Journal of Information Security and Applications 60 (2021) 102879

Contents lists available at ScienceDirect

Journal of Information Security and Applications

journal homepage: www.elsevier.com/locate/jisa

Low-rate DoS attack detection method based on hybrid deep neural networks

Congyuan Xu, Jizhong Shen *, Xin Du

College of Information Science and Electronic Engineering, Zhejiang University, Hangzhou 310027, China

ARTICLE INFO

Keywords:
Network security
Intrusion detection system
Low-rate DoS
Convolutional neural network
Gated recurrent unit
Deep learning

ABSTRACT

Low-rate Denial of Service (LDoS) attacks use the low-rate requests to achieve the occupation of the network resources and have strong concealment. The traditional signal analysis based detection methods are challenging to detect LDoS attacks in the fluctuating legitimate traffic. In this paper, an LDoS attack detection method based on hybrid deep neural networks is proposed using one-dimensional convolutional neural network and gated recurrent unit. In order to evaluate the proposed detection method in the real scenarios, we captured real legitimate traffic from a website in the datacenter, and carried out a variety of real LDoS attacks on the mirror of the website in the laboratory environment to obtain real attack traffic. The detection results on the real traffic show that the proposed detection method does not need to extract features manually and can effectively detect LDoS attacks in fluctuating HTTP traffic with an average detection rate of 98.68%, which is more advantageous than MF-DFA or power spectral density based detection methods.

1. Introduction

The Denial of Service (DoS) attack is one of the oldest attacks on the Internet that can date back to 1983 [1], and it is a kind of attack that prevents legitimate users from accessing services by over-occupying network bandwidth or computer resources [2]. Nowadays, DoS attacks are still one of the main cybersecurity threats. The traditional DoS attackers send network packets to the target servers at a very high rate, causing the target's network to be congested, so that the requests from legitimate users' requests are rejected. For example, the well-known Distributed Denial of Service (DDoS) attack causes great harm as the attackers usually control a large number of computers to attack the same target at the same time [3]. The theory of DDoS attack is simple, but it requires a large amount of computer resources, so that the cost is relatively high, and the high-rate traffic is easy to detect with inadequate concealment. In contrast, the Low-rate Denial of Service (LDoS) attacks can achieve the occupation of resources by constructing legal and low-rate requests on the application layer. It needs much less traffic to achieve the purpose of the attacks compared to the traditional DoS attacks [4]. LDoS attacks are low-cost and have strong concealment. The attack traffic is often overwhelmed by legitimate network traffic. As a result, the detection of LDoS attacks is more difficult and poses new challenges to cybersecurity.

The research on LDoS detection is not perfect at present. Most detection methods focus on signal analysis techniques and are verified on simulators rather than real network traffic. As LDoS attacks can be regarded as intricate patterns, deep learning technology can be used to automatically mine features from LDoS attacks and achieve end-to-end detection. In order to overcome the shortcomings of traditional signal analysis methods in detecting LDoS attacks, an LDoS attack detection method based on hybrid deep neural networks is proposed based on one-dimensional convolutional neural network (CNN) and gated recurrent unit (GRU). We obtain network traffic from real scenarios and conduct a detailed evaluation. We collect the real user traffic from the website of a university and select the part that does not contain DoS attacks as the legitimate traffic. Then an infiltration and forensics system is used to capture a variety types of LDoS attacks on the isolated website mirror and get the traffic containing the LDoS attacks. By analyzing the captured traffic, we can evaluate the proposed detection method.

This paper is organized as follows. Section 1 introduces the research background. Section 2 reviews the related works on LDoS attack detection and analyzes the pros and cons of them. Section 3 presents preliminaries including LDoS attacks and network traffic capture. Section 4 presents the proposed LDoS attack detection method based on hybrid deep neural networks in details. Section 5 presents the evaluation metrics and experiment results. Section 6 provides discussion of comparisons of two state-of-the-art detection methods and future work. Section 7 concludes the paper.

2. Related works

In order to detect LDoS attacks, several detection methods have proposed. Wu et al. proposed a method called MSABMS, which can detect

* Corresponding author.
E-mail address: jzshen@zju.edu.cn (J. Shen).

https://doi.org/10.1016/j.jisa.2021.102879

Available online 1 June 2021

LDoS attacks with only 30 to 60 s network traffic through small signal analysis, and the simulations were performed on the Network Simulator 2 (NS2) [5]. Wu et al. also proposed a method to estimate the Round-Trip Time (RTT) by spectrum analysis, and digital filters were used to complete the detection, and the simulation results on the NS2 showed that the detection rate could reach 81.36% [6]. Simsek et al. proposed a metric called Mean Internet Protocol Packet Delay Variation (MIPDV) to detect LDoS attacks, and simulations on the NS2 showed that the proposed detection method can achieve a high detection rate in a relatively short time [7]. Chen et al. introduced Fourier Power Spectrum Entropy (FPSE) and Wavelet Power Spectrum Entropy (WPSE) to detect LDoS attacks. The effectiveness of these two information metrics in detecting LDoS attacks were verified by both NS3 and real network traffic [8]. Besides, Chen et al. pointed out that due to the lack of public attack traffic, the real network traffic experiments only used legitimate traffic to detect, and evaluated the proposed method by calculating false alarm rate, which made the results unconvincing. Wu et al. utilized the multi-fractal characteristics of the network traffic, proposed a detection method based on Multi-Fractal Detrended Fluctuation Analysis (MF-DFA), and the attacks were detected by estimating the Holder exponent. The experiments on NS2 and laboratory environment showed that the Holder exponent changed when an attack occurred [9]. In the laboratory environment experiment, an FTP server was used as the target to be attacked. When the attacks occurred, the FTP server was performing stable data transmission, which is different from the real attack scenario with fluctuating traffic. Agrawal et al. used Power Spectral Density (PSD) to detect LDoS attacks and the detection of Slowloris attack was evaluated in a controlled environment, and the results showed that the PSD of attack traffic is different from legitimate traffic, which can be used as a detection criterion. The false positive rate was 3.7%, and the false negative rate was 4.9% [10]. Yue et al. proposed a detection method using a wavelet energy spectrum to extract network traffic features and an artificial neural network was used to classify network traffic, while the detection experiments were also performed on an FTP server [11]. Xiang et al. proposed a detection method based on the generalized entropy metric and the information distance metric. By analyzing the public network traffic, they artificially delineate the low-rate DoS attack traffic according to the packet rate. It is found that in the real attack scenario, the LDoS attack traffic is mixed with legitimate network traffic, which is very different from the high-rate DoS traffic [12]. Pratomo et al. proposed an unsupervised detection approach using an autoencoder to identify outliers in the network traffic. Experiments show that it can detect most of the low rate attack traffic [13]. It shows that deep learning techniques are feasible for detecting low-rate attacks.

The aforementioned work mainly used the signal analysis techniques to deal with network traffic, demonstrates the feasibility of detecting LDoS attacks, but has the following shortcomings. Firstly, it is not appropriate to use a network simulator (NS2 or NS3) to simulate LDoS attack traffic, while the real LDoS attack traffic is generated dynamically by the attack program, which is time-varying and more complex than the simulated traffic. Secondly, the types of real LDoS attacks are diverse. Currently, several open-source attack programs can be downloaded on the Internet, and the attack methods of different programs are very different. The LDoS attack is not one type of attack, but a general term for a class of attacks. For example, in order to detect an HTTP service LDoS attack, it is unreasonable to take FTP traffic as legitimate traffic. These two shortcomings will bring systematic errors to the evaluation of detection methods. For example, in the experiment in literature [9], whether in network simulator or laboratory simulation environment, legitimate traffic and attack traffic can be directly distinguished by packet rate in the time domain, without calculating the Holder exponent. This is because the simulated legitimate traffic is the data transmission process of an FTP server, which is almost unchanged in the time domain, while the attack traffic is simulated by periodic signals, which also deviates from the real scenarios. In addition, the above work relies on the feature extraction method carefully designed by security experts. If there are new types of attacks in the real environment, it is necessary to adjust the feature extraction parameters manually, so that the adaptability of the detection methods needs to be improved.

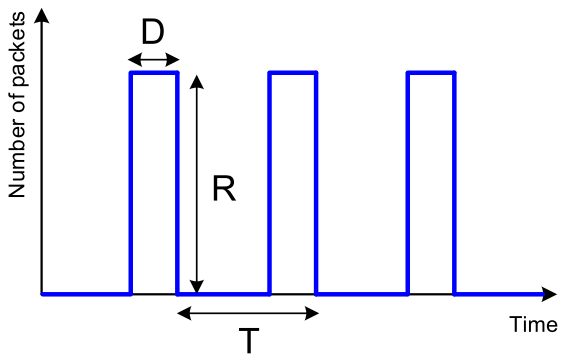

Fig. 1. The theoretical model of low-rate DoS attack.

3. Preliminaries

3.1. Low-rate denial of service attack

3.1.1. Theoretical model

The network traffic of LDoS attack can be described by a theoretical model as a triplet (T, D, R). As shown in Fig. 1, T represents the attack period, D represents the attack duration, and R represents the attack power [14]. In fact, the theoretical model can describe any network traffic with periodicity, and only the attack period T can be measured accurately. The attack duration D is designed to describe the duration of a burst transmission of attack traffic, but the LDoS attack program usually stays connected with the target for a long time until the target disconnects it. Therefore, the start and end times of the attack are difficult to measure accurately, and D cannot be measured accurately. Attack power R is usually expressed by the packet transmission rate, but the packet rate of an LDoS attack is low and does not fill the network bandwidth. Once an attack is started, the target server can still generate legitimate traffic before the target server completely dead. Therefore, in a real LDoS attack scenario, network traffic is superposed by legitimate traffic and attack traffic. To obtain the true attack power, the measured attack power R needs to be subtracted from the legitimate traffic. In summary, the triplet describes a theoretical model of an LDoS attack, which is an abstract depiction of attack traffic, and should not generate attack traffic based on this model in the network simulators.

3.1.2. Real attacks

In order to get real attack traffic, we collected several LDoS attack programs such as Slowloris, Slowhttptest, Pwnloris, Torshammer, and Httpbog that are available on the Internet. Then we construct an isolated LAN environment in the laboratory so that attack traffic can be obtained by running the attack programs. Each attack program runs for 60 s and traffic information was recorded at a statistical interval of 1 millisecond. According to the existing detection methods, the running time of 60 s and statistical interval of 1 millisecond is considered to be sufficiently long and sufficiently accurate [5]. Considering the time-varying and self-similarity of network traffic [15], the time domain is not enough to fully exhibit the features of the attack traffic, so we introduce the short-time Fourier transform method in time–frequency domain analysis [16], and the time–frequency domain is exhibited, too. Since the statistical interval is 1 millisecond, the sampling rate of the signal can be considered as 1000 Hz. According to the Nyquist sampling

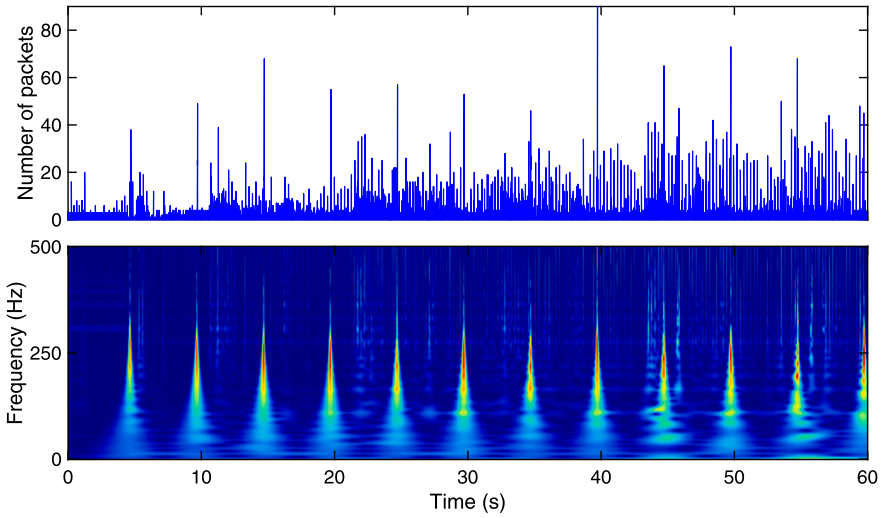

Fig. 2. Time domain and time–frequency domain diagram of the Slowloris attack traffic.

theorem, the frequency range of the time–frequency domain is 0 to 500 Hz.

(1) Slowloris

The Slowloris program was first published in Perl, and we use a Python rewritten and widely spread version in this paper [17]. The Slowloris can send a large number of HTTP headers and try to stay connected unless the target server disconnected actively. During the connecting time, Slowloris sends incomplete HTTP requests to make the server spend a lot of time waiting for the request to finish. The network bandwidth occupied by establishing and maintaining the connections is small, but it will occupy the target server resources for a long time, thus keeping the target server busying.

(2) Slow POST and Slow Read

Slow POST and Slow Read attacks are implemented by the Slowhttptest program [18], which is written in C++ and can conduct a variety of LDoS attacks.

The principle of Slow POST attack is similar to that of Slowloris. The attacker sends POST requests slowly to force the server to keep the connections. Since POST requests are typically used to submit data to the target server, the server usually sets a longer connection timeout value. The limitation is that there is an area on the web page provided by the target server that can submit data. Due to the increasing number of interactive elements on modern web pages, basically every website has areas where data can be entered, so POST requests are ubiquitous.

The Slow Read attack is exactly the opposite of the Slow POST attack. The attack program actively reduces the client's TCP window size, so that the target server's response data can only be received at a slow speed, then extending the connection time. When the Slowhttptest initiates the Slow Read attack, the TCP window size is set to 1, so each packet can only carry one byte, and a simple web page response data requires a large number of packets to be transmitted. The rate of packets will increase to some extent when the Slow Read attack occurs, but each packet is small and the occupied network bandwidth is still small.

(3) Pwnloris

The Pwnloris program is an improvement to the Slowloris program, first appeared in 2018, written in Python [19]. Pwnloris supports the use of TOR to enhance anonymity, while the traffic changes become more complex [20].

(4) Torshammer

The Torshammer is an improved Slow POST attack program written in Python [21]. Although in principle it is consistent with the Slow POST attack, the Torshammer program has abandoned the implementation of periodically generating network traffic, so no significant periodic features are found in both the time domain and the time–frequency domain.

(5) Httpbog

The Httpbog is an attack program that runs on Windows operating systems and is written in C# [22]. The principle of Httpbog attack is similar to the Slow Read, which implements the attack by slowly receiving the response data of the target server. Due to the widespread use of Windows operating systems, more clients can be used to launch attacks. Httpbog weakens the feature of attack power and attack period, so that it stays more concealed.

Figs. 2 and 3 are the time domain and time–frequency domain diagrams of the Slowloris attack and the Torshammer attack, respectively. The Slowloris attack did not exhibit periodicity as the theoretical model, but in the time–frequency domain diagram, the mid-high frequencies exhibited significant periodic peaks. The Torshammer attack did not have periodic features in both the time and time–frequency domains. Therefore, a simple theoretical model cannot fully describe the real attacks, and using periodic signals to simulate LDoS attacks is not appropriate.

3.2. Network traffic capture

Due to the limitations of the network simulator, we need to design a network capture system to get real network traffic, and then build a dataset to evaluate the proposed detection method. In order to capture legitimate traffic from a university website in the datacenter and capture attack traffic generated by the attack programs in the laboratory environment, we built a high-performance network traffic capture system consisting of a capture program, RAM buffers, a transfer program, and a disk array. As shown in Fig. 4, the capture program utilizes the dumpcap program to read the traffic data on the network interface card into the RAM buffers. Dumpcap program is also the capture program used by Wireshark, which is a widely used open-source network packet analyzer [23]. The whole RAM buffer is divided into four blocks, each of which is 2 GB, which is used to absorb the large instantaneous traffic that may occur and ensure the integrity of the captured data. The transfer program monitors the RAM buffers in real-time and transfers the data to the disk array. The disk array is used to persist the captured data for later analysis. The capture system uses a ping-pong-like operation to form four RAM buffers into a circular queue, which avoids the occurrence of any capture interruption accident caused by the storage system bottleneck. Fig. 4 shows that the capture program is writing data to RAM buffer C, and the transfer program is transferring data to the disk array after RAM buffer B has been filled. The stress test shows that the traffic capture system can stably capture the network traffic of a Gigabit Ethernet full-rate transmission without any packet loss.

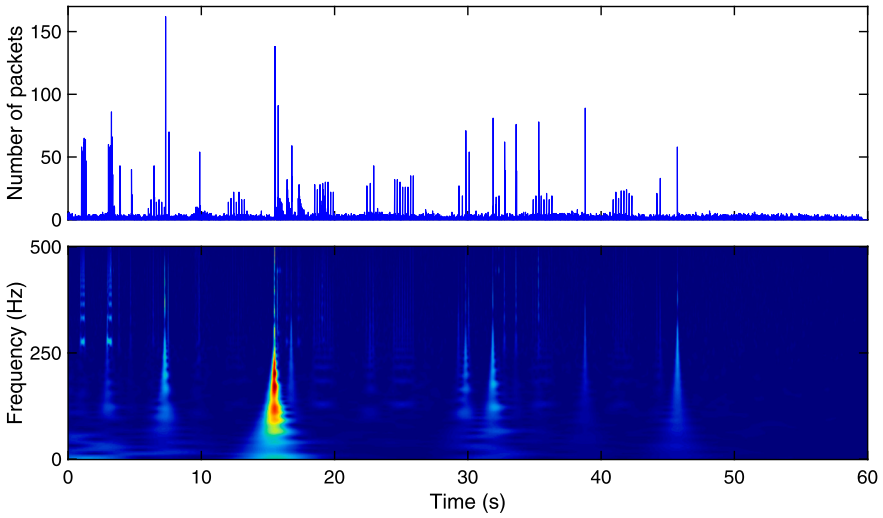

Fig. 3. Time domain and time–frequency domain diagram of the Torshammer attack traffic.

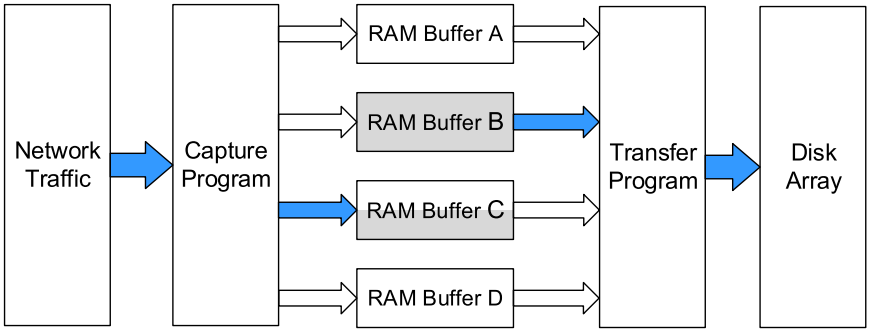

Fig. 4. The architecture of network traffic capture system.

There are two types of network traffic need to be collected: legitimate traffic and attack traffic. As it is difficult to simulate the traffic generated by a large number of legitimate users in the laboratory network environment, we chose a real website of a university as the source of legitimate traffic. A simplified network topology for capturing legitimate traffic is shown in Fig. 5. The Web server is connected to the Internet through a firewall, a switch, and an Internet access device. We connect the network traffic capture system to the Web server via the switch by port mirroring. In order to ensure the security of the datacenter, the capture of attack traffic cannot be carried out in the datacenter. Thus, we have built a LAN in the laboratory, and the network topology is shown in Fig. 6. A self-built Web server is deployed with a mirror of the website, which constitutes the same website as the Web server in the datacenter. Traditional firewalls currently have a weak defense against LDoS attacks, so the firewall will not interfere with the attack experiments. The network traffic capture system is still connected to the self-built Web server via the switch by port mirroring. An attacker can access the self-built Web server through the router and initiates attacks. Most LDoS attack programs run on the Linux operating system, and we used a computer running Kali Linux distribution [24] as the Attacker 1. The Pwnloris program needs to run on Windows, and we used a computer running the Windows 10 operating system as the Attacker 2.

A variety of LDoS attack programs are used to launch attacks on the self-built Web server, and the network traffic capture system captured all traffic during the attacks. It should be noted that the traffic captured in this way only contains the traffic generated by the attack program, while the LDoS attack does not completely occupy the network bandwidth, and in most cases, only a small portion of the bandwidth is

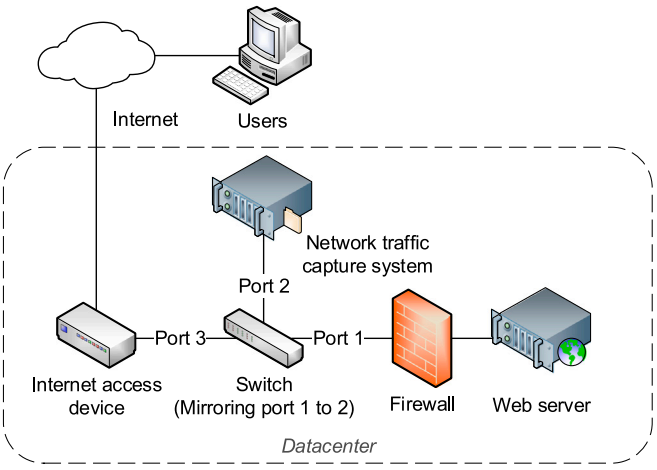

Fig. 5. Network topology for capturing legitimate traffic.

occupied. In a real scenario, before the target server is completely dead, the traffic we captured will be a superposition of attack traffic and legitimate traffic, which is different from the scenario of high-rate DoS attacks. Therefore, the best realistic scenario that can be achieved is that the "real" attack traffic should be the superposition of the attack traffic captured in the laboratory and the legitimate traffic captured in the datacenter. In this case, the features of the attack traffic may also be covered over by the features of legitimate traffic, which increased the detection difficulty. Besides, we use the traffic captured from the

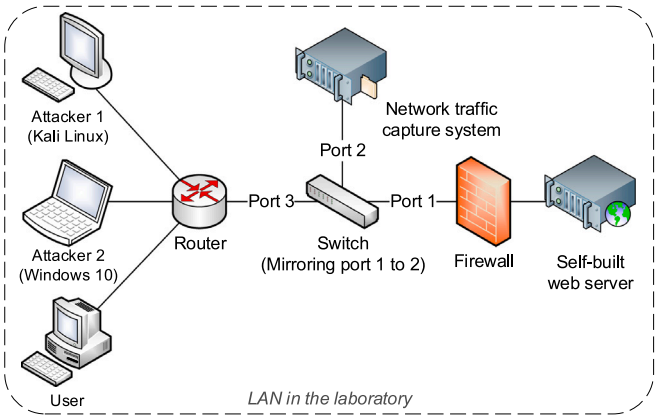

Fig. 6. Network topology for capturing attack traffic.

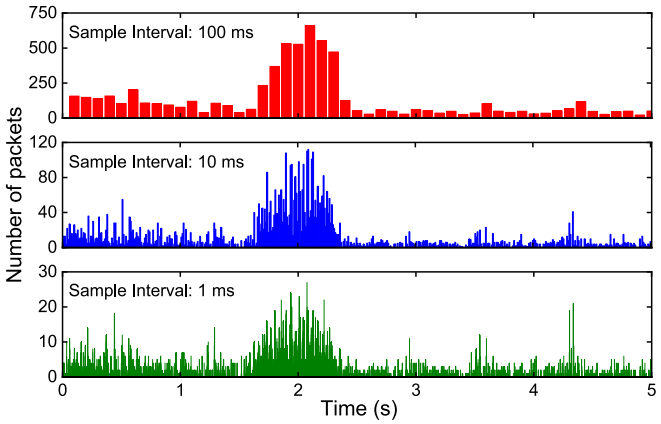

Fig. 7. Time domain of network traffic sampled with different sampling intervals.

datacenter as legitimate traffic because it has excluded denial of service attacks manually. Although there may be other types of attacks, the traffic generated by those attacks is very small, which can be ignored when detecting DoS attacks.

The dataset we used for evaluation is built in the following way. First, capture the six types of attack traffic generated by the attack program introduced in Section 3.1.2. In our experiments, each attack program runs 1000 times, and each run time is 60 s. It has been confirmed that all attack programs will start attacking after startup as soon as possible, so 60 s is long enough. In this way, 16 h and 40 min of pure attack traffic is captured for each attack program. Then, 120 s of legitimate traffic is randomly selected from the traffic captured from datacenter as a segment, and a total of 2000 segments are selected. In this way, 66 h and 40 min of legitimate traffic is selected for each attack program. 1000 segments are superimposed with pure attack traffic, marked as attack traffic samples; the other 1000 segments remains unchanged and are marked as legitimate traffic samples. In the real scenarios, the moment when an attack initiated is random. Therefore, in each attack traffic sample, the position of 60 s of pure attack traffic superimposed to 120 s of legitimate traffic is also randomly specified. In this way, we build a dataset that includes a total of 100 h of attack traffic and 400 h of legitimate traffic. We randomly select 60% of the samples for training and 40% of the samples for testing.

In addition, we should also consider that in the real scenarios, the type of attack is unknown before detection. Our detection method should have the ability to detect all the six types of attacks at the same time through one detection process. Therefore, from the aforementioned six types of attack samples, we randomly choose 800 samples for each type of attacks, and compose an "overall" dataset with 4800 samples. Ten "overall" datasets are generated randomly, so that ten parallel experiments can be performed to get the average metrics that are closer to real scenarios. Since the "overall" datasets contain multiple attack types, they are further increased the difficulty of detection. The overview of the created dataset is shown in Table 1.

4. Proposed detection method

4.1. Network traffic sampling

To convert network traffic to time series, the network traffic needs to be sampled. In terms of simplicity and efficiency, it is more convenient to count the packets than calculate the accurate data size. Let $s(t_1, t_2)$ be the number of packets that go through a network node in the time interval $t \in (t_1, t_2]$, and take a sampling interval $T \in \mathbb{R}^+$, a time series $x(n)$ is obtained:

$$x(n) = s((n-1)T,\ nT),\ n \in \mathbb{N}^+,\ x(n) \in \mathbb{N} \tag{1}$$

The sampling process for network traffic is similar to the sampling process for analog signals. If the sampling interval T is smaller, we can get a better time resolution. As shown in Fig. 7, a segment of network traffic is sampled with three different sampling intervals: $T = 100$ ms, 10 ms, and 1 ms, and is converted to three time series with different time resolutions. Since the minimum delay of the network is generally on the order of a millisecond, the sampling interval of less than 1 ms does not have the practical significance of improving the detection capability. The direct observation of the 100 ms sampling interval can reveal that most details have been lost. Therefore, when detecting an LDoS attack, a sampling interval of the order of 1 ms or 10 ms is suitable. Subsequent detections are based on the time series sampled with 10 ms sampling interval, and satisfactory detection results can be obtained. In the real scenario, the attack traffic will be submerged in the legitimate traffic, and our goal is to take attack traffic as target signal to be detected and legitimated traffic as noise. The proposed detection method needs to be able to recover target signal from noise.

4.2. Convolution filtering

Considering a manual feature extraction process for LDoS attack detection, Slowloris is taken as an example. Firstly, by observing the time domain representation of the signal, weak periodic peaks can be found, but not very significant. Then by observing the time–frequency domain representation of the signal, obvious periodic peaks can be found, that is, the intensity of fixed interval changes in the high frequency part. Therefore, the feature extraction of the traffic should start from the time domain and frequency domain.

Lots of successful applications of deep neural networks in computer vision indicate that the convolutional neural network (CNN) is a type of self-learning filter [25]. The one-dimensional convolutional neural network (1D-CNN) can extract short-term frequency domain features on time-domain signals [26]. A one-dimensional convolution operation is as follows:

$$f_{out} = \sigma(\sum w_{d,c} \circ X_{d,c} + b) \tag{2}$$

In Eq. (2), d is the size of the convolution kernel, c is the number of the filters, X is the tensor input to the convolution kernel, σ is a nonlinear activation function, $\circ$ is the Hadamard product, w and b are the weight and bias of the convolution kernel.

As the signal length is much longer than the convolution kernel, 1D-CNN performs convolution operation by sliding convolution kernel. As shown in Fig. 8, a convolution kernel of size d slide over a signal of 10 s. In order to clarify the figure, the convolution kernel and step in Fig. 8 are set larger, and the number of channels is set to 1.

The convolution operation has translation invariance. If the target signal shifts back or forth in time, the output of the convolution is

Table 1
Overview of the created dataset.

Sample type	Number of samples	Dimension of each sample[a]	Duration of each sample	Attack duration of each sample
Legitimate traffic	6000	12,000	120 s	0 s
Attack 1 (Slowloris)	1000	12,000	120 s	60 s
Attack 2 (Slow POST)	1000	12,000	120 s	60 s
Attack 3 (Slow Read)	1000	12,000	120 s	60 s
Attack 4 (Pwnloris)	1000	12,000	120 s	60 s
Attack 5 (Torshammer)	1000	12,000	120 s	60 s
Attack 6 (Httpbog)	1000	12,000	120 s	60 s
"Overall"[b]	4800	12,000	120 s	60 s

[a]Taking sampling interval $T = 10$ ms as an example.
[b]Sampled from attacks 1 to 6.

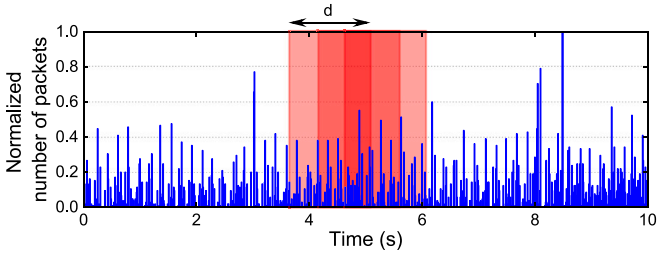

Fig. 8. One-dimensional convolution on network traffic.

only the movement in the corresponding direction, while the value does not change. This is very useful for detecting LDoS attack traffic, as in real scenarios, attack traffic may occur at any time. Even if the network traffic is segmented manually, the attack traffic may appear anywhere in a segment. CNN extracts all the features of all locations of the signal, while only part of the features is useful. For example, for the detection of the Slowloris attack, the mid-high frequency features of the signal are most significant, while the low frequency part is less important. Therefore, pooling operations need to be added after the convolution operations, taking the max pooling as an example, in the adjacent convolution output, the maximum value is selected as the pooling output.

The size of the receptive field should be considered when taking convolutional neural networks as signal filters. Taking a convolution kernel of size d_1 as an example, a non-overlapping convolution operation is considered here, that is, the size of step and convolution kernel are the same, and the pooling operation is not considered at this time. In this case, the receptive field for the signal is d_1. Then, consider a two-layer convolutional neural network with the sizes of the convolution kernels d_1 and d_2, respectively. After the first layer of convolution, a segment of length d_1 from the input signal is mapped to a point. In the second layer convolution operation, a segment of length $d_1 \times d_2$ from the input signal is involved in the convolution operation, so the receptive field of the second layer of convolution operation is $d_1 \times d_2$. It can be seen that one-dimensional CNN can achieve hierarchical feature extraction by cascading multiple CNN layers, and the receptive field increases layer by layer.

In summary, the convolutional neural network used as filters to extract features of LDoS attacks is cascaded by multiple convolution layers and max pooling layers.

In this paper, we cascaded 6 CNN layers with a convolution kernel size of 3, and the receptive field reached $3^6 = 729$. If the sampling interval is 10 ms, the maximum time span of feature extraction is 7.29 s, which can meet the feature extraction requirements of the LDoS Attacks.

4.3. Time series modeling

The features extracted by the convolutional neural network from network traffic are still time series, that is, the sequence of features in feature vectors remains strictly in the order of time. We can use the recurrent neural network (RNN) to model the time series and extract the features of time series for classification. Traditional RNNs encounter gradient vanishing or explosion problems, and some specific RNN structures are proposed to alleviate these problems. According to the literature [27], the experimental results on the network intrusion detection datasets KDD 99 and NSL-KDD showed that bidirectional gated recurrent unit (BGRU) is the most suitable structure to model time series for network traffic classification. KDD 99 and NSL-KDD datasets contain network traffic features that are extracted by fixed rules from raw network traffic. To detect LDoS attacks, we use CNN to extract the features of sampled network traffic. The features extracted from the sampled network traffic are essentially similar to the samples in the datasets. Therefore, BGRU is selected as the memory unit of the RNN for time series modeling of LDoS attack traffic features.

4.4. End-to-end detection

Combining CNN and RNN with BGRUs, we proposed hybrid deep neural networks for end-to-end detection of LDoS attacks. The structure of the proposed neural networks is shown in Fig. 9. The input of the hybrid deep neural networks is a sample of network traffic $x(n)$ to be detected, and the output is a label, which is the estimated probability of LDoS attack for the input sample.

Before input to the hybrid deep neural networks, the network traffic needs to be normalized. As $x(n) \in \mathbb{N}$, it only needs to be linearly scaled to $[0, 1]$, which can be achieved by dividing each point in the signal by a fixed maximum value. This maximum value is related to the sampling interval and can be denoted as Max. The exact value of Max is not hard to determine from the captured dataset. However, in the real scenarios, it cannot be guaranteed that the sampled network traffic value will not exceed Max. In order to prevent overflow, Max can be set larger, and the normalized preprocessing should be implemented as a truncated maximum:

$$x^*(n) = \begin{cases} x(n)/Max, & x(n)/Max \le 1 \\ 1, & x(n)/Max > 1 \end{cases} \tag{3}$$

In Eq. (3), $x(n)$ is a signal to be normalized, $x^*(n)$ is the normalized signal. Preprocessed signals are input into the proposed neural network. In Fig. 9, six convolution modules are used to extract features hierarchically, which are denoted as Conv 1 to Conv 6. Since the preprocessed signal is directly input to Conv 1, the number of filters of Conv 1 is set larger to extract the primary features as comprehensively as possible. Conv 2–6 modules extract the features of different scales layer by layer. "Conv1D, 3, 64" represents a one-dimensional convolution operation, the size of the convolution kernel is 3, the number of filters is 64; "ReLU" indicates that the activation function is the rectified linear unit [28]; "MaxPooling 1D, 3" represents a max pooling operation, the size is 3. "BGRU" denotes an RNN module with BGRUs as its memory units; "Flatten" is the connection layer, which flattens the output of the RNN into a one-dimensional vector; "FC" is a fully connected module, which is used for classifying operation. The output module

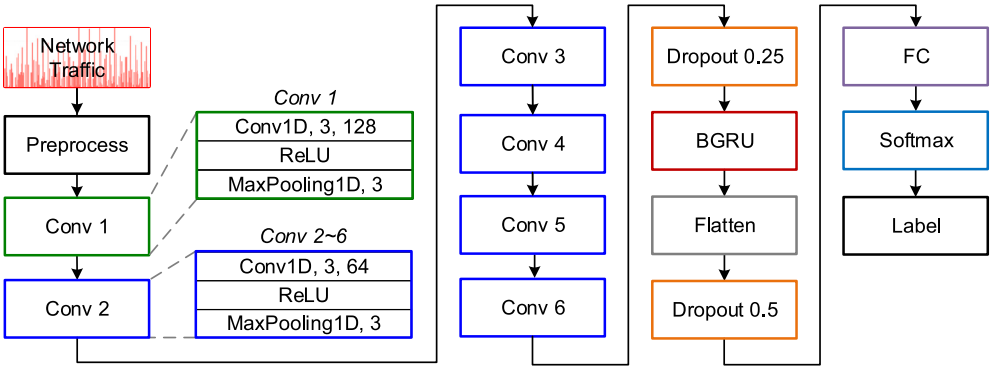

Fig. 9. The structure of the proposed hybrid deep neural networks for end-to-end detection of LDoS attacks.

uses Softmax regression to get the normalized estimated probability of LDoS attack. Before the "BGRU" and "FC" modules, the Dropout [29] modules are inserted to prevent the network from over-fitting, and the dropout probabilities are 0.25 and 0.5, respectively.

The proposed hybrid deep neural networks can be trained by the adam algorithm [30] and the loss function is defined by cross-entropy [31]:

$$L(\theta) = -\frac{1}{m}\sum_{i=1}^{m}[y^{(i)}\log(h_\theta(x^{(i)})) + (1-y^{(i)})\log(1-h_\theta(x^{(i)}))] \tag{4}$$

In Eq. (4), θ is the undetermined parameter, m is the batch size, $(x^{(i)}, y^{(i)})$ is the training data of the i-th pair input, h_θ is the model function defined by the parameter θ. The significance of end-to-end training is that only the input and output data need to be defined, and the network parameters need not be adjusted manually, which is equivalent to adaptive filtering of the input signal.

5. Evaluation

5.1. Metrics

Detecting LDoS attacks in legitimate network traffic can be considered as an object detection task. However, as described in Section 3.1.1, in the real scenarios, it is difficult to determine the start and end times of LDoS attacks accurately, and it is not necessary to determine them accurately. The samples in our dataset is consistent with the real scenarios, that is, given a network traffic segment, we need to determine whether there is LDoS attack traffic in it. In this way, the object detection task can be transformed into a classification task. For binary classification tasks, the results of classification can be correct or incorrect. All possible results can be divided into the following four outcomes:

1. True Positive (TP): attack samples are classified as attack ones;
2. True Negative (TN): legitimate samples are classified as legitimate ones;
3. False Positive (FP): legitimate samples are classified as attack ones (false alarms);
4. False Negative (FN): attack samples are classified as legitimate ones (missed detections).

For simplicity, TP, TN, FP, FN are used to represent the numbers of the four outcomes. On this basis, the precision, detection rate, accuracy, false positive rate, false negative rate, and F-measure can be defined as shown in Eq. (5) [32].

$$\begin{cases} Precision = \dfrac{TP}{TP+FP} \\ Detection\ Rate\ (DR) = \dfrac{TP}{TP+FN} \\ Accuracy\ (ACC) = \dfrac{TP+TN}{TP+TN+FN+FP} \\ False\ Positive\ Rate\ (FPR) = \dfrac{FP}{FP+TN} \\ False\ Negative\ Rate\ (FNR) = \dfrac{FN}{TP+FN} \\ F-measure = \dfrac{2\times Precision\times DR}{Precision+DR} \end{cases} \tag{5}$$

5.2. Detection results

The experiments were performed under the following hardware and software platforms: Intel Xeon E5-2660 v3 @2.6 GHz, 128 GB RAM, NVIDIA TESLA K40; Ubuntu 16.04 LTS, CUDA 9.0, cuDNN 7.0, TensorFlow 1.7.

We used Tshark (the command-line version of Wireshark) to implement network traffic sampling and statistics. As shown in Fig. 7 and the related description, the sampling interval should be between 1 ms and 100 ms. In order to determine the appropriate order of magnitude, we performed experiments on the datasets obtained at the sampling intervals of 10 ms and 1 ms. The results show that the sampling interval of 1 ms does not help improve the detection results, so 10 ms is sufficient. The following detection experiments use the sampling interval of 10 ms as the benchmark.

First of all, the convergence of the model needs to be confirmed, that is, check the value of the loss function after each epoch. As shown in Fig. 10, for the first 10 iterations, the value of the loss function drops rapidly. After 10 to 30 iterations, the value of the loss function tends to be stable. Therefore, the epoch value is set to 30 to conduct the following experiments. The values of the hyperparameters in the model are shown in Table 2. The distribution of parameter numbers of the proposed hybrid deep neural networks is shown in Table 3. Using the above configuration, it takes about 350 s to train a model and about 2 s to test one.

We did "1 vs. 1" detection firstly, which meant detecting one type of LDoS attacks at a time. Then we did "all vs. 1" detection that meant all six types of LDoS attacks in the dataset are mixed as an "overall" type. Detection results are shown in Table 4.

In Table 4, the "average" line refers to the average of the detection results of six types of LDoS attacks, while the "overall" line refers to the average detection results of the "overall" datasets to detect all the six types of LDoS attacks at the same time. Table 4 shows that the proposed LDoS attack detection method based on hybrid deep neural networks can effectively detect all the six LDoS attacks, with an average detection rate of 98.68% and an average F-measure of 0.9771. The type with the

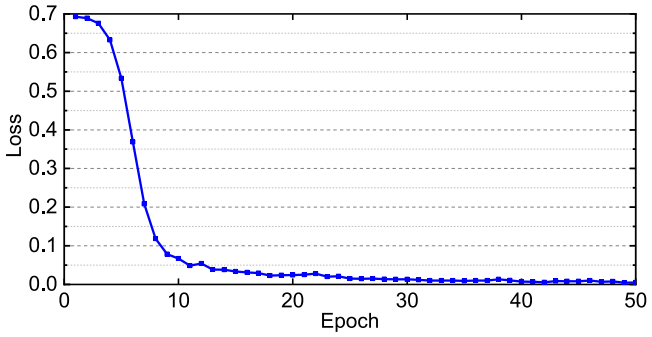

Fig. 10. The relationship between the loss and the epochs during training.

Table 2
Hyper-parameter configuration.

Hyper-parameter	Value
Max	1000
BGRU nodes	64
FC nodes	48
Batch size	32
Epoch	30
Learning rate	0.002
β_1	0.9
β_2	0.999
ϵ	10^{-8}

Table 3
Distribution of parameter numbers of the proposed hybrid deep neural networks.

Layer	Number of parameters
Conv 1	512
Conv 2	24,640
Conv 3	12,352
Conv 4	12,352
Conv 5	12,352
Conv 6	12,352
Dropout 0.25	0
BGRU	49,536
Flatten	0
Dropout 0.5	0
FC	295,058
Softmax	0
(Total)	419,154

Table 4
Detection results of six types of LDoS attacks.

Type	Accuracy (%)	DR (%)	FPR (%)	FNR (%)	Precision (%)	F-measure
Slowloris	98.25	98.99	2.49	1.01	97.52	0.9825
Slow POST	99.00	99.49	1.48	0.51	98.49	0.9899
Slow Read	97.75	99.48	3.83	0.52	95.96	0.9769
Pwnloris	97.25	98.48	3.94	1.52	96.04	0.9724
Torshammer	97.50	99.43	4.02	0.57	95.11	0.9722
Httpbog	96.75	96.19	2.63	3.81	97.58	0.9688
(Average)	97.75	98.68	3.06	1.32	96.78	0.9771
(Overall)[a]	96.74	96.82	3.20	3.18	96.71	0.9673

[a]The average results of ten experiments on ten "overall" datasets.

highest detection rate is Slow POST (99.49%) and the type with the lowest detection rate is Httpbog (96.19%). This is consistent with the observation in the time–frequency domain: Slow POST has significant periodic features in the time–frequency domain, so it is easy to be detected, while Httpbog features are not significant, so the detection is difficult. The average detection rate of "overall" is 96.82%, which is not far from the "1 vs. 1" detection. It can be concluded that the proposed detection method can effectively extract the features of LDoS attacks and has universality in detecting different types of LDoS attacks.

Since the 120-second sample contains only 60 s of the LDoS attack, we need to confirm whether the attack has been accurately detected. The output of the RNN is too abstract to interpret. However, the output of the CNN keeps the time series and can be compared with the input signal. So, we choose the output of Conv 6 module, the last layer of the convolutional neural network, as the target object. Here, we introduce the concept of class activation mapping (CAM) in the field of computer vision [33]. When classifying a specific sample, the proposed hybrid deep neural networks will output the classification result, but we also want to know which locations in the sample are the basis for decision making, that is, which locations activate the neurons in the neural network to classify a specific class. Due to the complex structure of the deep neural network and numerous parameters, it is not operability by manual derivation. Therefore, an algorithm named Grad-CAM is used to obtain Conv 6 class activation mapping [33]. Grad-CAM algorithm is used to visualize the basis of decision making, that is, by calculating the weighted average gradient of the specific predicted value to the target layer, the class activation mapping of the two-dimensional convolution neural network is obtained. For image classification, the class activation mapping of two-dimensional convolution can be plotted as the position of the image given by x-axis and y-axis coordinates. In this paper, Grad-CAM algorithm is used to obtain the activation of one-dimensional convolution neural network. One-dimensional activation value is obtained to plot the one-dimensional activation map. Here, the target layer is Conv 6, the target class is the class labeled as attack.

As shown in Figs. 11 and 12, we selected two representative samples for in-depth analysis. For each sample, the activation of the last layer of CNN is plotted, and it is compared with the time domain and time–frequency domain representations. The insertion location of attack traffic in the two samples starts at 30 s and ends at 90 s, and is marked with a red rectangle. The fixed attack start time is only used to facilitate plotting, in the actual scenario, the location of attack traffic insertion is random.

In Fig. 11, there is a strong legitimate traffic interference between 90 and 100 s, but it does not cause excessive activation of Conv 6. Conv 6 is specifically activated at the time period that contains only attacks. From the time–frequency domain representation, Conv 6 may be activated by some features of the high frequency part. In Fig. 12, Slow POST attacks are almost completely concealed due to the high intensity of legitimate traffic as background, and the distinguishing attack features cannot be directly found from time domain and time–frequency domain representations. However, the activation of Conv 6 showed that our proposed detection method can accurately find the attack location as the higher activation locations of Conv 6 correspond to the attack location. By using grad-CAM algorithm, we further verify that the proposed detection method can eliminate the interference of legitimate traffic and detect the location of LDoS attacks accurately.

6. Discussion

In order to compare with the existing methods, we randomly selected two segments of network traffic including LDoS attacks as the detection sample. As shown in Figs. 13 and 14, each segment of network traffic lasts 120 s. LDoS attacks start at 61 s and end at 120 s, and are marked with red rectangles. The activation of Conv 6 of the proposed detection method is still used to confirm whether the attacks are accurately detected. From the activations of Conv 6, it can be seen that for the two traffic segments, when attacks started, Conv 6 has obvious periodic fluctuations, and their higher activation areas correspond to the attack location, while in the legitimate part, the activations are lower. It can be concluded that the proposed detection method can accurately detect the LDoS attacks in the two traffic segments.

Due to the use of real traffic, the experimental results of proposed method cannot be directly compared with the results in the related

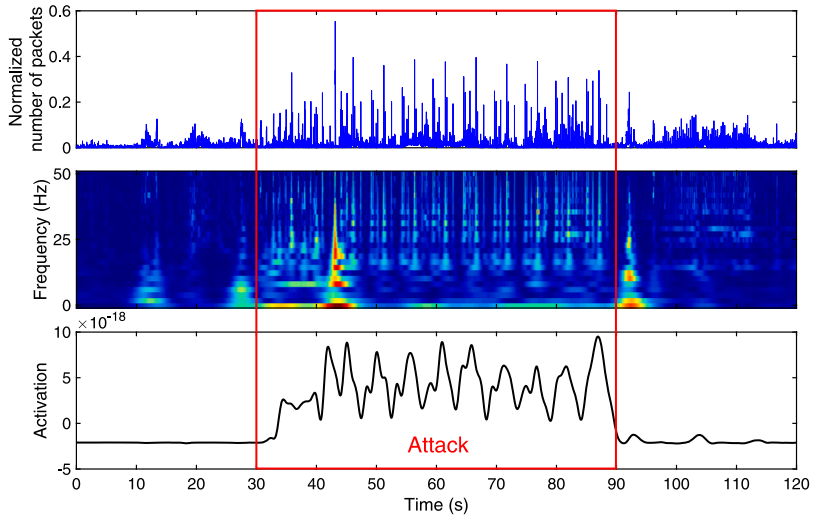

Fig. 11. Time domain, time–frequency domain, and activation of Conv 6 for a traffic sample containing a Pwnloris attack. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

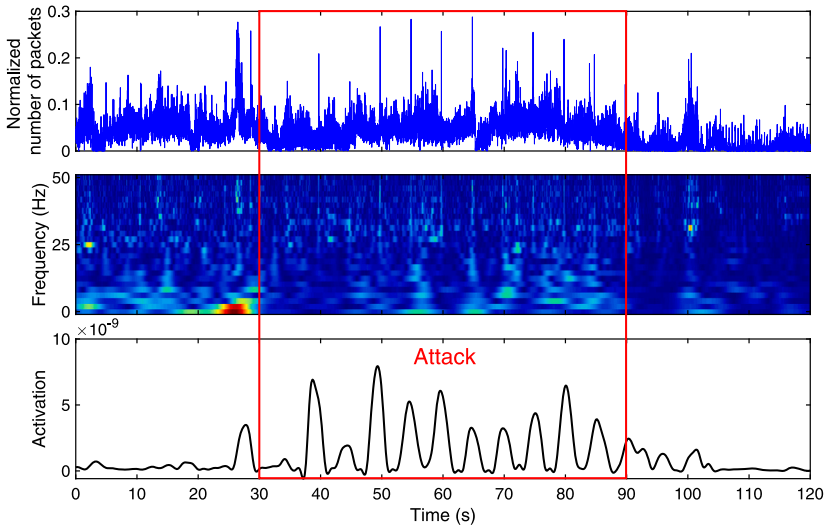

Fig. 12. Time domain, time–frequency domain, and activation of Conv 6 for a traffic sample containing a Slow POST attack. (For interpretation of the references to color in this figure legend, the reader is referred to the web version of this article.)

literature. Therefore, we implemented two of the state-of-the-art detection algorithms [9,10] and applied them to the real traffic that we captured, and the detection results show that the two algorithms both failed to detect LDoS attacks in the real traffic with a detection rate of 0%.

The first algorithm to be compared is the MF-DFA detection method proposed by Wu et al. [9], which used the multifractal characteristics of network traffic, and the wavelet-based Holder exponents were calculated. According to [9], when LDoS attacks started, the Holder exponents will be significantly decreased, and the threshold can be set to determine whether the attacks occurred. However, MF-DFA method is failed to detect the two traffic segments. When LDoS attacks started, the Holder exponents does not decrease significantly, and it is impossible to distinguish whether attacks occurred by setting the threshold. This is because the traffic we captured contains real HTTP traffic, which fluctuates greatly, rather than the smooth FTP traffic in literature [9]. Considering that the main target of LDoS attack is the HTTP server, it is more appropriate to use HTTP traffic than FTP traffic. Therefore, we think that in real scenarios, simply calculating Holder exponent is not enough to detect LDoS attacks effectively.

The second algorithm to be compared is the power spectral density (PSD) based detection method proposed by Agrawal et al. [10], and the power spectral densities of legitimate traffic and attack traffic were calculated, respectively. In literature [10], the basis of detecting LDoS attacks is that the power spectral density of attack traffic is higher in the low frequency part, while the legitimate traffic distributes more evenly. The detection results of the two traffic segments showed that there is no significant difference in power spectral density between attack traffic and legitimate traffic, and it is impossible to distinguish whether an attack occurs or not directly. This is because traffic intensity of LDoS attacks is very low, and the attack traffic is almost submerged in the legitimate traffic. The differences of power spectral density between the two parts of traffic in Figs. 13 and 14 are mainly due to the change of legitimate traffic in the two segments, not to the LDoS attacks.

Similar experiments are carried out on more different network traffic samples containing LDoS attacks. The results are similar to those in Figs. 13 and 14, and only the proposed method can detect LDoS attacks effectively. It can be concluded that the proposed LDoS attack detection method based on hybrid deep neural networks can effectively detect

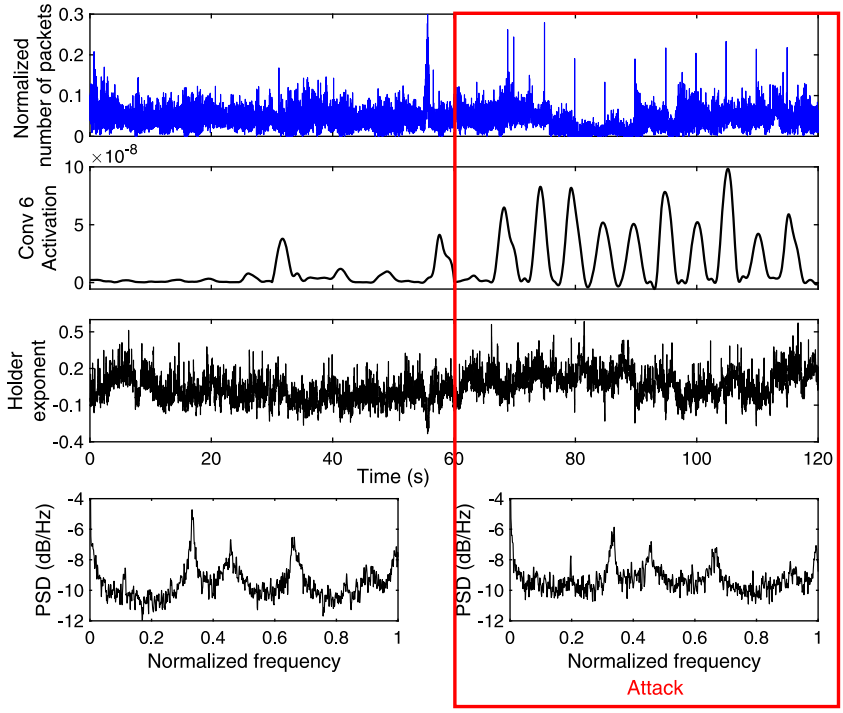

Fig. 13. Comparison with the Holder exponent and power spectral density based detection methods (sample 1).

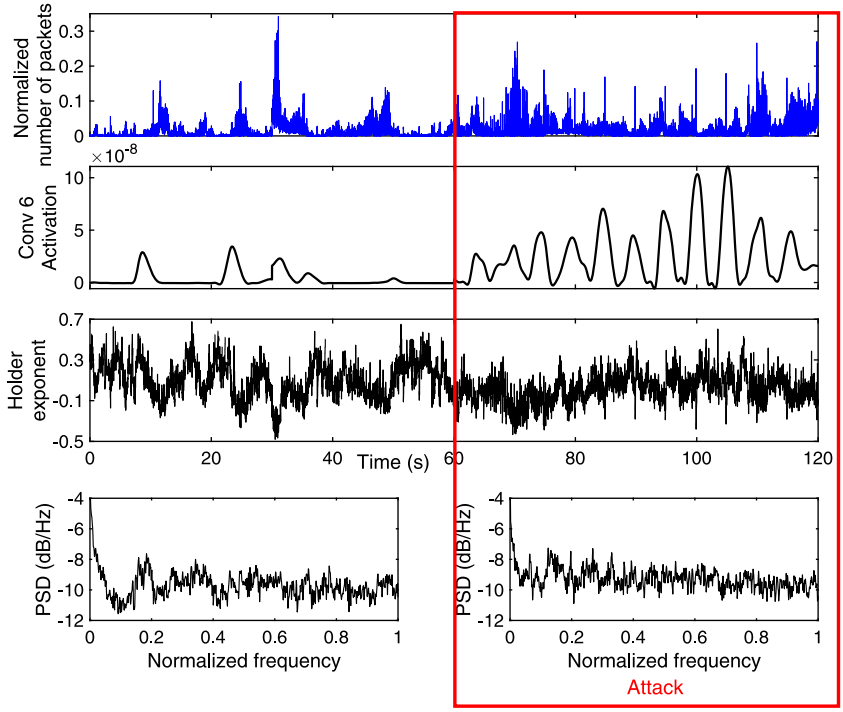

Fig. 14. Comparison with the Holder exponent and power spectral density based detection methods (sample 2).

LDoS attacks in fluctuating HTTP traffic, which is more advantageous than MF-DFA or power spectral density based detection methods.

In addition, as we converted LDoS attack detection as a time series classification problem, we also compared several time series classification methods, including traditional algorithms and deep learning-based SOTA methods. We implemented these methods and compared them on the "overall" dataset. The comparison results are shown in Table 5. It can be found from Table 5 that the proposed method stands out among all the compared methods.

Future work mainly includes the following points. First, deep neural networks are vulnerable to adversarial samples [40]. The detection method proposed in this paper is based on a hybrid deep neural network, so it is necessary to analyze the adversarial samples and take appropriate precautionary measures. Secondly, although the scale of

Table 5
Comparison with time series classification methods on the "overall" dataset.

Method	Accuracy (%)	DR (%)	FPR (%)	FNR (%)	Precision (%)	F-measure
KNN (K=1) [34]	76.35	69.05	6.60	30.95	96.07	0.8035
BOSSVS [35]	60.10	63.23	41.92	36.77	49.48	0.5552
ConvLSTM [36]	87.71	81.42	2.17	18.58	98.37	0.8909
CNTC [37]	83.23	83.69	17.13	16.31	78.96	0.8126
DTWCNN [38]	90.10	84.47	1.55	15.53	98.78	0.9106
MACNN [39]	89.48	84.55	3.53	15.45	97.14	0.9041
Proposed method	96.74	96.82	3.20	3.18	96.71	0.9673

our proposed hybrid deep neural network is not very large in the field of deep learning, it is still difficult to meet the needs of online detection of network traffic. The network traffic is bursty, so the neural network model needs to be compressed to reduce the time and space complexity and calculation delay, and meet the requirements of real-time detection. Also note that, due to the inherent limitations of supervised learning, the detection method we proposed is not a "silver bullet". For unknown types of attacks, since the model has not seen them, they cannot be detected. Future work can be carried out by studying semi-supervised and unsupervised learning algorithms.

The materials we used are available at https://ieis.ac.cn/ldos for academic use.

7. Conclusions

Low-rate Denial of Service (LDoS) attacks have strong concealment. Although traditional signal analysis methods can detect LDoS attacks in the simulation environment, the detection results in real scenarios are unsatisfying. In this paper, an LDoS attack detection method based on hybrid deep neural networks is proposed using one-dimensional convolutional neural network and gated recurrent unit. The proposed method only needs time statistics of network traffic to detect LDoS attacks. In order to truly and effectively evaluate the proposed method, we captured real legitimate traffic from a website in the datacenter, and carried out a variety of real LDoS attacks on the mirror of the website in the laboratory environment to capture attack traffic. Evaluation on the real dataset showed that the proposed LDoS attack detection method could effectively detect LDoS attacks in fluctuating HTTP traffic with an average detection rate of 98.68%, which is significantly outperformed the state-of-the-art detection methods.

CRediT authorship contribution statement

Congyuan Xu: Conceptualization, Methodology, Software, Writing - original draft. **Jizhong Shen:** Supervision, Project administration, Funding acquisition, Writing - review & editing. **Xin Du:** Validation, Investigation, Resources, Writing - review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgment

This work was supported in part by the National Natural Science Foundation of China under Grant 61471314.

References

[1] Mantas G, Stakhanova N, Gonzalez H, Jazi HH, Ghorbani AA. Application-layer denial of service attacks: Taxonomy and survey. Int J Inf Comput Secur 2015;7(2–4):216–39.
[2] Loukas G, Öke G. Protection against denial of service attacks: A survey. Comput J 2010;53(7):1020–37.
[3] Zargar ST, Joshi J, Tipper D. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Commun Surv Tutor 2013;15(4):2046–69.
[4] Kuzmanovic A, Knightly EW. Low-rate TCP-targeted denial of service attacks and counter strategies. IEEE/ACM Trans Netw 2006;14(4):683–96.
[5] Wu ZJ, Zhang HT, Wang MH, Pei BS. MSABMS-Based approach of detecting LDoS attack. Comput Secur 2012;31(4):402–17.
[6] Wu Z, Wang M, Yan C, Yue M. Low-rate DoS attack flows filtering based on frequency spectral analysis. China Commun 2017;14(6):98–112.
[7] Simsek M. A new metric for flow-level filtering of low-rate DDoS attacks. Secur Commun Netw 2015;8(18):3815–25.
[8] Chen Z, Yeo CK, Lee BS, Lau CT. Power spectrum entropy based detection and mitigation of low-rate DoS attacks. Comput Netw 2018;136:80–94.
[9] Wu Z, Zhang L, Yue M. Low-rate DoS attacks detection based on network multifractal. IEEE Trans Dependable Secure Comput 2016;13(5):559–67.
[10] Agrawal N, Tapaswi S. Low rate cloud DDoS attack defense method based on power spectral density analysis. Inform Process Lett 2018;138:44–50.
[11] Yue M, Liu L, Wu Z, Wang M. Identifying LDoS attack traffic based on wavelet energy spectrum and combined neural network. Int J Commun Syst 2018;31(2):E3449.
[12] Xiang Y, Li K, Zhou W. Low-rate DDoS attacks detection and traceback by using new information metrics. IEEE Trans Inf Forensics Secur 2011;6(2):426–37.
[13] Pratomo BA, Burnap P, Theodorakopoulos G. Unsupervised approach for detecting low rate attacks on network traffic with autoencoder. In: 2018 international conference on cyber security and protection of digital services. IEEE; 2018, p. 1–8.
[14] Kuzmanovic A, Knightly EW. Low-rate TCP-targeted denial of service attacks: The shrew vs. the mice and elephants. In: Proceedings of the 2003 conference on applications, technologies, architectures, and protocols for computer communications. ACM; 2003, p. 75–86.
[15] Crovella ME, Bestavros A. Self-similarity in World Wide Web traffic: Evidence and possible causes. IEEE/ACM Trans Netw 1997;5(6):835–46.
[16] Daubechies I. The wavelet transform, time-frequency localization and signal analysis. IEEE Trans Inform Theory 1990;36(5):961–1005.
[17] Damon E, Dale J, Laron E, Mache J, Land N, Weiss R. Hands-on denial of service lab exercises using slowloris and rudy. In: Proceedings of the 2012 information security curriculum development conference. ACM; 2012, p. 21–9.
[18] Dar AH, Habib B, Khurshid F, Banday MT. Experimental analysis of DDoS attack and it's detection in eucalyptus private cloud platform. In: 2016 international conference on advances in computing, communications and informatics. IEEE; 2016, p. 1718–24.
[19] Houssni. Pwnloris: An improved slowloris DoS tool. 2021, https://github.com/h0ussni/pwnloris. [Accessed 8 April 2021].
[20] Dingledine R, Mathewson N, Syverson P. Tor: The second-generation onion router. Report, Naval Research Lab Washington DC; 2004.
[21] Bukac V, Matyas V. Analyzing traffic features of common standalone DoS attack tools. In: International conference on security, privacy, and applied cryptography engineering. Springer; 2015, p. 21–40.
[22] Leitchj. HTTP Bog: A slow HTTP denial-of-service tool. 2021, https://sourceforge.net/projects/httpbog/. [Accessed 8 April 2021].
[23] Ghafir I, Prenosil V, Svoboda J, Hammoudeh M. A survey on network security monitoring systems. In: 2016 ieee 4th international conference on future internet of things and cloud workshops. IEEE; 2016, p. 77–82.
[24] Allen L, Heriyanto T, Ali S. Kali Linux - Assuring security by penetration testing. Packt Publishing Ltd; 2014.
[25] Zeiler MD, Fergus R. Visualizing and understanding convolutional networks. In: European conference on computer vision. Springer; 2014, p. 818–33.
[26] Zhao B, Lu H, Chen S, Liu J, Wu D. Convolutional neural networks for time series classification. J Syst Eng Electron 2017;28(1):162–9.
[27] Xu C, Shen J, Du X, Zhang F. An intrusion detection system using a deep neural network with gated recurrent units. IEEE Access 2018;6:48697–707.
[28] Nair V, Hinton GE. Rectified linear units improve restricted Boltzmann machines. In: Proceedings of the 27th international conference on machine learning. 2010. p. 807–14.
[29] Srivastava N, Hinton G, Krizhevsky A, Sutskever I, Salakhutdinov R. Dropout: A simple way to prevent neural networks from overfitting. J Mach Learn Res 2014;15(1):1929–58.
[30] Kingma DP, Ba J. Adam: A method for stochastic optimization. In: International conference on learning representations. 2015. p. 1–15.
[31] Shore JE, Gray RM. Minimum cross-entropy pattern classification and cluster analysis. IEEE Trans Pattern Anal Mach Intell 1982;4(1):11–7.

[32] Fawcett T. An introduction to ROC analysis. Pattern Recognit Lett 2006;27(8):861–74.
[33] Selvaraju RR, Cogswell M, Das A, Vedantam R, Parikh D, Batra D. Grad-CAM: Visual explanations from deep networks via gradient-based localization. In: Proceedings of the IEEE international conference on computer vision. 2017. p. 618–26.
[34] Altman NS. An introduction to kernel and nearest-neighbor nonparametric regression. Amer Statist 1992;46(3):175–85.
[35] Schäfer P. Scalable time series classification. Data Min Knowl Discov 2016;30(5):1273–98.
[36] Shi X, Chen Z, Wang H, Yeung DY, Wong WK, Woo WC. Convolutional LSTM network: A machine learning approach for precipitation nowcasting. Adv Neural Inf Process Syst 2015;2015:802–10.
[37] Kamara AF, Chen E, Liu Q, Pan Z. Combining contextual neural networks for time series classification. Neurocomputing 2020;384:57–66.
[38] Iwana BK, Uchida S. Time series classification using local distance-based features in multi-modal fusion networks. Pattern Recognit 2020;97:107024.
[39] Chen W, Shi K. Multi-scale attention convolutional neural network for time series classification. Neural Netw 2021;136:126–40.
[40] Biggio B, Roli F. Wild patterns: Ten years after the rise of adversarial machine learning. Pattern Recognit 2018;84:317–31.