Received December 21, 2018, accepted January 14, 2019, date of publication January 25, 2019, date of current version April 1, 2019.

Digital Object Identifier 10.1109/ACCESS.2019.2895092

SGS: Safe-Guard Scheme for Protecting Control Plane Against DDoS Attacks in Software-Defined Networking

YANG WANG[1], TAO HU[2], GUANGMING TANG[1], JICHAO XIE[2], AND JIE LU[2]
[1]Zhengzhou Information Science and Technology Institute, Zhengzhou 450002, China
[2]National Digital Switching System Engineering and Technological Research Center, Zhengzhou 450002, China

Corresponding author: Tao Hu (hutaondsc@163.com)

This work was supported in part by the Project of National Natural Science Foundation of China under Grant 61601517, Grant 61702547, Grant 61802429, Grant 61872382, and Grant 61502530, in part by the National Network Cyberspace Security under Grant 2017YFB0803204, and in part by the Foundation for Innovative Research Group of the National Natural Science Foundation of China under Grant 61521003.

ABSTRACT Software-defined networking (SDN) achieves flexible and efficient network management by decoupling control plane from the data plane, where the controller with a global network view is responsible for planning routing for packets. However, the centralized design makes the controller become a potential bottleneck, and adversaries can exploit this vulnerability to launch distributed denial-of-service (DDoS) attacks to the controller. Existing solutions are fundamentally based forged traffic analysis, increasing computational cost and being prone to produce false positives. This paper proposes a safe-guard scheme (SGS) for protecting control plane against DDoS attacks, and the main characteristic of SGS is deploying multi-controller in control plane through the controller's clustering. SGS procedures are organized in two modules: anomaly traffic detection and controller dynamic defense. Anomaly traffic detection focuses on switches in data plane to distinguish forged flows from legitimate ones by innovatively adopting four-tuple feature vector. Controller dynamic defense mitigates DDoS attacks' effects on control plane by remapping controller and sending the access control message to switches. The simulation results demonstrate the efficiency of our proposed SGS with real-time DDoS attack defense and high detection accuracy, as well as high-efficiency network resource utilization.

INDEX TERMS Software-defined networking, multi-controller, DDoS, network security, anomaly traffic detection.

I. INTRODUCTION

Software-Defined Networking (SDN) as a novel networking paradigm, has greatly changed the traditional network architecture [1]. It has three features that are centralized control, decoupling control plane from the data plane, and flexible programmability. For SDN, the network intelligence is logically centralized in control plane (e.g. controller), and the network devices (e.g. switch) in data plane become simple packet forwarders which can be configured through the southbound interface. Due to its behavior, SDN is highly concerned by both academics and industry [2], [3]. Further, more and more network scenarios including backbone network, wireless network, and data center have adopted SDN to improve network management [4].

For a typical SDN network, it contains controller and several switches. Those switches are defined as SDN switches and comply with OpenFlow protocol. In SDN, when a switch receives a packet that doesn't have a matching entry in its flow tables, it will first buffer and process the packet as a new flow [5]. Further, the packet is encapsulated in a message named as packet_in, which is sent to the controller to request a new flow entry. The controller, in turn, sends packet_out messages to program forwarding rules into switches [6]. Given this fact, it's clearly seen that the controller carries considerable overhead and would be easy to become a security bottleneck. An adversary can make use of this feature of SDN to launch *distributed denial-of-service*

The associate editor coordinating the review of this manuscript and approving it for publication was Xiangjie Kong.

(DDoS) attacks to the controller [7]. Specifically, attackers can hire a group of compromised hosts (e.g. zombies) to flood switches with a large number of faked flow arrivals, with non-repetitive random header patterns. In the absence of effective protection, this will flood the control plane and controller must generate corresponding flow entries for every spoofed packet, which exhausts the limited controller resources. Unlike the DDoS attacks of traditional TCP/IP network, SDN network will be quickly crashed once controller suffered DDoS attacks [8], [9].

In recent years, researchers have proposed several solutions to alleviate the impact of DDoS attacks in controller and SDN, which can be concluded into two aspects: *network traffic analysis*and *controller capacity scale-up*. The first method mitigates DDoS attacks through detecting and filtering the forged flow. For example, a lightweight protecting scheme is proposed, and it is based on a set of rules to characterize packets send to a network switch as malicious or not [10]. Another solution devotes to scale up the capacity of the controller to cope with the higher processing workloads under a DDoS attack. For example, the authors introduce a self-organized SDN controller cluster by multi-controller to defense DDoS attacks [11].

Though the above two types of approaches have mitigated the DDoS attacks towards SDN controller to some extent, there are still some shortcomings. Network traffic analysis depends on flow filtering design and easily presents high rates of false negatives and/or false positives if selecting improper filter parameters. On the other hand, controller capacity scale-up requires to deploy multiple controllers to prevent control plane from DDoS attacks, but those active controllers will increase network overhead and low control resources utilization.

In this paper, we combine the strengths of both two methods and make improvements. We propose SGS, a Safe-Guard Scheme for protecting control plane against DDoS attacks. SGS contains two stages: anomaly traffic detection in data plane and controller dynamic defense in control plane. Concretely, the first stage is designed to find the forged flow of network traffic to estimate whether the adversary has launched DDoS attacks towards SDN network. If so, the second stage will activate the controller remapping to implement dynamic defense. Here, the introduced control plane includes multiple controllers. One controller acts as the master controller, which is responsible for processing flow request send from switches, while the other controllers play as slave roles and are in dormant states under normal condition. Once detecting DDoS attacks, the master controller sends access control messages for switches, meanwhile, those slave controllers are activated and conduct dynamic remapping to mitigate DDoS attacks. The contributions of our work are summarized as follows.

- We propose a Safe-Guard Scheme (SGS), which combines two modules: anomaly traffic detection in data plane and controller dynamic defense in control plane, for protecting the control plane against DDoS attacks.
- We introduce a flow monitoring approach in anomaly traffic detection module. It extracts a four-tuple vector from the from the flows of switches based on rate feature and asymmetry feature and distinguishes normal flows and attack flows by improving Back Propagation Neural Network (BPNN) method.
- We put forward controller remapping and access control in controller dynamic defense module. Controller remapping activates the slave controllers in dormant states to share attacked master controller's loads. Meanwhile, the master controller sends access control messages to switches based on the traffic detecting results to block forged flows.
- We evaluate the performance of SGS against several baseline schemes. The results show that SGS can quickly detect and react DDoS attacks. Flow setup time has been reduced by 30.4% on average, and controller response time of SGS has been reduced by 42.1% at least.

The rest of this paper is organized as follows. Section II presents the related works. Section III illustrates the details of system design of SGS. The evaluation scheme and results are provided in Section IV. Finally, this paper is concluded in Section V.

II. RELATED WORK

SDN is a new type of network architecture that is currently abstracting much attention from academia and industry [2]. In the meantime, SDN security is a hot research point, one important aspect of which is the prevention against DDoS attacks. Differing from the traditional network, SDN employs centralized control logic to manage the entire network. Once SDN suffers a DDoS attack, the inadequate defense will destroy control plane and further crash network communications. In recent years, several defense methods for DDoS attacks have been proposed, which can be divided into two aspects: network traffic analysis and controller capacity scale-up.

To date, a large variety of DDoS mitigation mechanisms had been proposed, which are mainly based on network traffic analysis. Wang *et al.* [12] design and implement SGuard, that is a security application on top of the NOX controller. SGuard includes two modules: access control module and classification module, and those cooperate with each other to complete a series of tasks such as authorization, classification and so on. Alshamrani *et al.* [13] investigate the two type of DDoS attacks: misbehavior attack and new flow attack and propose a secure system that periodically collects network statistics from the forwarding elements and applies Machine Learning (ML) classification algorithms. ArOMA, as an autonomic DDoS defense framework, is proposed in [14]. ArOMA can systematically bridge the gaps between different security functions, ranging from traffic monitoring to anomaly detection to mitigation while sparing human operators from nontrivial interventions. It also facilitates the collaborations between ISPs and their customers on DDoS mitigation by logically distributing the essential security functions. Cui *et al.* [15] introduce a mechanism consisting of four modules, namely attack detection trigger, attack detection, attack traceback, and attack mitigation. The trigger of the attack detection mechanism is introduced for the first time to respond more quickly against DDoS attack and reduce the workload of controllers and switches. DDoS attack detection method based on neural network is implemented to detect the attack. Similar with our work, Yang *et al.* [16] propose an SDN-based DDoS attack detection framework with cross-plane collaboration named as OverWatch, which performs a two-stage granularity filtering procedure between coarse-grained detection data plane and fine-grained detection control plane for abnormal flows. However, it presents high rates of false negatives and/or false positives due to lack of reasonable flow filtering settings.

In terms of controller capacity scale-up, the authors firstly provide an increase of resilience in SDN using a component organization, which is named as the CPRecovery component [17]. The CPRecovery component is based on the primary-backup mechanism which offers resilience in case of DDoS attacks in a centralized controlled network. However, such an approach is based on replication, employing external hardware resources to build replicas. Lim *et al.* [18] propose a scheduling-based scheme that contains most of the attack traffic at attack ingress switches. And this architecture that helps the controller weather out the DoS attacks targeted at it independently of the attack mitigation measures working in the switches on the actual attack flow paths. Similarly, a multi-queue SDN controller scheduling algorithm is proposed in [19], which is based on a time slice allocation strategy. This method can take different time slice allocation strategies according to the intensity of DDoS attacks and use SDN controller to schedule processing flow request from different switches. FLOWGUARD [20] is designed to facilitate not only accurate detection but also the effective resolution of firewall policy violations in SDN networks. It checks network flow path spaces to detect firewall policy violations when network states are under DDoS attacks. Moreover, FLOWGUARD also implements automatic and real-time violation resolutions with the help of several innovative resolution strategies. Macedo *et al.* [21] propose the organizations of IdP (Identity Providers) clustering using optimization techniques to mitigate DDoS attacks. It is started based on the monitoring of processing and memory resources and minimizes the effects of the DDoS attacks using operational IdPs. PATMOS [11] presents a new protocol for DDoS attack mitigation in multi-controller SDN network through controller's clustering. PATMOS is divided three steps: i) exchanging control messages to identify overloaded controllers; ii) electing the best performance level controller to coordinate the mitigation process; ii) minimizing the effects of the DDoS attacks by using operational controllers.

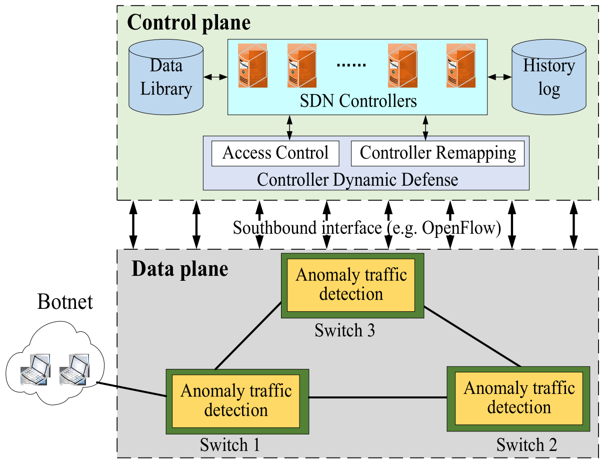
Control plane
Data Library
SDN Controllers
History log
Access Control
Controller Remapping
Controller Dynamic Defense
Southbound interface (e.g. OpenFlow)
Data plane
Botnet
Anomaly traffic detection
Switch 3
Anomaly traffic detection
Switch 1
Anomaly traffic detection
Switch 2

FIGURE 1. SGS overview.

III. SGS DESIGN

In this section, we introduce a Safe-Guard Scheme (SGS) for protecting the control plane against DDoS attacks. SGS can detect the anomaly traffic and implement controller dynamic defense to protect the controller from DDoS attacks. It has the characteristics of lightweight, scalability, and high-efficiency. Next, we will detailly illustrate SGS design.

A. ARCHITECTURE OVERVIEW

We present the architecture of our proposed scheme (SGS) in Fig. 1. It makes use of the decoupling characteristic of SDN and is developed in data plane and control plane in the meantime. Specifically, SGS performs a two-stage defense procedure, which includes anomaly traffic detection in data plane and controller dynamic defense in control plane. It firstly conducts traffic detecting on the data plane switches to find abnormal flows. Once abnormal flows are detected, the controllers in the control plane will perform dynamic defense including controller remapping and access control to deeply mitigate DDoS attacks.

Fig. 2 shows a finite-state machine, which is used to manage the entire SGS system. It includes four states and five events. The above four states are Initial state, Detection state, Defense state, and Safe state. Specially, Detection state and Defense state corresponding to anomaly traffic detection module and controller dynamic defense module of SGS, respectively. The explanation of each state is as follows.

1) INITIAL STATE

Initial state is the beginning of SGS. When the network has run, SGS will receive packets from host and transmit those to switches.

2) DETECTION STATE

This state is responsible for identifying anomaly traffic to find potential DDoS attack, which is executed in the switches in data plane. If system does not find any abnormal events, it comes to Safe state. Otherwise, it will turn to Defense state.

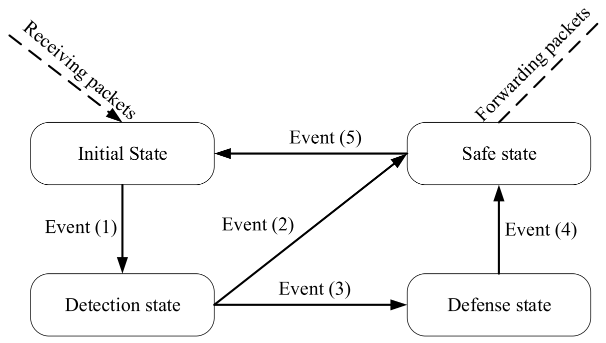

FIGURE 2. **The state diagram of SGS.**

3) DEFENSE STATE

Once in this state, the system has confirmed there are DDoS attacks in the network and enables controller dynamic defense module. It remaps controller to relief workload surge of attacked controller and sends access control messages to switches to block anomaly flows from source and clean malicious flow entry of affected switches. This state mainly runs in the control plane.

4) SAFE STATE

In this state, the system forwards packets and returns to Initial state.

The explanation of each event is as follows.

Event (1): The system completes the preparation work to detect the traffic from source.

Event (2): The system does not find DDoS attack.

Event (3): The system has found DDoS attack.

Event (4): The system runs controller dynamic defense module to mitigate DDoS attack.

Event (5): The system has been executed successfully.

The objective of this paper is that the network can detect DDoS attack exactly and react to it quickly. In order to achieve this objective, we cooperate data plane and control plane and design the corresponding modules to improve the defense ability. In the following subsection, we will present the module designs of anomaly traffic detection in data plane and controller dynamic defense in control plane.

B. ANOMALY TRAFFIC DETECTION

The anomaly traffic detection module runs on the switches of data plane, and its main objectives are to detect DDoS attacks and react those to controller. Therefore, we introduce three functionalities in this module: ***feature extracting, detection reacting, and results executing***. The running process is shown in Fig. 3. Generally, the switches must extract the key features of DDoS attacks. Based on this, detection reacting will estimate whether there is anomaly flow in the traffic. If so, the anomaly flow alert will be transmitted to controller dynamic defense module. Otherwise, normal flow is processed by results executing and the packets are forwarded normally.

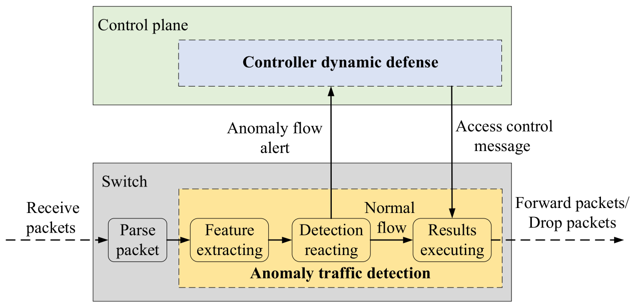

FIGURE 3. **The running process of anomaly traffic detection module.**

1) FEATURE EXTRACTING

In data plane, the switch is responsible for receiving and forwarding packets, so we can quickly extract the key features of packets in switches. In general, DDoS attacks behaviors are substantially different from normal network traffic. By deeply analyzing the characteristics of DDoS attacks, we find that DDoS attacks always produce large traffic in the network. Moreover, because the botnet forges many faked IP address to implement DDoS attacks, the number of asymmetric flows will have a huge proportion in the traffic. In order to improve DDoS attacks effect, most anomaly flows only have few packets so as to generate traffic rapidly. Based on the above analysis, we can conclude that rate feature and asymmetry feature play important roles in anomaly detection. Therefore, we introduce four novel features to elaborate on the characteristics of DoS attacks.

a: BYTE RATE (BR)

Here, we compute the byte count during specific time interval to reflect the average byte rate of flow, as shown in Eq. (1),

$$BR^f_{t_n} = \frac{B^f_{t_n} - B^f_{t_{n-1}}}{t_n - t_{n-1}} \tag{1}$$

where $B^f_{t_n}$ and $B^f_{t_{n-1}}$ is the number of bytes of specific flow f in time t_n and t_{n-1}, and $BR^f_{t_n}$ is the average byte rate of flow f between time t_n and t_{n-1}.

b: SYMMETRIC FLOWS PERCENTAGE (SFP)

We firstly introduce the definition of symmetric flows. For two flows f_1 and f_2, if they satisfy the two constraints, then two flows are reversible: (i) $f_1\,(SrcIP) = f_2\,(DstIP)$; (ii) $f_2\,(SrcIP) = f_1\,(DstIP)$.

Here, we compute the percentage of symmetric flows to reflect another feature of DDoS attack in Eq. (2),

$$SFP = \frac{SF_{num}}{F_{num}} \tag{2}$$

where SF_{num} is the number of symmetric flows, and F_{num} is the sum of flows.

c: VARIATION RATE OF ASYMMETRIC FLOW (VAFR)

In the real network environment, business surge can also cause traffic burst and produce a large number of packets.

Thus, it is necessary to distinguish normal traffic surge and DDoS attacks. Here, we introduce the variation rate of asymmetric flow to solve this problem, as shown in Eq. (3),

$$VAFR_{t_n} = \frac{F_{num} - SF_{num}}{t_n - t_{n-1}} \tag{3}$$

where $VAFR_{t_n}$ represents the variation rate of asymmetric flow between t_n and t_n.

d: FLOWS PERCENTAGE WITH SMALL AMOUNT PACKETS (FPSA)

In order to reach DDoS attacks quickly, the attacker always send lots of flows, which has small number of packets in each flow. Actually, in the normal traffic, each flow contains plenty of packets to transmit valid information. Therefore, we compute the percentage of flows with small amount packets to show DDoS attacks status, as shown in Eq. (4),

$$FPSA = \sum_{i}^{F_{sum}} \frac{F_i\,(PN_i < V_T)}{F_{sum}} \tag{4}$$

where F_i is the i^{th} flow, PN_i is the number of packets in F_i, and V_T is a threshold value to judge small amount packets.

2) DETECTION REACTING

In this subsection, we will complete the abnormal detection based on the four-tuple feature vector mentioned above. In order to perfect the detection efficiency and reduce false positives, we improve the existing Back Propagation Neural Network (BPNN) method and design a lightweight flow detection algorithm based on feature threshold [22].

As an error back propagation algorithm, BPNN includes two parts: computed information forward propagation and error information back propagation. The architecture of BPNN is shown in Fig. 4. It is clearly seen that BPNN contains three layers. In input layer, each neuron receives input information and transfers it to the middle layer. Middle layer is used for computing information. Finally, the computed information is passed to output layer. If the output results match the expected output or the training number comes up to the threshold, the results will be output. Otherwise, the back propagation starts.

Therefore, taking advantages of BPNN, we can improve the detection efficiency and reduce false positives during the traffic detection process. The run process of BPPN is illustrated as follows. Once the detection begins, the neural network will be first trained. At first, the system will collect the feature sets of normal traffic and anomaly traffic. Then, the extracted four-tuple feature vector and objective values will be combined with the trained dataset. When SGS begins to run, the trained data set is used to train the neural network. Moreover, the extracted four features will be considered as the input parameters, and the objective values is the output parameter.

Here, in order to further improve the accuracy of BPNN and the adaptability for various packets, we design a

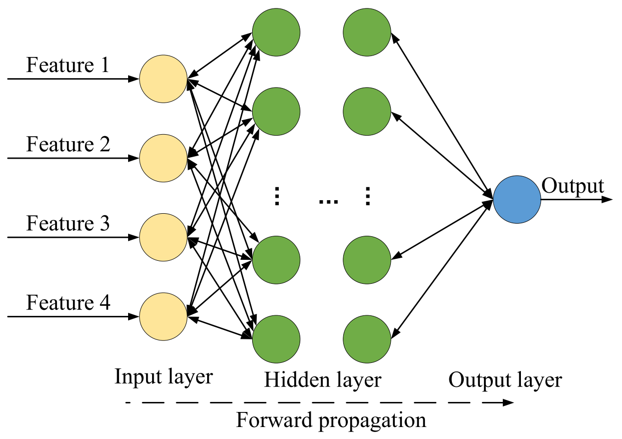

FIGURE 4. The architecture of BPNN.

lightweight flow detection algorithm based on feature threshold, which is described as follows. The input parameters of the neural network are the four-tuple feature vectors: byte rate, symmetric flows percentage, the variation rate of asymmetric flow and flows percentage with the small number of packets. The features can be captured by the feature extracting functionality. Specifically, it uses previous metric samples from the specific flow to evaluate the future value. Firstly, the input packets of each switch will be parsed. Then, the extracted feature information will be processed by BPNN one by one. Meanwhile, the detection results will be stored in an array to avoid the repeated processing in feedback state. If the actual values for the four feature vectors all fall into the threshold of the prediction, it shows the current flow is normal. Otherwise, it will indicate the flow is abnormal caused by DDoS attacks. The pseudo-code of the algorithm is shown in Table 1.

3) RESULTS EXECUTING

In Fig. 3, we can see that the results executing functionality has two inputs and one output. We illustrate this as follows. On the one hand, results executing receives the message from detection reacting. Specifically, this message usually indicates the passed flows are normal and the switches should forward the packets accurately. On the other hand, results executing will receive the access control message sent from the controller, and this message requires the switch to drop the specific packets, which are from the detected anomaly flow, to block the DDoS attack.

C. CONTROLLER DYNAMIC DEFENSE

In control plane, we design the controller dynamic defense module to eliminate DDoS attacks detected from anomaly traffic detection module, and this module is brain to SGS. According to our main objective, the cores of controller dynamic defense contains two aspects: (1) remapping controller to relief workload surge of attacked controller; (2) sending access control messages to switches to block anomaly flows from source switches. Based on those,

TABLE 1. A lightweight flow detection algorithm.

```
Algorithm 1: A lightweight flow detection algorithm
Input:     Receiving packets
Output:    Feature set $FS = [BR, SFP, VAFR, FPSA]$
           Abnormal flow/ normal flow
1:  Receive start command and all packets
2:  Get flow states message
3:  For each packet, get features
4:  Construct feature set $FS$
5:    for each $i \in [1,4]$
6:      BPNN ( $FS$ )
7:      *while* ( $FS[i] < V_T^i$ ) //Feature value is less than threshold
8:        label $f_i$ and store it into dataset DS
9:      *endwhile*
10      Detecting (Dataset $DS$, Feature set $FS$)
11:       *if* Abnormal flow *then*:
12:         Label the flow $f_{abnormal}$
13:         Send anomaly flow alert to controller
14:       *endif*
15:       *if* Normal flow *then*:
16:         Send normal flow message to results executing
17:      *endif*
18: Output the detection results
```

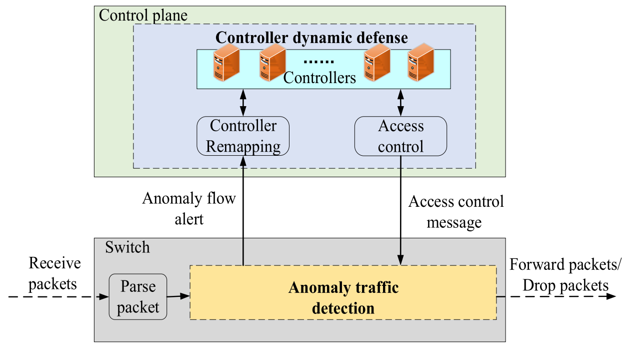

FIGURE 5. The running process of controller dynamic defense module.

we introduce ***controller remapping*** and ***access control*** in this module, as shown in Fig. 5. Controller remapping receives the anomaly flow alert, which includes DDoS attacks messages, sent from data plane. Then, it identifies the overloaded controller and remaps controllers to relief the loads of overloaded controller. After that, access control is enabled and it sends response message to data plane to block anomaly flows from source and clean malicious flow entry of affected switches.

1) CONTROLLER REMAPPING

In order to improve the scalability and reliability of SDN, multiple controllers are usually deployed in the network. Generally, the network is divided into several domains and each controller is responsible for one domain. As shown in Fig. 6, there are three controllers (C1, C2, C3) and eight switches (S1 to S8) in the network, and C1 controls S1 to S3, C2 controls S4 to S6, C3 controls S7 to S8. Moreover, the controllers share domain information (e.g. topology information, traffic information) with each other.

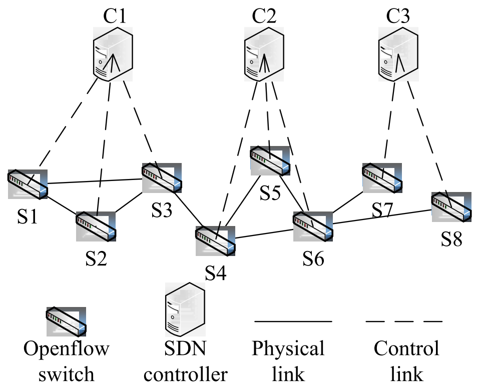

FIGURE 6. Multiple controllers in SDN.

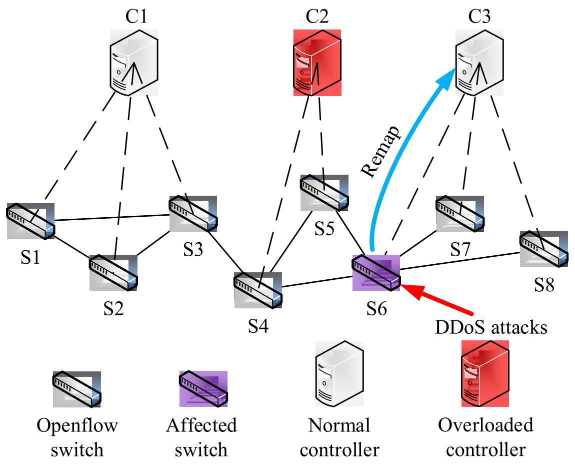

FIGURE 7. A motivating example for controller remapping.

With the help of multiple controllers, we remap the controllers to relief workload surge of attacked controller. The core idea follows three phases: (1) identifying phase; (2) electing phase; (3) composition phase. Identifying phase is responsible for finding the controller affected by DDoS attacks based on the results of anomaly traffic detection. Then, electing phase selects target controller from controller clustering so as to let it relief workload of overloaded controller. Finally, composition phase is responsible for remapping the connection relationship between controllers and switches. For example, in Fig. 7, the network is under DDoS attacks. Once SGS is enabled, the system will quickly find the abnormal flows from switch S6 based on the output of anomaly traffic detection. Then, controller remapping considers C2 as overloaded controller and elects C3 as remapped object. It remaps S6 from C2 to C3 and composite the corresponding connection relationships.

a: IDENTIFYING PHASE

This phase is aimed at identifying overloaded controllers. When suffering DDoS attacks, a controller may be overloaded due to consuming all the hardware resources to process malicious flow requests. Therefore, it is important to

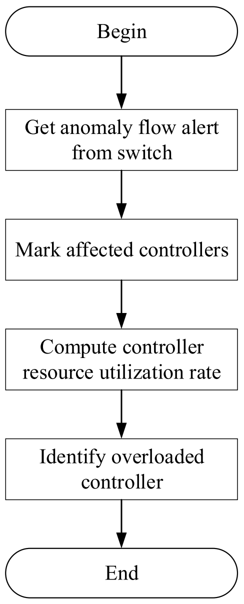
Begin
Get anomaly flow alert from switch
Mark affected controllers
Compute controller resource utilization rate
Identify overloaded controller
End

FIGURE 8. **The workflow of identify phase in controller remapping.**

identify the overloaded controllers in SDN network under DDoS attacks.

In order to complete this task, the workflow of this phase is shown in Fig. 8. Firstly, the control plane gets the anomaly flow alert sent from data plane. According to the mapping relationships between switches and controllers, the controller affected by abnormal flow are marked and stored in set C_a. Further, we compute the resource utilization rate η_i of controller that is in C_a, as shown in Eq. (5), where N_i is the number of switches managed by controller c_i, λ_j is the flow request rate of switch s_j, and ω_i is the processing capacity of controller c_i. In Eq. (6), we set the overload condition of controller by referring to [13], and then the controller state is determined by η_i. The controller c_i is considered as overloaded controller if meets $0.9 \leq \eta_i \leq 1$. Based on the above, this phase outputs the overloaded controllers set C_o finally.

$$\eta_i = \frac{\sum_{j=1}^{N_i} \lambda_j}{\omega_i} \tag{5}$$

$$\begin{cases} 0.9 \leq \eta_i \leq 1 \; controller\ c_i\ is\ overloaded \\ 0 < \eta_i < 0.9 \; controller\ c_i\ is\ normal. \end{cases} \tag{6}$$

b: ELECTING PHASE

In this phase, we select target controller from the normal controller set $C_n (C_n = C - C_a)$ so as to let it relief the workload surge of the overloaded controller. The selection of target controller depends on resource utilization rate η, hops between affected switch and controller H and controller state synchronization cost SC. State synchronization is implemented among controllers after controller remapping have been complete. Here, in order to simplify operation, we briefly consider the flow request rate λ_j as the synchronization cost. Therefore, when selecting target controller, we must consider η, H, SC simultaneously and set $\min(\eta, H, SC)$ as the objective function. This problem is optimized as the multi-objective mixed linear program and it is solved by greedy algorithm [22]. Greedy algorithm is a heuristic algorithm, which makes the local optimal choice based on specific measures without considering the overall situation. Therefore, it is suitable for selecting the optimal target controller under multiple objectives environment.

TABLE 2. **Target controller election algorithm.**

```
Algorithm2 Target controller election
Input: Overloaded controller: $c_o$
     Normal controller set: $C_n$
          Affected switch: $s_m$
Output: Target controller: $c_t$
1: Get normal controller set $C_n = C - C_a$
2: Select $c_n \in C_n$, Compute $\eta_n$, $H_{nm}$ and $SC_n$
3: if $c_n$ has $\max \eta_n$
4:   then Remove $c_n$
5:     if $c_n$ has $\min \eta_n$
6:      then Store $c_n$
7:     endif
8: endif
9: For $H_{nm}$ and $SC_n$, Repeat 3~8
10: Compute $Objective = \min(\eta, H, SC)$ for the stored $c_n$
11: if $Current\_Objective < Last\_Objective$
12:      then Set $\min(\eta, H, SC) = Cureent\_Objective$
13:   Select $c_t$ with $\min(\eta, H, SC)$ as target controller
14: endif
15: if no controller accord with objective function
16:   controller with $\min \eta$ will be set as $c_t$
17: endif
18: Output $c_t$
```

Based on the above analysis, we design target controller election algorithm. Initially, all controllers are in active state in normal controller set C_n. For one affected switch s_m, we select controller c_n from C_n and calculate the corresponding objective values η_n, H_{nm} and SC_n. If the calculation value is less than the previous minimum value, this controller will be stored. The algorithm tries to remove controller with the unilateral high cost in each round and uses the minimum value configuration as a starting point. When the round robin is over, the controller that meets $\min(\eta, H, SC)$ will be set as a target controller. Specifically, if no controller accords with objective function, the controller with minimum η will be set as target controller automatically. The algorithm pseudo code is shown in Table 2.

The complexity of the algorithm is related to the number of controllers. The algorithm will be implemented $|C_n| + (|C_n| -$

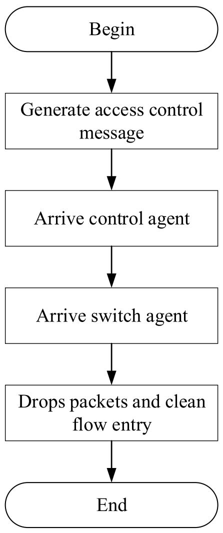
Begin
Generate access control message
Arrive control agent
Arrive switch agent
Drops packets and clean flow entry
End

FIGURE 9. **The workflow of access control.**

$1)+\ldots+(|C_n|-i)+\ldots+1$ times, so the complexity is $O(m^2)$ and $m=|C_n|-i$.

c: COMPOSITION PHASE

This phase remapping the connection relationship between controllers and switches, and the administrator of affected switch s_m will be transferred from overloaded controller c_o to target controller c_t. The overloaded controller c_o chooses the affected switch s_m to send *Remap* message to target controller c_t. c_t will respond a *Remap-Begin* message after receiving *Remap* and select a random number to start the countdown. Before countdown reduces to 0, if the composition completes, the administrator of s_m will be shifted from c_o to c_t. Meanwhile, *Update*message is broadcasted to the whole network, and c_t sends access control message to affected switch s_m (see section 3.3.2 in detail). However, if the countdown is overtime, the countdown is reset and the process of remapping will be repeated.

2) ACCESS CONTROL

The main objective of access control is sending response message to affected switch to block anomaly flows from source and clean malicious flow entry of affected switches. In most cases, the overloaded controller caused by DDoS attacks cannot communicate with switches normally. Therefore, based on the remapping results obtained in 3.3.1, the administration authority of affected switch has been transferred from overloaded controller to underload controller (target controller). Benefiting from the better performance of target controller, we implement access control function in the target controller. The flowchart of access control is shown in Fig. 9. From, the control mapping of affected switch has been changed, and the target controller is responsible for managing this switch. Firstly, in control plane, the target controller generates access control message based on the detected DDoS attacks. Then, this message will arrive control agent of controller and be packed into a packet_out message. After that, the packet_out arrives at switch agent through Openflow channel. Finally, the affected switch parses packet_out message, and drop the packet of abnormal flow and clean the corresponding flow entry according to the instructions of packet_out, the affected switch will drop all DDoS attacks packets and clean the corresponding flow entries to release the space occupied by attack traffic.

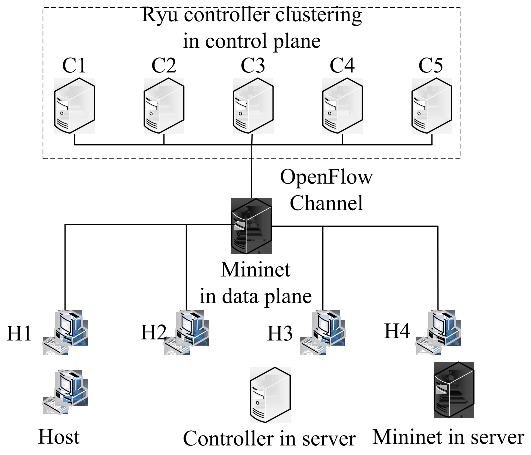

FIGURE 10. **The experimental topology.**

IV. EVALUATION

A. SIMULATION SETTING

In this section, we evaluate the performance of SGS under the experimental environment shown in Fig. 10, and make the following descriptions.

1) EXPERIMENTAL PLATFORM

We deploy two servers in the experiment, where one is used for control plane and another is used for data plane. For control plane, we have selected Ryu controllers [23] (C1-C5 in Fig. 10), which are installed as clustering in the server. We use Mininet [24] as a test platform to construct OpenFlow network environment. Initially, each controller manages 10 switches. According to OpenFlow 1.3, one switch can connect multiple controllers, where one plays master role and the others are slave roles. SGS is implemented extending the Ryu controller settings. Four laptop hosts represent DDoS attackers, victims, and normal traffic generators, respectively. All servers have the same configuration, including Intel Core i7 3.5GHz 8GB RAM, Ubuntu 14.04.

2) PARAMETERS SETTING

The transferred traffic in the experiment was composed of several protocols: 70% was TCP, 20% was UDP and 10% was ICMP. It is generated based on the analysis of traffic appeared in [25]. D-ITG is a traffic generator used in the test, which supports to generate TCP, UDP, and ICMP traffic [26]. Meanwhile, DDoS traffic is generated by hping3 [27], and it can produce most types of attacks, such as TCP flood

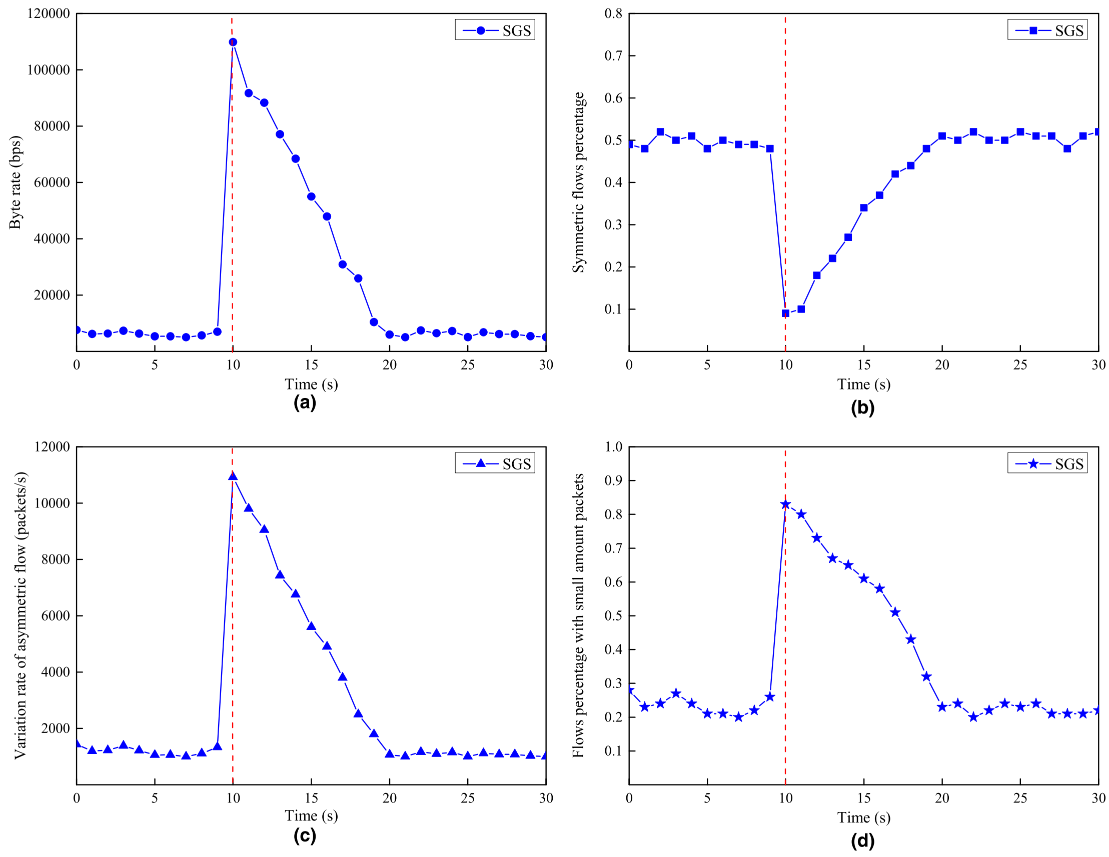

FIGURE 11. **The validation results of four-tuple vector for DDoS attacks detection. (a) Byte rate. (b) Symmetric flows percentage. (c) Variation rate of asymmetric flow. (d) Flows percentage with small amount packets.**

TABLE 3. **Parameters setting for simulation.**

Definition	Value
Byte rate	100000 bps
Symmetric flows percentage	10%
Variation rate of asymmetric flow	10000 packets/s
Flows percentage with small amount packets	80%
Number of neurons in input layer	4
Number of neurons in hidden layer	8
Number of neurons in output layer	1

attack, UDP flood attack, ICMP flood attack. Specifically, parameters are set as Table 3.

B. RESULTS ANALYSIS

1) EFFECTIVENESS OF FOUR-TUPLE VECTOR SETTING

In order to verify the effectiveness of four-tuple vector (*Byte rate*, *Symmetric flows percentage*, *Variation rate of asymmetric flow* and *Flows percentage with small amount packets*) for DDoS attacks detection, we firstly set two rules to enable traffic forwarding from H1 and H2 to H3. After that, hping3 is used for implementing DDoS attacks from H1 and H2 to H3 within ten seconds. The detection results are depicted in Fig. 11. We launch DDoS attack at 10s and observe the changing of metrics. It is clearly seen that four metrics have great changes at 10s. More precisely, byte rate metric reaches a high level (more than 100000bps), the value of symmetric flows percentage sharply drops to below 0.1, variation rate of asymmetric flow is several orders of magnitude higher than that in normal condition, and flows percentage with small amount packets is over 80%. Therefore, all of the above results demonstrate that the proposed four metrics in anomaly traffic detection module are able to capture great changes in rate and asymmetry features as soon as the attack occurs, which also evidently indicates the effectiveness of the algorithm 1. Besides, in this experiment, the proposed SGS runs continuously and we can see that all metric return to normal level within 10 seconds, which also validate the availability of SGS for defensing DDoS attacks.

2) PERFORMANCE EVALUATION UNDER VARYING ATTACK INTENSITIES

In this experiment, we compare the performances of different schemes proposed in Section 4.1. The evaluation indexes focus on flow setup time in data plane and controller response time in control plane. The simulation results are shown in Fig. 12.

Fig. 12(a) shows the variation of flow setup time of the four compared schemes under varying attack intensities. Here, we define flow setup time as the time from the sending of packet by host 1 until the handshake finishes.

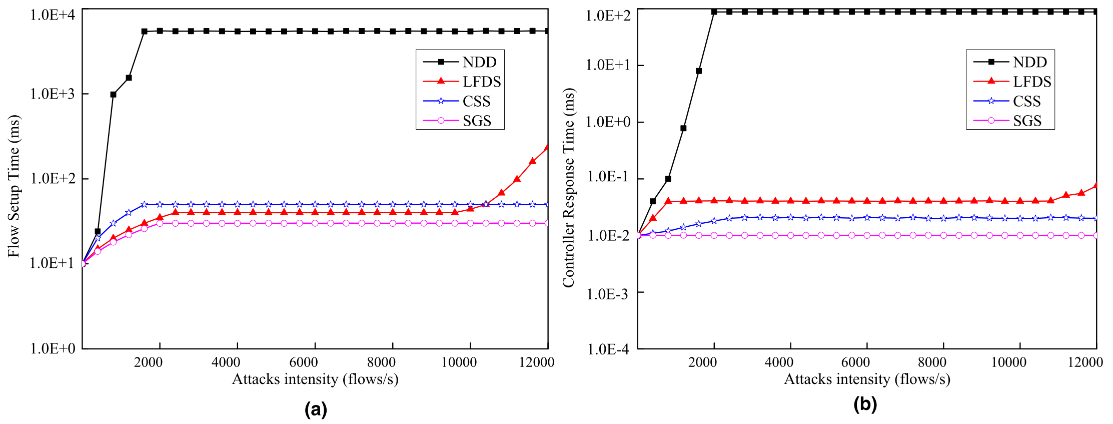

FIGURE 12. **Performance evaluation under varying attack intensities. (a) Flow setup time. (b) Controller response time.**

When attack intensity increases over 500 flows/s, NDD has no defense ability and gets overwhelmed quickly. In this moment, the network goes into paralysis, and the flow setup time is extremely large. Compared with single flow filtering in LFDS or single capacity scale-up in CSS, SGS does not drop any legitimate flow, as the anomaly traffic detection module provides fine-grained filter to identify abnormal flows, and the access control stops forged flows from flooding the controller. Therefore, the value of flow setup time of SGS is the lowest in all schemes under varying attack intensities, which has reduced by 30.4% on average compared with the other schemes.

Fig. 12(b) shows the variation of controller response time, under varying attack intensities, for the four schemes. As was mentioned above, NDD gets quickly overwhelmed and controllers are in crashed states, so its controller response time is highest. SGS not only provides anomaly traffic detection, but also remaps controller to relief workload surge. Thus, SGS can promise each controller has the enough capacity to process flow request, and its controller response time is lowest. Particularly, differing from flow setup case, the value of controller response time of LFDS is higher than CSS's. This is due to that vSwitch based overlay will significantly scale up control plane capacity, so as to reduce controller response time. Compared with CSS, the controller response time of SGS has been reduced by 42.1% at least.

3) CONTROLLER CPU USAGE RATE

In this experiment, all controllers have the similar processing states in the initial moment. A DDoS attack performed with H1 and H2 hosts during twenty seconds through 10 switches, and controller C5 is the overloaded controller caused by DDoS attacks. Then, we execute four schemes and observe the change of controller CPU usage. Results are an average of 20 repetitions for each scheme and are shown in Fig. 13.

In NDD scenario, C5 presents 94% of CPU usage. In LFDS scenario, the CPU usage of C5 has been reduced to 66%. In CSS scenario, the CPU usage of C5 has been decreased evidently, while the CPU usages of C3 and C4 have increased.

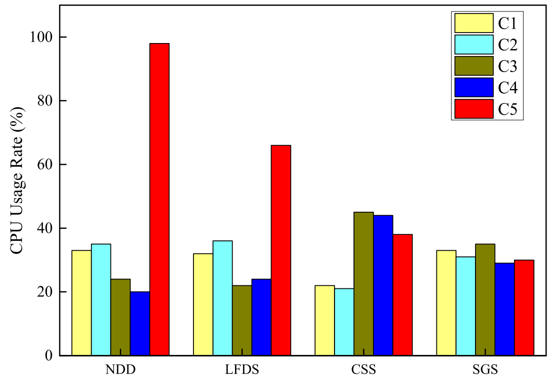

FIGURE 13. **CPU Usage Rate over DDoS attack.**

In SGS scenario, the CPU usage of C5 have declined and all controllers' CPU usages are similar. The reasons are explained as follows. NDD has no defense ability and its controller C5 always high CPU usage. LFDS filters most malicious flows based on a set of rules so as to CPU usage of C5 has reduced. However, due to the exists of false positives and false negative, the defense ability of LFDS is limited. CSS scales up the capacity of control plane to relief controller overload, but it is lacked of a global perspective. Thus, the CPU usages of five controllers have great difference. SGS not only implements abnormal flow detecting and stopping, but also remaps controller to employ adaptively unused resources. Therefore, in SGS, C5's CPU usage has been reduced obviously and all controllers' loads have been averaged approximatively, and network transmission performance has been improved effectively.

V. CONCLUSION

In this paper, we make the first attempt to protecting the control plane against DDoS attacks in SDN and propose a Safe-Guard Scheme (SGS). SGS implements two modules: anomaly traffic detection and controller dynamic defense. Anomaly traffic detection is designed to distinguish forged flows from legitimate ones by innovatively adopting four-tuple feature vector, while controller dynamic defense mitigates DDoS attacks effects by remapping controller and sending access control to the data plane. We also experimentally use Mininet and Ryu controllers to evaluate SGS. The simulation results verify the SGS efficiency. The flow setup time and controller response time have been reduced obviously under DDoS attack, and the network resource can be utilized enough. In the future work, we will plan to deploy SGS in the real network environment to further demonstrate its efficiency.

REFERENCES

[1] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," in *Proc. ACM SIGCOMM Comput. Commun. Rev.*, Seattle, WA, USA, 2008, pp. 69–74.

[2] B. Raghavan, M. Casado, T. Koponen, S. Ratnasamy, A. Ghodsi, and S. Shenker, "Software-defined Internet architecture: Decoupling architecture from infrastructure," in *Proc. 11th ACM Workshop Hot Topics Netw.*, New York, NY, USA, Oct. 2012, pp. 43–48.

[3] D. Levin, A. Wundsam, B. Heller, N. Handigol, and A. Feldmann, "Logically centralized?: State distribution trade-offs in software defined networks," in *Proc. 1st Workshop Hot Topics Softw. Defined Netw.*, New York, NY, USA, 2012, pp. 1–6.

[4] T. Hu, Z. Guo, P. Yi, T. Baker, and J. Lan, "Multi-controller based software-defined networking: A survey," *IEEE Access*, vol. 6, pp. 15980–15996, 2018.

[5] M. Antikainen, T. Aura, and M. Särelä, "Spook in your network: Attacking an SDN with a compromised OpenFlow switch," in *Proc. Secure IT Syst. Conf.*, Springer, 2014, pp. 229–244.

[6] T. Hu, P. Yi, Z. Guo, J. Lan, and J. Zhang, "Bidirectional matching strategy for multi-controller deployment in distributed software defined networking," *IEEE Access*, vol. 6, pp. 14946–14953, 2018.

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

[8] Q. Yan, F. R. Yu, Q. Gong, and J. Li, "Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: A survey, some research issues, and challenges," *IEEE Commun. Surveys Tuts.*, vol. 18, no. 1, pp. 602–622, 1st Quart., 2016.

[9] Z. Wang, H. Hu, and C. Zhang, "On achieving SDN controller diversity for improved network security using coloring algorithm," in *Proc. 3rd IEEE Int. Conf. Comput. Commun.*, Dec. 2017, pp. 1270–1275.

[10] C. Gkountis, M. Taha, J. Lloret, and G. Kambourakis, "Lightweight algorithm for protecting SDN controller against DDoS attacks," in *Proc. 10th IFIP Wireless Mobile Netw. Conf.*, Sep. 2017, pp. 1–6.

[11] R. Macedo, R. de Castro, A. Santos, Y. Ghamri-Doudane, and M. Nogueira, "Self-organized SDN controller cluster conformations against DDoS attacks effects," in *Proc. IEEE Global Commun. Conf.*, Dec. 2017, pp. 1–6.

[12] M. Semerci, A. T. Cemgil, and B. Sankur, "An intelligent cyber security system against DDoS attacks in SIP networks," *Comput. Netw.*, vol. 136, no. 1, pp. 137–154, 2018.

[13] A. Alshamrani, A. Chowdhary, S. Pisharody, D. Lu, and D. Huang, "A defense system for defeating DDoS attacks in SDN based networks," in *Proc. 15th ACM Int. Symp. Mobility Manage. Wireless Access*, Nov. 2017, pp. 83–92.

[14] R. Sahay, G. Blanc, Z. Zhang, and H. Debar, "*ArOMA*: An SDN based autonomic DDoS mitigation framework," *Comput. Secur.*, vol. 70, pp. 482–499, Sep. 2017.

[15] Y. Cui *et al.*, "SD-Anti-DDoS: Fast and efficient DDoS defense in software-defined networks," *J. Netw. Comput. Appl.*, vol. 68, pp. 65–79, Jun. 2016.

[16] X. Yang, B. Han, Z. Sun, and J. Huang, "SDN-based DDoS attack detection with cross-plane collaboration and lightweight flow monitoring," in *Proc. IEEE Global Commun. Conf. (GLOBECOM)*, Dec. 2017, pp. 1–6.

[17] P. Fonseca, R. Bennesby, E. Mota, and A. Passito, "A replication component for resilient openflow-based networking," in *Proc. IEEE Netw. Oper. Manage. Symp.*, Apr. 2012, pp. 933–939.

[18] S. Lim, S. Yang, Y. Kim, S. Yang, and H. Kim, "Controller scheduling for continued SDN operation under DDoS attacks," *Electron. Lett.*, vol. 51, no. 16, pp. 1259–1261, Jul. 2015.

[19] Q. Yan, Q. Gong, and F. R. Yu, "Effective software-defined networking controller scheduling method to mitigate DDoS attacks," *Electron. Lett.*, vol. 53, no. 7, pp. 469–471, Mar. 2017.

[20] H. Hu, W. Han, G.-J. Ahn, and Z. Zhao, "FLOWGUARD: Building robust firewalls for software-defined networks," in *Proc. 3rd Workshop Hot Topics Softw. Defined Netw.*, Aug. 2014, pp. 97–102.

[21] R. Macedo, A. Santos, Y. Ghamri-Doudane, and M. Nogueira, "A scheme for DDoS attacks mitigation in IdM systems through reorganizations," in *Proc. IEEE/IFIP Netw. Oper. Manage. Symp.*, Apr. 2016, pp. 298–305.

[22] C. Hua and S. C. Park, "Combination of modified BPNN algorithms and an efficient feature selection method for text categorization," *Inf. Process. Manage.*, vol. 45, no. 3, 2009, pp. 329–340.

[23] *Ryu*. [Online]. Available: https://ryu.readthedocs.io/en/latest/

[24] *Mininet*. [Online]. Available: http://mininet.org/

[25] P. Dong, X. Du, H. Zhang, and T. Xu, "A detection method for a novel DDoS attack against SDN controllers by vast new low-traffic flows," in *Proc. IEEE Int. Conf. Commun. (ICC)*, May 2016, pp. 1–6.

[26] S. Avallone, S. Guadagno, D. Emma, A. Pescape, and G. Ventre, "D-ITG distributed Internet traffic generator," in *Proc. 1st Int. Conf. Quant. Eval. Syst.*, Enschede, The Netherlands, 2004, pp. 316–317.

[27] K.-Y. Chen, A. R. Junuthula, I. K. Siddhrau, Y. Xu, and H. J. Chao, "SDNShield: Towards more comprehensive defense against DDoS attacks on SDN control plane," in *Proc. IEEE Conf. Commun. Netw. Secur.*, Oct. 2016, pp. 28–36.

YANG WANG is currently pursuing the Ph.D. degree with the Zhengzhou Information Science and Technology Institute. Her research interests include software-defined networking, network protocol, and network security.

TAO HU received the B.E. degree from Xi'an Jiaotong University. He is currently pursuing the Ph.D. degree in network cyberspace security with the National Digital Switching System Engineering and Technological Research Center (NDSC), Zhengzhou, China. His research interests include software-defined networking, control plane, and network security.

GUANGMING TANG is currently a Professor with the Zhengzhou Information Science and Technology Institute. Her contributions encompass the aspects of information theory and security, network architecture, and signal processing.

JICHAO XIE received the B.E. degree from the Nanjing University of Science and Technology. He is currently pursuing the M.S. degree in communication and information system with the National Digital Switching System Engineering and Technological Research Center, Zhengzhou, China. His research interests include cyber security, software-defined networking, and network function virtualization.

JIE LU received the bachelor's degree from Zhejiang University, in 2016. He is currently pursuing the master's degree with the National Digital Switching System Engineering and Technological Research Center, Zhengzhou, China. His research interests include software-defined networking, network management, and network security.