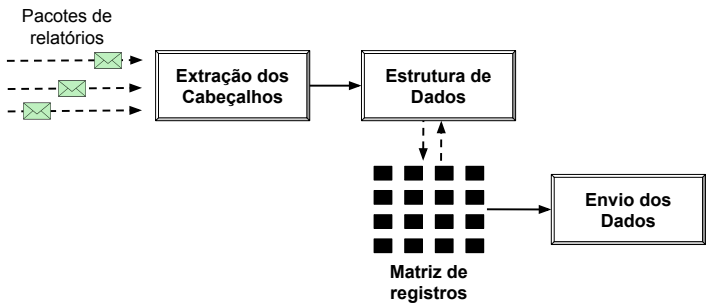


Figura 4.7: Extração dos Dados.

Os pacotes de relatórios encaminhados pelo módulo gerenciador de envio contêm o valor de confiança local mais recente de cada cliente computados pelos *switches*, que servirão como base para o cálculo da confiança global de cada um na rede, etapa que será descrita mais adiante na Subseção 4.3.2. Ao receber esses valores, o módulo extrai o cabeçalho personalizado dos pacotes de relatórios para obter as informações dos clientes. Essas informações são organizadas em uma estrutura de dados (matriz de registros), onde cada linha corresponderá a um cliente e cada coluna, às suas estatísticas específicas, que são os valores de confiança local coletados dos *switches* no qual o cliente obteve um novo valor de confiança local. A estrutura utiliza índices únicos, para mapear cada cliente a um índice específico na matriz, garantindo acesso rápido e eficiente aos valores de confiança local coletados. Por fim, os dados armazenados são organizados e enviados ao módulo (calcular a confiança global) em intervalos predefinidos.

A definição do intervalo de tempo para o envio dos dados referentes aos valores de confiança dos clientes ao módulo calcular a confiança global será apresentada no Capítulo 5.

4.3.2 Calcular a Confiança Global

O módulo (calcular a confiança global) é responsável por determinar a confiança global dos clientes da rede, refletindo a confiabilidade geral de cada um e, assim, classificando-os em "Não Confiável", "Parcialmente Confiável" e "Confiável", permitindo que os dispositivos de encaminhamento, presentes no plano de dados, tomem decisões locais com base em informações globais compartilhadas.

Combinar os diferentes valores de confiança local produzidos pelos clientes da rede para representar uma confiança global apresenta alguns desafios, como a dinâmica da rede, no qual o comportamento de um cliente pode variar ao longo do tempo, exigindo uma abordagem adaptativa que leve em conta mudanças recentes sem desconsiderar o histórico acumulado. Além disso, é necessário garantir que o processamento dos dados seja rápido e com um baixo custo computacional, permitindo a tomada de decisões em tempo real, sem sobrecarregar os recursos da rede. Isso requer estratégias eficientes capazes de lidar com esses dados, mantendo a precisão e a eficácia na consolidação das informações, sem introduzir latência significativa que prejudique o desempenho geral do mecanismo proposto.

A estratégia adotada por esse trabalho para superar os desafios listados anteriormente foi a utilização da lógica *fuzzy*, proposta por Lotfali Askar-Zadeh [102], que permite lidar com a incerteza e a variabilidade de um conjunto de dados, modelando os resultados de forma mais próxima à percepção humana. Essa abordagem possibilita definir a confiança global dos clientes de maneira flexível, considerando as nuances e dinâmicas da rede. Com isso, torna-se possível gerar uma classificação global mais precisa dos clientes e tomar decisões com base na confiança global atribuída a cada um, aprimorando as ações de mitigação e priorização na rede do mecanismo proposto.

A lógica *fuzzy* é um sistema de lógica que estende a lógica clássica ao lidar com graus de verdade, em vez de valores binários [102]. Ao invés de classificar elementos como pertencentes ou não a um conjunto (verdadeiro ou falso), ela permite que elementos tenham um grau de pertencimento aos conjuntos *fuzzy*, representados por valores no intervalo entre 0 e 1, com base nas funções de inferência usadas, em que 0 significa que um elemento não pertence a determinado conjunto, 1 significa completa pertinência ao conjunto, e valores entre 0 e 1 representam graus parciais de pertinências [111]. Assim, na lógica *fuzzy*, um elemento pode pertencer a um conjunto com um certo grau de pertinência. Essa abordagem permite modelar incertezas e situações que envolvem variabilidade ou imprecisão, aproximando o raciocínio computacional ao raciocínio humano.

O sistema *fuzzy* desenvolvido para o mecanismo proposto possui as seguintes etapas [111]:

- **Fuzzificação:** os dados a serem analisados pelo mecanismo são transformados em uma variável linguística, que é utilizada no sistema de inferência disponível;
- **Sistema de Inferência:** a variável linguística proveniente do processo de fuzzificação é aplicada a um conjunto de regras, produzindo uma variável linguística relacionada à saída da inferência;
- **Defuzzificação:** utiliza a variável linguística proveniente do sistema de inferência e a converte em um valor nítido (*crisp*), conforme a estratégia de defuzzificação que está sendo utilizada.

Na etapa de fuzzificação, os valores de confiança local dos clientes, provenientes do módulo de extração dos dados, são organizados em uma variável linguística denominada "confiança local". Uma variável linguística representa uma variável cujo valor não é numérico, mas sim descrito por termos linguísticos [112]. Dessa forma, os valores de confiança locais são organizados nos seguintes termos linguísticos: "alto", "médio" e "baixo", representando o conjunto *fuzzy* de entrada denominado "Confiança Local", tendo como base os valores produzidos pelo módulo (agregação do valor de confiança) para cada cliente presente na rede. Cada termo é associado a uma função de pertinência que define seu grau de associação em um intervalo contínuo, entre 0 e 1.

A função de pertinência μ é responsável por definir o grau de associação de um valor de entrada ao conjunto *fuzzy* [102, 113]. Essa função mapeia os valores de confiança local de cada cliente para um intervalo contínuo de valores entre 0 e 1. O valor retornado pela função indica o nível de pertencimento do elemento ao conjunto *fuzzy*, em que 0 significa que não pertence, 1 pertence completamente e um valor entre 0 e 1 pertence parcialmente com diferentes níveis de intensidade. Na literatura, há diferentes tipos de funções de pertinência para a análise dos conjuntos *fuzzy*, como [114]:

- **Triangular:** a função triangular tem três parâmetros: a, b e m. Sendo a onde a função começa, b onde ela termina e m o valor de x para o máximo grau de pertinência, com valor $\mu(x) = 1$. A função é dada pela seguinte equação:

$$\mu(x) = \begin{cases} 0 & \text{se } x \leq a \\ (x-a)/(m-a) & \text{se } x \in [a, m] \\ (b-x)/(b-m) & \text{se } x \in [m, b] \\ 0 & \text{se } x \geq b. \end{cases} \tag{4.6}$$

- **Trapezoidal:** a função trapezoidal tem quatro parâmetros: a, b, m e g. Sendo a o primeiro ponto da função, b o último ponto onde $\mu(x)$ é 0, e os parâmetros m e

g representam o intervalo de pontos onde $\mu(x)$ tem valor 1, representando o grau máximo de pertinência. A função é dada pela seguinte equação:

$$\mu(x) = \begin{cases} 0 & \text{se } x \leq a \\ (x-a)/(m-a) & \text{se } x \in [a, m] \\ 1 & \text{se } x \in [m, g] \\ (b-x)/(b-m) & \text{se } x \in [g, b] \\ 0 & \text{se } x \geq b. \end{cases} \tag{4.7}$$

- **Gaussiana:** a função gaussiana tem dois parâmetros: c e σ. Sendo c o centro (valor onde $\mu(x)$ é 1) e σ é o desvio padrão, responsável por controlar a largura da curva. A função é dada pela seguinte equação:

$$\mu(x) = e^{-\frac{(x-c)^2}{2\sigma^2}}. \tag{4.8}$$

A definição do tipo de função de pertinência e seus parâmetros para o conjunto *fuzzy* de entrada para o sistema proposto exige uma análise cuidadosa dos valores de confiança locais produzidos pelo módulo de agregação do valor de confiança, presente no plano de dados, no qual devem ser levadas em consideração as características dos dados, o comportamento do tráfego de rede observado e o custo computacional associado a cada função. Assim, a função e seus parâmetros precisam ser definidos com base na análise de dados históricos da rede coletados em um ambiente controlado, possibilitando definir o tipo de função e os parâmetros que melhor representam as características do conjunto de dados, representado pelos valores de confiança local dos clientes para o sistema proposto. A escolha do tipo de função e de seus parâmetros será apresentada no Capítulo 5.

Após realizar o processo de fuzzificação, o conjunto de valores da variável linguística (valores de confiança local de cada cliente) são aplicados a um conjunto de regras para serem submetidos ao sistema de inferência. As regras em um sistema *fuzzy* desempenham o papel de modelar a lógica por trás das decisões, baseando-se na forma como os valores de entrada (valores de confiança locais) interagem entre si. Essas regras são estruturadas em um formato do tipo SE-ENTÃO, em que uma ou mais condições de entrada, chamadas de antecedentes, são avaliadas para produzir uma ação ou resultado correspondente, chamado de consequente [115].

Assim, um conjunto de regras do tipo SE-ENTÃO é definido para o mecanismo proposto, para que o sistema *fuzzy* analise o conjunto de valores de confiança local de cada cliente e realize a associação com as regras definidas para o modelo de avaliação (sistema de inferência). As regras para o sistema *fuzzy* desenvolvido, foram definidas com base em

conhecimento especializado sobre o sistema proposto, considerando as características da rede, como os padrões de comportamentos dos clientes, de modo a categorizar o valor de confiança global do cliente em "Não Confiável", "Parcialmente Confiável" e "Confiável" para capturar nuances e variações no ambiente de rede, seguindo o modelo: SE valor de confiança é Baixo, ENTÃO o cliente é Não Confiável; SE valor de confiança é Médio, ENTÃO o cliente é Parcialmente Confiável; SE valor de confiança é Alto, ENTÃO o cliente é Confiável. Assume-se que o cliente pode assumir qualquer valor de confiança dentro de um intervalo contínuo, entre 0 e 1, permitindo uma avaliação mais granular de seu comportamento na rede.

Após definir o conjunto de regras, o sistema de inferência irá processar e ativar as regras estabelecidas com um grau máximo e mínimo de pertinência para cada regra, conforme o conjunto de valores de confiança local analisados para cada cliente, para que o modelo transforme a variável linguística de entrada, ou seja, os valores de confiança locais em uma variável linguística de saída que caracteriza o valor de confiança global do cliente correspondente a uma função de pertinência de saída (por exemplo, triangular, trapezoidal ou gaussiana) definida e parametrizada para o sistema proposto, com base na análise de dados históricos do comportamento do tráfego de rede em um ambiente controlado, para formar o conjunto *fuzzy* de saída denominado "Confiança Global" [116]. A variável linguística para o conjunto *fuzzy* de saída é organizada nos seguintes termos linguísticos: "Não Confiável", "Parcialmente Confiável" e "Confiável", para refletir os diferentes níveis de confiança global que podem ser associados a um cliente da rede para o mecanismo proposto.

A etapa final do sistema *fuzzy* é o processo de defuzzificação, que consiste em traduzir os valores da variável de saída inferida pelo conjunto de regras em um valor *crisp*, que corresponde a um único valor que melhor represente os valores *fuzzy* inferidos da variável linguística de saída, associada a um dos termos linguísticos propostos para o conjunto *fuzzy* de saída (Confiança Global) [116]. Esse valor corresponde a confiança global do cliente na rede. A literatura dispõe de diferentes tipos de métodos de defuzzificação, como [114]:

- **Centro de Gravidade:** método que calcula o ponto de equilíbrio de um conjunto *fuzzy*, ponderando os valores de saída com base na área sob a curva de pertinência. Dessa forma, o valor *crisp* é obtido calculando a média ponderada de todos os pontos da função de pertinência.

- **Máxima Pertinência:** método que consiste em determinar o valor *crisp* que corresponde ao ponto em que a função de pertinência atinge seu valor máximo.

- **Média Ponderada dos Máximos:** método que produz um valor considerando a média ponderada dos valores centrais ativados, onde os pesos são os graus de pertinência máximo de cada variável linguística de saída.

A definição do método para o processo de defuzzificação do sistema proposto requer uma análise cuidadosa do custo computacional associado, considerando que estamos lidando com um controlador SDN, dispositivo que possui memória e capacidade de processamento limitadas [117]. Dessa forma, é essencial que o método a ser escolhido seja capaz de combinar precisão e rapidez com um custo computacional reduzido, garantindo a eficiência no processamento dos valores de confiança locais sem comprometer o desempenho do sistema ou a integridade dos resultados. A escolha do tipo de método de defuzzificação para o mecanismo proposto será apresentada no Capítulo 5.

Por fim, com o valor de confiança global calculado para cada cliente, o mecanismo proposto pode determinar a ação de controle a ser tomada que pode incluir o bloqueio, a permissão sem prioridade associada ou a priorização do fluxo de tráfego de rede desse cliente, com base na classificação do valor de confiança global obtido. Os detalhe dessas ações de controle relacionadas ao valor de confiança global do cliente estão descritas na subseção 4.3.3.

4.3.3 Disseminação dos Dados

O módulo de disseminação dos dados é responsável por enviar aos *switches* as ações de controle referentes à confiabilidade global de cada cliente, por meio de pacotes personalizados, chamado de pacote de disseminação, organizados em janelas de observação.

As ações de controle referentes ao valor de confiança global de cada cliente definido pelo sistema *fuzzy*, correspondem as medidas a serem tomadas pelo mecanismo proposto para aplicar a política de restrição ou priorização do fluxo de tráfego de rede dos clientes de maneira global. As ações incluem bloquear (descartar) o fluxo de tráfego de rede para o cliente classificado como "Não Confiável", permitir o fluxo de tráfego de rede daquele classificado como "Parcialmente Confiável", porém sem prioridade associada, ou priorizar o fluxo de tráfego de rede para aquele classificado como "Confiável". Essas ações são incluídas em um pacote de disseminação, cuja estrutura de cabeçalho é personalizada para atender às demandas do mecanismo proposto, como mostra a Figura 4.8.

Para armazenar as ações de controle referentes a cada cliente de rede analisado, é criado um pacote a ser enviado pelo controlador para os dispositivos de encaminhamento, que contém a identificação dos clientes (endereço IP) e as respectivas ações de controle para cada um. Esse formato permite que as informações sejam processadas de maneira eficiente

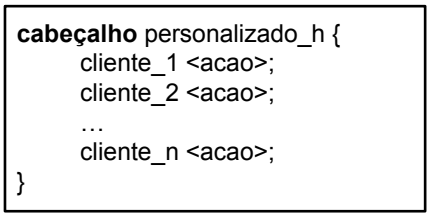


Figura 4.8: Cabeçalho de um pacote de disseminação.

pelos dispositivos de encaminhamento, garantindo uma resposta ágil e coordenada às condições dinâmicas da rede.

Foram utilizadas janelas de observação para a organização e envio dos pacotes de disseminação. Ao realizar a disseminação das informações relacionadas a política de restrição ou priorização do fluxo de tráfego de rede dos clientes, procura-se fazer com que os dispositivos de encaminhamento possam tomar decisões localmente com base nas ações informações globais compartilhadas, promovendo uma abordagem coordenada, abrangente e rápida para conter o ataque DDoS em toda a rede.

O Algoritmo 4 é executado no controlador Cr e mostra o funcionamento do módulo (disseminação dos dados) para enviar aos *switches* as ações de controle dos clientes presentes na rede SDN. O algoritmo recebe como entrada a ação de controle de cada um dos clientes $n_j \in N$, definida com base no valor de confiança global de cada um, estabelecido pelo sistema *fuzzy* presente no módulo (calcular a confiança global), com dados coletados ao longo da execução da janela de observação w_v para o módulo de disseminação dos dados. O passo 1 consiste em criar o pacote de disseminação d por meio do controlador Cr, cujo cabeçalho desse pacote é personalizado, contendo a identificação de cada cliente (endereço IP) e as suas respectivas ações de controle definidas pelo sistema *fuzzy*. Os dados referentes às ações de controle são agregados em um único pacote de disseminação d, respeitando os limites do MTU da rede, a fim de evitar que ele seja fragmentado e cause atrasos. Ao alcançar esse limite, um novo pacote de disseminação é criado para continuar a inclusão dos dados restantes. Assim, em um pacote de disseminação d temos um grupo de clientes e suas respectivas ações de controle, a serem enviadas. O passo 2 consiste no envio do pacote de disseminação d para os *switches* ao final de cada janela w_v (seja ela por tempo ou por pacote). Para o caso de janelas baseadas em pacotes, ou seja, de tamanho fixo, os dados podem ser enviados aos *switches* quando a janela atingir o tempo limite predefinido para o preenchimento dos dados, no qual os dados acumulados são enviados, mesmo que a janela não esteja completamente cheia para possibilitar uma transmissão estruturada e eficiente das informações relacionadas as ações de controle dos clientes. Em cada janela de observação, o mecanismo deve ser capaz de continuar organizando os dados

e, ao mesmo tempo, enviar os dados organizados na janela de observação anterior para os *switches*. Assim, esses dispositivos poderão aplicar as ações de controle recomendadas contidas em cada pacote de disseminação, proporcionando uma operação mais eficiente e coordenada por parte do mecanismo proposto.

Algoritmo 4: Disseminação dos Dados realizada pelo controlador Cr

Entrada: Ação de controle para cada um dos clientes $n_j \in N$ definida com base no valor de confiança global estabelecido pelo sistema *fuzzy*, coletadas ao longo da janela de observação w_v.

Saída: Pacote de disseminação d, enviado aos *switches* pelo controlador Cr, contendo as ações de controle de cada cliente.

Passo 1 - Criar e Agregar:

- **Passo 1.1:** Cria um pacote de disseminação d com cabeçalho personalizado por meio do controlador Cr e adiciona as ações de controle para os clientes ao cabeçalho personalizado.
- **Passo 1.2:** Agrega os dados dos clientes em um único pacote de disseminação d até o limite do MTU da rede.

Passo 2 - Envio: Envia o pacote de disseminação d para os *switches* ao final da janela de observação w_v.

Por fim, cabe ressaltar que o tipo de janela de observação w_v (por exemplo, baseada em tempo ou em pacotes) e o tempo limite, tempo este exclusivo para o preenchimento das janelas baseadas em pacotes, são ajustáveis, permitindo uma versatilidade na configuração do mecanismo proposto. A definição do tipo de janela e o tempo limite para o preenchimento dos dados será apresentada no Capítulo 5.

4.4 Considerações Finais

Neste capítulo, foi apresentado o mecanismo proposto para suprir as lacunas encontradas na revisão do estado da arte. O mecanismo visa detectar e mitigar ataques DDoS volumétricos diretamente no plano de dados por meio de operações que possibilitem detectar o fluxo do tráfego de rede de clientes maliciosos por meio de um modelo de classificação com base no algoritmo aprendizagem de máquina *Random Forest*, e o uso de listas de prioridade nos dispositivos de encaminhamento para priorizar o fluxo do tráfego daqueles com níveis de confiabilidades aceitáveis (legítimos) e bloquear (descartar) o tráfego para aqueles com baixo valor de confiança (maliciosos), para reduzir os atrasos no processo de detecção e confirmação desses ataques, o volume de mensagens de controle encaminhadas ao controlador para realizar as ações de detecção e mitigação, e evitar o bloqueio indiscriminado dos clientes. O mecanismo também inclui um modelo de compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o

bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, determinadas a partir do estabelecimento de uma confiança global calculado por um sistema *fuzzy*, por meio do controlador para o envio aos dispositivos de encaminhamento, promovendo uma abordagem híbrida que integra o plano de dados e de controle de uma rede SDN de modo a possibilitar a tomada de decisões locais baseadas em informações globais compartilhadas. No próximo capítulo, será detalhado o ambiente experimental utilizado para validar a eficácia do mecanismo proposto, incluindo as métricas e os resultados observados nos experimentos realizados.

Capítulo 5

Resultados

Este capítulo apresenta as informações relacionadas à avaliação de desempenho do mecanismo proposto descrito no Capítulo 4. Foi analisada a integração entre os módulos presentes nos procedimentos do mecanismo e o seu desempenho quando comparado com outro mecanismo presente no estado da arte, que também busca reduzir os impactos dos ataques DDoS volumétricos. As seções a seguir apresentam o ambiente experimental, a metodologia de avaliação, as métricas utilizadas em nossa análise e os resultados observados nos experimentos.

5.1 Configuração e metodologia de avaliação

Os experimentos foram realizados em um computador com sistema operacional Ubuntu 16.04 LTS, equipado com um processador Intel Core i7 de 3.0GHz e 16Gb de memória RAM. O ambiente SDN foi fornecido pelo emulador de rede Mininet na sua versão 2.3.0, que possibilita a criação de uma rede com *hosts* virtuais, *switches*, controladores e enlaces [50].

O mecanismo proposto recebeu o nome de "DataControl-ML"[1], que reflete a sua capacidade de combinar a rapidez de processamento do plano de dados em conjunto com a visão global do plano de controle, proporcionando uma abordagem híbrida para o controle e monitoramento eficaz do tráfego de rede no combate aos ataques DDoS volumétricos. O DataControl-ML foi comparado com o mecanismo desenvolvido por González *et al.* [18], denominado Bungee-ML, que também busca reduzir os impactos causados pelos ataques DDoS volumétricos. Esse mecanismo foi escolhido para comparação por adotar uma abordagem híbrida semelhante, utilizando a cooperação entre o plano de dados e de controle para otimizar a resposta a esses possíveis ataques.

[1]Repositório de implementação do mecanismo proposto disponível em: `https://github.com/ranyelsoncarvalho/DataControl-ML`

O conjunto de dados CICDDoS-2019 [118], fornecido pelo *Canadian Institute of Cybersecurity - University of New Brunswick in Fredericton* foi utilizado para a realização dos experimentos. Embora seja uma base de dados com mais de cinco anos, sua escolha ainda se justifica por continuar sendo uma das principais referências em estudos voltados à detecção e mitigação de ataques DDoS, devido à sua abgrangência, realismo e qualidade na representação de diferentes tipos de tráfegos, tanto malicioso quanto legítimo [119, 120, 121]. Sua utilização contribui para uma avaliação mais consistente do mecanismo proposto, permitindo a comparação com trabalhos da literatura e a validação dos resultados obtidos. Essa base de dados contém diferentes ataques DDoS volumétricos, mas, para nossos experimentos, foram selecionados apenas os ataques DNS-*Flood*, UDP-*Flood* e SYN-*Flood*, devido à sua predominância e impacto significativo em cenários reais de ataques DDoS [122]. Esses tipos de ataques exploram vulnerabilidades na comunicação entre dispositivos para sobrecarregar os recursos da rede, causando indisponibilidade de serviços por meio de um grande volume de tráfego.

Os três ataques selecionados (DNS-*Flood*, UDP-*Flood* e SYN-*Flood*) do conjunto de dados CICDDoS-2019 apresentam comportamentos que incluem a falsificação de endereços IP de origem, o uso de endereços IP únicos, picos e rajadas de tráfego com intensidades variáveis, gerando flutuações no volume total de dados transmitidos, tornando mais desafiador para o mecanismo de segurança desenvolvido diferenciar o tráfego de rede legítimo do malicioso. Para a validação do mecanismo proposto, o conjunto de dados foi dividido em 70% para treinamento e 30% para teste. A Tabela 5.1, apresenta o volume de pacotes legítimos e maliciosos para os ataques selecionados do conjunto de dados utilizado, permitindo que o mecanismo proposto seja avaliado sob fortes condições de ataque.

Tabela 5.1: Volume de pacotes legítimos e maliciosos.

Tipo de Ataque	**Pacotes Maliciosos**	**Pacotes Legítimos**	**Total de Pacotes**
DNS-*Flood*	1843224	395634	2238858
UDP-*Flood*	1362649	312334	1674983
SYN-*Flood*	1042131	204658	1246789

A seleção dos atributos mais representativos do conjunto de dados CICDDoS-2019 adotados nos experimentos deste trabalho, foi orientada por meio da análise da matriz de correlação. Essa técnica estatística permite identificar o grau de associação (relacionamento) entre duas variáveis e a intensidade dessa associação, auxiliando na escolha daqueles atributos que possuem maior influência na representação dos padrões de tráfego, especialmente no contexto da detecção de ataques DDoS [123]. Esse grau é obtido por meio do coeficiente de correlação, que varia de -1 a 1, onde 1 indica uma correlação positiva (as duas variáveis correlacionadas crescem simultaneamente), -1 indica uma correlação

negativa (à medida que uma variável aumenta, a outra diminui) e 0 aponta ausência de correlação (uma variável não tem efeito sobre a outra) [124]. Essa análise contribui para reduzir a dimensionalidade do conjunto de dados, eliminar atributos redundantes e melhorar o desempenho do modelo de classificação.

A Figura 5.1 mostra a correlação entre os principais atributos de cada fluxo, disponíveis no conjunto de dados utilizado e passíveis de serem implementados em um *switch* habilitado para a linguagem P4, no qual se pode observar que alguns atributos possuem uma forte correlação positiva, ou seja, têm uma relação direta, como protocolo (*Protocol*) e o tamanho médio dos pacotes (*Packet Length Mean*), duração do fluxo (*Flow Duration*) e o tempo médio entre dois pacotes enviados no fluxo (*Flow IAT Mean*), total de pacotes encaminhados na direção direta (*Total Fwd Packets*) e o total de pacotes encaminhados na direção inversa (*Total Backward Packets*). Essas correlações fornecem informações importantes para o entendimento da base de dados, permitindo a otimização do desempenho dos mecanismos de detecção e aprimorando a precisão na classificação do tráfego de rede.

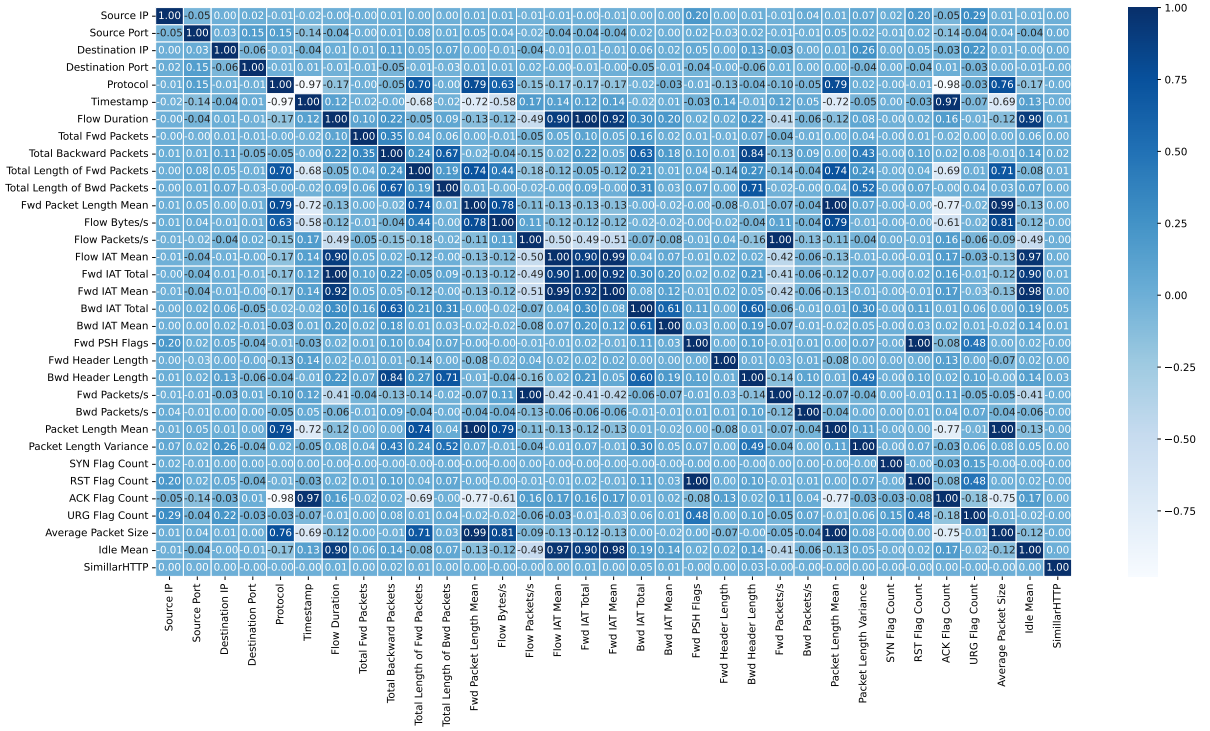


Figura 5.1: Matriz de correlação dos atributos mais representativos do conjunto de dados CICDDoS-2019.

Após a análise da matriz de correlação (Figura 5.1), optou-se por selecionar os atributos com base no grau de associação identificado por meio da correlação positiva entre eles, para compor os registradores do *switch* P4, conforme mostrado na Tabela 5.2. Foram

utilizados 22 registradores para o armazenamento dos valores dos atributos extraídos dos fluxos de rede dos clientes para o uso no processo de treinamento e classificação do modelo de predição. Adicionalmente, são empregados 5 registradores específicos para armazenar os campos da 5-tupla (IP de origem/destino, porta de origem/destino e protocolo), utilizados apenas para identificar um fluxo na rede de um cliente, não sendo considerados no treinamento do modelo.

Tabela 5.2: Lista de atributos para compor os registradores do *switch* P4.

Num.	**Atributo**	**Descrição**
1	*Source IP*	IP de origem
2	*Source Port*	Porta de origem
3	*Destination IP*	IP de destino
4	*Destination Port*	Porta de destino
5	*Protocol*	Protocolo
6	*Flow Duration*	Duração do fluxo
7	*Total Fwd Packets*	Total de pacotes encaminhados na direção direta
8	*Total Bwd Packets*	Total de pacotes encaminhados na direção inversa
9	*Total Length of Fwd Packets*	Tamanho total do pacote na direção direta
10	*Total Length of Bwd Packets*	Tamanho total do pacote na direção inversa
11	*Flow Bytes/s*	Número de bytes de fluxo por segundo
12	*Flow Packets/s*	Número de pacotes de fluxo por segundo
13	*Flow IAT Mean*	Tempo médio entre dois pacotes enviados no fluxo
14	*Fwd IAT Total*	Tempo total entre dois pacotes enviados na direção direta
15	*Fwd IAT Mean*	Tempo médio entre dois pacotes enviados na direção direta
16	*Bwd IAT Total*	Tempo total entre dois pacotes enviados na direção inversa
17	*Bwd IAT Mean*	Tempo médio entre dois pacotes enviados na direção inversa
18	*Fwd Header Length*	Total de bytes usados para cabeçalhos na direção direta
19	*Bwd Header Length*	Total de bytes usados para cabeçalhos na direção inversa
20	*Fwd Packets/s*	Número de pacotes encaminhados por segundo
21	*Bwd Packets/s*	Número de pacotes de retorno por segundo
22	*Packet Length Mean*	Tamanho médio de um pacote
23	*SYN Flag Count*	Número de pacotes com a flag SYN
24	*RST Flag Count*	Número de pacotes com a flag RST
25	*ACK Flag Count*	Número de pacotes com a flag ACK
26	*URG Flag Count*	Número de pacotes com a flag URG
27	*Idle Mean*	Tempo médio em que um fluxo ficou ocioso antes de se tornar ativo

Para os experimentos, os parâmetros do DataControl-ML foram configurados com base em avaliações empíricas obtidas em experimentos anteriores [29]. A Tabela 5.3 apresenta os parâmetros utilizados nos experimentos. Foram realizadas 5 simulações para cada tipo de ataque DDoS volumétrico selecionado (DNS-*Flood*, UDP-*Flood* e SYN-*Flood*), com o objetivo de avaliar a robustez do mecanismo proposto sob diferentes condições de tráfego e variações nos padrões de comportamento dos clientes. A escolha por 5 repetições é uma prática comum em experimentos com variabilidade moderada, sendo útil para fornecer estimativas estáveis, ao mesmo tempo que possibilita resultados mais rápidos dos testes. Essa repetição permite uma avaliação mais confiável do desempenho do mecanismo proposto, reduzindo a influência de valores atípicos ou oscilações pontuais, garantindo maior confiabilidade na análise. Os resultados são apresentados com base na média dos valores obtidos nessas simulações, acompanhada de um intervalo de confiança de 95%. Cada simulação teve duração de 180 segundos, com o tráfego legítimo sendo

gerado continuamente ao longo de todo o período de simulação. Enquanto, o tráfego malicioso teve duração de 60 segundos (60s - 120s).

Tabela 5.3: Lista de Parâmetros do DataControl-ML.

Parâmetro	**Valor**
Número de simulações	5
Tipos de Ataques	DNS-*Flood*; UDP-*Flood*; SYN-*Flood*.
Tempo de cada simulação	180 segundos
Início do ataque	60 segundos
Fim do ataque	120 segundos
Janela de observação (w): *Coleta de Dados	150 fluxos ou 100ms
Hiperparâmetros do algoritmo de aprendizagem de máquina: *Random Forest*	$n\_estimators = 10$ $max\_depth = 6$ $max\_features = sqrt$ $min\_samples\_split = 6$ $min\_samples\_leaf = 4$
Coeficiente de agregação do valor de confiança local (β)	0,5
Intervalo de tempo: Fator de Esquecimento (Tm)	15 segundos
Coeficiente de redução: Fator de Esquecimento (δ)	0,9
Coeficiente de suavização: Lista de Confiança (α)	0,5
Tempo de Bloqueio (Ψ)	60 segundos
Janelas de Observação (w): *Gerenciador de Envio *Calculo Confiança Global *Disseminação	250 pacotes ou 150ms
Função de pertinência: Conjunto *Fuzzy* (Confiança Local)	Baixo: trapezoidal $(0; 0; 0{,}2; 0{,}5)$ Médio: triangular $(0{,}2; 0{,}5; 0{,}8)$ Alto: trapezoidal $(0{,}5; 0{,}8; 1; 1)$
Função de pertinência: Conjunto *Fuzzy* (Confiança Global)	Não Confiável: trapezoidal $(0; 0; 0{,}2; 0{,}4)$ Parcialmente Confiável: trapezoidal $(0{,}2; 0{,}4; 0{,}6; 0{,}8)$ Confiável: trapezoidal $(0{,}6; 0{,}8; 1; 1)$
Regra de Inferência	Se o valor de confiança é Baixo, então o cliente é Não Confiável. Se o valor de confiança é Médio, então o cliente é Parcialmente Confiável. Se o valor de confiança é Alto, então o cliente é Confiável.
Método de Defuzzificação	Média Ponderada dos Máximos

A janela de observação w para o módulo de coleta de dados foi definida como janela por fluxo com um tamanho de 150 cada ou até atingir o tempo limite de 100ms para o preenchimento dos dados. A escolha desse tamanho e limite de tempo para a janela porque eles oferecem um tempo de processamento mais eficiente, permitindo a coleta de dados de maneira ágil sem comprometer o desempenho da rede e garantindo que os dados sejam extraídos de forma contínua e consistente. A classificação do tráfego de rede é realizada pelo algoritmo de aprendizagem de máquina do tipo supervisionado *Random Forest*, devido ao seu alto grau de precisão e à redução do risco de *overfitting* [83], permitindo uma análise eficaz e a identificação de padrões complexos no tráfego de rede. O algoritmo teve os hiperparâmetros otimizados por meio da função `RandomizedSearchCV` [107] para maximizar o desempenho do mecanismo de detecção na tarefa de classificação. Esses hiperparâme-

tros foram ajustados para os seguintes valores: $n\_estimators = 10$, $max\_depth = 6$, $max\_features = sqrt$, $min\_samples\_split = 6$, $min\_samples\_leaf = 4$.

O coeficiente de agregação do valor de confiança local foi ajustado para $\beta = 0{,}5$, permitindo equilibrar as informações passadas e recentes do cliente, garantindo que tanto os dados históricos quanto os mais recentes tenham peso semelhante no cálculo do valor de confiança local, assegurando uma avaliação precisa e atualizada do comportamento do cliente na rede. O intervalo de tempo para o fator de esquecimento é de $Tm = 15$ segundos e o coeficiente de redução é de $\delta = 0{,}9$, de modo a priorizar os clientes mais ativos e engajados na rede, permitindo que o mecanismo proposto ajuste dinamicamente os valores de confiança com base no comportamento mais recente dos clientes.

O coeficiente de suavização para estabelecer o limiar da lista de prioridade (confiança) é de $\alpha = 0{,}5$, balanceando as informações atuais com as anteriores fornecidas pelos clientes. O tempo de restrição para a lista de bloqueio é de $\Psi = 60$ segundos, a fim de permitir que clientes com problemas de conectividade não sejam penalizados na rede, permitindo que esses clientes reconectem sem enfrentar penalidades prolongadas. As janelas de observação w para os módulos de gerenciador de envio, calcular a confiança global e disseminação dos dados foram definidas para 250 pacotes ou até atingir o tempo limite de 150ms para o preenchimento dos dados, pois proporcionam um processamento mais rápido e eficiente, equilibrando a capacidade de capturar e organizar as informações dos clientes.

Para os conjuntos *fuzzy*, os tipos de funções de pertinência e seus respectivos parâmetros foram escolhidos com base em uma avaliação dos dados históricos coletados durante o desenvolvimento do sistema *fuzzy*, de modo a atender às necessidades do mecanismo em relação às demais disponíveis na literatura e reduzir o custo computacional. O sistema *fuzzy* proposto utiliza o modelo Mamdani, ou seja, para todas as regras cujo grau de relevância da função de pertinência é maior que zero, elas contribuirão para o cálculo da saída correspondente do sistema de inferência [111]. As regras foram definidas com base em conhecimento especializado sobre o sistema proposto, considerando as características da rede coletadas durante o desenvolvimento do sistema *fuzzy*, como os padrões de comportamentos dos clientes, de modo a capturar nuances e variações no ambiente de rede, possibilitando decisões mais precisas e adaptativas. O método de defuzzificação escolhido foi a média ponderada dos máximos, que considera o valor médio ponderado dos valores centrais ativados, onde os pesos são os graus de pertinência máximo de cada variável linguística de saída [114]. Esse método foi escolhido porque é capaz de combinar precisão e rapidez com um custo computacional reduzido, tornando-se adequado para a proposta deste trabalho, que utiliza um controlador SDN com memória e capacidade de processamento limitadas.

Os parâmetros de configuração do Bungee-ML (mecanismo presente no estado da

arte) foram definidos conforme as recomendações dos autores [18]. A otimização dos hiperparâmetros para o algoritmo de classificação do tráfego de rede do mecanismo seguiu um processo semelhante ao adotado em nosso mecanismo proposto.

A Figura 5.2 apresenta a topologia de rede utilizada nos experimentos para a avaliação do mecanismo proposto.

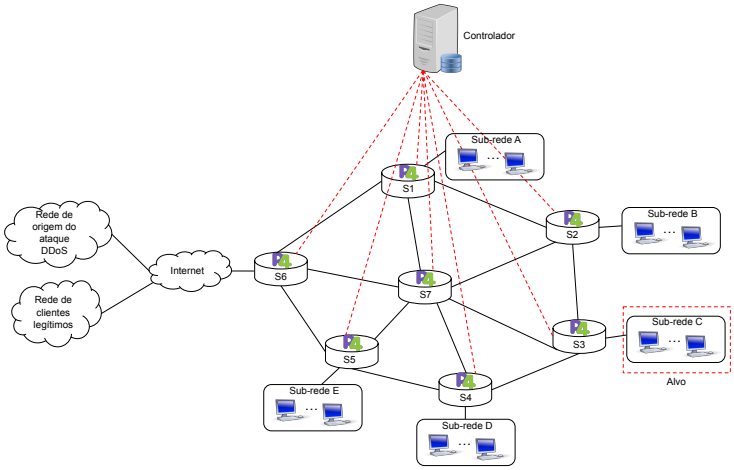


Figura 5.2: Topologia de rede utilizada nos experimentos.

A topologia de rede consistiu em: *i*) sete dispositivos de encaminhamento; *ii*) e um controlador. Os dispositivos de encaminhamento S1, S7 e S5 são encarregados de executar os módulos do mecanismo de detecção/mitigação (Procedimento 1) no combate aos ataques DDoS volumétricos, enquanto o dispositivo de borda S6 é responsável por realizar o roteamento do tráfego de rede de entrada (composto por tráfego malicioso e legítimo proveniente de fontes externas) com base na faixa de endereço IP de origem para os dispositivos que possuem o mecanismo de segurança, conforme apresentado na Tabela 5.4. Cabe destacar que, neste estudo, considera-se exclusivamente o cenário de ataques externos, não sendo analisados ataques originados dentro da própria rede. Assim, o dispositivo S6 opera roteando os pacotes, sem os enviar ao controlador para o preenchimento da tabela de fluxo. Dessa forma, ele encaminha os pacotes conforme a rota correspondente para os dispositivos que possuem o mecanismo de segurança. A sub-rede C é o alvo dos ataques DDoS, ela é gerenciada pelo dispositivo S3. O controlador possui os módulos do mecanismo (Procedimento 2) para definir o valor de confiança global dos clientes e realizar o compartilhamento das ações de controle com base nesse valor para os dispositivos de encaminhamento.

A ferramenta Tcpreplay [125] foi utilizada para replicar o tráfego de rede do conjunto de dados CICDDoS-2019 no ambiente de teste. Essa ferramenta permite a reprodução

Tabela 5.4: Tabela de roteamento do dispositivo S6.

Faixa de IP de origem	**Ação**
0.0.0.0/8 até 85.0.0.0/8	Encaminhar para S1
86.0.0.0/8 até 170.0.0.0/8	Encaminhar para S7
171.0.0.0/8 até 255.0.0.0/8	Encaminhar para S5

de pacotes de redes a partir de arquivos no formato `pcap` (*Packet Capture*) [126], que armazenam pacotes de dados capturados de uma rede de computadores, preservando as características do tráfego capturado, como os endereços IP, protocolos utilizados e os dados transportados. Com isso, foi possível recriar, de forma realista, os cenários de ataques contidos na base de dados, possibilitando avaliar o desempenho e a eficácia do mecanismo proposto em condições semelhantes às encontradas em situações reais de ataques DDoS, o que contribui para validar a robustez do mecanismo frente a uma ampla gama de técnicas de ataque utilizadas em cenários reais.

As seguintes métricas são utilizadas para avaliar o mecanismo proposto:

- **Variabilidade do valor de confiança local dos clientes:** indica a variabilidade do valor de confiança local dos clientes (legítimos e maliciosos) ao longo da simulação.
- **Número de pacotes encaminhados e descartados:** descreve a quantidade de pacotes encaminhados e descartados pelo mecanismo nas listas locais dos *switches*.
- **Confiança global e classificação global:** definição do valor de confiança global e a classificação global do cliente mediante o sistema *fuzzy* desenvolvido.
- **Desempenho da detecção:** indica o percentual de fluxos legítimos e maliciosos que o mecanismo classificou corretamente, obtido através da acurácia (número de classificações corretas realizadas pelo modelo em relação ao número total de previsões realizadas), precisão (representa a proporção de previsões corretas para uma classe específica em relação ao número total de previsões realizadas para essa classe), *recall* (simboliza a proporção de dados relevantes que o modelo consegue encontrar) e F1-*score* (combina a precisão e *recall* em uma única medida para avaliar o desempenho do modelo de classificação, proporcionando um equilíbrio entre ambas as métricas).
- **Tempo de detecção:** tempo decorrido entre o início do ataque e a sua detecção (confirmação do fluxo de rede como malicioso) por parte do mecanismo proposto.

- **Tempo de convergência:** tempo decorrido desde o início do tempo de recebimento de um pacote de disseminação até o momento em que todos os *switches* possuem as mesmas informações de controle desse pacote.
- **Volume de mensagens de controle:** número de pacotes modificados encaminhados (pacote de relatório e de disseminação) entre o *switch* e o controlador, tanto do *switch* para o controlador quanto do controlador para o *switch*, para executar as ações de detecção e/ou mitigação.
- **Eficácia da mitigação:** indica o percentual de Verdadeiros Positivos (VP): proporção de pacotes provenientes de endereços IP maliciosos que foram realmente descartados; e Falsos Positivos (FP): proporção de pacotes provenientes de endereços IP legítimos que foram descartados indevidamente.
- **Consumo de CPU e memória:** indica a quantidade de processamento (CPU e memória) que o mecanismo requer para executar as ações de detecção e mitigação desenvolvidas.

5.2 Resultados

Nesta seção, são discutidos e analisados os resultados alcançados pelo mecanismo proposto no combate aos ataques DDoS volumétricos. Inicialmente, analisou-se a integração entre os módulos presentes nos procedimentos do mecanismo para a realização das ações de detecção e mitigação. Em seguida, realizou-se uma comparação entre o mecanismo proposto e uma abordagem presente no estado da arte.

5.2.1 Integração entre os módulos presentes nos procedimentos do mecanismo proposto

Os resultados apresentados a seguir foram obtidos a partir do experimento realizado durante o ataque SYN-*Flood*, presente no conjunto de dados analisado CICDDoS-2019. O objetivo desse resultado é demonstrar a integração entre os módulos presentes nos procedimentos do mecanismo proposto para a realização das ações de detecção e mitigação, apresentando a função de cada módulo no processo, desde a coleta e análise dos dados de tráfego até a aplicação das medidas de contenção. Para isso, analisou-se o comportamento de dois clientes com padrões de tráfego de rede distintos. O primeiro cliente (1) apresenta um padrão de comunicação estável e consistente ao longo do tempo, representando o tráfego legítimo. Já o segundo cliente (2) é responsável pelo ataque DDoS em questão, caracterizado por um grande volume de requisições em um curto intervalo de tempo.

A análise dos resultados dessa subseção será guiada por métricas como o valor de confiança local de cada cliente ao longo do tempo, o número de pacotes encaminhados e descartados conforme as listas locais do *switch* às quais os clientes estão associados, além do valor de confiança global e a classificação global atribuída a esses clientes. Dessa forma, primeiro será explicado como os módulos presentes no mecanismo proposto contribuem para a visualização dos dados relacionados às métricas citadas anteriormente. Em seguida, é explicado os resultados obtidos por cada cliente durante a realização do experimento.

A Figura 5.3 mostra a variabilidade dos valores de confiança local dos clientes na rede ao longo do tempo do experimento, que durou 180 segundos.

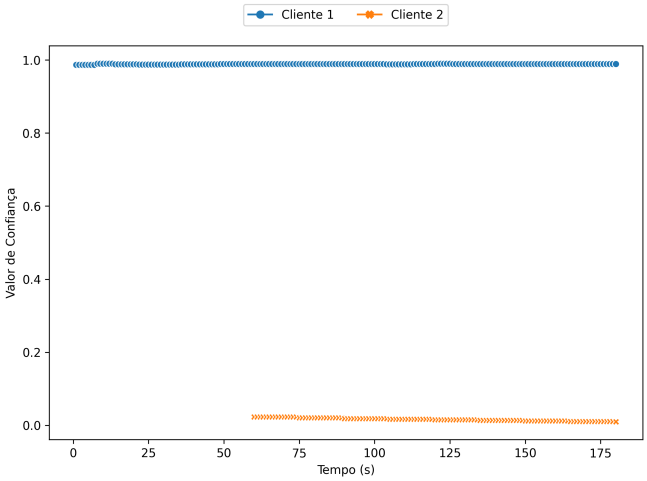


Figura 5.3: Variabilidade do valor de confiança local dos clientes.

Os valores de confiança atribuídos aos clientes, observados na Figura 5.3 pelo mecanismo proposto, são resultados da ação conjunta dos módulos de coleta de dados, classificação e agregação do valor de confiança, ambos presentes no plano de dados. O primeiro módulo coleta os dados de pacotes de rede dos clientes, como o tamanho do pacote em *bytes*, o número de pacotes encaminhados, as *flags* de controle, etc., agrupa-os em fluxos específicos identificados pela 5-tupla (IP de origem/destino, porta de origem/destino e protocolo), em seguida, encaminha essas informações para o segundo módulo (classificação).

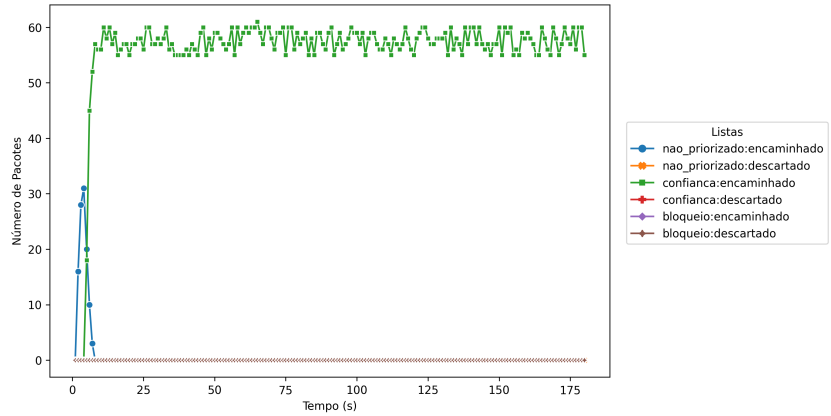
O módulo de classificação analisa os dados recebidos, atribui uma classificação (legítimo ou malicioso) por meio do algoritmo de aprendizagem de máquina (*Random Forest*) para cada fluxo de rede do cliente e, em seguida, o método *Platt Scaling* determina um valor de confiança para a classificação realizada, tendo como base o comportamento do fluxo desse cliente. Assim, quanto mais alto o valor de confiança (próximo a 1), maior a indicação de que o cliente apresenta um comportamento consistente, regular e alinhado às características de um cliente legítimo, sem evidências de ações maliciosas. Esse valor

reflete a probabilidade de o cliente estar agindo de forma confiável, contribuindo para uma análise mais precisa e eficaz no processo de tomada de decisão sobre a sua priorização ou restrição na rede. Para valores próximos a 0, indica-se que o cliente possui comportamento potencialmente malicioso. Por fim, a cada novo valor de confiança obtido pelos clientes durante o período de simulação, o terceiro módulo (agregação do valor de confiança) se encarrega de formar um valor de confiança local para esse cliente para representar, de maneira mais consistente e estável, o comportamento local de cada cliente ao longo do tempo. Esse valor reflete tanto as características recentes quanto o histórico do tráfego de rede do cliente, permitindo uma análise equilibrada que reduz o impacto de flutuações momentâneas.

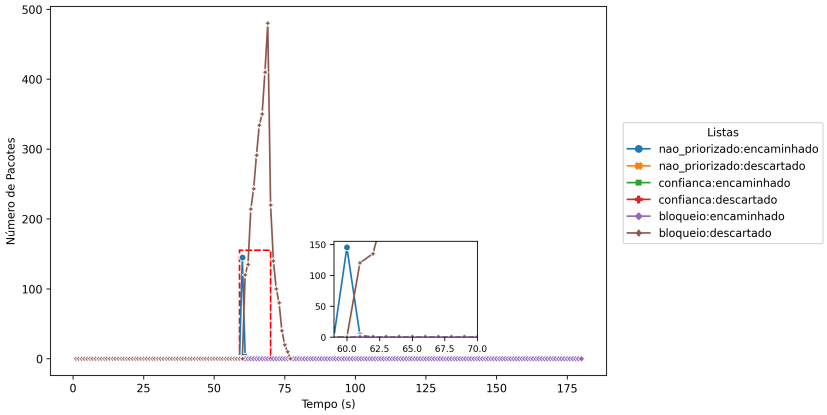
Observa-se na Figura 5.3 que o cliente 1 apresentou um valor de confiança alto em razão do seu comportamento consistente, caracterizado por padrões de tráfego regulares. Entre esses padrões, destacam-se o envio e recebimento de pacotes dentro dos limites esperados para uma comunicação normal, ausência de conexões suspeitas e outros sinais que poderiam indicar comportamento malicioso capaz de comprometer a segurança da rede. Nota-se que, no tempo 60s, ocorre o início do ataque, momento em que o tráfego malicioso do cliente 2 é detectado na rede pelo mecanismo. Devido ao comportamento potencialmente malicioso, caracterizado por um volume massivo de requisições do tipo SYN em um curto intervalo de tempo para um destino específico (padrão típico de um ataque DDoS volumétrico), o mecanismo atribui a esse cliente um valor de confiança baixo em razão da sua classificação como malicioso e aciona o procedimento de mitigação, bloqueando o cliente e descartando imediatamente o tráfego malicioso gerado por ele, que poderá ser visto mais adiante. Esse resultado demonstra a capacidade do mecanismo desenvolvido de observar o comportamento do fluxo de tráfego de rede dos clientes e caracterizá-los em tempo real, de forma eficiente e precisa.

A Figura 5.4 apresenta o número de pacotes encaminhados e descartados a cada segundo pelo mecanismo, de acordo com o dispositivo de encaminhamento e a lista à qual o cliente está associado. A associação de cada cliente às listas é determinada com base no valor de confiança local calculado, que reflete o comportamento observado e orienta as ações de encaminhamento ou descarte a serem aplicadas pelo mecanismo.

A associação dos clientes às listas presentes nos dispositivos de encaminhamento (Figura 5.4) decorre da ação do módulo de gerenciamento das listas, presente no plano de dados. Esse módulo atribui o fluxo de tráfego de rede dos clientes presentes na rede em três listas locais: não priorizado, confiança e bloqueio, conforme a classificação e o valor de confiança local obtido por cada cliente durante a análise dos seus fluxos de redes. A lista de “não priorizado” engloba os pacotes encaminhados de clientes sem prioridade associada, como clientes legítimos que ainda não atingiram o limiar de confiança estabelecido



(a) Cliente: 1



(b) Cliente: 2

Figura 5.4: Número de pacotes encaminhados e descartados de conforme a lista à qual o cliente está associado.

para a lista de confiança ou aqueles que ainda não foram classificados pelo mecanismo. A lista de "confiança" gerencia os pacotes provenientes de clientes com bons níveis de confiança (legítimos), ou seja, que estão acima o limiar de confiabilidade estabelecido, proporcionando tratamento preferencial (priorização) no encaminhamento de pacotes dos clientes dessa lista. Por último, a lista de "bloqueio" gerencia os pacotes dos clientes classificados como maliciosos (baixo valor de confiança), na qual realiza a ação de descarte dos pacotes maliciosos.

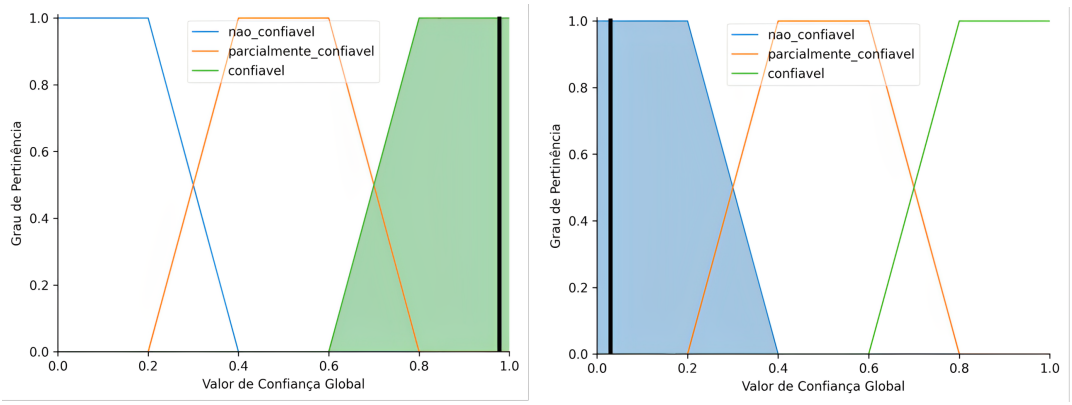
Constata-se que o cliente 1 (Figura 5.4(a)), classificado como legítimo pelo mecanismo proposto, possui um número considerável de pacotes encaminhados, o que está diretamente relacionado ao seu comportamento estável e consistente ao longo do tempo, como a realização de conexões TCP de maneira completa e a transmissão de dados dentro dos padrões típicos de tráfego legítimo. Note que, inicialmente, os seus pacotes são gerenciados pela lista de "não priorizado", mas, à medida que o cliente acumula mais confirmações de fluxos legítimos, o número de pacotes encaminhados aumenta, o que resulta em novos

valores confiança progressivamente mais altos em razão da consistência do seu comportamento na rede. Como resultado, o cliente acaba superando o limiar de confiabilidade estabelecido para o ingresso na lista de prioridade, levando a sua promoção a lista de "confiança". Ao ser gerenciado por essa lista, os seus pacotes passam a receber uma maior prioridade de recursos, o que resulta em uma maior quantidade de pacotes encaminhados, promovendo uma melhor qualidade de serviço para esse cliente. Observe que, mesmo durante o ataque (60s - 120s), o seu tráfego de rede não é prejudicado, demostrando a resiliência do mecanismo desenvolvido e sua capacidade de priorizar o fluxo de tráfego de rede de clientes com níveis de confiabilidade aceitáveis (legítimos), a fim de evitar o bloqueio indiscriminado.

Para o cliente 2 (Figura 5.4(b)), nota-se que ele possui um grande volume de pacotes descartados devido à ação do mecanismo proposto para lidar com clientes maliciosos, que inclui o descarte imediato dos seus pacotes e a inclusão na lista de "bloqueio". O cliente em questão ingressa na rede no instante 60s, enviando uma quantidade significativa de pacotes maliciosos desde o início, por meio de requisições do tipo SYN, para sobrecarregar os recursos do seu alvo localizado na sub-rede C e o controlador. Esse volume de requisições gerou um pico abrupto de tráfego em um curto intervalo de tempo direcionado a um destino específico. Observe que, inicialmente, esses pacotes são encaminhados pela lista de "não priorizado", pois essa lista trata de pacotes de clientes ainda não classificados pelo mecanismo ou aqueles já classificados, mas que ainda não atingiram o limiar de confiança estabelecido para a priorização. O mecanismo, ao detectar rapidamente esse tipo de padrão de tráfego do cliente 2 — menos de 1 segundo após o início do ataque, como pode ser observado no detalhe da Figura 5.4(b) —, ajusta imediatamente o tratamento dado ao cliente e o seu tráfego de rede. O ajuste envolve a classificação do cliente como malicioso e o seu gerenciamento pela lista de "bloqueio", na qual é aplicada a política de restrição (bloqueio/descarte) para impedir que esses pacotes sobrecarreguem a rede. Isso garante a proteção da rede contra o ataque DDoS, demonstrando a rapidez e eficácia do mecanismo desenvolvido. Além disso, a ação de realizar o bloqueio de maneira seletiva evita que o tráfego malicioso comprometa o desempenho e a estabilidade da rede, como pode ser observado no volume de pacotes descartados para o cliente em questão na lista de "bloqueio".

A Figura 5.5 mostra o valor de confiança global e a classificação global dos clientes, conforme a execução do sistema *fuzzy* desenvolvido. Essa classificação é baseada no grau de pertinência do valor de confiança global obtido em relação às funções de pertinências definidas para o conjunto *fuzzy* "Confiança Global".

O valor de confiança global e a classificação global dos clientes observados na Figura 5.5 são frutos do trabalho em conjunto dos módulos de gerenciador de envio (presente no



(a) Cliente: 1 (b) Cliente: 2

Figura 5.5: Confiança global e classificação global.

plano de dados), extração dos dados e calcular a confiança global (ambos presentes no plano de controle). O primeiro módulo envia para o controlador o valor mais recente de confiança local dos clientes presentes nas listas dos dispositivos de encaminhamento, por meio de pacotes de relatórios organizados em janelas de observação. O segundo módulo (extração dos dados) extrai dos pacotes de relatórios os valores de confiança local dos clientes para organizá-los em uma estrutura de dados. Os dados presentes nessa estrutura são encaminhados ao terceiro módulo (calcular a confiança global).

O módulo (calcular a confiança global) recebe os valores de confiança local dos clientes computados pelos dispositivos de encaminhamento e os organiza em diferentes termos linguísticos: “alto”, “médio” e “baixo”, com base nas funções de pertinências definidas, representando o conjunto *fuzzy* de entrada “Confiança Local”. Em seguida, são aplicadas as regras do sistema de inferência, que processa e ativa as regras estabelecidas com graus de pertinência para cada regra, conforme o conjunto de valores de confiança local analisados de cada cliente. Esse processo transforma o conjunto de valores de confiança local de um cliente em um valor de confiança global que melhor representa os valores *fuzzy* inferidos, que será identificado no conjunto *fuzzy* de saída “Confiança Global”, que classifica o valor de confiança global de cada cliente conforme o grau de pertinência obtido em “não confiável”, “parcialmente confiável” e “confiável”. Isso permite que o controlador estabeleça ações de controle que podem incluir o bloqueio (descarte) do tráfego para o cliente classificado como não confiável, a permissão do tráfego daquele classificado como parcialmente confiável (sem prioridade associada) ou a priorização do tráfego para aquele classificado como confiável. Tais ações são compartilhadas com os dispositivos de encaminhamento para aplicar a política de restrição e/ou priorização de tráfego em diferentes pontos da rede, por meio de pacotes de disseminação organizados em janelas de observação por meio do módulo de disseminação dos dados, presente no plano de controle.

Identifica-se que o cliente 1 (Figura 5.5(a)) teve a sua classificação global definida como "confiável", uma vez que seus valores de confiança local, extraídos dos pacotes de relatórios enviados pelos dispositivos de encaminhamento ao controlador, ao serem processados pelo sistema *fuzzy*, resultaram em um alto valor de confiança global que indicou um elevado grau de pertinência ao nível confiável. Esse alto grau de pertinência reflete a consistência e estabilidade do comportamento do cliente em questão ao longo do tempo, com fluxos de comunicação que se mantiveram dentro dos padrões esperados para um comportamento legítimo.

Para o cliente 2 (Figura 5.5(b)), nota-se que ele teve a sua classificação global definida como "não confiável", em razão de seus valores de confiança local indicarem um padrão de comportamento anômalo característico de um ataque DDoS. Ao ser processado pelo sistema *fuzzy*, esses valores resultaram em um baixo valor de confiança global que refletiu o alto grau de pertinência o nível não confiável. Esse alto grau de pertinência revela o comportamento malicioso apresentado pelo cliente em questão, que inclui o envio de uma quantidade significativa de pacotes maliciosos por meio de requisições do tipo SYN. Assim, o sistema foi capaz de atribuir aos clientes (1 e 2) classificações globais de confiança condizentes com os padrões de comportamentos apresentados por eles durante a execução do experimento.

Por fim, com a classificação global obtida para cada cliente, o controlador estabelece a ação de controle destinada a cada um deles, como o descarte (bloqueio) do tráfego de rede do cliente 2 classificado como não confiável e a priorização do tráfego para o cliente 1 classificado como confiável. Essas ações são compartilhadas com os dispositivos de encaminhamento presentes na rede (Figura 5.2), por meio de pacotes de disseminação encaminhados em intervalos regulares, conforme o ajuste da janela de observação do módulo de disseminação dos dados. Assim, as informações compartilhadas sobre os clientes permitem que os dispositivos de encaminhamento, mantenham a coerência nas decisões de encaminhamento e bloqueio, garantindo que a política de restrição (bloqueio/descarte) e priorização sejam aplicadas corretamente em toda a rede, proporcionando maior celeridade nos procedimentos de detecção e mitigação do mecanismo proposto em diferentes pontos da rede. Esse resultado demonstra a capacidade do mecanismo desenvolvido em realizar o compartilhamento de informações globais, para que os dispositivos de encaminhamento possam tomar decisões locais com base nessas informações compartilhadas.

5.2.2 Comparação com o estado da arte

Os resultados apresentados a seguir foram obtidos a partir dos experimentos realizados para avaliar o comportamento do mecanismo proposto em relação aos diferentes tipos de ataques DDoS volumétricos selecionados (DNS-*Flood*, UDP-*Flood* e SYN-*Flood*), pre-

sentes no conjunto de dados analisado CICDDoS-2019. O objetivo desses resultados é demonstrar a eficácia e a qualidade do mecanismo proposto, examinando sua resposta para diferentes tipos de ataques DDoS, quando comparado a outro mecanismo presente no estado da arte (o Bungee-ML).

A análise dos resultados dessa subseção será guiada por métricas como o desempenho da detecção do mecanismo, avaliado por meio da acurácia e do F1-*score* na classificação do fluxo de tráfego de rede do cliente como legítimo ou malicioso, o tempo gasto para classificar os fluxos de rede dos clientes, o tempo de convergência para a sincronização das informações nos dispositivos de encaminhamento, além do volume de mensagens de controle encaminhadas para executar as ações de controle dos clientes, a eficácia da mitigação e o consumo de recursos (CPU e memória). Dessa forma, será explicado os resultados obtidos pelo nosso mecanismo proposto, o DataControl-ML, e o mecanismo presente no estado da arte, o Bungee-ML.

A Figura 5.6 mostra o desempenho do DataControl-ML e Bungee-ML para classificar o fluxo de tráfego de rede dos clientes em legítimo ou malicioso. Nota-se que os mecanismos demonstraram capacidade de identificar os ataques de maneira coerente, evidenciando sua eficácia em separar o tráfego legítimo de padrões associados a atividades maliciosas referentes aos ataques DDoS analisados.

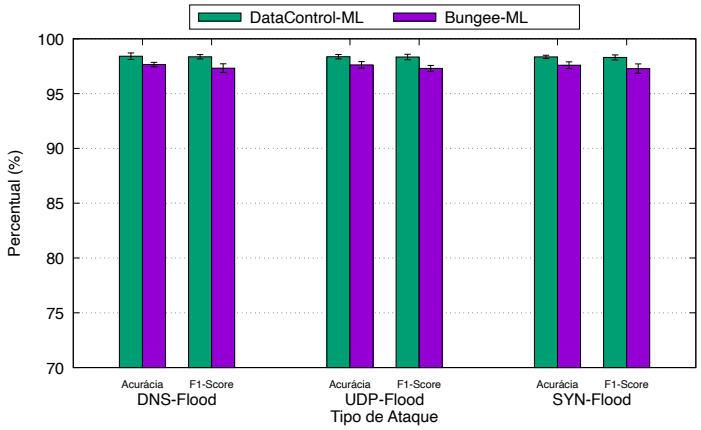


Figura 5.6: Desempenho da detecção.

Observa-se na Figura 5.6 que o DataControl-ML foi $\approx$ 1,05% mais preciso que o Bungee-ML, demonstrando um número maior de classificações corretas (acurácia) e um bom equilíbrio entre a capacidade de identificar corretamente o tráfego malicioso (*recall*) e de minimizar o número de falsos positivos (precisão) por meio do F1-*Score*. Esse bom desempenho está relacionado à classificação mediante a técnica de aprendizagem de máquina empregada diretamente no plano de dados, que foi capaz de identificar padrões não lineares e relações mais complexas entre as características do tráfego no conjunto

de dados analisado, como o tamanho dos pacotes, o número de *bytes*, o total de pacotes encaminhados, o tempo médio entre dois pacotes enviados no fluxo, etc., o que permitiu uma classificação mais precisa por parte do mecanismo para os diferentes tipos de ataque DDoS analisados. Por outro lado, o Bungee-ML apresentou um desempenho inferior, em razão da técnica estatística entropia empregada no plano de dados para filtrar os pacotes suspeitos, o que acabou apresentando uma filtragem mais imprecisa, o que limitou a capacidade do mecanismo em capturar informações mais complexas e sutis dada as características presentes no conjunto de dados utilizado, diminuindo a precisão geral do mecanismo.

Os ataques DDoS precisam ser detectados de forma rápida para que as ações de mitigação sejam acionadas e, assim, preservem os recursos da rede. A Figura 5.7 apresenta o tempo gasto pelo DataControl-ML e Bungee-ML para classificar e confirmar um fluxo como malicioso na rede. Os mecanismos demonstraram agilidade para identificar os diferentes ataques DDoS volumétricos, garantindo uma resposta rápida às ameaças, com tempos abaixo de 1 segundo.

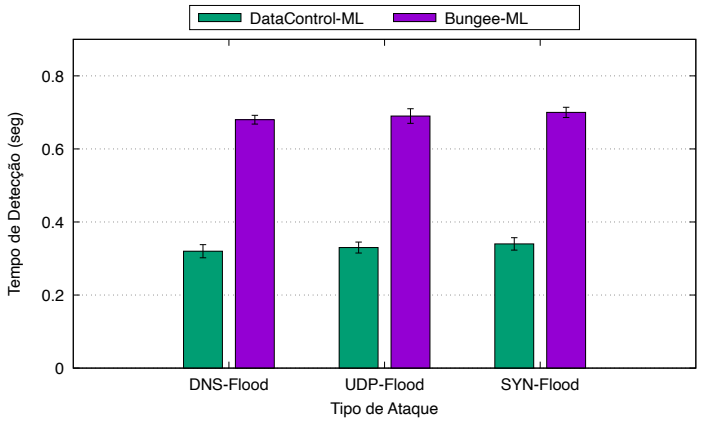


Figura 5.7: Tempo de detecção.

Nota-se na Figura 5.7 que o DataControl-ML é $\approx$ 52,18% mais rápido que o Bungee-ML para confirmar que um fluxo é malicioso, demonstrando a capacidade do mecanismo em identificar o fluxo de tráfego de rede dos clientes maliciosos próximos aos pontos de ingresso na rede, reduzindo os atrasos no processo de confirmação de qualquer atividade suspeita associada aos ataques DDoS. Essa rapidez está relacionada à ação de detecção realizada no plano de dados por parte do mecanismo proposto, sem a necessidade de enviar dados adicionais ao controlador, além de agrupar os pacotes observados em fluxos com base em seus atributos comuns e à organização das janelas de observação por fluxo, proporcionando um gerenciamento mais eficiente e rápido dos dados coletados. Em contrapartida, o tempo mais elevado do Bungee-ML está associado à estratégia adotada pelo

mecanismo, que consiste em duas etapas. A primeira etapa busca identificar as fontes suspeitas no plano de dados e encaminhá-las juntamente com os pacotes provenientes dessas fontes para o plano de controle. O controlador, ao receber essas fontes e os pacotes subsequentes, aciona a segunda etapa, que busca extrair as características dos pacotes dessas fontes, agrupando-os em fluxos com base em suas características comuns, para então classificá-las como maliciosas. Tais etapas acabam por aumentar o tempo de detecção por parte do mecanismo presente no estado da rede, como pode ser observado.

A Figura 5.8 mostra o tempo necessário para que os dispositivos de encaminhamento possuam as mesmas informações presentes em um pacote modificado (criado) pelo DataControl-ML e Bungee-ML. Observa-se que os mecanismos apresentaram tempos abaixo de 10ms para garantir a sincronização das informações entre os dispositivos para os diferentes tipos de ataques executados. Essa rapidez é crucial para assegurar que as políticas de segurança sejam aplicadas de forma consistente em toda a rede, minimizando riscos e garantindo a eficácia das medidas de detecção e mitigação adotadas pelos mecanismos de segurança.

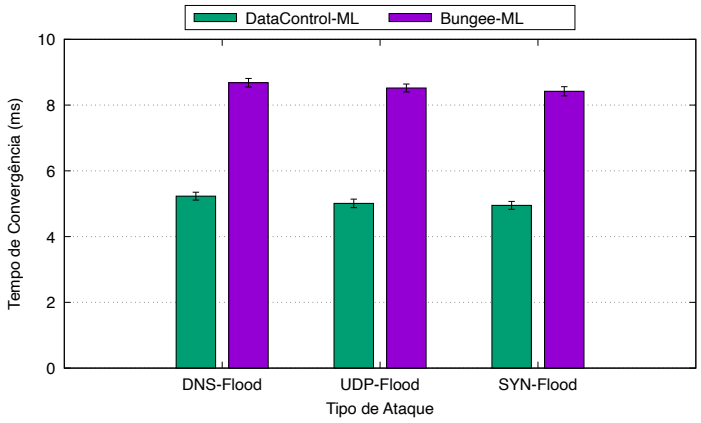


Figura 5.8: Tempo de convergência.

Observa-se na Figura 5.8 que o DataControl-ML é $\approx$ 46,71% mais rápido que o Bungee-ML, pois, ao enviar o pacote de disseminação de maneira paralela, ou seja, do controlador para todos os dispositivos de encaminhamento por meio de um canal fora de banda (*out-of-band*), reduz significativamente o tempo de comunicação e evita congestionamentos no canal de comunicação principal (tráfego de dados). Esse método permite que os pacotes de disseminação sejam entregues simultaneamente a todos os dispositivos de encaminhamento conectados diretamente ao controlador por meio de uma porta lógica, garantindo que o tráfego de controle não interfira no tráfego de dados, otimizando a distribuição de informações e minimizando a latência de comunicação [127]. Por outro lado, no Bungee-ML observou-se um atraso maior na propagação de uma informação, devido à estratégia

adotada pelo mecanismo, que consiste em enviar o pacote de alarme (confirmação de um fluxo suspeito) do controlador para o dispositivo que originou a fonte do fluxo suspeito, para que ele alerte seus dispositivos mais próximos (intermediários). Esse processo gera um efeito em cascata, no qual a informação precisa ser propagada por meio de múltiplos dispositivos intermediários antes de alcançar todos os outros dispositivos da rede. Assim, o tempo adicional necessário para a comunicação entre os dispositivos intermediários aumenta a latência geral do mecanismo, como pode ser observado.

A Figura 5.9 mostra o número de pacotes modificados trocados entre os dispositivos de encaminhamento e o controlador, incluindo tanto os pacotes enviados dos dispositivos para o controlador quanto do controlador para os dispositivos, para que o DataControl-ML e Bungee-ML executem as ações de detecção e/ou mitigação. Nota-se que os mecanismos apresentaram variações no volume de pacotes trocados em função do número de pacotes para os diferentes tipos de ataques analisados (Tabela 5.1) e as estratégias desenvolvidas por cada mecanismo para reduzir os impactos causados pelos ataques observados.

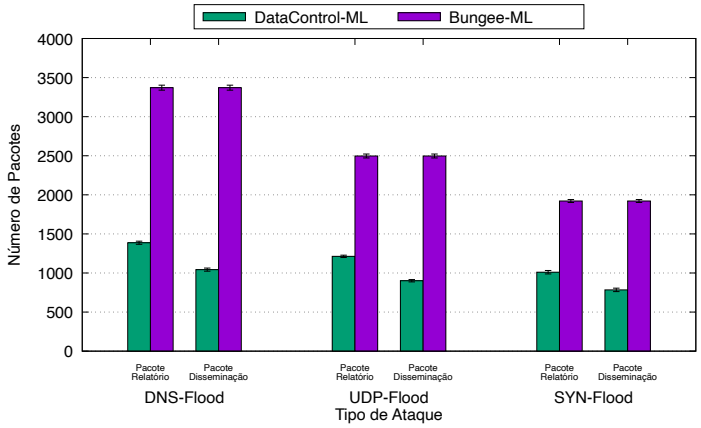


Figura 5.9: Volume de mensagens de controle.

Nota-se na Figura 5.9 que o DataControl-ML apresentou uma redução de $\approx$ 58,31% quando comparado ao Bungee-ML, visto que é adotada uma política de agregação das informações relacionadas aos valores de confiança local dos clientes em um único pacote de relatório e às ações de controle também em um único pacote, agora de disseminação, assim, para cada pacote temos um grupo de clientes reportados ao controlador e ações de controle encaminhadas aos dispositivos de encaminhamento. Essa política reduz o número de pacotes trafegados entre o controlador e o dispositivo de encaminhamento para o estabelecimento da confiança global e o compartilhamento das ações de controle, que podem incluir a priorização ou bloqueio dos clientes em diferentes pontos da rede. Como resultado, a eficiência do mecanismo é significativamente melhorada, pois diminui o volume de mensagens de controle trafegadas no canal de comunicação entre o plano de

dados e de controle, reduzindo a sobrecarga da rede e permitindo uma resposta mais rápida às mudanças na rede. Em contrapartida, o Bungee-ML adota uma política de clonagem individual de cada pacote considerado suspeito, nomeado de pacote de relatório, filtrado no plano de dados para envio ao controlador. Assim, cada pacote clonado contém o endereço IP de origem de uma fonte classificada como suspeita, permitindo que o controlador realize uma análise mais refinada por meio da coleta dos pacotes subsequentes dessa fonte. Após essa análise, o controlador envia uma resposta ao dispositivo de encaminhamento por meio de um pacote de confirmação, nomeado de pacote de disseminação, indicando se a fonte suspeita é maliciosa ou não. Tal abordagem gera um aumento significativo no número de pacotes trafegados e, consequentemente, um aumento do volume de mensagens de controle presentes no canal de comunicação entre os planos e no tempo necessário para realizar as ações de detecção e mitigação, como pode ser observado.

A eficácia da mitigação do DataControl-ML e Bungee-ML no que tange ao método de mitigação empregado pode ser analisada na Figura 5.10. Os mecanismos apresentaram taxas plausíveis de eficácia para reduzir os impactos dos diferentes tipos de ataques executados.

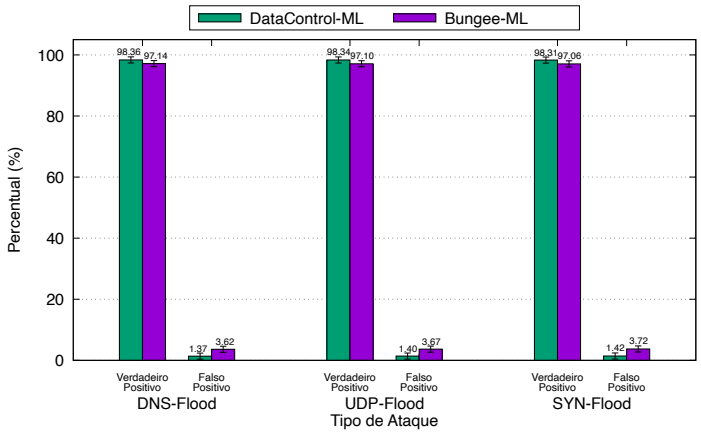
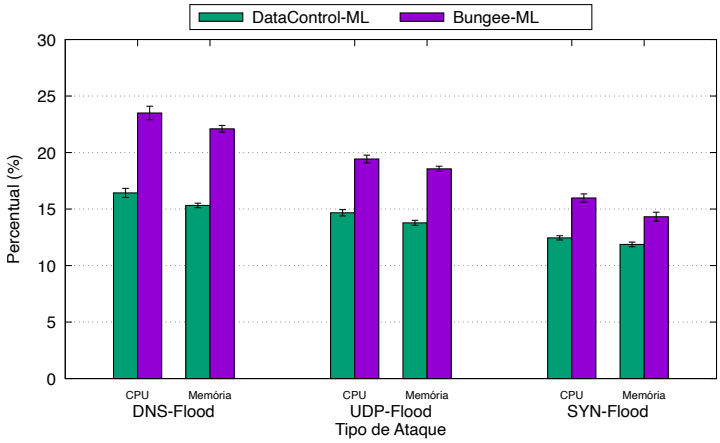


Figura 5.10: Eficácia da mitigação.

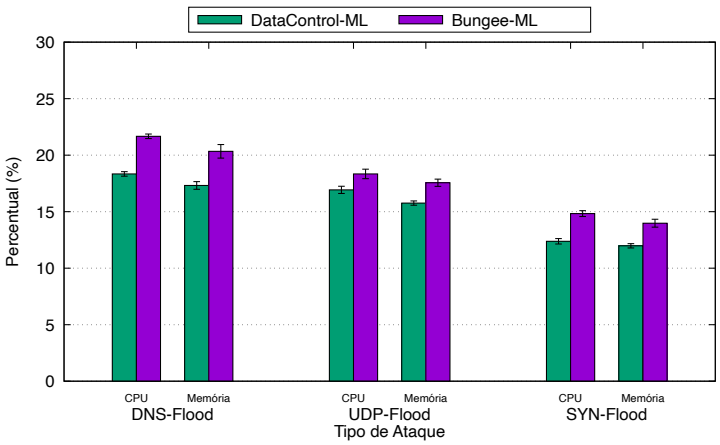
Observa-se na Figura 5.10 que o DataControl-ML, ao priorizar o fluxo de tráfego de rede dos clientes legítimos tendo como base o alto valor de confiança obtido por meio da análise das informações de perfil de tráfego dos clientes e bloquear corretamente o tráfego malicioso (baixo valor de confiança), devido aos altos índices de precisão (Figura 5.6) e rapidez no processo de detecção (Figura 5.7), consegue evitar o bloqueio indiscriminado do tráfego. Assim, reduz a taxa de falso positivo (a proporção de pacotes provenientes de endereços IP legítimos que foram descartados indevidamente) em 61,94% quando comparado ao Bungee-ML, ao mesmo tempo que aumenta a taxa de verdadeiro positivo (a proporção de pacotes provenientes de endereços IP maliciosos que foram realmente

descartados), promovendo um equilíbrio entre o bloqueio do tráfego malicioso e a continuidade do tráfego legítimo, permitindo o mecanismo alcançar uma taxa de verdadeiro positivo superior a 98%. Por outro lado, a filtragem imprecisa dos pacotes suspeitos e o atraso gerado no processo de confirmação das fontes suspeitas reduziram a eficácia de mitigação do Bungee-ML. Essa redução ocorreu porque o mecanismo impede que uma parcela do tráfego da fonte identificada inicialmente como suspeita siga seu caminho até o seu destino, aumentando assim a taxa de falso positivo e reduzindo a taxa de verdadeiro positivo. Como resultado, a eficácia de mitigação do mecanismo ficou abaixo de 98% para a taxa de verdadeiro positivo e acima de 3% para a taxa de falso positivo.

A Figura 5.11 mostra o consumo de recursos (CPU e memória) exigidos pelos mecanismos DataControl-ML e Bungee-ML para a execução das ações de detecção e mitigação aplicadas no combate aos ataques DDoS volumétricos analisados. Observa-se que os mecanismos apresentaram um consumo abaixo de 30%, demonstrando boa eficiência na utilização dos recursos computacionais diante dos cenários de ataque.



(a) Consumo de recursos controlador.



(b) Consumo de recursos *switch*.

Figura 5.11: Consumo de recursos.

Nota-se na Figura 5.11(a) que o DataControl-ML apresentou uma redução de ≈ 24,89% no consumo de recursos do controlador quando comparado ao Bungee-ML. Esse resultado está associado a uma série de otimizações estruturais do mecanismo proposto, como: (*i*) a agregação das informações relacionadas aos valores de confiança local em um único pacote de relatório, o que reduz significativamente o volume de pacotes encaminhados ao plano de controle para a definição da confiança global e a ação de controle; (*ii*) a atribuição de decisões diretamente no plano de dados, permitindo respostas rápidas aos ataques sem a necessidade de encaminhamento imediato ao controlador; e (*iii*) o envio periódico das informações (valores de confiança locais) em janelas de observação, que evita o disparo contínuo de eventos para o controlador, contribuindo para a diminuição da carga de processamento e a preservação dos recursos computacionais no plano de controle. Por outro lado, o consumo mais elevado do Bungee-ML está associado ao seu modelo de funcionamento, no qual o controlador é responsável por confirmar cada suspeita produzida no plano de dados, aumentando o volume de pacotes encaminhados ao dispositivo central, elevando o seu consumo de recursos, como pode ser observado.

Observa-se na Figura 5.11(b) que o DataControl-ML apresentou uma redução no consumo de recursos do *switch* de ≈ 13,21% em comparação com o Bungee-ML, visto que a ação mais rápida de descarte dos pacotes maliciosos por parte do mecanismo proposto em resposta aos ataques e a priorização dos clientes legítimos, reduz o tempo de permanência do tráfego malicioso no dispositivo, otimizando os recursos do dispositivo. Além disso, a adoção de janelas de observação permite o processamento mais eficiente para a coleta de dados de maneira ágil sem comprometer o desempenho da rede. Enquanto, o consumo mais elevado por parte do Bungee-ML está relacionado ao recebimento contínuo de pacotes provenientes de fontes suspeitas, já que o bloqueio é realizado apenas quando há a confirmação por parte do controlador, o que gera um acúmulo de tráfego no *switch* fazendo com que o tráfego malicioso permaneça no dispositivo por mais tempo, e, consequentemente, um aumento no uso de recursos computacionais para gerenciar esse tráfego.

Capítulo 6

Conclusão

As redes SDN, embora ofereçam uma série de vantagens, como o gerenciamento centralizado, maior flexibilidade e o controle da infraestrutura, elas também apresentam diversos desafios relacionados a segurança [8]. Um dos principais desafios está relacionado à como reduzir os impactos causados por ataques DDoS [9]. Assim, torna-se essencial projetar mecanismos que assegurem a escalabilidade e a resiliência da rede, contribuindo para torná-la mais segura e estável no combate a esses ataques.

A literatura tem apresentado diversas estratégias para reduzir os impactos dos ataques DDoS volumétricos em redes SDN [10]. No entanto, ainda persistem algumas lacunas que podem deixar essas redes vulneráveis, como a centralização das ações de detecção e mitigação no plano de controle, que pode provocar atrasos na detecção e confirmação de mudanças no comportamento do tráfego de rede e o aumento do volume de mensagens de controle encaminhadas ao controlador para realizar as medidas de identificação e contenção desses ataques a rede, aumentando o seu consumo de recursos. Além disso, muitas das abordagens desenvolvidas pelas soluções para mitigar os impactos desses ataques, que, embora visem restringir o tráfego malicioso, acabam penalizando uma parte significativa do tráfego legítimo, provocando bloqueios de maneira indiscriminada.

Este trabalho buscou suprir as lacunas presentes nas soluções de segurança analisadas, como a centralização das ações de detecção e mitigação no plano de controle e o bloqueio indiscriminado do tráfego de rede de clientes legítimos. Para isso, foi proposto um mecanismo de detecção e mitigação no plano de dados, denominado DataControl-ML, para permitir que à rede reaja rapidamente para conter o fluxo de tráfego de clientes maliciosos e priorize o fluxo de tráfego de clientes legítimos, para evitar bloqueios indiscriminados e reduzir o volume de mensagens de controle encaminhadas ao controlador, preservando os seus recursos, evitando a sobrecarga e garantindo sua disponibilidade mesmo durante situações de ataque. Além disso, o mecanismo busca integrar os planos da arquitetura SDN (dados e controle) para reduzir o impacto desses ataques por meio de um modelo de

compartilhamento de informações globais no plano de controle, organizadas em ações de controle que incluem o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes, para promover uma colaboração eficiente entre os planos e maior rapidez nos processos de detecção e mitigação em diferentes pontos da rede.

O DataControl-ML demonstrou capacidade de identificar e conter os fluxos de tráfego maliciosos relacionados aos ataques DDoS volumétricos diretamente no plano de dados, e assim reduzir os atrasos no processo de confirmação de qualquer atividade suspeita associada a esses ataques e o volume de mensagens de controle encaminhadas ao controlador, preservando os seus recursos, bem como a priorização de clientes com níveis de confiabilidade aceitáveis (legítimos) para evitar o bloqueio indiscriminado. Para isso, o mecanismo combina uma abordagem de detecção mediante um modelo de classificação com base no algoritmo de aprendizagem de máquina *Random Forest*, capaz de realizar a classificação do tráfego em tempo real, com alta precisão e baixo custo computacional. Seguido, por um método de mitigação mediante o gerenciamento de diferentes listas nos dispositivos de encaminhamento, utilizando níveis de confiabilidade, para a priorização dos fluxos confiáveis (legítimos) e o bloqueio daqueles não confiáveis (maliciosos) e, assim, evitar o bloqueio indiscriminado dos clientes.

Além das abordagens no plano de dados para a detecção e mitigação dos ataques DDoS volumétricos, também foi proposto um modelo de compartilhamento de informações globais no plano de controle, que busca integrar os planos da arquitetura SDN (dados e controle) para que os dispositivos de encaminhamento presentes no plano de dados tomem decisões locais, baseando-se em uma visão ampliada da rede, construída a partir de dados compartilhados pelo controlador. Os dados compartilhados incluem ações de controle que realizam medidas como o bloqueio, a permissão sem prioridade associada ou a priorização dos clientes. Essas ações são determinadas a partir do estabelecimento de uma confiança global, calculada por meio de um sistema *fuzzy*, possibilitando criar uma visão única da rede e uma mitigação mais abrangente, coordenada e capaz de proteger a infraestrutura em diferentes pontos.

A avaliação do DataControl-ML foi realizada por meio de simulações no emulador de rede Mininet [50], considerando a aplicação de diferentes tipos de ataques DDoS volumétricos, como DNS-*Flood*, UDP-*Flood* e SYN-*Flood* presentes no conjunto de dados CICDDoS-2019 [128], para avaliar o desempenho do mecanismo em condições adversas. Os resultados obtidos foram comparados com uma solução representativa para o estado da arte, o Bungee-ML [18]. O DataControl-ML apresentou uma redução de $\approx$ 52,18% no tempo necessário para a confirmação (detecção) de uma fonte maliciosa, responsável por gerar os ataques DDoS volumétricos, quando comparado ao Bungee-ML. Além disso, o mecanismo mostrou-se mais eficiente na sincronização das informações entre os

dispositivos da rede e na preservação dos recursos (CPU e memória) do controlador e do *switch*, diminuindo o tempo de convergência e reduzindo significativamente o volume de mensagens de controle enviadas ao controlador. Outro destaque foi a eficácia da detecção e mitigação alcançada, sendo superior a 98%, evidenciando a capacidade do mecanismo em bloquear o tráfego malicioso, sem comprometer a continuidade do tráfego legítimo. Tais resultados, evidenciam o potencial do DataControl-ML como uma solução eficaz e avançada para a detecção e mitigação de ataques DDoS volumétricos em redes SDN.

6.1 Trabalhos Futuros

Como trabalhos futuros, propõe-se a ampliação do mecanismo para combater os ataques DDoS não volumétricos, atribuindo essa responsabilidade ao controlador, enquanto os dispositivos de encaminhamento permanecem responsáveis pelo tratamento dos ataques DDoS volumétricos diretamente no plano de dados. Essa abordagem visa aprimorar a eficácia na resposta a diferentes tipos de ameaças, distribuindo as responsabilidades de forma estratégica entre os elementos da arquitetura SDN.

Outro aspecto a ser explorado é a utilização de novos modelos de treinamento para atualização dinâmica dos classificadores de tráfego, considerando o uso de técnicas de aprendizado profundo e aprendizado não supervisionado, visando lidar com padrões de ataque mais complexos e em constante evolução. Nesse contexto, também se propõe o estudo do uso de *honeypots* como fonte de dados reais, a fim de enriquecer o processo de treinamento e refinar os critérios de detecção de tráfego malicioso.

Além disso, planeja-se a integração do mecanismo com múltiplos controladores, para fornecer estratégias de cooperação entre domínios, promovendo a troca coordenada de informações e a tomada de decisões distribuídas. Essa abordagem visa melhorar a escalabilidade da solução, garantir maior consistência no tratamento de ataques em ambientes amplos, e fortalecer a resiliência da rede frente a ameaças coordenadas que afetam múltiplas regiões simultaneamente.

Por fim, para fortalecer a avaliação do mecanismo proposto, pretende-se ampliar os cenários experimentais, incluindo o uso de diferentes conjuntos de dados e a realização de testes em ambiente real, visando validar o desempenho e a viabilidade do mecanismo em contextos físicos, explorando os desafios práticos de implementação. Assim, espera-se aprimorar a segurança e a resiliência do mecanismo, proporcionando um desempenho mais robusto e adaptável em ambientes cada vez mais complexos e dinâmicos.

Referências

[1] Bosshart, P., D. Daly, G. Gibb, M. Izzard, N. McKeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese e D. Walker: *P4: Programming Protocol-independent Packet Processors*. SIGCOMM Comput. Commun. Rev., 44(3):87–95, jul 2014, ISSN 0146-4833. `http://doi.acm.org/10.1145/2656877.2656890`. xii, 12, 13, 27, 47

[2] Alsaeedi, Mohammed, Mohd Murtadha Mohamad e Anas A. Al-Roubaiey: *Toward adaptive and scalable openflow-sdn flow control: A survey*. IEEE Access, 7:107346–107379, 2019. 1, 8, 9, 10, 11, 12, 13

[3] Cox, Jacob H., Joaquin Chung, Sean Donovan, Jared Ivey, Russell J. Clark, George Riley e Henry L. Owen: *Advancing software-defined networks: A survey*. IEEE Access, 5:25487–25526, 2017. 1, 8, 9, 10, 15

[4] Presuhn, Randy: *Management information base (mib) for the simple network management protocol (snmp)*. Request for comments 3418, RFC Editor, dezembro 2002. `https://www.rfc-editor.org/rfc/rfc3418.html`. 1

[5] Park, Chang Keen, Joon Myung Kang, Mi Jung Choi, James Won Ki Hong, Yong hun Lim, Seongho Ju e Moon suk Choi: *Definition of common plc mib and design of mib mapper for multi-vendor plc network management*. Em *2008 IEEE International Symposium on Power Line Communications and Its Applications*, páginas 152–157, 2008. 1

[6] Bhatia, Jitendra, Radha Govani e Madhuri Bhavsar: *Software defined networking: From theory to practice*. Em *2018 Fifth International Conference on Parallel, Distributed and Grid Computing (PDGC)*, páginas 789–794, 2018. 1, 8, 9

[7] Mostafa, Naneese, Khaled Metwally e Khaled Badran: *Survey on sdn-based intrusion detection systems*. Em *2024 14th International Conference on Electrical Engineering (ICEENG)*, páginas 317–322, 2024. 2, 41

[8] Abdi, Abdinasir Hirsi, Lukman Audah, Adeb Salh, Mohammed A. Alhartomi, Haroon Rasheed, Salman Ahmed e Ahmed Tahir: *Security control and data planes of sdn: A comprehensive review of traditional, ai, and mtd approaches to security solutions*. IEEE Access, 12:69941–69980, 2024. 2, 8, 41, 91

[9] Varghese, Josy Elsa e Balachandra Muniyal: *Trend in sdn architecture for ddos detection- a comparative study*. Em *2021 IEEE International Conference on Dis-*

tributed Computing, VLSI, Electrical Circuits and Robotics (DISCOVER), páginas 170–174, 2021. 2, 41, 91

[10] Kaur, Sukhveer, Krishan Kumar, Naveen Aggarwal e Gurdeep Singh: *A comprehensive survey of DDoS defense solutions in SDN: Taxonomy, research challenges, and future directions*. Computers & Security, 110:102423, 2021, ISSN 0167-4048. https://www.sciencedirect.com/science/article/pii/S0167404821002479. 2, 3, 19, 26, 27, 32, 34, 37, 38, 39, 91

[11] Wang, Danni e Sizhao Li: *Automated ddos attack mitigation for software defined network*. Em *2022 IEEE 16th International Conference on Anti-counterfeiting, Security, and Identification (ASID)*, páginas 100–104, 2022. 2

[12] Shin, S., Y. Vinod, P. Phillip e G. Guofei: *AVANT-GUARD: Scalable and Vigilant Switch Flow Management in Software-defined Networks*. Em *Proceedings of the 2013 ACM SIGSAC Conference on Computer & Communications Security*, CCS '13, páginas 413–424, New York, NY, USA, 2013. ACM, ISBN 978-1-4503-2477-9. http://doi.acm.org/10.1145/2508859.2516684. 3, 32, 37, 38

[13] Ambrosin, M., M. Conti, F. De Gaspari e R. Poovendran: *LineSwitch: Tackling Control Plane Saturation Attacks in Software-Defined Networking*. IEEE/ACM Transactions on Networking, 25(2):1206–1219, abril 2017, ISSN 1063-6692. 3, 32, 37, 38

[14] Lapolli, Angelo Cardoso, Jonatas Adilson Marques e Luciano Paschoal Gaspary: *Offloading Real-time DDoS Attack Detection to Programmable Data Planes*. Em *IFIP*. IFIP/IEEE Symposium on Integrated Network and Service Management (IM), 2019. 3, 37, 38

[15] Zhang, Xiaoquan, Lin Cui, Kaimin Wei, Fung Po Tso, Yangyang Ji e Weijia Jia: *A survey on stateful data plane in software defined networks*. Computer Networks, 184:107597, 2021, ISSN 1389-1286. https://www.sciencedirect.com/science/article/pii/S1389128620312305. 3, 4, 38, 39

[16] Macías, Sebastián Gómez, Luciano Paschoal Gaspary e Juan Felipe Botero: *ORACLE: An Architecture for Collaboration of Data and Control Planes to Detect DDoS Attacks*. Em *2021 IFIP/IEEE International Symposium on Integrated Network Management (IM)*, páginas 962–967, 2021. 3, 34, 38, 43

[17] Yu, Shanshan, Jicheng Zhang, Ju Liu, Xiaoqing Zhang, Yafeng Li e Tianfeng Xu: *A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN*. EURASIP Journal on Wireless Communications and Networking, 2021(1), abril 2021. https://doi.org/10.1186/s13638-021-01957-9. 3, 29, 32, 34, 38

[18] González, Libardo Andrey Quintero, Lucas Castanheira, Jonatas A. Marques, Alberto E. Schaeffer-Filho e Luciano Paschoal Gaspary: *Bungee-ML: A Cross-Plane Approach for a Collaborative Defense Against DDoS Attacks*. Journal of Network and Systems Management, 31(4), agosto 2023. https://doi.org/10.1007/s10922-023-09769-6. 3, 32, 34, 38, 69, 75, 92

[19] Imran, M., H. Durad, F. Khan e A. Derhab: *Toward an optimal solution against Denial of Service attacks in Software Defined Networks*. Future Generation Computer Systems, 92, setembro 2018. 3, 24, 41

[20] Tuan, Nguyen Ngoc, Pham Huy Hung, Nguyen Danh Nghia, Nguyen Van Tho, Trung V. Phan e Nguyen Huu Thanh: *A Robust TCP-SYN Flood Mitigation Scheme Using Machine Learning Based on SDN*. Em *2019 International Conference on Information and Communication Technology Convergence*, páginas 363–368. International Conference on Information and Communication Technology Convergence (ICTC), 2019. 3, 4, 34, 38, 39, 43

[21] Kumar, P., M. Tripathi, A. Nehra, M. Conti e C. Lal: *SAFETY: Early Detection and Mitigation of TCP SYN Flood Utilizing Entropy in SDN*. IEEE Transactions on Network and Service Management, 15(4):1545–1559, 2018. 3, 4, 27, 28, 30, 38, 39

[22] Salem, Fatty M., Hoda Youssef, Ihab Ali e Ayman Haggag: *A variable-trust threshold-based approach for DDOS attack mitigation in software defined networks*. PLOS ONE, 17(8):1–19, agosto 2022. `https://doi.org/10.1371/journal.pone.0273681`. 3, 4, 30, 38, 39

[23] Saini, Chiman e Sandeep Singh Kang: *Trust based Mechanism for Detection and Isolation of DDOS Attack in Software Defined Networks*. Em *2021 3rd International Conference on Advances in Computing, Communication Control and Networking (ICAC3N)*, páginas 1254–1259, 2021. 3, 29, 30, 38, 39

[24] Kuerban, M., Y. Tian, Q. Yang, Y. Jia, B. Huebert e D. Poss: *FlowSec: DOS Attack Mitigation Strategy on SDN Controller*. Em *2019 IEEE 9th Annual Computing and Communication Workshop and Conference (CCWC)*, páginas 701–707, Aug 2019. 4, 28, 38, 39

[25] Nurwarsito, Heru e Muhammad Fahmy Nadhif: *Ddos attack early detection and mitigation system on sdn using random forest algorithm and ryu framework*. Em *2021 8th International Conference on Computer and Communication Engineering (ICCCE)*, páginas 178–183, 2021. 4, 35, 38

[26] Dalmazo, Bruno L., Jonatas A. Marques, Lucas R. Costa, Michel S. Bonfim, Ranyelson N. Carvalho, Anderson S. da Silva, Stenio Fernandes, Jacir L. Bordim, Eduardo Alchieri, Alberto Schaeffer-Filho, Luciano Paschoal Gaspary e Weverton Cordeiro: *A systematic review on distributed denial of service attack defense mechanisms in programmable networks*. International Journal of Network Management, 31(6):e2163, 2021. `https://onlinelibrary.wiley.com/doi/abs/10.1002/nem.2163`. 6

[27] Carvalho, Ranyelson N., Lucas R. Costa, Jacir L. Bordim e Eduardo Alchieri: *Dossec: A reputation-based dos mitigation mechanism on sdn*. Em Barolli, Leonard, Isaac Woungang e Tomoya Enokido (editores): *Advanced Information Networking and Applications*, páginas 757–770, Cham, 2021. Springer International Publishing, ISBN 978-3-030-75075-6. 6

[28] Carvalho, Ranyelson N., Lucas R. Costa, Jacir L. Bordim e Eduardo A. P. Alchieri: *Detecting ddos attacks on sdn data plane with machine learning.* Em *2021 Ninth International Symposium on Computing and Networking Workshops (CANDARW)*, páginas 138–144. 2021 Ninth International Symposium on Computing and Networking Workshops (CANDARW), 2021. 6

[29] Carvalho, Ranyelson N., Lucas R. Costa, Jacir L. Bordim e Eduardo A. P. Alchieri: *Dataplane-ml: An integrated attack detection and mitigation solution for software defined networks.* Concurrency and Computation: Practice and Experience, n/a(n/a):e7434, 2022. `https://onlinelibrary.wiley.com/doi/abs/10.1002/cpe.7434`. 6, 72

[30] Carvalho, R., J. L. Bordim e E. A. P. Alchieri: *Entropy-based dos attack identification in sdn.* 21st Workshop on Advances in Parallel and Distributed Computational Models, 2019. 7

[31] Carvalho, R., L. R. Costa, J. L. Bordim e E. A. P. Alchieri: *Enhancing an sdn architecture with dos attack detection mechanisms.* Advances in Science, Technology and Engineering Systems Journal, 2020. 7

[32] Carvalho, R., L. R. Costa, J. L. Bordim e E. A. P. Alchieri: *New programmable data plane architecture based on p4 openflow agent.* 34th Advanced Information Networking and Applications, 2020. 7

[33] Foundation, Open Network: *Openflow switch specification 1.1.* fevereiro 2011. 9

[34] Chowdhury, D.: *ForCES: An Elastic Routing Architecture for NextGen SDN.* Network Virtualization 101 Series, agosto 2016. 9, 10

[35] Prabha, Chander, Anjuli Goel e Jaspreet Singh: *A survey on sdn controller evolution: A brief review.* Em *2022 7th International Conference on Communication and Electronics Systems (ICCES)*, páginas 569–575, 2022. 10

[36] Ravuri, Hemanth Kumar, Maria Torres Vega, Jeroen van der Hooft, Tim Wauters, Bin Da e Filip De Turck: *On routing scalability in flat sdn architectures.* Em *2020 11th International Conference on Network of the Future (NoF)*, páginas 23–27, 2020. 10, 15

[37] Bannour, Fetia, Stefania Dumbrava e Damien Lu: *A flexible graphql northbound api for intent-based sdn applications.* Em *NOMS 2022-2022 IEEE/IFIP Network Operations and Management Symposium*, páginas 1–5, 2022. 10

[38] Oktian, Yustus Eko, SangGon Lee, HoonJae Lee e JunHuy Lam: *Secure your Northbound SDN API.* Em *2015 Seventh International Conference on Ubiquitous and Future Networks*, páginas 919–920, 2015. 10

[39] Gude, N., T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown e S. Shenker: *NOX: Towards an operating system for networks.* Computer Communication Review, 38:105–110, janeiro 2008. 11

[40] Murpyh, M.: *NOX*. https://github.com/noxrepo/pox. 11, 15

[41] Daha, Muhammad Yunis, Mohd Soperi M Zahid, Khaleel Husain e Firas Ousta: *Performance evaluation of software defined networks with single and multiple link failure scenario under floodlight controller*. Em *2021 International Conference on Computing, Communication, and Intelligent Systems (ICCCIS)*, páginas 959–965, 2021. 11

[42] Projects, The Linux Foundation: *Open Day Light - Platform Overview*. https://www.opendaylight.org/what-we-do/odl-platform-overview. 11, 15

[43] RYU: *RYU SDN Framework*. https://osrg.github.io/ryu/. 11

[44] Stancu, A., A. Avram, M. Skorupski, A. Vulpe e S. Halunga: *Enabling SDN application development using a NETCONF mediator layer simulator*. Em *2017 Ninth International Conference on Ubiquitous and Future Networks (ICUFN)*, páginas 658–663, July 2017. 11, 12

[45] ONOS Project: *ONOS*. https://wiki.onosproject.org/display/ONOS/ONOS. 12, 15

[46] Martinez-Yelmo, I., J. Alvarez-Horcajo, M. Briso-Montiano, D. Lopez-Pajares e E. Rojas: *ARP-P4: Deep Analysis of a Hybrid SDN ARP-Path/P4Runtime Switch*. Telecommunication Systems, 72(4):555–565, 2019. 12

[47] Miano, S., F. Risso e H. Woesner: *Partial offloading of OpenFlow rules on a traditional hardware switch ASIC*. Em *2017 IEEE Conference on Network Softwarization (NetSoft)*, páginas 1–9, July 2017. 12

[48] NSnam: *Ns-3 - Network Simulator*. https://www.nsnam.org/. 14

[49] Estinet: *Estinet - The network simulator and emulator that supports SDN/OpenFlow*. http://www.estinet.com/ns/. 15

[50] Team, Mininet: *Mininet - An Instant Virtual Network on your Laptop (or other PC)*. http://mininet.org/. 15, 69, 92

[51] Rawat, Surabhi Gusain, Mohammad S. Obaidat, Sumit Pundir, Mohammad Wazid, Ashok Kumar Das, Devesh Pratap Singh e Kuei Fang Hsiao: *A survey of ddos attacks detection schemes in sdn environment*. Em *2023 International Conference on Computer, Information and Telecommunication Systems (CITS)*, páginas 01–06, 2023. 15

[52] Al-Duwairi, Basheer, Eslam Al-Quraan e Yazeed AbdelQader: *Isdsdn: Mitigating syn flood attacks in software defined networks*. Journal of Network and Systems Management, 28(4):1366–1390, junho 2020, ISSN 1573-7705. http://dx.doi.org/10.1007/s10922-020-09540-1. 15

[53] Kaur, Kulwinder e John Ayoade: *Analysis of ddos attacks on iot architecture*. Em *2023 10th International Conference on Electrical Engineering, Computer Science and Informatics (EECSI)*, páginas 332–337, 2023. 16

[54] Yevsieieva, O. e S. M. Helalat: *Analysis of the impact of the slow HTTP DOS and DDOS attacks on the cloud environment*. Em *2017 4th International Scientific-Practical Conference Problems of Infocommunications. Science and Technology (PIC S T)*, páginas 519–523, outubro 2017. 16, 18

[55] Gondim, João J.C., Robson de Oliveira Albuquerque e Ana Lucila Sandoval Orozco: *Mirror saturation in amplified reflection Distributed Denial of Service: A case of study using SNMP, SSDP, NTP and DNS protocols*. Future Generation Computer Systems, 2020, ISSN 0167-739X. `http://www.sciencedirect.com/science/article/pii/S0167739X19322745`. 16

[56] Feinstein, L., D. Schnackenberg, R. Balupari e D. Kindred: *Statistical approaches to DDoS attack detection and response*. Em *Proceedings DARPA Information Survivability Conference and Exposition*, volume 1, páginas 303–314 vol.1, abril 2003. 16, 19

[57] Hussain, Khalid, Syed Jawad Hussain, NZ Jhanjhi e Mamoona Humayun: *Syn flood attack detection based on bayes estimator (sfadbe) for manet*. Em *2019 International Conference on Computer and Information Sciences (ICCIS)*, páginas 1–4, 2019. 16

[58] Biagioni, Edoardo: *Preventing udp flooding amplification attacks with weak authentication*. Em *2019 International Conference on Computing, Networking and Communications (ICNC)*, páginas 78–82, 2019. 16, 17

[59] Mekala, Srinivas, Kishorebabu Dasari e Divya Katta: *Dns ddos amplification attack detection using multi-layer perceptron classification algorithm*. Em *2024 IEEE 3rd World Conference on Applied Intelligence and Computing (AIC)*, páginas 1355–1360, 2024. 16, 17

[60] Postel, J.: *User Datagram Protocol*. RFC, 768:1–3, agosto 1980. `http://dblp.uni-trier.de/db/journals/rfc/rfc700-799.html`. 17

[61] Mahjabin, Tasnuva, Yang Xiao, Tieshan Li e C. L. Philip Chen: *Load distributed and benign-bot mitigation methods for iot dns flood attacks*. IEEE Internet of Things Journal, 7(2):986–1000, 2020. 17

[62] Lyu, Minzhao, Hassan Habibi Gharakheili, Craig Russell e Vijay Sivaraman: *Hierarchical anomaly-based detection of distributed dns attacks on enterprise networks*. IEEE Transactions on Network and Service Management, 18(1):1031–1048, 2021. 17

[63] *Internet Control Message Protocol*. RFC 792, setembro 1981. `https://rfc-editor.org/rfc/rfc792.txt`. 17

[64] Nielsen, Henrik, Jeffrey Mogul, Larry M Masinter, Roy T. Fielding, Jim Gettys, Paul J. Leach e Tim Berners-Lee: *Hypertext Transfer Protocol – HTTP/1.1*. RFC 2616, junho 1999. `https://www.rfc-editor.org/info/rfc2616`. 18

[65] Sabri, Shima, Noraini Ismail e Amir Hazzim: *Slowloris DoS Attack Based Simulation*. IOP Conference Series: Materials Science and Engineering, 1062(1):012029, fevereiro 2021. `https://doi.org/10.1088/1757-899x/1062/1/012029`. 18

[66] S. Lee, H. Kim, J. Na e J. Jang: *Abnormal traffic detection and its implementation.* Em *The 7th International Conference on Advanced Communication Technology, 2005, ICACT 2005.*, volume 1, páginas 246–250, fevereiro 2005. 19

[67] Carl, G., G. Kesidis, R. Brooks e R. Rai: *Denial-of-Service Attack-Detection Techniques.* IEEE Internet Computing, 10(1):82–89, janeiro 2006, ISSN 1089-7801. http://dx.doi.org/10.1109/MIC.2006.5. 19

[68] Shannon, C.E: *A Mathematical Theory of Communication.* 27:379–423, 1948. 20, 28

[69] Behal, Sunny e Krishan Kumar: *Detection of DDoS attacks and flash events using novel information theory metrics.* Computer Networks, 116:96–110, 2017, ISSN 1389-1286. 20

[70] Agoramoorthy, Moorthy, Ahamed Ali, D. Sujatha, Michael Raj. T F e G. Ramesh: *An analysis of signature-based components in hybrid intrusion detection systems.* Em *2023 Intelligent Computing and Control for Engineering and Business Systems (ICCEBS)*, páginas 1–5, 2023. 20

[71] Montes-Gil, José Albeiro, Gustavo Isaza-Cadavid e Néstor Darío Duque-Méndez: *Efecto de la selección de atributos en el desempeño de un IDS basado en machine learning para detección de intrusos en ataques DDoS.* South Florida Journal of Development, 4(2):918–928, maio 2023. https://doi.org/10.46932/sfjdv4n2-023. 21

[72] Elejla, Omar E., Mohammed Anbar, Shady Hamouda, Bahari Belaton, Taief Alaa Al-Amiedy e Iznan H. Hasbullah: *Flow-Based IDS Features Enrichment for ICMPv6-DDoS Attacks Detection.* Symmetry, 14(12):2556–2556, 2022. 21

[73] Jiawei, Chen, Xie Wenwei e Wang Lipeng: *Data proxy method and system, and proxy server.* 2021. 21

[74] You Chwen Yeu, Gabriel de Souza Fedel: *Aceleração no acesso à Internet: estudo sobre o servidor 12 proxy/cache Squid.* evista Tecnológica da Fatec Americana, 2(1):12–34, mar 2016. 21

[75] Sahoo, Kshira Sagar, Bata Krishna Tripathy, Kshirasagar Naik, Somula Ramasubbareddy, Balamurugan Balusamy, Manju Khari e Daniel Burgos: *An Evolutionary SVM Model for DDOS Attack Detection in SDN.* IEEE Access, 8:132502–132513, 2020. 21

[76] Müller, A.C. e S. Guido: *Introduction to Machine Learning with Python: A Guide for Data Scientists.* O'Reilly Media, 2016, ISBN 9781449369897. 21, 50

[77] Aggarwal, Charu C.: *Machine Learning for Text.* Springer Publishing Company, Incorporated, 1st edição, 2018, ISBN 3319735306. 22, 50

[78] Wu, Xindong e Shichao Zhang: *Synthesizing high-frequency rules from different data sources.* IEEE Transactions on Knowledge and Data Engineering, 15(2):353–367, 2003. 22

[79] Coretes, C. e V. Vapnik: *Support-vector networks*. Machine Learning, 28:273–297, 1995. 22, 50

[80] Smola, Alex e B. Schölkopf: *A tutorial on support vector regression*. Statistics and Computing, 14:199–222, 2004. 22

[81] Bishop, Christopher M.: *Pattern Recognition and Machine Learning (Information Science and Statistics)*. Springer-Verlag, Berlin, Heidelberg, 2006, ISBN 0387310738. 23

[82] Ns, Abdiansah: *Time complexity analysis of support vector machines (svm) in libsvm*. Int. Journal of Comp. Apps., 128:975–8887, outubro 2015. 23

[83] Breiman, Leo: *Random Forests*. Machine Learning, 45, 2001, ISSN 0885-6125. 23, 50, 73

[84] Louppe, Gilles: *Understanding Random Forests: Theory to Practice*, 2015. 23

[85] Bawany, N.Z., Shamsi, J.A. e Salah K.: *DDoS Attack Detection and Mitigation Using SDN: Methods, Practices, and Solutions*. Arab J Sci Eng, 42:425–441, feb 2017. 24

[86] *IEEE Xplore*. `https://ieeexplore.ieee.org/`. 26

[87] *ACM Digital Libray*. `https://dl.acm.org/`. 26

[88] Lapolli, A. C., J. A. Marques e L. P. Gaspary: *Offloading Real-time DDoS Attack Detection to Programmable Data Planes*. Em *IFIP/IEEE International Symposium on Integrated Network Management (IM 2019)*, 2019. 27

[89] Sumantra, I. e S. Indira Gandhi: *DDoS attack Detection and Mitigation in Software Defined Networks*. Em *2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, páginas 1–5, 2020. 28, 30, 38

[90] Li, Runyu e Bin Wu: *Early detection of ddos based on φ-entropy in sdn networks*. Em *2020 IEEE 4th Information Technology, Networking, Electronic and Automation Control Conference (ITNEC)*, volume 1, páginas 731–735, 2020. 28, 38

[91] Ravi, Nagarathna, S. Mercy Shalinie, Chhagan Lal e Mauro Conti: *AEGIS: Detection and Mitigation of TCP SYN Flood on SDN Controller*. IEEE Transactions on Network and Service Management, 18(1):745–759, 2021. 29, 38

[92] Mishra, Anupama, Neena Gupta e B. B. Gupta: *Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller*. Telecommunication Systems, 77(1):47–62, janeiro 2021. `https://doi.org/10.1007/s11235-020-00747-w`. 30, 38

[93] Dawod, Ammar, Huda S. Abdulkarem e Aymen M. Al-Kadhimi: *Software-defined Network with SNMP Monitoring Sensor Against SYN Flooding Attack*. Em *2022 3rd International Informatics and Software Engineering Conference (IISEC)*, páginas 1–5, 2022. 31, 38

[94] Shalini, P. V., V. Radha e Sriram G. Sanjeevi: *Early detection and mitigation of tcp syn flood attacks in sdn using chi-square test*. The Journal of Supercomputing, fevereiro 2023, ISSN 1573-0484. http://dx.doi.org/10.1007/s11227-023-05057-x. 31, 38

[95] Dridi, L. e M. F. Zhani: *Sdn-guard: Dos attacks mitigation in sdn networks*. Em *2017 5th IEEE International Conference on Cloud Networking (Cloudnet)*, páginas 212–217, outubro 2017. 33, 38

[96] Kim, DongHyuk, Phuc Trinh Dinh, Sichul Noh, Junmin Yi e Minho Park: *An Effective Defense Against SYN Flooding Attack in SDN*. Em *2019 International Conference on Information and Communication Technology Convergence (ICTC)*, páginas 369–371. International Conference on Information and Communication Technology Convergence (ICTC), 2019. 33, 38

[97] Fan, Chun I, Jun Huei Wang, Cheng Han Shie e Yu Lung Tsai: *Software-Defined Networking Integrated with Cloud Native and Proxy Mechanism: Detection and Mitigation System for TCP SYN Flooding Attack*. Em *2023 17th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, páginas 1–8, 2023. 33, 38

[98] https://kubernetes.io/pt-br/. 33

[99] Das, Tapadhir, Osama Abu Hamdan, Shamik Sengupta e Engin Arslan: *Flood control: Tcp-syn flood detection for software-defined networks using openflow port statistics*. Em *2022 IEEE International Conference on Cyber Security and Resilience (CSR)*, páginas 1–8, 2022. 35, 38

[100] Ashodia, Namita e Kishan Makadiya: *Detection and mitigation of ddos attack in software defined networking: A survey*. Em *2022 International Conference on Sustainable Computing and Data Communication Systems (ICSCDS)*, páginas 1175–1180, 2022. 41

[101] Dang, V. T., T. T. Huong, N. H. Thanh, P. N. Nam, N. N. Thanh, A. Marshall e S. Furnell: *SDN-Based SYN Proxy—A Solution to Enhance Performance of Attack Mitigation Under TCP SYN Flood*. The Computer Journal, 62(4):518–534, April 2019. 41

[102] Zadeh, L.A.: *Fuzzy sets*. Information and Control, 8(3):338–353, 1965, ISSN 0019-9958. https://www.sciencedirect.com/science/article/pii/S001999586590241X. 45, 61, 62

[103] Kurose, James F. e Keith W. Ross: *Computer Networking: A Top-Down Approach*. Pearson, 8th edição, 2021. 45

[104] Consortium, The P4 Language: *P4 - language specification*. https://p4.org/. 47, 48, 57

[105] M. Slee, A. Agawal, M. Kwiatkowski: *Thrift - scalabre cross-language services implementation*. https://thrift.apache.org/. 49

[106] Carvalho, Ranyelson, Lucas Costa, Jacir Bordim e Eduardo Alchieri: *New programmable data plane architecture based on p4 openflow agent*. Em *AINA*, páginas 1355–1367. Springer Int. Publishing, 2020, ISBN 978-3-030-44041-1. 49

[107] Learn, Scikt: *Sklearn model selection: RandomizedSearchCV*, 2022. `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.RandomizedSearchCV.html`. 50, 73

[108] Platt, John C.: *Probabilistic outputs for support vector machines and comparisons to regularized likelihood methods*. Em *Advances in Large Margin Classifiers*, páginas 61–74. MIT Press, 1999. 50, 52

[109] Deo, Amit, Santanu Kumar Dash, Guillermo Suarez-Tangil, Volodya Vovk e Lorenzo Cavallaro: *Prescience: Probabilistic guidance on the retraining conundrum for malware detection*. Em *Proceedings of the 2016 ACM Workshop on Artificial Intelligence and Security*, AISec '16, página 71–82, New York, NY, USA, 2016. Association for Computing Machinery, ISBN 9781450345736. `https://doi.org/10.1145/2996758.2996769`. 50, 51

[110] Ryszard S. Michalski, Jaime G. Carbonell e Tom M. Mitchell: *Machine Learning: An Artifcial Intelligence Approach*, volume 1. Springer, 1983. 51

[111] Lopes Gomes, Rafael, Waldir Moreira Junior, Eduardo Cerqueira e Antônio Jorge Abelém: *Using fuzzy link cost and dynamic choice of link quality metrics to achieve qos and qoe in wireless mesh networks*. Journal of Network and Computer Applications, 34(2):506–516, 2011, ISSN 1084-8045. `https://www.sciencedirect.com/science/article/pii/S1084804510000615`, Efficient and Robust Security and Services of Wireless Mesh Networks. 61, 74

[112] Zadeh, L.A.: *The concept of a linguistic variable and its application to approximate reasoning—i*. Information Sciences, 8(3):199–249, 1975, ISSN 0020-0255. `https://www.sciencedirect.com/science/article/pii/0020025575900365`. 62

[113] Malvezzi, W. R., A. M. Mourao e G. Bressan: *Learning evaluation in classroom mediated by technology model using fuzzy logic at the university of amazonas state*. Em *Proceeding of 40th ASEE/IEEE Frontiers in Education Conference*, páginas S2C–1–S2C–6, Washington, DC, October 2010. 62

[114] Ross, Timothy J.: *Fuzzy Logic with Engineering Applications*. John Wiley & Sons, 3rd edição, 2010. `https://onlinelibrary.wiley.com/doi/book/10.1002/9781119994374`. 62, 64, 74

[115] Zimmermann, H. J.: *Fuzzy Set Theory—and Its Applications*. Kluwer Academic Publishers, Boston, 4th edição, 2001. ISBN: 978-0792372041. 63

[116] Mamdani, E.H. e S. Assilian: *An experiment in linguistic synthesis with a fuzzy logic controller*. International Journal of Man-Machine Studies, 7(1):1–13, 1975, ISSN 0020-7373. `https://www.sciencedirect.com/science/article/pii/S0020737375800022`. 64

[117] DeLany, Ryan, Andrew Smith, Yan Li e Liang Du: *Sdn dynamic controller configuration to mitigate compromised controllers*. Em *2023 IEEE Transportation Electrification Conference e Expo (ITEC)*, páginas 1–5, 2023. 65

[118] Sharafaldin, Iman, Arash Habibi Lashkari, Saqib Hakak e Ali A. Ghorbani: *Developing realistic distributed denial of service (ddos) attack dataset and taxonomy*. Em *International Carnahan Conference on Security Technology (ICCST)*, páginas 1–8, 2019. 70

[119] Alashhab, Abdussalam Ahmed, Aisha Edrah, Mohd Soperi Mohd Zahid e Md. Siddikur Rahman: *Ensemble based detection model for ddos attacks in sdns using advanced feature selection*. Em *2024 17th International Conference on Signal Processing and Communication System (ICSPCS)*, páginas 1–5, 2024. 70

[120] Saranya, N, T Abbinavu, P Arun Kumar, R Gnanasekar, P Guru Prasanth e R Subha: *Rtids: A robust transformer-based approach for intrusion detection system*. Em *2024 10th International Conference on Advanced Computing and Communication Systems (ICACCS)*, volume 1, páginas 1461–1464, 2024. 70

[121] Dhirta, Tarun e Akashdeep Sharma: *Unveiling the effectiveness of cnn-based models for multiclass ddos attack detection and classification: A comparative analysis*. Em *2023 International Conference on Data Science, Agents Artificial Intelligence (ICDSAAI)*, páginas 1–8, 2023. 70

[122] Yoachimik, Omer e Jorge Pacheco: *Ddos threat report for 2024 q2*, 2024. `https://blog.cloudflare.com/ddos-threat-report-for-2024-q2`. 70

[123] Johnson, Richard A. e Dean W. Wichern: *Applied Multivariate Statistical Analysis*. Pearson, 6ª edição, 2007. 70

[124] Cohen, Jacob, Patricia Cohen, Stephen G. West e Leona S. Aiken: *Applied Multiple Regression/Correlation Analysis for the Behavioral Sciences*. Routledge, 3ª edição, 2002. 71

[125] Turner, Aaron: *Tcpreplay - pcap editing and replaying utilities*, 2013. `https://tcpreplay.appneta.com/`. 75

[126] Harris, Guy: *Pcap capture file format*. `https://www.ietf.org/archive/id/draft-gharris-opsawg-pcap-01.html`, October 2022. Internet-Draft, IETF. 76

[127] Sharma, Sachin, Dimitri Staessens, Didier Colle, Mario Pickavet e Piet Demeester: *In-band control, queuing, and failure recovery functionalities for openflow*. IEEE Network, 30(1):106–112, 2016. 86

[128] Cybersecurity, Canadian Institute for: *Cicddos2019: A dataset for ddos attack detection*. `https://www.unb.ca/cic/datasets/ddos-2019.html`, 2019. 92