An Intelligent SDN DDoS Detection Framework

Xiang Zhang(✉), Chaokui Zhang, Zhenyang Zhong, and Peng Ye

University of Electronic Science and Technology of China, Chengdu, China
zhangx@uestc.edu.cn

Abstract. With the development and popularity of computer networks, more and more devices, services and applications are running on the Internet. While it is convenient to the public, more security problems have also brought to the public. Distributed Denial of Attack (DDoS) is just one of the most difficult malicious attacks. It has many different attack forms, causing high damages of services, and is usually hard to detect and defend against. However, the development of Software Defined Networking (SDN) brought new possibilities, due to abilities of global awareness and centralized control. This paper proposes an intelligent SDN DDoS detection framework. In this framework, a security-oriented flow monitoring and sampling algorithm with low-latency is proposed. Meanwhile, we designed a service-oriented recognition model SC-VAE for packet classification. This model combines spectral clustering and variational auto-encoder to detect abnormal traffic by identifying normal streams. It is adaptive to hybrid DDoS attacks, and has a certain predictive effect for unknown DDoS attack not involved in training datasets. Simulation results demonstrate the effectiveness of the proposed framework.

Keywords: SDN · DDoS detection · Network status sampling

1 Introduction

The development of network technology is also accompanied by new threats. With the development of edge computing, Internet of Things (IOT), wireless and other technologies, the demand for network quality and network security are also increasing. The demand for the network is increasing, and many new types of attack are emerging in an endless stream. Many new technologies can also be used in network technology. Time-series content requests and update edge caching accordingly can be predicted by bidirectional deep recurrent neural network (BRNN) model to reduce the network pressure [1]. SDN can also promote the realization of wireless network virtualization to improve the efficiency of resource allocation [12]. However, traditional attacks, DDoS attacks, is rather destructive and difficult to defense. DDoS attack take usage of system software vulnerabilities. It is easy to perform but hard to prevent, quite different

H. Gao and X. Wang (Eds.): CollaborateCom 2021, LNICST 406, pp. 333–347, 2021.
https://doi.org/10.1007/978-3-030-92635-9_20

from the ordinary infiltration attack. The main goal of this attack is to consume the resources of the network equipment, so that the target equipment cannot provide services normally.

Software Defined Networking (SDN) is a new type of network architecture [8]. SDN separates the control plane and forwarding plane in traditional network, and introduces a network controller to manage the lower-level forwarding equipment generally. SDN uses southbound protocols, as OpenFlow, to dynamically obtain network topology, calculate routing information, send routing instructions to network devices, and eventually obtain network status data such as switch ports and flow tables in real time. Therefore, real-time monitoring and routing of network traffic can be realized at the controller level. Compared with traditional networks, it has a better perception of network traffic changes, what is more, it can control network traffic more flexibly. This can making it easier to detect and protect against DDoS attacks.

The first step to defend against DDoS attacks is to perceive real-time status of the network in SDN. By using OpenFlow, the controller can periodically send requested messages of flow table, and then dynamically obtain flow table information forwarded by the OpenFlow switch. With the analysis of real-time network status, the DDoS attacks can be discovered. The existing methods of detection DDoS attacks are mainly divided into two categories: the first one is attack traffic detection based on entropy or threshold [3], and the second one relies on machine learning and deep learning algorithms to distinguish between attack traffic and normal traffic [4,5,11]. In addition, there is also a multi-level attack detection method that combines entropy and deep learning algorithm [7]. On one hand, by using entropy, the algorithm can achieve a fast recognition speed, which can reduce resource consumption, but with a relatively low accuracy. On the other hand, detection accuracy of using deep learning is very high while it consumes more network resources due to high computational complexity.

These existing approaches that use SDN to defend DDoS attacks are to collect data from all switches in the entire network. However, repeated network flow data will be collected in this way, which will cause a waste of network bandwidth resources. In order to deal with that drawback, the proposed approach in this paper is to collect the flow table data of specific switches by combining the flow sampling algorithm to optimize the resource consumption of the DDoS DDoS detection framework in the early stage of collecting flow information. Also, this approach uses machine learning technology to establish a service-oriented traffic recognition model and a service flow passlist via that model. It becomes possible to recognize more types of DDoS attacks and emerging types of DDoS attacks whit its high generalization ability.

In summary, The main contributions of this paper are as follows:

1. Propose a SDN DDoS detection framework, including:
   (1) An optimization model for traffic monitoring and sampling, in order to reduce the resource consumption when collecting network stream information;

(2) A service flow-oriented classification and identification model, adaptive to hybrid DDoS attacks and potential unknown DDoS attacks.
2. Set up a simulation environment, evaluate and compare the performance of the proposed framework with some latest solutions.

2 Related Works

At present, the main purpose of many researches using SDN for network monitoring is to monitor various link information or message information transmitted in SDN through the ability of centralized control. In SDN, the OpenFlow protocol only provides a message interface to access the flow table of the switch. The paper [9] points out that all approaches of network monitoring are implemented in SDN is based on flow tables, and approaches based on messages haven't been implemented yet. Therefore, an extension package for packet sampling is proposed: FleXam, FleXam extends the OpenFlow protocol and randomly samples a specified number of packet with a certain probability. There are also many network monitoring methods optimized for resource consumption in network monitoring [2,10]. Specific optimization indicators have communication costs. For example, FlowCover [10] established the integer programming problem to minimize the payload of the communication. Using greedy algorithm, it preferentially selected the switch with a small payload consumption load for collection, so as to realize the flow table sampling with low communication payload. The paper [2] proposes a compression algorithm for network monitoring in a hybrid SDN network. Based on the SVD algorithm and linear regression model, this method can replace the link information of all switch links by collecting the link information of a subset of the switches, thereby achieving the purpose of compressing the collected data.

DDoS detection methods in SDN environments use artificial intelligence algorithms such as SVM, deep learning, and random forest to make network attack defense more intelligent [4,5,11]. There is also a simple entropy-based method for identifying abnormal traffic. The former has better results in the accuracy method, while the latter focuses on the minimum resource consumption, For example, Conti M, Lal C, etc. proposed a lightweight method to protect against DDoS attacks in SDN [3], The algorithm proposed by this method can prevent two types of DDoS attacks, routing spoofing and resource consumption. Among them, routing spoofing detection is divided into IP spoofing detection and MAC spoofing detection. Resource consumption detection is divided into two asynchronous detection modules. The first module uses the entropy method to detect random changes in the packet to determine whether there is a DDoS attack. The second module sets up a packet inspection table to determine whether there is a DDoS attack by counting the average number of packets received from a source MAC within a certain time interval. Liu Z, He Y, Wang Wd combined entropy and DNN algorithm to carry out DDoS detection, which was achieved good results [7], the method using entropy detection method for detection firstly. If abnormal flow is found, then the flow will be detected again by particle swarm

optimization neural network. Therefore, on the basis of ensuring the rapid detection speed, the accuracy of detection is also improved through the neural network. Kübra, Kalkan, Levent proposed a shared entropy method JESS [6] that can detect and mitigate DDoS attacks. This method can not only detect known DDoS attacks, but also has good accuracy in detecting unknown DDoS attacks. At the same time, this method selects the optimal packet attributes through strategies to reduce the bandwidth resources consumed in the packet collection phase.

3 Security-oriented Flow Monitoring and Sampling

3.1 Performance Analysis of Security-Oriented Flow Table Sampling

In SDN, the most commonly used method for network status awareness is to use the OpenFlow protocol to sample the flow table of forward switches. This can collect current network flow information in the network through the Match field in the flow table. The additional packet payload required to monitor the network in an attack detection framework will have a significant impact on framework resources. Due to this phenomenon, we analyzed the packet payload required for flow table sampling. Suppose the topology of the entire network is graph G, and the set of nodes is V, Each node is mapped to an OpenFlow switch s_i, $s_i \in S,\ i = 1, 2, ..., n$. Define the flow set existing in the network as F, $f_i \in F,\ i = 1, 2, ...m$. Each flow can be uniquely confirmed by the OpenFlow switch sequence P_i that the route passes through, and $P_i = \{s_{i_1}, s_{i_2}, s_{i_3}, ..., s_{i_l}\},\ i_q \in [1, n],\ q \in [1, l]$. Let the length of the OpenFlow request flow table packet be L_{req}. The length of each flow table entry is L_{entity}. Because the sampled data is the flow table response, if a switch node s_i has k_i flows through, then at least the packet size $L_{payload_i}$ required to sample the flow table of the switch is:

$$L_{payload_i} = L_{req} + k_i * L_{entity} \tag{1}$$

To get all the flow data in the network, the easiest way is to traverse all the switches to get their flow table data, but this will cause the problem of repeated sampling. Zhiyang Su et al. proposed the FlowCover algorithm [10] to find the smallest set of switches that can collect all flows. This method reduces the number of flow table request messages sent, so that each flow table request brings back as many independent flow table entries as possible. However, from the perspective of monitoring, the polling communication mode of OpenFlow protocol request response to obtain the flow table statistics is indeed a waste of network resources. On the contrary, the subscription notification mode can better realize the periodic flow monitoring tasks. This communication mode only requires the controller to send a subscription message and set the subscribed data type. The OpenFlow switch periodically returns flow table data to the controller, significantly reducing the amount of monitoring packets transmitted

in the network. Depending on the subscription notification mode, the flow table sampling and transmission payload of each switch s_i is $L_{new\_payload_i}$:

$$L_{new\_payload_i} = k_i * L_{entity} \tag{2}$$

3.2 Flow Monitoring and Sampling with Low-Latency Based on Optimization Theory

Based on the traffic monitoring service of notification subscription, the traffic is actively automatically reported to the controller, completely avoiding the OpenFlow protocol request message. So it is possible to find out the switches that cover all of the flows in the network during traffic monitoring, and collect the fingerprint on each switch. In this way, the smallest packet sampling payload can be achieved, that is, the sum of the size of the flow table items of all flows.

In the actual network deployment, the controller is deployed in the existing network and relies on the underlying network link. The distance and link bandwidth from each switch to the controller are different, so the transmission delay is also different. Another difficulty in network monitoring is the requirement for real-time network status awareness. We need to be able to perceive changes in the network traffic as quickly as possible, so the latency of monitoring packets transmission in network monitoring must be low enough. Since the sampling is performed on multiple switches, we should set the total latency of packet transmission on multiple switch as the evaluation metric, and establish a model to minimize the total latency required for sampling.

Assuming that the delay vector from the OpenFlow switch to the controller in the network is D, D_i, $i = 1, 2, 3..., n$ represents the network delay from the switch s_i to the controller, and the unit is ms. P_f represents the switches that forward flow f. In order to minimize the total network delay of transmitting traffic packets during sampling, an optimization model is established:

$$\begin{aligned} &\min \quad \sum_{i=1}^{n} D_i * X_i \\ s.t. \quad &\sum_{i:s_i \in P_f} X_i \geq 1, \forall f \in F \\ &X_i \in \{0, 1\}, i = 1, 2, 3, ..., n \end{aligned} \tag{3}$$

There are two possibilities for each X_i, and there are 2^n in all cases sampled. This problem is an NP problem, and the algorithm complexity is very high to solve directly through polling. Here, heuristic methods can be used to approximate the problem, and the problem can be decomposed into two small goals:

1. The total delay required for one sampling is the smallest
2. Sampled switches can cover all network flows

Then we can adopt a greedy strategy, preferentially select low-latency switches for sampling, and then remove the sampled flows from the flow set F, and repeat this process until all flows are sampled, See the Algorithm 1 for the detailed process.

```
Data: F, P_f,D
Result: X = x_1, x_2, ..., x_n
for x_i in X do
 |  x_i ← 0
end
while F ≠ ∅ do
 |  d ← MAX_INT, k ← -1 ;
 |  for d_i in D do
 |  |  if x_i = 1 then
 |  |  |  continue
 |  |  end
 |  |  if d ¿ d_i then
 |  |  |  d ← d_i, k ← i ;
 |  |  end
 |  end
 |  x_k ← 1 ;
 |  for f in F do
 |  |  if x_k ∈ P_f then
 |  |  |  remove f from F ;
 |  |  end
 |  end
end
```

Algorithm 1: Solve the set of sampling switches with the smallest total latency in single sampling

4 Service Flow-Oriented Attack Recognition Model

4.1 Service Flow Features Required by the Model

The basic features of data mining are directly derived from the flow table collection and packet sampling of the network monitoring module. The flow table collection includes source IP address, destination IP address, source port number (UDP/TCP), destination port number (UDP/TCP), IP message protocol type, flow table matching counter, and flow table matching interval. Packet sampling can obtain the length of the IP message, the partial truncation information of the IP message, and the interval between the arrival of adjacent messages. The source IP address indicates the source address of the message or a forged IP address. In some reflection attacks, it may also be the address of an intermediate server. As a result, the distribution of IP messages may potentially reflect the source and the type of attack of DDoS attacks; The destination IP and port

indicate the access address and port provided by the network service. Under normal circumstances, the distribution of packet request a service also has a certain pattern. This feature can reflect the application flow in the network. IP protocol and IP Packet truncation data are used to determine the type of application flow in the network. Basic features are rather limited in representing the pattern of the traffic, but sampling multiple such features continuously can reflect the transaction fragment information of the packets sent by the sender. These transaction fragments allow to infer the sender's intent and determine whether the sender has the intention to attack, so as to identify whether the traffic is normal or abnormal. For example, a sequence of time intervals where traffic matches. When a user accesses a service, multiple packets will be generated on the network. The time interval for the flow matching of these consecutive packets will also reflect the features of the transaction performed by the user to a certain extent. When visiting a portal site, it can be divided into the process of opening a webpage, reading or interacting, and jumping to another webpage to continue reading. There will be a certain dwell time between these actions. These time intervals are also the generation of packets interval.

4.2 DDoS Attack Detection Model Based on Clustering and VAE

Both spectral clustering and variational auto-encoder can be used alone to detect DDoS attacks, but a single model is prone to over-fitting and under-fitting, and cannot achieve good attack recognition results. Therefore, this paper proposes spectral clustering Combine with the variational auto-encoder to build a predictive model. The data used for training is the network flow data X obtained after the network monitoring module data pipeline. First, X is trained through the variational auto-encoder. Both the encoder and decoder of the variational auto-encoder select multi-layer perceptrons (MLPs). After training, we can get the distribution parameters of the potential features of the original data μ and ϵ, Then you can sample a potential feature code X^i_{code} for each sample data X^i, and then concatenate the original data X^i with this hidden feature code, you can get extended data X^i_{ext}, Finally, through spectral clustering, the number of clusters is set to 2, and the new extended data set X_{ext} is trained to obtain a model for detection DDoS attacks. The specific model architecture is shown in Fig. 1.

The main part of model training is the training of the VAE. The VAE only uses normal applications to train, so that the model can learn the potential feature representation of the service flow, and then make the clustering algorithm to find the normal application flow characteristic group. When the model is deployed, the normal application flow data and a small amount of attack application flow data of the automatic application encoder are trained. In order to obtain the cluster numbers of the normal application flow and abnormal attack flow. When new traffic reaches the detection module, the model can be combined with historical traffic data to determine whether the new traffic is DDoS attack traffic, so as to achieve the purpose of attack detection.

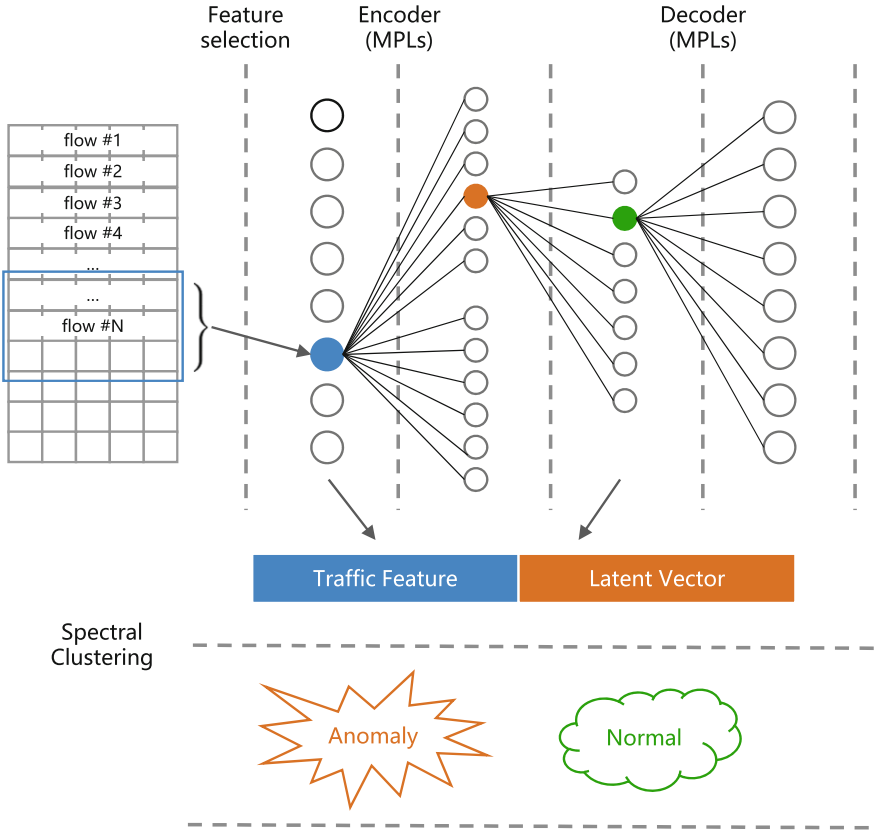

Fig. 1. DDoS attack detection model architecture

4.3 DDoS Attack Defense Based on Recognition Result

When the detection model recognizes a DDoS flow, a series of operations can be used to reduce the damages. First, since we have got the source IP, port and other metadata of the flow, specific flow tables can be sent to block the DDoS traffic on SDN switches in the flow forwarding path. The other way is much gentle, to send specific meter tables to SDN switches, in order to limit traffic flow forwarding rate. However, the IP addresses used by DDoS flow are usually fake, then the real malicious host cannot be located directly through the IP address. But with the global view of SDN, it can be easy to trace the source of malicious traffics. Based on the metadata of traffic flow and the data in switches, malicious traffic can be traced hop by hop to the source. And then, the edge switch can be sent a flow table to block the malicious host, or a meter table to limit the forwarding rate.

5 Simulation and Performance Evaluation

5.1 Simulation Setup

In order to evaluate the approach, we designed a simple SDN-based campus network. The campus network topology is shown in Fig. 2. A Leaf-Spine structure is used to achieve interconnection through the IP network, which is a two-layer network structure. Unlike the traditional three-layer (core layer, convergence layer, access layer), Leaf-Spine is a two-layer network structure. The Leaf layer switches are responsible for accessing all servers and terminal devices. The Spine layer switches act as a backbone switch and is fully connected to the Leaf layer switches. Two edge leaf nodes (Edge Leaf) are configured in the entire network topology, and external networks such as the Internet can be accessed through these two edge nodes.

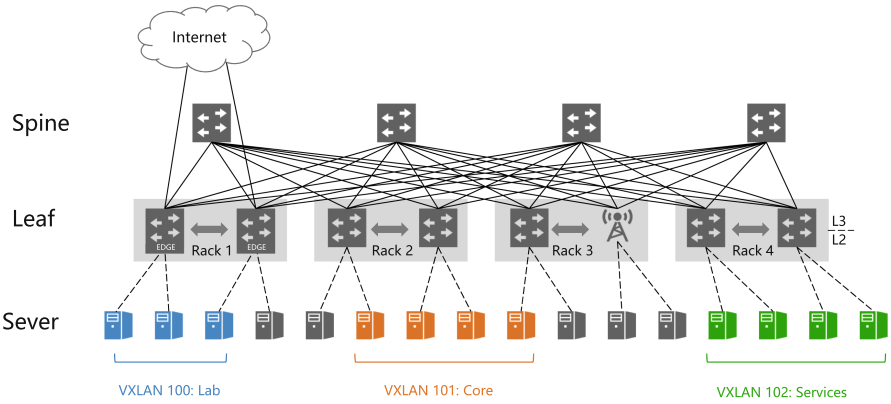

Fig. 2. Campus network topology for simulation

5.2 Network Traffic Sampling Efficiency Evaluation

Three application streams are set up in the test environment, web page information stream, video information stream and background traffic simulated by IPerf tool. Among them, the web page information flow can randomly select three servers on the campus network as the Web server, and access the Web server at a fixed frequency by simulating the client side. Each client uses its own frequency to send request data to server. At the same time, in order to simulate the generation and disappearance of traffic, the client is set to stop for a period of time after every 50 requests to invalidate the flow entry in flow table. The web server has only a fixed-size page, and the client accesses this fixed page every time, so as to facilitate the comparison of the sampling load of different methods. For video stream, we build an Real Time Streaming Protocol (RTSP) video stream server

through VLC, and choose two servers in the network as the video stream server, and choose three clients as the consumer end of the video stream. Unlike web streaming, the video streaming is continuous, and the video streaming protocol is set to the RTSP. In addition to the basic application flow, in order to simulate a more realistic environment, we use IPerf to set three background traffic, the bandwidth of the background traffic is set to 1 Mbps, the detailed application flow deployment is shown in Fig. 3.

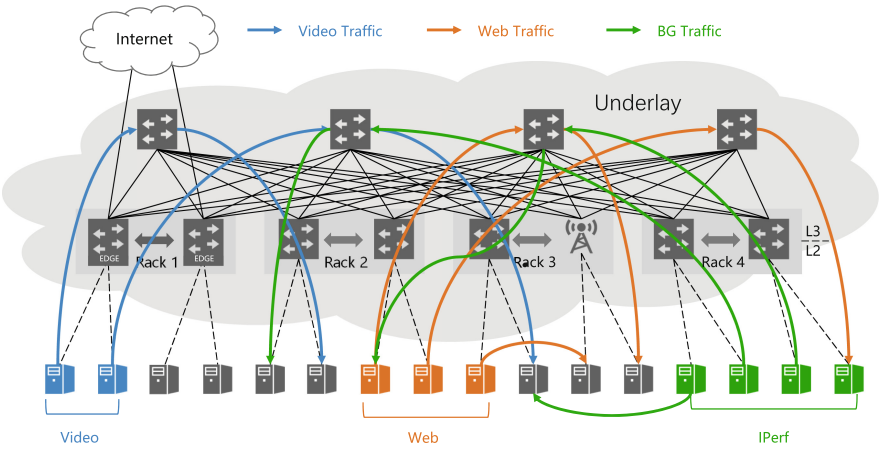

Fig. 3. Application flow deployment in the evaluation of network monitoring module

In order to reflect the payload of network monitoring, the average sampling payload of the flow is introduced into $\overline{f_{payload}}$, which is defined as the average sampling payload of each flow over a period of time. Suppose the payload of each sample in the sampling time is P_i, the flow set of each sample is F_i, and the total number of samples is m, then the average sample payload of the flow is:

$$\overline{f_{payload}} = \frac{\sum {P_i}^m}{\sum |F_i|^m} \tag{4}$$

Regarding the timeliness of network flow sampling, we also defines the metric, total delay D_{simple}, which can better reflect the time cost of acquiring the entire flow status per sample. When the number of flows in the network increases, this metric will increase, so it can reflect how congestion the network behaves. Let the number of packets in a single sampling be N_{simple}, and the end-to-end delay of each sampling is t_i, then D_{simple} is

$$D_{simple} = \sum {t_i}^{N_{simple}} \tag{5}$$

Through the monitoring test in the simulation environment, the network monitoring performance test result is shown in Fig. 4.

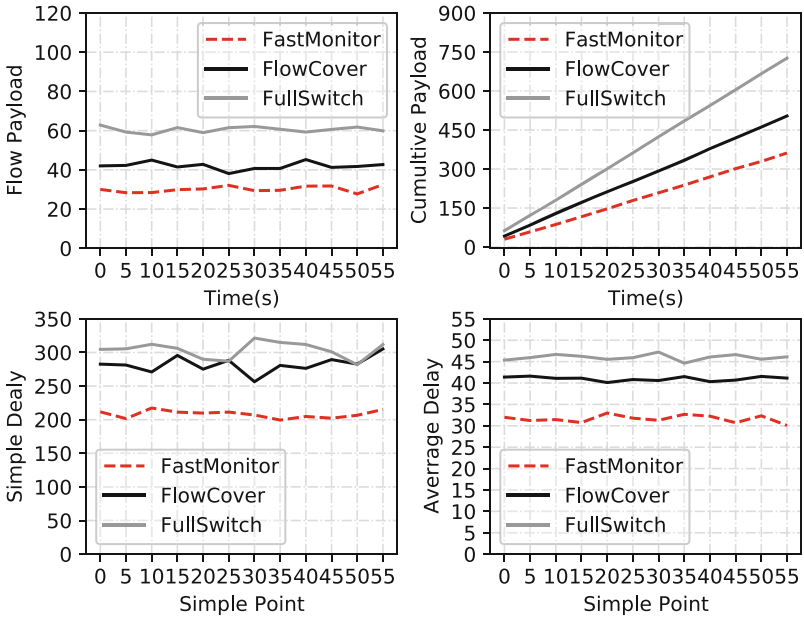

Fig. 4. Network monitoring module performance test results

According to the test results, because of the duplicate sampling problem of FullSwitch, its average message load size is larger than the theoretical load needed per flow. The sampling load of FlowCover algorithm is closer to the theoretical average load for it implements finding the smallest set of switches that can get all flow information, avoiding duplicate sampling. However, the use of polling to obtain information, which causes the request message, will still consume some network resources. The FastMonitor method proposed in this paper adopts a subscription notification mode based on the FlowCover algorithm. In this way, the switch can actively send data to the controller at regular intervals, reducing the bandwidth consumption required for sending request messages and the latency. The average load of FastMonitor method in this test can reduce to only 30 bytes per flow, which decreases the time of each sample by about 60ms compared to FlowCover algorithm. FastMonitor can have a better sense of the changes in the network and thus can react to attacks faster.

5.3 Attack Detection Model Evaluation

To evaluate the model, except the normal flow, we also need to simulate the attack flow. Three common denial-of-service attack flows are selected for simulation, which are the common SYN Flood, the Memcache reflection attack with

the most lethality, and the slow attack Slowloris. The data set is 24-h packet capture data that simulates normal application flow data and attack flow. and the Memcache reflection attack is hidden in the training set, which appears in the test set. The test set contains all the three types of attack. This evaluation uses a mixed DDoS attack, which is the current mainstream DDoS attack method. And the generalization ability of these models can also be well tested.

For the SYN Flood attack and the Slowloris slow attack, both attack packets are sent directly to the target server. Three attack nodes are randomly selected and each malicious node sends packets to the target server at 10 min intervals, 3 min each time (See Fig. 5).

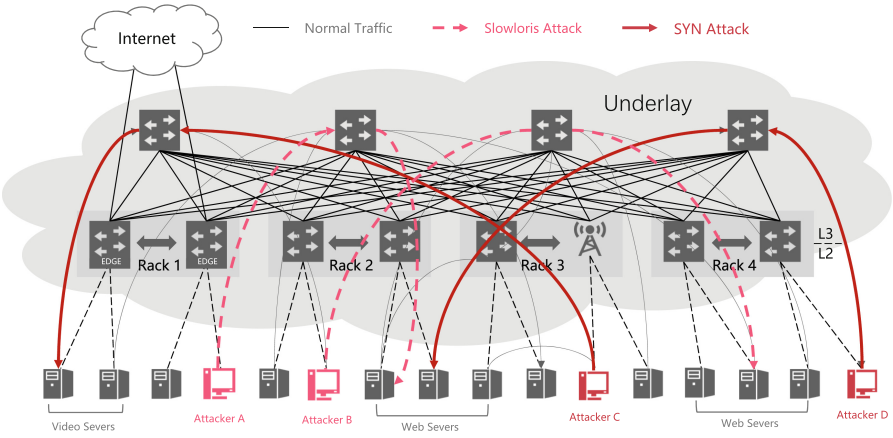

Fig. 5. Deployment of SYN Flood and Slowloris attack simulation

The Memcache reflection attack is different from other types of attacks that send requests directly to the attacked server. This attack forges the source IP of the Memcache request message as the target IP, and then uses the response message of the distributed Memcache servers to achieve an indirect attack. In order to simulate this type of attack, it is necessary to set up two Memcache servers in the network, namely Memcache Sever A and B, and randomly select three attacking nodes to send fake Memcache request messages whose source IP address is the IP address of the attack target. The specific deployment method of Memcache reflection attack is shown in Fig. 6.

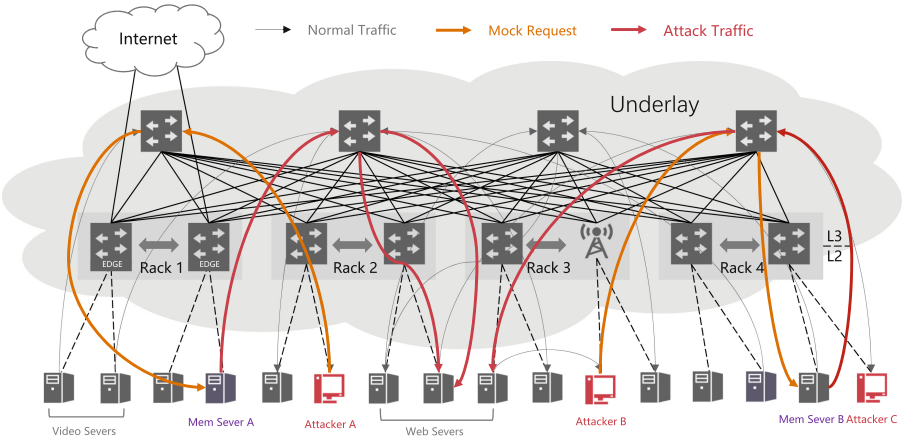

Fig. 6. Deployment of memcache reflection attack simulation

In order to compare the performance of the DDoS attack recognition model proposed in this article, we also test several other common recognition models, such as SVM and deep neural network, and then compare the evaluation scores of different models. Here, multiple evaluation scores are used to reflect the differences of the models. Performance mainly includes commonly used evaluation indicators such as Accuracy, True Postive Rate (TRP), False Postive Rate (FPR) and F1-score (the harmonic mean of precision and recall). For the data used for testing, we collected 24-h data through simulated application flow and attack flow, and use the 24-h data as the test data set to evaluate the model recognition ability. The test results are shown in Table 1.

According to the test results, the SC-VAE attack detection model proposed in this paper has certain advantages over other common models in terms of the accuracy of detection and the ability to correctly determine the anomaly class. Since the SC-VAE distinguishes unknown DDoS attack streams by detecting real application stream features, it has certain ability to detect unknown DDoS attack types. The training set of this test does not contain Memcache reflection attack types, but the test results still maintain a high accuracy. In contrast, common classification-based learning methods such as SVM and DNN can achieve

Table 1. Performance comparison of different models

Model	ACC	FPR	TPR	PPV	F1
SC-VAE	0.9725	0.0373	0.9826	0.9634	0.9728
SVM	0.9668	0.0480	0.9818	0.9533	0.9672
DNN	0.9541	0.0531	0.9615	0.9465	0.9536
VAE	0.9533	0.0712	0.9761	0.9522	0.9583
Spectral clustering	0.9122	0.1035	0.9547	0.8870	0.9311

prediction results only when there are the same distributions of training and test sets. They require collecting a large amount of sample data, thus reducing the generalization ability of these algorithms.

6 Conclusion

Accurate DDoS identification is the basis for effective DDoS defense. Rapid and timely flow sampling is no doubt the basis of accurate DDoS identification.

In this paper, we propose an intelligent SDN DDoS detection framework. In detail, first, an network monitoring and sampling model is proposed. This model helps to find the most suitable switch set with smallest sampling latency, and then help to enhance the network monitoring and sampling efficiency. Based on OpenFlow extension design, this model realizes a subscription service for real-time sampling data collection. Meanwhile, the framework provides a DDoS detection model SC-VAE, consisting of algorithms of spectral clustering and variational auto-encoder. This detection model cluster flows by flow features, then abnormal hybrid DDoS and potential unknown DDoS attack can be detected. Simulation results show that the accuracy of SC-VAE is 97.3%, and the FPR is only 3.7%, which can ensure a good prediction effect, and is better than traditional classification models based on SVM.

Next, we will continue to optimize this intelligent SDN DDoS detection framework, fill in it with defence strategy and implementation.

References

1. Ale, L., Zhang, N., Wu, H., Chen, D., Han, T.: Online proactive caching in mobile edge computing using bidirectional deep recurrent neural network. IEEE Internet Things J. **6**(3), 5520–5530 (2019)
2. Cheng, T.Y., Jia, X.: Compressive traffic monitoring in hybrid SDN. IEEE J. Sel. Areas Commun. **36**(12), 2731–2743 (2018)
3. Conti, M., Lal, C., Mohammadi, R., Rawat, U.: Lightweight solutions to counter DDoS attacks in software defined networking. Wireless Netw. **25**(5), 2751–2768 (2019)
4. Gangadhar, S., Sterbenz, J.P.: Machine learning aided traffic tolerance to improve resilience for software defined networks. In: 2017 9th International Workshop on Resilient Networks Design and Modeling (RNDM), pp. 1–7. IEEE (2017)
5. Arevalo Herrera, J., Camargo, J.E.: A survey on machine learning applications for software defined network security. In: Zhou, J., et al. (eds.) ACNS 2019. LNCS, vol. 11605, pp. 70–93. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-29729-9_4
6. Kalkan, K., Altay, L., Gür, G., Alagöz, F.: Jess: Joint entropy-based DDoS defense scheme in SDN. IEEE J. Sel. Areas Commun. **36**(10), 2358–2372 (2018)
7. Liu, Z., He, Y., Wang, W., Zhang, B.: DDoS attack detection scheme based on entropy and PSO-BP neural network in SDN. China Commun. **16**(7), 144–155 (2019)
8. McKeown, N., et al.: OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comput. Commun. Rev. **38**(2), 69–74 (2008)

9. Shirali-Shahreza, S., Ganjali, Y.: Flexam: flexible sampling extension for monitoring and security applications in openflow. In: Proceedings of the Second ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, pp. 167–168 (2013)
10. Su, Z., Wang, T., Xia, Y., Hamdi, M.: Flowcover: low-cost flow monitoring scheme in software defined networks. In: 2014 IEEE Global Communications Conference, pp. 1956–1961. IEEE (2014)
11. Tang, T.A., Mhamdi, L., McLernon, D., Zaidi, S.A.R., Ghogho, M.: Deep learning approach for network intrusion detection in software defined networking. In: 2016 International Conference on Wireless Networks and Mobile Communications (WINCOM), pp. 258–263. IEEE (2016)
12. Zhang, N., et al.: Software defined networking enabled wireless network virtualization: challenges and solutions. IEEE Netw. **31**(5), 42–49 (2017)