Computers & Security 115 (2022) 102604

Contents lists available at ScienceDirect

Computers & Security

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

A hybrid method of entropy and SSAE-SVM based DDoS detection and mitigation mechanism in SDN

Zhang Long, Wang Jinsong*

National Engineering Laboratory for Computer Virus Prevention and Control, Technology, Department of Computer Science and Engineering, Tianjin University of technology, No 319 Binshui West Road, Tianjin 300384, China,

ARTICLE INFO

Article history:
Received 17 August 2020
Revised 18 October 2021
Accepted 3 January 2022
Available online 10 January 2022

Keywords:
Software defined networks
Distributed denial of service
Machine learning
Entropy
Hybrid mechanism

ABSTRACT

Software-defined networking (SDN) is a new network architecture that offers considerable management convenience and efficiency relative to conventional networks. However, the centralized control employed in SDN incurs a high risk of single point failure that is susceptible to distributed denial of service (DDoS) attacks. The present work addresses this issue by proposing a hybrid approach for detecting DDoS attacks using an initial detection module based on information entropy to quickly identify anomalous traffic and a second detection module based on machine learning with a stacked sparse autoencoder (SSAE)–support vector machine (SVM) architecture to confirm the suspected anomalous traffic. If DDoS traffic is detected, a defense module is implemented to restore normal network communication in a timely manner via an issued flow table. The effectiveness and efficiency of the proposed approach for DDoS detection is experimentally evaluated using both real-time and benchmark datasets in comparison with state-of-the-art methods. The results demonstrate that the proposed approach provides superior detection performance and identifies greater than 98% of existing DDoS traffic with greatly reduced training time and computational burden.

1. Introduction

Software-defined networking (SDN) is a novel network architecture that separates the control and transmission functions in the intermediate facilities of conventional networks . The controller centralizes the control privileges of SDN and dynamically controls the transmission of data packets based on software programming . Accordingly, SDN differs dramatically from conventional networks. However, while the SDN architecture provides considerable convenience and efficiency for managing network resources, this centralized management also leads to a series of security issues . One of the most challenging issues stems from the fact that a failure at any single point can lead to paralysis of the entire network . Consequently, the SDN architecture is uniquely susceptible to distributed denial of service (DDoS) attacks, which are relatively easy to launch and difficult to defend. They are not only highly harmful to the target network, but they also affect controllers and switches and can result in complete network collapse in severe cases. Attackers send a large number of specially processed packets to switches, which are then uploaded to the controller and force the controller to consume excessive resources for processing these data packets, resulting in a high response delay for legitimate network requests. Simultaneously, the large number of useless flow entries issued by the controller occupy the flow table spaces of switches, which further interferes with normal network communication because legitimate flow entries cannot be stored.

The seriousness of DDoS attacks and the unique susceptibility of SDN architectures to this threat have generated considerable interest in recent years, and a number of detection and defense schemes suited to SDN have been proposed . Generally, DDoS detection technologies involve the detection of anomalous network communication traffic based on either statistical methods or machine learning methods . One of the primary characteristics of network communication that is typically employed in both statistical and machine learning methods involves the concept of information entropy, which is a measure of the degree of uncertainty in the value of a random variable. The main advantage of statistical methods is that they provide good real-time performance. In contrast, machine learning methods obtain a much better detection accuracy, but at a much greater computational expense along with the complications arising from the need to obtain manually extracted network characteristics for training purposes. Both types of existing DDoS detection methods suffer from additional disadvantages as well. With the in-depth study of deep learning, a growing number of researchers have used deep learning methods to solve such problems. Some common deep learning methods such as Stacked

* Corresponding author.
E-mail address: jswang@tjut.edu.cn (W. Jinsong).

https://doi.org/10.1016/j.cose.2022.102604
0167-4048/© 2022 Elsevier Ltd. All rights reserved.

Autoencoders (SAE) and Deep Neural Networks (DNN) have been used. These methods greatly improve the accuracy of detection, but the improvement in detection time is often overlooked.

To solve the aforementioned problems, we proposed a novel solution to effectively detect DDoS traffic in SDN. Our proposed solution uses asynchronous messages and Openflow entries. In this work, we uses the stacked sparse autoencoder (SSAE) model based on deep learning theory for unsupervised dimensionality reduction of intrusion detection samples. Compared with the existing methods, the SSAE can not only compress the feature dimension of the original data effectively but can also speed up the classification process of the classifier, and the feature extraction ability of the SSAE is significantly better than that of known methods. Not only that, we also use the method based on information entropy, using this method we can realize the preliminary screening of a large number of data. The present work proposed a hybrid DDoS detection approach that obtains a high detection rate using an initial detection module based on information entropy to quickly identify anomalous traffic, a second detection module based on machine learning with a stacked sparse autoencoder (SSAE)–support vector machine (SVM) (i.e., SSAE-SVM) architecture to confirm the suspected anomalous traffic, and a defense module. The SVM is employed to classify the deep sparse features extracted by the SSAE, and the proposed overall approach is denoted herein as the HESS method. The proposed mechanism also considers the impact of DDoS attacks on the SDN infrastructure and controller. The HESS method is utilized as a plug-in of the SDN controller, and it monitors the security status of the SDN architecture in real time. The major contributions of this paper include the following.

(1) Data preparation and preprocessing is developed to streamline the DDoS detection process. The data packets sent to the controller from switches are employed as attack detection data sources that are processed according to the characteristics of SDN control and transmission separation combined with the message reception mechanism of the OpenFlow protocol. Four features, the source IP, source port, destination IP, and destination port, are extracted from the packets. The data packet collection process adapts to the message reception mechanism of the controller and reduces the controller load compared with the periodic collection of data flow entries by the controller.
(2) The features employed as the input of the SSAE-SVM module are directly extracted OpenFlow protocol fields and computed features. Compared to other machine learning methods (e.g., k-nearest neighbors (K-NN) and SVM), the SSAE-SVM does not require the calculation of a large number of input features, and it provides a higher detection rate with less training time than other deep learning methods (e.g., a deep neural network (DNN) and SAE). The HESS method not only addresses the low detection precision of methods based on information entropy but also alleviates the greatly increased detection time and computing resources required by existing methods with increased network traffic.
(3) A defense module was developed to quickly locate the access ports of switches subject to DDoS attacks, which downloads defense measures to protect the SDN-based network and restore normal communication in a timely manner.
(4) An experimental platform is constructed to launch DDoS attacks and to verify the detection and defense effects of the corresponding modules within a realistic simulation environment. Simultaneously, these effects are verified using classical intrusion detection datasets.

The remainder of this paper is structured as follows. Our motivation for the developed HESS method is outlined by providing a general discussion of the advantages and disadvantages of some existing DDoS attack detection approaches in Section 2. Section 3 briefly explains the main concepts employed in the HESS method. The explicit scheme and experimental evaluations are discussed in Section 4. Finally, conclusions and future research directions are presented in Section 5.

2. Related work

2.1. Attack detection based on statistics

Jisa et al. proposed a predictive DDoS attack method based on a network traffic threshold. The method constructed the traffic characteristics by extracting NetFlow data and assigned the traffic threshold through a detection function. However, this method required extensive processing of the collected data traffic in advance, and the detection performance was closely related to the subjective experience of the researchers.

Ra et al. proposed a fast entropy approach for detecting DDoS attacks via the rapid calculation of information entropy based on the statistics of network traffic. This method successfully increased the detection speed relative to previously proposed methods but failed to significantly increase the detection accuracy. Moreover, the method provided a particularly high false alarm rate.

These issues were addressed by the work of Ahn et al., who proposed a large-scale DDoS detection approach based on flow entropy and packet sampling. However, the threshold was set according to the relevant experience of the researchers, which increased the influence of human factors on the detection performance.

Kalkan et al. proposed a DDoS attack detection approach appropriate to SDN that employed different combinations of extracted traffic characteristics, generated feature matrices, and calculated the combinatorial entropy to detect abnormal network traffic. This method provided a greatly enhanced detection range and detection accuracy relative to those of previously proposed entropy-based detection algorithms. However, the approach required a long period of preparation and statistics requiring complex calculations, and it provided relatively poor scalability and low real-time detection performance.

Kumar et al. adopted information entropy with an adaptable threshold to develop an early detection technique for TCP SYN flooding attacks in SDN-based networks. The randomness of the data packets received by the SDN controller was measured based on the entropy of the traffic characteristics in a time-based data packet window sequence. However, this approach only considered attack patterns with a single target.

Other previously proposed schemes could also detect DDoS attacks in SDN-based networks at an early stage based on the entropy of data packet flows at the destination IP addresses. However, as discussed, the use of only a single feature allowed attackers to bypass detection schemes easily.

2.2. Attack detection based on machine learning

Zhu et al. proposed a highly accurate DDoS traffic detection algorithm based on the K-NN algorithm. Hu et al. proposed a framework for accountable decision making (FADM) to detect and mitigate DDoS attacks in SDN. Network traffic information was collected through the SDN controller and sFlow agents. An entropy-based method was then used to measure network features, and an SVM classifier was applied to identify network packet flow anomalies. Compared with previously proposed machine learning methods, the approach provided a decreased response time and high detection accuracy, but higher recovery delays occurred for SYN flood attacks than for other flooding attacks.

Cui et al. proposed a DDoS attack detection and defense mechanism based on cognitive-inspired computing address entropy. The approach employed a switch flow table data to calculate the entropy of data packet flows at source and destination IP addresses, and an SVM was used for training to obtain specific DDoS attack detection modes.

Ye et al. also proposed a DDoS detection approach using an SVM classifier. The proposed detection framework consisted of flow status collection, feature extraction, and classification of the extracted feature values. In addition, the authors implemented a feature extraction module to extract the features related to DDoS attacks to train the classifier and demonstrated the performance of the proposed approach using flood-based attack traffic (i.e., TCP, UDP, and ICMP). The approach was shown to provide an average detection accuracy of 95.24% and a lower false alarm rate than that provided by previously proposed methods. However, it must be noted that all of the above methods suffered from the problem of unknown actual detection effects as well as the discussed requirement for extracting a large number of features manually for training.

Li et al. proposed a network traffic anomaly detection method based on a DNN, which provided superior detection accuracy compared with previously proposed machine learning methods. A typical DNN consists of three main components: an input layer, hidden layers, and an output layer, where each layer contains a different number of neurons. A series of parameters x_i are transmitted from the input layer to the neurons of the first hidden layer with a corresponding weight w_i. The neurons in the first hidden layer correct the input signal according to a bias parameter b and then output the signal obtained from an activation function f to the next hidden layer. The hidden layers generate a multi-hidden-layer neuron network by iteration, and the results are finally output through the output layer. The gradient descent algorithm based on reverse error propagation is a common objective function employed for network optimization. However, the improper selection of initial parameters and the use of large-scale datasets makes DNNs readily susceptible to the curse of dimensionality and gradient diffusion, which can result in local minima, overfitting, underfitting, and other problems. As such, this method alone is not suitable for large input dimensions and a large volume of data, and more reasonable algorithms are required. In addition, the detection efficiency of the method is low because the detection algorithm consumes considerable computational resources, and the computation time increases dramatically with increasing network traffic.

These issues were addressed by Quamar et al., who presented a multi-vector DDoS detection system based on deep learning using an SAE for SDN applications. The system was deployed on the SDN controller. The SAE model demonstrated better performance than the DNN model in terms of detection accuracy. However, considerable scope for improvement remained in terms of the detection accuracy and the excessively long processing time required under heavy network traffic conditions.

3. DDoS attack detection and defense

3.1. Data preparation and preprocessing

The SDN controller is the central network component and can easily access the global network information required to evaluate the network status in real time using the OpenFlow protocol, which is the communication interface standard operative between the SDN controller and network switches. The OpenFlow protocol is the first standardized communication protocol defined between the control layer and the forwarding layer in the SDN architecture. The switch uses the concept of flow to identify network traffic and log its information through a counter. Flow is a group of consecutive messages with the same properties in the same network. The controller configuration, switch management, and data packet flow are illustrated in Fig. 1. OpenFlow messages mainly include three types: symmetric, controller-to-switch, and asynchronous. Symmetric messages are used to configure SDN information, and these messages are prominent during start-up and SDN operation . Controller-to-switch messages are initiated actively by the controller as packet-out messages and are typically employed for the flow table management of switches. Asynchronous messages are initiated by switches and are used as packet-in messages to notify the controller regarding changes in the states of switches or network events.

Some researchers have developed DDoS attack detection and defense systems based on the flow entry information of OpenFlow switches that are obtained periodically using controller-to-switch messages. The disadvantage of this method is that the periodic task of active inquiry increases the computational overhead of the controller, and this overhead increases substantially with increasing network size. In contrast, asynchronous messages actively sent by switches typically incur greatly reduced computational overhead for the controller because the controller is much more of a passive recipient in this case. For example, a switch sends a request processing message to the controller when receiving a new type of data packet at one its ports. The controller then processes this packet-in message and transfers the installation rules to the switch, after which all subsequent processing for that type of data packet is conducted by the switch with no further message exchange. Accordingly, the number of packet-in messages received by a controller is relatively stable in an SDN setting when the network scale remains constant. The packet-in messages generated by switches include some useful information, such as the number of switches, the port number associated with a matching failure, and the content of the matching failure data packet. These data can be used directly in subsequent DDoS detection and defense activities. Therefore, the present work collects data via asynchronous messages and directly adapts to the message reception function of the controller. However, the original data contain a considerable volume of useless and redundant information, which must be omitted to improve the quality of the data. Therefore, the present work applies data preprocessing, which includes eliminating useless data by filtering both non-IPv4-type data packets and consecutive identical packet-in messages and recovering missing information.

3.2. Attack detection module based on information entropy

The information entropy associated with a random variable increases as the dispersion of its distribution increases . Accordingly, the entropy of a random variable x with possible outcomes x_i ($i = 1, 2, ...N$), each of which is composed of a number of samples n_i, can be calculated using the Shannon formula:

$$H(x) = -\sum_{i-1}^{N}(\frac{n_i}{S})\log\left(\frac{n_i}{S}\right), \tag{1}$$

where $S = \sum_{i}^{N} n_i$ is the total number of samples in x. Accordingly, n_i/S represents the probability of outcome x_i. In addition, the base of the logarithm is generally 2. As known from the Shannon formula, the entropy of a sample varies in the interval $(0, \log S)$, and is 0; when the distribution of the sample is the most dispersed, the entropy is the largest, $\log S$ and concentrated, such as when $n_1 = n_2 = \cdots = n_i$, and is $\log S$ when the distribution of the sample is the most dispersed, such as when all sample values are different.

Due to the self-similarity of network traffic, the packet density is related only to the sample quantity. For example, consider two sample sequences X and Y with the same number of samples, where X is a sample sequence collected within 30 s, and Y is a sample sequence collected within 45 s. X and Y can be considered

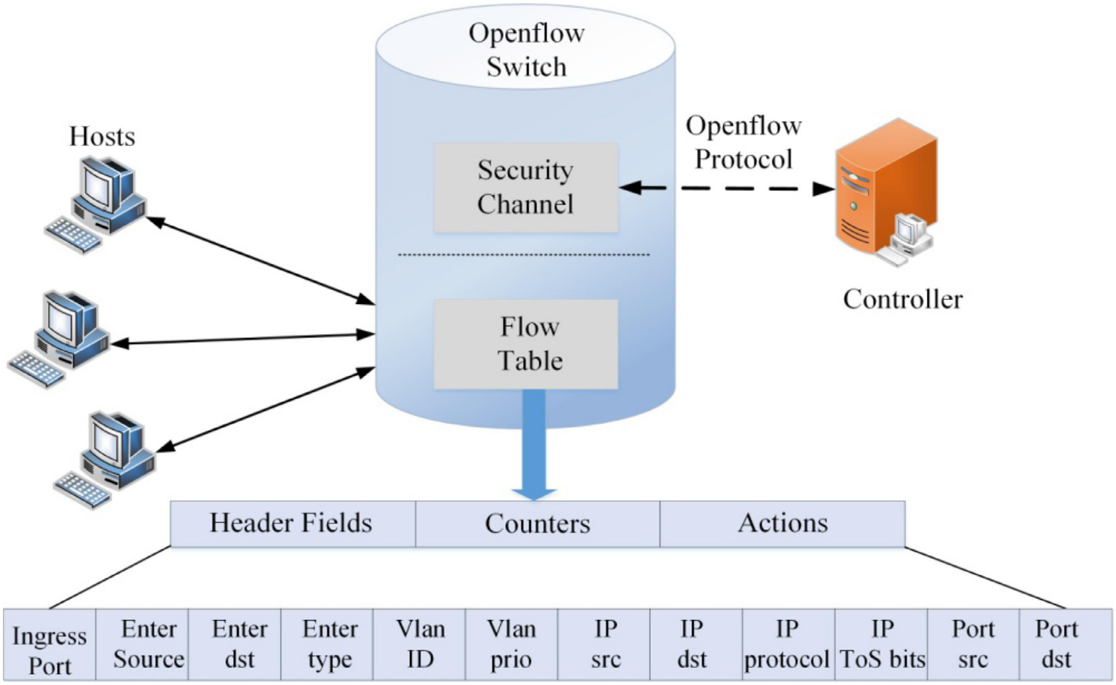


Fig. 1. OpenFlow standard process and flow entry.

Table 1
DARPAR data statistics.

Traffic condition		
Normal	227	256
DDoS	34,168	9

Algorithm 1 Initial detection algorithm.

```
window size $W$;
collect $W$ asymmetric data packets in sequence
$H$={ $H_{srcip}$, $H_{dstip}$, $H_{srcp}$, $H_{dstp}$}
if any value of $H$ exceeds the threshold $T$ then
report DDoS attack to the confirmation module
end if
```

to have the same distribution sparsity degree when the entropy of the two sample sequences is the same .

The usefulness of information entropy for detecting DDoS attacks can be illustrated by considering the number of source IP addresses and the number of destination IP addresses encountered by a network during normal operating conditions and under a DDoS attack. In this regard, Table 1 presents the number of source and destination IP addresses within 70,000 data packets collected by the MIT Lincoln Laboratory during normal network traffic in the first week of DARPA 1999 and during the LSDDoS2.0.2 attack of DARPA 2000. We note that the numbers of network source and destination IP addresses differ little under normal circumstances, while the number of source IP addresses is extremely large due to forgery, and the number of destination IP addresses is significantly lower under DDoS attack conditions. Accordingly, we can expect the information entropy associated with the number of source IP addresses to increase greatly under attack because the distribution of source IP addresses is much more dispersed than that obtained under normal traffic. In addition, we expect the information entropy associated with the number of destination IP addresses to decrease under attack because the distribution of destination IP addresses is more concentrated. These characteristic variables are often selected to detect DDoS attacks in conventional networks.

Based on the above discussion, we adopt the number of source IP addresses, destination IP addresses, source port numbers, and destination port numbers as characteristics in the initial attack detection module based on the information entropy. In this module, a fixed number W of data packets associated with packet-in messages are defined as a packet bin, and the characteristic values in the packet bin are employed as sample data to calculate the entropy. The selection of W controls the extent to which sample characteristics change over time. An overly large value of W will result in overly consistent values of entropy, which will diminish the detection accuracy of the module. In principle, the choice of W is related to the communication load of the experimental network. A value of $W = 100$ was found to be a good compromise based on the number of hosts and level of network traffic employed in the experimental dataset that was adopted in the present work. Accordingly, the entropy values associated with the number of source IP addresses (H_{srcip}), destination IP addresses (H_{dstip}), source port numbers (H_{srcp}), and destination port numbers (H_{dstp}) were calculated for the first consecutive W packets using formula (1), and then they were calculated consecutively for subsequent adjacent W data packets. In addition, we set the threshold T experimentally, as discussed later in Section 4.2. Here, if any of the four entropy values are greater than T, it indicates the possibility of a DDoS attack. The computing process of the module is given in Algorithm 1 and can be described as follows.

1 Collect W asymmetric data packets received by the controller and initialize the entropy values H_{srcip}, H_{dstip}, H_{srcp}, and H_{dstp}.
2 Calculate H_{srcip}, H_{dstip}, H_{srcp}, and H_{dstp} for each i in the sliding window, and determine if H_{srcip}, H_{dstip}, H_{srcp}, or $H_{dstp} > T$.
3 If so, go to the following detection module based on machine learning for confirmation; otherwise, repeat steps 1–3.

The fact that the final decision regarding the occurrence of a DDoS attack is not based on this attack detection module allows us to avoid the complication of setting confidence intervals, as is required by other entropy-based methods. In addition, this fact fur-

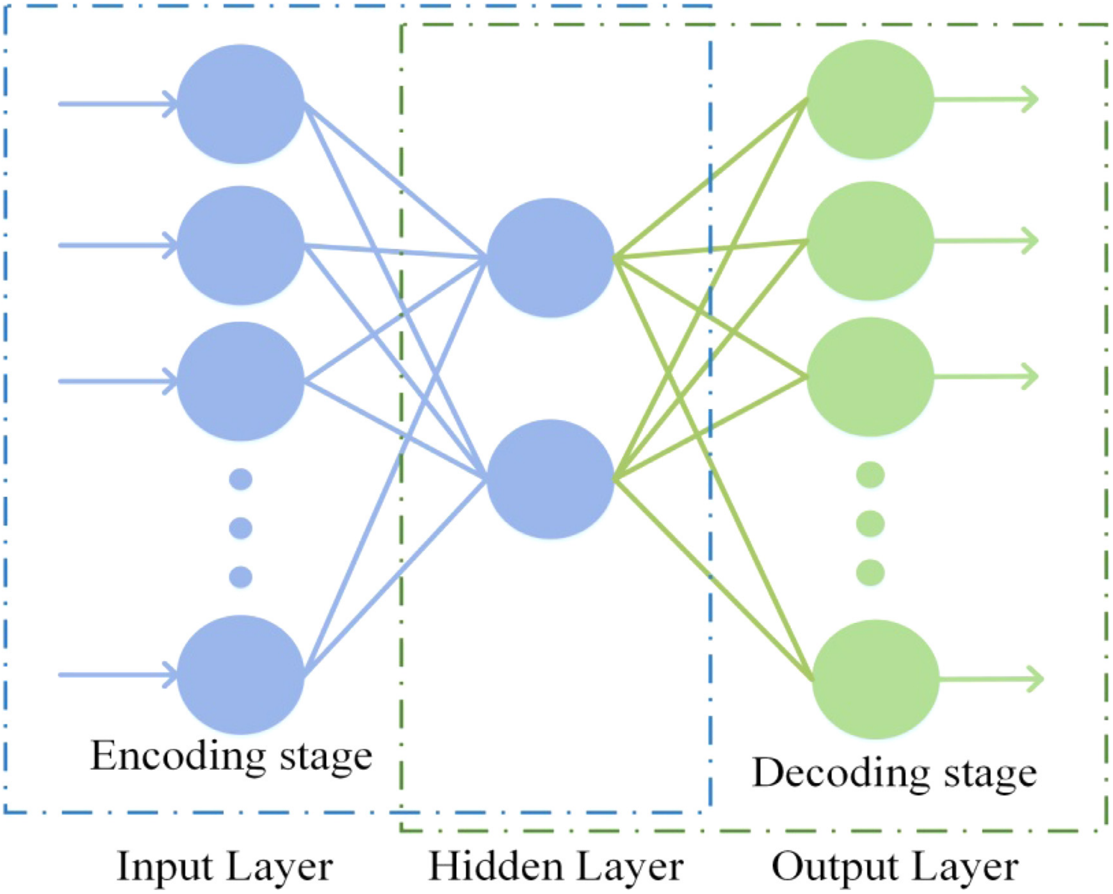


Fig. 2. Working process of autoencoder (AE) network.

ther simplifies the setting of the threshold value, such that the threshold can be set to a smaller value to increase the detection sensitivity, albeit with an increased false alarm rate as well.

3.3. Attack detection module based on ssae-svm

3.3.1. Stacked sparse auto encoder (SSAE) network

An auto encoder (AE) network is a deterministic and unsupervised neural network that uses the output of a neural network to reconstruct the input data, and it adopts the backpropagation gradient descent algorithm to correct the model parameters. The number of nodes in the hidden layer of the AE network is generally less than those of the output and input layers. This facilitates the representation of the original data with reduced dimensionality. The working process of an AE network is illustrated in Fig. 2. Here, the working principles of an AE network are to obtain a network structure and to fit the original input as much as possible through the output of the network structure. The input of the network structure is usually a set of samples without labels. The input layer data are then moved to the hidden layer through an activation operation in what is denoted as the encoding process. The output value of the hidden layer enters the output layer through another activation operation in what is denoted as the decoding process, which reconstructs the original input sample set in the output layer. Finally, a reconstruction process is conducted to minimize the error in the reconstruction at the output layer. The encoding, decoding, and reconstruction processes are described as follows.

1 The encoding process yields the following hidden layer output:

$$h = g_1(x) = \rho(W_1 x + b_1), \tag{2}$$

where x is the input data, and W_1, b_1, and $\rho(\cdot)$ are the weight matrix, bias matrix, and activation function of the input and hidden layers, respectively.

2 The decoding process yields the following output layer result:

$$\hat{x} = g_2(h) = \rho(W_2 h + b_2), \tag{3}$$

where W_2, b_2, and $\rho(\cdot)$ are the weight matrix, bias matrix, and activation function of the hidden and output layers, respectively.

3 The reconstruction process involves minimizing the following reconstruction error:

$$J_E(W, b) = \frac{1}{m}\sum_{r=1}^{m}\frac{1}{2}\parallel \quad \hat{x}^r - x^r \quad \parallel \tag{4}$$

where m is number of neuron nodes in the input layer, x is the input data, and $\hat{x}$ is the output representation after decompression.

A sparse AE (SAE) network can acquire deeper features than the original unlabeled sample set . Similar to an AE network, an SAE seeks to make the output fit the input characteristics as much as possible. However, an SAE network adds a sparsity restriction to the hidden layer to avoid simply mapping the input to the output. An SAE network is modeled according to the working process of the human brain, which includes many neurons. However, most neurons are inactive during cognition, and only a relatively small number of neurons are activated. Accordingly, the sparsity restriction severely limits the number of active neurons in the output of each layer by setting the output of the majority of neurons to 0, while active nodes have an output of 1. This can be accomplished by adopting the sigmoid function for $\rho(\cdot)$ above. Of course, the tanh function can also be used for this purpose. When the prediction result of neurons is −1, the state of the neurons is inhibited.

To obtain a sparse representation, we first note that the output of the neurons in the hidden layer is a dimensionless expression of input data x. Accordingly, we seek to limit the output of neurons in the hidden layer to 0 as much as possible. To this end, we define an average degree of activation $\hat{\rho}_j$ for node j in the hidden layer as follows:

$$\hat{\rho}_j = \frac{1}{m}\sum_{i=1}^{m}\left[a_j^{(2)}\left(x^{(i)}\right)\right], \tag{5}$$

where m represents the number of data elements and $a_j^{(2)}(x)$ indicates the output activation value of node j in the hidden layer for input x. Accordingly, a sparse representation of the nodes in the hidden layer can be obtained by ensuring that the values of $\widehat{\rho_j}$ in the hidden layer are as close to zero as possible. This is accomplished by first defining $\widehat{\rho_j} = \rho$, where ρ is a real-valued sparseness parameter that is assigned a value that approaches zero. A penalty term is then introduced into the objective function that penalizes the scenarios with significant differences between the values of $\widehat{\rho_j} = \rho$ and ρ, which achieves the sparsity limit while simultaneously optimizing the network. The present work adopts the relative entropy, which is denoted as Kullback–Leible divergence (KL divergence), to regularize the network and assure sparsity as follows:

$$KL\left(\rho \parallel \hat{\rho}_j\right) = \rho\log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}. \tag{6}$$

The penalty factor is formulated as follows:

$$\sum_{j=1}^{S}\rho\log\frac{\rho}{\hat{\rho}_j} + (1-\rho)\log\frac{1-\rho}{1-\hat{\rho}_j}, \tag{7}$$

where s is the number of hidden layer neurons. In addition, the penalty factor definition in formula (7) can also be expressed as $\sum_{j=1}^{S} KL(\rho \parallel \hat{\rho}_j)$ based on the KL divergence definition in formula (6). Finally, the loss function of the network is given as

$$J_{sparse}(W,b) = J_E(W,b) + \beta\sum_{j=1}^{S}KL\left(\rho \parallel \hat{\rho}_j\right), \tag{8}$$

where β is a parameter that adjusts the relative significance of formula (6) relative to that of formula (4) in the network optimization process. In addition, overfitting is avoided by adding a weight penalty term to the objective function as follows:

$$J_{SAE}(W,b) = J_{sparse}(W,b) + \frac{\omega}{2}\sum_{l=1}^{n_l-1}\sum_{i=1}^{s_l}\sum_{j=l}^{s_l+1}\left(w_{ij}^{(l)}\right)^2, \tag{9}$$

where ω is a regularization parameter, n_l is the number of layers in the network, l represents the current layer, s_l and s_l+1 represent the number of neurons in adjacent connected layers, and $w_ij^{(l)}$ are the elements in the l-th weight matrix. The network matrices W and b are optimized according to the following objective functions:

$$W_{ij}(l) = W_{ij}(l) - \eta\frac{\partial}{\partial w_{ij}(l)}J_{SAE}(W,b), \tag{10}$$

$$b_i(l) = b_i(l) - \eta\frac{\partial}{\partial w_{ij}(l)}J_{SAE}(W,b), \tag{11}$$

where η is the learning rate, and the optimal W and b can be obtained by backpropagation (BP) using the stochastic gradient descent (SGD) method.

The SSAE network is composed of a stack of many SAE networks. The SSAE network adopts layer-wise pre-training and overall fine-tuning training according to the training process illustrated in Fig. 3. The first SAE contains layers x, h_1, and $\hat{x}$, which are optimized according to formula (9), and then w_1 and b_1 are obtained through optimization based on formulas (10) and (11). The second SAE contains layers h_1, h_2, and $\widehat{h_1}$, which are optimized in a manner similar to that employed for the layers of the first SAE, and w_2 and b_2 are again obtained through optimization. All the parameters in the SSAE network can be obtained by repeating the above optimization steps, where the parameters of the upper layer of the network remain unchanged when optimizing subsequent network layers. Here, the number of neurons gradually decreases during the training process, and deep sparse features are finally obtained. Once the parameters of all the layers have been established in this manner, the entire network must be finely tuned by adding label data to the final layer and employing the BP algorithm to adjust all the network parameters. Assigning weights to neural networks through pre-training has been demonstrated to facilitate network convergence better than assigning weights randomly .

3.3.2. Support vector machine

The SVM is a data processing method based on statistical learning theory. The basic mechanism employs the inner product kernel function to construct a high-dimensional feature space, and then it finds an optimal classification hyperplane in the feature space that maximizes the distance between hyperplanes while ensuring good classification accuracy. For N nonlinear separable samples, the problem of finding the optimal classification hyperplane can be transformed into the following optimization problem:

$$min\frac{1}{2}\parallel \quad \omega \quad \parallel^2 + C\sum_{i=1}^{N}\xi_i, \tag{12}$$

where ω is the normal vector to the hyperplane, C is a penalty factor, and ξ_i is the i-th relaxation factor ($\xi_i \geq 0$). The constraint condition is

$$y_i\left(\omega^T\phi(x_i)\right) \geq 1 - \xi_i, \tag{13}$$

where $\phi(\cdot)$ is a nonlinear transformation function of x_i. The kernel functions most commonly used in the SVM are the linear, polynomial, radial basis, and sigmoid kernel functions. The radial basis kernel function was selected in the present study because its transformation matrix is characterized positively, it has a wide convergence domain, and it contains only a single parameter, which makes the calculation process relatively simple.

3.3.3. SSAE input features

The selection of features can greatly affect the success of machine learning models. However, the use of numerous calculated features increases the complexity of the model and affects the detection speed of the model. The 19 input features employed in the present work for the SSAE are listed in Table 2. These input characteristics are derived from fields in the flow entries of the OpenFlow protocol. In addition, we extracted all but two of the input features directly from the OpenFlow flow table, while the final two features (i.e., *n_packets_ave* and *n_bytes_ave*) were easily calculated.

3.3.4. Process flow of attack detection module based on ssae-svm

The working process of the DDoS detection module based on SSAE-SVM is illustrated in Fig. 4, and the algorithm itself is summarized in Algorithm 2. In this module, the communication network traffic data are first normalized and the input features are extracted. Training data are then employed to pre-train and fine-tune the SSAE network structure, and the output of the final hidden layer of the SSAE network is input into the SVM for classification.

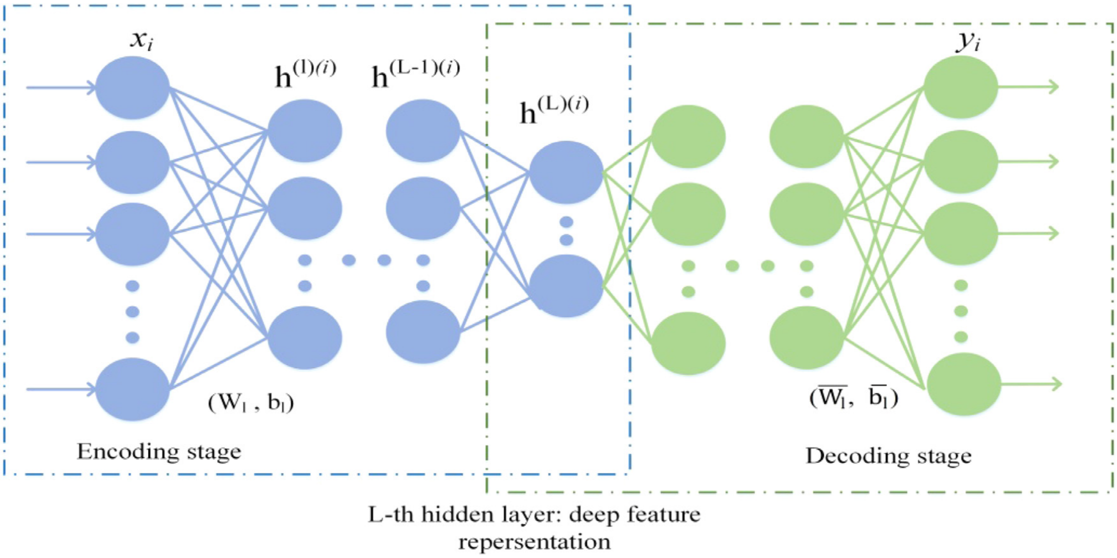


Fig. 3. Working process of stacked sparse autoencoder (SSAE) network.

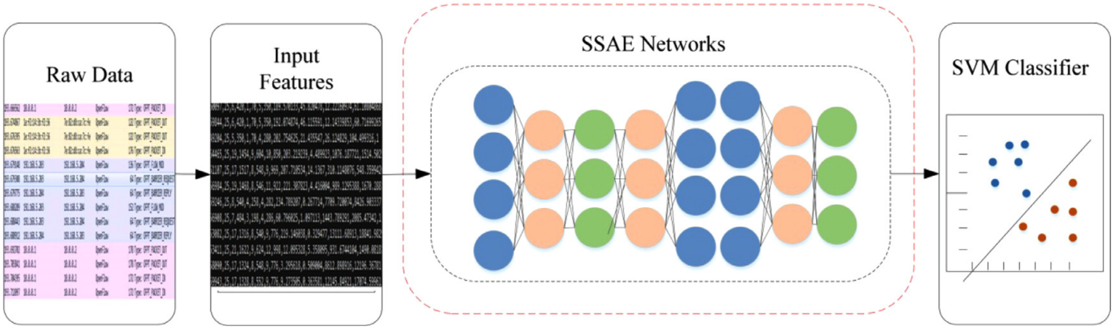


Fig. 4. Process flow of the DDoS attack detection module based on SSAE-SVM.

Table 2
Descriptions of input features for stacked sparse autoencoder–support vector machine (SSAE-SVM).

Feature Name	Description
cockie	Cockie
duration	Duration of flow (s)
table_id	ID
n_packets	Number of packets
n_bytes	Number of bytes
idle_timeout	Time-out period
priority	Priority of flow
protocol	IP protocol
mac_src	Source MAC address
mac_dst	Destination MAC address
ip_src	Source IP address
ip_dst	Destination IP address
tcp_src	TCP source port number
tcp_dst	TCP destination port number
udp_src	UDP source port number
udp_dst	UDP destination port number
actions	Action of switch
n_packets_ave	*n_packets*/*duration* (pps)
n_bytes_ave	*n_bytes*/*duration* (bps)

Algorithm 2 DDoS attack detection algorithm based on SSAE-SVM.

```
Input: Training dataset data[][]; sample sizeM; label data label[][]; number of network layers N; number of network pre-training iterations PNum; number of network fine-tuning iterations FNum. Output: Out data[][]
Output: Detection results
for $i = 1$ to N layers do
    reconstruction error: sda[i]
    model optimized: train opt
    for echo=1 to PNum do
        bNum=int(M/batch size)
        for $i = 1$ to bNum do
            train opt
        end for
    end for
end for
obtain the classification loss of the entire network: finetune cost
for echo=1 to Fnum do
    bNum=int(M/batch size)
    for $j = 1$ to bnum do
        finetune the network
    end for
end for
if SVM(Out data[][] == 1) then
    confirm and report DDoS attack
end if
```

4. Attack defense module

The algorithm employed in the attack defense module is given in Algorithm 3. First, the confirmation detection module identifies the port through which a DDoS attack is launched, and the defense module downloads filtering rules to the identified port to facili-

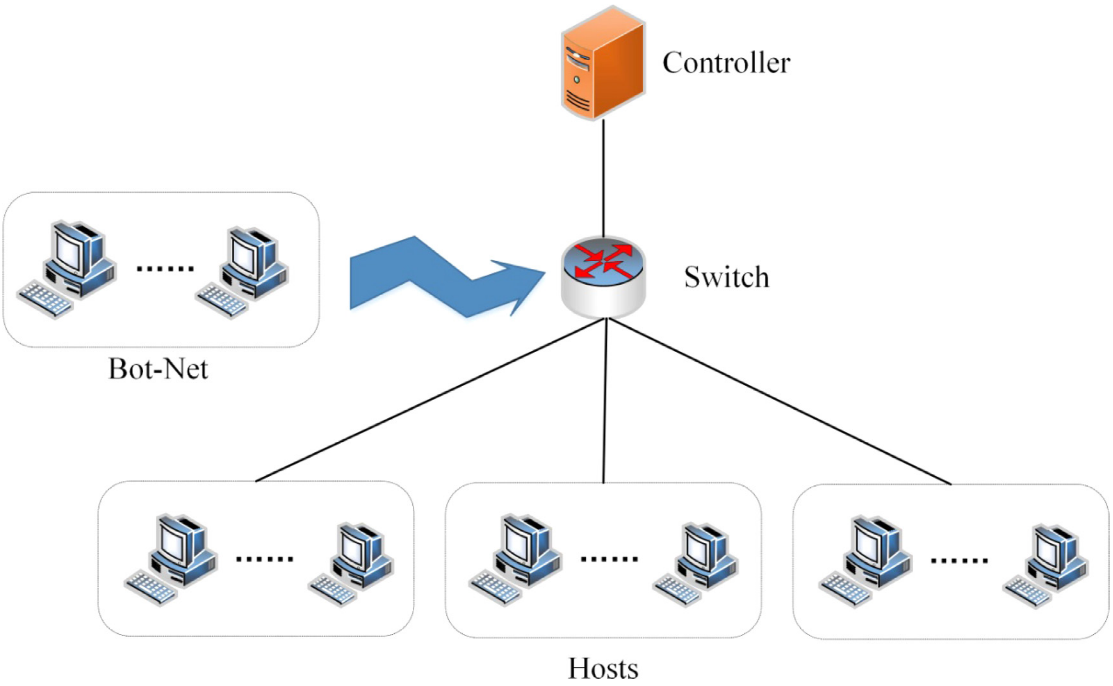


Fig. 5. Experimental topology.

Algorithm 3 Defense algorithm.

```
input: the port in identified by the confirmation detection module as being associated with the DDoS attack
create flow table entry a: port_in, priority, hard_timeout, ide_timeout, action
if flow table entry a does not exist in buffer queue then
    add flow table entry a to buffer queue
end if
deliver entry a to OpenFlow switches
if port = port_in then
    action=drop
end if
```

tate the abandonment of data packets associated with the attack, thereby protecting the SDN controller and network infrastructure.

5. Experiments and evaluation

5.1. Experimental setup

The simulation experiments employed the OpenFlow controller Floodlight as the SDN controller, and the SDN-based network environment was simulated using Mininet . The Floodlight controller was located on a server with a 32-core processor and 64 GB of memory. The network topology illustrated in Fig. 5 was simulated by Mininet and consisted of a floodlight controller (C), an OpenFlow switch (S), and 20 hosts. The victim host is located in the network of switch S, and the attacker launched a DDoS attack from another network. On-line traffic data in the simulated SDN-based network were collected under both normal traffic and DDoS attack traffic conditions. For the normal traffic conditions, the traffic associated with TCP, UDP, and ICMP protocol types were mixed in a proportion of 85:10:5. This proportion is based on that obtained over a seven year period for a communication link between the United States and Japan . For the attack traffic condition, the standard DDoS attack tool hping3 was employed to launch the hybrid synflood, udpflood, and icmpflood attacks in the experimental network with a mixture ratio of 1:1:1 and various levels of DDoS attack traffic ranging from 25% to 90%.

Table 3
Descriptions of datasets employed in the simulation experiments

Dataset	Training Dataset		Testing Dataset	
	Normal	DDoS	Normal	DDoS
Dataset 1	-	-	20,000	40,000
Dataset 2	70,000	70,000	30,000	30,000
DARPAR dataset	14,000	14,000	6,000	6,000

The three datasets used in the experiments are listed in Table 3, which included the DARPAR dataset and Datasets 1 and 2 collected from the controller of the simulated network in real time. All three datasets included data packets collected under both normal traffic and attack traffic conditions. Dataset 1 was employed to test the initial attack detection module based on information entropy, and it included the number of source IP addresses, number of destination IP addresses, source port numbers, and destination port numbers obtained from 60,000 data packets collected under the two conditions with the proportions listed in Table 3. Dataset 2 was employed for training and testing the confirmation attack detection module based on the SSAE-SVM, and it included the input features listed in Table 2 obtained from 100,000 data packets collected under normal conditions and 100,000 data packets collected under different degrees of DDoS attack. The proportion of training and testing samples are listed in the table. In addition to Dataset 2, the experiments also employed 20,000 data packets collected from the first Monday of DARPA 1999 under normal conditions and 20,000 data packets collected from phase 5 of DARPAR 2000 under attack conditions as comparison datasets.

The experiments employed three performance evaluation indicators: the detection rate (DR), accuracy (ACC), and false alarm rate (FAR). The DR is defined as follows:

$$DR = \frac{TP}{TP + TF}, \tag{14}$$

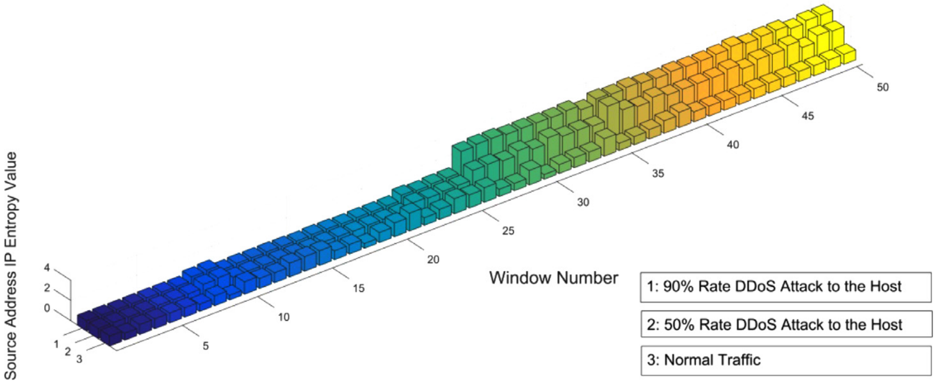


Fig. 6. Entropy value of source IP addresses (H_{srcip}).

where TP is the proportion of true positives, which represents the relative number of correctly identified attacks from the attack detection results, and FN is the proportion of false negatives, which represents the relative number of actual attacks that were not identified. Accordingly, the DR represents the probability of correctly identifying actual attacks, which increases as DR approaches 1.

ACC is defined as follows:

$$ACC = \frac{TP + TN}{TP + TN + FP + FN}, \quad (15)$$

where TN is the proportion of true negatives, which represents the relative number of correctly identified non-attacks (i.e., normal conditions) from the attack detection results, and FP is the proportion of false positives, which represents the relative number of identified attacks that were not actual attacks. Accordingly, the ACC represents the probability of correctly identifying actual attacks while not falsely identifying attacks as normal conditions and normal conditions as attacks, which increases as ACC approaches 1.

FAR is defined as follows:

$$FAR = \frac{FP}{FP + TN}. \quad (16)$$

Accordingly, the FAR represents the probability of incorrectly identifying attacks, which increases as FAR approaches 1, and should be close to 0.

5.2. Results of detection module based on information entropy

The effectiveness of the initial detection module based on the information entropy was evaluated by calculating the values of H_{srcip}, H_{dstip}, H_{srcp}, and H_{dstp} obtained at each data window ($W = 100$) for 5000 data packets collected under normal conditions, 5000 data packets collected under a DDoS attack traffic level of 50%, and 5000 data packets collected under a DDoS attack traffic level of 90%. The values of H_{srcip}, H_{dstip}, H_{srcp}, and H_{dstp} obtained under the three conditions are presented in Figs. 6, 7, 8, and 9, respectively. The results clearly demonstrated that the entropy values were substantially altered under the DDoS attack conditions.

Based on the stated purpose of the initial detection module, it must have a high DR value, while a low FPR value is not essential. This is the basis for setting the threshold value T. Separate analyses (not shown) were conducted to determine the minimum threshold Tmin required to obtain a DR value of 100% (i.e., FN = 0) based solely on Hsrcip, Hdstip, Hsrcp, or Hdstp. Based on Hsrcip, Tmin = 2.3 yielded an FAR of 71%. Based on Hdstip, Tmin = 3.4 yielded an FAR of 68%. Based on Hsrcp, Tmin = 1.9 yielded an FAR of 56%. Finally, based on Hdstp, Tmin = 1.6 yielded an FAR of 76%. Accordingly, we can conclude that the initial detection module based on information entropy provides a detection rate of 100% based on any one of the four entropy terms with an appropriately set threshold between 1.6 to 3.4, and it can therefore identify DDoS attack traffic effectively and quickly. As such, the confirmation detection module based on the SSAE-SVM needs only to distinguish between true attacks and false alarms.

5.3. Impact of different network structures

In deep neural networks, there is no fixed method to determine the number of hidden layers and the number of neurons in each layer. We need to set up different network structures according to different experimental backgrounds. If the number of hidden layers is small and the number of neurons in each layer is insufficient, the model may not be able to effectively match the distribution of data. For the SSAE network, the high-dimensional features would not be able to be effectively compressed. In contrast, if the number of hidden layers is too large and the number of neurons in each layer is too large, it may lead to an extremely complex training process of the model, which greatly increases the training time and the consumption of computational resources. At the same time, it may lead to over-fitting of the model.

Table 4 shows the impact of the SSAE on the performance of the classifier when different hidden layers were used on the validation dataset Dataset 2, including the mean and standard deviation of each indicator. To achieve the purpose of the controlling variables, we connected the same Softmax classifier to the SSAE output layer for different structures, and the remaining algorithm parameters in the experiment were consistent. Under the above settings, with the increase in the number of hidden layers, the classification results of the classifier became more and more satisfactory. However, a major drawback of multi-layer network architecture is that it requires more time to train, which makes it infeasible in real network environments. In addition, when the model is in a big data environment, the training time blows up. Therefore, after a comprehensive comparison, the four-layer hidden structure was the most suitable network structure for our experiment.

Fig. 10 shows the influence of the sparse parameters between 0.05 and 0.5 on the SSAE classification results of the four-hidden layer model [80,60,60,40]. When the sparse parameter value was 0.2, the comprehensive performance of the SSAE was the best. Below this value, it would lead to a deficit in the number of neurons used for training, so there was a downward trend in detection performance, and the specific value, the input function cannot be effectively compressed, then the training time of the model is maintained over a long period of time.

In this work, an SSAE network of 80–60–60–40 was implemented in TensorFlow, and the corresponding sparse factors were set to 0.2. The performance of the SSAE-SVM DDoS attack detection module for Dataset 2 and the DARAR dataset was compared

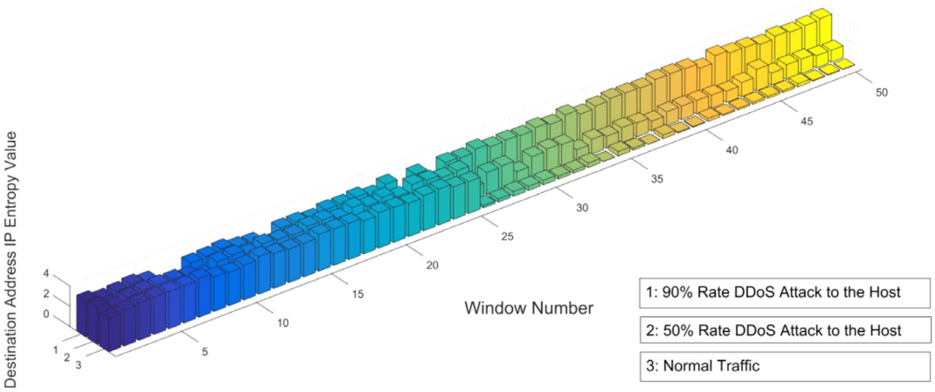


Fig. 7. Entropy value of destination IP addresses (H_{dstip}).

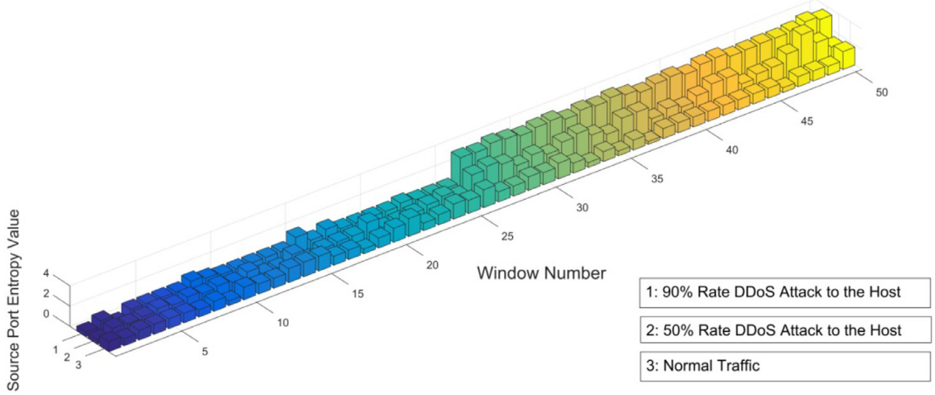


Fig. 8. Entropy value of source port number (H_{srcp}).

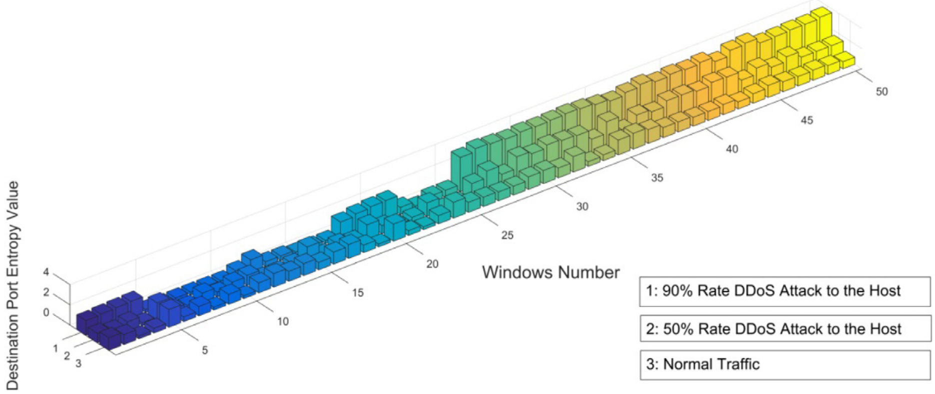
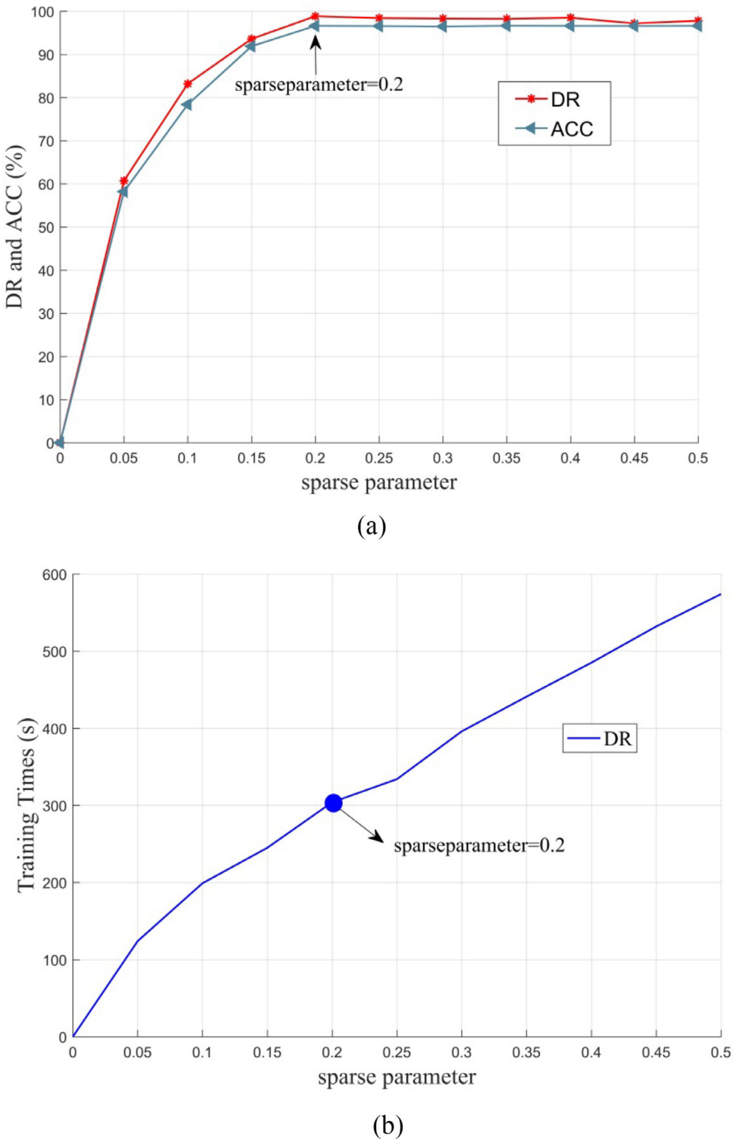


Fig. 9. Entropy value of destination port number (H_{dstp}).

Table 4
Detection rate (DR), accuracy (ACC), false alarm rate (FAR) and training time of SSAE with different hidden layer structures.

Hidden layers	DR(%)	ACC(%)	FAR(%)	Training times/s
[110,90,70,50, 30, 15]	98.87 ± 0.122	98.63 ± 0.142	2.10 ± 0.0295	392
[100,80,80,60,40]	98.88 ± 0.124	98.62 ± 0.136	2.15 ± 0.035	361
[80,60,60,40]	98.86 ± 0.119	98.63 ± 0.127	2.13 ± 0.031	304
[80,60,40]	93.65 ± 0.269	93.12 ± 0.247	3.12 ± 0.038	256
[80,40]	80.22 ± 0.372	79.32 ± 0.386	6.47 ± 0.143	123
[60]	72.31 ± 0.421	71.72 ± 0.413	8.31 ± 0.160	116



(a)

(b)

Fig. 10. (a) Detection rate (DR) and accuracy (ACC) and (b) training times as the sparse parameter changes from 0.05 to 0.5.

Table 5
Comparison of the results obtained for the proposed SSAE-SVM DDoS attack detection module with those of previously proposed attack detection algorithms.

Algorithm	DARPAR Dataset				Dataset 2			
	DR (%)	ACC (%)	FAR (%)	Training Time/s	DR (%)	ACC (%)	FAR (%)	Training Time/s
K-NN[18]	92.08± 0.132	95.31 ±0.165	7.76 ±0.052	385	94.75 ± 0.141	95.81 ± 0.139	7.98 ± 0.038	634
SVM[21]	94.27 ± 0.214	94.61 ± 0.237	8.68 ± 0.066	323	94.88 ± 0.208	95.71 ±0.215	4.32 ± 0.063	679
DNN-softmax[11]	95.78 ±0.169	97.90 ± 0.184	4.98 ±0.043	293	95.42 ±0.174	97.87 ± 0.182	3.16 ± 0.051	467
SAE-softmax[12]	95.81 ± 0.146	97.04± 0.153	4.98 ± 0.066	133	95.67 ±0.152	97.82 ± 0.143	3.15 ± 0.068	376
SSAE-SVM	**96.43 ± 0.129**	**99.01 ±0.139**	**4.03 ± 0.032**	**43**	**96.86 ± 0.119**	**98.63 ± 0.127**	**2.13 ± 0.032**	**304**

with those obtained using other state of the art methods, including DNN and SAE methods with the same number of layers, including a Softmax layer. Moreover, comparative experiments were also conducted with unsupervised attack detection methods, including the K-NN method and SVM algorithm . All the experiments were conducted on a personal computer with a Linux operating system and a CPU operating at 3.4 GHz with 64 GB of memory.

5.4. Results of detection module based on SSAE-SVM

The detection results obtained for the various DDoS attack detection algorithms considered are listed in Table 5, which also includes the model training times required by the SSAE-SVM, DNN, and SAE methods. We note that the proposed SSAE-SVM module provided superior detection performance based on all indicator values relative to all other methods considered, and the model training time required by the SSAE-SVM module was particularly reduced. The unsupervised, non-linear feature extraction algorithm based on the SSAE network proposed in this paper clearly improved the overall attack detection effect of the method relative to those of the other machine learning methods based on the DNN and SAE. The results demonstrated that the SSAE network could extract deep features and provide better generalization capabilities due to the addition of sparseness restrictions. At the same time,

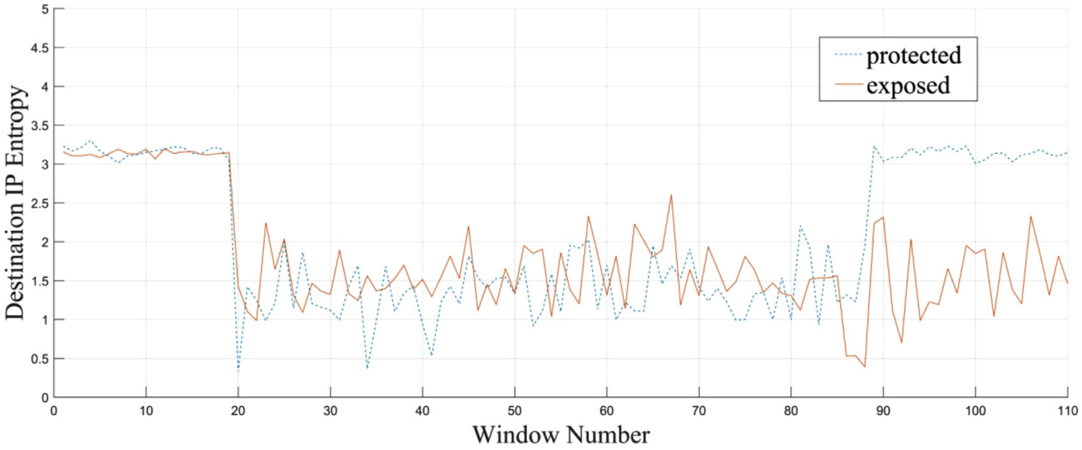


Fig. 11. Comparison of the destination IP address entropy values obtained with and without the proposed defense module.

Table 6
Comparison of the results obtained for the proposed SSAE-SVM DDoS attack detection module with the entropy-based method.

Method	DR(%)	ACC(%)	FAR(%)
Entropy	91.43 ± 0.261	90.80 ± 0.237	13.12 ± 0.142
SSAE-SVM	96.86 ± 0.119	98.63 ± 0.127	2.13 ± 0.032

Table 7
Comparison of the accuracy (ACC) and time cost of the SSAE-SVM algorithm obtained for Dataset 2.

Input features	ACC(%)	Time cost(s)
17 OpenFlow extracted features	97.16	67.54
All 19 features	98.63	67.57

the SSAE network required only 60 pre-training and 80 fine-tuning iterations, while the DNN required 500 training iterations. Accordingly, we can conclude that the pre-training and fine-tuning methods facilitated the more rapid convergence of the model.

Table 6 shows the recognition rate, accuracy rate, and false alarm rate of the detection method based on the information entropy and the SSAE-SVM for DataSet 2. The results show that the detection method based on the SSAE-SVM made up for the deficiency of the information-entropy-based method, which greatly improved the detection recognition rate and accuracy while significantly reducing the false alarm rate.

The impact of the calculated features *n_packets_ave* and *n_bytes_ave* on the attack detection performance of the SSAE-SVM algorithm was also investigated by comparing the ACC values obtained for Dataset 2 using only the 17 directly extracted OpenFlow feature fields and all 19 extracted features. The results are listed in Table 7. Including the features *n_packets_ave* and *n_bytes_ave* increased the accuracy of the DDoS attack recognition by 1.5%, while the time cost was only increased by 0.03 s.

5.5. Results of overall HESS method

The effectiveness of the overall HESS method was first investigated under hybrid DDoS attack conditions with traffic levels of 25%, 50%, and 80%. We compared the detection accuracy, CPU utilization, and time cost obtained using the initial detection module based on information entropy in sequence with the confirmation detection module based on the SSAE-SVM with those obtained using only the detection module based on the SSVE-SVM, and the results are listed in Table 8. Both methods obtained high accuracy, with values greater than 98%, for detecting abnormal network traffic. However, the combined method greatly decreased the CPU utilization and time cost owing to the greatly reduced computational burden associated with the initial detection module based on the information entropy, which significantly reduced the extent to which the SSAE-SVM module must be called.

The effect of the defense module can be evaluated in terms of its impact on the DDoS attack and the period required to locate the attack ports and issue filtering rules from the forwarding layer. The defense performance obtained by the proposed HESS method can be evaluated according to the value of H_{dstip} calculated with respect to the window number ($W = 100$) under a hybrid DDoS attack, as shown in Fig. 11. The values of H_{dstip} obtained with and without the attack defense module are given by the dashed and solid lines, respectively. The DDoS attack was initiated in the 20th data packet window, and the values of H_{dstip} decreased sharply both with and without the defense module. However, the value of H_{dstip} returned to its original level at the 90th data packet window under the action of the defense module. Accordingly, we concluded that the DDoS attack was completely nullified by the defense module after a period equivalent to 70 data packet windows.

6. Conclusion

The present work addressed the unique susceptibility of networks based on SDN to DDoS attacks. A hybrid approach was pro-

Table 8
Comparison of the detection performances of the combined information entropy and SSAE-SVM detection modules and the SSAE-SVM detection module alone for different levels of DDoS attack traffic.

	ACC (%)	CPU Utilization (%)	Time Cost (s)	ACC (%)	CPU Utilization (%)	Time Cost (s)	ACC (%)	CPU Utilization (%)	Time Cost (s)
SSAE-SVM	98.64	47	45	98.71	52	57	98.69	67	67
Entropy and SSAE-SVM	**98.76**	**18**	**19**	**98.79**	**22**	**21**	**98.73**	**46**	**34**

In pdf:

posed to detect DDoS attacks using an initial detection module based on information entropy to quickly identify anomalous traffic and a second detection module based on machine learning with an SSAE-SVM architecture to confirm the suspected anomalous traffic. If DDoS traffic is detected, a defense module is implemented to restore normal network communication in a timely manner via an issued flow table. The results of simulation experiments demonstrated that the SSAE-SVM module provided significantly greater detection rates and accuracies with reduced false alarm rates and greatly reduced training times than other state-of-the-art DDoS attack detection methods. Moreover, the CPU utilization and time cost when using the initial detection module based on information entropy in sequence with the confirmation detection module based on the SSAE-SVM were substantially less than those obtained using the detection module based on the SSAE-SVM alone, indicating that the combined approach is much more computationally efficient. Finally, the defense module was demonstrated to provide effective and timely DDoS attack defense. In the future, we plan to increase the real-time performance of the HESS method by introducing correlations between features.

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.

Supplementary materials

Supplementary material associated with this article can be found, in the online version, at doi:10.1016/j.cose.2022.102604.

Zhang Long was born in Hebei Province, China, in 1987, PhD candidate. His-current research interests include cyber security, deep learning and software-defined networking (SDN).

Wang Jinsong was born in Tianjin, China, in 1970. He is professor and PhD supervisor in Tianjin university of technology, director of China Computer Federation. His-main research interests include computer network, information security and blockchain.