Hindawi
Security and Communication Networks
Volume 2022, Article ID 7409083, 16 pages
https://doi.org/10.1155/2022/7409083

Research Article

Cross-Plane DDoS Attack Defense Architecture Based on Flow Table Features in SDN

Meng Yue,[1] **Qingxin Yan**,[2] **Han Zheng**,[2] **and Zhijun Wu**[1]

[1]*College of Safety Science and Engineering, Civil Aviation University of China, Tianjin 300300, China*
[2]*College of Electronic Information and Automation, Civil Aviation University of China, Tianjin 300300, China*

Correspondence should be addressed to Meng Yue; myue@cauc.edu.cn

Received 11 May 2022; Revised 2 September 2022; Accepted 9 September 2022; Published 30 September 2022

Academic Editor: Wenjuan Li

Software-Defined Networking (SDN) actualizes the separation of control and forwarding and innovates network functionalities with a logically centralized controller. Contemporary SDN infrastructure exposes the potential bottlenecks which are prone to engage in distributed denial of service attack (DDoS) thus posing an ever-increasing threat. This paper adopts the idea of "cross-plane collaboration" accomplishing DDoS attack defense and incorporates a two-phase project deploying the lightweight detection mechanism in data layer and the fine-grained filtering model in control layer. The coadjutant detection mechanism introduces a novel three-dimensional entropy consisting of five flow table features performing real-time feature detection; the defense strategy schedules an attack classification algorithm based on neural network by means of extracting four flow rule features designed to locate compromised interfaces occupied by malicious traffic. Extensive experiments are implemented to demonstrate the method we proposed brings excellent superiority. The detection rate of the classification filtering model is 99.4%, and it is real-time, with a detection time of 0.51s. In addition, the method of cross-layer defense reduces the CPU utilization of the controller.

1. Introduction

Software-Defined Networking (SDN) is a brand-new network canonical form with a centralized controller governing the whole network, which decouples the control plane and data plane, and breaks the dilemma of "relative closure" of traditional networks. However, emerging technologies bring new security concerns and new threats that do not exist in current traditional networks; DDoS attacks [1] with highly concealed and destructive traits comprise attack host groups by initiating massive zombie hosts or virtual machines. The attacker governs the attack host groups to send continuously a large number of forged data packets to the victims. At one time, generous mendacious data packets swarm into the victim server, consuming the server's memory, CPU, link bandwidth, and other resources; this severely damages the server and prevents the service reaching legitimate users.

DDoS attacks in traditional networks are primarily divided into resource exhausted DDoS attacks and bandwidth exhausted DDoS attacks. The former primarily depletes resources of the victims by sending abundant data packets, including CPU computation, memory, and TCP connection. The routine SYN Flood attack sends a large amount of SYN messages to the server. After receiving the SYN message, the server returns a confirmation message to the client, but it never receives a response from the client. Then the server keeps sending confirmation messages until it crashes. Bandwidth exhausted attack is communication links–oriented. Immeasurably, the server can scarcely offer reliable service if the bandwidth of the communication link is completely consumed, typically concentrating on ARP flooding attack and UDP flooding attack. When an attacker initiates an ARP flooding attack, the attacker first sends substantial ARP messages to the target link and subsequently transmits numerous ARP responses, thus occupying link bandwidth until link congestion [2].

The controller emblematizes the most critical sector of the SDN. The attacker counterfeits a large volume of data packets with false IP addresses to trigger malicious requests, which overloads the centralized controller, resulting in single point failure of controller and network paralysis.

DDoS attacks against SDN data layer primarily focus on Openflow switch. According to Openflow southbound protocol, whenever data packets arrive at the switch, the switch initially queries whether there are matching forwarding rules. If the switch has no way to handle these packets conversely sending encapsulated packet-in messages to the controller, the controller sends packet-out messages to instruct flow rules installation. All flow rules in the switch forming flow table are stored in the ternary content addressable memory (TCAM). Due to cost and power consumption, the TCAM space of Openflow switch is considerably limited can merely store a certain amount of flow rules [3–5]. Therefore, exhausting TCAM resources has become the main target of DDoS attacks against the data layer. The DDoS attacks against the data layer in SDN are shown in Figure 1.

The attacker inculcates the elaborate data packets periodically, and each distinctive data packet evokes the controller dispatching flow rules. The periodically sent data packets prevent attack flow rules from being deleted due to triggering of the timeout mechanism. When the flow table is gradually completely filled with the malicious flow rules, the switch barely provides additional resources to install flow rules for the legitimate requests, and ultimately the attacker accomplishes the purpose of denial of service.

DDoS attacks against the control layer require extraordinarily extensive botnet; the attack characteristics are relatively obvious with lower concealment [6–8]. By comparison, the DDoS attack on the data layer owns lower attack rate and more powerful concealment, which can be quite challenging to detect, getting focal point in recent research [9–11].

Attackers utilize the security vulnerabilities of SDN to launch DDoS attacks, which tremendously threatens the security of SDN [12–14]. The scale of attack is piecemeal escalating and the means of attack are evolving in a steady stream, which leads to the emergence of new problems ceaselessly. Therefore, how to deal with DDoS attacks under SDN has always been a research hotspot, and it is of certain significance to study the corresponding defense methods. However, current detection solutions suffer from high false positive rate in the flash crowd scene; meanwhile, the detection method now available based on single entropy intersperse serious boundedness and scarcely counteract diversified patterns of attacks; besides, the existing defense methods insufficiently engage efficacious filtering mechanisms.

By exploiting the vulnerability that the current SDN switches supporting OpenFlow universally have restricted TCAM space and can only conserve flow rules with the preset capacity, we constructed DDoS attacks with forged source IP addresses and forged source ports in this paper. These fake IP addresses and port numbers quickly swarm into the victim to occupy the TCAM capacity and consume flow table resources, which brings nonnegligible destructiveness for normal users. Giving top priority to researching the characteristics of DDoS attacks in SDN, this paper presents a cross-plane cooperatively DDoS attack defense architecture based on the characteristics of SDN switch flow

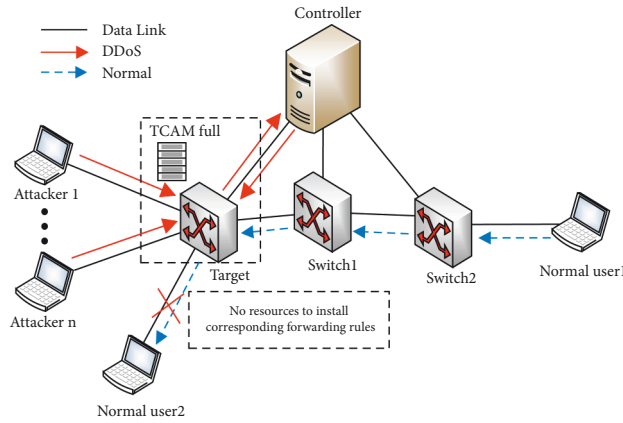

Figure 1: DDoS attack on SDN data plane.

table, which consists of a coarse-grained detection on the data layer and a fine-grained defense on the control layer. The data layer has certain programming ability. Existing OpenFlow switches have one or more central processing units, which have powerful computing power and abundant unused resources, making it possible to deploy detection methods in the data layer. We deploy the coarse-grained entropy detection in the data layer and the flow rule filtering based on neural networks in the control layer, which not only ensures that the attack flow rules can be filtered out timely and accurately after the attack is detected but also reduces the communication overhead on the basis of ensuring the real-time detection. The detection method is directly deployed in the data layer, so that the switch can directly extract and process data from the flow table, which reduces the detection delay to the minimum, and reduces the southbound link blocking and controller load caused by controller polling. In the detection stage, the switch performs lightweight real-time detection based on three-dimensional entropy. If there are abnormalities, the controller receives the alarm information reported by the switch; the machine learning classification algorithm deployed in the control layer will extract four flow features for sophisticated classification and filter attack flow rules to ensure the access requirements of normal users. The cross-plane defense architecture proposed ensures real-time detection and avoids frequent polling between the controller and the switch, so that the defense architecture can complete effective defense with low resource overhead.

In conclusion, the contributions of this paper are as follows:

(1) For distributed denial of service attacks under SDN, a cross layer defense method based on flow table characteristics is proposed according to the idea of "cross layer cooperation," which can effectively defend attacks and reduce the amount of computation.

(2) Aiming at the problems now available in the existing DDoS attack detection methods under SDN, a three-dimensional entropy detection based on the characteristics of switch flow table is proposed, consisting of five flow table features: source/destination IP

address entropy, source/destination port entropy, and packet size entropy, which alleviates the problem of high false alarm rate in flash scene and reduces the detection time.

(3) To be directed against the problems in the existing DDoS attack defense methods under SDN, we exploit neural network classification mechanism extracting four flow features of the switch, constituted by packet interval variance, number of matching packets, flow duration, and the relative dispersion of matching bytes, so that the neural network can locate the specific attack flow rules according to the detection features. At the same time, the proposed classification model is improved to enhance the efficiency of the model.

The rest of this article is organized as follows: Section 2 presents the related work, summarizes the current research status and subsistent bottlenecks, and presents the motivation for this paper. Section 3 expounds the architecture of our proposed framework. Section 4 covers the experiments to verify the proposed defense architecture and conducts a comparison with others methods. Section 5 draws the conclusion and indicates the future work prospect.

2. Related Work

DDoS attack has shown a great ability to cause damage, which is one of the biggest menaces to large networks and information systems in the period of cloud computing, 5G, and big data environment, and the security is the most prominent issue in the Internet of Things (IoT) [15]. The emerging technology creates new security concerns and new threats that do not exist in the current traditional networks. Giving a high-performance solution to defend DDoS attacks has always been a puzzle. A lot of researches have been devoted to revealing new vulnerabilities or designing effective anti-DDoS countermeasures. Liming Fang et al. [16] introduced the smart contracts to analyze malicious behavior of users and execute punishment measures in the 5G scenario, which can launch counterattacks before attacks occur, thus increasing the cost of attacks. AI module was embedded in smart contracts to accurately detect the attack. They also introduced a game mechanism while maintaining efficiency of communication, and constructed a mapping relationship between the smart contract account and network location so that external attacks cannot avoid the audit of the smart contracts. Zhou et al. [17] established a mechanism of hybrid active defense combined with Moving Target Defense (MTD) technology to confuse attackers by spreading camouflaging information through network spoofing in IoT networks.

SDN has a broad development prospect due to its flexible deployment rules and global view of the controller. However, given that SDN is still in the development stage, it readily grabs attention from multifarious attacks, especially of DDoS attacks. This section summarizes the existing types of DDoS attacks and analyzes the problems in the detection of and defense against DDoS attacks in SDN and proposes solutions to the existing problems.

2.1. Types of DDoS Attacks. DDoS attacks derive numerous variants such as the classic SYN Flood and UDP Flood, ICMP Flood, HTTP Flood [18], DDoS Amplification Attacks, Link Flooding Attack (LFA), etc. The SYN-Flood attack exploits a vulnerability in TCP and sends a large number of SYN packets with false source addresses to the target host. Shin et al. [19] introduced connection migration and actuating triggers into the SDN architecture to cope with the challenge when encountering the SYN Flood attack. Recently, a new type of DDoS attack, known as LFA, has emerged and shows tremendous collapsing force, and is already being used by attackers to flood and congest network critical links [20]. Hong et al. [21] proposed an SDN-assisted Slow HTTP DDoS attack defense method that can detect and mitigate Slowloris and Slow HTTP POST attacks in SDN.

In Reference [22], Yue et al. proposed M-DoS attack and S-DoS attack, respectively, representing the DoS attack with multiple flow entries (M-DoS) to exhaust the Ternary Content-Addressable Memory resource of the switch and DoS attack with a single well-designed entry (S-DoS) to overwhelm the target link then further impacting the controller. For these two kinds of attacks, they extracted six characteristics of the flow table, and used BP neural network to construct a classifier to distinguish the attack flow from normal flow.

2.2. Methods for Detecting DDoS Attacks. DDoS attack detection is principally divided into two aspects: threshold- and feature-based detection. Threshold-based detection mainly detects whether some specific indexes, such as entropy, exceed the threshold to determine whether the network is under attack. This detection method is ordinary and effortless to implement and owns supernal real-time performance. However, the detection rate of this method is tremendously affected by the dynamic changes of the network. How to find a suitable detection index is the master key to threshold-based detection. Yu et al. [23] believed that attacks are cyclic and periodic; thus, they raised a detection method based on "time to live" (TTL). For each switch loop, the number of data packets in the TTL is counted, and the data packets in the switch are observed as to whether they have periodicity through frequency domain transformation. Although this detection method allows for faster detection performance, it is affected with little hindrance in flash crowd scenes. Chou et al. [24] proposed a detection method based on correlation in SDN. The controller uses Spearman rank correlation and the round-trip time of each link layer discovery protocol frame to judge whether the network is under attack. Liu et al. [25] exploited a detection method combining generalized entropy and neural networks. The switch is determined whether there is an exception by detecting its generalized entropy value. For abnormal switches, a neural network extracts features further distinguishing whether DDoS attacks have occurred. Experiments showed that the method reduces the central processing unit (CPU) load of the controller and improves detection ability while ensuring detection accuracy. Feature-based detection methods generally extract and train the

features of normal traffic and attack traffic to classify unknown traffic through machine learning methods. Dalati [26] et al. proposed a real-time monitoring method, in which the controller periodically requests the switch to report information, and analyzes the traffic characteristics to determine whether it is malicious traffic. However, this method may bring great load pressure to the controller. Berezinski et al. [27] proposed the network anomaly detection technology primarily based on correlation entropy. Entropy metric is acceptable for detecting contemporary botnet malware mainly based on association anomaly patterns in the network.

More superior detection performance can be achieved in feature-based detection methods. However, they are obliged to perform feature extraction and train a large amount of data, leading to large detection calculations and a certain delay in feature extraction and classification. In view of the possibility of a single point of failure of the controller, Kirutika et al. [28] proposed an external monitoring system by using the random forest algorithm to monitor the behavior of the controller and calculate the probability of the controller being attacked; they also proposed some metrics to improve the accuracy of the detection process. Long Short-Term Memory (LSTM) and Support Vector Machine (SVM) can achieve the higher detection rate [29, 30]. Beny et al. [31] proposed a hybrid convolutional neural network—long short-term memory model (CNN-LSTM) to detect slow DDoS attacks. They also established six characteristics for training (e.g., average number of packets per flow, average packet size per flow, packet change rate, flow change rate, average number of packets per second, and average number of bytes per second). According to the model classification results to determine whether the network is under attack, the hybrid CNN-LSTM model is better than standard learning models, such as multilayer perceptron (MLP) and support vector machine (SVM). In response to DDoS attacks in SDN, Pérez-Díaz et al. [32] proposed a flexible structure which executes six machine learning algorithms, including random forest, MLP, random tree, J48, REP tree, and SVM, to train the intrusion detection system in the architecture. They also used the DoS dataset of the Canadian Institute of Cybersecurity to evaluate their performance. The assessment results show that the system effectively mitigates attacks. Chonka et al. [33] proposed a method based on chaos theory to detect normal traffic from the attack traffic according to flow similarity. In this paper, the system based on neural network is used to detect abnormal traffic. Johnson et al. [34] proposed a fresh victimization detection system—artificial neural network. This paper is dedicated to studying two formula multilayer perceptron (MLP) and genetic algorithm (GA). DDoS detection adopts MLP formula and genetic algorithm as learning formula. Projection technology will provide higher accuracy and sensitivity than normal system. Said ElSayed et al. [35] utilized the Information Gain (IG) and Random Forest (RF) in order to analyze the most comprehensive relevant features of DDoS attacks in SDN. Cao et al. [36] proposed a detection method based on Spatial-Temporal Graph Convolutional Network (ST-GCN). It can sense the state of switches and input the network state into the spatiotemporal graph convolution network detection model by in-band sampling network telemetry (INT), and finally find the switch through which DDoS attack flow passes.

Threshold- and feature-based detection have their respective charm and deficiency. The dominating problem in detection is that the method based on entropy cannot simultaneously detect DDoS attacks against IP addresses and ports. By contrast, flash crowd is also a potential interference factor. Flash crowd refers to numerous legitimate requests for access under normal conditions, such as the Spring Festival travel and World Cup live broadcasts. We call this kind of vast legitimate requests in a short period of time as a flash crowd scene. Legal burst traffic is frailly judged as a DDoS attack because it comes into being semblable behavior pattern like an attack.

2.3. Methods for Defending DDoS Attacks. DDoS attack defense methods in SDN are mainly divided into two types, the intermediate network resource control method and the source-side restriction method. The intermediate network resource control method is that when the network sustains attack, the controller firstly uses global view to pull the traffic of the victim switch to the neighboring switch to keep the link unobstructed. At the same time, the method detects the data packets and forwards the legitimate data packets firstly. The resource control method can avoid the filter of legitimate flow rules due to detection errors, but as the attack traffic increases, adjacent switches will also face the danger of resource exhaustion. Using the idea of intermediate network resource control, Wang et al. [37] proposed an attack mitigation method called BWManager, which is based on bandwidth prediction using an adaptive condition score time-series model to predict the broadband utilization of each user. The model predicts on the basis of the user's historical data, and the system determines the priority of resource allocation on the basis of the difference between its actual usage rate and the predicted value. In terms of attack filtering, BWManager extracts features and uses neural network classifiers to determine which switch is under attack. Shang Gao et al. [38] proposed a defense architecture called FloodDefender to mitigate and filter malicious traffic. The architecture detects the attacks through frequency characteristics and has a low false-positive rate. Once an attack is detected, the defense method bypasses the victim switch to protect the link and then filters the attack traffic. At the same time, the switch flow table is cached to protect TCAM resources. Biao Han et al. [39] developed a collaborative DDoS attack detection mechanism, which consists of a coarse-grained flow monitoring algorithm on the data plane and a fine-grained attack classification algorithm on the control plane. A novel defense policy offloading mechanism is proposed, in which defense applications are dynamically deployed between controllers and switches to achieve fast attack response and accurate botnet location.

The source-side restriction method is to detect the flow through indicators such as the number of flow requests, and take measures such as restricting access to data packets that

are considered to be attacks to protect the security of the switch flow table. Although the source-side restriction method can effectively filter attack traffic, the false positive rate is comparatively high, and legitimate requests will be restricted. Aimed at the source-side restriction method, Muhammad Imran [40] proposed a system called DAISY for detecting and mitigating attacks. In terms of detection, DAISY sets thresholds for the number of flow requests, thresholds for the number of warnings, and thresholds for the number of blockings. A judgment is made that the flow considered suspicious is blocked and blocked for a longer time when considered a malicious flow. The blocking time interval gradually increases or decreases in accordance with the malicious communication behavior. In order to prevent the nodes in the network from being attacked by DDoS attack, the literature [41] proposed a defense mechanism based on hybrid entropy. The hybrid entropy-based defense mechanism first judges whether the node is abnormal by detecting the changes in the source IP address entropy value, destination IP address entropy value, and time interval of the node, and adopts corresponding restriction measures according to the malicious degree of the node. If there is obscure incorrect behavior, the forwarding is restricted. When the node has distinct inaccurate behavior, it will be banned immediately. Kandoi et al. [42] analyzed low-rate DDoS attacks on the control plane and data plane, and proposed a method to mitigate DDoS attacks by limiting data transmission rate.

Considering comprehensively, source-side restriction is the most effective and direct method, but it is necessary to ensure extremely high detection accuracy to prevent legal flow rules from being filtered. The main problem in the contemporary defense method is the lack of effective attack filtering mechanisms. Most of the existing features are for the flow table as a whole and cannot distinguish specific attack flow rules. Moreover, the combination of detection and filtering should be more reasonable and functional to reduce CPU consumption while filtering out attack flow in time.

To solve the problems of the existing detection methods, we propose a three-dimensional entropy detection method based on the features of the switch flow table, which avoids the trouble of flash crowd scenes and can be capable of detecting multiple patterns of attacks. In response to the problems in the attack defense method, we propose a neural network-based flow rule filtering method to complete the precise positioning and filtering of attack flow rules. Lastly, we combine lightweight detection based on flow table features in the data plane and fine-grained filtering based on neural network in the control plane to form a cross-plane DDoS attack defense architecture, which not only ensures efficient filtering of attack flow rules but also reduces resource consumption.

3. Defense Architecture Description

To solve the problems of detection and defense methods, we propose a cross-plane DDoS attack defense architecture. This section principally describes the defense architecture across three aspects: defense architecture design, detection methods, and defense methods.

3.1. Defense Architecture Design. Several defense methods and defense architectures exist in the existing research. However, most of them are deployed in the control plane with a global view, leading the controller to access the data of switch for each detection. To ensure real-time detection, the controller needs to poll the switch frequently, bringing about huge communication overhead between the switch and the controller, even the southbound interface link is blocked. In addition, the controller firstly accesses the switch information and then performs detection function, which renders a high detection delay. To ensure real-time detection and reduce the communication overhead of the southbound interface, we deploy coarse-grained detection in the data plane with the purpose of separating detection and defense mechanism. Once the data plane detects the attack, the control plane initiates the defense procedure. Literature [43] pointed out that the existing OpenFlow switches have one or more CPUs. These processors have vigoroso computing power and abundant unused resources, making it possible to deploy detection methods in the data plane. On this basis, we design a cross-plane DDoS attack defense architecture. The architecture is mainly composed of data plane detection and control plane defense. Deploying coarse-grained entropy detection in the data plane guarantees real-time detection and reduces communication overhead. Fine-grained flow rule filtering based on neural network is deployed in the control plane to ensure that the attack flow rules can be filtered out in time and accurately after an attack is detected. The workflow of the defense architecture is shown in Figure 2.

Figure 2 illustrates that, in the face of a new forwarding request, the switch encapsulates packet-in message to the controller aiming to request flow rules installation, and the controller issues packet-out message to install flow rules which are stored in the flow table. In our defense architecture, the switch can directly perform real-time entropy detection on the flow table, thus reducing detection delay and communication overhead. Once the switch discovers that the entropy value of the flow table is abnormal, it sends an alarm message to the controller; then the controller accesses the flow table information of the switch and extracts fine-grained features, transmitting them to the neural network classifier consequently. In accordance with the output result of the classifier, the specific attack flow rule is located. The controller sends a Flow-Mod message to delete the attack flow rule and no longer installs the flow rule for the data packets from the host and port. At this point, the defense architecture has completed attack filtering, guaranteeing the smooth operation of switches and networks. The next subsection describes our defense architecture in detail through two aspects: detection method in the data plane and defense method in the control plane.

3.2. Detection Method in Data Plane

3.2.1. Entropy Detection. Detection methods for DDoS attacks in SDN are mainly divided into threshold- and feature-based detection. Considering the advantages and

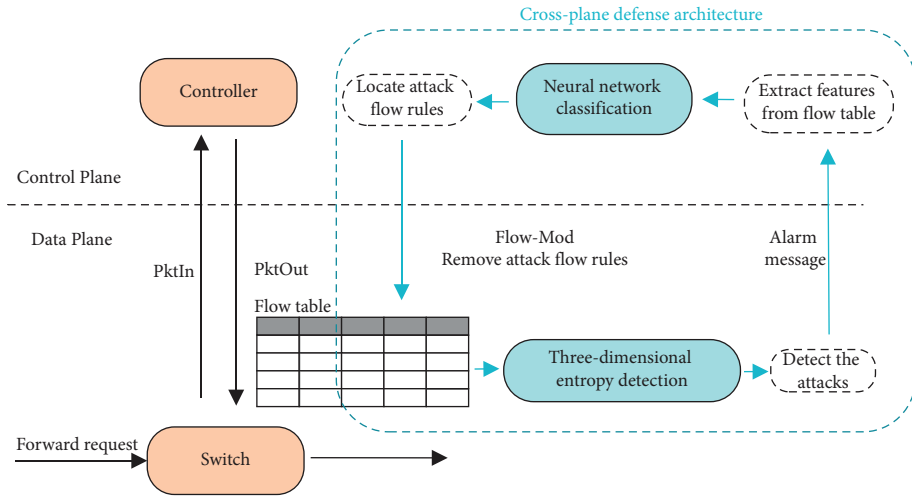

Figure 2: Defense architecture.

disadvantages of the two detection methods we analyzed in the Related Work section, we propose a three-dimensional entropy detection method. Entropy value is a measure of the degree of randomness and an index used to measure the maximum amount of information that the system can transmit. The higher the degree of randomness of a system and the more dispersed the information distribution, the higher the entropy value. The lower the degree of randomness, the lower the entropy value [44, 45]. When a DDoS attack is encountered, numerous data packets are sent to the victim, and several data packets with the same destination address cause a sudden drop in the entropy value; thus, the entropy value can well reflect the state of the network. However, detection based on a single entropy value has great limitations. For example, a detection method based on the entropy value of the IP address hardly detects DDoS attacks on the destination port. In addition, how to avoid the high false positive rate in the flash crowd scene is key to the detection method. To detect simultaneous DDoS attacks against IP addresses and ports and incidentally avert flash crowd scenarios, we compose the five table features of destination/source IP address entropy, destination/source port entropy, and packet size entropy. The three-dimensional entropy is used for attack detection, and the detection method is shown in Figure 3.

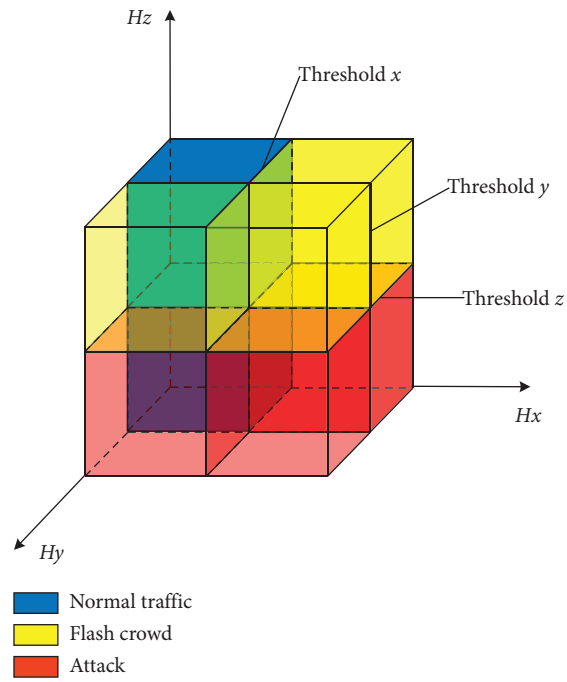

Figure 3: Three-dimensional entropy detection method.

Figure 3 shows that our detection method is divided into three dimensions. By using these three dimensions, unknown traffic can be divided into attack traffic, normal traffic, and flash crowd. *Hx* represents the source/destination IP address entropy, and the formula of *Hx* is

$$x = \sum_{i=0}^{k} P_{x1i} \log_2 p_{x1i} - \sum_{i=0}^{k} P_{x2i} \log_2 p_{x2i}. \quad (1)$$

In equation (1), $P_{x1i} = x_1 i/n$ corresponds to the probability of the i-th source IP address in the flow table, and $P_{x2i} = x_2 i/n$ corresponds to the probability of the i-th destination IP address in the flow table. When a DDoS attack against an IP address occurs, numerous data packets with forged source IP addresses flood into the same destination address, leading to a sudden increase in the entropy value of the source IP address and a sudden drop in the entropy value of the destination IP address. In turn, it leads to an increase in the *Hx* entropy value. *Hy* represents the entropy value of the destination/source port. The formula of *Hy* is

$$Hy = \sum_{i=0}^{k} P_{y1i} \log_2 p_{y1i} - \sum_{i=0}^{k} P_{y2i} \log_2 p_{y2i}. \quad (2)$$

In equation (2), $P_{y1i} = y_1 i/n$ corresponds to the probability of the i-th source port in the flow table, and $P_{y2i} = y_2 i/n$ corresponds to the probability of the i-th destination port in the flow table. In addition to forging the source IP address, DDoS attacks can guide the controller to install flow rules by changing the port number. When a DDoS attack against a

port occurs, numerous data packets with forged source ports flood into the switch and attack the same port of the host, thus consuming flow table resources. This occurrence leads to a sudden increase in the entropy value of the source port and a sudden drop in the entropy value of the destination port, subsequently increasing the *Hy* entropy value.

The combination of the entropy of *Hx* and *Hy* can simultaneously detect DDoS attacks against IP addresses and ports. However, how to avoid the false detection of a flash crowd traffic as an attack is one of the difficulties of this detection method. To this end, we propose packet size entropy *Hz*, whose formula is as follows:

$$Hz = -\sum_{i=0}^{k} P_{zi} \log_2 p_{zi}. \tag{3}$$

In equation (3), $P_{zi} = zi/n$ corresponds to the probability that the matching packet size is i in the flow table. Although flash crowd traffic and DDoS attacks are similar in behavior, they both increase numerous requests in a short period. Nevertheless, legitimate requests in a flash crowd scene have actual meaning; thus, the packet size of each data packet is different. On the contrary, to occupy numerous flow rules for a long time, DDoS attacks have to send several repeated, forged, and meaningless data packets, causing the packet size entropy *Hz* to drop sharply when a DDoS attack occurs. Hence, it is not significantly changed in a flash crowd scene.

Through the combination of the three dimensions of *Hx*, *Hy*, and *Hz*, the three-dimensional entropy value can simultaneously detect DDoS attacks against IP addresses and ports. At the same time, it can distinguish between attack traffic and legal burst traffic, thus efficaciously reducing the false positive rate in flash crowd scenes.

3.2.2. Adaptive Threshold. Threshold is a key parameter that directly affects false positive rate, false negative rate, and detection rate. In actual networks, network conditions are constantly changing, and it is difficult to adapt to various types of network traffic using fixed thresholds. If the network traffic is stable but the detector's threshold is high, then the false negative rate increases. If the network traffic is flexible but the threshold is low, the detection rate becomes low. To prevent the excessively high false negative rate and false positive rate due to improper threshold setting and to accommodate the dynamically changing network preferably, this paper adopts an adaptive threshold method to detect the entropy value. We define a sliding window with a size of T and a step size of β. The threshold $H(i)$ corresponding to the i-th detection window is

$$H(i) = \mu(i) + 3\sigma(i). \tag{4}$$

In equation (4), $\mu(i)$ represents the mean value of entropy in the i-th detection window, and $\sigma(i)$ represents the standard deviation. In the network, normal data obey the normal distribution. According to statistical theory, the mean plus or minus three standard deviations contains 99% of the distribution, and the detection method regards the values exceeding this interval as outliers [46]. The high confidence interval composed of the mean and standard deviation can effectively reduce the false negative rate and false positive rate. The small step size β of the sliding window ensures that the threshold can adapt to a dynamically changing network, and the larger window size T prevents the threshold from becoming volatile with network changes. Once the detected entropy value exceeds the threshold, that is, when the switch is under DDoS attack, the update of the threshold is stopped, thus avoiding the formation of an excessively high threshold due to the appearance of attack traffic.

In summary, this section proposes a detection method based on three-dimensional entropy, which can detect DDoS attacks with forged IP addresses and ports without being affected by flash crowd scenes.

An adaptive threshold is also designed to adapt to a dynamically changing network. In addition, the switch can directly use the information in the flow table to process and calculate the entropy value, thereby reducing communication consumption and improving the real-time performance of detection. Moreover, detection based on the entropy value is easy to implement and thus is suitable for attack detection.

3.3. Defense Method in Control Plane

3.3.1. Defense Features. Whether an attack can be accurately detected in real-time is the first step of the defense architecture. How to effectively filter out attack traffic and ensure the smooth operation of the network is pivotal. Most existing defense methods lack an efficient filtering mechanism. Some methods detect an attack but fail to delete the flow rule, and some methods delete attack flow rules while deleting legitimate flow rules. To this end, we present a DDoS defense method based on flow features, propose four fine-grained flow features, and use neural networks to learn and train the extracted features through classification results to position the attack flow rule. Lastly, we filter out the flow rules considered as malicious to complete the attack defense.

At present, there is much literature focusing on the defense against DDoS attacks through feature extraction. For example, literature [37] used features extraction mostly for short flow, proportional growth rate, and entropy value, which are only used for the overall flow table of the switch. These features are not designed for a single flow rule and cannot make the classifier distinguish which flow rules belong to attack flow rules. Thus, through the study of DDoS attack traffic, we propose four features for a single flow rule: duration time of the flow rule, number of packets matched by the flow rule n_packet, and the relative dispersion of matching bytes and idle_age. The content and meaning of the four features are described below.

(1) Duration. On the basis of the existing literature analysis, we found that 0.1% of the traffic duration can reach 200 s, and about 80% traffic duration is about 10 s [47]. However, in order to exhaust the flow table resources, the DDoS attack in data plane will uninterruptedly send attack data packets to ensure that the attack flow rules are not deleted. Therefore,

duration can be one of the characteristics of distinguishing flow rules. When a DDoS attack occurs, the duration of the attack flow rule will aggrandize continuingly, whereas the normal flow rule stabilizes at 10 s and below.

(2) n_packet. In order to occupy the flow rules with minimal resource consumption, DDoS attackers will periodically send packets to the victim, and the interval between packets is slightly shorter than the timeout period. Timeout is a way of switch resource management. The timeout is divided into hard timeout and soft timeout. When the flow rule exceeds the soft timeout and does not match the data packet, the switch will send a request to delete the flow rule to the controller. When the flow rule exceeds the hard timeout and no packet is matched, the switch directly deletes the flow rule. However, normal traffic is random and sends numerous packets constantly; thus, the number of packets matched by the flow rule n_packet can be used as an apparent detection feature. When a DDoS attack occurs, the n_packet under the attack scene is diminutive and fixed at a certain number, whereas the n_packet under a normal scene is prodigious and unstable.

(3) The Relative Dispersion of Matching Bytes. In addition to the number of packets, the size of the packets under an attack scene is significantly different from the normal scene. To occupy the flow table resources with the minimum resources, the attacker sends plenty of packets, which have no practical meaning and have no changes. However, the size of packets sent by a normal user varies in accordance with the practical needs of the user. Therefore, we propose packet variance as one of the features that can commendably describe the degree of dispersion of the packet size (RDMB). The calculation formula for RDMB is

$$\text{RDMB} = \frac{\sum_{i=1}^{N} (X_i - \mu)^2}{N}. \tag{5}$$

In equation (5), N is the number of packets matched by the flow rule, $X_i\{i = 1, \ldots, N\}$ is the size of each packet, and μ is the mean value of the packet size in the detection window. When a DDoS attack occurs, the attacker periodically sends forged packets of the same size to occupy the flow table resources. Thus, the RDMB of the attack flow rules always remains low, whereas that of the normal flow rule maintains a higher value.

(4) The Relative Dispersion of idle_age. We call the time between two data packets in the traffic the idle time or the packet sending interval idle_age. Through our analysis of the attack traffic, we find that to use the smallest resource to occupy the largest flow table space, the sending interval idle_age of the attack packet is periodic and slightly less than the timeout period. The idle_age of the normal packet varies in accordance with the practical needs of users. Under normal scenes, multiple packets can be sent in a short time, or any packet cannot be sent for a long time. Therefore, we design the relative dispersion of idle_age (RDIA) as our classification feature. RDIA is calculated as follows:

$$\text{RDIA} = \frac{\sum_{i=1}^{M} |T_i - \lambda|}{M}. \tag{6}$$

In equation (6), M represents the number of idle_age in the window function, $T_i\{i = 1, \ldots, M\}$ represents the time of each idle_age, and λ represents the mean value of idle_age. When sending a DDoS attack, the packets are periodic and single, and their RDIA is always kept low; in contrast, packets under normal scenes have practical meaning and randomness, and their RDIA will remain a higher value.

3.3.2. Improved Neural Network Model. After the neural network has used the above four features for learning, the flow rules in the flow table can be divided into attack and normal flow rules. However, there are considerable classifiers, each with its own set of strengths and weaknesses. We need to select a suitable classifier according to the actual situation. The naive Bayes algorithm is one of the most widely used classification algorithms, and it demonstrates exceptional accuracy when the attributes of the dataset are independent of each other. However, the algorithm itself is relatively ordinary. The classification effect is poor when the attributes of the dataset are related to each other. K-nearest neighbor can effectively reduce training costs, but its computational complexity is large; thus, the effect on sample classification must be eliminated in advance. SVM is a commonly used supervised learning method with high classification efficiency. However, it is sensitive to missing data and is hardly implemented in the face of a large-scale training dataset. Random forest uses an ensemble algorithm, which has a satisfactory classification effect on most data. However, when the number of decision trees is large, it consumes considerable calculation time and space, while appearing tremendous noise it readily falls into overfitting. Backpropagation (BP) neural networks are currently the most usual forms of neural networks because they have strong nonlinear mapping ability. XGBoost is an optimized distributed gradient enhancement library designed to bring efficient, flexible, and portable performance, but XGBoost algorithm is complicated to adjust parameters, and it is not suitable for processing ultra-high dimensional feature data. Adaboost is an iterative algorithm whose core idea is to train different classifiers (weak classifiers) for the same training set, and then combine these weak classifiers to form a stronger final classifier (strong classifier). However, it is sensitive to abnormal samples, and abnormal samples will get a higher weight, which will affect the final performance.

BP neural networks can work normally even when the system is locally damaged and have strong fault tolerance and generalization performance [31, 48]. A comparison of the false negative rate, false positive rate, and detection accuracy of the above five classification algorithms reveals that the performance of the BP neural network best fits our experiments. The results are shown in Table1. Thus, we select circumspectly the BP neural network to train our attribute sets. The BP neural network model is shown in Figure 4.

In Figure 4, the four features we proposed correspond to four input layer neurons (v_0, v_1, v_2, v_3). The classifier results

Table 1: Comparison of classification models.

Model	P_D (%)	P_{FN} (%)	P_{FP} (%)	TN (epochs)
SVM	97.7	2.5	3.2	101
Random forest	98.3	2.2	2.3	298
Naïve Bayesian	97.7	5.9	5.8	119
KNN	98.1	3.1	3.3	223
XGBoost	98.7	1.8	2.3	98
Adaboost	98.9	2.1	1.9	65
IBP	99.4	0.5	1.4	15

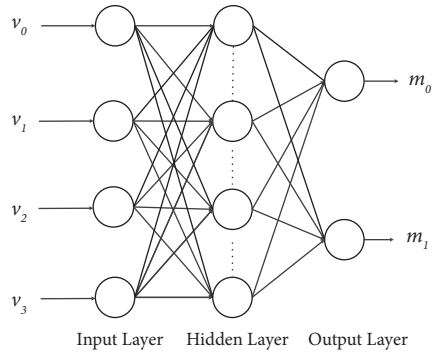

Figure 4: BP classification model.

correspond to the two output layer neurons (m_0, m_1), which represent normal traffic and DDoS attack traffic, respectively. The Ramp function will make a minority of neurons and may never be activated, resulting in the corresponding parameters never being updated. The zero point of the Threshold function is not differentiable, and the derivatives of other parts are all 0. The Sigmoid activation function was chosen for the following reasons: the Sigmoid function has an output range of 0 to 1, it normalizes the output of each neuron and takes the predicted probability as the output. Since the probability ranges from 0 to 1, the Sigmoid function is suitable, the function is differentiable which means that we can find the slope of the Sigmoid curve at any two points. In terms of the activation function, we found that compared with the Ramp function and the Threshold function, the detection accuracy is the best when we use the Sigmoid function.

In the experiment, we also found certain shortcomings of the BP neural network (e.g., long training time, easily falls into local minima). These shortcomings are mainly due to the fact that the convergence speed will decrease with the increase of training times, and the learning rate setting is unreasonable. The back propagation neural network uses the gradient descent method to adjust the weight parameters and reduce the training error. However, when the objective function is complex, the gradient descent method will cause "sawtooth phenomenon" and reduce the efficiency of the BP algorithm. Therefore, the momentum term is introduced to comprehensively adjust the weight. The back propagation neural network looks for the extreme point according to the maximum gradient direction of error decline. Every time it searches for the low point, the neural network will adjust the parameters and continue to search along the gradient until the lowest point is found. The distance to go in each search is the step size, also known as the learning rate. However, when the local lowest point appears in the network and the learning rate is too low, all solutions in the local search range of the neural network point to the local lowest point, and the neural network falls into the local minimum. To solve these problems, we partially optimized the BP algorithm by increasing the momentum term to enhance the convergence speed and by adaptively adjusting the learning rate to avoid the problem of minuscule learning rate.

(1) Increase Momentum Term. The slow learning speed of BP neural networks is mainly due to the explicit occurrence oscillation phenomenon when using the gradient descent method to adjust the weight. To this end, we introduce a momentum term to adjust the weights comprehensively, thereby accelerating the convergence of the neural network. The weight adjustment formula is

$$\begin{cases} \Delta M(t+1) = (1-\varphi)\eta\delta X + \varphi\Delta M(t), \\ \Delta\beta(t+1) = (1-\varphi)\eta\delta + \varphi\Delta\beta(t). \end{cases} \tag{7}$$

In equation (7), M is the weight coefficient matrix of a certain layer, t is the number of training, $\varphi \in (0, 1)$ is the momentum coefficient, η is the learning rate, δ is the output error of a certain layer, X is the input vector of a certain layer, and β is the certain layer of the threshold coefficient matrix. Through the above formula, we introduce a momentum term and comprehensively adjust the weight and threshold in accordance with the fluctuation of the error surface to reduce the effect of oscillation and accelerate the convergence of the neural network.

(2) Adaptive Learning Rate. The learning rate is used to control the learning progress of the model and determine whether the objective function can converge to the minimum at an appropriate time. Learning rate is a key parameter of the BP neural network and is also known as the step size. The fixed step length likely causes problems, such as slow convergence speed, excessive training time, and overadjustment, in the later stage. For this reason, we adopt an adaptive adjustment method to adjust the step length η. The adjustment formula is as follows:

$$\eta(t+1) = \begin{cases} \dfrac{\eta(t)}{\lambda}, & E(t+1) < E(t), \\ \lambda\eta(t), & E(t+1) > E(t). \end{cases} \tag{8}$$

As shown in equation (8), we introduce a scale factor $\lambda \in (0, 1)$ to adjust the step size, where $\eta(t+1)$ and $\eta(t)$ represent the learning rate at time $t+1$ and time t, and $E(t+1)$ and $E(t)$, respectively, represent the mean square error at time $t+1$ and time t. When the error increases, the learning rate is reduced, and the weight is corrected. When the error decreases, the learning rate is increased to accelerate the convergence.

In this section, to locate and filter the attack flow rules accurately, we propose four fine-grained features for a single

flow rule. At the same time, we optimize the neural network to improve the convergence effect. After learning and classification by the classifier, the flow rules in the flow table can be classified once a DDoS attack is detected. The attack flow rules are filtered out and no longer installed.

4. Experiment and Analysis

In this section, we test our cross-plane defense architecture through hardware test-bed and compare it with other methods.

4.1. Experimental Environment. The experiment topology is composed of five hosts, three switches, and a controller. The host system is Ubuntu 19.1 and has a 4G running memory, and its processor is Intel Xeon E 2224 @ 3.5 GHz. The switch model is Centec V350-48T4X, which supports OpenFlow 1.3 protocol and high-speed forwarding. The controller is Ryu version 4.31, and the host system running the Ryu controller is Ubuntu 19.1, which has 32 G of running memory. Its processor is Intel Xeon E 2224 @ 3.5 GHz. The topology is shown in Figure 5.

In our experiments, host h1 acts as an attacker and sends attack traffic to host h4, and switch sw1 is the attack target, which can store up to 1500 flow rules [3–5], h2 and h3 send background traffic, including flash crowd to h5. To simulate the actual situation as much as possible, we replay the dataset collected in the real network on the host to simulate the real networks. This experimental method has been commonly used to replay datasets on the basis of simple hardware topology [12, 31, 49].

The background traffic is mainly composed of the *FIFA World Cup dataset* and the *CAIDA dataset*. The *FIFA World Cup dataset* can generate a flash crowd scene, which contains all applications received by the venues during the World Cup [50]. The *CAIDA dataset* collects and analyzes global Internet data, including the flow of various data types. The *CAIDA dataset* [51] is used to generate normal scenes.

DDoS attacks are mainly generated by the *CICIDS2017 dataset* and the *HogzillaIDS dataset*. The *CICIDS2017 dataset* [52] contains anonymous traffic tracking for approximately five days from July 2nd to July 7th, 2017. Attacks include brute force DDoS, web attacks, and FTP attacks. The *HogzillaIDS dataset* is an intrusion detection system for network anomaly detection, supported by HBase, GrayLog, libnDPI, and Snort [53]. The main protocols of DDoS attack traffic are the TCP and ICMP protocols, of which the ICMP protocol accounts for 92%, and TCP data packets account for 7%.

4.2. Attack Effect. To verify the effect of the DDoS attack on the SDN data plane, we took the switch sw1 as the attack target and sent a DDoS attack to the host h4 through the host h1 to occupy the TCAM resources of the switch sw1. At the same time, hosts h2 and h3 send background traffic to h5. To validate the efficiency of the attack, we test the number of flow rules of switch sw1 and the connection success rate of host h5. Through these two indicators, we can observe the occupancy of the flow table and the effect of the attack on the network. In the experiment, we set the window size to 30 s, and the attack is launched after 10 s. The experimental results of the number of flow rules and connection success rate are shown in Figures 6 and 7.

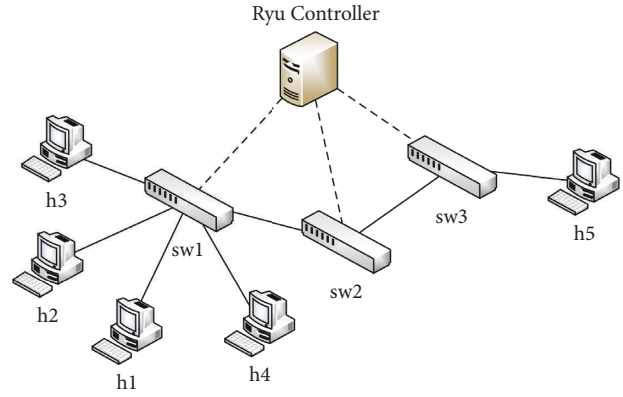

Figure 5: Experiment topology.

Figure 6 reveals that the number of flow rules in the switch is maintained at approximately 400 under normal circumstances. The DDoS attack is launched after 10 s, and numerous attack packets guide the controller to install flow rules. The number of flow rules in the flow table increases rapidly until it reaches the peak of the flow table capacity, which is 1500. When the flow table capacity of switch sw1 is full, sw1 will have no resources to install flow rules for other legitimate requests.

Figure 7 shows that under normal circumstances, the connection success rate of messages that hosts h2 and h3 send to host h5 through switch sw1 is 100%. The DDoS attack is launched after 10 s. After 15 s, the flow table is gradually filled, and the connection success rate of host h5 begins to drop. After 18 s, the flow table of sw1 is basically full, and the connection success rate of h5 remains at a low value because some legal flow rules exist before the flow table is full; thus, communication can still be performed. When normal forwarding is completed, these legal flow rules without subsequent data packets will be deleted due to the SDN timeout mechanism, and new legal flow requests cannot be installed because the flow table is filled up, resulting in the h5 connection success rate gradually reducing to 0; moreover, the attacker achieves the purpose of denial of service.

Through the number of flow rules in sw1 and the connection success rate of h5, we can see that DDoS attacks on the data plane have significant attack effects and destructiveness.

4.3. Detection Effect. To verify the performance of the detection method, we collect 200 three-dimensional entropy values of the flow table, of which 50 are collected under normal conditions, 50 are collected in a flash crowd scene, 50 are collected from DDoS attacks against IP addresses, and 50 are collected from DDoS attacks against port numbers. The three-dimensional entropy of 200 times is shown in Figure 8.

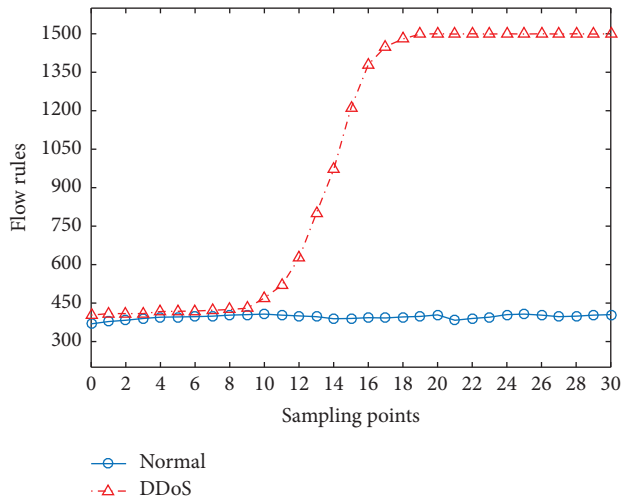

Figure 6: Number of flow rules.

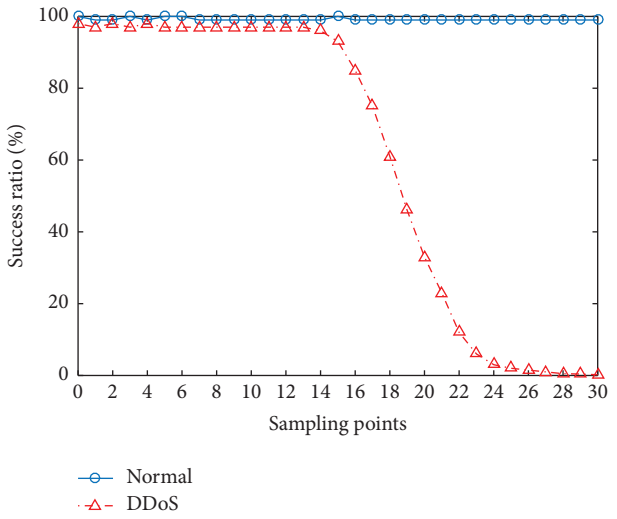

Figure 7: Connection success rate.

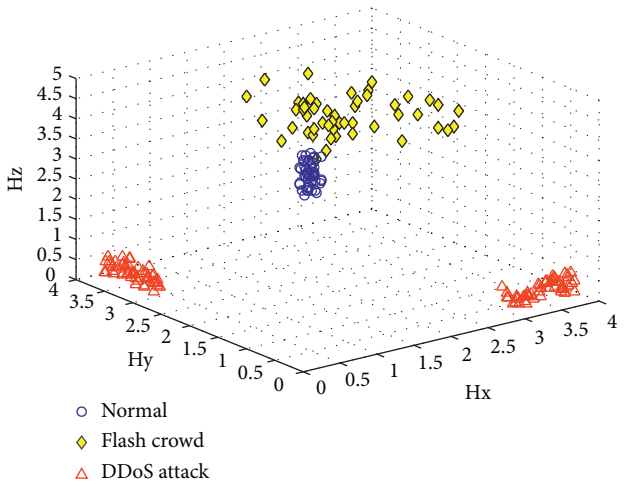

Figure 8: Three-dimensional entropy value.

Figure 8 reveals that the *Hx* and *Hy* of samples under normal flow are maintained at a low state (between 0.3 and 0.7), whereas *Hz* is maintained at a high value (between 4 and 5). The *Hx*, *Hy*, and *Hz* of the flash crowd scene are kept at a high value (*Hx* and *Hy* are kept between 2 and 4, and *Hz* is kept between 3 and 4). For the samples under the DDoS attack against the IP address, its *Hx* value is high (between 3 and 4), whereas the *Hy* and *Hz* entropy values are low (both below 1). Similarly, for the samples under the DDoS attack against the port number, its *Hy* entropy value is high (between 2.9 and 4), whereas the *Hx* and *Hz* entropy values are low (both below 0.6). The three-dimensional entropy value has a commendable distinction between the two modes of DDoS attacks and flash crowd scene.

To reflect the detection effect of the three-dimensional entropy value, we test the accurate detection probability (P_D), false negative probability (P_{FN}), false positive probability (P_{FP}), and detection time. In addition, we prove the superiorities of our proposed method by comparing with other methods. We implement and compare four other existing methods under the same experimental conditions [25, 28, 31, 54]. The results are shown in Table 2.

Table 2 demonstrates that the naive Bayesian and generalized entropy algorithms are relatively simple with low detection accuracy, high false positive rate, and false negative rate; however, it has short detection time. The CNN-LSTM model extracts features and uses a hybrid convolution model for classification. Its detection accuracy is higher, and the false positive and false negative rates are lower. Random forest is a type of deep learning with supernal detection accuracy and potency. Given that CNN-LSTM and random forest use machine learning methods, the algorithm complexity is high, and the detection time is long. After comparison, we find that the detection method of the three-dimensional entropy value can ensure high detection accuracy, extremely low false positive rate and false negative rate; the detection time is only 0.51 s. At the same time, the switch can straightforwardly exploit the flow table for detection because the detection method is deployed in the data plane. Thus, communication consumption and delay are reduced, and low detection time is maintained, making the switch suitable for the detection part of the defense architecture.

4.4. Defense Effect. First, we evaluate the performance of different classification models, including improved back propagation (IBP). We select 1000 groups of features as the training set and 400 groups of features as the test set in normal as well as attack scenarios. The maximum number of training target, iterations, and learning rate are set to 0.0001, 10000, and 0.001. We test the accuracy, false positive rate, false negative rate, and training times under the same conditions. The test results are shown in Table 1.

Table 1 shows that the improved IBP classification algorithm is appropriate for our method; it can be seen that the accuracy, false positive rate, and false negative rate of IBP classification model achieve a favorable consequence. At the

Table 2: Comparison of detection methods.

Approach	P_D (%)	P_{FN} (%)	P_{FP} (%)	Detection time
Generalized entropy [25]	96.2	6.2	6.3	0.91
Random forest [28]	97.7	1.1	3.3	3.22
CNN-LSTM [31]	97.1	5.9	5.8	1.89
Naïve Bayesian [54]	96.1	7.5	7.2	1.11
Our approach	98.9	0.5	1.4	0.51

same time, the improved IBP algorithm has the least number of iterations and the fastest convergence speed.

When a DDoS attack occurs in the switch, the switch sends an alarm message to the controller, and the controller immediately filters the flow table in the attacked switch. The filtering effect of the attack flow rules mainly depends on the selection of features. To verify the classification effects of the four features mentioned in the defense method, we conduct experiments on the four proposed features.

We collect the duration of 50 sets of flow rules in normal and attack scenes with a window size of 20 s. The experimental results are shown in Figure 9.

Figure 9 shows that within 20 s, the duration of the normal flow rule is maintained at 10 s and below; the duration of each flow rule is random because every forwarding rule of its normal traffic has practical significance. When the forwarding is completed, the corresponding flow rules are deleted through timeout mechanism. The duration of the flow rule of the attack traffic is kept at 20 s, indicating that the flow rule always exists, and its duration is not random. As shown in Figure 9, the duration of the flow rule can distinguish attack traffic from normal traffic. With the same window size, we collect 50 sets of *n_packet* under attack and normal scenes. The experimental results are shown in Figure 10.

Figure 10shows that the *n_packet* under normal scene is between 10 and 30, and it has great randomness. Meanwhile, the *n_packet* under attack scene remains at two, and no quantitative change occurs between different flow rules. This result also verifies that attackers often use the least consumption to occupy the largest resources (the timeout period is 10 s, and sending two packets in 20 s just to ensure that the flow rules continue to exist). As shown in Figure 10, the *n_packet* can distinguish normal traffic from attack traffic.

With the same window size, we collect 50 sets of RDMB of flow rules under normal and attack scenes. The experimental results are shown in Figure 11.

Figure 11 shows that in a normal scene, the RDMB maintains a high value (between 190 and 240) because the size of the matching packet under normal circumstances depends on the user's access needs; thus, it has substantial randomness. The DDoS attacker only sends forged data packets to fill the flow table. These data packets have an identical size and are of no practical significance. As a result, the RDMB of the attack flow rule in the figure is maintained below 1. As shown in Figure 11, the degree of dispersion of the packet size can distinguish normal traffic from attack traffic.

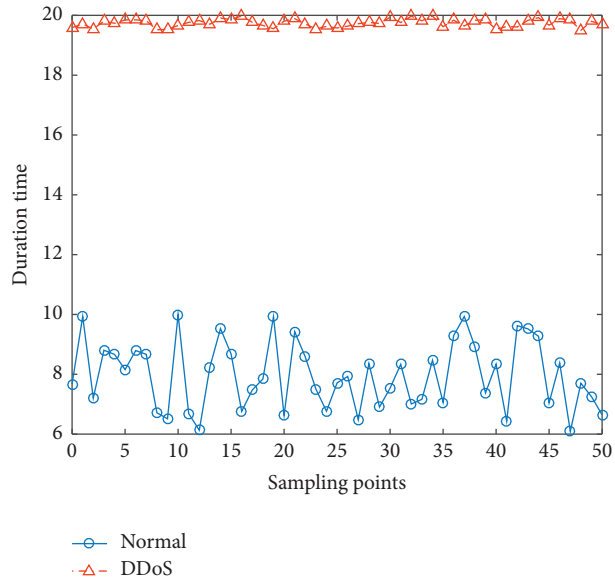

Figure 9: Duration of flow rules.

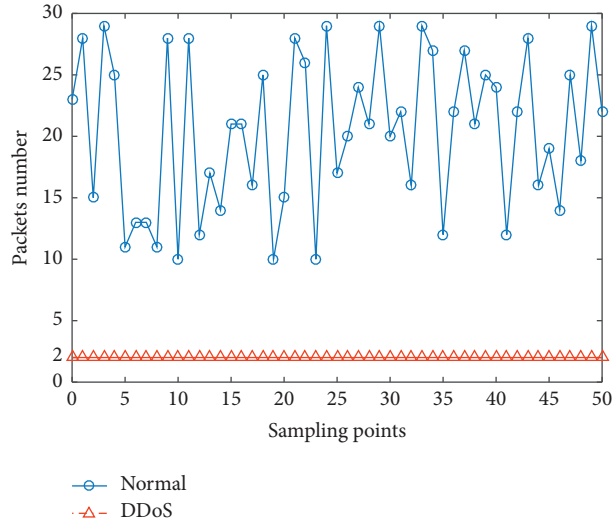

Figure 10: Number of matching packets.

Lastly, we collect 50 sets of RDIA in normal and attack scenes with a 20 s window. The experimental results are shown in Figure 12.

Figure 12 shows that under normal circumstances, the RDIA of the flow rule maintains a high value (between 12 and 18) because the *idle_age* of the legal flow rule depends on the specific access requirements of the users, which have high randomness. Thus, the RDIA becomes larger. To fill the flow table with minimal consumption, the *idle_age* of attack packets are fixed and a bit smaller than the timeout period. Therefore, the RDIA under the attack scene is maintained below 0.3. As shown in Figure 12, RDIA can distinguish legal flow rules from attack flow rules.

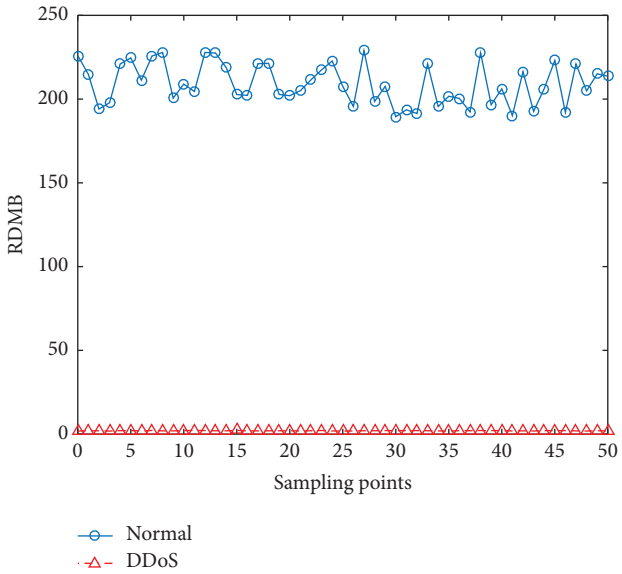

Figure 11: RDMB of flow rules.

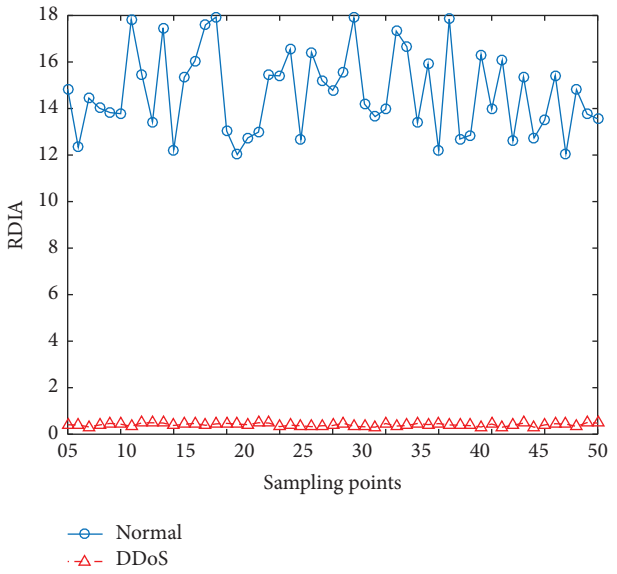

Figure 12: RDIA of flow rules.

4.5. Whole Architecture Defense Effect. To verify the defense effect of our proposed defense architecture, we test the occupancy of the flow table, the success rate of the host connection, and the CPU occupancy of this architecture. The architecture is also compared with other defense methods that have appeared in recent years (mentioned in Section 2). In the experiment, we launch the attack in 10 s. The experimental results are shown in Figure 13.

Figure 13shows that without any defensive measures, the number of flow rules in the flow table gradually increases with the launch of the attack until it reaches the upper limit of the flow table (1500). When a BWManager is deployed, detecting attacks takes a long time (the number of flow rules has reached 1100) probably due to the use of

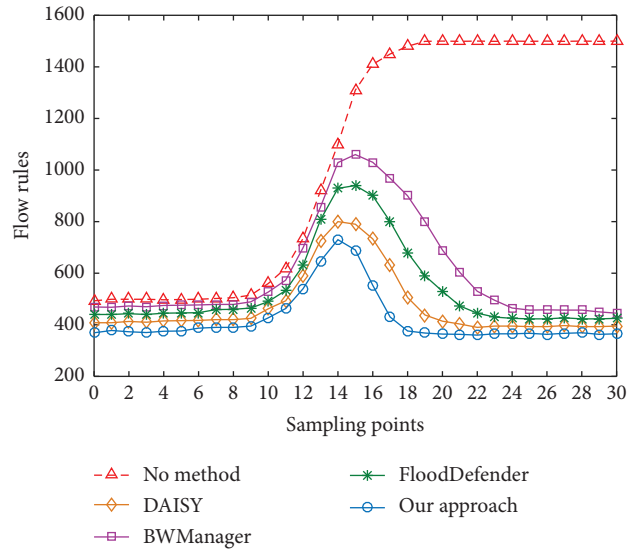

Figure 13: Number of flow rules.

predictive models for detection; moreover, there will be a certain delay. When an attack is detected, the BWManager performs feature detection and filtering on the entire table, which may reduce the filtering speed. When a FloodDefender is deployed, the frequency characteristics are used for detection, and the attack is judged on the basis of the changes in the frequency of matching flow rules, leading to a certain delay in detection. DAISY is based on threshold detection. Once a suspicious flow rule is found, it is blocked within a short period, leading to a shorter response time and faster filtering speed. Our method is also based on threshold detection. The data plane detects the changes of the three-dimensional entropy in real-time to ensure a fast response speed. Once an attack is found, the controller deletes the flow rules that the classifier considers to be malicious so that the flow table can quickly return to normal.

From the connection success rate in Figure 14, we can see that when no measures are taken, the host connection success rate decreases as the flow table is full until it cannot be connected. After deploying our method, the smooth connection of the host is ensured because the attack flow rules can be cleared in time when the flow table is first attacked; moreover, the flash crowd scene is considered to avoid the problem of excessive false positive rate. The other three methods do not consider the flash crowd scene too much, leading to the accidental deletion of legal burst traffic, eventually resulting in a decrease in the connection success rate; especially in DAISY, once suspicious traffic is found, it is immediately blocked, leading to a high false positive rate.

From the CPU occupancy in Figure 15, we can see that our cross-plane defense architecture only consumes the computing resources of the switch during detection, and the amount of calculation for entropy detection is small. When the switch detects an attack, the controller combines with the neural network for traffic filtering. After filtering, it continues to enter the state of entropy

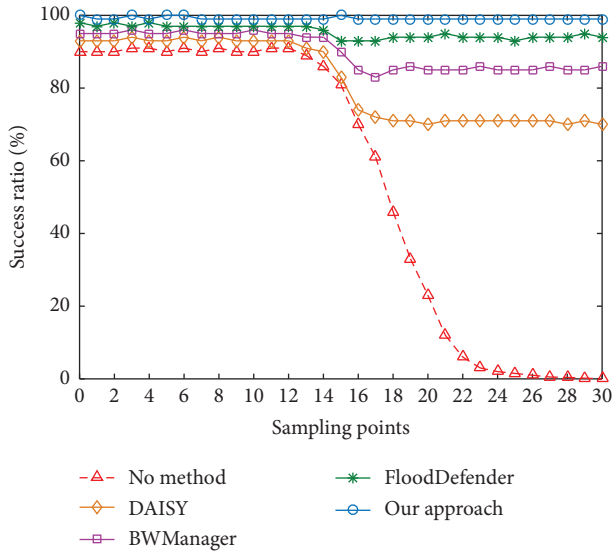

Figure 14: Success rate of the host connection.

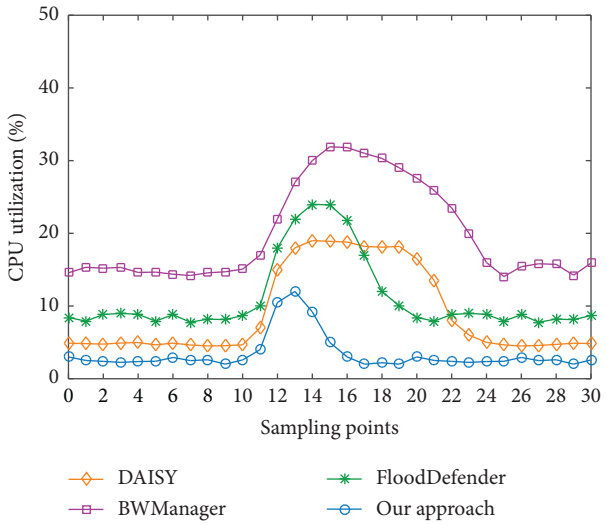

Figure 15: CPU utilization.

detection; thus, the CPU usage is small (below 12%). DAISY also applies threshold detection, but once an attack occurs, it will perform the function of blocking and defense. If the attack is detected again, the blocking time is extended, resulting in a longer CPU usage time. BWManager and FloodDefender have slightly higher CPU usage due to the complexity of their detection and defense methods. We summarized the response time, host connection success rate, and CPU usage of the four methods, as shown in Table 3.

As can be concluded from Table 3, the response time of the cross-plane defense method is reduced by 1 to 2.3 s compared with the existing method, the host connection success rate is increased by 10% to 30%, and the average CPU usage is increased by 5% to 15%. Our method has certain advantages in terms of reaction time, defense effect, and CPU occupancy.

Table 3: Summary of four defense methods.

Method	Response time (s)	Host connection success rate (%)	CPU usage (%)
DAISY [40]	2.9	69	10
BWManager [37]	4.2	82	20
FloodDefender [38]	3.8	90	11
Our method	1.9	99.8	5

5. Conclusion and Future Prospect

In this paper, we present a cross-plane DDoS attack defense architecture in SDN. We design a detection mechanism which takes advantage of three-dimensional entropy and implements a defense procedure by extracting flow feature for traffic filtering; the formed entropy value can simultaneously detect DDoS attacks against IP addresses and ports under a flash crowd scene. Extensive experiments show that this method has certain outstanding properties in terms of effectiveness, promptness, and resource conservation.

In future research, we will focus on the combination of traffic filtering and resource management strategies. We will also adjust the amplitude of flow table filtering through the occupancy of the flow table, use the idle resources of the other switches to mitigate the victim switch, and strive to minimize the time that the flow table is occupied.

Data Availability

The data used to support the findings of this study are available from the corresponding author upon request.

Conflicts of Interest

The authors declare that there is no conflict of interest regarding the publication of this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China (Nos. 62172418, U1933108, and 61802276), The Natural Science Foundation of Tianjin China (21JCZDJZ00830) the Scientific Research Project of Tianjin Municipal Education Commission (No. 2019KJ117), and the Fundamental Research Funds for the Central Universities of CAUC (No. 3122020076).

References

[1] Z. Li and W. Meng, "Mind the Amplification: Cracking Content Delivery Networks via DDoS Attacks," in *Proceedings of the Wireless Algorithms, Systems, and Applications. WASA 2021*, pp. 186–197, Nanjing, China, September 2021.

[2] H. Polat, M. Turkoglu, and O. Polat, "Deep network approach with stacked sparse autoencoders in detection of DDoS attacks on SDN-based VANET," *IET Communications*, vol. 14, no. 22, pp. 4089–4100, 2021.

[3] C. Zhang, G. W. Hu, G. L. Chen et al., "Towards a SDN-Based Integrated Architecture for Mitigating IP Spoofing Attack," *IEEE Access*, vol. 7, Article ID 22764, 2018.

[4] S. Misra, N. Saha, and R. Bhakta, "Traffic-aware rule-cache assignment in SDN: security implications," in *Proceedings of the 2020 IEEE International Conference on Communications Workshops (ICC Workshops)*, pp. 1–6, Dublin, Ireland, 2020.

[5] S. Xu, X. Wang, G. Yang, J. Ren, and S. Wang, "Routing optimization for cloud services in SDN-based Internet of Things with TCAM capacity constraint," *Journal of Communications and Networks*, vol. 22, no. 2, pp. 145–158, 2020.

[6] J. Xing, J. Cai, B. Zhou, and C. Wu, "A deep ConvNet-based countermeasure to mitigate link flooding attacks using software-defined networks," in *Proceedings of the 2019 IEEE Symposium on Computers and Communications (ISCC)*, pp. 1–6, Barcelona, Spain, June 2019.

[7] D. Kim, P. T. Dinh, S. Noh, J. Yi, and M. Park, "An effective defense against SYN flooding attack in SDN," in *Proceedings of the Information and Communication Technology Convergence (ICTC)*, pp. 369–371, Jeju, Korea, October 2019.

[8] I. Sumantra and S. Indira Gandhi, "DDoS attack detection and mitigation in software defined networks," in *Proceedings of the 2020 International Conference on System, Computation, Automation and Networking (ICSCAN)*, pp. 1–5, Pondicherry, India, July 2020.

[9] T. Y. Lin, J. P. Wu, H. Pei-Hsuan, S. Ching-Hsuan, and W. Yu-Ting, "Mitigating SYN flooding attack and ARP spoofing in SDN data plane," in *Proceedings of the 2020 21st Asia-Pacific Network Operations and Management Symposium (APNOMS)*, pp. 114–119, Daegu, Korea, September 2020.

[10] F. Hauser, M. Schmidt, M. Haberle, and M. Menth, "P4-MACsec: dynamic topology monitoring and data layer protection with MACsec in P4-based SDN," *IEEE Access*, vol. 8, Article ID 58845, 2020.

[11] V. Varadharajan and U. Tupakula, "Counteracting attacks from malicious end hosts in software defined networks," *IEEE Transactions on Network and Service Management*, vol. 17, no. 1, pp. 160–174, 2020.

[12] T. V. Phan and M. Park, "Efficient distributed denial-of-service attack defense in SDN-based cloud," *IEEE Access*, vol. 7, Article ID 18701, 2019.

[13] M. Du and K. Wang, "An SDN-enabled pseudo-honeypot strategy for distributed denial of service attacks in industrial Internet of Things," *IEEE Transactions on Industrial Informatics*, vol. 16, no. 1, pp. 648–657, 2020.

[14] R. Sanjeetha, S. Srivastava, A. Kanavalli, A. Pattanaik, and A. Gupta, "Mitigation of combined DDoS attack on SDN controller and primary server in software defined networks using a priority on traffic variation," in *Proceedings of the 2020 International Conference for Emerging Technology (INCET)*, pp. 1–5, Belgaum, India, June 2020.

[15] S. A. Latif, F. B. X. Wen, C. Iwendi et al., "AI-empowered, blockchain and SDN integrated security architecture for IoT network of cyber physical systems," *Computer Communications*, vol. 181, pp. 274–283, 2022.

[16] L. Fang, B. Zhao, Y. Li, Z. Liu, C. Ge, and W. Meng, "Countermeasure based on smart contracts and AI against DoS/DDoS attack in 5G circumstances," *IEEE Network*, vol. 34, no. 6, pp. 54–61, 2020.

[17] Y. Zhou, G. Cheng, and S. Yu, "An SDN-enabled proactive defense framework for DDoS mitigation in IoT networks," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 5366–5380, 2021.

[18] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks," *IEEE Communications Surveys & Tutorials*, vol. 15, no. 4, pp. 2046–2069, 2013.

[19] S. Shin, V. Yegneswaran, P. Porras, and G. Gu, "Avant guard: scalable and vigilant switch flow management in software-defined networks," in *Proceedings of the ACM SIGSAC Conf. Comput. Commun. Security*, pp. 413–424, Berlin, Germany, November 2020.

[20] J. Wang, R. Wen, J. Li, F. Yan, B. Zhao, and F. Yu, "Detecting and mitigating target link-flooding attacks using SDN," *IEEE Transactions on Dependable and Secure Computing*, vol. 16, no. 6, pp. 944–956, 2019.

[21] K. Hong, Y. Kim, H. Choi, and J. Park, "SDN-assisted slow HTTP DDoS attack defense method," *IEEE Communications Letters*, vol. 22, no. 4, pp. 688–691, 2018.

[22] M. Yue, H. Wang, L. Liu, and Z. Wu, "Detecting DoS attacks based on multi-features in SDN," *IEEE Access*, vol. 8, Article ID 104688, 2020.

[23] T. Yu, L. Yu, D. Chen, H. Cui, and J. Wang, "An SDN oriented loop detection mechanism based on TTL statistics," *China Communications*, vol. 17, no. 6, pp. 1–12, 2020.

[24] L. D. Chou, C. C. Liu, M. S. Lai, and K. C. Chiu, "Behavior anomaly detection in SDN control plane: a case study of topology discovery attacks," in *Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC)*, pp. 357–362, 2019.

[25] Z. P. Liu, Y. P. He, W. S. Wang, and B. Zhang, "DDoS attack detection scheme based on entropy and PSO-BP neural network in SDN," *China Communications*, vol. 16, no. 7, pp. 144–155, 2019.

[26] M. S. Dalati, W. Meng, and W.-Y. Chiu, "NGS: mitigating DDoS attacks using SDN-based network gate shield," *2021 IEEE Global Communications Conference (GLOBECOM)*, pp. 1–6, 2004.

[27] P. Berezinski, B. Jasiul, and M. Szpyrka, "An entropy-based network anomaly detection method," *Entropy*, vol. 17, no. 4, pp. 2367–2408, 2015.

[28] K. Kirutika, V. Vetriselvi, R. Parthasarathi, and G. S. V. Rao, "Controller monitoring system in software defined networks using random forest algorithm," in *Proceedings of the 2019 International Carnahan Conference on Security Technology (ICCST)*, pp. 1–6, Chennai, India, October 2019.

[29] M. Mittal, C. Iwendi, S. Khan, and A. Rehman Javed, "Analysis of security and energy efficiency for shortest route discovery in low-energy adaptive clustering hierarchy protocol using Levenberg- Marquardt neural network and gated recurrent unit for intrusion detection system," *Trans Emerging Tel Tech*, vol. 32, no. 6, 2020.

[30] C. Iwendi, K. Mahboob, Z. Khalid, Abdul Rehman Javed, M. Rizwan, and U. Ghosh, "Classification of COVID-19 Individuals Using Adaptive Neuro-Fuzzy Inference System," *Multimedia Systems*, vol. 28, pp. 1223–1237, 2022.

[31] B. Nugraha and R. N. Murthy, "Deep Learning-Based Slow DDoS Attack Detection in SDN-Based Networks," in *Proceedings of the 2020 IEEE Conference on Network Function Virtualization and Software Defined Networks (NFV-SDN)*, pp. 51–56, Leganes, Spain, November 2020.

[32] J. A. Pérez-Díaz, I. A. Valdovinos, K.-K. R. Choo, and D. Zhu, "A flexible SDN-based architecture for identifying and mitigating low-rate DDoS attacks using machine learning," *IEEE Access*, vol. 8, Article ID 155859, 2020.

[33] A. Chonka, J. Singh, and W. Zhou, "Chaos theory based detection against network mimicking DDoS attacks," *IEEE Communications Letters*, vol. 13, no. 9, pp. 717–719, 2009.

[34] K. Johnson Singh, K. Thongam, and T. De, "Entropy-based application layer DDoS attack detection using artificial neural networks," *Entropy*, vol. 18, no. 10, p. 350, 2016.

[35] M. S. E. Sayed, N.-A. Le-Khac, M. A. Azer, and A. D. Jurcut, "A flow based anomaly detection approach with feature selection method against DDoS attacks in SDNs," *IEEE Transactions on Cognitive Communications and Networking*, p. 1, 2022.

[36] Y. Cao, H. Jiang, Y. Deng, J. Wu, P. Zhou, and W. Luo, "Detecting and mitigating DDoS attacks in SDN using spatial-temporal graph convolutional network," *IEEE Transactions on Dependable and Secure Computing*, 2021.

[37] T. Wang, Z. Guo, H. Chen, and W. Liu, "BWManager: mitigating denial of service attacks in software- defined networks through bandwidth prediction," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1235–1248, 2018.

[38] S. Gao, Z. Peng, B. Xiao, A. Hu, Y. Song, and K. Ren, "Detection and mitigation of DoS attacks in software defined networks," *IEEE/ACM Transactions on Networking*, vol. 28, no. 3, pp. 1419–1433, 2020.

[39] B. Han, X. Yang, Z. Sun, J. Huang, and J. Su, "OverWatch: A Cross-Plane DDoS Attack Defense Framework with Collaborative Intelligence in SDN," *Security And Communication Networks*, vol. 2018, Article ID 9649643, 15 pages, 2018.

[40] M. Imran, M. H. Durad, F. A. Khan, and H. Abbas, "DAISY: a detection and mitigation system Against denial-of-service attacks in software-defined networks," *IEEE Systems Journal*, vol. 14, no. 2, pp. 1933–1944, 2020.

[41] N. M. Abdelazim, S. F. Fahmy, M. A. Sobh, and A. M. Bahaa Eldin, "A hybrid entropy-based DoS attacks detection system for software defined networks (SDN): a proposed trust mechanism," *Egyptian Informatics Journal*, vol. 22, no. 1, pp. 85–90, 2021.

[42] R. Kandoi and M. Antikainen, "Denial-of-service attacks in OpenFlow SDN networks," in *Proceedings of the 2015 IFIP/IEEE International Symposium on Integrated Network Management*, pp. 1322–1326, Ottawa, ON, Canada, May 2015.

[43] J. Mao, B. Han, Z. Sun, X. Lu, and Z. Zhang, "Efficient mismatched packet buffer management with packet order-preserving for OpenFlow networks," *Computer Networks*, vol. 110, pp. 91–103, 2016.

[44] Z. Abou El Houda, L. Khoukhi, and A. Senhaji Hafid, "Bringing intelligence to software defined networks: mitigating DDoS attacks," *IEEE Transactions on Network and Service Management*, vol. 17, no. 4, pp. 2523–2535, 2020.

[45] R. Li and B. Wu, "Bringing Intelligence to Software Defined Networks: Mitigating DDoS Attacks," *IEEE Transactions on Network and Service Management*, vol. 17, pp. 731–735, 2020.

[46] Y. Ding, "A special integral and goodness-of-fit test for multivariate normal distributions," *Statistics & Decisions*, vol. 36, no. 4, pp. 18–21, 2020.

[47] M. Prasath and B. Perumal, "Network attack prediction by random forest: classification method," in *Proceedings of the 3rd International Conference on Electronics and Communication and Aerospace Technology (ITNEC)*, pp. 647–654, Coimbatore, India, June 2019.

[48] T. A. Pascoal, Y. G. Dantas, I. E. Fonseca, and V. Nigam, "Slow TCAM exhaustion DDoS attack," in *IFIPI nternational Conference on ICT Systems Security and Privacy Protection*-Springer, Cham, New York, NY, USA, 2017.

[49] B. Yuan, D. Q. Zou, S. Yu, H. Jin, W. Z. Qiang, and J. A. Shen, "Defending against flow table overloading attack in software-efined networks," *IEEE Transactions on Services Computing*, vol. 12, no. 2, pp. 231–246, 2019.

[50] FIFA, "FIFA world cup 1998 dataset," 1998, http://ita.ee.lbl.gov/html/contrib/WorldCup.html.

[51] C. A. I. D. A. Datasets, "Anonymized Internet Traces," 2019, https://data.caida.org/datasets.

[52] C. I. C. Datasets, "IDS," 2017, https://www.unb.ca/cic/datasets/ids-2017.html.

[53] H. I. D. S. Datasets, "Hogzilla IDS," 2018, https://ids-hogzilla.org.

[54] N. Sophakan and C. Sathitwiriyawong, "A secured OpenFlow-based software defined networking using dynamic bayesian network," in *Proceedings of the 2019 19th International Conference on Control, Automation and Systems (ICCAS)*, pp. 1517–1522, Jeju, Korea, October 2019.