mathematics

MDPI

Article

P4-HLDMC: A Novel Framework for DDoS and ARP Attack Detection and Mitigation in SD-IoT Networks Using Machine Learning, Stateful P4, and Distributed Multi-Controller Architecture

Walid I. Khedr, Ameer E. Gouda * and Ehab R. Mohamed

Department of Information Technology, Zagazig University, Zagazig 44519, Egypt; wkhedr@fci.zu.edu.eg (W.I.K.); ehab.rushdy@zu.edu.eg (E.R.M.)
* Correspondence: amir.gouda@zu.edu.eg

Abstract: Distributed Denial of Service (DDoS) and Address Resolution Protocol (ARP) attacks pose significant threats to the security of Software-Defined Internet of Things (SD-IoT) networks. The standard Software-Defined Networking (SDN) architecture faces challenges in effectively detecting, preventing, and mitigating these attacks due to its centralized control and limited intelligence. In this paper, we present P4-HLDMC, a novel collaborative secure framework that combines machine learning (ML), stateful P4, and a hierarchical logically distributed multi-controller architecture. P4-HLDMC overcomes the limitations of the standard SDN architecture, ensuring scalability, performance, and an efficient response to attacks. It comprises four modules: the multi-controller dedicated interface (MCDI) for real-time attack detection through a distributed alert channel (DAC), the MSMPF, a P4-enabled stateful multi-state matching pipeline function for analyzing IoT network traffic using nine state tables, the modified ensemble voting (MEV) algorithm with six classifiers for enhanced detection of anomalies in P4-extracted traffic patterns, and an attack mitigation process distributed among multiple controllers to effectively handle larger-scale attacks. We validate our framework using diverse test cases and real-world IoT network traffic datasets, demonstrating high detection rates, low false-alarm rates, low latency, and short detection times compared to existing methods. Our work introduces the first integrated framework combining ML, stateful P4, and SDN-based multi-controller architecture for DDoS and ARP detection in IoT networks.

Keywords: SD-IoT; DDoS detection; ARP detection; machine learning; stateful P4; multi-controller; traffic monitoring

MSC: 68T01

Citation: Khedr, W.I.; Gouda, A.E.; Mohamed, E.R. P4-HLDMC: A Novel Framework for DDoS and ARP Attack Detection and Mitigation in SD-IoT Networks Using Machine Learning, Stateful P4, and Distributed Multi-Controller Architecture. *Mathematics* **2023**, *11*, 3552. https://doi.org/10.3390/math11163552

Academic Editor: Antanas Cenys

Received: 21 July 2023
Revised: 9 August 2023
Accepted: 14 August 2023
Published: 17 August 2023

1. Introduction

The Internet of Things (IoT) has become ubiquitous and is increasingly being deployed in various applications, including healthcare, transportation, and smart homes [1]. However, IoT networks are highly vulnerable to security threats, including Distributed Denial of Service (DDoS) attacks and Address Resolution Protocol (ARP) attacks [2,3]. The increasing scale and complexity of IoT networks make traditional security mechanisms ineffective, highlighting the need for new approaches to secure IoT networks [4,5]. In addition, it is essential to ensure the security of every layer of the IoT ecosystem. The IoT can be vulnerable to attacks at three distinct levels: the node layer, where data are collected; the network layer, where data are transmitted for processing; and the cloud layer, where data are stored [6]. In this study, the proposed framework primarily emphasizes securing the first two layers.

DDoS and ARP attacks pose significant threats to IoT networks. These attacks exploit vulnerabilities in network infrastructure and can have a severe impact on the overall

functioning and security of IoT systems. Understanding DDoS and ARP attacks and their relationship is crucial for mitigating their effects and protecting IoT networks from potential damage [7]. DDoS attacks are malicious attempts to disrupt the normal operation of a network or a specific service by overwhelming it with a massive volume of traffic. These attacks can be devastating for IoT networks as they target the limited resources of IoT devices and the underlying infrastructure [8]. By flooding the network with an enormous amount of data, DDoS attacks consume bandwidth, processing power, and memory, rendering IoT devices and services inaccessible to legitimate users [9].

On the other hand, ARP attacks exploit vulnerabilities in the ARP protocol, which is responsible for mapping IP addresses to MAC addresses in a network. In an ARP attack, an attacker sends fake ARP messages, known as ARP spoofing, to associate their MAC address with the IP address of a legitimate device on the network. In this manner, the attacker gains the ability to capture network data, divert it towards their own device, and engage in a range of nefarious actions, including eavesdropping, unauthorized access, and manipulation of data.

The relationship between DDoS and ARP attacks becomes evident when considering the potential collaboration between attackers. In some scenarios, DDoS attacks can be used as a smokescreen to divert attention and resources while ARP attacks are carried out to exploit vulnerabilities in the network. For example, during a DDoS attack, the network and security infrastructure might become overwhelmed, leading to decreased monitoring capabilities and increased susceptibility to ARP attacks. This collaboration between different attack vectors can magnify the impact on IoT networks, causing severe disruptions and compromising the security and privacy of connected devices and their users.

Additionally, the compromised security of IoT networks due to these attacks opens the door to further exploits and unauthorized access. Once an attacker gains control over IoT devices through ARP attacks, they can leverage these compromised devices to launch more sophisticated attacks, such as botnet-based DDoS attacks or data breaches. This not only poses a threat to the IoT network itself but also to other connected systems and networks that may interact with the compromised devices.

Software-Defined Networking (SDN) has emerged as a promising paradigm for managing and securing large-scale IoT networks [10]. SDN's centralized control enables network administrators to dynamically manage network resources and deploy security mechanisms to detect and mitigate attacks. Despite the various advantages of SDN architecture, it has two major weaknesses. Firstly, the standard SDN architecture centralizes all intelligence in a single controller, resulting in a Single Point of Failure (SPOF) and limiting scalability and performance. To overcome this, the integration of multiple controllers becomes crucial to address the SPOF problem and enhance system performance. Secondly, scalability and performance issues persist due to the limited intelligence of OpenFlow switches, which adopt a stateless approach for packet processing [11]. These switches heavily rely on the controller for network traffic forwarding and monitoring, leading to communication overhead between the data and control planes [12]. Furthermore, OpenFlow switches exhibit fixed behavior determined by the OpenFlow version, processing packets with a predefined set of actions [13,14]. This lack of flexibility makes it challenging for network administrators to customize header fields and actions according to diverse application requirements.

To overcome these limitations, researchers have proposed P4 (Programming Protocol-independent Packet Processors), a domain-specific language for programming the data plane [15]. P4 allows for the programming of additional functionalities and packet processing details, which enables the definition of different header structures and corresponding functions for matching and defining actions that the switch can take on each packet. The programmable pipeline provided by P4 offers flexibility, which is a significant advantage over OpenFlow switches [16].

Inspired by all of the abovementioned factors, we propose a new secure framework, P4-HLDMC, for detecting and mitigating DDoS and ARP attacks in SD-IoT using machine learning and stateful P4 with a hierarchical logically distributed multi-controller architec-

ture. The proposed framework employs a distributed architecture with multiple controllers, each responsible for a specific domain of the network. The stateful P4 algorithm is used to extract, monitor, and analyze network traffic features, while machine learning is used to identify anomalies in the traffic patterns.

The proposed detection framework utilizes a range of ML models, including Support Vector Machine (SVM), k-Nearest Neighbor (kNN), Gaussian Naive Bayes (GNB), Binomial Logistic Regression (BLR), Decision Tree (DT), Random Forest (RF), and Modified Ensemble Voting (MEV). The selection of these models for our DDoS attack detection in SD-IoT study was based on careful consideration of several key criteria. Firstly, we focused on the ability of these models to handle complex and high-dimensional data, which is often encountered in SD-IoT networks due to the diverse and dynamic nature of IoT devices and traffic patterns. SVM, kNN, GNB, BLR, DT, and RF have demonstrated effectiveness in dealing with such data through various mechanisms such as non-linear decision boundaries, feature similarity measures, and probabilistic reasoning. Secondly, the models were chosen for their interpretability and the ability to provide insights into the decision-making process. This is crucial in the context of SD-IoT security, as understanding the reasons behind an attack detection can help in devising appropriate countermeasures. Decision Tree and Logistic Regression, for example, offer clear decision paths, while kNN provides intuitive distance-based classification. Furthermore, the selected models are well-known in the field of machine learning and have been widely used in various domains, including intrusion detection. This familiarity ensures that the models have established performance benchmarks and comparison points. It also facilitates easier integration and interpretation of results within the existing body of research. Regarding the exclusion of other models, such as Neural Networks, the decision was made based on the complexity of implementation and the potential need for extensive tuning. While these models can offer high accuracy, they often require a larger amount of data and computational resources, which may be a challenge in resource-constrained SD-IoT environments. In summary, our choice of SVM, kNN, GNB, BLR, DT, RF, and Ensemble voting was driven by their capabilities to handle complex data, provide interpretability, and align with existing research benchmarks. The criteria considered were a combination of performance, interpretability, and practical feasibility within the context of SD-IoT DDoS attack detection. The effectiveness and efficiency of the proposed framework in detecting and mitigating DDoS and ARP attacks in SD-IoT networks are evaluated using a set of current and relevant datasets of real-world IoT network traffic, including the Edge-IIoTset [17], TON_IoT [18], and X-IIoTID dataset [19]. The primary contributions of the proposed framework are as follows:

1. The proposed framework introduces 17 new features, including 12 computed features, and 5 P4-extracted features, to effectively identify DDoS and ARP attacks. By using these features, the model can overcome the problem of over-fitting and provide a good fit.
2. This study presents the first integrated framework for DDoS and ARP detection in SD-IoT, which combines ML, stateful P4, and SDN-based hierarchical logically distributed multi-controller architecture. This framework is unique and innovative, and it provides a new approach for attack detection in IoT.
3. The study presents the Improved OpenDayLight (IODL) controller as the optimal choice for implementing the proposed framework. This selection is attributed to its superior network resource allocation, scalability, and stability, which were assessed through a comprehensive evaluation alongside other controllers using six criteria.
4. The first framework's module introduces the Multi-Controller Dedicated Interface (MCDI), which is a new proposed interface for Controller-to-Controller (C2C) communication. MCDI ensures a consistent and real-time attack detection process through a distributed alert channel (DAC). DAC reduces both the data-control overhead and communication overhead between the controllers by sharing only necessary information.
5. The second framework's module uses a novel proposed Multi-State Matching Pipeline Function (MSMPF), which employs nine state tables to monitor, analyze, extract, and

detect IoT network traffic features to be used as input to the third module. As a result, the controller has an extra layer of protection against both DDoS and ARP attacks.

6. The final module of the proposed framework presents a unique attack mitigation strategy consisting of 10 steps, which utilizes a distributed approach to mitigate attacks. The distributed approach spreads the attack mitigation process among multiple controllers, allowing the system to handle larger-scale attacks effectively and respond rapidly. This approach enhances the scalability and reliability of the system, enabling it to protect against sophisticated and coordinated attacks that conventional defenses may not be able to handle.

The rest of this paper is organized as follows. Section 2 provides a literature review of existing approaches to securing SD-IoT networks. Section 3 describes the proposed framework in detail. Section 4 discusses the experimental results and the implications of the proposed framework. Finally, Section 5 concludes the paper and provides directions for future research.

2. Related Works

The landscape of attack detection and mitigation in SD-IoT networks, encompassing DDoS and ARP attacks, has experienced notable growth in recent years. To enhance clarity and organization, we categorize the related works based on their analytical criteria.

2.1. DDoS Detection-Related Works

2.1.1. Non-ML DDoS Detection Approaches

One method for detecting DDoS attacks relies on statistical techniques. In [20], researchers introduced a statistics-based approach to identify DDoS attacks by assessing the entropy of packet payloads. They harnessed machine learning to evaluate entropy values and categorize traffic as normal or malicious. However, this approach neglected the stateful nature of SD-IoT networks, posing challenges in detecting attacks spanning multiple packets. Zhang et al. [21] proposed a method to detect low-rate (LR) DoS attacks using Power Spectral Density (PSD). In this context, distinct PSD entropy limits were established for normal and attack groups. Their non-AI-based intrusion detection system (IDS) exhibited a trade-off between accuracy and detection rates. Another strategy involves flow-based mechanisms for DDoS detection. Xie et al. [22] leveraged traffic-flow patterns to discern DDoS attacks, demonstrating effective detection with relatively low overhead compared to other methods. However, this approach proves less effective in high network traffic scenarios, necessitating the adoption of more advanced security measures. In [23], authors introduced a flow-based technique to uncover DDoS attacks in SDN. By employing OpenFlow switches to gather flow statistics and detect abnormal traffic patterns, this method did not account for the dynamic nature of SD-IoT networks, where devices frequently join and depart. Approaching the issue uniquely, ref. [24] introduced an innovative approach to actively detect attacks in resource-constrained cyber-physical systems, focusing on thwarted actuation attacks. These attacks disrupt communication between controllers and actuators. The proposed system comprises two core elements: (1) detection and (2) control. The detection module employs parallel detectors crafted through a multiple-model adaptive estimation strategy to identify attack occurrences and targeted actuators. The control unit employs a constrained optimization technique to compute optimal control inputs that satisfy both control and detection goals. A probabilistic framework was adopted to formulate detection and control objectives, capitalizing on available a priori information. To underscore the approach's effectiveness, a simulation study was conducted on an irrigation channel, yielding demonstrative outcomes.

2.1.2. ML and DL DDoS Detection Approaches

To address the limitations inherent in statistical and flow-based methodologies, recent research endeavors have proposed an amalgamation of machine and deep learning techniques with SDN to enhance DDoS detection in IoT environments. For instance, a

DDoS detection solution tailored to SDN-based IoT networks, known as LEDEM, was introduced in [25]. However, LEDEM's effectiveness is hindered by its rigid reliance on a single classification method, rendering it inadequate for combating diverse DDoS attack types. In a similar vein, Yin et al. [26] outlined a comprehensive architecture for SD-IoT, intending to scrutinize IoT network traffic and detect DDoS attacks through network attribute analysis. Regrettably, this model's potential is curtailed due to limitations in its ML-based categorization algorithms. Taking an innovative approach, researchers in [27] put forth a novel framework comprised of two integral components: DoS/DDoS detection and DoS/DDoS mitigation. This novel approach facilitates precise identification of attack types and associated packet types, thereby enabling targeted application of mitigation strategies. Operating as a multi-class classifier based on the "Looking-Back" concept, the proposed DoS/DDoS detection component was evaluated using the Bot-IoT dataset, culminating in an impressive 99.81% accuracy rate with a Looking-Back-enabled Random Forest classifier. Ullah et al. [28] contributed an anomaly-based detection system for IoT networks, featuring a multiclass classification technique employing a convolutional neural network (CNN) algorithm. Despite its commendable performance, ML approaches are the preferred choice for intrusion detection systems (IDSs) necessitating robust security capabilities [29]. In [29], an exploration of diverse ML models revealed that the XGBoost technique consistently yielded superior performance outcomes compared to other classifiers. While the performance results were drawn from two test cases, it was acknowledged that the dataset's scope was inadequate for comprehensively analyzing IoT network traffic behavior. To bridge the gap, Yousuf et al. [30] introduced DALCNN, leveraging OpenDayLight (ODL) as a suitable SDN controller to identify DDoS attacks in IoT. A notable limitation was observed, wherein the RNN algorithm's training using the NSL-KDD dataset did not align with the intricacies of IoT network traffic characteristics. Another noteworthy approach involved a Deep Neural Network (DNN) method proposed in [31] to identify DDoS attacks in SDN scenarios. Test results demonstrated the efficacy of the Deep IDS system with minimal network load, and without impacting the functionality of the POX controller. Yet, refinement is warranted to enhance detection rates and minimize false alarms across multiple OpenFlow Controllers. Shifting the focus to [32], authors introduced a framework for DDoS detection in SDN-IoT incorporating machine learning and stateful packet processing. A novel Double-Check Mapping Function (DCMF) was proposed to process packets and extract features at the data plane level. While machine learning techniques were employed to analyze the extracted features and classify traffic, the approach neglected the potential of a hierarchical, logically distributed multi-controller architecture to enhance scalability and reliability. A distinctive deep reinforcement learning (DRL)-based approach for detecting low-rate DDoS attacks in SDN was introduced by [33]. This approach embraced features such as traffic monitoring, traffic flow sampling, and a lightweight intrusion prevention system (IPS) for swift mitigation. However, the approach fell short in addressing the scalability nuances of SD-IoT networks. In a different vein, [34] proposed a novel DL approach intertwining CNN with the SD-Reg method to classify flow traffic as normal or attack. While effective in enhancing NIDSs' ability to detect unseen intrusion events, this stateless approach necessitated testing across diverse datasets encompassing various attack scenarios. Similarly, a CNN-based system was proposed in [35] for detecting DDoS attacks in IoT networks, focusing on blocking attacks at the source. However, the evaluation on the CIC-DDoS2019 dataset highlighted its limitations in effectively analyzing IoT network traffic behavior. Lastly, a study conducted by the authors of [36] harnessed an AutoML intrusion detection framework to identify suitable supervised classifiers, subsequently crafting an optimal ensemble strategy via soft voting. The proposed framework exhibited high accuracy in detecting intrusions; however, it was recognized that the datasets employed were not the most up-to-date and did not comprehensively encompass all types of attack intrusions. Furthermore, network stateful cases were not considered.

2.1.3. P4-Based DDoS Detection Approaches

In [37], DDoS attack detection using P4 is addressed by implementing a hash value calculation of the source IP and MAC address in the data plane switch, which is then compared to the previously stored hash value. The detection of an attack occurs when there is no match and the time difference between the last attack and the current packet is less than 5 s. However, this method has some limitations such as increased CPU consumption at switches, inability to detect complex attack patterns, and failure to differentiate between flash and attack traffic. Febro et al. [38] proposed a source-based DDoS defense solution that detects attacks close to the source to save computational and bandwidth resources. In this approach, P4-enabled edge switches count the number of packets sent by the hosts connected to each port, and the controller compares these values with a static threshold. Once the threshold is exceeded, the controller sends a command to the P4 switch to drop all subsequent packets from the same ingress port. However, this method cannot differentiate between legitimate traffic and attack traffic, as it relies on a static threshold value. Lastly, a study conducted by the authors of [39] delved into the potential of AI and ML algorithms for automating the detection of Transmission Control Protocol (TCP) flood attacks. A comparison between Standalone and Correlated DDoS attack detection (DAD) architectures was conducted, whereby traffic feature collection and attack detection were performed locally at network switches or controllers. However, the approach failed to account for the nuanced features of IoT traffic and was confined to detecting a single type of DDoS attack. Furthermore, comprehensive dataset testing was lacking.

2.2. ARP Detection-Related Works

Existing solutions for detecting ARP attacks involve methods such as analyzing traffic patterns, utilizing cryptographic solutions, creating flow graphs, or applying statistical techniques. However, these approaches can be time-consuming, computationally intensive, or complex in terms of processing power. Additionally, some of these methods rely on threshold-based analysis of only one parameter. In contrast, our proposed approach overcomes these limitations by considering multiple significant parameters, resulting in promising results.

2.2.1. Non-ML ARP Detection Approaches

Hong et al. [40] proposed a detection mechanism that collects dynamic information about the network's topology, including the switches, flow paths, IP addresses, and MAC addresses. By analyzing these features, the attack can be detected. Sebbar et al. [41] focused on detecting Man-in-the-Middle (MITM) and traffic redirection attacks. They check the state of a new node connecting to the controller and drop the connection if it is not labeled as "New", indicating a potential malicious node. Additionally, suspicious delays in host responses are monitored, and responses exceeding a specific threshold are considered possible delay attacks. Zhang et al. [42] detect MITM attacks by calculating packet delays in TCP connections. They compare the mean delay of a session with predefined reference values. If the delay exceeds the threshold, it is flagged as a suspicious outlier and reported to the monitoring module. Deng et al. [43] tackle controller attacks by validating the legitimacy of Packet-In messages. When a new packet_in arrives at the controller, it is compared to the MAC addresses in the Mac-Port mapping table. If a match is found, the packet is processed; otherwise, it is dropped. Kaur [44] presented three distinct approaches for detecting ARP spoofing attacks, namely a signature-based method, a manual Wireshark packet analysis method, and a machine learning method. Among these, the Naive Bayes algorithm demonstrated the highest accuracy of 93% and the lowest false alarm rate (FAR).

2.2.2. ML-Based ARP Detection Approaches

Ma et al. [45] introduced a Bayesian method to calculate the probability of an attack and employed various ML algorithms for attack detection. Despite utilizing only four features and lacking experimental data verification, the author acknowledged detecting

the attack. In [46], the utilization of ML was evident in their endeavor to identify ARP attacks within SDN. They constructed a Python application within the SDN controller via Mininet, tasked with gathering and recording the requisite attack-detection features into a designated file referred to as the traffic dataset. This dataset was subsequently harnessed for both model training and attack detection purposes. The amalgamated Convolutional Neural Network-Long Short Term Memory (CNN-LSTM) model demonstrated superior performance compared to other ML models. Overall, while some works have proposed effective methods for DDoS and ARP detection in SDN or IoT networks, there is still a need for a comprehensive framework that takes into account the stateful nature of SD-IoT networks, the dynamic topology, and the need for consistency, scalability, and reliability.

3. The Proposed HLDMC Framework

The proposed P4-HLDMC framework comprises four modules that work together to detect and mitigate DDoS and ARP attacks on SD-IoT networks. The first module is a novel proposed dedicated interface called MCDI, which creates a new interface for communication between multiple controllers and ensures that attack detection is consistent and real-time through the use of a distributed alert channel called DAC. The second module, MSMPF, uses a P4-enabled stateful multi-state matching pipeline function to analyze and monitor IoT network traffic features through nine state tables. This information is then sent to the third module, which employs a modified ensemble voting algorithm with six classifiers to identify anomalies in the traffic patterns extracted by P4, achieving early and accurate attack detection at the data plane level. Finally, the fourth module involves an attack mitigation process that includes ten steps and is distributed among multiple controllers to handle larger-scale attacks effectively and respond more quickly. These four modules will be discussed in detail in the following sections. Figure 1 illustrates the integration scheme among the four modules of the proposed framework.

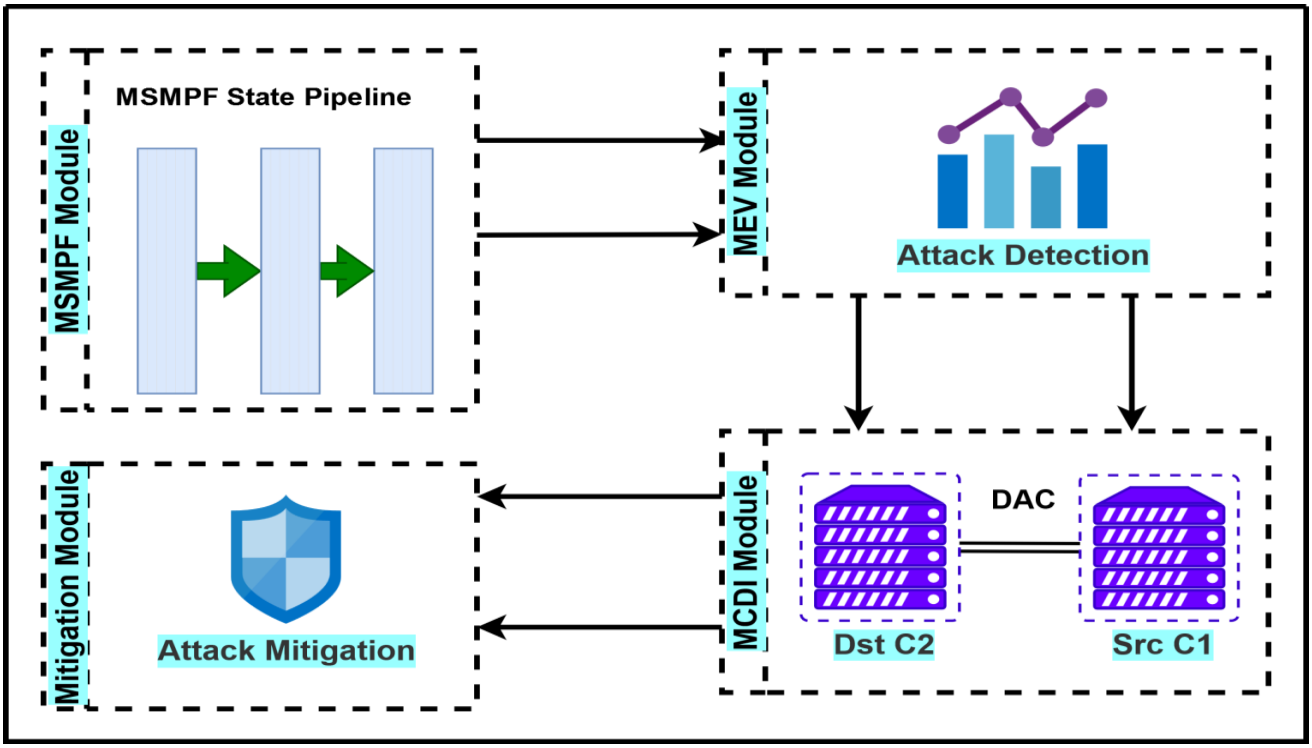

Figure 1. The integration scheme of the four P4-HLDMC modules.

The layered architecture depicted in Figure 2 illustrates the comprehensive structure of the proposed framework. Each layer fulfills a specific role in enhancing the security and resilience of the network. Through a combination of monitoring, detection, and mitigation

mechanisms, the proposed framework strengthens the defense against attacks, reducing their impact on the overall network infrastructure. By adopting a logically distributed architecture, the P4-HLDMC framework acknowledges the interconnected nature of the Internet and focuses on preventing the spread of attacks from local area networks (LANs) to the Internet service provider (ISP) level.

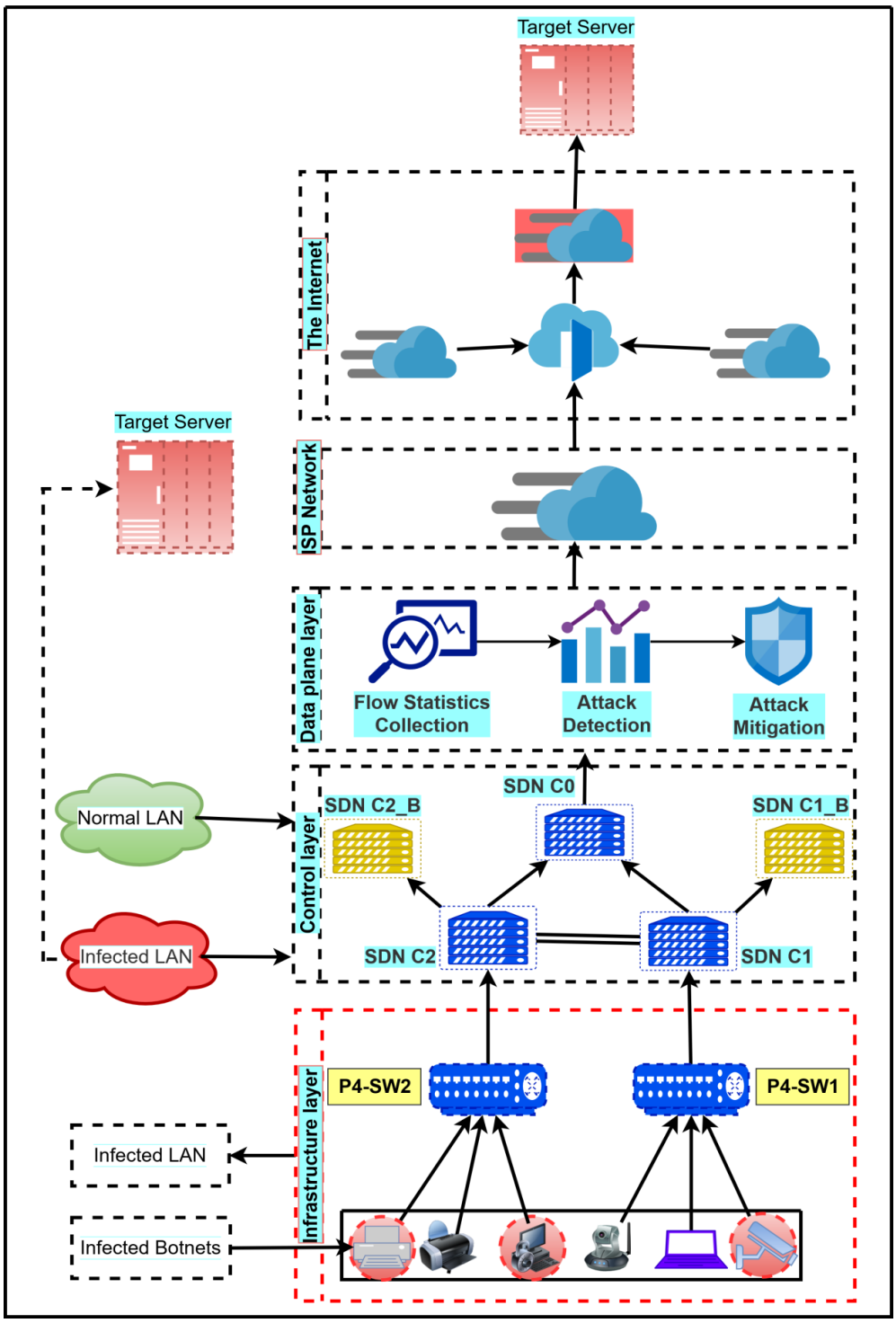

Figure 2. The logically distributed architecture of the proposed P4-HLDMC framework.

3.1. The MCDI Module

In this module, a hierarchical multi-controller SDN design is proposed for network management, which partitions the network into multiple domains with each controller

responsible for managing its own SD-IoT domain. The proposed design involves two layers of controllers: the domain controllers and the master controller. The domain controller manages the P4 switches in its local domain and executes local control applications, while the master controller manages the domain controllers and maintains a global view of the network. Each domain controller has a backup controller as an extra layer of security in case of controller loss, which has the same state as the main one. The backup controllers are labeled as "SDN C1_B" and "SDN C2_B" in the diagram depicted in Figure 3 below. To ensure correct packet transmission across the network, the controllers need to exchange domain information to maintain a consistent view. Thus, achieving controller consistency is vital for effective attack detection and mitigation in real-time. However, transmission delays, concurrent strategic conflicts among controllers, and issues with flow table order can lead to inconsistencies in the controller state, resulting in packet loss and service disruptions. To address this, the study proposes two control layers for the first module: (1) adding control functions to P4 switches to improve controller consistency, utilizing P4 programmability, and (2) proposing a novel multi-controller dedicated interface (MCDI) for consistent and real-time attack detection processes. This interface uses an asynchronous threading system with one thread and a voter to facilitate communication between multiple controllers, where the thread serves as the communication channel and the voter indicates the mode of communication. Figure 3 depicts the implementation of the proposed MCDI interface.

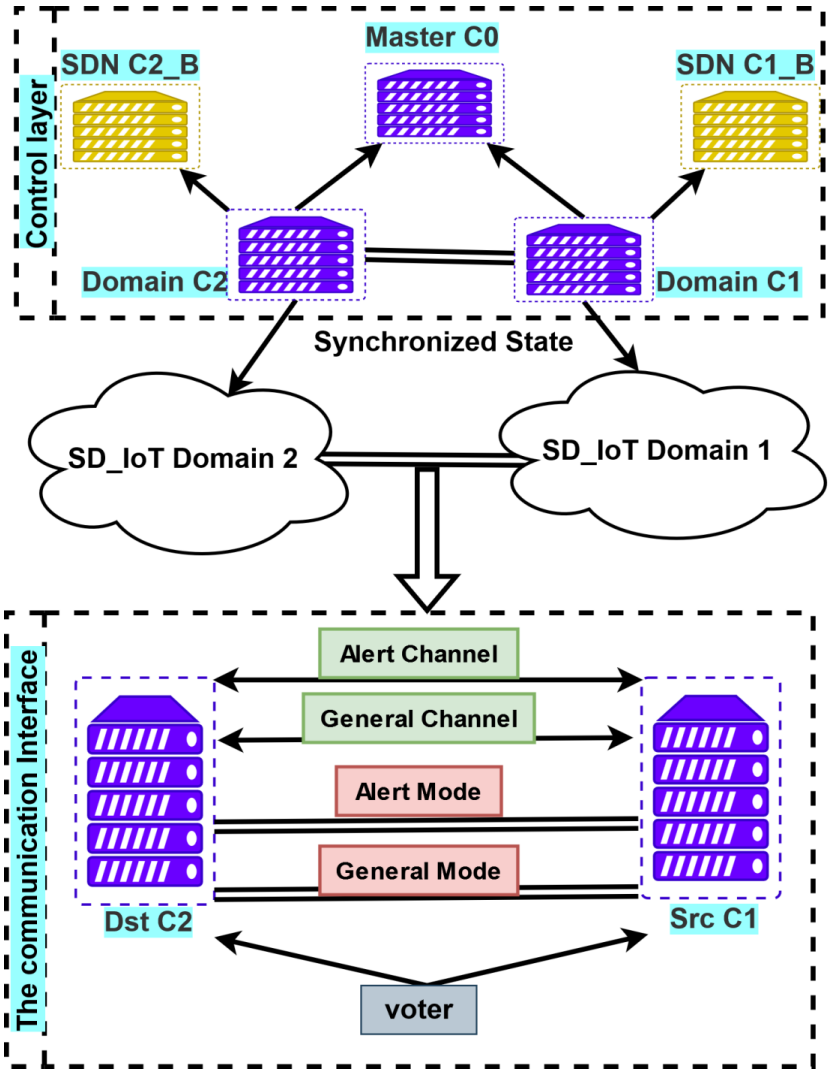

Figure 3. The implementation of the proposed MCDI interface.

The current model utilized in distributed SDN controllers synchronizes data between controllers without considering the specifics of the network or necessary services. This can result in an excessive amount of data being transmitted to other controllers that is not always needed, leading to decreased bandwidth and lower controller throughput. Therefore, a more tailored approach is necessary, based on the required service. To address this issue, two synchronization modes are proposed in the new MCDI that are appropriate for varying scenarios and infrastructures. The two synchronization modes are: (1) the alert mode and (2) the general mode. The general mode incorporates two channels, namely the general channel and the alert channel, which is leveraged to exchange regular network events between the controllers. In contrast, the alert mode comprises only the alert channel, which is built on the Advanced Message Queuing Protocol (AMQP). The primary objective of the alert mode is to detect the neighboring controllers and establish a distributed alert channel (DAC) between them. Other controllers utilize this channel to share network information with one another. When a domain controller undergoes a DDoS or ARP attack, a notification is transmitted to other controllers through the DAC channel of the alert mode. The event is then published, and other controllers update their network status to achieve synchronization. This proactive approach ensures the protection of critical network components, mitigates potential resource scarcity, and fortifies the overall security posture, enabling IoT networks to operate in a secure and reliable manner. Algorithm 1 shows the proposed MCDI algorithm.

Algorithm 1: The Proposed MCDI Algorithm.

Inputs: Service required by the IoT network, IoT Network infrastructure specifications.
Outputs: Synchronized data between controllers based on the required service.
Determine the service required by the network and the infrastructure specifications.
Choose the appropriate synchronization mode based on the requirements using the **Voter**.
    If (General mode is selected)
        Establish a general channel and an alert channel between the controllers.
        Exchange regular network events between the controllers through the general channel.
        Use the alert channel to exchange important network events between the controllers.
    End if
    If (Alert mode is selected)
        Establish an alert channel between the controllers.
        Use Advanced Message Queuing Protocol (AMQP) for the alert channel.
        Detect neighboring controllers.
        Establish a Distributed Alert Channel (DAC) between them.
        Share network information with other controllers through the DAC.
    End if
    If (DDoS/ARP attack is detected):
        Notify other controllers through the DAC channel of the alert mode.
        Publish the event to the IoT network.
        Update the network status of other controllers to achieve synchronization.
    End if

3.2. The MSMPF Module

The MSMPF, a novel multi-state matching pipeline function, forms the core of this module and comprises nine state tables. Each state table has a feature entry that, when combined with other state tables, accurately detects four types of DDoS attacks (TCP, UDP, ICMP, and HTTP) and two types of ARP attacks (ARP spoofing and ARP poisoning). This module utilizes State_Tables 1, 2, 8 and 9 to detect and mitigate ARP attacks. The first two state tables detect fake source IP/MAC addresses and prevent them from sending traffic in the future. State_Table 1 contains entries for source IP addresses, while State_Table 2 contains entries for source MAC addresses. The Source IP/MAC state tables pair (SIMP)

verifies that each source MAC address is associated with only one source IP address within a 2-s window size.

The MSMPF uses one-to-one mapping to validate the first two state tables' entries. If the state value of any state table entry exceeds one, it indicates that either the source IP or the source MAC is fraudulent. The output of the state matching process performed by SIMP is utilized as an input for State_Table 3. The third state table calculates Destination IP Entropy (DIPE) for a 2-s time window size. If any entry in the DIPE table is lower than a predefined threshold of 1.28, it is considered an attack. This threshold is based on the methodology and comprehensive experimental findings presented in our research paper [28].

The MSMPF function uses the output of the first three state tables as an input for two state table pairs, namely (SPair_4), and (SPair_5). SPair_4 is the state table pair of State_Tables 4 and 6, while SPair_5 is the state table pair of State_Tables 5 and 7. SPair_4 counts the number of source and destination TCP port numbers during a 2-s window, while SPair_5 counts the number of source and destination UDP port numbers. An attack may generate a large number of random port numbers, leading to a high frequency of entries for source and destination ports. By measuring the speed of these port entries to the switch, it becomes possible to compare it with the threshold to detect the attack.

Finally, the last seven State_Tables are used to feed State_Table 8. The last two State_Tables 8 and 9 detect fake destination IP/MAC addresses. State_Table 9 contains entries for destination IP addresses, while State_Table 8 contains entries for destination MAC addresses. The Destination IP/MAC state tables pair (DIMP) stores the original destination IP/MAC addresses. The DIMP checks every 2 s whether the original destination IP/MAC address exists in the state tables. The DIMP pair is continuously updated by sending ARP requests and removing any pair that does not receive a reply within a threshold time. The last two state tables are responsible for verifying the legitimacy of each destination IP/MAC address, and any entry value greater than 1 indicates a forged IP/MAC address. The output of the MSMPF is then sent to the controller for appropriate action. Algorithm 2 outlines the steps for detecting ARP spoofing/poisoning attacks using the proposed MSMPF, while Algorithm 3 demonstrates the steps for detecting DDoS attacks.

By employing this methodology, DDoS and ARP attacks can be detected and blocked, providing an additional level of protection to the controller against such attacks. This module offers two significant features: (1) detection of the attack at the data plane level using a new P4 application installed on the P4-enabled switches, and (2) differentiation between attack traffic and flash crowds. In order to analyze network traffic and detect attacks, it is necessary to undergo an initial learning phase where the algorithm calculates the feature ranges and distribution limits. This learning phase is important to provide a baseline to the MSMPF for comparison with future traffic. We run both normal and attack traffic with different attack rates ranging from 20% to 80% at a time interval of 30 s to learn the normal and attack metrics. After this learning phase, the process is repeated at regular time periods to analyze the features, such as source and destination IP addresses and ports, and detect any anomalies or attacks that may have occurred. Figure 4 depicts the implementation of the proposed MSMPF.

Algorithm 2: Second Module MSMPF (ARP Attack Detection and Mitigation Algorithm).

Inputs: IoT-NT, ST1-EV, ST2-EV, SPair-4, SPair-5, T-WS
Output: NT-C → (Normal: 0, Attack: 1)

IoT-NT ← IoT Network Traffic
NT-C ← Network Traffic Class Label (Legitimate = 0, Attack = 1)
PKT-EC ← Packet Entry Count
D-IPC ← Destination IP Count
T-WS ← Time-based Traffic Window (2 s)
D-IP ST ← Destination IP Address State Table
DIPE-ETH ← DIPE State Table Entry Threshold

Algorithm 2: *Cont.*

```
PKT-IN ← IoT Packet_IN Event
ST ← State Table
ST-3 ← Third State Table
ST1-EV ← First State Table Entry Value
S-IP ← Source IP Address
D-IP ← Destination IP Address
S-MAC ← Source MAC Address
D-MAC ← Destination MAC Address
MSMPF-V ← Multi-State Matching Pipeline Function Value
SPair-4 ← State Table Pair of Tables 4 and 6
SPair-5 ← State Table Pair of Tables 5 and 7
SPair-8 ← State Table Pair of Tables 8 and 9
PKT-EC = 0, T-WS = 2 s, D-IPC = 0, and DIPE-ETH = 1.28
For each (PKT-IN):
    If (ST1-EV > 1):
        S-IP → spoofed
        MSMPF-V = 1
        Block Source port
    Else if (ST2-EV > 1):
        S-MAC → spoofed
        MSMPF-V = 1
        Block Source port
    Else
        Legitimate S-IP and S-MAC
        MSMPF-V = 0
        Send MSMPF-V→ ST-3
    End If
End for
For each (PKT-IN):
    If (D-MAC exists in SPair-8):
        D-MAC is legitimate
    Else
        D-MAC → forged
        MSMPF-V = 1
        ARP Poisoning attack detected
    End If
End for
For each (PKT-IN):
    If (D-IP exists in SPair-8):
        D-IP is legitimate
    Else
        D-IP → spoofed
        MSMPF-V = 1
        ARP Spoofing attack detected
    End If
End for
```

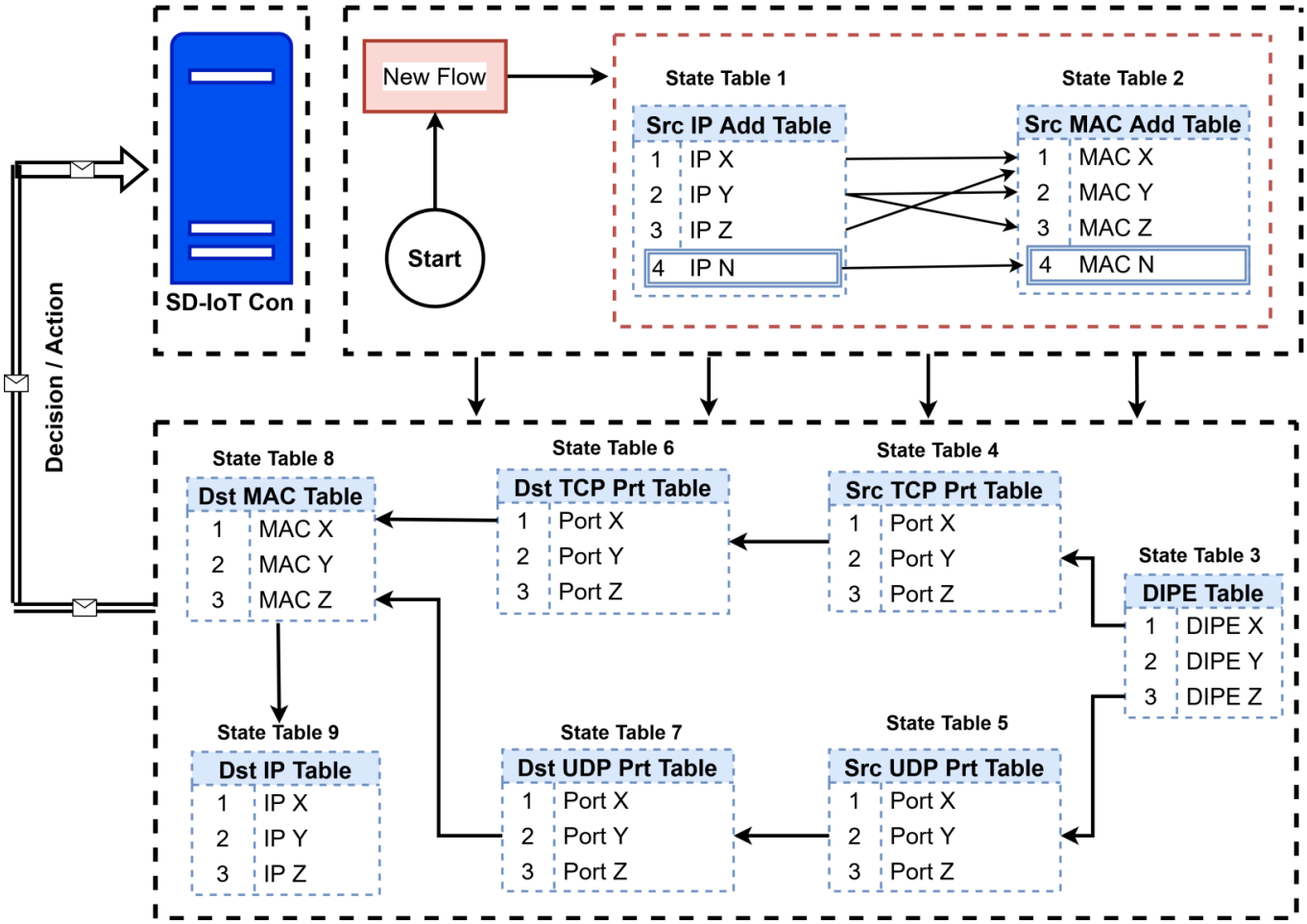

Figure 4. The proposed MSMPF state table pipeline implementation.

Algorithm 3: Second Module MSMPF (DDoS Attack Detection and Mitigation Algorithm).	
1	**Inputs:** IoT-NT, ST1-EV → ST7-EV, SPair-4, SPair-5, T-WS
2	**Output:** NT-C → (Normal: 0, Attack: 1)
3	IoT-NT ← IoT Network Traffic
4	NT-C ← Network Traffic Class (Legitimate = 0, Attack = 1)
5	T-WS ← Time-based Traffic Window (2 s)
6	WSPS-ETH ← The Source Port Speed Threshold
7	WSTPE-ETH ← The Source TCP Port Entropy Threshold
8	WSUPE-ETH ← The Source UDP Port Entropy Threshold
9	STCP-Prt ← The Source TCP Port;
10	SUDP-Prt ← The Source UDP Port
11	PKT-EC ← Packet Entry Count
12	D-IPC ← Destination IP Count
13	D-IP ST ← Destination IP Address State Table
14	DIPE-ETH ← DIPE State Table Entry Threshold
15	ST ← State Table; ST-EV ← State Table Entry Value
16	D-IP ← Destination IP Address
17	SPair-4 ← State Table Pair of Tables 4 and 6
18	SPair-5 ← State Table Pair of Tables 5 and 7
19	PKT-EC = 0, T-WS = 2 s, D-IPC = 0, and DIPE-ETH = 1.28

Algorithm 3: *Cont.*

```
For each (PKT-IN):
    If (D-IP exists in D-IP ST):
        Assign D-IP to D-IPC
    Else
        Assign D-IP to D-IP ST
    End If
    If (PKT-EC % T-WS == 0):
        Calculate DIPE → ST3
    Else
        PKT-EC = PKT-EC + 1
    End If
End for
For each (ST3-E):
    If (ST3-EV < DIPE-ETH)
        MSMPF-V = 1
    Else
        MSMPF-V = 0
        Send MSMPF-V→ SPair-4, SPair-5
    End If
End for
For each (ST3 PKT-IN):
    If ((ST4-EV > WSPS-ETH)||(ST4-EV < WSTPE)):
        STCP-Prt → spoofed
        Block STCP-Prt
        MSMPF-V = 1

    Else if (ST6-EV> WSPS-ETH) || (ST6-EV< WSTPE)):
        DTCP-Prt → spoofed
        Block STCP-Prt
        MSMPF-V = 1
    Else
        MSMPF-V = 0
        Send MSMPF-V→ ST8
    End If
End for
For each (ST3 PKT-IN):
    If ((ST5-EV > WSPS-ETH) || (ST5-EV < WSUPE)):
        SUDP-Prt → spoofed
        Block SUDP-Prt
        MSMPF-V = 1
    Else if ((ST7-EV> WSPS-ETH)||(ST7-EV< WSUPE)):
        DUDP-Prt → spoofed
        Block SUDP-Prt
        MSMPF-V = 1
    Else
        MSMPF-V = 0
        Send MSMPF-V→ ST8
        Send ST8-V→ P4 Compiler
    End If
End for
```

3.3. The MEV Detection Module

The third module employs a novel ML-based stateful P4 application to detect DDoS and ARP attack using only seventeen P4 extracted and computed features. The module uses an ML-enabled modified ensemble voting (MEV) algorithm with six classifiers to identify anomalies in the extracted traffic patterns. The use of stateful P4 allows the application to

maintain a detailed view of the network state, enabling accurate and early attack detection at the data plane level.

3.3.1. The MEV Classification Phase

In this paper, a modified ensemble voting (MEV) algorithm is proposed for DDoS and ARP detection in SD-IoT networks. The algorithm combines the predictions of multiple ML classifiers, including SVM, kNN, GNB, BLR, DT, and RF, to improve the detection accuracy. The proposed algorithm is modified by incorporating a new weighted voting scheme (NWVS), where the weight of each classifier is dynamically adjusted based on its performance on the training dataset. The first step of the MEV algorithm is the feature selection, which can improve the detection performance and reduce the computational complexity. This step involves removing irrelevant and redundant features from the extracted IoT network traffic features and selecting the most relevant features using a mutual information-based feature selection method. Next, in the training phase, the labeled IoT network traffic features are split into training and validation sets. Each ML classifier, including SVM, kNN, GNB, BLR, DT, and RF, is trained on the training dataset with the selected features, and its performance is evaluated using the validation set. The weight of each classifier is calculated based on its performance, and the weights are normalized. In the detection phase, new IoT network traffic features are received, and the selected features are extracted from the received traffic features using P4. Each ML classifier is then used to predict the class label for the received traffic features, and the predicted class labels are multiplied by their corresponding weights. The weighted class labels are summed to obtain the final predicted class label, which is compared to the threshold. If the final predicted class label is above the threshold, the traffic is classified as an attack; otherwise, it is classified as normal traffic. Finally, the performance of the proposed algorithm is evaluated on the test dataset, including various evaluation metrics that will be discussed in Section 4. The experimental results demonstrate that the proposed algorithm achieves higher detection accuracy and outperforms the traditional ensemble voting algorithm in DDoS detection in SD-IoT networks. Algorithm 4 shows the MEV algorithm steps.

Algorithm 4: The Third Module (MEV Detection Algorithm).

Inputs: P4-IoT-NT, MSMPF, IoT-Train, IoT-L, T-WS
Output: NT-C → (Normal: 0, Attack: 1), NT-MC

P4-IoT-NT ← IoT Network Traffic features extracted using P4
NIoT-NT ← New IoT Network Traffic
N-IoT-T ← Normal IoT Traffic; A-IoT-T ← Attack IoT Traffic
NT-BC ← Network Traffic Binary Class Label (Legitimate = 0, Attack = 1)
NT-MC ← Network Traffic Multi-Class Label (four DDoS and three ARP) classes.
W-CL ← Weighted class labels; F-PCL ← Final predicted class labels
MSMPF ← P4-enabled stateful multi-state matching pipeline function
T-WS ← Time-based Traffic Window (2 s)
IoT-Train ← IoT Training dataset; IoT-Test ← IoT Test set
IoT-Train-SF ← IoT Training dataset using selected features
IoT-Val ← IoT validation set
IoT-L ← Labeled IoT network traffic features
SDP4-ML ← Set of ML classifiers including SVM, kNN, GNB, BLR, DT, and RF
SDP4-ML-P ← The performance of each ML classifier on the IoT-Val
SDP4-ML-W ← The weight of each ML classifier on the IoT-Val
IoT-TH ← Threshold for binary classification
MSMPF-V ← Multi-State Matching Pipeline Function Value
MIB-FS ← Mutual information-based feature selection method
P4-IoT-SF ← Selected P4 IoT features
P4-SF Table ← Selected features table

Algorithm 4: *Cont.*

```
EV-CR ← Evaluation criteria (ACC, F1-score, TPR, TNR, PPV, NPV, MCC, AUC, FPR, FNR, FDR, AVG latency, and ADT)
EC-R Table ← Evaluation criteria results table
Feature Selection Phase
For each (P4-IoT-NT):
    Apply MIB-FS
    Select (P4-IoT-SF) from (P4-IoT-NT)
    Store P4-IoT-SF → P4-SF Table
End for
Training Phase
For each (IoT-L):
    Split IoT-L → (IoT-Train) and (IoT-Test)
    Train SDP4-ML → IoT-Train-SF
    Evaluate SDP4-ML-P → IoT-Val
    Calculate SDP4-ML-W → SDP4-ML-P
    Normalize SDP4-ML-W
End for
Attack Detection Phase
For each (NIoT-NT):
    Extract P4-IoT-SF → from → NIoT-NT
    Use each SDP4-ML → to predict: NT-BC and NT-MC
    Multiply NT-BC or NT-BC → by their corresponding weight
    Sum W-CL → to get final predicted NT-BC or NT-MC
                If (F-PCL > IoT-TH)
                    Attack traffic is detected
                Else
                    The traffic is normal
                End if
End for
MEV Evaluation Phase
For (IoT-Test):
    Calculate EV-CR → IoT-Test
    Store EV-CR → EC-R Table
    Improve MEV → by updating → SDP4-ML-W
End for
```

3.3.2. P4-Enabled Feature Extraction Phase

The feature extraction phase in machine learning is crucial as it helps to select relevant information from the raw data to train the model effectively. In our proposed framework, we used various P4-enabled feature extraction techniques to extract important features from network traffic. Specifically, we introduced 17 new features, 12 computed features, and 5 extracted features. The P4-extracted features include the TCP packets rate (*TCPR*), the UDP packets rate (*UDPR*), the ICMP packets rate (*ICMPR*), the HTTP packets rate (*HTTPR*), and Rx bytes rate (*RxBR*). The computed features include the average window flow packets (*AWFP*), average window flow bytes (AWFB), The Packet_In rate per window (*WPIR*), flow entry rate per window (*WFER*), Destination IP entropy per window (*WDIPE*), the source port entry speed from the P4 ingress switch port per window (*WSPS*), the source TCP port entropy per window (*WSTPE*), the TCP packets percentage to all packets per window (*TCPP*), and others. All these features were selected based on their significance in identifying DDoS and ARP attacks in SD-IoT. When new IoT network traffic features are received, they are analyzed to determine whether they indicate normal traffic or an attack. During a DDoS attack, certain features such as *AWFP*, *AWFB*, *WPIR*, *RxBR*, *WSPS*, *WFER*, *TCPR*, *UDPR*, and others may sharply increase over time, while the values of other features

such as *WDIPE*, *WSTPE*, and *WSUPE* decrease. Overall, the feature extraction phase is critical to the success of our proposed framework in effectively identifying and mitigating the attacks. Table 1 lists the details of P4-extracted and computed features, including their description and the corresponding methods of computation or extraction. The notation "TW" in the table stands for a time window size of 2 s.

Figure 5 illustrates the P4-enabled feature extractor module. When IoT nodes transmit traffic to the network, P4 switches receive and relay the traffic information to the feature extractor module. The module extracts the features and forwards them to the MEV classifier module for traffic classification. The decision regarding the traffic's normal or attack status is then relayed back to the P4 switch for appropriate action. The workflow of the P4 code is as follows: First, the received packets are parsed to extract the relevant protocol headers. The code then enters the ingress pipeline, consisting of a sequence of nine state tables defined in the second module of the proposed framework. Each table updates the feature occurrences in the registers and packet metadata. After updating the statistics, the ingress pipeline undergoes a conditional check. If the window period concludes, the code executes actions to generate a report on packet analysis and classification, which is sent to the P4 switch. Otherwise, it follows a standard procedure for packet forwarding or blocking based on flow entries from the MEV classifier module. This enables the P4 switch to function as a firewall by allowing or blocking suspected flows based on the outcomes of the MEV classifier module.

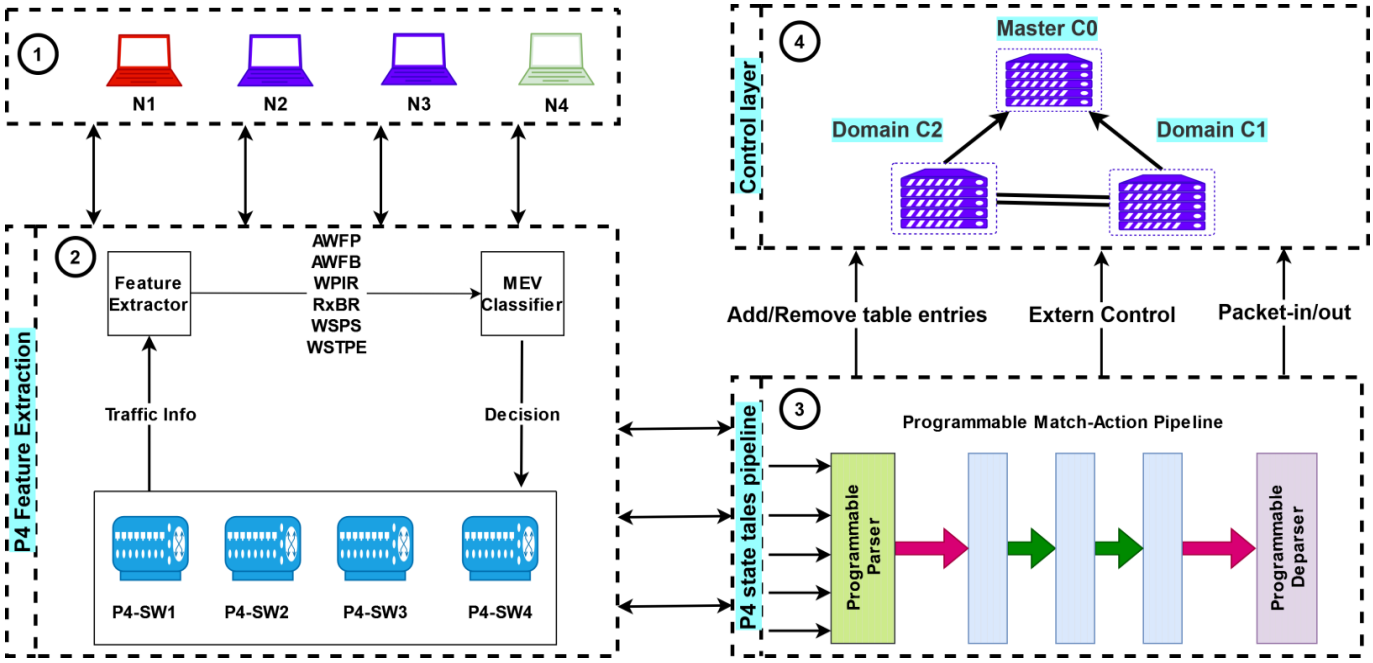

Figure 5. The proposed P4-enabled feature extraction phase.

In the proposed framework, stateful features are generated, processed, and stored within the P4 switch, ensuring overall system scalability. P4 switches have demonstrated scalability with the number of flow entries, allowing for the analysis of numerous flows using the same pipeline control section.

Table 1. The P4-extracted and computed features.

F. No	Name	Description	Type	Extraction/Computation Method	
1	AWFP	Average flow packets/window	Computed	$AWFP = (no.\ of\ packets\ in\ all\ flows\ per\ window/total\ no.\ of\ flows)$	(1)
2	AWFB	Average flow bytes/window	Computed	$AWFB = (no.\ of\ bytes\ in\ all\ flows\ per\ window/total\ no.\ of\ flows)$	(2)
3	WPIR	The Packet_In rate/window	Computed	$WPIR = (total\ no.\ of\ Packet\_In\ entries/TW)$	(3)
4	RxBR	The Rx bytes rate	Extracted	$RxBR = (total\ no.\ of\ Rx\ bytes/s)$	(4)
5	WSPS	The source port entry speed from P4 ingress switch port/window	Computed	$WSPS = (total\ no.\ of\ src\ ports/TW)$	(5)
6	WFER	Flow entry rate/window	Computed	$WFER = (total\ no.\ of\ flow\ entry\ count/TW)$	(6)
7	WDIPE	Destination IP entropy/window	Computed	Calculated using WDIPE function added to the P4 application	
8	WSTPE	Source TCP port entropy/window	Computed	Calculated using WSTPE function added to the P4 application	
9	WSUPE	Source UDP port entropy/window	Computed	Calculated using WSUPE function added to the P4 application	
10	TCPR	The TCP packets rate	Extracted	$TCPR = (total\ no.\ of\ TCP\ packets/s)$	(7)
11	UDPR	The UDP packets rate	Extracted	$UDPR = (total\ no.\ of\ UDP\ packets/s)$	(8)
12	ICMPR	The ICMP packets rate	Extracted	$ICMPR = (total\ no.\ of\ ICMP\ packets/s)$	(9)
13	HTTPR	The HTTP packets rate	Extracted	$HTTPR = (total\ no.\ of\ HTTP\ packets/s)$	(10)
14	TCPP	The TCP packets percentage: the percentage of TCP packets to all packets/window	Computed	$TCPP = (total\ no.\ of\ TCP\ packets/no.\ of\ packets\ per\ window)$	(11)
15	UDPP	The UDP packets percentage: the percentage of UDP packets to all packets/window	Computed	$UDPP = (total\ no.\ of\ UDP\ packets/no.\ of\ packets\ per\ window)$	(12)
16	ICMPP	The ICMP packets percentage: the percentage of ICMP packets to all packets/window	Computed	$ICMPP = (total\ no.\ of\ ICMP\ packets/no.\ of\ packets\ per\ window)$	(13)
17	HTTPP	The HTTP packets percentage: the percentage of HTTP packets to all packets/window	Computed	$HTTPP = (total\ no.\ of\ HTTP\ packets/no.\ of\ packets\ per\ window)$	(14)

3.4. The Distributed Attack Mitigation Module

The mitigation module takes action upon receiving instructions from the first three modules, upon identification of a malicious traffic flow. It protects IoT nodes by adding extra high-priority flow rules to P4-switch flow tables, which match the attack packet rules. Upon receiving a *FlowMod* message from the controller, the switch adds a new flow table entry and drops all packets originating from the attack source, thereby reducing and mitigating the attack's impact. Moreover, this module involves isolating the attacked device and rerouting network traffic to prevent further damage. The proposed MCDI interface's DAC is utilized by this module to publish alert notifications to other controllers. By employing the P4-HLDMC architecture, the module distributes the attack mitigation process across multiple controllers, enabling the system to effectively handle larger-scale attacks and respond more quickly. Figure 6 illustrates the proposed framework's general flowchart, which depicts the mitigation process. Algorithm 5 outlines the steps of the attack mitigation process. It begins by categorizing the traffic into different types, such as new IoT traffic, normal IoT traffic, and attack IoT traffic as informed by the P4 switches and the controllers. For each new IoT traffic entry, the algorithm checks if it matches the attack traffic. If a match is found, the algorithm proceeds with isolating the infected IoT nodes, rerouting traffic, adding new flow rules, dropping packets from the attack source, blocking attacking MAC, IP, and port, and publishing alert notifications. Overall, the mitigation module and the accompanying algorithm provide a systematic approach to detect and respond to malicious traffic flows in an IoT network. The use of P4-switch flow tables, DAC, and the distributed nature of the system contribute to efficient attack mitigation and enhanced network security.

Algorithm 5: Distributed Attack Mitigation Algorithm.

Inputs: P4-IoT-NT, NIoT-NT, A-IoT-T, and N-IoT-T

Output: Mitigating the attack

P4-IoT-NT ← IoT Network Traffic features extracted using P4
NIoT-NT ← New IoT Network Traffic
N-IoT-T ← Normal IoT Traffic
A-IoT-T ← Attack IoT Traffic
Src-IP ← Source IP Address
Src-prt ← Source port number
Src-MAC ← Source MAC Address
Dst-Node ← Destination IoT node
DAC ← Distributed Alert Channel

For each (NIoT-NT):
    If (NIoT-NT == A-IoT-T)
        Isolate infected, attacked IoT nodes
        Reroute IoT traffic
        Add extra high-priority flow rules → P4 switch flow tables
        Send FlowMod → All P4 Switches
        Add new flow entry → P4 Switch flow table
        Drop All packets → From attack source
        Block attacking Src-Mac, Src-IP, and Src-prt
        Publish Alert notification → using DAC → Other Controllers

    Else

        Forward N-IoT-T → Controller
        Send N-IoT-T → Dst-Node
    End if
End for

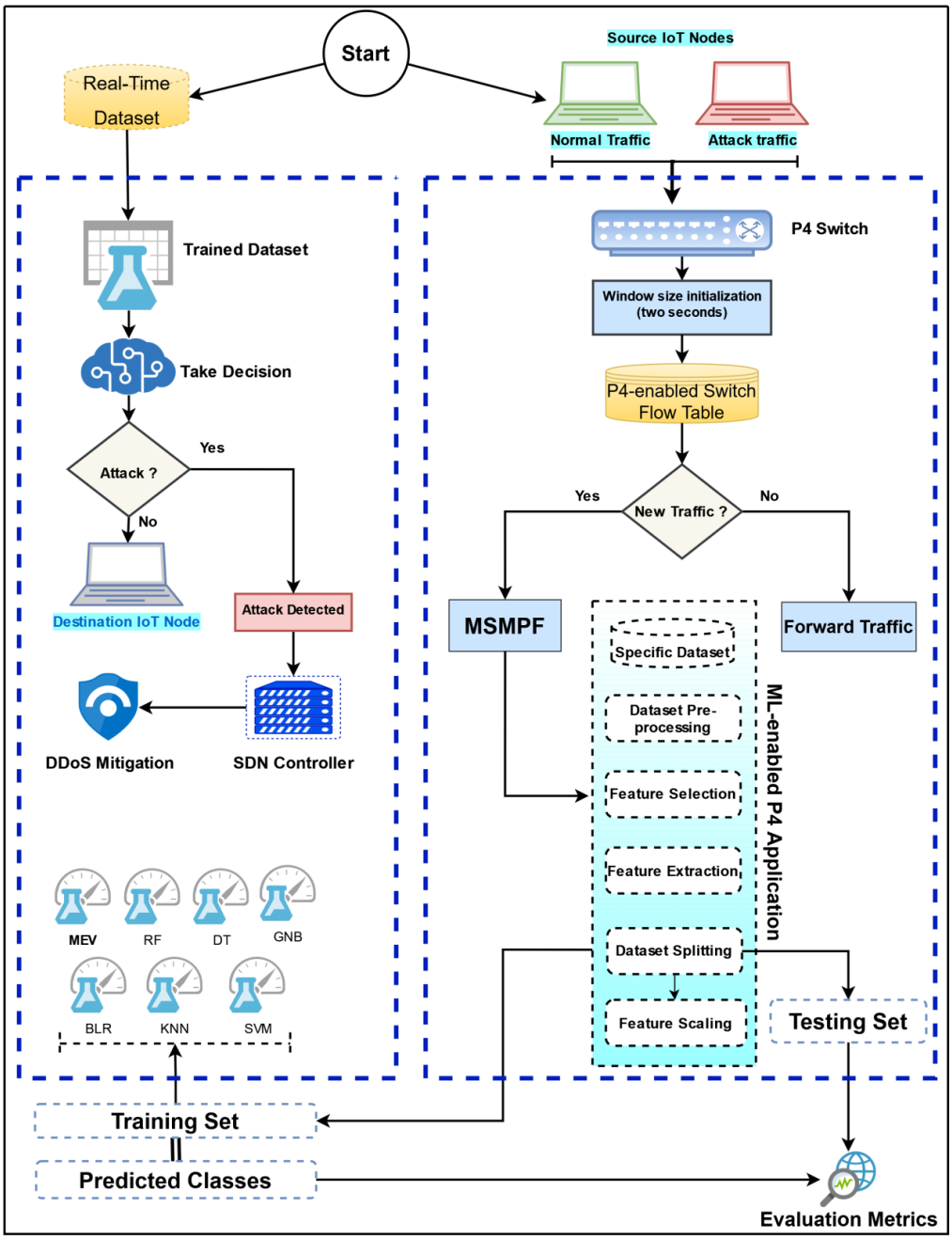

Figure 6. The general flowchart of the proposed P4-HLDMC framework.

4. Experimental Results

4.1. Experimental Settings

The experiments were conducted on an Ubuntu server 22.04 LTS virtual machine with an Intel core i7-1165G7 processor and 16 GB of RAM. The software tools used for developing and running the IoT test topologies included VMware Workstation 16 Pro, P4 compiler, Mininet-Wifi, and sFlow-RT. The SDN-based IoT network topology architecture included five Improved OpenDayLight (IODL) controllers, one master controller, and two domain controllers with two backup controllers, two P4-enabled switches, four access points (AP), and 20 IoT nodes. Four test cases were employed to evaluate the performance of the proposed detection framework: (1) multi-controller single-target attack (MC-STA), (2) multi-controller two-target attack (MC-TTA), (3) multi-controller four-target attack (MC-FTA), and (4) multi-controller eight-target attack (MC-ETA). Figure 7 shows the SDN-based IoT network test topology architecture of the proposed P4-HLDMC framework.

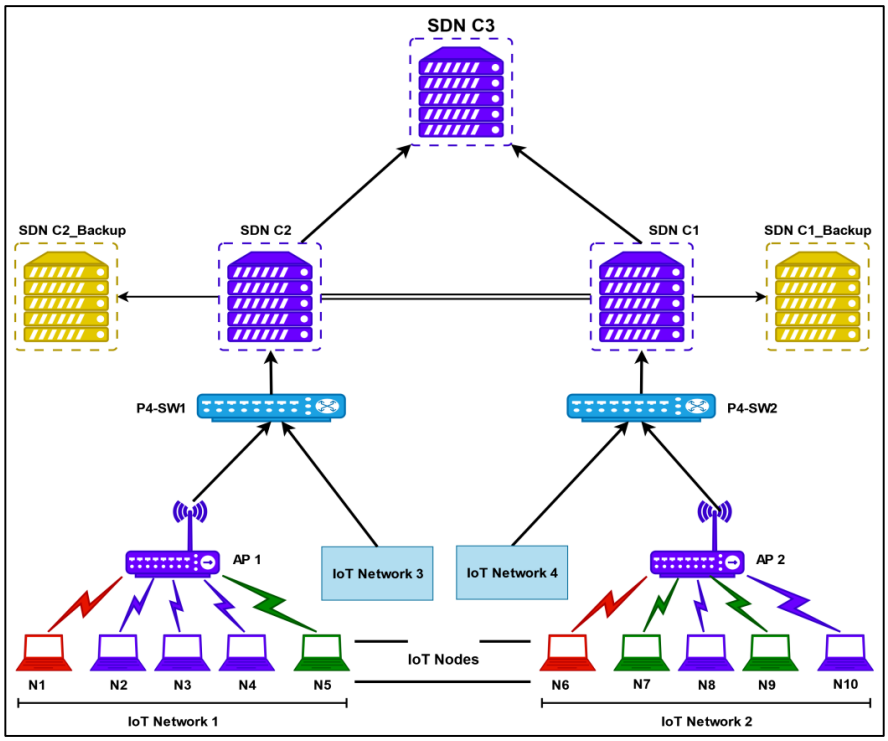

Figure 7. The SD-IoT network topology architecture of the proposed P4-HLDMC framework.

4.2. Experimental Test Cases

In the MC-STA test case, we utilized the (IoT Network 1) topology as shown in Figure 7 to perform two different experiments for both DDoS and ARP attacks, where the network was under the control of a master IDOL controller and two domain controllers. The proposed detection and mitigation P4 application was deployed on the P4-enabled switch, which was denoted by P4-SW1. The network consisted of N1 to N5 IoT nodes, with N2, N3, and N4 acting as legitimate nodes for regular traffic. N1 acted as an attacker, with N5 designated as the attack target. These IoT nodes were directly connected to the P4-SW1 switch through a wireless access point (AP 1). To evaluate the normal traffic conditions, we executed the *pingall* command to generate and exchange ICMP packets between the nodes and collect real-time P4 features. After that, we established a connection for normal traffic between all the network nodes, which remained open during the test. This allowed legitimate and attack traffic to be combined. Following this, we employed the *xterm* command from Mininet-Wifi to access the terminal of the attacker node, N1. We then launched two attacks, TCP flooding and ARP spoofing, on the N5 node. We utilized the *hping3* tool to generate the DDoS flooding attack, with an attack traffic rate of 15% of all the traffic routed to N5, while the remaining 85% were normal packets.

During an attack, only three traffic features, namely WDIPE, WSTPE, and WSUPE, which are dependent on entropy, will decrease, while the remaining 17 features will increase. The IO graph during the MC-STA test case is displayed in Figure 8, which depicts a Wireshark capture of the P4-SW1 eth1 interface during the attack. The attack traffic is so intense that the destination is unable to respond to legitimate hosts, as TCP requests peak at 800 packets per second, which is significantly higher than the normal rate of 4 packets per second.

The ARP flooding/poisoning attack is conducted by launching two scripts from the attacker's node, N1, using *Scapy*. In the first attack scenario, a script called IoT-ARP-Flood.py is executed on N1 to broadcast a large number of ARP requests with spoofed IP addresses to the entire network. The second attack scenario involves the use of a Scapy-written script called IoT-N5-Poison.py to manipulate the ARP table of node N5.

Upon executing the attack simulation, the legitimate host's ARP table is maliciously updated with MAC-ID credentials controlled by the attacker. This indicates

that the IP address of IoT node N5 is associated with the attacker's MAC address. As a result, an ARP Poisoning attack occurs, redirecting traffic intended for node N1 to be routed to node N3 instead of node N5.

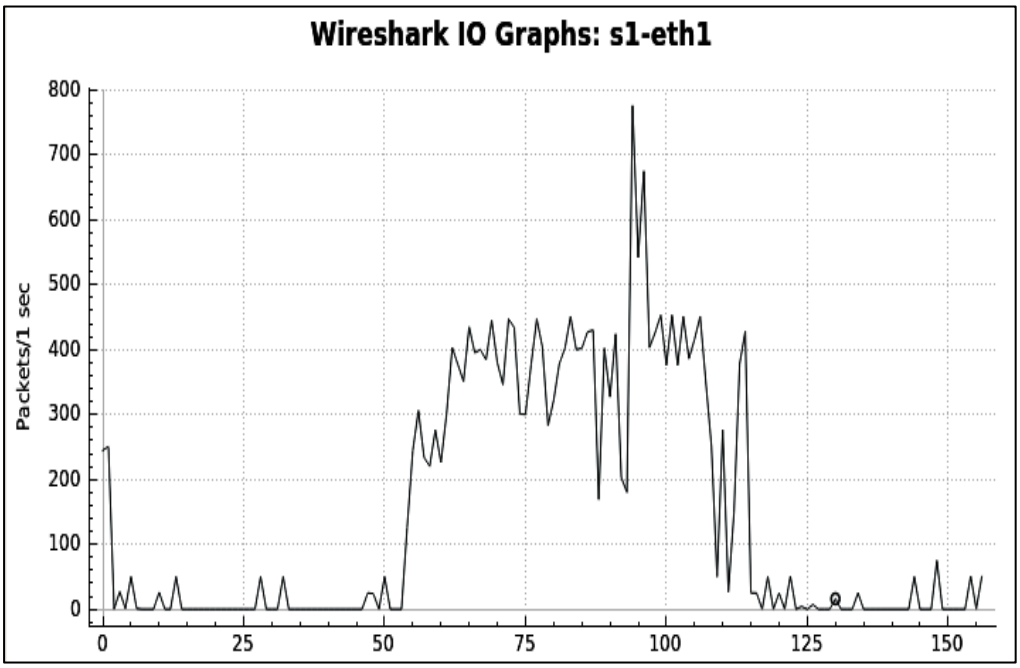

Figure 8. Wireshark capture of the TCP flooding attack during MC-STA test scenario.

In the MC-TTA test case, the network setup remained consistent with the previous scenario, except for the simultaneous targeting of multiple nodes by the attacker node (N6) from (IoT Network 2). N7 and N9 were both subjected to attack packets in this case. The proposed framework had to account for any differences in specific network features to achieve the highest possible accuracy in attack detection. Entropy was one such feature that could deviate from the previous scenario, increasing to values close to normal due to the attack being distributed among multiple victims. To detect attacks even at low rates, additional features beyond the WDIPE, WSTPE, and WSUPE had to be considered when calculating them within a given window. We assigned these features to four distinct levels to efficiently detect DDoS attacks in various testing scenarios, with attack rates varying from 10% to 90%. IoT Network 2, from Figure 7, was the topology utilized in this test case. The Mininet-Wifi dashboard displays a graphical representation of the total traffic exchanged between IoT network nodes in attack situations in the MC-TTA scenario, as presented in Figure 9. The two nodes 10.0.0.2 and 10.0.0.4 are targeted simultaneously. A notable difference can be observed in the total traffic transferred between the two scenarios. When all 20 nodes are connected for regular communication, the traffic peaked at 20 kbps. On the other hand, during the attack, the traffic escalated to over 65 Mbps, as shown in Figure 9.

The two test cases, MC-FTA and MC-ETA, were conducted using IoT Network 3 and IoT Network 4, as shown in Figure 7. The former involved an attack on four nodes simultaneously, while the latter involved an attack on eight nodes concurrently. The aim was to evaluate the effectiveness and efficiency of the proposed framework in handling diverse types and intensities of multi-target attacks. Attack rates ranging from 30% to 90% were used in these scenarios. The attack can be either UDP flood attack, TCP SYN flood attack, ICMP flood attack, ARP flooding, ARP poisoning or HTTP flood attack. An attack rate of 20% was not considered because it was distributed among four or eight victims, resulting in each victim receiving a number of packets within the expected normal range. The network traffic demand increases exponentially as the simulation runs longer, leading to significant resource consumption for both the P4 switches and controllers. Based on network measurements, the CPU utilization is at its maximum, with 63% of the RAM being used.

Additionally, a high packet rate of 65.8 packets per second and a significant bandwidth usage of 684 Kbps have been recorded. In Figure 10, the high traffic flow peaks correspond to DDoS attack-related network traffic, while the low dips indicate the impact of P4-HLDMC functionality. This functionality efficiently detects and mitigates the attack, restoring regular traffic flow via P4 switches. The evidence presented in Figure 10 convincingly demonstrates that the proposed P4-HLDMC framework can successfully identify and stop a multi-target attack within only 3 ms, restoring the network to its pre-attack state.

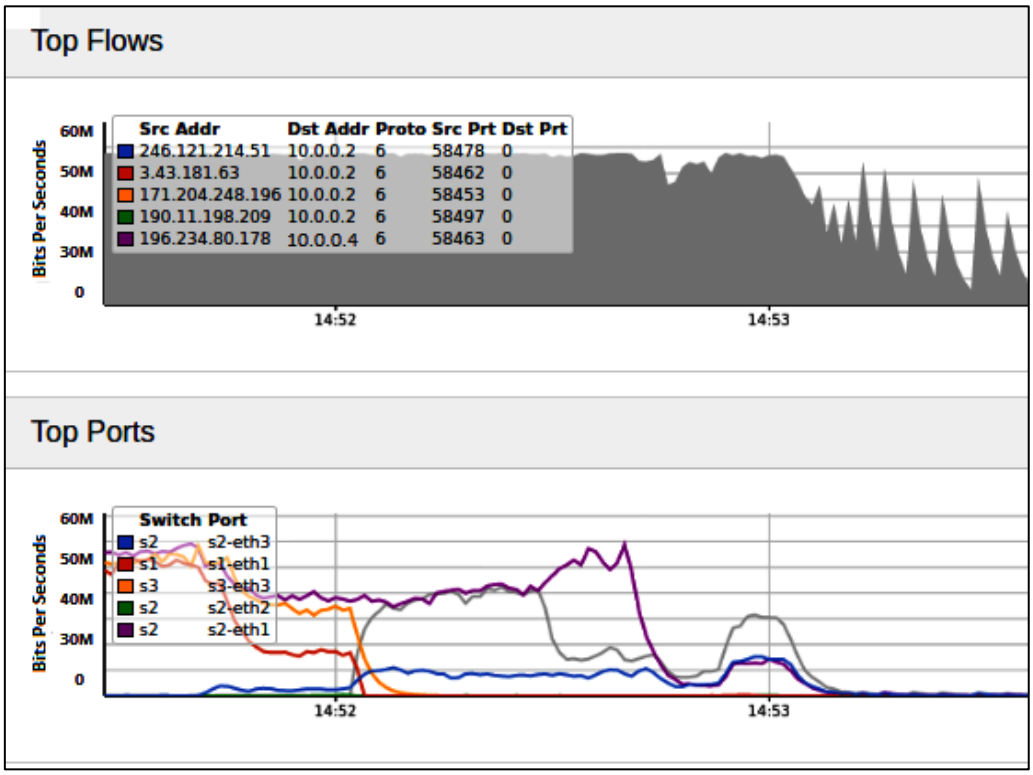

Figure 9. IoT traffic trends during the MC-TTA scenario.

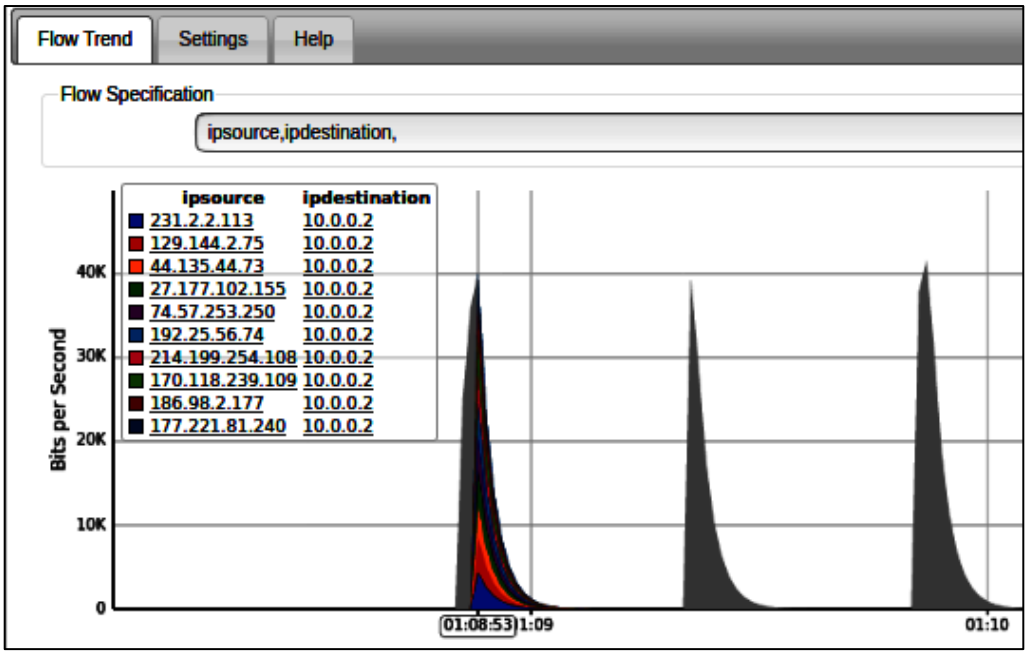

Figure 10. The mitigation process of the multi-target attack scenarios.

4.3. Performance Analysis and Evaluation

4.3.1. ML Models Evaluation

We evaluated the performance of the proposed framework by measuring several performance metrics, including accuracy (ACC), precision (PRE), F1-score (F1), recall (REC), true negative rate (TNR), negative predictive value (NPV), false positive rate (FPR), false discovery rate (FDR), false negative rate (FNR), Matthews Correlation Coefficient (MCC), area under the curve (AUC), average detection time (ADT), attack mitigation effectiveness (AME), and network latency (NL). The detection accuracy was measured as the percentage of correctly identified attack traffic out of the total attack traffic, while the false positive rate was measured as the percentage of legitimate traffic misidentified as attack traffic. The detection time was measured as the time it took the proposed framework to detect the attack traffic. The mitigation effectiveness was measured as the percentage of attack traffic that was successfully mitigated by the proposed framework. Finally, the network latency was measured as the time it took for a packet to travel from the source to the destination. All the metrics, except ADT, AME, and NL, were computed based on the confusion matrix presented in Figure 11. The mathematical definitions of these evaluation metrics are listed in Equations (15)–(25) as follows:

$$ACC = \frac{TP + TN}{FP + FN + TP + TN} \tag{15}$$

$$PRE = \frac{TP}{FP + TP} \tag{16}$$

$$NPV = \frac{TN}{FN + TN} \tag{17}$$

$$REC = \frac{TP}{FN + TP} \tag{18}$$

$$F1 = 2 * \frac{(PRE * REC\ )}{(PRE + REC\ )} \tag{19}$$

$$TNR = \frac{TN}{FP + TN} \tag{20}$$

$$FPR = \frac{FP}{TN + FP} \tag{21}$$

$$FDR = \frac{FP}{TP + FP} \tag{22}$$

$$FNR = \frac{FN}{TP + FN} \tag{23}$$

$$ADT = \frac{\sum_{j=1}^{tn} DTR_f}{tn} \tag{24}$$

$$MCC = \frac{(TP * TN) - (FP * FN)}{\sqrt{((TP + FP) * (TP + FN) * (TN + FP) * (TN + FN))}} \tag{25}$$

In these equations, FP and FN are critical factors that can affect the effectiveness of the attack detection algorithms. A FP is an event that the system incorrectly identifies as an attack, while a FN is when an actual attack is not detected by the system. TP and TN refer to the correct identification of an attack and non-attack event, respectively. In Equation (24), the variable *'DTRf'* is used to denote the time taken for detecting a single instance in a test, and the letter *'tn'* denotes the total number of experimental trials. In DDoS and ARP detection, the

aim is to minimize the occurrence of false positives and negatives while maximizing the true positives and negatives to ensure the efficient and accurate detection of attacks.

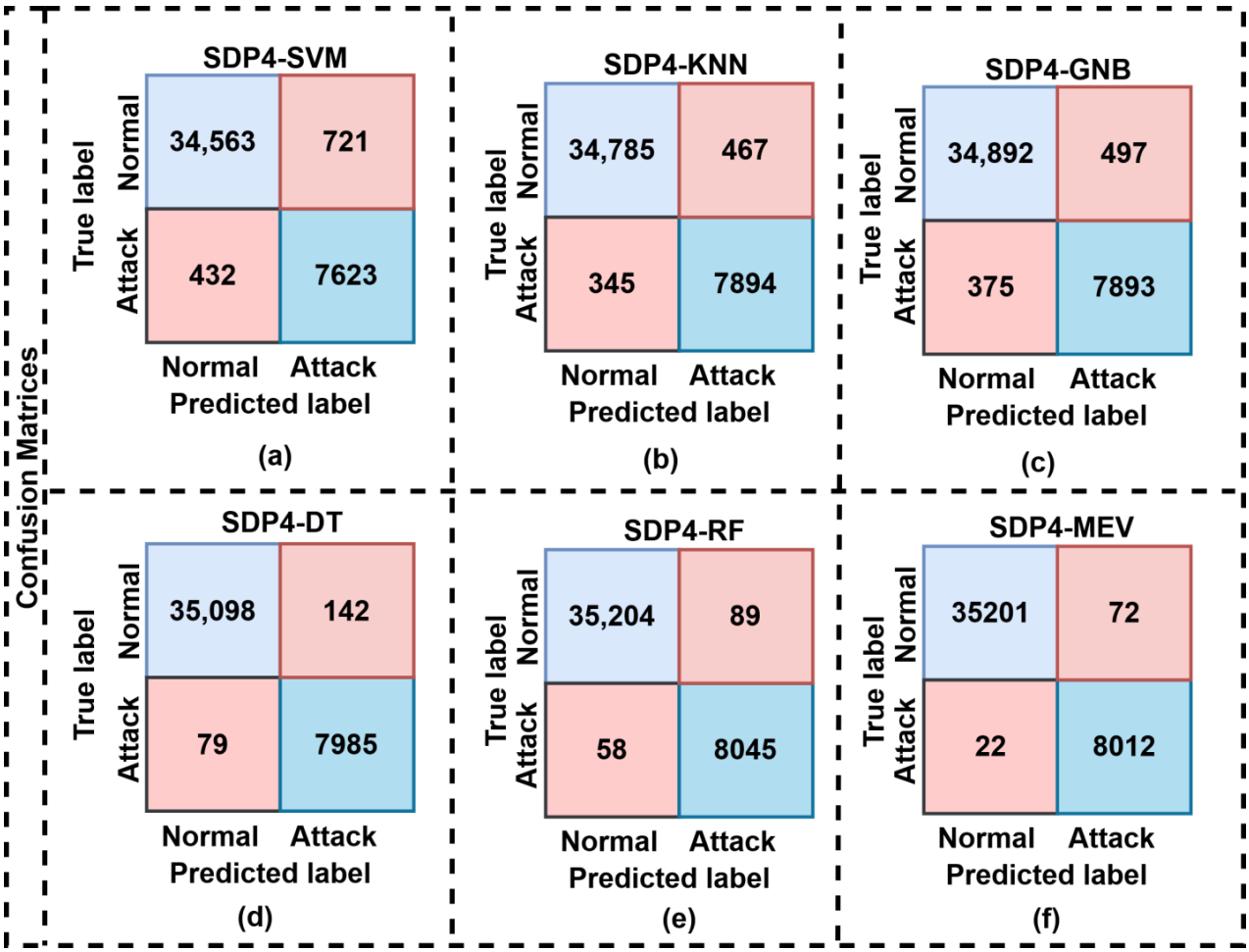

Figure 11. The confusion matrices for the tested ML mode.

Table 2 presents the evaluation results of the binary classification, while Table 3 showcases the results of the multi-class classification using seven ML classifiers: SDP4-MEV, SDP4-RF, SDP4-DT, SDP4-kNN, SDP4-GNB, and SDP4-BLR in the MC-ETA test scenario. These models were tested with three distinct datasets: Edge-IIoTset, TON_IoT, and X-IIoTID. The purpose of testing the models with different datasets was to assess their performance on unseen data and various attack types and conditions. Consequently, the models successfully addressed the issue of overfitting and demonstrated a good fit, particularly the SDP4-MEV model. The evaluation focused on the models' capability to detect and counter DDoS and ARP attacks in SD-IoT networks. We utilized several evaluation metrics, including ACC, PRE, F1, REC, SPC, NPV, FPR, FDR, AUC, MCC, FNR, and ADT. Analyzing these metrics is essential in evaluating the effectiveness of each model in detecting and mitigating the attacks. Table 4 provides the hyper-parameter configurations employed for all models.

Table 2. The evaluation results of the binary classification for the seven ML classifiers in MC-ETA test scenario using the three datasets.

Dataset	Model	ACC	PRE	F1	REC	TNR	NPV	AUC	MCC	FPR	FDR	FNR	ADT (ms)
Edge-IIoTset	SDP4-MEV	99.89%	99.54%	99.67%	99.79%	99.69%	99.97%	99.86%	99.22%	0.16%	0.46%	0.27%	2.24
	SDP4-RF	98.78%	98.48%	98.63%	98.53%	98.60%	99.51%	99.67%	97.57%	0.25%	1.52%	0.54%	19.36
	SDP4-DT	98.11%	97.51%	97.81%	98.19%	98.25%	97.96%	98.69%	96.72%	1.75%	2.44%	0.81%	29.17
	SDP4-kNN	97.92%	96.64%	97.47%	96.94%	97.08%	96.82%	96.71%	95.93%	2.92%	3.36%	1.81%	17.29
	SDP4-GNB	97.84%	96.04%	97.39%	96.14%	96.98%	96.71%	96.78%	95.47%	4.72%	3.06%	1.96%	9.95
	SDP4-SVM	96.74%	95.54%	96.97%	95.81%	95.08%	94.90%	95.18%	94.09%	4.12%	6.66%	3.76%	62.31
	SDP4-BLR	91.54%	92.98%	94.27%	89.87%	90.83%	92.78%	90.68%	92.87%	5.17%	9.02%	8.13%	20.8
TON_IoT	SDP4-MEV	98.35%	97.78%	98.11%	98.62%	98.72%	97.93%	98.45%	97.31%	1.28%	2.22%	1.38%	15.69
	SDP4-RF	97.62%	96.94%	97.38%	97.10%	97.24%	96.82%	97.12%	95.24%	2.76%	3.06%	2.90%	40.45
	SDP4-DT	96.89%	95.76%	96.43%	96.07%	96.20%	95.82%	96.04%	94.12%	3.80%	4.24%	3.93%	45.18
	SDP4-kNN	95.78%	94.32%	95.06%	94.62%	94.78%	94.52%	94.64%	92.96%	5.22%	5.68%	5.38%	53.27
	SDP4-GNB	95.41%	93.87%	94.77%	93.98%	94.28%	93.82%	93.92%	92.13%	5.72%	6.13%	6.02%	49.81
	SDP4-SVM	94.27%	92.73%	93.44%	93.05%	93.15%	92.87%	92.99%	90.49%	6.85%	7.27%	6.95%	74.36
	SDP4-BLR	89.14%	88.46%	90.38%	87.94%	88.23%	89.28%	88.32%	88.21%	11.77%	11.54%	12.06%	27.94
X-IIoTID	SDP4-MEV	99.23%	98.65%	99.02%	98.82%	98.92%	98.76%	98.98%	97.66%	1.08%	1.35%	1.18%	19.42
	SDP4-RF	98.56%	97.92%	98.35%	98.19%	98.28%	98.09%	98.23%	96.97%	1.72%	2.08%	1.81%	21.78
	SDP4-DT	97.84%	97.12%	97.54%	97.56%	97.65%	97.34%	97.48%	96.05%	2.35%	2.88%	2.44%	27.91
	SDP4-kNN	97.03%	95.98%	96.77%	96.57%	96.72%	96.43%	96.60%	95.24%	3.28%	4.02%	3.43%	36.79
	SDP4-GNB	96.72%	95.43%	96.34%	95.62%	95.84%	95.58%	95.68%	93.97%	4.16%	4.57%	4.38%	31.57
	SDP4-SVM	95.61%	94.28%	95.01%	94.84%	94.98%	94.77%	94.91%	92.94%	5.02%	5.72%	5.16%	46.85
	SDP4-BLR	90.48%	89.67%	91.12%	89.84%	90.08%	89.72%	89.92%	88.85%	9.92%	10.33%	10.16%	19.79

The ACC metric measures the overall accuracy of the model in classifying both attack and normal traffic. The results in Table 2 show that SDP4-MEV outperforms all other models, achieving an average accuracy rate of 99.89%. SDP4-RF and SDP4-DT follow closely with accuracy rates of 98.78% and 98.11%, respectively. SDP4-kNN, SDP4-GNB, SDP4-SVM, and SDP4-BLR achieve accuracy rates of 97.92%, 97.84%, 96.74%, and 91.54%, respectively. These results indicate that the SDP4-MEV model is the most effective in accurately identifying the attack, followed by SDP4-RF and SDP4-DT.

Additionally, we evaluated the models' accuracy consistency rate (ACR) by changing the flow intensity within the range of 5000 to 50,000. SDP4-BLR and SDP4-SVM exhibited the lowest consistency rate, as their accuracy increased initially but then decreased as the flow rate increased. Specifically, SDP4-BLR achieved an accuracy of 89.84% at a flow rate of 5000, which increased to 93.43% at 3000, but then decreased to 90.57% at a flow rate of 50,000. Similarly, SDP4-SVM demonstrated an accuracy of 92.01% at a flow rate of 5000, which increased to 94.28% at 35,000, but then dropped to 91.64% at a flow rate of 50,000. This inconsistent detection accuracy while varying the traffic rate could negatively impact the models' performance during high flow intensities, such as in the case of a DDoS attack. In contrast, SDP4-GNB and SDP4-kNN performed similarly, while SDP4-DT and SDP4-RF were the best models following SDP4-MEV. Figure 12 shows the ACR of the seven ML models.

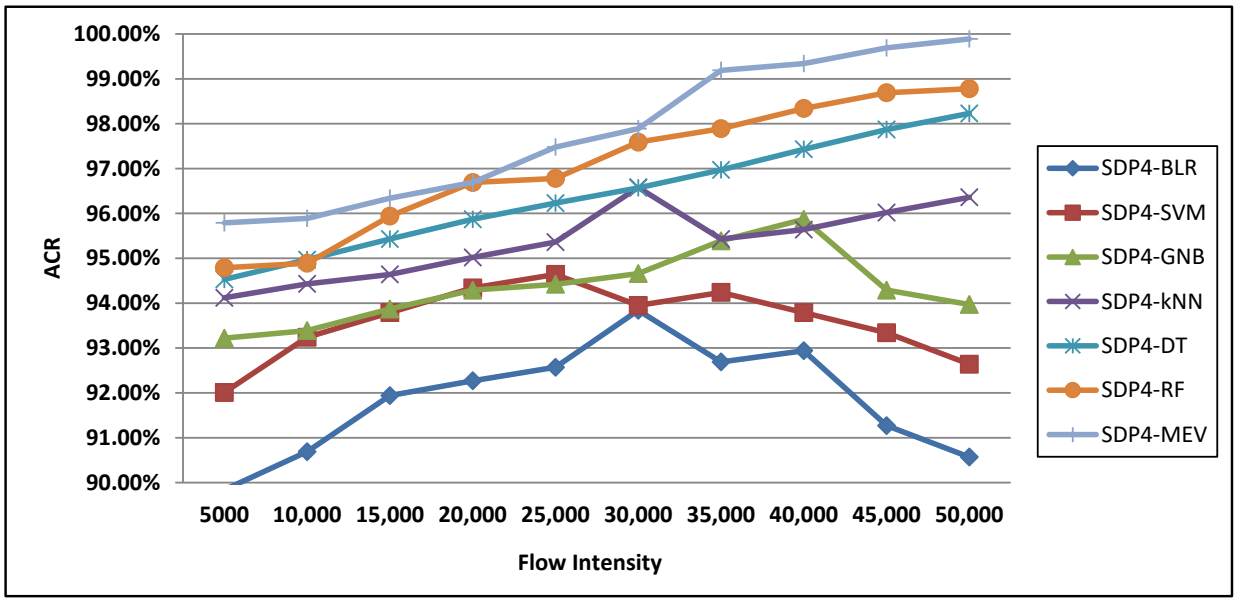

Figure 12. The accuracy consistency rate (ACR) of the tested ML models.

The PRE and FDR metrics are measures of the model's ability to identify true positive samples, or in other words, correctly identifying DDoS and ARP attacks. SDP4-MEV achieves the highest PRE and lowest FDR values of 99.54% and 0.46%, respectively. SDP4-RF and SDP4-DT follow with PRE values of 98.48% and 97.51% and FDR values of 1.52% and 2.44%, respectively. The SDP4-kNN, SDP4-GNB, SDP4-SVM, and SDP4-BLR models achieve lower PRE values of 96.64%, 96.04%, 95.54%, and 92.98%, respectively, indicating a higher rate of false positives.

The F1 metric is the harmonic mean of PPV and REC and is an effective measure of the model's overall performance. SDP4-MEV achieves the highest F1 score of 99.67%, indicating its effectiveness in detecting DDoS and ARP attacks. SDP4-RF and SDP4-DT follow closely with F1 scores of 98.63% and 97.81%, respectively. The SDP4-kNN and SDP4-GNB achieve similar results with F1 scores of 97.47% and 97.39%. SDP4-SVM and SDP4-BLR models achieve lower F1 scores of 96.97% and 94.27%, respectively.

The REC, SPC, and NPV metrics measure the ability of the model to identify true positive, true negative, and correctly identify normal traffic. SDP4-MEV achieves the

highest REC and SPC values of 99.79% and 99.69%, respectively. SDP4-RF and SDP4-DT follow closely with REC values of 98.53% and 98.19% and SPC values of 98.60% and 98.25%, respectively. The SDP4-kNN and SDP4-GNB again achieve similar results with REC values of 96.94% and 96.14%. SDP4-SVM follows with REC value of 95.81%. SDP4-BLR model achieves the lowest REC and SPC values of 89.87% and 90.83%, indicating a higher rate of false positives and negatives. SDP4-MEV also achieves the highest NPV value of 99.97%, indicating its effectiveness in correctly identifying normal traffic.

Table 3. The evaluation results of the multi-class classification for the seven ML classifiers in MC-ETA test scenario using the Edge-IIoTset dataset.

Attack Type	Model	ACC	PRE	F1	REC	TNR	NPV	AUC	MCC	FPR	FDR	FNR	ADT (ms)
DDoS-TCP	SDP4-MEV	99.45%	98.92%	99.19%	99.28%	99.52%	99.29%	99.15%	98.61%	1.12%	0.72%	2.27%	4.74
	SDP4-RF	98.76%	97.93%	98.34%	98.09%	98.63%	97.88%	98.07%	97.24%	2.07%	1.91%	3.54%	8.27
	SDP4-DT	97.92%	96.81%	97.28%	97.43%	97.65%	97.47%	97.12%	97.72%	3.19%	2.57%	3.61%	39.45
	SDP4-kNN	97.61%	96.42%	97.07%	96.87%	97.03%	96.89%	96.76%	96.76%	3.58%	3.13%	4.71%	42.34
	SDP4-GNB	96.92%	95.74%	96.33%	95.94%	96.76%	96.00%	95.87%	95.43%	4.26%	4.06%	6.06%	29.58
	SDP4-SVM	95.88%	94.62%	95.07%	94.82%	95.11%	94.89%	94.43%	94.43%	5.38%	5.18%	8.36%	68.59
	SDP4-BLR	92.35%	91.18%	91.96%	90.81%	92.89%	91.04%	89.90%	85.91%	8.82%	9.19%	12.67%	34.56
DDoS-UDP	SDP4-MEV	99.37%	98.79%	99.08%	99.15%	99.47%	96.12%	98.45%	96.12%	1.34%	1.21%	2.78%	4.56
	SDP4-RF	98.71%	97.87%	98.29%	98.02%	98.57%	94.76%	97.12%	94.76%	2.45%	2.18%	4.03%	16.87
	SDP4-DT	97.85%	96.78%	97.23%	97.37%	97.57%	93.82%	96.04%	93.82%	3.59%	2.98%	4.56%	25.44
	SDP4-kNN	97.56%	96.34%	96.99%	96.83%	96.96%	92.67%	94.64%	92.67%	4.12%	3.87%	5.71%	14.62
	SDP4-GNB	96.85%	95.68%	96.25%	95.91%	96.75%	91.32%	93.92%	91.32%	4.83%	4.21%	7.02%	7.88
	SDP4-SVM	95.81%	94.54%	94.99%	94.74%	95.06%	89.76%	92.99%	89.76%	6.02%	5.44%	9.12%	54.79
	SDP4-BLR	94.45%	92.24%	93.34%	92.73%	95.19%	79.12%	88.32%	79.12%	9.87%	11.19%	15.47%	18.32
DDoS-ICMP	SDP4-MEV	99.44%	98.66%	99.05%	99.23%	99.62%	99.20%	99.10%	97.21%	1.09%	0.91%	2.14%	9.98
	SDP4-RF	98.39%	96.95%	97.66%	97.31%	98.59%	97.42%	97.54%	95.87%	2.35%	2.06%	3.78%	17.65
	SDP4-DT	97.24%	95.21%	96.08%	95.47%	96.61%	95.70%	95.73%	96.44%	3.42%	2.87%	3.92%	27.12
	SDP4-kNN	97.02%	94.57%	95.66%	94.89%	96.46%	94.99%	94.97%	94.67%	3.91%	3.72%	4.82%	15.77
	SDP4-GNB	96.73%	94.19%	95.26%	94.54%	96.34%	94.64%	94.61%	92.76%	4.62%	4.34%	6.58%	8.54
	SDP4-SVM	95.91%	92.90%	93.73%	93.20%	96.12%	93.46%	93.21%	90.64%	5.89%	5.76%	8.21%	59.08
	SDP4-BLR	93.94%	89.83%	92.22%	90.68%	95.12%	91.78%	89.78%	82.46%	8.19%	9.67%	11.92%	21.56
DDoS-HTTP	SDP4-MEV	98.62%	97.25%	98.02%	97.54%	98.71%	97.40%	98.16%	99.12%	0.76%	0.53%	1.02%	11.75
	SDP4-RF	97.91%	96.73%	97.30%	96.98%	98.08%	96.84%	97.32%	98.02%	1.48%	1.19%	2.57%	12.93
	SDP4-DT	96.79%	95.46%	96.11%	95.66%	97.01%	95.62%	96.00%	98.45%	2.12%	1.87%	3.01%	21.78
	SDP4-kNN	96.15%	94.86%	95.61%	95.11%	96.24%	95.03%	95.34%	97.32%	2.79%	2.54%	4.15%	13.47
	SDP4-GNB	94.82%	93.37%	94.14%	93.41%	95.03%	93.75%	94.03%	96.01%	3.42%	3.19%	5.64%	12.21
	SDP4-SVM	92.73%	90.82%	91.76%	91.24%	93.22%	91.45%	91.39%	92.56%	4.57%	4.42%	7.38%	48.21
	SDP4-BLR	89.46%	86.93%	88.06%	87.28%	90.12%	87.40%	88.34%	87.92%	7.98%	8.84%	11.56%	17.89
ARP-Spoof	SDP4-MEV	99.15%	98.86%	96.38%	96.61%	97.60%	98.44%	97.20%	98.76%	1.23%	0.81%	1.45%	9.87
	SDP4-RF	96.42%	94.98%	95.73%	95.21%	96.64%	94.92%	95.98%	97.54%	2.01%	1.78%	3.21%	18.67
	SDP4-DT	95.06%	92.79%	94.53%	93.58%	95.58%	92.95%	94.14%	96.98%	3.01%	2.43%	3.36%	28.56
	SDP4-kNN	94.23%	91.73%	93.47%	92.36%	94.68%	91.94%	93.05%	95.89%	3.45%	3.01%	4.92%	16.21
	SDP4-GNB	92.58%	89.98%	91.77%	90.32%	93.07%	90.23%	92.15%	93.76%	4.12%	3.94%	5.89%	19.76
	SDP4-SVM	89.97%	86.46%	88.04%	87.07%	91.04%	86.93%	88.02%	91.34%	5.21%	5.02%	8.02%	61.32
	SDP4-BLR	86.02%	81.95%	83.48%	82.40%	88.63%	82.56%	84.24%	86.87%	8.67%	9.01%	12.12%	19.45
ARP-Poison	SDP4-MEV	99.78%	98.39%	98.08%	99.63%	99.20%	99.57%	97.88%	98.34%	1.34%	0.97%	1.78%	7.56
	SDP4-RF	95.92%	94.34%	95.20%	94.68%	96.14%	94.48%	94.99%	95.12%	2.15%	1.79%	3.12%	17.23
	SDP4-DT	94.42%	92.06%	93.49%	92.68%	95.12%	92.74%	93.01%	94.56%	3.23%	2.51%	3.68%	27.45
	SDP4-kNN	93.57%	90.85%	92.62%	91.32%	94.29%	91.40%	92.42%	95.67%	3.78%	3.23%	4.89%	15.98
	SDP4-GNB	91.68%	88.05%	90.47%	89.36%	93.00%	88.77%	90.13%	94.43%	4.43%	4.15%	6.45%	29.32
	SDP4-SVM	88.52%	83.64%	86.05%	85.04%	91.37%	84.23%	87.71%	90.98%	5.56%	5.32%	8.01%	60.12
	SDP4-BLR	84.63%	78.35%	81.23%	79.61%	89.84%	78.82%	83.22%	85.12%	8.91%	9.34%	12.89%	18.76

Among all the models, SDP4-MEV has the lowest false-alarm rates, with an FPR of 0.16%, FDR of 0.46%, and FNR of 0.27%. The AUC values for all models were above 90%, ranging from 90.68% to 99.86%. The AUC is a measure of the overall performance of a model and is used to evaluate the model's ability to distinguish between positive and negative samples. The AUC of SDP4-MEV is the highest at 99.86%, indicating that it has a high ability to distinguish between attack and normal flows.

The MCC values for all models were high, ranging from 92.87% to 99.22%. The MCC is a measure of the correlation between the predicted and actual labels and takes into account both true and false positives and negatives. SDP4-MEV also has the highest MCC of 99.22%, indicating a strong correlation between predicted and actual classifications. In conclusion, the SDP4-MEV exhibits the lowest ADT value of 2.24 ms, whereas the SDP4-SVM demonstrated the highest ADT of 62.31 ms. Based on the results, the ML models exhibited the highest performance in the Edge-IIoTset dataset, followed by the X-IIoTID dataset, and then the TON_IoT dataset.

Our experimental results show that the proposed framework is effective in detecting and mitigating the attacks in SD-IoT networks. The detection accuracy ranged from 98.9% to 99.97%, depending on the number of victim nodes and the severity of the attack. The false positive rate was very low, ranging from 0.11% to 0.21%, indicating that the proposed framework can effectively distinguish between legitimate traffic and attack traffic. The detection time ranged from 2 ms to 6 ms, and the mitigation effectiveness ranged from 98.5% to 100%, depending on the number of victim nodes and the severity of the attack. Finally, the network latency was very low, ranging from 1 ms to 5 ms, indicating that the proposed framework can operate in real-time.

Table 4. The hyperparameter configurations for the ML models.

Model	Parameter	Selected Value
SDP4-MEV	Base classifier	RF, DT, kNN, GNB, SVM, and BLR
	No. of Base classifiers	6
	Ensemble method	Weighted voting
SDP4-RF	Splitting criterion	Gini
	Number of trees	100
	Min. samples leaf	1
	Min. samples split	2
	Max features	17
SDP4-DT	Splitting criterion	Entropy
	Number of trees	10
	Min. samples leaf	1
SDP4-kNN	Number of neighbors K	3
	Neighbors weight	Uniform
SDP4-GNB	Regularization parameter	10^4
SDP4-SVM	Kernel	Sigmoid
	Reg. parameter coefficient	10^3
	Kernel coefficient	10^{-2}
SDP4-BLR	Regularization parameter (C)	10^2
	Solver	liblinear

4.3.2. Comparing the Proposed Framework with Other Existing Methods

According to the experimental results, the proposed framework outperforms most of the existing cutting-edge approaches across 14 different evaluation criteria. P4-HLDMC achieves exceptional benchmarks across various metrics, including accuracy, F1-Score, sensitivity, true negative rate, positive predictive value, negative predictive value, Matthews Correlation Coefficient (MCC), area under the curve (AUC), false positive rate, false negative rate, false detection rate, average latency, and average detection time. These bench-

marks are as follows: 99.89%, 99.67%, 99.79%, 99.69%, 99.54%, 99.97%, 99.86%, 99.22%, 0.16%, 0.27%, 0.46%, 3.83 ms, and 2.24 ms, respectively. This demonstrates the effectiveness and superiority of the proposed framework over other existing solutions. Comparing our proposed method to existing approaches, the best accuracy among the existing methods was 93% with a single controller and without consistent or stateful packet processing. In contrast, our proposed method achieved a higher average accuracy of 99.89% with multi-controller architecture and a distributed mitigation strategy. Figure 13 shows the graphical representation of the accuracy metric for the compared ML/DL methods. Furthermore, the effectiveness of the P4-HLDMC method is demonstrated across multiple datasets, including Edge-IIoTset, TON_IoT, and X-IIoTID dataset, which are considered the most comprehensive SD-IoT datasets used for intrusion detection. This indicates its ability to handle diverse network environments and various types of attacks. In contrast, other methods such as CNN, DALCNN, and FFCNN, listed in Table 5, utilized datasets that are not suitable for detecting current IoT attacks, while some relied on simulated or generated traffic such as DRL-IPS, SMO, and kNN. Additionally, certain methods focused on specific datasets or particular types of attacks, such as DRL-IPS and DAD. Overall, the P4-HLDMC method stands out as the superior approach due to its low false positive rate, high average accuracy, minimum and most relevant number of features, and versatility in handling different datasets and attack scenarios. Its utilization of P4-enabled switches, modular design, and hierarchical logically distributed multi-controller architecture ensures efficient communication and synchronization between controllers, enabling rapid detection and mitigation of attacks. Moreover, this approach enhances the scalability and reliability of the system, enabling it to effectively defend against sophisticated and coordinated attacks that conventional defenses may struggle to handle.

Table 5. Settings comparison of the proposed framework to existing ML/DL approaches.

Method	Architecture	Dataset	Data Plane	Attack Type	Environement	Scalability	FPR	ACC
DRL-IPS [33]	Single	Mininet simulation	OpenFlow	Low-rate	SDN	No	N/A	98.00%
AutoML [36]	N/A	UNSWNB15, and CICIDS2017	Traditional	Intrusion	N/A	No	02.70%	97.30%
SD-Reg [34]	Single	InSDN, and CSE-CIC-IDS2018	OpenFlow	Intrusion	SDN	Yes	N/A	98.54%
CNN [35]	Single	CicDDoS 2019	OpenFlow	DDoS	SD-IoT	No	00.50%	99.50%
DT [47]	N/A	X-IIoTID dataset	N/A	Intrusion	IIoT	Yes	00.40%	99.58%
Kaur [44]	N/A	Kismet simulation	N/A	ARP	Traditional	No	Low	93.00%
DAD [39]	Single	P4-extracted data	P4	TCP flooding	SDN	Yes	02.00%	98.00%
SMO [48]	Single	Mininet simulation	OpenFlow	DoS	SDN	No	N/A	97.60%
kNN [49]	Single	Mininet simulation	OpenFlow	DDoS	SDN	No	00.97%	98.80%
FFCNN [50]	Single	CIC DoS 2017	OpenFlow	Low-rate	SD-IoT	No	N/A	99.00%
P4-HLDMC	Multi	Edge-IIoTset, TON_IoT, and X-IIoTID dataset	P4	DDoS-ARP	SD-IoT	Yes	0.16%	99.89%

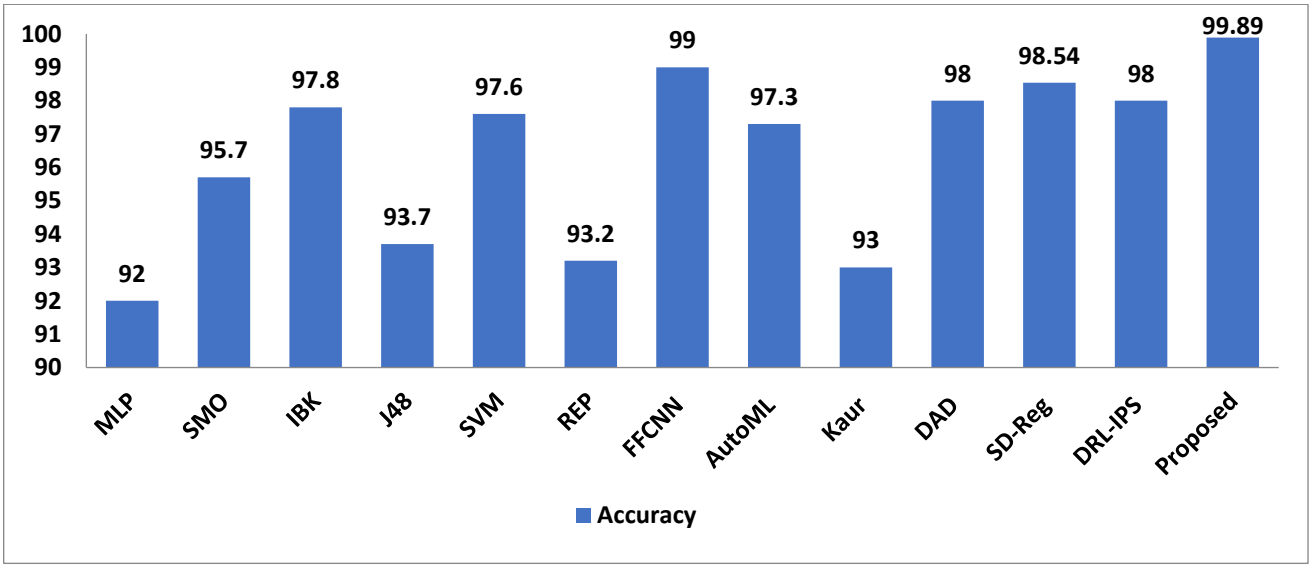

Figure 13. Graphical representation of the accuracy metric for the compared ML/DL methods.

4.3.3. The Distributed Multi-Controllers Evaluation

In this section, we evaluate the performance of different controllers in a proposed SD-IoT network topology using the P4-HLDMC framework. The controllers evaluated include open-source controllers such as OpenDayLight, Floodlight, Ryu, POX, and our proposed Improved OpenDay Light (IODL) controller. We measure the performance of each controller based on six metrics: throughput, delay, packet loss (PL), the number of packets processed per second (PPPs), CPU consumption, and memory consumption. Throughput measures the amount of data that can be transmitted through the network within a given time period, while delay measures the time it takes for data packets to be transmitted from the source to the destination. The evaluation results for the five aforementioned controllers in the four test cases—MC-STA, MC-TTA, MC-FTA, and MC-ETA—are listed in Table 6.

For the MC-STA test case, the IODL controller outperformed the other controllers in terms of throughput and delay. Specifically, the IODL controller achieved a throughput of 17.62 Gbps, which is higher than the throughput achieved by the other controllers. The delay of the IODL controller was also lower than the delay of all controllers, indicating a faster response time to attack detection and mitigation.

For the MC-TTA test case, the IODL controller achieved a higher throughput of 18.23 Gbps and lower delay of 3.26 ms compared to the other controllers. For the MC-FTA and MC-ETA test cases, the OpenDayLight, Floodlight, and Ryu controllers achieved similar throughput and delay, while the IODL controller achieved the highest throughput of 19.57 Gbps and the lowest delay of 4.12 ms, indicating better performance than the other controllers. The results of the experiments showed that the OpenDayLight controller outperformed the other controllers in terms of the number of packets processed per second, with a maximum throughput of 23,485 packets per second, followed by Floodlight with a maximum throughput of 21,945 packets per second and Ryu with a maximum throughput of 20,895 packets per second. The POX controller achieved the lowest PPPs, with a maximum of 18,350 packets per second across all test cases. The IODL controller had the highest maximum throughput of 28,640 packets per second, indicating better processing capacity than the other controllers.

The results also showed that the OpenDayLight controller had lower CPU and memory usage compared to the other controllers. The IODL controller had the lowest CPU consumption of 9.5% and memory consumption of 8.7%, indicating better resource utilization than the other controllers. From Table 6, we can obtain the average throughput and latency results for the different SDN controllers across all four test cases. As can be seen from the table, the IODL controller outperforms the other controllers. The average

throughput achieved with IODL is 18.54 Gbps, while the average latency is 3.83 ms. In contrast, the average throughput achieved with Floodlight and Ryu is 13.17 Gbps and 12.24 Gbps, respectively, while the average latency is 34.72 ms and 42.71 ms, respectively. These results demonstrate that the proposed IODL controller is the best choice for implementing the proposed framework. This is due to the fact that IODL provides improved network resource allocation, scalability, and stability. It also enhances the overall performance of the network by reducing the delay and increasing the throughput. The utilization of the MCDI interface and DAC channel significantly contributes to the overall improvement of the network's performance. Notably, the delay is reduced, allowing for faster and more responsive communication within the network. This reduction in delay leads to enhanced real-time interactions between the controllers. Moreover, the throughput is increased, enabling higher data transmission rates and improved network efficiency. In addition to the benefits in delay and throughput, the proposed MCDI interface and DAC channel also address the issue of packet loss. By employing advanced detection and mitigation techniques, these components effectively mitigate packet loss, ensuring the reliable and uninterrupted flow of data throughout the network. Consequently, the network achieves higher reliability and data integrity.

Table 6. Performance evaluation results for different controllers.

Test Case	Controller	Throughput (Gbps)	Delay (ms)	Packet Loss	PPPs	CPU	Memory
MC-STA	OpenDayLight	14.48	19.22	0.17	23,485	19.7%	22.6%
	Floodlight	12.58	25.96	0.21	21,945	21.2%	27.2%
	Ryu	12.02	36.11	0.23	20,895	28.9%	29.5%
	Pox	10.54	47.52	0.29	18,350	32.1%	34.1%
	Proposed IODL	17.62	02.19	0.12	28,640	09.5%	08.7%
MC-TTA	OpenDayLight	14.85	22.19	0.44	19,250	36.4%	27.6%
	Floodlight	12.63	33.87	0.76	20,680	41.2%	31.2%
	Ryu	11.34	45.19	0.81	18,460	37.8%	37.5%
	Pox	11.21	56.53	0.82	18,145	42.1%	32.1%
	Proposed IODL	18.23	03.26	0.36	25,040	11.1%	10.2%
MC-FTA	OpenDayLight	15.22	26.45	0.27	9805	23.6%	22.6%
	Floodlight	14.12	37.21	0.31	9425	25.1%	27.2%
	Ryu	13.56	37.36	0.33	9130	26.2%	29.5%
	Pox	13.21	48.01	0.35	8965	27.8%	32.1%
	Proposed IODL	18.74	04.12	0.23	22,475	12.9%	12.4%
MC-ETA	OpenDayLight	12.58	36.23	0.29	8215	24.8%	22.6%
	Floodlight	13.36	41.87	0.44	8745	26.3%	27.2%
	Ryu	12.07	52.19	0.47	8230	28.5%	29.5%
	Pox	11.94	74.53	0.49	8075	29.9%	32.1%
	Proposed IODL	19.57	05.76	0.33	19,480	14.8%	12.7%

5. Conclusions and Future Work

This study introduces a novel and comprehensive framework for the detection and mitigation of DDoS and ARP attacks in SD-IoT networks. The proposed P4-HLDMC framework leverages a combination of machine learning (ML), stateful P4, and SDN-based hierarchical logically distributed multi-controller architecture to provide a groundbreaking approach for attack detection in the IoT domain. With the incorporation of 17 new features, including 12 computed features and 5 P4-extracted features, the framework effectively identifies DDoS and ARP attacks. The ML model overcomes over-fitting challenges by accurately identifying attacks across diverse datasets and attack scenarios.

The framework consists of four modules: the Multi-Controller Dedicated Interface (MCDI), the Distributed Alert Channel (DAC), the Multi-State Matching Pipeline Function (MSMPF), and the Modified Ensemble Voting Algorithm (MEV). The MCDI interface ensures real-time and consistent attack detection through an asynchronous threading system,

while the DAC channel facilitates efficient communication and data-control overhead reduction between controllers. The MSMPF module employs nine state tables to analyze and detect IoT network traffic features, enhancing the controller's protection against DDoS and ARP attacks. The MEV module utilizes six classifiers to identify anomalies in P4-extracted traffic patterns, achieving early and accurate attack detection at the data plane level. Additionally, the framework employs a distributed approach for effective handling of larger-scale attacks.

The proposed framework successfully detects attacks in multi-target attack scenarios, surpassing conventional defense mechanisms. Experimental results using real-world IoT network traffic datasets demonstrate the framework's high detection rates, low false-alarm rates, low latency, high throughput, and minimal average detection time, outperforming existing methods. The study also introduces the Improved OpenDayLight (IODL) controller as the optimal choice for implementing the proposed framework, thanks to its enhanced network resource allocation, scalability, and stability. In conclusion, the proposed framework presents an effective solution for detecting and mitigating DDoS and ARP attacks in SD-IoT networks. Its unique approach, incorporating ML techniques and P4 language, enables accurate and timely detection and mitigation of various attack types, thereby enhancing the security of IoT networks. Future research directions may involve expanding the framework's capabilities to detect additional attack types in SDN-based IoT networks.

Author Contributions: Conceptualization, A.E.G.; methodology, A.E.G.; supervision, W.I.K. and E.R.M.; writing—original draft, A.E.G.; writing—review and editing, W.I.K. and E.R.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The data presented in this study are available upon reasonable request from the corresponding author.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Shah, H.; Shah, D.; Jadav, N.K.; Gupta, R.; Tanwar, S.; Alfarraj, O.; Tolba, A.; Raboaca, M.S.; Marina, V. Deep learning-based malicious smart contract and intrusion detection system for IoT environment. *Mathematics* **2023**, *11*, 418. [CrossRef]
2. Aldhyani, T.H.; Alkahtani, H. Cyber Security for Detecting Distributed Denial of Service Attacks in Agriculture 4.0: Deep Learning Model. *Mathematics* **2023**, *11*, 233. [CrossRef]
3. Omolara, A.E.; Alabdulatif, A.; Abiodun, O.I.; Alawida, M.; Alabdulatif, A.; Arshad, H. The internet of things security: A survey encompassing unexplored areas and new insights. *Comput. Secur.* **2022**, *112*, 102494. [CrossRef]
4. Katib, I.; Ragab, M. Blockchain-Assisted Hybrid Harris Hawks Optimization Based Deep DDoS Attack Detection in the IoT Environment. *Mathematics* **2023**, *11*, 1887. [CrossRef]
5. Mothukuri, V.; Khare, P.; Parizi, R.M.; Pouriyeh, S.; Dehghantanha, A.; Srivastava, G. Federated-Learning-Based Anomaly Detection for IoT Security Attacks. *IEEE Internet Things J.* **2021**, *9*, 2545–2554. [CrossRef]
6. Ahanger, T.A.; Tariq, U.; Dahan, F.; Chaudhry, S.A.; Malik, Y. Securing IoT Devices Running PureOS from Ransomware Attacks: Leveraging Hybrid Machine Learning Techniques. *Mathematics* **2023**, *11*, 2481. [CrossRef]
7. Touqeer, H.; Zaman, S.; Amin, R.; Hussain, M.; Al-Turjman, F.; Bilal, M. Smart home security: Challenges, issues and solutions at different IoT layers. *J. Supercomput.* **2021**, *77*, 14053–14089. [CrossRef]
8. Shieh, C.-S.; Nguyen, T.-T.; Horng, M.-F. Detection of Unknown DDoS Attack Using Convolutional Neural Networks Featuring Geometrical Metric. *Mathematics* **2023**, *11*, 2145. [CrossRef]
9. Ahmed, A.A.; Malebary, S.J.; Ali, W.; Alzahrani, A.A. A Provable Secure Cybersecurity Mechanism Based on Combination of Lightweight Cryptography and Authentication for Internet of Things. *Mathematics* **2023**, *11*, 220. [CrossRef]
10. Zhao, X.; Su, H.; Sun, Z. An Intrusion Detection System Based on Genetic Algorithm for Software-Defined Networks. *Mathematics* **2022**, *10*, 3941. [CrossRef]
11. Isyaku, B.; Bakar, K.B.A.; Ghaleb, F.A.; Al-Nahari, A. Dynamic Routing and Failure Recovery Approaches for Efficient Resource Utilization in OpenFlow-SDN: A Survey. *IEEE Access* **2022**, *10*, 121791–121815. [CrossRef]
12. Paolucci, F.; Cugini, F.; Castoldi, P.; Osiński, T. Enhancing 5G SDN/NFV edge with P4 data plane programmability. *IEEE Netw.* **2021**, *35*, 154–160. [CrossRef]
13. Zhang, X.; Cui, L.; Wei, K.; Tso, F.P.; Ji, Y.; Jia, W. A survey on stateful data plane in software defined networks. *Comput. Netw.* **2021**, *184*, 107597. [CrossRef]

14. Mahmood, A.; Casetti, C.; Chiasserini, C.F.; Giaccone, P.; Härri, J. Efficient caching through stateful SDN in named data networking. *Trans. Emerg. Telecommun. Technol.* **2018**, *29*, e3271. [CrossRef]
15. Kaur, S.; Kumar, K.; Aggarwal, N. A review on P4-Programmable data planes: Architecture, research efforts, and future directions. *Comput. Commun.* **2021**, *170*, 109–129.
16. Chen, X.; Wu, C.; Liu, X.; Huang, Q.; Zhang, D.; Zhou, H.; Yan, Q.G.; Khan, M.K. Empowering Network Security with Programmable Switches: A Comprehensive Survey. *IEEE Commun. Surv. Tutor.* **2023**. [CrossRef]
17. Ferrag, M.A.; Friha, O.; Hamouda, D.; Maglaras, L.; Janicke, H. Edge-IIoTset: A new comprehensive realistic cyber security dataset of IoT and IIoT applications for centralized and federated learning. *IEEE Access* **2022**, *10*, 40281–40306. [CrossRef]
18. Moustafa, N. A new distributed architecture for evaluating AI-based security systems at the edge: Network TON_IoT datasets. *Sustain. Cities Soc.* **2021**, *72*, 102994.
19. Al-Hawawreh, M.; Sitnikova, E.; Aboutorab, N. X-IIoTID: A connectivity-agnostic and device-agnostic intrusion data set for industrial Internet of Things. *IEEE Internet Things J.* **2021**, *9*, 3962–3977. [CrossRef]
20. Long, Z.; Jinsong, W. A hybrid method of entropy and SSAE-SVM based DDoS detection and mitigation mechanism in SDN. *Comput. Secur.* **2022**, *115*, 102604. [CrossRef]
21. Zhang, N.; Jaafar, F.; Malik, Y. Low-rate DoS attack detection using PSD based entropy and machine learning. In Proceedings of the 2019 6th IEEE International Conference on Cyber Security and Cloud Computing (CSCloud)/2019 5th IEEE International Conference on Edge Computing and Scalable Cloud (EdgeCom), Paris, France, 21–23 June 2019; pp. 59–62. [CrossRef]
22. Xie, J.; Yu, F.R.; Huang, T.; Xie, R.; Liu, J.; Wang, C.; Liu, Y. A survey of machine learning techniques applied to software defined networking (SDN): Research issues and challenges. *IEEE Commun. Surv. Tutor.* **2018**, *21*, 393–430. [CrossRef]
23. Hosny, K.M.; Gouda, A.E.-S.; Mohamed, E.R. New detection mechanism for distributed denial of service attacks in software defined networks. *Int. J. Sociotechnol. Knowl. Dev. (IJSKD)* **2020**, *12*, 1–30. [CrossRef]
24. Hosseinzadeh, M.; Sinopoli, B. Active attack detection and control in constrained cyber-physical systems under prevented actuation attack. In Proceedings of the 2021 American Control Conference (ACC), New Orleans, LA, USA, 25–28 May 2021; pp. 3242–3247.
25. Ravi, N.; Shalinie, S.M. Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture. *IEEE Internet Things J.* **2020**, *7*, 3559–3570. [CrossRef]
26. Yin, D.; Zhang, L.; Yang, K. A DDoS attack detection and mitigation with software-defined Internet of Things framework. *IEEE Access* **2018**, *6*, 24694–24705. [CrossRef]
27. Mihoub, A.; Fredj, O.B.; Cheikhrouhou, O.; Derhab, A.; Krichen, M. Denial of service attack detection and mitigation for internet of things using looking-back-enabled machine learning techniques. *Comput. Electr. Eng.* **2022**, *98*, 107716. [CrossRef]
28. Ullah, I.; Mahmoud, Q.H. Design and development of a deep learning-based model for anomaly detection in IoT networks. *IEEE Access* **2021**, *9*, 103906–103926. [CrossRef]
29. Gad, A.R.; Nashat, A.A.; Barkat, T.M. Intrusion detection system using machine learning for vehicular ad hoc networks based on ToN-IoT dataset. *IEEE Access* **2021**, *9*, 142206–142217. [CrossRef]
30. Yousuf, O.; Mir, R.N. DDoS attack detection in Internet of Things using recurrent neural network. *Comput. Electr. Eng.* **2022**, *101*, 108034. [CrossRef]
31. Tang, T.A.; Mhamdi, L.; McLernon, D.; Zaidi, S.A.R.; Ghogho, M. Deep learning approach for network intrusion detection in software defined networking. In Proceedings of the 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), Fez, Morocco, 26–29 October 2016; pp. 258–263. [CrossRef]
32. Khedr, W.I.; Gouda, A.E.; Mohamed, E.R. FMDADM: A Multi-Layer DDoS Attack Detection and Mitigation Framework Using Machine Learning for Stateful SDN-Based IoT Networks. *IEEE Access* **2023**, *11*, 28934–28954. [CrossRef]
33. Yungaicela-Naula, N.M.; Vargas-Rosales, C.; Pérez-Díaz, J.A.; Carrera, D.F. A flexible SDN-based framework for slow-rate DDoS attack mitigation by using deep reinforcement learning. *J. Netw. Comput. Appl.* **2022**, *205*, 103444. [CrossRef]
34. ElSayed, M.S.; Le-Khac, N.-A.; Albahar, M.A.; Jurcut, A. A novel hybrid model for intrusion detection systems in SDNs based on CNN and a new regularization technique. *J. Netw. Comput. Appl.* **2021**, *191*, 103160. [CrossRef]
35. de Assis, M.V.; Carvalho, L.F.; Rodrigues, J.J.; Lloret, J.; Proença, M.L., Jr. Near real-time security system applied to SDN environments in IoT networks using convolutional neural network. *Comput. Electr. Eng.* **2020**, *86*, 106738. [CrossRef]
36. Khan, M.A.; Iqbal, N.; Jamil, H.; Kim, D.-H. An optimized ensemble prediction model using AutoML based on soft voting classifier for network intrusion detection. *J. Netw. Comput. Appl.* **2023**, *212*, 103560. [CrossRef]
37. Simsek, G.; Bostan, H.; Sarica, A.K.; Sarikaya, E.; Keles, A.; Angin, P.; Alemdar, H.; Onur, E. Dropppp: A P4 approach to mitigating dos attacks in SDN. In Proceedings of the Information Security Applications: 20th International Conference, WISA 2019, Jeju Island, Republic of Korea, 21–24 August 2019; Revised Selected Papers 20. pp. 55–66.
38. Febro, A.; Xiao, H.; Spring, J. Distributed SIP DDoS defense with P4. In Proceedings of the 2019 IEEE Wireless Communications and Networking Conference (WCNC), Marrakesh, Morocco, 15–18 April 2019; pp. 1–8.
39. Musumeci, F.; Fidanci, A.C.; Paolucci, F.; Cugini, F.; Tornatore, M. Machine-learning-enabled DDoS attacks detection in P4 programmable networks. *J. Netw. Syst. Manag.* **2022**, *30*, 21. [CrossRef]
40. Hong, S.; Xu, L.; Wang, H.; Gu, G. Poisoning network visibility in software-defined networks: New attacks and countermeasures. In Proceedings of the Ndss, San Diego, CA, USA, 8–11 February 2015; pp. 8–11.

41. Sebbar, A.; Zkik, K.; Boulmalf, M.; El Kettani, M.D.E.-C. New context-based node acceptance CBNA framework for MitM detection in SDN Architecture. *Procedia Comput. Sci.* **2019**, *160*, 825–830. [CrossRef]
42. Zhang, K.; Qiu, X. CMD: A convincing mechanism for MITM detection in SDN. In Proceedings of the 2018 IEEE International Conference on Consumer Electronics (ICCE), Las Vegas, NV, USA, 12–14 January 2018; pp. 1–6.
43. Deng, S.; Gao, X.; Lu, Z.; Gao, X. Packet injection attack and its defense in software-defined networks. *IEEE Trans. Inf. Forensics Secur.* **2017**, *13*, 695–705. [CrossRef]
44. Kaur, J. Wired LAN and wireless LAN attack detection using signature based and machine learning tools. In *Networking Communication and Data Knowledge Engineering: Volume 1*; Springer Nature: Singapore, 2018; pp. 15–24.
45. Ma, H.; Ding, H.; Yang, Y.; Mi, Z.; Yang, J.Y.; Xiong, Z. Bayes-based ARP attack detection algorithm for cloud centers. *Tsinghua Sci. Technol.* **2016**, *21*, 17–28. [CrossRef]
46. Ahuja, N.; Singal, G.; Mukhopadhyay, D.; Nehra, A. Ascertain the efficient machine learning approach to detect different ARP attacks. *Comput. Electr. Eng.* **2022**, *99*, 107757. [CrossRef]
47. Alanazi, M.; Aljuhani, A. Anomaly detection for internet of things cyberattacks. *Comput. Mater. Contin.* **2022**, *72*, 261–279. [CrossRef]
48. Das, S.; Mahfouz, A.M.; Venugopal, D.; Shiva, S. DDoS intrusion detection through machine learning ensemble. In Proceedings of the 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), Sofia, Bulgaria, 22–26 July 2019; pp. 471–477. [CrossRef]
49. Tan, L.; Pan, Y.; Wu, J.; Zhou, J.; Jiang, H.; Deng, Y. A new framework for DDoS attack detection and defense in SDN environment. *IEEE Access* **2020**, *8*, 161908–161919. [CrossRef]
50. Ilango, H.S.; Ma, M.; Su, R. A FeedForward–Convolutional Neural Network to Detect Low-Rate DoS in IoT. *Eng. Appl. Artif. Intell.* **2022**, *114*, 105059. [CrossRef]