MWD-CFM: Detection and Mitigation of DDoS Attack Against SDN Flow Tables

Dan Tang, Chenguang Zuo, Xinmeng Li, Siyuan Wang, Wei Liang, Keqin Li, *Fellow, IEEE*, and Jiliang Zhang, *Senior Member, IEEE*

***Abstract*—The decoupling of SDN control plane and data plane allows control to be carried out independently, enhancing the programmability and manageability of the whole network. However, though this new architecture brings convenience for control, it also sets a pool for attacks. Due to the expensive and resource-consumption characteristics, SDN switches usually have a limited ternary content addressable memory to cache the flow rules in the data plane. Attackers can launch a space-consuming attack to maliciously preempt the flow table, forcing switches to reduce the quality of service. Among them, DDoS attacks should be taken seriously, especially the low-rate ones that have a lower attack rate with stronger concealment. In this paper, an architecture named MWD-CFM is proposed to protect against low-rate DDoS attacks in the switch flow tables. Multi-windows work collaboratively to improve detection performance. The feature of flow rules is corrected to do better classification, improving the effectiveness of attack traffic removal. Experiments based on real network topology and datasets are conducted to verify the deployment of MWD-CFM. Results prove that our architecture can greatly reduce the survival time of attack flows and ensure the availability of flow tables for legitimate services.**

***Index Terms*—SDN, low-rate DDoS attack, data plane, Fisher score, multi-window detection, corrected feature mitigation.**

I. Introduction

As A new architecture based on centralized control, Software Defined Networking (SDN) has been favored by not only academia but also industry since it was proposed in 2006 [1]. By separating the control plane from the data plane, control may be implemented as software separately, improving the network's overall programmability and manageability [2]. For networks, quality of service can always be worth considering, especially as distributed architectures become increasingly complex. After SDN is continuously improved, it has been extensively utilized in cloud [3], [4], VANET [5], [6], and data centers for better network design.

However, SDN not only brings convenience but also provides opportunities for new threats. Although it is a new design for decoupling control, it still incorporates the attributes of traditional networks. The application plane runs the applications from different developers, which may be vulnerabilities in themselves [7]. Attackers can monitor control channels and flood them, causing interaction disruption [8]. DDoS attacks against traditional networks can also be launched in SDN. Both three planes and two communication channels can be targets of attacks. For the control plane, controllers can be consumed by saturation attacks with massive new useless flows. The data plane is the same subject to various attacks, especially since it is still based on various basic devices and traditional network protocols, including routers and switches [9].

Among these attacks, DDoS attacks receive sufficient attention, given that they consume various resources while reducing the quality of service [10]. Especially in the data plane, SDN switches usually are limited by the expensive Ternary Content Addressable Memory (TCAM). Flooding-based DDoS attacks can not only cause congestion or even paralysis of links between switches [11], but also maliciously occupy the flow table and cause overflow. Even more, the low-rate ones have a lower average rate and higher concealment in the early stage [12], which makes the detection more challenging when detecting them. However, some existing detection methods are mainly aimed at high-rate DDoS attacks, which may not perform well in the low-rate types this paper focused on. In the process of detecting low-rate DDoS attacks and deleting malicious flow rules, we face the following technical challenges:

- The behavior of low-rate DDoS attacks is covert, and its features are similar to normal traffic.
- The network traffic pattern is prone to change, leading to a decrease in the performance of the detection model.
- How to minimize the impact on benign traffic while removing malicious flow rules?

Received 19 January 2024; revised 22 October 2024 and 28 April 2025; accepted 7 December 2025; approved by IEEE TRANSACTIONS ON NETWORKING Editor N. Zhang. Date of publication 17 March 2026; date of current version 25 March 2026. This work was supported in part by the National Natural Science Foundation of China under Grant 62472153, in part by the Open Research Fund of State Key Laboratory of Internet Architecture under Grant HLW2025MS18, and in part by Hunan Provincial Natural Science Foundation of China under Grant 2025JJ50350. *(Corresponding author: Jiliang Zhang.)*

Dan Tang is with the College of Cyber Science and Technology, Hunan University (HNU), Changsha 410082, China, and also with the State Key Laboratory of Internet Architecture, Tsinghua University, Beijing 100084, China (e-mail: Dtang@hnu.edu.cn).

Chenguang Zuo, Xinmeng Li, and Siyuan Wang are with the College of Cyber Science and Technology, Hunan University (HNU), Changsha 410082, China (e-mail: chenguangzuo@hnu.edu.cn; lixinmeng@hnu.edu.cn; siywang@hnu.edu.cn).

Wei Liang is with the School of Computer Science and Engineering, Hunan University of Science and Technology (HNUST), Xiangtan 411199, China (e-mail: wliang@hnust.edu.cn).

Keqin Li is with Department of Computer Science, State University of New York, New Paltz, NY 12561 USA (e-mail: lik@newpaltz.edu).

Jiliang Zhang is with the College of Semiconductors (College of Integrated Circuits), Hunan University (HNU), Changsha 410082, China (e-mail: zhangjiliang@hnu.edu.cn).

Digital Object Identifier 10.1109/TON.2026.3674973

2998-4157 © 2026 IEEE. All rights reserved, including rights for text and data mining, and training of artificial intelligence and similar technologies. Personal use is permitted, but republication/redistribution requires IEEE permission.
See https://www.ieee.org/publications/rights/index.html for more information.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

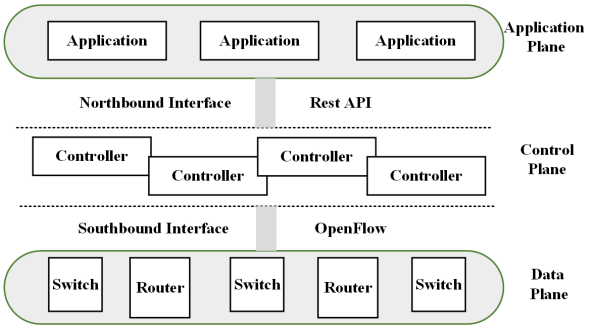


Fig. 1. Brief view of SDN architecture.

Motivated by this, an architecture named MWD-CFM is designed in this work to protect against the low-rate DDoS attacks in the switch flow tables of the SDN data plane. This architecture is deployed as an extension of the controller to monitor and protect switches. Multi-window detection (MWD) is introduced to do feature collection and collaborative attack detection while corrected feature mitigation (CFM) based on flow rules conducts the deletion task to keep available table space. MWD-CFM is verified on a real network topology and under real traffic datasets. This work can make the following contributions:

- A detection and mitigation architecture named MWD-CFM is presented for low-rate DDoS attacks against flow tables in SDN.
- Multiple windows are set to perform collaborative detection of attacks simultaneously to cope with the impact of changes in network traffic patterns.
- The feature importance and sensitivity-based thresholds are combined to improve the recognition ability of LDDoS attacks when detecting.
- The attributes of flow rules are corrected for classification and recognition, improving the ability to remove attack traffic and reducing misjudgments of benign traffic.
- Experiments are designed to confirm the efficacy of MWD-CFM and prove the rationality and viability.

Section II is background about SDN, OpenFlow, and low-rate DDoS attacks against flow tables. Section III provides a concise summary of related work to state and summarize state-of-art solutions. Section IV presented the detailed algorithm and design of MWD-CFM. In Section V, we conduct the simulation and evaluation. The contribution and conclusion will be discussed in Section VI.

II. Background

A. Software Defined Networking

The separation of control and data processing is what makes SDN unique. It allows control to be performed separately in the form of software, rather than relying on traditional hardware devices for integrated operations. Fig. 1 shows a sketch. A typical SDN has three planes with two interfaces. Different applications can be installed on the application plane without considering the specific underlying logic and design, as SDN provides convenient APIs. The control plane is commonly regarded as the decision maker of SDN. Controllers usually lie in this plane to obtain global management and handle the information interaction of the data plane. The northbound channel links the applications and the controllers, while the southbound channel is a bridge between the controllers and the switch devices.

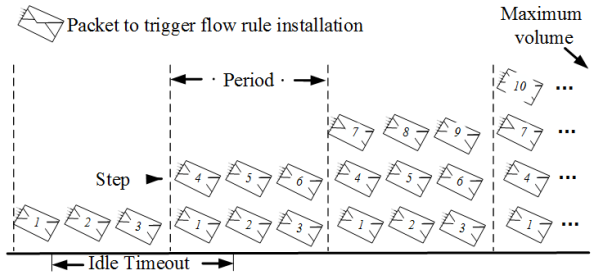


Fig. 2. Low-rate DDoS attack against flow tables in SDN.

B. OpenFlow

One of the most popular protocols in the southbound interface for defining event handling, flow processing, and message exchange is OpenFlow.[1] A matching rule has been created in the flow table for an old flow that already matches, and it will be sent straight to the next hop. In the event that a fresh flow arrives that hasn't been handled by this route, the switch will alert the controller by sending a *packet_in* message. And then a corresponding rule will be generated by the controller to make a decision on it. The controller responds with an action to install rules in the flow table considering there will be a next packet to arrive. Only when the flow table is full will a new packet cause the $Table\_full$ to be triggered. In order to provide enough room, the controller must then choose whether to reject this stream or remove the current ones. As a result, an attacker can generate a lot of mismatched packets, which will continuously cause *packet_in* and $Table\_full$ to be triggered. This will leave the flow table with too little room to accommodate legal flows.

C. Threat Model

For the SDN switches, there can be limited TCAM spaces as they are usually expensive and with high consumption. Low-rate DDoS attacks against flow tables in SDN do take advantage of this weakness. The purpose of an attacker is not to create flooding to congest links but to maliciously occupy limited flow table space. As shown in Fig. 2, low-rate DDoS attacks against flow tables are defined by step, period, and maximum volume. An attacker accumulates new packets with a length of step to install new flow rules each time, while including the old ones that have already been installed to ensure that they are not removed within the idle timeout. At last, the attacker sends maximum packets continuously to preempt the flow table constantly. IP, port, and protocol types can all be transformed to create meaningless fake packets for injection.

Our objective is to create a highly accurate and cost-effective system to detect DDoS attacks. It checks the SDN switch

[1]Openflow version 1.3. https://opennetworking.org/wp-content/uploads/2013/04/openflow-spec-v1.3.1.pdf

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

regularly through the controller to ensure that the system does not interfere with traffic forwarding. After identifying the attack, it can work in conjunction with existing low-rate DDoS attack defenses to eliminate detected malicious flow rules. Our threat model has three key participants: attackers, switches, and controllers.

Attackers. Attackers can obtain the flow table capacity and idle timeout parameters of the switch, create a large number of mismatched data packets, and send them to the switch at the bottleneck link. At this point, both the data layer switch and the control layer controller must prioritize handling these events. This prioritization can adversely affect the processing and forwarding of legitimate data packets. Simultaneously, low-rate DDoS attackers deploy the attacking hosts in a distributed manner, resulting in attack packets originating from various addresses, making it more difficult to locate the attackers. If attackers can obtain the features and thresholds used by defense strategies, they can also consume controller resources by frequently triggering mitigation modules.

Switches. Each switch is independently monitored and managed by a controller, and different host devices are set up under each switch. They are often limited by expensive TCAM space and easily occupied by low-rate DDoS attacks, resulting in the inability to provide services for normal traffic. Frequent flow table updates may increase packet processing latency and affect normal traffic forwarding performance. The switches are compromised because attackers can overload the flow table.

Controllers. Controllers implement the installation of new rules and the forwarding of new flows by issuing flow table rules to the switches. When an attacker sends a large number of mismatched packets, the controller needs to generate flow table entries for each unmatched flow. The attack flow causes congestion in the control link, leading to processing delays. The large number of packet-in messages will occupy the bandwidth of the control link and interfere with the normal communication between the controller and the switch. Because we deploy the detection response method in the control layer, we consider the controller to be trusted. We assume that there are no internal attackers who directly maliciously manipulate the flow table or controllers.

III. Related Work

A. DDoS Attacks Targeting SDN Architecture

DDoS attacks targeting SDN architecture can be launched in all three planes including the control channels.

For the application plane, attackers launch DDoS attacks against various applications and some common protocols. Slow HTTP read attacks can be conducted by using incomplete HTTP requests to reduce the performance of HTTP-based programs [13]. For the control plane, saturation attacks are a typical threat to make it refuse service [14]. Attackers trigger malicious interactive messages to overwhelm the controller with events, exploiting limited resources on the controller such as buffers [15]. In addition, parameters could be inferred in advance, and carefully designed attacks were constructed through reflection to launch DDoS attacks against controllers [16]. For the data plane, it is under threat not only of traditional attacks but also of new attacks. DRDoS attack against the data plane [17] utilizes the reflection characteristics of programmable data planes, achieving higher attack efficiency and stronger concealment. DoS attacks against the bottleneck link cut down the quality of service [18] while overflow attacks cause the limited flow table overload [19]. The channels especially the southbound one may be flooded by DDoS attacks to destroy the timely response [8]. Low-rate DDoS attacks have a lower average rate and less obvious characteristics [20], which makes it harder to defend in time.

B. SDN DDoS Attack Detection

The techniques for identifying DDoS attacks fall into three primary categories: machine learning, deep learning, and entropy-based. They are usually combined with IP-based, packet-based, and byte-based features.

In entropy-based detection, it is crucial to consider the distribution and variation of traffic. Kalkan et al. [21] selected the TCP attributes and destination IPs to obtain the entropy value in their work. Meanwhile, they used dynamic thresholds instead of fixed thresholds to improve the adaptive ability. However, this method only worked on the TCP SYN DDoS attacks. Renyi Entropy was another method to check the traffic distribution [22]. In another Renyi-based work, Ahalawat et al. [23] recorded all the IP data in a hash table and computed the corresponding ID of flow to calculate the Renyi entropy. Mishra et al. [24] adopted Shannon entropy to conduct DDoS attack detection in SDN controllers. They set the experience thresholds to assist in improving detection efficiency. Although these methods are typically lightweight and fast, they require a reliable set of data, which can be difficult to adapt to complex and volatile environments.

For deep learning detection, thorough analysis of input data and finding internal correlations are often more important. Ahuja et al. [25] collected the packets, bytes, duration, and *packet_in* of the flow table to do DDoS attack detection. The port traffic was considered and then the ANN was applied. Tang et al. [26] tested their approach on NSL-KDD datasets which was based on RNN. Fouladi et al. collected the source IP address of all traffic, counted their total number, and set a hash table for destination IPs [27]. Then they used an auto-encoder neural network for decision. The CNN and gated recurrent unit which used sailfish optimizer to improve parameters were verified by CICLDOS2017 and CIADA datasets [28]. However, these methods depend more on the training datasets which may lose performance when applied under other circumstances.

For machine learning detection, extracting the traffic's information helps improve classification accuracy. Chouhan et al. [29] input the byte, packet, and IP into multi classifiers including SVM, RF, KNN, and four other machine-learning algorithms. It is worth learning that they introduced the concept of combinatorial pairs. Similar to this work, Aslam et al. [30] took the same approach, combining multiple classifiers into an adaptive model with different weights. DDoS attack detection could be performed by clustering the number of packets and connections [31]. Not

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

only continuous flows were considered, but also discontinuous ones. To detect malicious flows of congestion-related assaults, Dai et al. [32] use an in-network machine learning model that makes use of queue and packet properties. In addition, multiple classifier combination models often require more training, which may result in more time and resource overhead.

C. Mitigation Concepts Against DDoS Attacks in SDN

DDoS attacks usually cause continuous malicious occupancy of SDN switches, making it insufficient to provide normal services for legitimate flows. Thus, maintaining the available flow table for the network traffic forwarding is particularly important to keep the quality of service. To mitigate DDoS attacks in SDN, rule deletion and migration are the universal technical means.

A lightweight extension module named DoSGuard [33] was proposed, which quickly discovered anomalies based on *packet_in* messages. After detecting an attack, it would lock the malicious host and remove the malicious flows while SIFT [34] deleted rules according to the setting of a probability. However, DoSGuard built a switch host relationship to conduct analysis that may increase the load as the complexity of the network increases. Since the flows in the SDN flow tables all have a priority attribute, this specifies the order in which they are matched. Batool et al. [35] mitigated DDoS attacks by downgrading the priority of the flow and migrating to other ports. In the LLDM scheme [36], a two-stage mitigation method was adopted which migrated suspicious packets to the agent first and then assigned priority based on their legitimacy. DDoS attacks may also be mitigated by rate limiting [37], which stops *packet_in* from entering. The eviction [38] was another basic default in OpenFlow. However, these methods affect legitimate flows and may be useless to prevent the continued inflow of attack traffic.

Although there is currently much work to protect against DDoS attacks in SDN, more research is needed to address low-rate DDoS attacks against flow tables. For existing work on flow tables, the goals of effectiveness and interpretability need further improvement which emphasizes the ability to sense especially low-rate DDoS attacks and the correctness of classifying malicious flow rules. Thus, we propose the MWD-CFM architecture to protect limited flow tables, keeping the quality of service for legal flow rule installation and traffic forwarding.

IV. MWD-CFM Design

In this section, how architecture called MWD-CFM detects and mitigates low-rate DDoS attacks against flow tables in SDN is introduced in detail.

A. Feature Analysis

One of the characteristics of DDoS attacks against SDN flow tables is to maliciously occupy the limited TCAM space by creating an extensive number of mismatched packets, making it impossible for the switch to provide normal forwarding processing services for legitimate flows [39]. The low-rate ones have a much lower average rate, and they usually expand

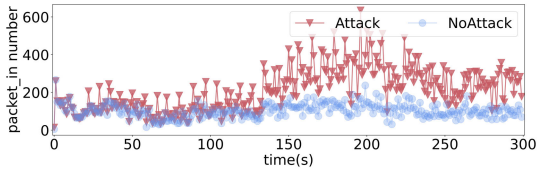


Fig. 3. Attack and NoAttack pakcet_in messages.

the maliciously occupied capacity by gradually accumulating the idle timeout of the flow table and continuously triggering old flows to ensure that they are not deleted while installing new ones. Before it reaches the maximum attack capacity, new packets are inevitably sent to trigger the generation of *packet_in* messages. Fig. 3 illustrates the variation in the quantity of *packet_in* messages before and after the attack is launched. During the attack initiation phase, which spans from 0 to 150 seconds, the periodic injection of new streams in low-rate DDoS attacks results in a corresponding periodic increase in the quantity of *packet_in* messages. Once the attacker maintains a constant attack rate, the quantity of *packet_in* messages exhibits more pronounced characteristics compared to a switch that is not under attack. Consequently, features derived from the *packet_in* data packets will be extracted.

Low-rate DDoS attacks share the common traits with traditional DDoS attacks, where attackers typically flood and occupy target resources that have bottleneck limitations by utilizing a series of hosts [40]. However, because these hosts are often decentralized or even employ address spoofing, accurately tracing and locating the attacker's IP address becomes challenging. For the network, while it may not be feasible to effectively locate attackers, the injection of malicious packets from these manipulated hosts will inevitably make the IP distribution abnormal [41]. Analyzing data packet IP addresses to identify anomalies is now one of the most prevalent methods for detecting DDoS attacks. Therefore, IP addresses will also be regarded as a significant feature, and the concept of entropy, which measures chaos, will be applied to the analysis of IP addresses.

The global management function of the controller enables it to monitor the status of the switch and query the necessary information. All flows injected into the SDN data layer must originate from a port monitored by the controller, be matched, and then forwarded to a specified port before exiting. Low-rate DDoS attacks exhibit specific characteristics, including periodic bursts in their attack behavior [42]. Consequently, with the onset of the attack flow, the incoming ports of the switch will also display noticeable changes. Fig. 4 illustrates the incoming port load of the monitored switches before and after the attack. Overall, following the initiation of the low-rate DDoS attack, the incoming port load exhibits fluctuations and distinct differentiation. We will use the traffic information flowing into the port as the extraction object, calculating its mean, standard deviation, and coefficient of variation, and further analyze the abnormal distribution of port traffic.

Fig. 5 shows the variation in the quantity of flow entry bytes relative to the duration of the flow entry, in addition to the link between the quantity of flow entry packets and bytes. In Fig. 5a, it is evident that the attack packets are

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

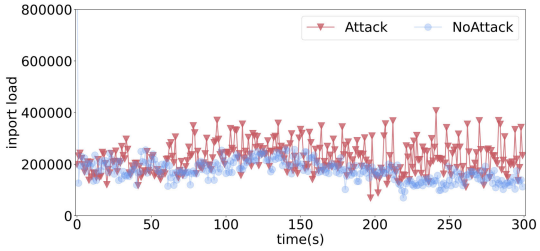
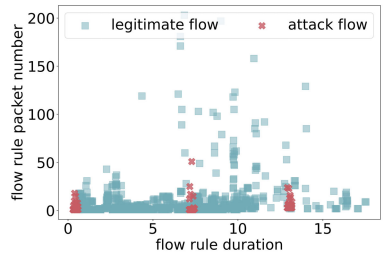
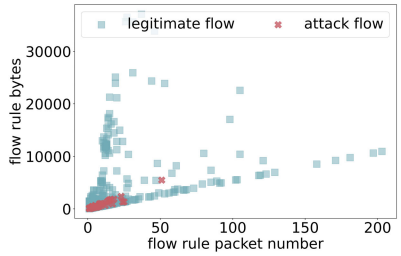


Fig. 4. Attack and NoAttack inport_load.



(a) flow table item number and duration



(b) flow table item number and bytes

Fig. 5. Flow table item number, duration, and bytes.

injected in a centralized manner, with the flow rules generated by the attack flow being aggregated in the flow table at a single attack cycle. As the attack cycle extends, the newly installed entries in the attack flow table at different times also exhibit intervals corresponding to the cycle length. Fig. 5b depicts the relative change between the total number of bytes and packets in the flow table over a monitoring period. Due to their persistent presence in the flow table, both metrics demonstrate an accumulative growth characteristic. Consequently, extracting only the duration [33], packet count [27], and byte count [28] of flow table entries for classification purposes may lead to misidentification. This section will assess the average performance of each flow entry by calculating the average packet byte count and average packet duration. This section will measure the average performance of each flow entry by calculating the average packet byte count and average packet duration. In addition, based on the typically single nature of attack packets and the parsing of events and data in the OpenFlow 1.3 protocol, the total_len field of each packet at the time scale will be considered in the feature space in this section. In Table I, we list the abbreviations with their descriptions that will be used later.

B. Detection

After the feature collection by polling monitor, the detection module conducts further processes. The detection module

TABLE I
ABBREVIATIONS AND DESCRIPTION

Abbreviation	Description
IOR	Inport occupation rate of a SDN switch.
$EIOR$	Entropy of IOR.
$FIPT$	Feature importance obtained by Fisher Score.
TH	Threshold to judge feature's anomalies.
SC	Sensitivity coefficient that works with TH.
ABP	Average bytes per packet.
ADP	Average duration per packet.
$CVADP$	Corrected ADP by coefficient of variation.

consists of multi windows that are manifested by setting different lengths, acting as a cooperative task. The final result of whether an attack has occurred is determined by their joint analysis.

The inport traffic, *packet_in* messages, IP, and *total_len* are all collected for each window. Since entropy performs well in the analysis of data distribution [22], [24], it is adopted in the inport and IP analysis in this paper. The EIOR is calculated as Eqs. (1) and (2). It means the entropy of the switch's inport occupation rate. Among them, n means the number of the switch inport. i is in range of 0 to n. $port_i$ is the bytes of the i port during a collection.

$$IOR_i = \frac{port_i}{\sum_{i=1}^{n} port_i} \tag{1}$$

$$EIOR = -1 \times \sum_{i=1}^{n} IOR_i \times \log_2 IOR_i \tag{2}$$

Different feature dimensions are usually in different range intervals. If we simply set weights on the features to obtain the overall score, this will make the individual features be magnified and the other features will be ignored. Therefore, in this paper, the features are processed based on thresholds and combined with the Fisher score algorithm [43] to enhance usability. The core idea of this algorithm is to narrow the intra-class distance while keeping the inter-class distinct enough for the features to be identified. We define that there are c classes and n sets of samples. Each set of samples contains m features and each class has n_i samples. $f^{(k)}$ denotes the k_{th} feature of sample x, $F_i^{(k)}$ is the mean of the sample's k_{th} feature in class i, and $F^{(k)}$ stands for the mean of all classes' k_{th} feature. Define the k_{th} feature's interclass variance as $S_{out}^{(k)}$, which is calculated by Eq. (3). Define the k_{th} feature's intra-class variance as $S_{in}^{(k)}$ in Eq. (4). The importance score of the k_{th} feature is $FIPT(k)$, as shown in Eq. (5). k is in $[1, m]$ and i is in $[1, c]$.

$$S_{out}^{(k)} = \sum_{i=1}^{c} \frac{n_i}{n} \left(F_i^{(k)} - F^{(k)}\right)^2 \tag{3}$$

$$S_{in}^{(k)} = \frac{1}{n} \sum_{i=1}^{c} \sum_{x \epsilon c_i} \left(f^{(k)} - F_i^{(k)}\right)^2 \tag{4}$$

$$FIPT_k = \frac{S_{out}^{(k)}}{S_{in}^{(k)}} \tag{5}$$

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

The importance of features is obtained by the Fisher score, and features within the window are compared based on thresholds and sensitivity coefficients. The combined scores are added up by threshold comparison to narrow down the total score range of features. Finally, the total score of multiple windows is obtained to determine whether a low-rate DDoS attack has occurred. The detailed process is presented by Algorithm 1.

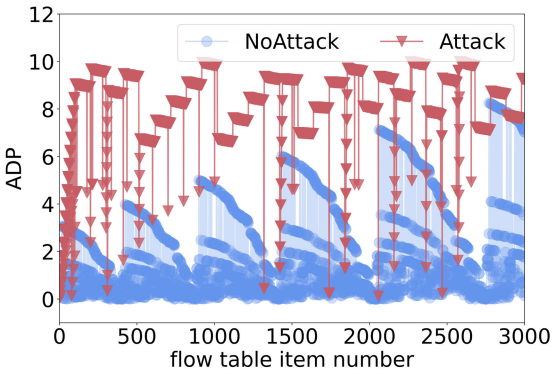
Algorithm 1 Conduct the MWD

Input: $packet\_in$, MW,$weight$, $portdata$, TH, SC, $FIPT$
Output: $PollingScore$
Function DoMWD($packet\_in$, MW,$weight$, $portdata$ TH, SC, $FIPT$):
  Initialize $PollingScore$;
  for $w \in$ MW **do**
    $score_w \leftarrow GetWS(TH, SC, FIPT, w)$;
    $PollingScore \leftarrow PollingScore + score_w \times weight_w$;
  return $PollingScore$
Function GetWS(TH, SC, $FIPT$, w):
  Initialize $score$ **for** $f \in$ *GetWF(packet_in,portdata, w)* **do**
    if $f \geq TH_f \times SC$ **then**
      $score \leftarrow score + FIPT_f$;
  return $score$
Function GetWF($packet\_in$,$portdata$, w):
  Initialize $pktnum$, $tlen$, $pktinIP$, $eior$,$f1$,$f2$,$f3$,$f4$,$f5$,$f6$,$f7$,$f8$,$f9$;
  while *Polling* **do**
    Add $packet\_in.count$ to $pktnum$;
    Add $packet\_in.IP$ to $pktinIP$;
    Add $packet\_in.totallen$ to $tlen$;
    Add $portdata$ to $eior$;
    if $timecount == w$ **then**
      $f1 \leftarrow eior.mean()$;
      $f2 \leftarrow eior.std()$;
      $f3 \leftarrow eior.cv()$;
      $f4 \leftarrow tlen.mean()$;
      $f5 \leftarrow tlen.std()$;
      $f6 \leftarrow tlen.cv()$;
      $f7 \leftarrow pktnum.mean()$;
      $f8 \leftarrow pktnum.std()$;
      $f9 \leftarrow pktinIP.entropy()$;
  return $f1$,$f2$,$f3$,$f4$,$f5$,$f6$,$f7$,$f8$,$f9$

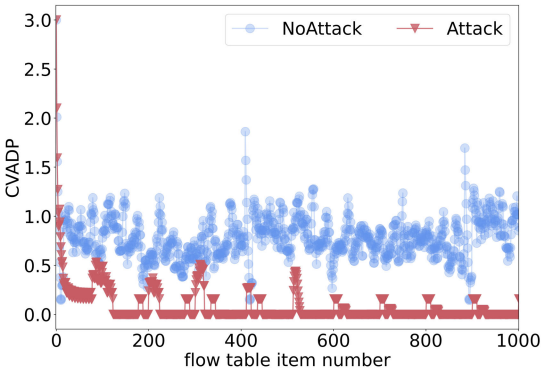
C. Mitigation

After collaborative multi-window detection by the detection module, if it is determined that a low-rate DDoS attack has occurred, the mitigation module will begin to do classification and deletion.

There are multi-attribute fields for each flow rule in the flow table. In this paper, we extract the duration [33], packets [27], and bytes [28] for analysis. There are usually a lot of pointless flow rules in the limited TCAM space, based on the analysis of low-rate DDoS attacks on the flow tables. The attacker ensures that they can persist for a long time, which finally manifests in unusually long durations of the attack flows [12]. We define the *ADP* as one of the input features for the identification task, which means the average duration per packet. In addition, since attackers often launch a single form and construct simple nonsensical packets to achieve the goal of crowding the flow table space, we define *ABP* as the other feature to stand for the average bytes per packet. For complex real networks, packets exhibit diversity in both size and form, which differ from those of attack ones. *ADP* and *ABP* are calculated as Eqs. (6) and (7).



(a) Attack and NoAttack ADP



(b) Attack and NoAttack CVADP

Fig. 6. The ADP and CVADP of flow rules.

$$ABP = \frac{bytes}{packets} \tag{6}$$

$$ADP = \frac{duration}{packets} \tag{7}$$

However, it's found that the distribution of *ADP*s is mixed due to the time as well as frequency differences when obtaining the flow tables, as shown in Fig. 6. The initiator of a low-rate DDoS attack needs to repeatedly trigger old flows and install a certain number of new ones within an idle timeout to achieve the attack. And this usually means that the attack flows will not arrive only alone, but accompanied by companions. Given this, this paper corrects the *ADP* features by using their coefficients of variation instead of the original *ADP* input to

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

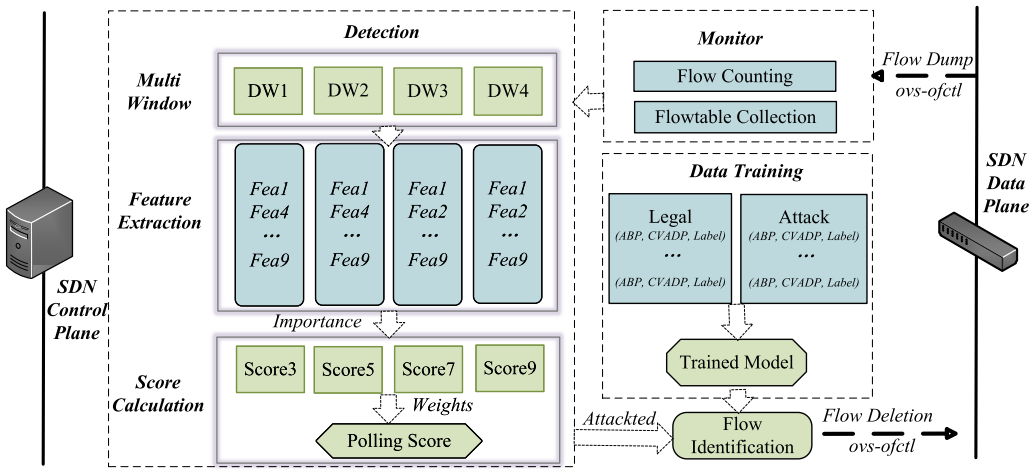


Fig. 7. Overall framework of MWD-CFM.

the classifier, named *CVADP*. It is computed by defining a sequence that contains both the flow rule to be classified and its neighbors. Eq. (8) shows the *ADP*'s coefficient of variation. adp is the mean of an *ADP* sequence. The corrected *ADP*s show a more concentrated distribution both for the attack flows and the legitimate ones. Also, the attack ones are more distinguishable from the normal states.

$$CVADP = \frac{\sum_{i=1}^{n}(ADP_i - adp)^2}{adp} \tag{8}$$

The supervised learning-based MLP is chosen as the classifier serving the recognition task. As a fully connected neural network, MLP is simple to implement and fast to apply [44]. In addition, if additional training samples need to be added, the training can be repeated by using both old and new training samples. The calibrated *ADP*s as well as the *ABP*s are fed as input features to the trained MLP classifier. As demonstrated by Algorithm 2, a flow that has an attack label applied to it will be deleted from the flow table.

Algorithm 2 Flow Deletion With CVADP

Input: $flowtable$, $PollingScore$, MLP
Initialize ABP, $ADPs$,$CVADP$;
if $PollingScore$ *is Attack* **then**
  for $flow \in$ flowtable **do**
    $ABP \leftarrow \frac{flow.byte}{flow.packet}$;
    $ADP \leftarrow \frac{flow.interval}{flow.packet}$;
    $ADPs.pop()$;
    Add ADP to $ADPs$;
    $CVADP \leftarrow ADPs.cv()$;
    if $MLP(ABP, CVADP)$ *is Illegal* **then**
      ovs-ofctl delete $flow$;

D. Overall Framework

The proposed architecture named MWD-CFM for SDN low-rate DDoS attacks is shown in Fig. 7.

MWD-CFM is installed on the control layer, where the controller polls the managed switch for flow entry counting and flow rule information. Subsequently, each detection window begins to be responsible for its own collection, processing, analysis, and detection tasks. Each window extracts nine variables in total: the EIOR mean, standard deviation, and coefficient of variation, *packet_in*, IP, and *total_len*. After obtaining the importance of each feature given by the Fisher score, the values for each window will be summarized with the weights. The polling score obtained will be used to judge attack events. Before the MLP classifier starts working, it will first be supervised and trained by a labeled dataset of legal and attack types. After detecting an attack, the MLP identifies and classifies each flow entry in the flow table. If the *ABP* and *CVADP* of a flow rule are labeled as an attack by MLP, the flow rule will be removed. In order to protect against low-rate DDoS attacks, the MWD-CFM architecture continuously monitors the SDN switches on the controller and makes sure that the SDN switch's flow table is always available.

V. Experiment

In this section, we validate the MWD-CFM architecture proposed in our paper. We conduct the simulation on a real network topology. The effectiveness of the detection and mitigation module demonstrates the efficacy of the proposed MWD-CFM.

A. Setup

Experimental environment. We simulated the MWD-CFM architecture of this paper based on a real network topology dataset named topology zoo.[2] This dataset covers 261 topologies of countries and regions with real critical node switches. Among them, we used a European topology named GlobalNetRu, a 2011 Backbone. Fig. 8 shows the simulation

[2]The Internet Topology Zoo. http://www.topology-zoo.org/dataset.html

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

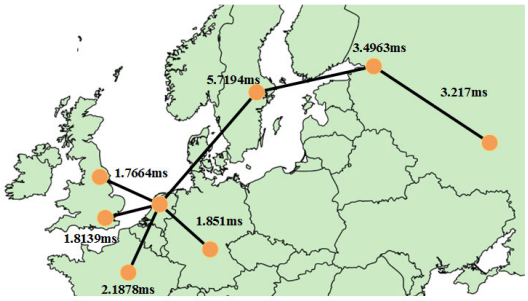


Fig. 8. The real network topology named GlobalNetRu.

topology, consisting of 8 core switch nodes. Each switch is independently monitored and managed by a controller, and different host devices are set under each switch. Referring to existing work [34], [45], [46], we set the flow table space of the switches to 2000 and the idle timeout to a default of 10s. The bandwidth and delay of the links between the switches were decided according to their latitude and longitude references. A Github project[3] was used to convert the GlobalNetRu topology into a python script for experimentation in Ubuntu. The controller was Ryu[4] and Mininet was selected as the emulator. A Layer-4-based switch message handler was adopted to run the default management and forwarding of the flows. The emulation hardware environment was Intel(R) Xeon(R) CPU E5-2680 v4 @2.40GHz. The MWD-CFM overall design was based on the OpenFlow protocol.[5]

Background traffic. To simulate real network traffic, IMC dataset[6] was chosen as background traffic to simulate message forwarding between switches. It is recorded from a data center. This dataset was cut to the appropriate length and replayed at different rates by different hosts under different core switches by tcpreplay tool. The load of each packet was populated by tcprewrite, and the IP address was also rewritten so that it could be forwarded efficiently in the topology. The low-rate DDoS attacks against the SDN switches were launched by python scripts based on scapy. The packet size satisfied a Poisson distribution. The attacker's IP address was a random IP to satisfy the denial of service attack.

Dataset acquisition. Since the proposed MWD-CFM architecture includes feature selection, parameter setting, and model training, we need to obtain the training and testing dataset first. A 3882 seconds-long overall extraction of flow table information was used for feature selection. 45,562 flow rules were used to train the classifier and 44,109 flow rules were for the performance test of them.

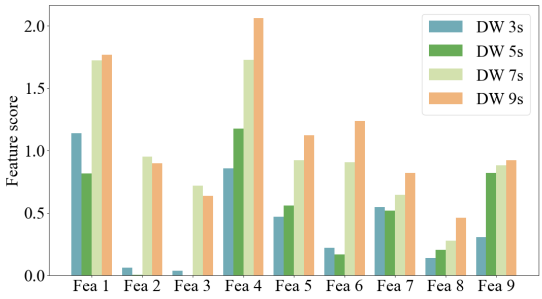
Metrics. To measure the detection and mitigation module, we used the classic metrics of *Accuracy*, *Precision*, *Recall*, and $F_1 - score$ in machine learning. The prediction time and memory overhead of classifiers were also taken into consideration. In addition, the average percent of attack flow, average flow table usage, average survival time of attack flow, and TableFull count were other aspects to verify the overall effect.

[3]Graphml-To-Mininet. https://github.com/yyd19981117/Graphml-To-Mininet

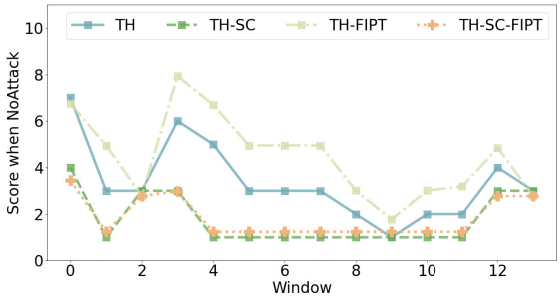
[4]Ryu sdn controller. https://github.com/faucetsdn/ryu

[5]Openvswitch version 2.5.9.https://www.openvswitch.org/download/

[6]Data set for imc 2010 data center measurement. http://pages.cs.wisc.edu/tbenson/imc10 data.html



(a) Features's importance value



(b) Score with SC, FS, and TH

Fig. 9. The score of feature and detection window.

B. Feature Selection and Parameter Setting

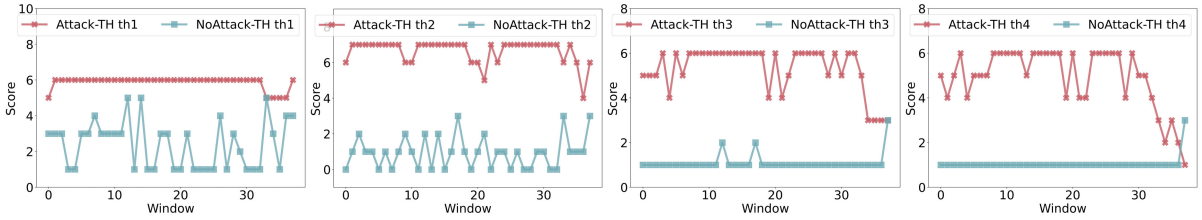
MWD-CFM is an architecture based on feature selection, threshold comparison, and classification. It is important to choose effective features that can help in better attack defense.

Feature selection. In the detection module, we extracted duration, packets, and bytes from all flow rules for feature analysis. For new flows, *packet_in* messages were also collected, which included count, IP, and *total_len* features. In addition, we monitored each incoming port of the switch and recorded the flow. In the mitigation section, the flow features including the 5-tuple were examined for matching and deletion. A total of 9 features were obtained as shown in Fig. 9a. The features within each window were given importance value by the Fisher Score algorithm. In windows 3s and 5s, features 2 and 3 do not have reference values, so they were not used in the actual detection.

We conducted feature ablation experiments to further validate the effectiveness of the flow features used. We divided the eight features into three groups: EIOR, LEN, and NUM, and used the Fisher score algorithm to obtain evaluation metrics, as shown in Table II. We can see that each feature category has a positive impact on detection performance. Therefore, feature extraction is reasonable.

Parameter setting. Due to the feature complexity of the flow rules in the actual network, they tend to show a scattered diversity. To make the threshold-based detection module as accurate as possible, the polling score is wished to converge

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.



(a) TH1, DW 3s **(b)** TH2, DW 3s **(c)** TH3, DW 3s **(d)** TH4, DW 3s

Fig. 10. The thresholds in detection window 3s.

TABLE II
FEATURE CATEGORY ABLATION EXPERIMENTS

Feature Group	Accuracy	Recall	F1-score
EIOR	0.9328	0.8535	0.9032
LEN	0.9424	0.9239	0.9314
NUM	0.7843	0.9144	0.7925
EIOR+LEN	0.9589	0.9372	0.9426
EIOR+NUM	0.9664	0.9435	0.9522
LEN+NUM	0.9664	0.9473	0.9557
EIOR+LEN+NUM	**0.9705**	**0.9526**	**0.9643**

as centrally as possible. It is evident from Fig. 9b that the polling score converges best when the threshold is paired with both the sensitivity coefficient and the significance value. Based on the feature values under each window, the threshold value range for each feature is established. The effect of different threshold (TH) values on the detection window 3 score under normal circumstances and during low-rate DDoS attacks is illustrated in Fig. 10. As the TH value rises, the discriminability of the total window score initially rises and then declines. We selected the thresholds that most effectively differentiate between the scores of attack flows and legitimate flows for our experimental setup. The same method was also used in other detection windows to obtain the most suitable threshold. The feature thresholds used in each window are illustrated in Table III. The effect of the SC on the window score is displayed in Fig. 14. As the SC increases, the differentiation between the scores of malicious flow rules and legitimate flow rules increases. But as it continues to rise, they become mixed. In the detection module, the SC set in this paper was 1.2. Fig. 9a shows the importance value of 9 features in different detection windows. The weights of the four detection windows were 0.135202, 0.251884, 0.300977, and 0.249590. Our motivation lies in the desire to remove as many malicious flow rules as possible, and by definition, the *Recall* metric is taken more into account.

C. Attack Detection and Mitigation

Using real-time detection, we evaluated the MWD module's performance. Depending on the actual needs, the controller turned on polling detection for the managed switches.

Fig. 11 displays the real characteristics of the attack and legitimate flows under DW 3s and DW 9s. It can be seen that when the detection window is smaller, the rules within a window are relatively limited, and therefore the complexity of the features is also limited in portrayal. When the detection window is longer, more flow rules are contained within a window, which also contains a wider variety of mixed cases of legitimate and attack ones. This makes the magnitude of fluctuations in features vary more drastically compared to a small window. Small windows and large windows have different scales, and we use their combination to improve the overall differentiation of the enhanced features.

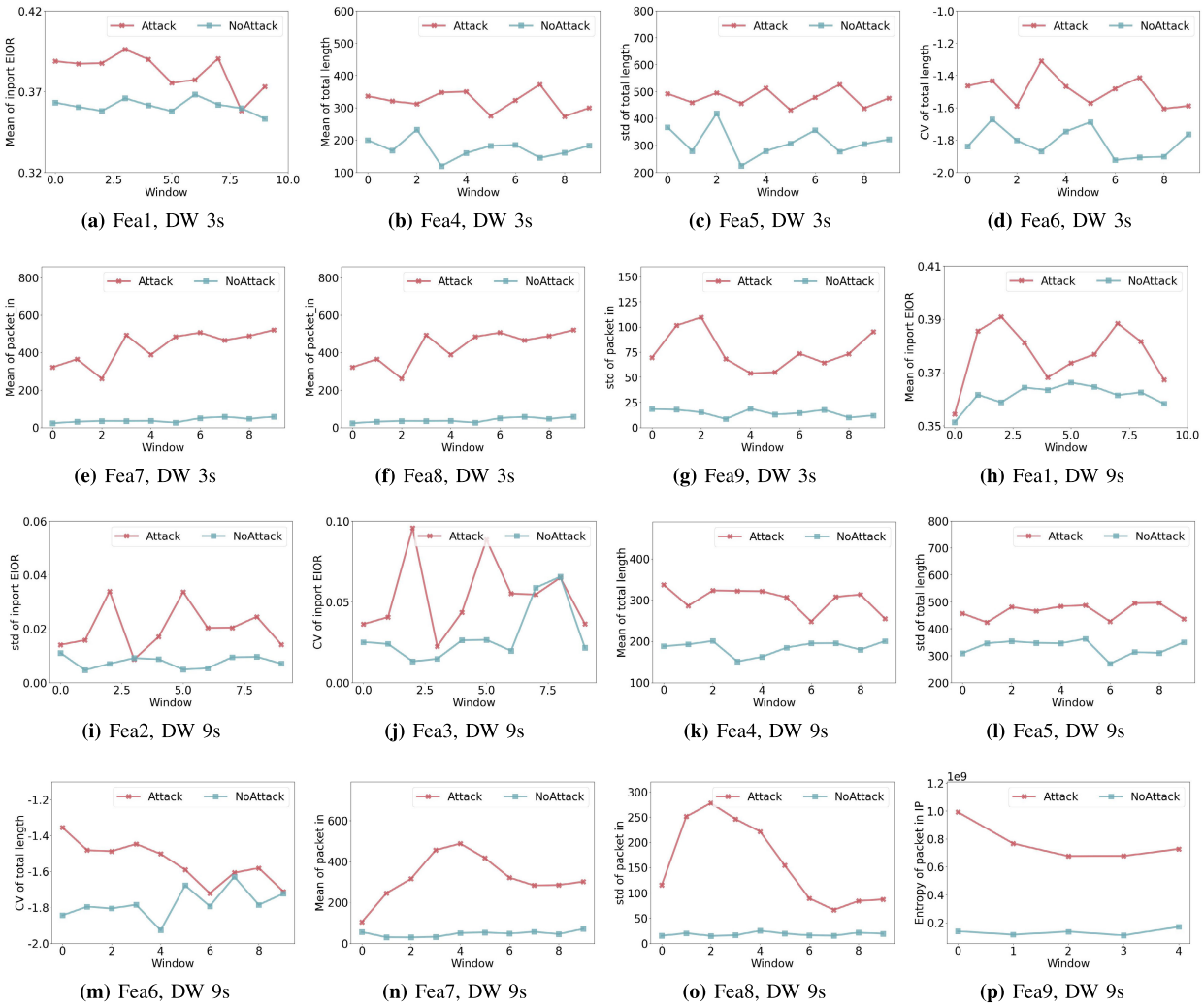
Timely and effective mitigation measures are especially important after low-rate DDoS attacks are monitored. Fig. 12 shows the *CVADP* features in the flow rules. It can be seen that the CFM module improves the differentiation after correcting the *ADP* features of each flow. The overall differentiation between *ABP* and *CVADP* is also more focused, which will help the MLP classifier for identification.

We also compared other commonly used classifiers, including SVM, GaussianNB, Linear Discriminant, Logistic Regression, Decision Tree, and KNN. Table IV displays the outcomes. Although KNN exhibits excellent performance, its prediction time is slightly longer and its memory consumption is high. Gaussian NB has a fast prediction time, but its *Recall* performance is not satisfactory. Since we are looking for higher recognition of malicious flow rules, we are more concerned about its *Recall* performance. The MLP used in this paper has the highest *Recall*, which is the reason why we choose MLP for the identification task.

Fig. 13a shows the flow rule changes during normal state, during the slow-rate DDoS attack state, and during the MWD-CFM architecture is implemented. Slow-rate DDoS attacks are distinguished by their gradually rising flow table usage. This may escape from the defense mechanisms for traditional high-rate DDoS attacks. As the attack continues and without mitigation, the attacker achieves the goal of maliciously consuming flow table space. As shown in Fig. 13, the percentage of legitimate ones in the limited TCAM gradually decreases. After the mitigation module is implemented, the malicious flows are deleted and the table space is freed. It is hard to totally remove the attack flows because of the actual network's complexity and variety, which might also cause the attack flows to resemble legitimate ones. However, the implementation of the CFM mitigation module gains more usable space for legitimate flows. In Fig. 15, we substituted the background traffic by the WIDE2024 dataset[7] to test the generality of MWD-CFM. Experimental results have shown

[7]Data set for wide 2024 data center measurement. https://mawi.wide.ad.jp/mawi/samplepoint-f/2024/

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.



(a) Fea1, DW 3s **(b)** Fea4, DW 3s **(c)** Fea5, DW 3s **(d)** Fea6, DW 3s

(e) Fea7, DW 3s **(f)** Fea8, DW 3s **(g)** Fea9, DW 3s **(h)** Fea1, DW 9s

(i) Fea2, DW 9s **(j)** Fea3, DW 9s **(k)** Fea4, DW 9s **(l)** Fea5, DW 9s

(m) Fea6, DW 9s **(n)** Fea7, DW 9s **(o)** Fea8, DW 9s **(p)** Fea9, DW 9s

Fig. 11. The feature in detection window 3s and 9s.

TABLE III
THRESHOLDS OF DIFFERENT DETECTION WINDOWS

Detection Window	Fea 1	Fea 2	Fea 3	Fea 4	Fea 5	Fea 6	Fea 7	Fea 8	Fea 9
DW 3	0.35872162	-	-	184.412792965	324.19878857	-1.76108506	56.51315789	26.63097081	51142479.22427577
DW 5	0.35604629	-	-	185.959907852	329.27056634	-1.77261799	52.86086956	18.93689962	70444129.5574309
DW 7	0.35777366	0.00889675	0.0251514470	184.18253832	326.79962257	-1.77867703	57.94789915	35.75577576	116234278.34388034
DW 9	0.35629593	0.00970590	0.0277592276	183.91987271	326.88107452	-1.77847360	58.04563492	37.84123313	146639666.08191738

TABLE IV
COMPARISON RESULTS WITH OTHER CLASSIFIERS

Classifier	Accuracy	Precision	Recall	$F_1 - score$	Prediction time (s)	Memory overhead
SVM	0.974699	0.989356	0.956620	0.972713	0.000186	57KB
GaussianNB	0.941577	**0.998959**	0.876978	0.934003	0.000008	1KB
Linear Discriminant	0.909996	0.892163	0.920310	0.906018	0.000160	1KB
Logistic Regression	0.915233	0.933062	0.883567	0.907640	0.000157	1KB
Decision Tree	0.965472	0.992235	0.934064	0.962271	**0.000004**	13KB
KNN	**0.981478**	0.985703	0.974847	**0.980345**	0.000433	2603KB
MLP	0.958943	0.922498	**0.996633**	0.958134	0.000185	19KB

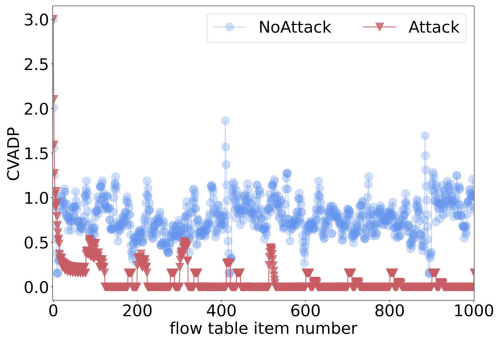
that MWD-CFM can still effectively identify malicious flows in different network environments.

A test lasting 480s was set to compare with other mitigation strategies in Table V. EVICT [38] is a strategy supported

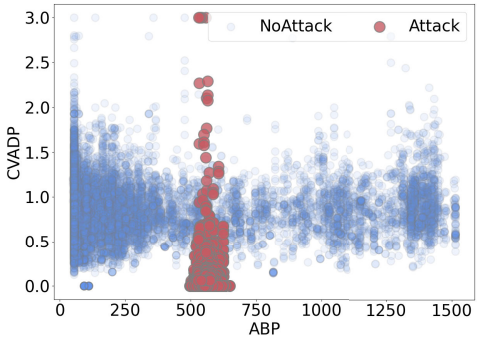
Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

TABLE V
COMPARISON RESULTS WITH OTHER METHODS

Method	Deploy Plane	Average Percent of Attack Flow	Average Flowtable Usage	Average Survival Time of Attack Flow	TableFull Count
None	None	60.43%	94.35%	198.51	69262
EVICT	Data	60.16%	96.08%	161.58	0
SIFT	Control	60.22%	94.44%	190.01	68697
SFTOGuard	Control	25.43%	77.67%	8.62	1095
LtRFT	Data	21.87%	74.11%	7.80	600
MWD-CFM	Control	19.69%	72.81%	7.55	490

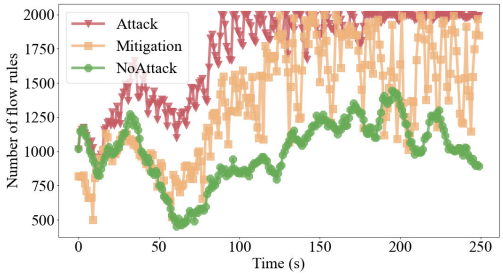


(a) Feature CVADP

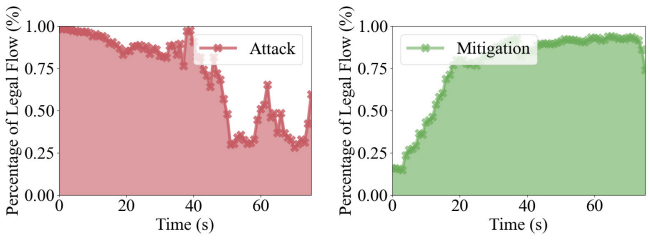


(b) Feature ABP with CVADP

Fig. 12. The ABP with CVADP under attack and no attack.



(a) The number of flowrules when flow table is 2000



(b) The proportion of legel flow rules under attack **(c)** The proportion of legel flow rules under mitigation

Fig. 13. The proportion of legal flow rules.

by the OpenFlow protocol, which operates at the data level. Enabling this strategy prevents the generation of table_full messages. When the flow table becomes saturated, EVICT autonomously removes entries to free up available space. In contrast, SIFT [34] is a response strategy implemented at the control layer that removes flow table entries with a specific probability when saturation occurs. However, this method does not distinguish between different flow table entries, which can lead to more normal flows being removed and attack flows being continuously retained. SFTOGuard [47] and LtRFT [12] are both methods deployed at the data layer. SFTOGuard uses LRCN to predict the number of flow entries and adaptively deletes flow rules. LtRFT sorts flow entries and uses the LtR algorithm to prioritize malicious flows, in order to identify and mitigate attacks. The MWD-CFM architecture shown in this research effectively minimizes the attack flows' survival time within the flow table when compared to the EVICT and SIFT approaches. Simultaneously, both the average percentage of attack flows and the flow table's average usage rate have decreased. Combined with Fig. 16, there are also fewer TableFull messages, which alleviates the pressure on the southbound channel. The southbound channel can have fewer threats when it is not filled with a large number of switch status messages.

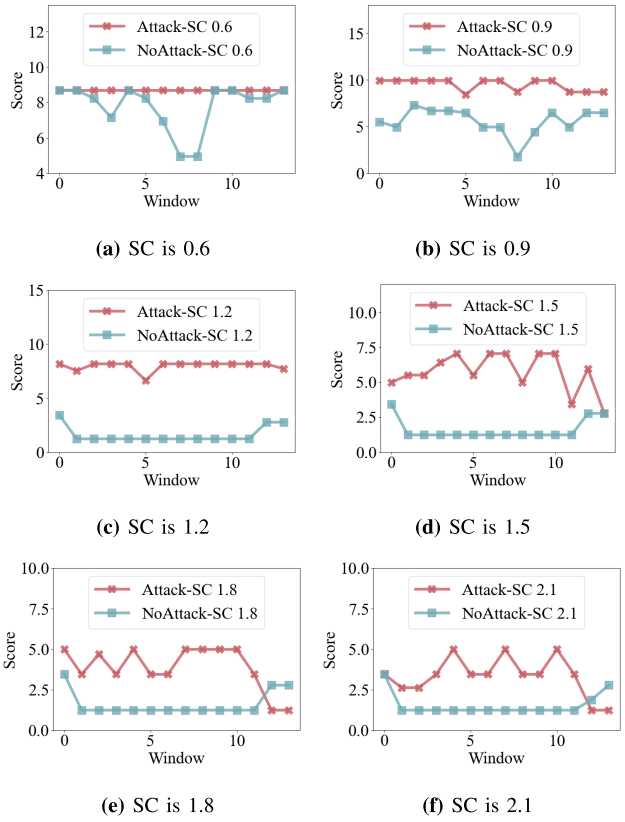
Fig. 17 shows the consumption of the MWD-CFM architecture deployment for CPU and memory. After deploying the architecture of this paper, the CPU growth is barely more than 1%, which can be considered a lightweight program for the controller. The entire program consumes no more than 100MB of memory, which is also acceptable.

Experiments have proven that MWD-CFM deployment can identify and counteract low-rate DDoS attacks against SDN flow tables, effectively freeing up maliciously occupied flow table space while reducing the burden for controllers and communication links.

VI. DISCUSSION

In this research, we also examined the possibility of attackers exploiting defenses and proposed mitigation measures. If the attacker obtains the features and thresholds used by the defense method, they can modify the attack frequency

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.



(a) SC is 0.6 **(b)** SC is 0.9

(c) SC is 1.2 **(d)** SC is 1.5

(e) SC is 1.8 **(f)** SC is 2.1

Fig. 14. The value of SC.

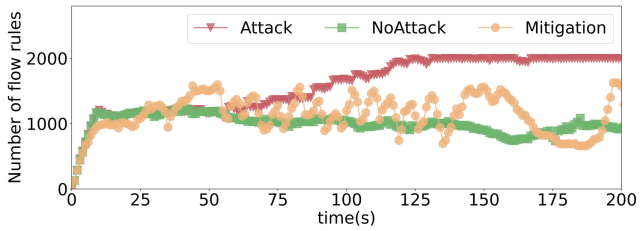


Fig. 15. The number of flowrules when background traffic is WIDE.

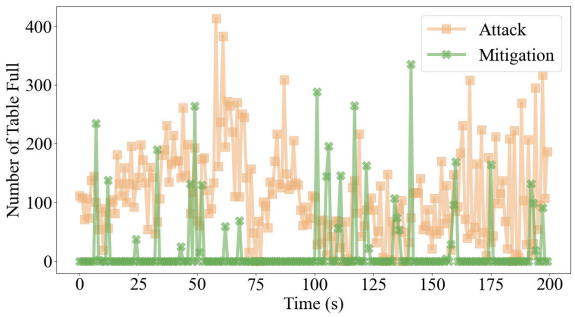
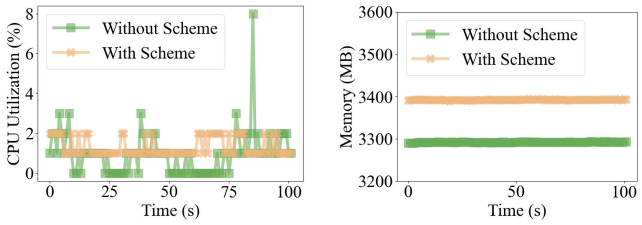


Fig. 16. The number of TableFull messages.

and attack packet size to keep the score slightly above the threshold, thereby consuming controller resources. In response to this, we will temporarily use a higher sensitivity coefficient to avoid the attacker's influence and prevent the defense module from being frequently triggered. After a period of time,



(a) The CPU utilization change **(b)** The memory utilization change

Fig. 17. The CPU and memory utilization.

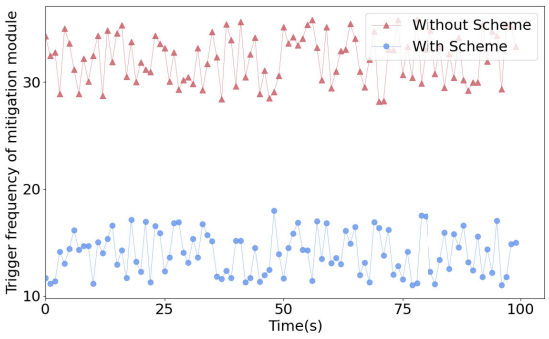


Fig. 18. The number of triggers for defense module.

if the attack no longer persists, the sensitivity coefficient will be restored to its original value. The corresponding algorithm is as follows:

Algorithm 3 Defense Against Adversarial Attacks

Input: $CurrentSC$, $updateSC$, $Threshold1$, $Threshold2$, $PollingScore$

Initialize $count$, θ;
if $0 < PollingScore - Threshold1 < \theta$ **then**
  $count \leftarrow count + 1$;
  if $count > Threshold2$ **then**
    Set $CurrentSC \leftarrow updateSC$;
    Sleep Ns;
    Set $CurrentSC \leftarrow origin\_value$;
else if $PollingScore < Threshold1$ **then**
  $count \leftarrow 0$;

Fig. 18 shows the effect of increasing sensitivity coefficient, with a significant decrease in the number of times the defense module is triggered.

VII. Conclusion

In this paper, we presented MWD-CFM to defend the SDN data plane's slow-rate DDoS attacks against flow tables. We set multi-detection windows to work cooperatively when polling, monitoring, and detecting. Different features were assigned different importance with the Fisher score algorithm in each window. In the mitigation module, duration, packet, and byte of flow rules were taken into consideration for MLP classification. Flow rule features were corrected to improve the effectiveness of attack recognition and traffic removal.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

We evaluated MWD-CFM on real network topology and datasets. Experiments showed that our architecture had a good performance in greatly reducing the survival time of attack flows, ensuring the available space of legitimate flows with a low CPU and memory consumption.

In future work, we will further enhance the ability of MWD-CFM to protect switch flow tables in SDN. Its adaptability should be strengthened to effectively respond to a variety of complex network environments. In addition, since the source of DDoS attacks has not been cut off, there is still a risk of overflow in the flow tables. Further research is needed on how to locate and prevent attackers accurately and completely.

REFERENCES

[1] M. Casado et al., "SANE: A protection architecture for enterprise networks," in *Proc. USENIX Secur. Symp.*, 2006, p. 50.

[2] J. Kim et al., "Enhancing security in SDN: Systematizing attacks and defenses from a penetration perspective," *Comput. Netw.*, vol. 241, Mar. 2024, Art. no. 110203.

[3] C. J. Kaur, B. Abhinav, and B. Sunny, "DDoS attacks & defense mechanisms in SDN-enabled cloud: Taxonomy, review and research challenges," *Comput. Sci. Rev.*, vol. 53, Aug. 2024, Art. no. 100644.

[4] J. Chen et al., "Fault tolerance oriented SFC optimization in SDN/NFV-enabled cloud environment based on deep reinforcement learning," *IEEE Trans. Cloud Comput.*, vol. 12, no. 1, pp. 200–218, Jan. 2024.

[5] Z. H. Ali, N. El-Rashidy, M. A. Elhosseini, and S. M. Ayyad, "SDN-based reliable emergency message routing schema using digital twins for adjusting beacon transmission in VANET," *J. Netw. Comput. Appl.*, vol. 230, Oct. 2024, Art. no. 103944.

[6] C. Fan, J. Cui, H. Zhong, I. Bolodurina, and D. He, "MM-SDVN: Efficient mobility management scheme for optimal network handover in software-defined vehicular network," *IEEE Internet Things J.*, vol. 11, no. 19, pp. 32089–32104, Oct. 2024.

[7] D. Tang, R. Dai, Y. Yan, K. Li, W. Liang, and Z. Qin, "When SDN meets low-rate threats: A survey of attacks and countermeasures in programmable networks," *ACM Comput. Surveys*, vol. 57, no. 4, pp. 1–32, Apr. 2025.

[8] H. Younis and M. M. N. Hamarsheh, "ONOS flood defender: A real-time flood attacks detection and mitigation system in SDN networks," *Concurrency Comput., Pract. Exper.*, vol. 37, nos. 4–5, p. 8388, Feb. 2025.

[9] M. Yue, M. Wang, and Z. Wu, "Low-high burst: A double potency varying-RTT based full-buffer shrew attack model," *IEEE Trans. Dependable Secure Comput.*, vol. 18, no. 5, pp. 2285–2300, Sep. 2021.

[10] D. Tang, S. Wang, B. Liu, W. Jin, and J. Zhang, "GASF-IPP: Detection and mitigation of LDoS attack in SDN," *IEEE Trans. Services Comput.*, vol. 16, no. 5, pp. 3373–3384, Sep. 2023.

[11] A. Kaur, C. Rama Krishna, and N. V. Patil, "A comprehensive review on software-defined networking (SDN) and DDoS attacks: Ecosystem, taxonomy, traffic engineering, challenges and research directions," *Comput. Sci. Rev.*, vol. 55, Feb. 2025, Art. no. 100692.

[12] D. Tang, Y. Yan, C. Gao, W. Liang, and W. Jin, "LtRFT: Mitigate the low-rate data plane DDoS attack with learning-to-rank enabled flow tables," *IEEE Trans. Inf. Forensics Security*, vol. 18, pp. 3143–3157, 2023.

[13] R. F. Kapourchali, R. Mohammadi, and M. Nassiri, "P4httpGuard: Detection and prevention of slow-rate DDoS attacks using machine learning techniques in P4 switch," *Cluster Comput.*, vol. 27, no. 6, pp. 8047–8064, Sep. 2024.

[14] M. Sinha, P. Bera, M. Satpathy, K. S. Sahoo, and J. J. P. C. Rodrigues, "DDoSBlocker: Enhancing SDN security with time-based address mapping and AI-driven approach," *Comput. Netw.*, vol. 259, Mar. 2025, Art. no. 111078.

[15] M. Sinha, P. Bera, and M. Satpathy, "SYN-monitor: An energy efficient defense system against TCP-SYN flooding attacks in SDN," in *Proc. 26th Int. Conf. Distrib. Comput. Netw.*, Jan. 2025, pp. 346–351.

[16] M. Zhang et al., "Control plane reflection attacks and defenses in software-defined networks," *IEEE/ACM Trans. Netw.*, vol. 29, no. 2, pp. 623–636, Apr. 2021.

[17] D. Tang, X. Wang, P. Tan, Z. Qin, K. Li, and J. Zhang, "DNSGreen: A comprehensive defense system against bounce-style DNS DDoS attacks with P4," *IEEE Trans. Comput.*, vol. 75, no. 1, pp. 247–260, Jan. 2026.

[18] D. Tang, B. Liu, K. Li, S. Xiao, W. Liang, and J. Zhang, "PLUTO: A robust LDoS attack defense system executing at line speed," *IEEE Trans. Dependable Secure Comput.*, vol. 22, no. 3, pp. 2855–2872, May 2025.

[19] D. Kong, C. Wu, Y. Shen, X. Chen, H. Liu, and D. Zhang, "TableGuard: A novel security mechanism against flow table overflow attacks in SDN," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2022, pp. 4167–4172.

[20] J. Cao, M. Xu, Q. Li, K. Sun, and Y. Yang, "The LOFT attack: Overflowing SDN flow tables at a low rate," *IEEE/ACM Trans. Netw.*, vol. 31, no. 3, pp. 1416–1431, Jun. 2023.

[21] K. Kalkan, L. Altay, G. Gur, and F. Alagoz, "JESS: Joint entropy-based DDoS defense scheme in SDN," *IEEE J. Sel. Areas Commun.*, vol. 36, no. 10, pp. 2358–2372, Oct. 2018.

[22] V. A. Shirsath, M. M. Chandane, C. Lal, and M. Conti, "SYNTROPY: TCP SYN DDoS attack detection for software defined network based on Rényi entropy," *Comput. Netw.*, vol. 244, May 2024, Art. no. 110327.

[23] A. Ahalawat, K. S. Babu, A. K. Turuk, and S. Patel, "A low-rate DDoS detection and mitigation for SDN using Renyi entropy with packet drop," *J. Inf. Secur. Appl.*, vol. 68, Aug. 2022, Art. no. 103212.

[24] A. Mishra, N. Gupta, and B. B. Gupta, "Defense mechanisms against DDoS attack based on entropy in SDN-cloud using POX controller," *Telecommun. Syst.*, vol. 77, no. 1, pp. 47–62, May 2021.

[25] N. Ahuja, G. Singal, D. Mukhopadhyay, and N. Kumar, "Automated DDOS attack detection in software defined networking," *J. Netw. Comput. Appl.*, vol. 187, Aug. 2021, Art. no. 103108.

[26] T. A. Tang, L. Mhamdi, D. McLernon, S. A. R. Zaidi, and M. Ghogho, "Deep recurrent neural network for intrusion detection in SDN-based networks," in *Proc. 4th IEEE Conf. Netw. Softwarization Workshops (NetSoft)*, Jun. 2018, pp. 202–206.

[27] R. F. Fouladi, O. Ermiş, and E. Anarim, "A DDoS attack detection and countermeasure scheme based on DWT and auto-encoder neural network for SDN," *Comput. Netw.*, vol. 214, Sep. 2022, Art. no. 109140. [Online]. Available: https://www.sciencedirect.com/science/article/pii/S1389128622002560

[28] W. Sun, S. Guan, P. Wang, and Q. Wu, "A hybrid deep learning model based low-rate DoS attack detection method for software defined network," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 5, p. 4443, May 2022.

[29] R. K. Chouhan, M. Atulkar, and N. K. Nagwani, "A framework to detect DDoS attack in ryu controller based software defined networks using feature extraction and classification," *Appl. Intell.*, vol. 53, no. 4, pp. 4268–4288, Feb. 2023.

[30] M. Aslam et al., "Adaptive machine learning based distributed denial-of-services attacks detection and mitigation system for SDN-enabled IoT," *Sensors*, vol. 22, no. 7, p. 2697, Mar. 2022.

[31] M. Shakil, A. Fuad Yousif Mohammed, R. Arul, A. K. Bashir, and J. K. Choi, "A novel dynamic framework to detect DDoS in SDN using Metaheuristic clustering," *Trans. Emerg. Telecommun. Technol.*, vol. 33, no. 3, p. 3622, Mar. 2022.

[32] R. Dai, D. Tang, Z. Qin, K. Chen, K. Li, and J. Zhang, "Detecting congestion-related attacks via fine-grained queue diagnosis," *IEEE Trans. Cognit. Commun. Netw.*, vol. 12, pp. 1255–1268, 2026.

[33] J. Li, T. Tu, Y. Li, S. Qin, Y. Shi, and Q. Wen, "DoSGuard: Mitigating denial-of-service attacks in software-defined networks," *Sensors*, vol. 22, no. 3, p. 1061, Jan. 2022.

[34] T. A. Pascoal, Y. G. Dantas, I. E. Fonseca, and V. Nigam, "Slow TCAM exhaustion DDoS attack," in *Proc. IFIP Int. Conf. ICT Syst. Secur. Privacy Protection*, 2017, pp. 17–31.

[35] S. Batool et al., "Lightweight statistical approach towards TCP SYN flood DDoS attack detection and mitigation in SDN environment," *Secur. Commun. Netw.*, vol. 2022, pp. 1–14, Jan. 2022.

[36] Z. Huang, X. Huang, J. Li, K. Xue, Q. Sun, and J. Lu, "LLDM: Low-latency DoS attack detection and mitigation in SDN," in *Proc. IEEE 23rd Int. Conf. High Perform. Switching Routing (HPSR)*, Jun. 2022, pp. 169–174.

[37] A. El Kamel, H. Eltaief, and H. Youssef, "On-the-fly (D)DoS attack mitigation in SDN using deep neural network-based rate limiting," *Comput. Commun.*, vol. 182, pp. 153–169, Jan. 2022.

[38] B. Isyaku, K. B. Abu Bakar, F. A. Ghaleb, and S. Sulaiman, "Performance evaluation of flowtable eviction mechanisms for software defined networks considering traffic flows variabilities," in *Proc. IEEE 12th Symp. Comput. Appl. Ind. Electron. (ISCAIE)*, May 2022, pp. 71–75.

[39] B. Yuan, D. Zou, S. Yu, H. Jin, W. Qiang, and J. Shen, "Defending against flow table overloading attack in software-defined networks," *IEEE Trans. Services Comput.*, vol. 12, no. 2, pp. 231–246, Mar. 2019.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.

[40] M. Abu Rajab, J. Zarfoss, F. Monrose, and A. Terzis, "A multifaceted approach to understanding the botnet phenomenon," in *Proc. 6th ACM SIGCOMM Conf. Internet Meas.*, Oct. 2006, pp. 41–52.
[41] X. Liu, X. Lu, X. Fang, and L. Shang, "Low-rate denial-of-service attack detection method under software defined network environment," *J. Comput. Appl.*, vol. 42, no. 4, p. 1301, 2022.
[42] D. Tang, R. Dai, C. Zuo, J. Chen, K. Li, and Z. Qin, "A low-rate DoS attack mitigation scheme based on port and traffic state in SDN," *IEEE Trans. Comput.*, vol. 74, no. 5, pp. 1758–1770, May 2025.
[43] L. Sun, T. Wang, W. Ding, J. Xu, and Y. Lin, "Feature selection using Fisher score and multilabel neighborhood rough sets for multilabel classification," *Inf. Sci.*, vol. 578, pp. 887–912, Nov. 2021.
[44] K. J. Singh and T. De, "MLP-GA based algorithm to detect application layer DDoS attack," *J. Inf. Secur. Appl.*, vol. 36, pp. 145–153, Oct. 2017.
[45] S. Xie, C. Xing, G. Zhang, and J. Zhao, "A table overflow LDoS attack defending mechanism in software-defined networks," *Secur. Commun. Netw.*, vol. 2021, pp. 1–16, Jan. 2021.
[46] M. Yu, T. Xie, T. He, P. McDaniel, and Q. K. Burke, "Flow table security in SDN: Adversarial reconnaissance and intelligent attacks," *IEEE/ACM Trans. Netw.*, vol. 29, no. 6, pp. 2793–2806, Dec. 2021.
[47] D. Tang, D. Zhang, Z. Qin, Q. Yang, and S. Xiao, "SFTO-guard: Real-time detection and mitigation system for slow-rate flow table overflow attacks," *J. Netw. Comput. Appl.*, vol. 213, Apr. 2023, Art. no. 103597.

Dan Tang received the Ph.D. degree from the Huazhong University of Science and Technology in 2014. He is currently an Associate Professor with the College of Cyber Science and Technology, Hunan University (HNU), Changsha, China. His research interests include the areas of computer network security, computer information security, and architecture of future internet.

Chenguang Zuo received the B.E. degree in information security from Hunan University (HNU), Changsha, China, in 2023, where he is currently pursuing the master's degree with the College of Cyber Science and Technology. His research interests include RDMA, programmable data plane, and network attack detection.

Xinmeng Li received the B.E. degree in information security from Hunan University (HNU), Changsha, China, in 2022, where she is currently pursuing the master's degree with the College of Computer Science and Electronic Engineering (CSEE). Her research interests include SDN, programmable data plane, and network attack detection.

Siyuan Wang received the B.E. degree in information security from Hunan University, Changsha, China, in 2021, where she is currently pursuing the master's degree with the College of Computer Science and Electronic Engineering (CSEE). Her research interests include network attack detection and mitigation.

Wei Liang received the Ph.D. degree in computer science and technology from Hunan University, China, in 2013. He was a Post-Doctoral Scholar with Lehigh University, Bethlehem, PA, USA, from 2014 to 2016. He is currently a Professor with the School of Computer Science and Engineering, Hunan University of Science and Technology. He has authored or co-authored more than 140 journals/conference papers, such as IEEE TRANSACTIONS ON COMPUTERS, IEEE TRANSACTIONS ON INDUSTRIAL INFORMATICS, IEEE TRANSACTIONS ON INTELLIGENT TRANSPORTATION SYSTEMS, IEEE TRANSACTIONS ON EMERGING TOPICS IN COMPUTING, IEEE TRANSACTIONS ON COMPUTATIONAL BIOLOGY AND BIOINFORMATICS, and IEEE INTERNET OF THINGS JOURNAL. His research interests include blockchain security technology, networks security protection, embedded system and hardware IP protection, fog computing, and security management in wireless sensor networks (WSN).

Keqin Li (Fellow, IEEE) is currently a SUNY Distinguished Professor of computer science with The State University of New York and also a National Distinguished Professor with Hunan University. He is listed in a Scilit Top Cited Scholars from 2023 to 2025 and is among the Top 0.02% out of over 20 million scholars Worldwide based on top-cited publications in the last ten years. His research interests include cloud computing, high performance computing, computer networking, and machine learning. He is an AAAS Fellow and an AIIA Fellow. He is a member of European Academy of Sciences and Arts. He is a member of Academia Europaea (Academician of the Academy of Europe).

Jiliang Zhang (Senior Member, IEEE) is currently a Full Professor with the College of Integrated Circuits, Hunan University. He is also the Vice Dean of the College of Integrated Circuits, Hunan University, and the Secretary-General of CCF Fault-Tolerant Computing Professional Committee. He has authored more than 80 technical papers in leading journals and conferences. His current research interests include hardware security, integrated circuit design, and intelligent systems. He was a recipient of the CCF Integrated Circuit Early Career Award, the CCF Distinguished Speaker, and the winner of Excellent Youth Fund of the NSFC. He has been a Program Committee Member for a number of well-known conferences, such as DAC, ASP-DAC, GLSVLSI, and FPT.

Authorized licensed use limited to: HUNAN UNIVERSITY. Downloaded on March 26,2026 at 00:42:47 UTC from IEEE Xplore. Restrictions apply.