Hindawi
Security and Communication Networks
Volume 2021, Article ID 9409473, 10 pages
https://doi.org/10.1155/2021/9409473

Research Article

RT-SAD: Real-Time Sketch-Based Adaptive DDoS Detection for ISP Network

Haibin Shi,[1,2] Guang Cheng,[1,2] Ying Hu,[1,2] Fuzhou Wang,[1,2] and Haoxuan Ding[1,2]

[1]*School of Cyber Science and Engineering, Southeast University, Nanjing 211189, China*
[2]*Key Laboratory of Computer Network and Information Integration, Southeast University, Ministry of Education, Nanjing 21189, China*

Correspondence should be addressed to Guang Cheng; gcheng@njnet.edu.cn

Received 16 April 2021; Revised 20 June 2021; Accepted 14 July 2021; Published 28 July 2021

Academic Editor: Weiwei Liu

Copyright © 2021 Haibin Shi et al. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

With the great changes in network scale and network topology, the difficulty of DDoS attack detection increases significantly. Most of the methods proposed in the past rarely considered the real-time, adaptive ability, and other practical issues in the real-world network attack detection environment. In this paper, we proposed a real-time adaptive DDoS attack detection method RT-SAD, based on the response to the external network when attacked. We designed a feature extraction method based on sketch and an adaptive updating algorithm, which makes the method suitable for the high-speed network environment. Experiment results show that our method can detect DDoS attacks using sampled Netflowunder high-speed network environment, with good real-time performance, low resource consumption, and high detection accuracy.

1. Introduction

Distributed denial of service (DDoS) attack has been one of the most difficult attacks in the network. DDoS attacks can interrupt the network service temporarily or even make the system break down. DDoS attacks are usually launched by botnet devices. In recent years, the number of IoT devices is increasing rapidly, which are more vulnerable [1] than traditional network equipment. The IoT botnet expands the scales of DDoS attacks significantly. In 2016, DNS service provider Dyn was attacked by massive IoT devices controlled by Mirai Botnet, which directly led to a large area of services unavailable on the east coast of the United States. Another difficulty in defense against DDoS attacks is the rise of reflection amplification attacks. In 2018, GitHub was attacked by a reflection amplification DDoS attack by leveraging the Memcached protocol vulnerability, with the reflection multiple as high as 50,000 times and the peak traffic of 1.35 Tbps.

According to Akamai's annual summary [2] of DDoS attacks in 2020, the number of large-scale DDoS attacks has increased significantly. In the largest DDoS attack event [3], the attack traffic has reached 1.44 Tbps, and the attack is very complex. It is necessary to combine multiple mitigation methods as soon as possible to block the attack. However, for large-scale DDoS attacks, it is difficult to deploy attack detection and defense devices near the victims for effective defense. A more effective way is to collect traffic and detect DDoS attacks on the backbone network.

In the past decades, researchers have proposed many detection methods for DDoS attacks. Most of the existing methods are based on machine learning or deep learning. These methods need to train the model on a large number of labeled network traffic data in advance to ensure the accuracy of attack detection. However, there are some problems in these methods:

(a) At present, new attack vectors are constantly being mined. For example, at the end of July 2020, the FBI issued an alert [4] that CoAP, WS-DDARMS, and other protocols may be used to launch DDoS attacks. DDoS attacks based on new attack vectors may have great changes in the statistical characteristics such as packet speed and packet spacing used in traditional

methods, which makes traditional methods less adaptable to different attacks.

(b) Most of the models need to be trained in advance before they are used for detection. If the network environment changes, the current network traffic may not follow the data distribution of the pretrained model. At this time, the traditional methods need to retrain the model to maintain high accuracy. However, in the scene of attack detection in the backbone network of operators, it is very difficult to obtain labeled data and retrain the model frequently. In addition, it is also difficult to determine the right time to update the detection model.

(c) For DDoS attack detection method design and performance evaluation, most of the methods only consider the detection accuracy, false alarm rate, and false alarm rate but do not consider the real-time performance and resource consumption of the method. Although their methods can work in small DDoS attacks simulated by tools such as hping3 [5] and LOIC [6], they did not consider the performance of such methods in the real-world high-speed network environment, like the ISP network.

In order to adapt to various types of DDoS attacks in the high-speed network environment, we propose an real-time adaptive DDoS detection method based on sketch for ISP network. The method implements dynamic adjustments of parameters of the detection model according to the current network situation, and realizes the real-time adaptive DDoS detection in a high-speed network. Compared with the previous DDoS attack detection method, the main contributions in this paper are as follows:

(1) We proposed an adaptive DDoS attack detection algorithm, which can update the model adaptively according to the network situation without manually setting the detection threshold parameters in advance.

(2) We collected high-speed network traffic from the real-world backbone network boundary. In addition, we sampled the network traffic at different rates to make it closer to the real-world network detection environment.

(3) We evaluated our detection method in comprehensive aspects, including the resource consumption, the real-time performance which rarely appeared in previous work.

The rest of this paper is arranged as follows: Section 2 describes the related work. Section 3 introduces the attack detection method. Section 4 is the experiment and verification. Section 5 is the summary and prospect.

2. Related Work

DDoS attack detection is different from the deployment of detection points, which can be divided into source detection, intermediate network detection, and victim detection. Some of the work is summarized as follows:

(1) For the scene of DDoS attack detection at the attack source, Mergendahl et al. [7] proposed an improved FR-WARD method based on D-WARD for IoT environment, which can accurately detect and defend DDoS attacks and reduce the retransmission overhead of benign IoT devices. Tang et al. [8] proposed a framework FDDA for fast detection and defense of DDoS attacks in the web application environment. They used the DBSCAN method to establish the blacklist in the scanning stage, which makes attack mitigation faster. Biswas et al. [9] proposed a DDoS attack detection method based on behavior similarity between virtual machines for DDoS attacks in the data center.

(2) For the scene of DDoS attack detection at the victim end, Rahmani et al. [10] proposed a statistical method based on network anomaly and joint entropy of multiservice distribution, which judges the occurrence of attacks by measuring the statistical correlation between the time series of the number of IP flows and the total traffic size. Compared with some methods only using the traffic size, the method has fewer false positives. Mallikarjunan et al. [11] used PCA to reduce the dimension of features and tested the accuracy of machine learning algorithms such as naive Bayes, j48, and random forest on the data set created by the author. The results show that the performance of naive Bayes is better. Aamir et al. [12] used a semisupervised machine learning method to cluster the data using traffic rate, processing delay, and CPU utilization information collected by the victim.

(3) For the scene of DDoS attack detection at the intermediate network, Barati et al. [13] proposed a DDoS attack detection algorithm based on hybrid machine learning. The method uses a genetic algorithm to select features and the multilayer perception (MLP) in ANN to detect attacks. The accuracy of the algorithm is higher than that of the simple machine learning algorithm. Yusof et al. [14] proposed a method of attack detection of PTA-SVM, by combiningSVM with data packet threshold algorithm (PTA). Compared with the improved k-means and logistic regression technology, the PTA-SVM method has a smaller false alarm rate and higher accuracy.

Attack detection in ISP level large-scale network environments is a typical example of intermediate network detection. Compared with the other two attack scenes, more network traffic data can be obtained in intermediate network detection, which makes the detection more accurate and flexible. However, at the same time, the network traffic collected in the intermediate network is larger and the network flow rate is faster, which puts forward higher requirements for the feature storage and calculation.

Many researchers focus on sampling technology for the measurement and statistics of the high-speed network. Ujjan et al. [15] used deep learning, with sFlow sampling and adaptive polling sampling, to detect DDoS attacks. Biswas et al. [9] proposed a flow grouping method based on the behavior similarity between virtual machines and combined with the optimization solver to specify a better sampling rate. The main work of our paper is to focus on the use of light features and based on sketch to achieve high-speed network traffic processing.

In the implementation of DDoS attack detection methods, most of the methods are based on machine learning. Zekri et al. [16] proposed an attack detection algorithm based on decision tree in the cloud environment. Both Hou et al. [17] and Filho et al. [18] used the random forest method to identify attacks. The method proposed by Idhammad et al. [19] combines entropy estimation with Extra-Trees to detect DDoS attacks.

In addition, some researchers have compared different machine learning methods. For example, Priya et al. [20] used three classification algorithms KNN, Random Forest, and Naive Bayesian to detect DDoS Attacks based on the features of incremental time and packet size. Saini et al. [21] used random forest algorithm, Naive Bayes algorithm, and j48 algorithm to detect attacks, and the j48 algorithm produced the best results.

There are also some works based on deep learning. The method proposed by Doshi et al. [22] uses a combination of deep learning and support vector machine to detect attacks. Yuan et al. [23] proposed a DDoS attack detection method based on the recurrent neural network (RNN).

Since most of these methods are supervised or semi-supervised, it is time-consuming to training the classifier on a large amount of network traffic data. Therefore, real-time detection is not guaranteed if the algorithm is deployed in a high-speed network.

Given above, we propose a real-time sketch-based adaptive DDoS detection method. We address more practical issues in real-world detection, such as real-time performance and adaptive ability in the high-speed network environment.

3. Real-Time Sketch-Based Adaptive DDoS Detection

In this paper, we designed an adaptive DDoS attack detection method named RT-SAD, which is based on the asymmetry of network traffic when DDoS attacks occur.

This section is divided into four parts. Firstly, we will describe the overall framework of the detection method. Secondly, we will explain the principle of attack detection. Finally, we will introduce the realization of two core functions: feature statistics and model updating.

3.1. Overview. The overall architecture of this DDoS detection system is shown in Figure 1. The system is mainly composed of the feature statistics module, the attack detection module, and the model updating module, which is implemented based on sketches.

In the detection process, multiple flow records in fixed time intervals will form a time window. When each flow record in the time window arrives, it will go through the feature statistics module first. Two sketch tables in the module work together to realize the statistics and update asymmetric flow features. After the feature statistics, the attack detection module will use three sketch tables to detect the attack. The three tables used in the detection module are dynamically updated. After the detection module detects all the flow records in a time window, the model updating module will start to work. The module updates the predictive value and threshold of the current window by learning the features of the history window. The predictive value and threshold used in the next time window for attack detection are the updated predictive value and threshold.

The meanings and functions of the five sketch tables in the detection system are shown in Table 1.

In the next part of the article, we will introduce the principle of attack detection in detail.

3.2. Attack Detection. In the network communication model of client-server, there should be both requests and responses. When the server suffers a DDoS attack, the request traffic sent by the botnet will be much larger than the response traffic returned by the server. Because the attacker wants to exhaust the resources of the server as much as possible, the network traffic between clients and the server will show the phenomenon of asymmetry.

In order to quantify the asymmetry of network flow, we propose a quantitative method of asymmetry. We use a pair of IP addresses to represent the flow record. As shown in Figure 2, there are bidirection data transfers between IP-A and IP-C, so it is considered that the request from A to C is normal. As for IP-B and IP-C, there is only traffic from B to C and no traffic from C to B, it is considered that traffic between IP-B and IP-C is asymmetric. And in the current time window, the asymmetric flow feature of IP-C will be increased by 1.

After the analysis of real network traffic, we found that when a DDoS attack occurs, the victim server usually cannot respond to all the clients. There will be a large number of one-way traffic whose destination address is the victim host. That is, when a DDoS attack occurs, the value of asymmetric feature corresponding to some IP addresses will be significantly higher than the normal situation, as shown in Figure 3. In this paper, we mainly use asymmetric of traffic to detect DDoS attacks.

The complete attack detection process is shown in Figure 4. There are two important parts during the detection process. One is the feature statistics and attack detection when each flow record arrives, and the other is the model updating process, including predicted value update and threshold update at the end of the current time window.

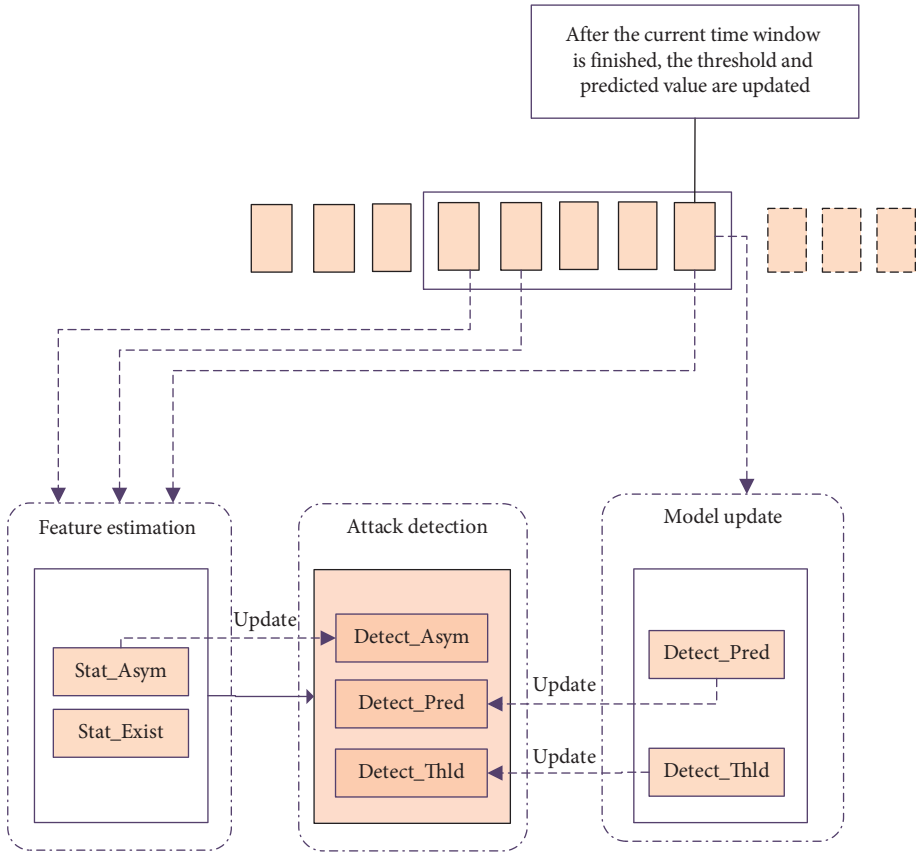

Figure 1: Overall architecture.

Table 1: Function of sketch.

Sketch name	Meaning	Function
Stat_Asym	The asymmetric flow features of destination IP	To complete the statistical function of feature in a single time window
Stat_Exist	The existence of SIP and DIP	
Detect_Asym	The feature value in the attack detection window	To provide the judgment conditions for the attack detection module to work
Detect_Pred	The predicted value in the attack detection window	
Detect_Thld	The threshold value in the attack detection window	

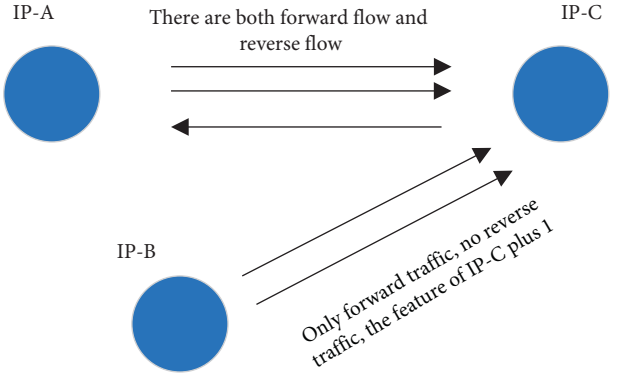

Figure 2: Definition of asymmetric flow.

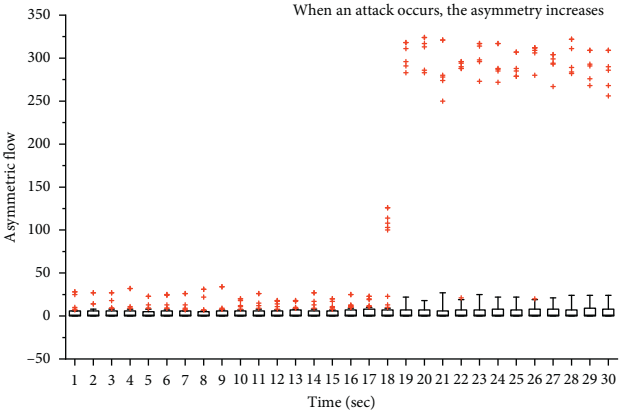

Figure 3: Asymmetric feature when the attack occurs.

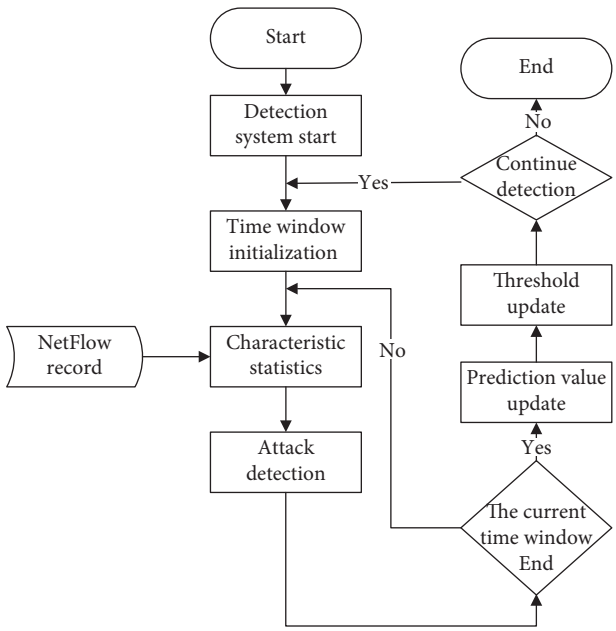

Figure 4: DDoS attack detection process.

The two important parts are as follows:

(1) When a flow record (SIP and DIP) arrives at the detection system, the system will first update the feature corresponding to the DIP in the current window. And then the system will use the feature of DIP, the predicted value, and the threshold calculated according to the feature in the history window, to identify whether the destination IP address in the current flow record is suffering from DDoS attacks. The system will give an attack warning of the victim IP if the detection result is true.

(2) At the end of the current time window, the system will update the predicted value and threshold of the feature corresponding to the flow records, and the update is only for the normal IP without attack warning, while the feature of attacked IP will not be updated until they return to normal.

The specific detection method is shown as Algorithm 1.

The next part of this paper will describe the algorithm and implementation of feature statistics and model updating.

3.3. Sketch-Based Estimation of Asymmetric Flows. As shown in Figure 1, we use two Sketch tables, Stat_Asym and Stat_Exist to record and update the features of asymmetric flows corresponding to IP in the current time window. More specifically, Stat_Asym is used to record the actual asymmetric flow value of each IP, and stat_ Exist is used to record the existence of IP pairs (SIP and DIP).

Figure 5 shows the statistics and update rules of the features. When the flow record arrives, the system will update the Stat_Asym according to the existence of IP pairs recorded in Stat_Exist.

More specifically, for arriving flow record (SIP and DIP), the system finds the values of Stat_Exist[SIP|DIP] and Stat_Exist[DIP|SIP], respectively, which represents the existence of two tuples (SIP and DIP) and (DIP and SIP), and updates the current asymmetric feature according to the different existence conditions of these IP pairs. The value of Stat_Exist represents the existence of IP pairs. If Stat_Exist [SIP|DIP] is 0, it means that the traffic corresponding to the tuple (SIP and DIP) has not appeared in this time window. If the value is greater than 0, it means that the traffic corresponding to the tuple (SIP and DIP) has appeared in this time window.

There are four combinations of Stat_Exist[SIP|DIP] and Stat_Exist[DIP|SIP]. In the attack detection process, we mainly focus on whether the destination IP is attacked; that is, we mainly consider the asymmetric feature of DIP, so only in some cases, the system needs to update the Stat_Exist. The specific update algorithm is shown in Algorithm 2.

After the above steps, the statistics and updates of features are completed. Sketch Stat_Asym[DIP] represents the feature value corresponding to DIP in the current time window.

3.4. Model Updating. At the end of the current time window, the detection system will update the predicted value, threshold, and model parameters. In the system implementation, the sketch table Detect_Thld is responsible for the storage of threshold, and the calculation of threshold is related to the sequence of historical residuals $(\text{res}_1, \text{res}_2, \ldots, \text{res}_n)$. The residuals of an IP in the time window m, res_m, means the difference between the feature value and the predicted feature value.

The threshold corresponding to an IP in the current window is calculated by three-sigma rule, as shown in the following equation:

$$\text{threshold} = \text{mean}(\text{residual}) + 3 * \text{std\_dev}(\text{residual}), \tag{1}$$

where residual refers to the sequence of historical residuals $(\text{res}_1, \text{res}_2, \ldots, \text{res}_n)$. The mean (residual) is the average value of the historical residual sequence. The std_dev (residual) is the standard deviation of the historical residual sequence.

For the storage and update of dynamic threshold, if the historical values of residuals corresponding to all IP in the past n windows are completely recorded and then the mean and variance of residuals are calculated, too much storage space will be consumed. Therefore, in order to save resources as much as possible, the residual values in all the latest n historical time windows are not directly recorded, but the threshold values are updated by rolling update. And the variance and mean values in multiple historical windows are replaced by progressive variance and mean values. In this paper, the online mean and variance algorithm proposed by Welford [24] is used. The specific formula is shown as

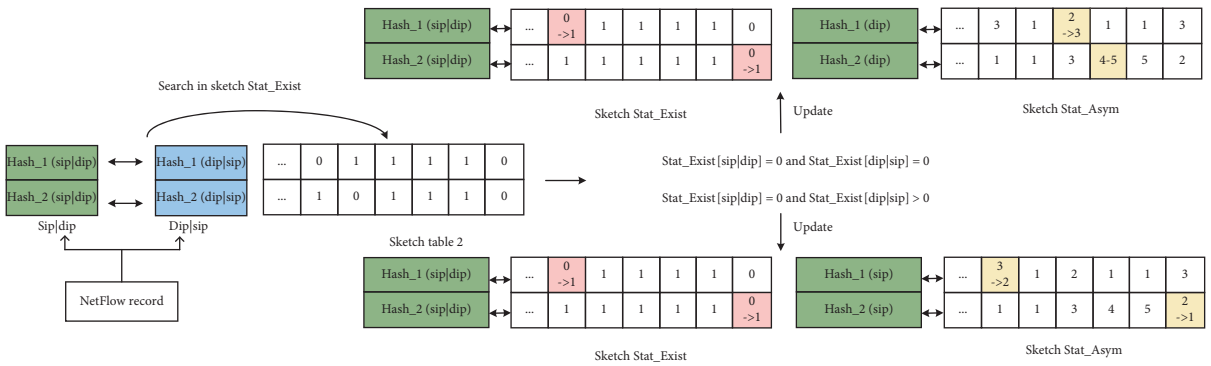

Figure 5: Estimation of asymmetric flows.

$$\text{mean}_n(X) = \text{mean}_{n-1}(X) + \frac{x - \text{mean}_{n-1}(X)}{n}, \tag{2}$$

where X is a random variable, x is the nth number of X, and $\text{mean}_n(X)$ represents the mean of the first n numbers in X.

The progressive variance calculation method is shown as

$$\begin{aligned} \text{Var}\, D_n(X) = {} & \text{Var}\, D_{n-1}(X) \\ & + (x - \text{mean}_{n-1}(X))(x - \text{mean}_n(X)), \end{aligned} \tag{3}$$

$$\text{Var}_n(X) = \frac{\text{Var}D_n(X)}{n}, \tag{4}$$

where X is a random variable, x is the nth number of X, $\text{mean}_n(X)$ represents the mean of the first n numbers in X, and $\text{Var}_n(X)$ represents the variance of the first n numbers in X.

In order to improve the calculation accuracy, only the intermediate value $\text{Var}D_n(X)$ is recorded, which will be used to calculate the variance value. Therefore, we only need to record the progressive mean value, the progressive variance median value, and the number of cycles to update the mean value and variance.

In the process of threshold calculation, the predicted value used in the calculation also needs to be updated adaptively. The system uses the table Detect_Pred to record and update the predicted value. The update rule adopts a simple and efficient single exponential smoothing method, as shown in

$$\begin{aligned} \text{predAsymVal}_{\text{new}}[\text{IP}] = {} & (1 - \alpha) * \text{predAsymVal}_{\text{old}}[\text{IP}] \\ & + \alpha * \text{currAsymVal} \end{aligned} \tag{5}$$

where $\text{predAsymVal}_{\text{old}}[\text{IP}]$ is the predicted value of the old asymmetric flow number features of the current IP, and the new one is $\text{pred Asym Val}_{\text{New}}[\text{IP}]$; currAsymVal is the number of asymmetric flows corresponding to the IP in the current window.

The value of parameter α in the single exponential smoothing formula is usually set to a specific value between 0.3 and 0.7, but this setting method does not take into account the changes of the current traffic and detection situation. The method in the paper updates α by learning the historical traffic by adopting a specific strategy. The specific parameter value update algorithm is shown in Algorithm 3.

The above strategy, used to update the α parameter, can make the system recover as soon as possible after the occurrence of false positives, to reduce the possibility of continuous false positives caused by one false positive.

In the current system, we set $\alpha_{\min} = 0.3$, $\alpha_{\max} = 0.7$, and $\Delta\alpha = 0.1$.

4. Experiment and Evaluation

4.1. Dataset. The experiment dataset is a mixture of the background network traffic collected from the real-world backbone network and the attack traffic generated by stress testing tool:

(1) In the mixed traffic, the background traffic data are collected from the CERNET backbone network for 60 minutes. In addition, we only intercepted the first 64 bytes of each packet. The intercepted data are about 83 GB, the total amount of original data is about 1373 GB, and the actual flow rate is about 3 Gbps.

(2) The attack traffic data in the mixed traffic are composed of 13 kinds of network layer and transport layer DDoS attack traffic, including UDP Flood, UDP Fragmentation Flood, ICMP Flood, ICMP Fragmentation Flood, TCP SYN Flood, TCP-SYN ACK Flood, TCP ACK Flood, TCP ACK Fragmentation Flood, TCP PUSH ACK Flood, TCP RST Flood, TCP FIN Flood, TCP URG Flood, and X_MAS Flood. The attack traffic is generated by stress testing tool and mixed with normal background traffic at different time points. The scale of each attack traffic is about 1 Gbps, and the IP number of the attack target is 5.

The network flow speed under different sampling rates in this experiment is shown in Table 2.

4.2. Evaluation Criteria. Our solution uses the sliding window method in the detection process, so we use the time window as the unit to evaluate our experiment result.

```
Input: NetFlow Record, Sketch Detect_Asym, Detect_Pred, Detect_Thld
Output: DDoS attack detection results
(1) while Current window not end do
(2) Get a NetFlow record (SIP, DIP)
(3) Update the Detect_Asym [DIP] by using (SIP, DIP)
(4) Residual = abs (Detect_Pred [DIP] – Detect_Asym [DIP])
(5) if Residual > Detect_Thld[DIP] then
(6)         Alert of DDoS
(7) else
(8)         Put the DIP into Update_Set
(9) end if
(10) end while
(11) for each DIP in Update_Set do
(12) Update Detect_Pred [DIP]
(13) Update Detect_Thld[DIP]
(14) end for
```

Algorithm 1: DDoS attack detection algorithm.

In the whole attack detection process, there are four kinds of detection results corresponding to the actual data for the current window, as shown in Table 3.

In each subsequent experiment, we use three indicators to evaluate the effectiveness of the method, namely, accuracy rate (AR), false positive rate (FPR), and false negative rate (MR).

4.3. Resource Evaluation. In this paper, the sampling technique and probability data structure in high-speed network measurement are used to optimize the cost of storage and computing resources. In order to evaluate the resource cost of the proposed algorithm, we conducted two experiments, sampling rate experiment and sketch size experiment. The sketch experiment is to evaluate the detection performance of the algorithm when using different sizes of sketch data structures. The sampling rate experiment evaluates the detection performance of the algorithm for network traffic data with different sampling rates.

The size of the sketch will have an impact on the accuracy of the detection algorithm. Therefore, our sketch resource consumption experiment mainly compares the detection performance of the algorithm by setting a fixed sampling rate and selecting different sizes of the sketch. The sampling rate of the experiment data is 10 : 1, and the size of the sliding window is 1 second. Under this configuration, the number of flows per second is about 9000 without attack and 17,000 under attack. Therefore, the sketch size is set to the following five groups, ranging from 2^{12} (4K) to 2^{15} (32K). The experiment results are shown in Table 4.

Under this condition, when the sketch size is 2^{13} or higher, the algorithm can achieve better results. At the same time, if we want to get better performance, we need to consume more storage resources. In practice, we need to select an appropriate sketch size according to current network flow speed and current hardware performance.

In the case of a high-speed network, the performance of the detection algorithm not only depends on the complexity of its own algorithm but also has a great relationship with the current flow speed. In order to make the network flow speed match the processing flow speed of the algorithm as much as possible, we conducted four groups of experiments using the mixed network flow with different sampling rates: 10 : 1, 20 : 1, 100 : 1, and 200 : 1.

The size of the sketch is 2^{15}, and the size of the detection window is 1 second. Table 5 shows the performance results of the detection algorithm under different sampling rates.

It can be seen from the experiment results that when the algorithm configuration is appropriate, it has good detection performance for different sampling rates of network traffic.

4.4. Real-Time DDoS Detection. In order to evaluate the real-time performance of the algorithm for DDoS attack detection, we design a real-time evaluation experiment. We take the time from attack occurrence to detection algorithm alarm as the experiment measurement criteria. eWe tested the detection time of the current algorithm for different sampling rates, and the experiment results are shown in Table 6.

From the experiment results, it can be seen that in the current experiment, the algorithm has good real-time performance for the different sampling rates of network traffic. In addition, as the sampling rate increases, the number of flows per unit time decreases, so the processing efficiency of the algorithm increases and the detection time decreases.

4.5. Results on Different DDoS Attack Detection. In order to evaluate the applicability of our algorithm for different DDoS attacks, we designed an attack detection applicability experiment. In this experiment, we generate attack traffic for each attack and then mix it into the background traffic separately to detect the performance. The results of the performance for different attacks are shown in Table 7, with the sketch size of 2^{15} and sampling rate of 10 : 1.

From the experiment results, it can be seen that the algorithm has higher detection accuracy for different types of the network layer and transport layer DDoS attacks with lower false alarm rate and missing alarm rate.

```
Input: Sketch Stat_Exist, Stat_Asym
Output: Sketch Stat_Exist, Stat_Asym
(1)  if Stat_Exist[SIP|DIP] = 0 then
(2)    Stat_Exist[SIP|DIP] = Stat_Exist[SIP|DIP] + 1
(3)    if Stat_Exist[DIP|SIP] = 0 then
(4)      Stat_Asym[SIP] = Stat_Asym[SIP] + 1
(5)    end if
(6)    if Stat_Exist[DIP|SIP] > 0 then
(7)      Stat_Asym[SIP] = Stat_Asym[SIP] − 1
(8)    end if
(9)  end if
```

Algorithm 2: Feature updating algorithm.

```
Input: Current residual res, Historical residual sequence res_list, Δα, α, α_min, α_max
Output: α
(1)  if res > mean(res_list) + 3 ∗ std_dev(res_list) then
(2)    if α < α_max then
(3)      α = α + Δα
(4)    end if
(5)  else if res < mean(res_list) + std_dev(res_list) then
(6)  if α > α_min then
(7)  α = α − Δα
(8)  end if
(9)  end if
```

Algorithm 3: Parameter value α update algorithm.

Table 2: Flow speed with the different sampling rates.

Sampling rate	Flow speed/flows per second	
	Without attack	Under attack
10 : 1	9476	17,418
20 : 1	5403	9382
100 : 1	1351	2151
200 : 1	727	1132

Table 3: Confusion matrix.

		Algorithm detection results in the current window	
		Attack detected	No attack detected
Actual situation in current window	Under attack	Correct (TP)	Failing to report (FN)
	Without attack	Wrong report (FP)	Correct (TN)

Table 4: Performance of different sketch sizes.

Sketch size	AR (%)	FPR (%)	MR
2^{12}	89.87	14.03	0
2^{13}	96.89	4.34	0
2^{14}	99.82	0.24	0
2^{15}	99.77	0.32	0

Table 5: Performance of different sampling rates.

Sampling rate	AR (%)	FPR (%)	MR (%)
10 : 1	99.77	0.32	0
20 : 1	99.77	0.12	0.48
100 : 1	99.11	0	2.98
200 : 1	99.88	0	0.39

Table 6: Real-time performance of the algorithm.

Sampling rate	Detection time/millisecond
10 : 1	1064
20 : 1	1029
100 : 1	232
200 : 1	122

Table 7: Performance of different DDoS attacks.

The type of DDoS attack	AR (%)	FPR (%)	MR (%)
SYN Flood	99.48	0.59	0.29
SYN-ACK Flood	99.60	0.32	0.58
ACK Flood	99.40	0.28	1.37
ACK&PUSH Flood	99.74	0.36	0
RST Flood	99.57	0.35	0.59
FIN Flood	99.68	0.20	0.56
TCP URG Flood	99.71	0.80	0.77
X_MAS Flood	99.74	0.20	0.37
ACK Fragmentation Flood	99.57	0.28	0.77
UDP Fragmentation Flood	99.48	0.32	0.96
ICMP Fragmentation Flood	99.63	0.08	1.04
ICMP Flood	99.60	0.44	0.29
UDP Flood	99.45	0.28	1.15

5. Conclusions

Given the current threat of DDoS attacks, we propose a real-time DDoS attack detection method based on sketch for intermediate networks. In this paper, the sketch is used to record and update the features which are needed for attack detection, and the adaptive threshold of the feature is dynamically updated by the historical network traffic. The experiment results show that the method has good performance in accuracy, resource consumption, and real-time performance. At the same time, there are still some improvements in this method, such as adaptive network traffic sampling and adaptive size adjustment of sketch structure changed with the network situation. This is also the content of our following work.

Data Availability

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

Conflicts of Interest

The authors declare that they have no conflicts of interest.

Acknowledgments

This work was supported by the joint fund of the Ministry of Education of China and China Mobile (MCM20180506).

References

[1] K. Abbas, L. A. Tawalbeh, A. Rafiq, A. Muthanna, I. A. Elgendy, and A. A. Abd EI-Latif, "Convergence of blockchain and IoT for secure transportation systems in smart cities," *Security and Communication Networks*, vol. 2021, Article ID 5597679, 13 pages, 2021.

[2] T. Emmons, "Part I: retrospective 2020: DDoS was back-bigger and badder than ever before," 2021, https://blogs.akamai.com/2021/01/part-i-retrospective-2020-ddos-was-back-bigger-and-badder-than-ever-before.html.

[3] L. Jakober, "Akamai mitigates sophisticated 1.44 Tbps and 385 Mpps DDoS attack," 2020, https://blogs.akamai.com/2020/06/akamai-mitigates-sophisticated-144-tbps-and-385-mpps-ddos-attack.html.

[4] C. Cimpanu, "FBI warns of new DDoS attack vectors: CoAP, WS-DD, ARMS, and Jenkins," 2020, https://www.zdnet.com/article/fbi-warns-of-new-ddos-attack-vectors-coap-ws-dd-arms-and-jenkins/.

[5] Kalitool, "Hping3 package description," 2019, https://tools.kali.org/%20information-gathering/hping.

[6] abatishchev, "Loic a network stress testing application," 2019, https://sourceforge.net/projects/loic/.

[7] S. Mergendahl, D. Sisodia, J. Li, and H. Cam, "Source-end DDoS defense in IoT environments," in *Proceedings of the 2017 Workshop on Internet of Things Security and Privacy*, Dallas, TX, USA, November 2017.

[8] T. M. Thang and K.-V. Nguyen, "FDDA: a framework for fast detecting source attack in web application DDoS attack," in *Proceedings of the Eighth International Symposium on Information and Communication Technology*, Nha Trang, Vietnam, December 2017.

[9] R. Biswas, S. Kim, and J. Wu, "Sampling rate distribution for flow monitoring and DDoS detection in datacenter," *IEEE Transactions on Information Forensics and Security*, vol. 16, pp. 2524–2534, 2021.

[10] H. Rahmani, N. Sahli, and F. Kammoun, "Joint entropy analysis model for DDoS attack detection," in *Proceedings of the Fifth International Conference on Information Assurance and Security*, Xi'an, China, August 2009.

[11] K. Narasimha Mallikarjunan, A. Bhuvaneshwaran, K. Sundarakantham, and S. Mercy Shalinie, "DDAM: detecting DDoS attacks using machine learning approach," *Computational Intelligence: Theories, Applications and Future Directions-Volume I*, Springer, New York, NY, USA, 2017.

[12] M. Aamir and S. M. Ali Zaidi, "Clustering based semi-supervised machine learning for DDoS attack classification," *Journal of King Saud University-Computer and Information Sciences*, vol. 33, no. 2, 2019.

[13] M. Barati, A. Abdullah, N. I. Udzir, R. Mahmod, and N. Mustapha, "Distributed Denial of Service detection using hybrid machine learning technique," in *Proceedings of the International Symposium on Biometrics and Security Technologies (ISBAST)*, Kuala Lumpur, Malaysia, August 2014.

[14] M. A. M. Yusof, H. M. A. Fakariah, and Y. D. Mohamad, "Detection and defense algorithms of different types of DDoS attacks," *International Journal of Engineering and Technology*, vol. 9, p. 410, 2017.

[15] R. M. A. Ujjan, Z. Pervez, K. Dahal, A. K. Bashir, R. Mumtaz, and J. González, "Towards sFlow and adaptive polling sampling for deep learning based DDoS detection in SDN," *Future Generation Computer Systems*, vol. 111, pp. 763–779, 2020.

[16] M. Zekri, S. E. Kafhali, N. Aboutabit, and Y. Saaadi, "DDoS attack detection using machine learning techniques in cloud computing environments," in *Proceedings of the 3rd International Conference of Cloud Computing Technologies and Applications (CloudTech)*, Rabat Marocco, October 2017.

[17] J. Hou, P. Fu, Z. Cao, and A. Xu, "Machine learning based DDoS detection through NetFlow analysis," in *Proceedings of*

the MILCOM 2018-2018 IEEE Military Communications Conference (MILCOM), Los Angeles, CA, USA, October 2018.

[18] F. S. d. L. Filho, F. A. F. Silveria, A. d. M. B. Junior, G. Vargas-solar, and L. F. Silverira, "Smart detection: an online approach for DoS/DDoS attack detection using machine learning," *Security and Communication Networks*, vol. 2019, Article ID 1574749, 15 pages, 2019.

[19] M. Idhammad, K. Afdel, and M. Belouch, "Semi-supervised machine learning approach for DDoS detection," *Applied Intelligence*, vol. 48, no. 10, pp. 3193–3208, 2018.

[20] S. S. Priya, M. Sivaram, D. Yuvaraj, and A. Jayanthildevi, "Machine learning based DDoS detection," in *Proceedings of the International Conference on Emerging Smart Computing and Informatics (ESCI)*, Pune, India, March 2020.

[21] P. S. Saini, S. Behal, and S. Bhatia, "Detection of DDoS attacks using machine learning algorithms," in *Proceedings of the 7th International Conference on Computing for Sustainable Global Development (INDIACom)*, New Delhi, India, March 2020.

[22] R. Doshi, N. Apthorpe, and N. Feamster, "Machine learning DDoS detection for consumer internet of things devices," in *Proceedings of the IEEE Security and Privacy Workshops (SPW)*, San Francisco, CA, USA, May 2018.

[23] X. Yuan, C. Li, and X. Li, "DeepDefense: identifying DDoS attack via deep learning," in *Proceedings of the IEEE International Conference on Smart Computing (SMARTCOMP)*, Hongkong China, May 2017.

[24] B. P. Welford, "Note on a method for calculating corrected sums of squares and products," *Technometrics*, vol. 4, no. 3, pp. 419-420, 1962.