Yu *et al. J Wireless Com Network* (2021) 2021:90
https://doi.org/10.1186/s13638-021-01957-9

EURASIP Journal on Wireless Communications and Networking

RESEARCH Open Access

A cooperative DDoS attack detection scheme based on entropy and ensemble learning in SDN

Shanshan Yu[1], Jicheng Zhang[1,2*], Ju Liu[1], Xiaoqing Zhang[1], Yafeng Li[1] and Tianfeng Xu[1]

*Correspondence: 17861436995@163.com
[2] NetEase D&R Center Lab, Hangzhou, China
Full list of author information is available at the end of the article

Abstract

In order to solve the problem of distributed denial of service (DDoS) attack detection in software-defined network, we proposed a cooperative DDoS attack detection scheme based on entropy and ensemble learning. This method sets up a coarse-grained preliminary detection module based on entropy in the edge switch to monitor the network status in real time and report to the controller if any abnormality is found. Simultaneously, a fine-grained precise attack detection module is designed in the controller, and a ensemble learning-based algorithm is utilized to further identify abnormal traffic accurately. In this framework, the idle computing capability of edge switches is fully utilized with the design idea of edge computing to offload part of the detection task from the control plane to the data plane innovatively. Simulation results of two common DDoS attack methods, ICMP and SYN, show that the system can effectively detect DDoS attacks and greatly reduce the southbound communication overhead and the burden of the controller as well as the detection delay of the attacks.

Keywords: Software-defined network, Distributed denial of service, Edge switch, Entropy, Ensemble learning

1 Introduction

With the development of cloud computing, big data and other emerging technologies, network traffic is constantly increasing, and the traditional network architecture with IP as the core is difficult to meet the needs of network scalability, management and flexibility. Software-defined network (SDN), as a new network architecture, its core idea is the control plane and data plane are decoupled, where the state of the network is logically centralized, and the controller is abstracted from the underlying network facility. The emergence of SDN greatly improves the manageability, extensibility, controllability and dynamics of the network. However, with the popularity of SDN applications, the security of SDN has become one of the key research topics in the field of SDN. The distributed denial of service (DDoS) attack, as one of the most important security threats facing the internet today, is particularly dangerous in SDN due to its strong destructive power, simple implementation and lack of simple and feasible countermeasures. In order to block the attacked target providing services to legitimate users, the attacker builds a

botnet through the puppet host, launches a network attack to consume CPU, bandwidth, memory and other resources of the attacked targets [1]. Most of the traditional network defense schemes against DDoS attacks focus on traffic cleaning and firewall blocking, which makes it difficult to achieve unified scheduling of the entire network, and has not effective results even with large resources overhead. While the emergence of SDN brings a new opportunity for the detection of DDoS attack, which provides a basis for real-time monitoring of the whole network and the traffic situation of each node with its feature of the centralized control as well as the programmability.

The existing methods range from traditional statistical methods and modern machine learning algorithms, to the combination of multiple methods, and then to complex deep learning methods, all take good advantage of the global view and centralized management ability of the control plane to improve the accuracy of DDoS attack detection. However, due to flow collection, statistics and classification and so on are all need to process on SDN controller, when network scale is increasing, the controller must face to huge overhead, resulting in attack detection delay. And the worst situation is that before the DDoS attack is detected, the controller has already overburdened or even down. Simultaneously, the controller needs to frequently obtain flow table and packet information from the edge switch for attack detection. Thus when the network scale increases, the burden on the southbound interface will be heavy.

Therefore, in the process of DDoS attack detection, how to reduce the burden of the controller and the southbound interface as well as improve the attack detection speed while ensuring the detection accuracy is an important research topic. For this purpose, we design a cooperative DDoS attack detection scheme based on SDN. Considering the programmable ability of the OpenFlow switch, there are usually some remaining computing resources that are not fully utilized. So appropriate task of data statistics and analysis are arranged on the edge switch, which can implement part of the attack detection function to reduce the burden on the controller and improve the response speed of attack detection.

Our main contributions are as follows:

- We propose a cooperative DDOS attack detection framework based on entropy and ensemble training in SDN, which innovatively utilizes the computing power of the edge switch to offload part of the detection tasks from the control plane to the data plane. The lightweight algorithm in the edge switch and the precise method in the controller cooperate to accomplish the whole detection, which greatly reduce the burden of the controller and the overhead of the southbound communication.
- In the edge switch at the data plane, a fast anomaly detection algorithm based on information entropy is designed. Its low-complexity ensures low overhead of edge switch resources meanwhile monitoring of the traffic in real time.
- In the controller at the control plane, we construct a 5-feature set covering the typical basic characteristics of the network traffic and utilize the random forest algorithm to further detect the traffic in the whole domain accurately so as to ensure that the abnormal traffic can be identified quickly and effectively, then a dropping packet command can be timely delivered to the relevant switches through updating flow table to remove the attack threat.

The remainder of the paper is organized as follows. In Sect. 2, we investigate the related work of existing detection methods on DDoS attacks. In Sect. 3, the system model and the architecture of our proposed framework is presented, then the preliminary attack detection algorithm in the data plane and the precise attack detection scheme in the control plane are proposed in detail, respectively. In Sect. 4, the performance of the proposed framework is shown with the simulation experiments and analysis. Finally, the conclusion of this paper is given in Sect. 2.

2 Related work

Recently, the extensive researches have been conducted to apply SDN in detecting and mitigating DDoS attacks in a global point of view. Most schemes mainly utilize the controller to collect traffic information periodically and detect abnormal attacks by different centralized attack detection algorithms. Li et al. [2] proposed a controller scheduling method that uses the normalized waiting time, length and extent of the switch being attacked to choose the request that needs to be processed by the controller. And Lim et al. [3] also developed a scheduling-based architecture for the SDN controller that leads to effective attack confinement and network protection during DDoS attacks. In [4], Zheng et al. designed reinforcing anti-DDoS actions in real time to detect and throttle DDoS attacks via adaptive correlation analysis built upon unmodified commercial off-the-shelf SDN switches. It is a practical system to defend against a wide range of flooding-based DDoS attacks, while requiring neither modifications in SDN switches/protocols nor extra appliances. In addition, Kalkan et al. [5] proposed a joint entropy-based security scheme (JESS) to improve the security of SDN and enhance the ability to resist DDoS attacks.

The above articles [2–5] are based on traditional detection methods. Nowadays, since DDoS attack identification can be regarded as a classification problem, more and more intelligent March Learning methods for classification are used in DDoS attack detection in SDN. In which, the K-Nearest Neighbors (KNN) algorithm is a famous statistical method for pattern recognition and one of the best text classification algorithms. It occupies a considerable position in machine learning classification algorithms and is one of the simplest machine learning algorithms. Dong et al. [6] applied an improved KNN algorithm based on Machine Learning to discover the DDoS attack. And Xu et al. [7] also presented a method based on K-means++ and Fast K-Nearest Neighbors for DDoS detection in SDN, which has been verified that improves the detection accuracy and efficiency of KNN, and has high precision and stability of DDoS detection in SDN. KKN is simple and easy to use, easy to understand, but KNN relies heavily on the data set and has relatively high requirements for the accuracy of the data in the training set.

Moreover, the support vector machine (SVM) is a very classical and efficient classification model. The classic SVM itself is very suitable for two classification, many articles have adopted many improved method based on SVM. Meti et al. [8] compared the performance of Bayes, support vector machine and neural network methods in detecting DDoS attacks, and found the performance of SVM is the best. Phan et al. [9] proposed a method to detect DDoS attack by using the SVM-SOM combined model in view of the fuzzy region problem in SVM algorithm, which has improved the detection rate, accuracy, false positives and resource loss. Yang et al. [10] proposed a method to detect

DDoS attack by combining information entropy and SVM algorithm for the campus network scene. Yu et al. [11] designed a platform to efficiently detect and rapidly respond to the DDoS attack in SDN, which determine all flow entry by the trained SVM. In [12], Sahoo et al. developed a model which is assisted by kernel principal component analysis (KPCA) with genetic algorithm (GA). KPCA is used for reducing the dimension of feature vectors, and GA is used for optimizing different SVM parameters.

Nowadays, neural networks have made a comeback and are playing an increasingly important role in machine learning as a new approach. Liu et al. [13] designed a DDoS attack detection scheme based on the combination of generalized information entropy and BP neural network in SDN environment and used the particle swarm optimization algorithm to optimize the BP neural network-related parameters and improve the detection ability. Tang et al. [14] introduced a SDN intrusion monitoring system based on Gate Recurrent Unit—Recurrent Neural Network (RNN), which achieved an accuracy of 89% with 6 original features. Sun et al. [15] proposed a real-time DDoS detection attack method for SDN Controller utilizing the BiLSTM-RNN neural network algorithm to train the data set, and the BiLSTM model is generated to classify the real-time traffic to realize the DDoS attack detection. Novaes et al. [16] presented a system of detection and mitigation of DDoS attacks and Portscan attacks in SDN environments, which is able to efficiently assist in network management, detect and mitigate the occurrence of the attacks. In the work of [17], a deep convolutional neural network (CNN) ensemble framework for efficient DDoS attack detection in SDNs is proposed. The proposed framework is evaluated on a current state-of-the-art Flow-based dataset under established benchmarks and proved to have an improved accuracy. Phan and Park [18] first introduced a new hybrid machine learning model based on SVM and self-organizing map (SOM) algorithms to improve the traffic classification and proposed an enhanced history-based IP filtering scheme to improve the attack detection rate and speed. Ravi et al. [19] proposed a novel mechanism named learning-driven detection mitigation that detects DDoS using a semi-supervised machine-learning algorithm, which achieved an improved accuracy rate of 96.28% in detecting DDoS attack.

In addition, some other advanced methods also have been studied. Assis et al [20] presented a fast SDN defense system against DDoS and port scan attacks, which runs directly into the central controller and uses a game theoretical approach for attack mitigation. Wang et al. [21] developed 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. Yuan et al. [22] designed a QoS-aware mitigation strategy, namely, peer support strategy, which integrates the available idle flow table resource of the whole SDN system to mitigate such an attack on a single switch of the system. Houda et al. [23] proposed a blockchain-based approach, called Cochain-SC, which combines two levels of mitigation, intra-domain and inter-domain DDoS mitigation. Cochain-SC is the first scheme that proposes to deal with both intra-domain and inter-domain DDoS attacks mitigation combining SDN, blockchain and smart contract.

The goal of almost all these methods is to use different strategies and tools to improve the accuracy and real-time performance of attack detection, and some also consider reducing the complexity of the algorithm to reduce the burden on the controller.

3 Methods

3.1 System model and architecture

In this paper, all OpenFlow switches administrated by the same controller are defined as a domain as shown in Fig. 1. In addition, domains are connected to each other by edge switches. Different from the traditional DDoS attack detection method which only relies on the controller, we have developed a novel DDoS attack detection framework based on SDN [24], in which the entire detection process is divided into two different stages, one in the data plane and the other in the control plane. To the best of our knowledge, most SDN switches (e.g., OpenFlow switches) are equipped with one or more CPUs running operating system with abundant computational resource that are currently far from fully utilized. In this case, we can liberate the controller from the heavy traffic by fully utilizing the computing capabilities in switches to undertake part of detection task. At the same time, this idea also draws on the concept of edge computing.

The main structure of the scheme is as shown in Fig. 2. First, we set up the preliminary detection module in the edge switch at the data plane. A lightweight anomaly detection algorithm based on information entropy is designed to monitor the network traffic entering the domain in real time and report to the controller immediately once suspected anomalies are found. Meanwhile, the controller at the control plane guides each switch of the domain to forward data through topological perception, path calculation and flow table update when the network is normal, which is the routine operation at all time. At the same time, only when received a reported abnormal warning from edge switches, the controller launch an attack detection process immediately to distinguish the anomaly more precisely through the flow collecting, the feature extraction and the classification by random forest method. Finally, if this suspected abnormal traffic is confirmed as an attack, the abnormal flow must be dropped through flow table update.

3.2 Detection module of data plane based on entropy

As the gateway of a certain domain in the network, the edge switch can reflect the traffic flow through the network. Therefore, it is more accurate and reasonable to set the preliminary detection module in edge switches. Compared with the traditional centralized attack detection methods, our scheme can make full use of the idle computing resources of each edge switch in the network to find the abnormal traffic timely and notify the controller for further detection and processing immediately.

In SDN, the controller mainly directs the switch to forward data by update a flow table, which contains multiple flow entries. The structure of the flow entry of the OpenFlow1.3 [25] is shown in Fig. 3.

Considering the limited resources of the edge switch, the algorithm design of the preliminary detection module should reduce the complexity as much as possible. Generally, the information entropy can be used to measure the uncertainty of random variables, which can effectively reflect the change of traffic characteristics. The attack detection algorithm based on information entropy has the advantages of real-time, lightweight, low computational cost and high detection rate. Therefore, the preliminary detection algorithm proposed in this paper mainly counts the number of flow entries and calculates the information entropy of destination IP. Assuming $X = (X_1, X_2 \ldots, X_m)$

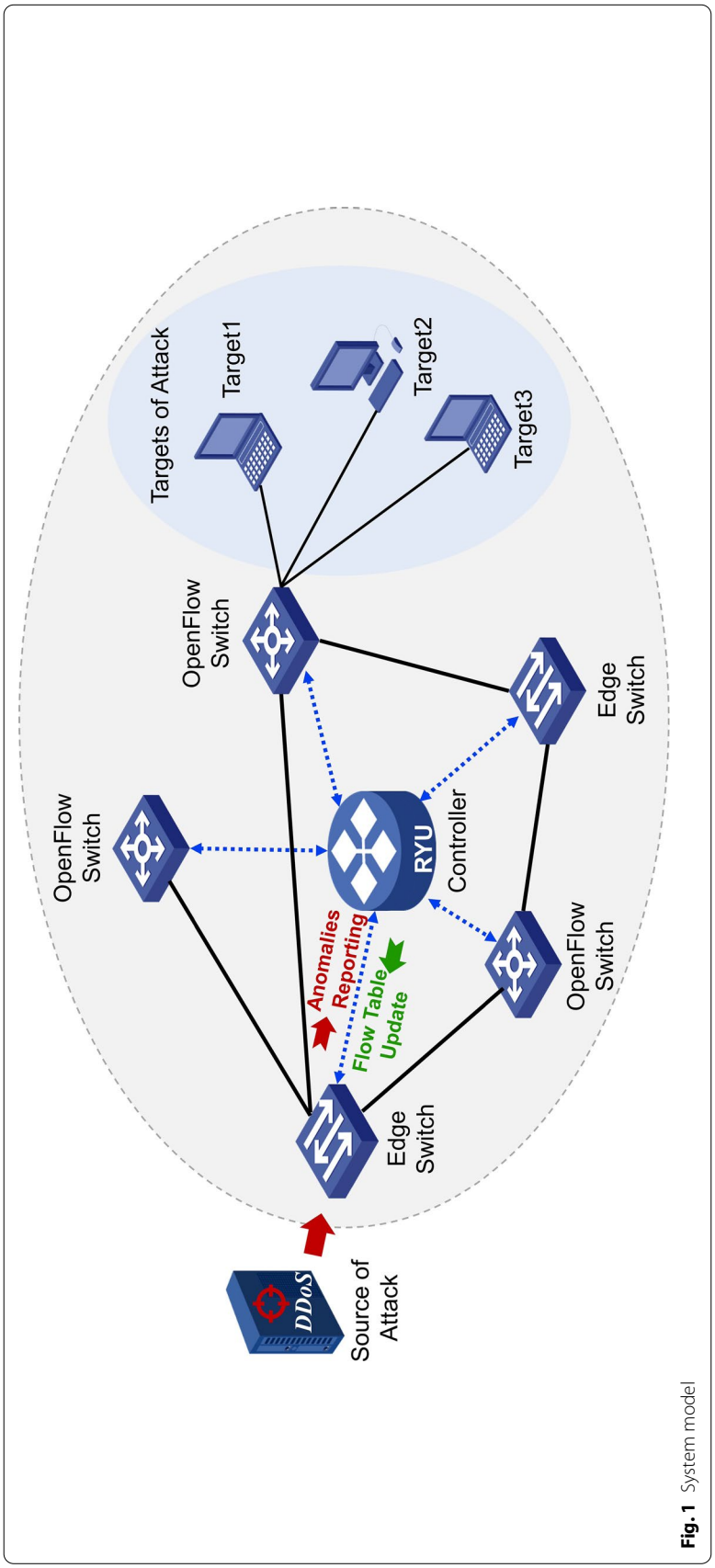

Fig. 1 System model

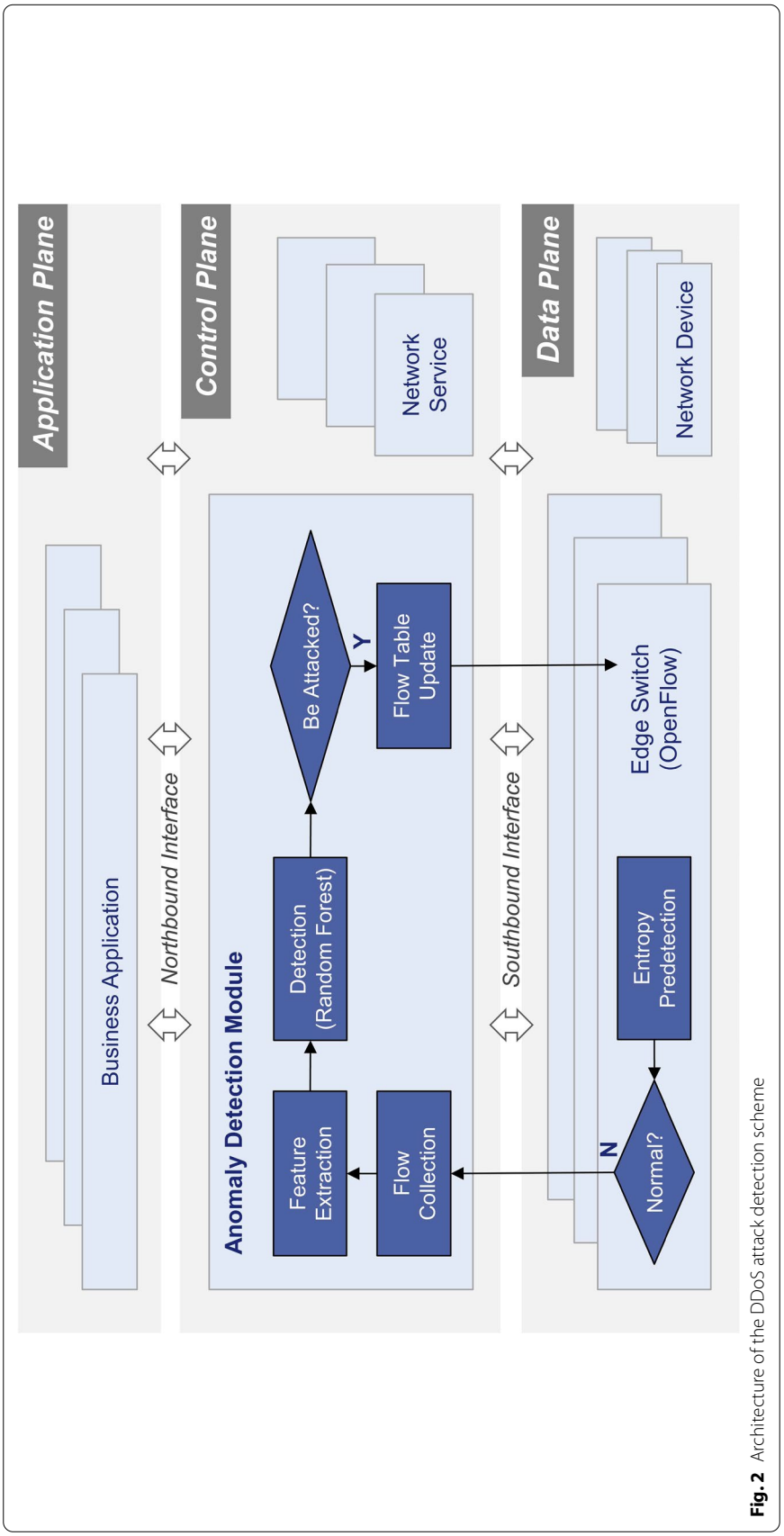

Fig. 2 Architecture of the DDoS attack detection scheme

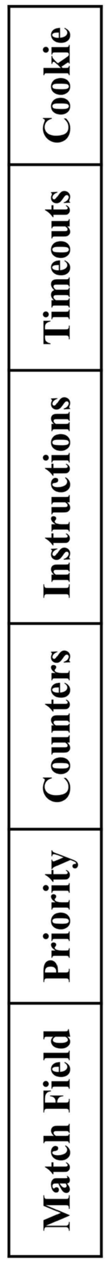

Fig. 3 Structure of the flow entry

constitutes the destination address state space of the edge switch, and X_i represents the number of packets to one same destination IP address within the detection period $\triangle t$, then the probability of the occurrence of packets to this IP address on the edge switch during $\triangle t$ is:

$$p_i = \frac{X_i}{\sum_{i=1}^{m} X_i}. \tag{1}$$

Furthermore, we assume that the normal traffic always exists and m presents the number of active hosts which is always greater than 1, then the normalized information entropy of the destination IP is as follows:

$$H(X) = \frac{-\sum_{i}^{m} p_i \log p_i}{\log m}. \tag{2}$$

The network is divided into normal state and abnormal state, and the corresponding information entropy values are represented as $H_n(X)$ and $H_a(X)$, respectively. Under normal state, the information entropy value will fluctuate up and down in a small range. When a DDoS attack occurs, the traffic with the same IP address will increase sharply, resulting in a smaller entropy value. Therefore, $H_n(X)$ and $H_a(X)$ satisfy the following expression:

$$H_n(X) - H_a(X) > \delta, \tag{3}$$

where the value of δ is determined according to the statistical information entropy under normal network state.

At the same time, considering the occurrence of SYN flooding attack, a large number of packets with forged source addresses will be generated in the network, and the change of the information entropy of the destination IP is usually not obvious at this time, but the number of flow entries in the edge switch will increase sharply. Therefore, when calculating the information entropy of the destination IP, it is also necessary to count the number of flow entries in the switch which is represented as L. Assuming the number of flow entries in normal state is L_n. If $L > L_n$, the network traffic is considered abnormal. In addition, the value of L_n is determined according to the maximum value of flow entries under normal network state.

Algorithm 1 A Fast Anomaly Detection Algorithm Based on Information Entropy in Edge Switch

Input: Data Flow, $\triangle t = 1s$
Output: Anomaly Report
procedure A NOMALY DETECTION IN EDGE SWITCH
  Initialization $H_n(X), L_n, \delta$
top:
  Q $\leftarrow [0, 0, 0]$
start:
  Delay $\triangle t$
  for i=1:m **do**
    Calculate $countL, H(X)$
  end for
  if $H_n(X) - H(X) > \delta$ or $L > L_n$ **then**
    Add (Q , 1)
    if $Num(1, \mathrm{Q}) == 2$ **then**
      Call $attack\_report$
      goto *top*
    else
      goto *start*
    end if
  else
    Add (Q , 1)
    goto *start*
  end if
end procedure

The entire and detailed workflow of the preliminary detection module of the edge switch is shown in Fig. 4. In general, the jitter and latency of traffic in the network range from tens to hundreds of milliseconds, to avoid the interference of network jitter and the statistical error, we set up the interval of traffic detection at edge switches as $\triangle t = 1$ s, at the same time, we also establish a queue Q with the length 3 to record the periodic detection status of the network. As long as there are 2 times to satisfy $H_n(X) - H_a(X) > \delta$ or $L > L_n$ within the adjacent 3 monitoring interval in edge switches, we confirm that there are suspected abnormal traffic in the network. At this moment, the edge switch immediately sends an exception report to the controller and clear the queue Q.

The corresponding pseudo-code for our proposed fast anomaly detection algorithm based on information entropy in the edge switch has been described in Algorithm 1. Each edge switch executes this procedure to monitor the traffic through polling mode.

3.3 Detection module of control plane based on ensemble learning

Based on the preliminary detection module at the edge switch, when no suspected abnormalities are found, the controller just only executes the normal task of guiding switches in the domain to forward data and does not need to detect the traffic status of the network in real time. Therefore, the burden of the controller is effectively reduced. However, the detection method based on information entropy adopted at the edge switch will result in deviation and low accuracy when the network state is unstable, so the controller is required to carry out further precise detection.

In the control plane, we set up an attack detection module on the controller, in which workflow is shown in the control plane part of Fig. 2. When receiving the suspected abnormality warning reported by the edge switch, the controller executes the flow collection submodule immediately and periodically sends the OFPFlowStatsRequest

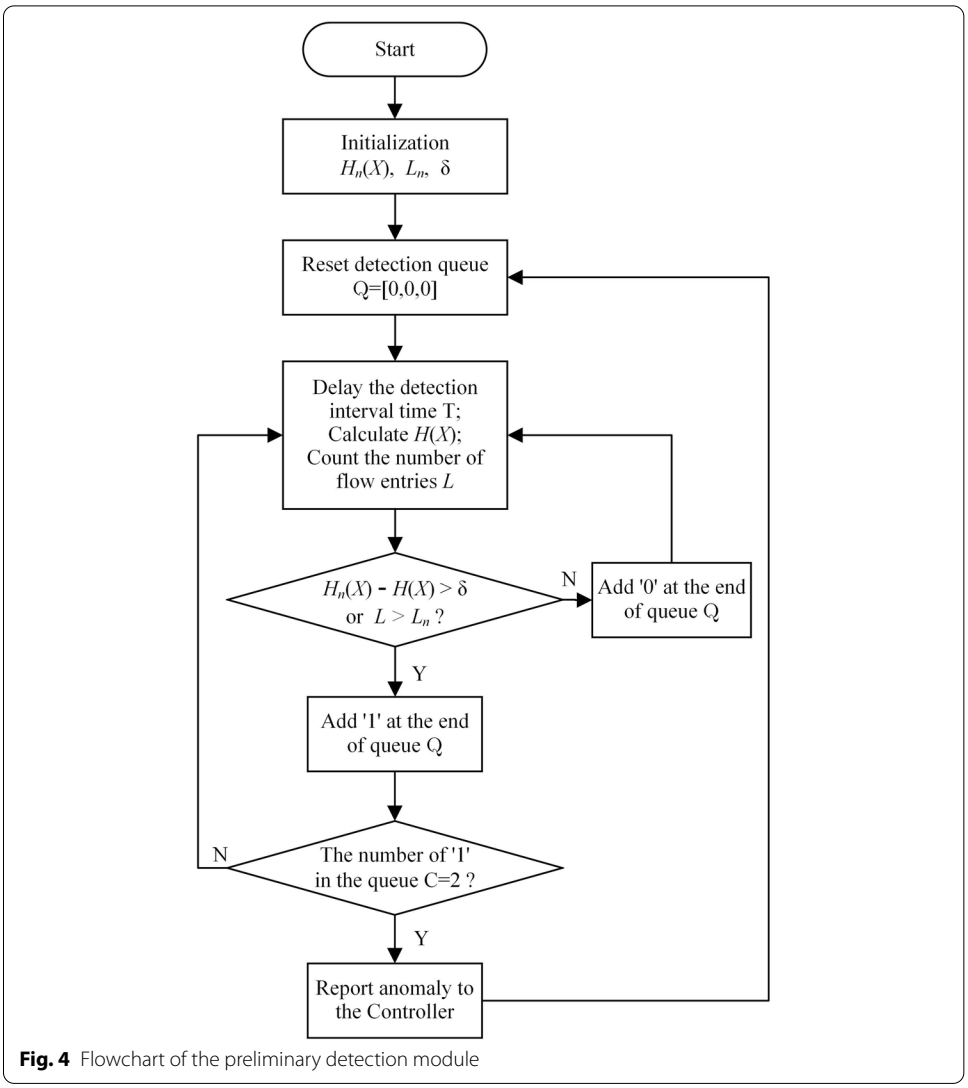

Fig. 4 Flowchart of the preliminary detection module

instruction to all switches, so as to make statistics on the flow number, bit number, packet number, port number and source IP number of the network traffic. Next, the feature extraction procedure is carried out for the network flow information. And using classification model trained by the random forest algorithm, the extracted flow features are classified to accurately conclude whether the flow anomaly exists. Once the conclusion is an abnormal flow, then immediately update the flow table, guiding the corresponding switch to drop the attack packets so as to eliminate the threat.

As mentioned earlier, many current DDoS detection algorithms in single control plane are based on machine learning technology, which has been proved being used as a classifier with high accuracy. In this paper, we also utilize ensemble learning method to make more accurate detection of abnormal flow. First, we need to establish a feature group of the classification model for training and testing, according to the characteristic changes of network traffic when a DDoS attack occurs. Kalkan et al. [5] adopt a 4-feature tuple including four attributes: number of packet, number of byte, duration and protocol for attack detection. While Tang et al. [14] select 6-feature tuple based on their SDN-related

nature without any feature selection algorithms, which are duration, protocol type, number of source bytes, number of destination bytes, service count and rate of the same destination host with the same source port. More literature indicates that increasing the number of features is benefit to the improvement of detection accuracy, but it will also increase the cost of the system greatly.

Based on the consideration of ensuring detection accuracy while minimizing system overhead, we select five most typical features to construct a 5-feature tuple that are average number of packet, average number of packets' bits, growth rate of port, growth rate of flow and growth rate of source IP for subsequent machine learning training and testing. The specific expressions for the 5-feature tuple are shown in Table 1, where T represents the traffic collection cycle, *sum_pkt_num* represents the total number of packets, *sum_pkt_bits* represents the total number of packets' bits, *sum_flows* represents the total number of flows, *sum_ports* represents the total number of ports, and *sum_sip* represents the total number of source IP.

These five features are determined by the parameters which have obvious differences between the values in the normal package and the attack package through the experimental comparison. (1) *avg_pkt_num* represents the average number of packets. DDoS attackers usually launch an attack on an illegal IP within a short time; thus, the flow generation speed will increase considerably, and the number of packets per flow will decrease. (2) *avg_pkt_bits* represents the average number of bits. In order to generate a large number of attack flows in a short period of time, the number of bits per attack packet will be very small, even almost no content, then the average number of packet bits of the flow will be reduced substantially. (3) *rate_port* represents the growth rate of the port. Normally port changes in the network will be relatively stable, but in the case of DDoS attack, port numbers will be generated randomly, which will increase the port growth rate obviously. (4) *rate_flow* represents the growth rate of flows. As the network is normal, the change of data flow is relatively stable. Once an attack occurs, a large number of attack flows are generated in the network rapidly, and the growth rate of the flow will increase significantly. (5) *rate_sip* represents the growth rate of source IP. During the attack, a large number of packets with forged IP addresses will be generated, making the growth rate of source IP increase greatly.

Random forest is a classifier containing multiple decision trees in ensemble learning, and the output results are determined according to the score formed by the number of decision tree votes [26]. Compared with other machine learning algorithms, random forest algorithm is a very convenient and practical algorithm which is more suitable for multivariate classification with less resource consumption and fast training speed. Moreover, in the training process, the mutual influence between features can be detected.

Therefore, considering the adopted multiple feature tuple and requirement of less overhead in the detection process, we utilize the random forest algorithm to further detect the suspicious flow. First, a virtual network topology was built on mininet, and scapy [27] and nettich-ng were used to generate normal traffic and attack traffic, respectively. We collect 13920 traffic records as the data set, which specific composition is shown in Table 2.

In our modeling process, the bagging sampling method [28] was exploited to randomly select multiple training subsets from the original training set, while the CART algorithm

was leveraged to generate K decision trees to form the random forest according to the principle of minimum impurity. Select the features that maximize the decrease in the Gini coefficient of the current node to split. When the decision tree reaches the maximum depth or the number of samples per leaf node reaches the preset lower limit, the random forest model construction is completed. The final anomaly decision was determined by voting the results of K trees in the test set. Therefore, the test accuracy of the trained classification model on the test set is 0.997, indicating that this classification model has a very high accuracy for the detection of attack traffic.

4 Experiment and discussion

To evaluate the performance of our proposed DDoS attack detection framework based on SDN, we use mininet2.2.2 to build the network topology as shown in Fig. 5. And the testbed hardware parameters are as follows: CPU: Intel(R) Core(TM) i7-7700k@4.20 GHz 8 Cores, memory: 8 GB DDR4-2400 MHz RAM, operating system: Ubuntu18.04, controller: Ryu [29] open source controller. The experimental network is composed of one controller and 7 switches s1–s7, among which s1 is the edge switch, and s1, s4, s5, s6 and s7 are connected to 3 hosts, respectively, with numbers from h1 to h15. During the experiment, scapy was used to inject normal traffic into the network as the background traffic, and then a DDoS attack was launched from the switch s1 to the host h15. According to the history statistics, the information entropy value of the background traffic is about 0.8 under the normal network state. Therefore, we set $H_n(X) = 0.8$ and $\delta = 0.2$ in the simulation.

We carry out experiments on the two most common methods of DDoS attack, ICMP flood attack [30] and SYN flood attack [31], respectively. At the edge switch, the number of flow entries and the information entropy are all counted in real time, and the flow situation of the attacked port is monitored by the wireshark tool.

4.1 Experiment of ICMP flood attack

When an ICMP flooding attack occurs, a large number of packets with a specific destination IP are generated in a short period of time [32]. As shown in Fig. 6a, under normal circumstances, the information entropy difference of the destination IP is below 0.2. While at the 20 s, DDoS attack is launched, and the information entropy difference of the destination IP increases rapidly. At the same time, it is found from Fig. 6b that the number of flow entry does not change significantly after the attack occurs. According to the preliminary detection algorithm of the edge switch, the decision result is shown in Fig. 6c. From 22 s, the edge switch reports abnormalities to the controller.

To demonstrate the advantage of response time in our proposed cooperative detection scheme, we compare with a traditional centralized scheme which does not install the preliminary detection module in the edge switch, but the detection method on the controller is the same as our scheme in the control plane. The difference is that the controller needs to make real-time statistics of the traffic in the network all the time and carry out abnormal attack detection. As can be observed from the comparison of the flow from the attacked port of the two schemes in Fig. 7, in the ICMP flooding attack mode, compared with the traditional centralized scheme, the attack response processing time of proposed scheme is shortened by 1 s.

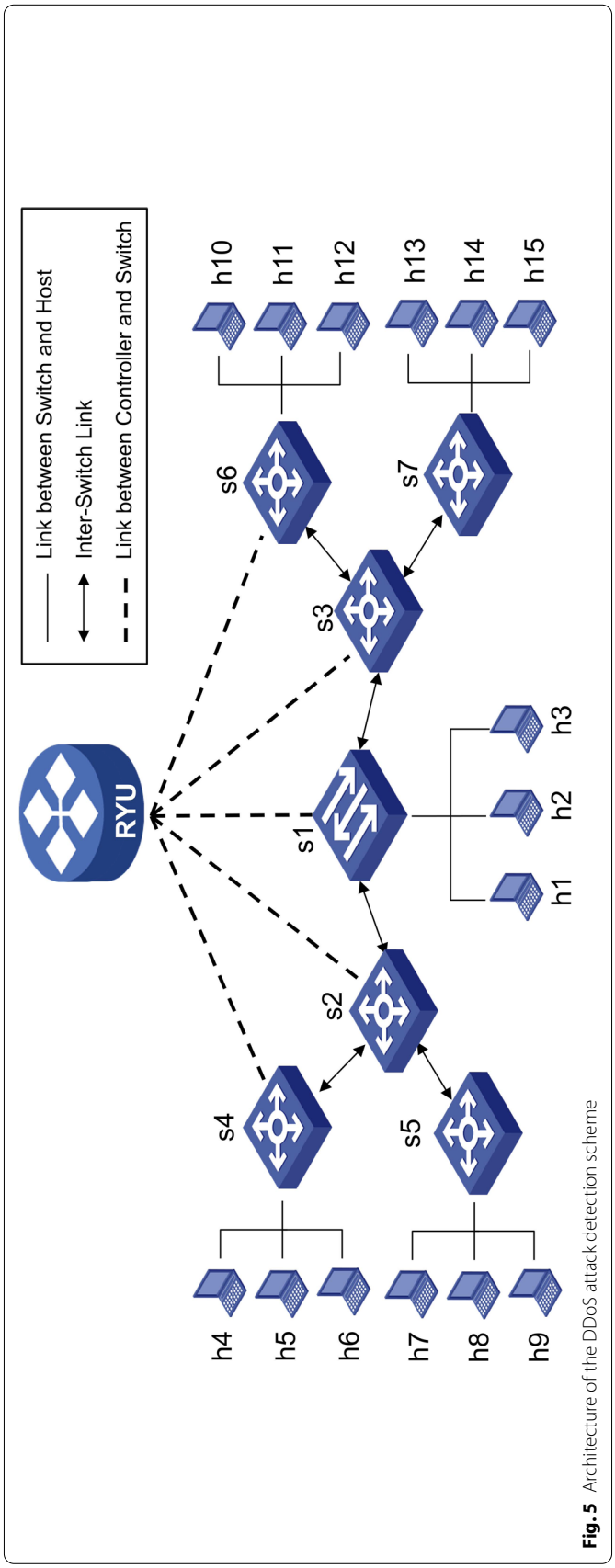

Fig. 5 Architecture of the DDoS attack detection scheme

Table 1 5-feature tuple

Name of feature	Notation
Average number of packets	$avg\_pkt\_num = \frac{sum\_pkt\_num}{sum\_flows}$
Average number of bits	$avg\_pkt\_bit = \frac{sum\_pkt\_bits}{sum\_flows}$
Growth rate of port	$rate\_port = \frac{sum\_ports}{T}$
Growth rate of flow	$rate\_flow = \frac{sum\_flows}{T}$
Growth rate of source IP	$rate\_sip = \frac{sum\_sip}{T}$

Table 2 Data set of flows

Data set	Flow type	Number of flows	Proportion
All	NormalAttack	(69276993)13920	1
Training set	NormalAttack	(46184662) 9280	2/3
Test set	NormalAttack	(23092331) 4640	1/3

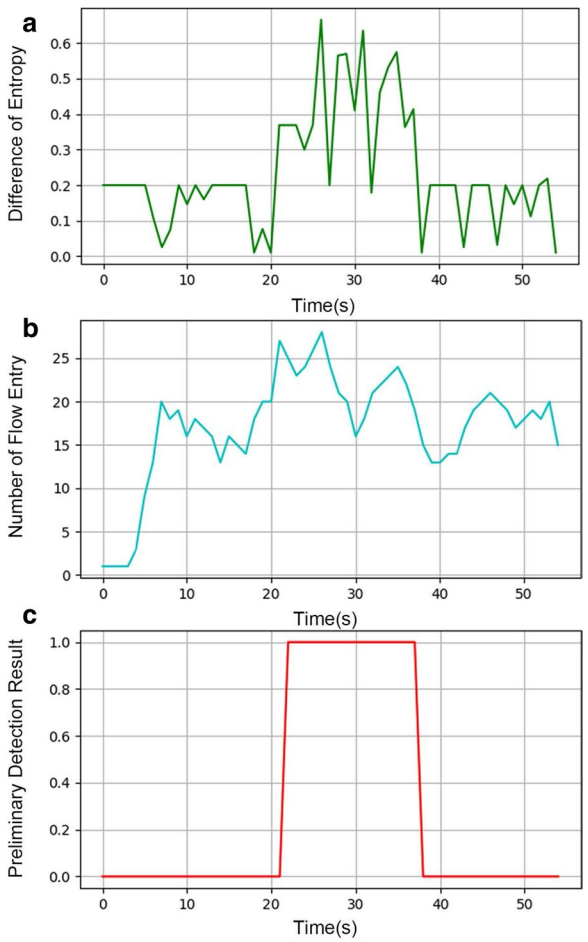

Fig. 6 Preliminary detection result in the edge switch (ICMP). **a**: Real-time results of information entropy difference. **b**: Real-time statistic results of flow entries. **c**: Detection decision in the edge switch (0 represent normal, 1 represent abnormal)

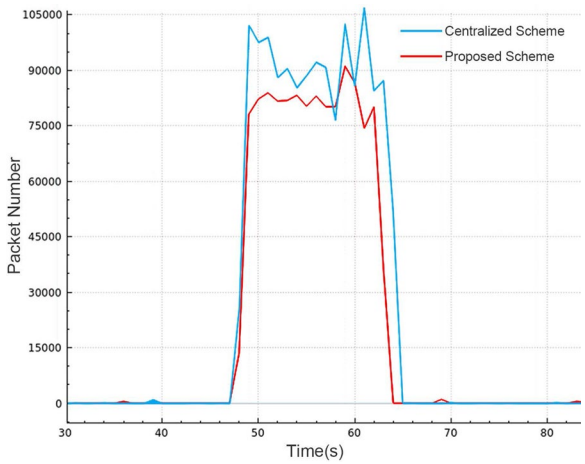

Fig. 7 Flow comparison of attacked port (ICMP)

4.2 Experiment of SYN flood attack

When the SYN flood attack occurs, a large number of packets with forged source addresses will be generated in the network. In this case, the difference of information entropy of the destination IP is not abnormal as shown in Fig. 8a, and to be noted that at 26 s is a jitter in the network, which has been eliminated in our designed preliminary detection algorithm our designed. However, the number of flow entry in edge switches will increase sharply as shown in Fig. 8b. SYN attack is launched at 20 s, according to the preliminary inspection algorithm of the edge switch, the decision result is shown in Fig. 8c. At 22 s, the edge switch reports the exception to the controller.

Received the warning from the edge switch, the controller starts further detection and confirmed that the DDoS attack occurs, then update the flow table to the switches immediately to drop the attack packets. As shown in Fig. 9, after 15 s (47–62 s) from the attack occurs, the attacked port returns to normal. Similarly to the ICMP attack, compared with the traditional centralized scheme, the attack response time of the proposed scheme is reduced by 4 s. To be noted that the simulations in Figs. 7 and 9 are generated by the wireshark tool, so the time of the horizontal axis was not synchronized with the time in Figs. 6 and 8.

In summary, the proposed cooperative detection framework can effectively cope with DDoS attacks. The ICMP and SYN experiment results all demonstrate that the detection process of proposed scheme is faster than the traditional centralized method.

4.3 Evaluation of controller overhead

In order to evaluate the improvement of controller overhead brought by the proposed scheme, the controller CPU utilization of the proposed scheme is compared with that of a centralized scheme. In the centralized scheme we use to compare, the controller is continuously running the same precise detection procedure of our proposed scheme in a polling manner on the control plane, and at the same time there is no any other extra processing in the edge switch on the data plane.

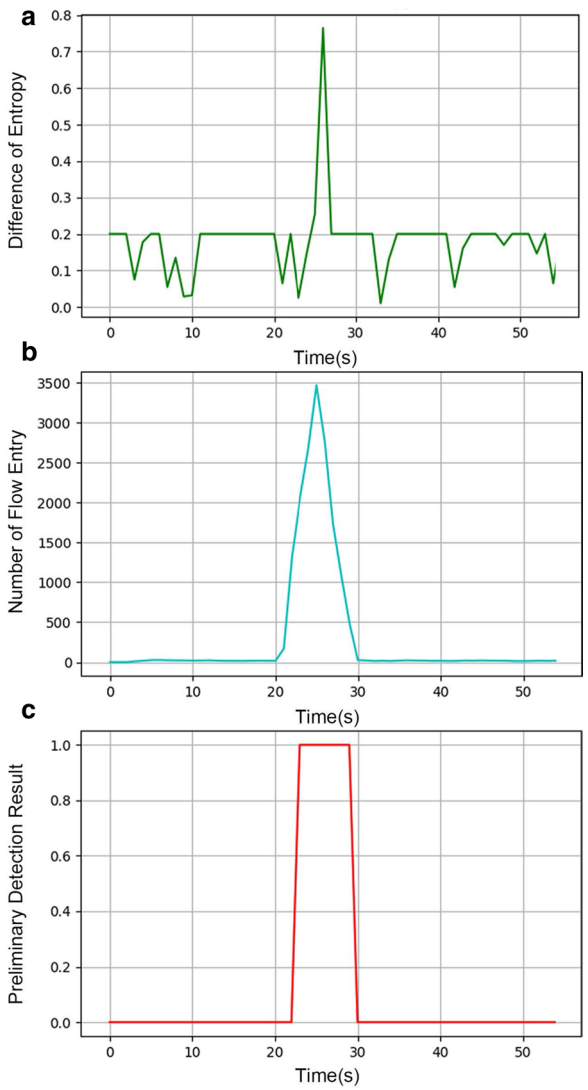

Fig. 8 Preliminary detection result in the edge switch (SYN). **a**: Real-time results of information entropy difference. **b**: Real-time statistic results of flow entries. **c**: Detection decision in the edge switch (0 represent normal, 1 represent abnormal)

First, when the ICMP flood attack occurs, there is no new source or destination address in the network, just a large number of forged ICMP packets blocking the link. Because there are already flow entries in the switch to guide the forwarding, the switch does not need to ask the controller for directing, as a result, the CPU utilization of the controller will not have a too large fluctuation. As shown in Fig. 10, the attack is launched at 7 s, and the CPU utilization of both schemes increased, but the average CPU utilization of the proposed scheme is obviously lower than that of the traditional centralized scheme.

Second, when SYN flood attack occurs, a large number of packets with pseudo-source addresses will appear in the network, and the switch needs to report to the controller for guiding. Therefore, as shown in Fig. 11, when SYN flood attack is

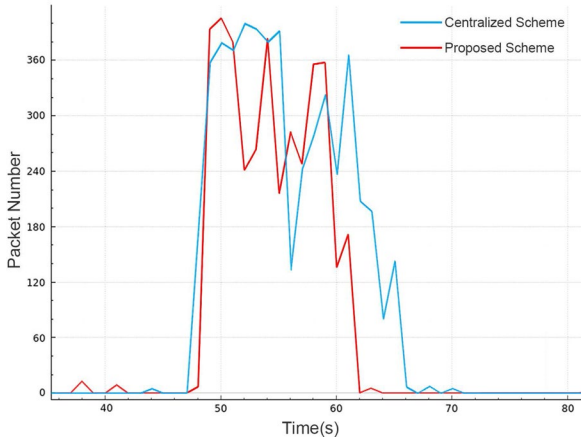

Fig. 9 Flow comparison of attacked port (SYN)

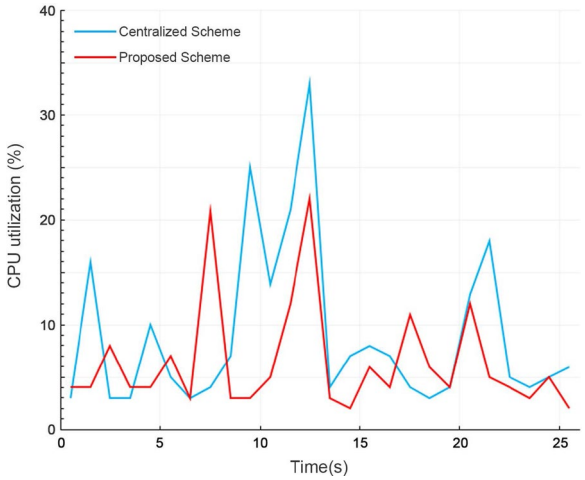

Fig. 10 CPU utilization of the controller (ICMP)

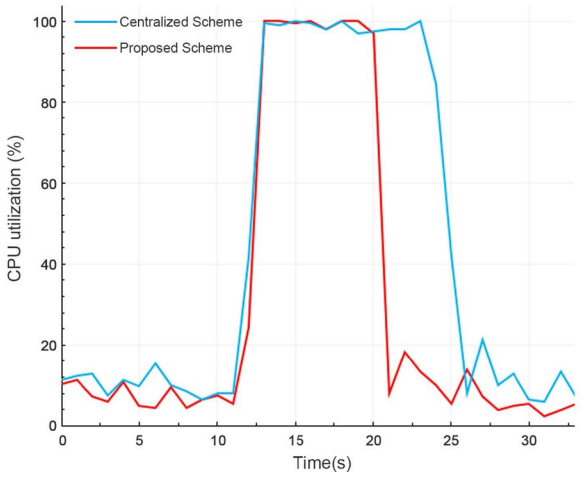

Fig. 11 CPU utilization of the controller (SYN)

launched, the CPU utilization of the controller increases significantly, up to 100%. Meanwhile, in the proposed scheme, the controller only starts the detection module after receiving the abnormal report from the edge switch, which cause that the CPU occupancy rate is significantly lower than that of the traditional centralized scheme when the attack does not occur. In addition, the duration of the peak CPU occupancy rate in proposed scheme is 5 s shorter than the traditional centralized scheme.

Finally, in order to further verify the advantages of the proposed scheme in alleviating controller overhead when the network scale increases, we extend the scale of the network based on the topology in Fig. 5. Limited by the hardware of the test platform, we only used an about twice scale network for testing. We established a tree topology network with a depth of 4, consisting of 15 switches and 30 hosts, and only inject normal traffic to the network. Next, we separately count the CPU utilization of the two schemes. The total statistical time is 65 min, and the average of the CPU utilization is taken every 5 min for comparison.

Figure 12 illustrates that the proposed scheme can reduce the burden of controller more obviously after the network scale is increased. As the scale of the network grows, the advantages of our framework will become more significant in terms of controller overhead.

5 Conclusion

In this paper, a cooperative DDoS attack detection method based on entropy and ensemble learning in SDN is proposed. In the data plane, the preliminary inspection module is set up on the edge switch to collect real-time statistics of network traffic information, and the controller will be noticed when abnormalities are found through the designed fast detection algorithm. And in the control plane, the precise attack detection module is developed on the controller, in which the five-element feature group aiming at the characteristics of DDoS attack is constructed, and the random forest algorithm is used to further identify the abnormal traffic. Finally, once the attack traffic is confirmed, the controller immediately delivers a dropping packet command to the edge switch through flow table update, thus the attack will be blocked. This cooperative scheme innovatively

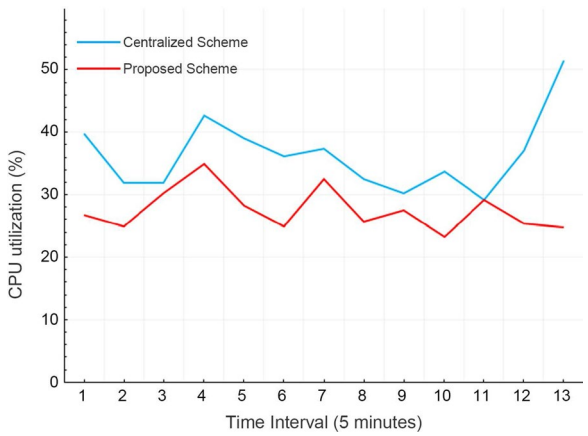

Fig. 12 CPU utilization of the controller in the larger scale network (normal traffic state)

utilized the idle computing power of the edge switch to offload some of the detection tasks from the control plane to the data plane. The simulation results of ICMP and SYN flood attack show that our method can detect DDoS attack fastly and effectively. At the same time, with the increase in the network scale, this strategy can more effectively reduce the CPU utilization of the controller and shorten the duration of the peak CPU utilization.

Abbreviations
DDoS: Distributed denial of service; SDN: Software-defined network; KNN: K-nearest neighbors; SVM: Support vector machine; KPCA: Kernel principal component analysis; GA: Genetic algorithm; CNN: Convolutional neural network; SOM: Self-organizing map; SGS: Safe-guard scheme.

Authors' contributions
All authors have contributed to this research work. SY and JZ conceived the idea and performed the experiments, SY, JZ and XZ analyzed the data. SY, JZ and JL wrote the manuscript. YL and TX reviewed and edited the manuscript. All authors read and approved the final manuscript.

Funding
This work was supported by the National Key R&D Plan of China under Grant No. 2017YFC0803400 and the National Natural Science Foundation of China under Grant No. 62071275.

Declarations

Competing interests
The authors declare that they have no competing interests.

Author details
[1] School of Information Science and Engineering, Shandong University, Qingdao, China. [2] NetEase D&R Center Lab, Hangzhou, China.

Received: 19 January 2021 Accepted: 23 March 2021
Published online: 13 April 2021

References

1. Q. Yan, F.R. Yu, Q. Gong, 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. Surv. Tutor. **18**(1), 602–622 (2016)
2. S. Li, Y. Cui, Y. Ni, L. Yan, An effective SDN controller scheduling method to defence DDoS attacks. Chin. J. Electron. **28**(2), 404–407 (2019)
3. S. Lim, S. Yang, Y. Kim, S. Yang, H. Kim, Controller scheduling for continued SDN operation under DDoS attacks. Electron. Lett. **51**(16), 1259–1261 (2015)
4. J. Zheng, Q. Li, G. Gu, J. Cao, D.K.Y. Yau, J. Wu, Realtime DDoS defense using cots SDN switches via adaptive correlation analysis. IEEE Trans. Inf. Forensics Secur. **13**(7), 1838–1853 (2018)
5. K. Kalkan, L. Altay, G. Gür, F. Alagöz, Jess: joint entropy-based DDoS defense scheme in SDN. IEEE J. Sel. Areas Commun. **36**(10), 2358–2372 (2018)
6. S. Dong, M. Sarem, Ddos attack detection method based on improved KNN with the degree of DDoS attack in software-defined networks. IEEE Access **8**, 5039–5048 (2020)
7. Y. Xu, H. Sun, F. Xiang, Z. Sun, Efficient DDoS detection based on k-fknn in software defined networks. IEEE Access **7**, 160536–160545 (2019)
8. N. Meti, D.G. Narayan, V.P. Baligar, Detection of distributed denial of service attacks using machine learning algorithms in software defined networks, in *2017 International Conference on Advances in Computing, Communications and Informatics (ICACCI)* (2017)
9. T.V. Phan, N.K. Bao, M. Park, A novel hybrid flow-based handler with DDoS attacks in software-defined networking, in *2016 International IEEE Conferences on Ubiquitous Intelligence Computing, Advanced and Trusted Computing, Scalable Computing and Communications, Cloud and Big Data Computing, Internet of People, and Smart World Congress (UIC/ATC/ScalCom/CBDCom/IoP/SmartWorld)* (2016), pp. 350–357
10. L. Yang, H. Zhao, Ddos attack identification and defense using SDN based on machine learning method, in *2018 15th International Symposium on Pervasive Systems, Algorithms and Networks (I-SPAN)* (2018), pp. 174–178
11. Y. Yu, L. Guo, Y. Liu, J. Zheng, Y. Zong, An efficient SDN-based ddos attack detection and rapid response platform in vehicular networks. IEEE Access **6**, 44570–44579 (2018)
12. K.S. Sahoo, B.K. Tripathy, K. Naik, S. Ramasubbareddy, B. Balusamy, M. Khari, D. Burgos, An evolutionary SVM model for DDoS attack detection in software defined networks. IEEE Access **8**, 132502–132513 (2020)
13. Z. Liu, Y. He, W. Wang, B. Zhang, DDoS attack detection scheme based on entropy and PSO-BP neural network in SDN. China Commun. **16**(7), 144–155 (2019)

14. T.A. Tang, L. Mhamdi, D. McLernon, S.A.R.M. Zaidi, Ghogho, Deep recurrent neural network for intrusion detection in SDN-based networks, in *2018 4th IEEE Conference on Network Softwarization and Workshops (NetSoft)* (2018), pp. 202–206
15. W. Sun, Y. Li, S. Guan, An improved method of DDoS attack detection for controller of SDN, in *2019 IEEE 2nd International Conference on Computer and Communication Engineering Technology (CCET)* (2019), pp. 249–253
16. M.P. Novaes, L.F. Carvalho, J. Lloret, M.L. Proença, Long short-term memory and fuzzy logic for anomaly detection and mitigation in software-defined network environment. IEEE Access **8**, 83765–83781 (2020)
17. S. Haider, A. Akhunzada, I. Mustafa, T.B. Patel, A. Fernandez, K.R. Choo, J. Iqbal, A deep CNN ensemble framework for efficient DDoS attack detection in software defined networks. IEEE Access **8**, 53972–53983 (2020)
18. T.V. Phan, M. Park, Efficient distributed denial-of-service attack defense in SDN-based cloud. IEEE Access **7**, 18701–18714 (2019)
19. N. Ravi, S.M. Shalinie, Learning-driven detection and mitigation of DDoS attack in IoT via SDN-cloud architecture. IEEE Internet Things J. **7**(4), 3559–3570 (2020)
20. M.V.O. De Assis, M.P. Novaes, C.B. Zerbini, L.F. Carvalho, T. Abrãao, M.L. Proença, Fast defense system against attacks in software defined networks. IEEE Access **6**, 69620–69639 (2018)
21. Y. Wang, T. Hu, G. Tang, J. Xie, J. Lu, SGS: safe-guard scheme for protecting control plane against DDoS attacks in software-defined networking. IEEE Access **7**, 34699–34710 (2019)
22. B. Yuan, D. Zou, S. Yu, H. Jin, W. Qiang, J. Shen, Defending against flow table overloading attack in software-defined networks. IEEE Trans. Serv. Comput. **12**(2), 231–246 (2019)
23. Z. Abou El Houda, A.S. Hafid, L. Khoukhi, Cochain-SC: an intra- and inter-domain DDoS mitigation scheme based on blockchain using SDN and smart contract. IEEE Access **7**, 98893–98907 (2019)
24. J.C. Zhang, Research on DDoS attack defense mechanism based on SDN. Ph.D. thesis, Shandong University (2020)
25. V. Šulák, P. Helebrandt, I. Kotuliak, Performance analysis of openflow forwarders based on routing granularity in openflow 1.0 and 1.3, in *2016 19th Conference of Open Innovations Association (FRUCT)* (2016), pp. 236–241
26. Y. Wang, S. Xia, Q. Tang, J. Wu, X. Zhu, A novel consistent random forest framework: Bernoulli random forests. IEEE Trans. Neural Netw. Learn. Syst. **29**(8), 3510–3523 (2018)
27. R.S. Rohith, R. Rohith, M. Minal, G. Shobha, Scapy—a powerful interactive packet manipulation program, in *2018 International Conference on Networking, Embedded and Wireless Systems (ICNEWS)* (2018), pp. 1–5
28. L. Breiman, Bagging predictors. Mach. Learn. **24**, 123–140 (1996)
29. S. Asadollahi, B. Oswami, M. Sameer, Ryu controller's scalability experiment on software defined networks, in *2018 IEEE International Conference on Current Trends in Advanced Computing (ICCTAC)* (2018), pp. 1–5
30. N. Gupta, A. Jain, P. Saini, V. Gupta, DDoS attack algorithm using ICMP flood, in *2016 3rd International Conference on Computing for Sustainable Global Development (INDIACom)* (2016), pp. 4082–4084
31. L. Arshadi, A.H. Jahangir, Entropy based SYN flooding detection, in *2011 IEEE 36th Conference on Local Computer Networks* (2011), pp. 139–142
32. C. Li, H. J. Yang, F. Sun, J. M. Cioffi, L. Yang, Multiuser overhearing for cooperative two-way multiantenna relays. IEEE Trans. Vehi. Tech. **65**(5), 3796–3802 (2016)

Publisher's Note