SCIENCE CHINA
Information Sciences

• RESEARCH PAPER •

March 2019, Vol. 62 032106:1–032106:15
https://doi.org/10.1007/s11432-017-9439-7

CATH: an effective method for detecting denial-of-service attacks in software defined networks

Yi GUO[1,2*], Fu MIAO[1], Liancheng ZHANG[1] & Yu WANG[1,3]

[1]*State Key Laboratory of Mathematical Engineering and Advanced Computing, Zhengzhou* 450001*, China;*
[2]*Institute for Network Science and Cyberspace, Tsinghua University, Beijing* 100084*, China;*
[3]*School of Computer Science, Henan University of Engineering, Zhengzhou* 451191*, China*

Received 24 November 2017/Revised 22 February 2018/Accepted 30 March 2018/Published online 12 February 2019

Abstract Software defined networks (SDNs) are innovative network frameworks that have recently received wide attention. Their programming flexibility facilitates automatic network management and control, thus mitigating existing issues in the traditional network architecture. However, SDNs face several security risks, in particular denial-of-service (DoS) attacks, the most common and serious network attacks. To address such a threat, an SDN-DoS attack detection method is proposed based on fusing multiple flow features for describing the network catastrophe between the normal and the attack state. Several statistic attributes of SDN flow information are first chosen as detection features; subsequently, the cusp model is used to establish a catastrophe equilibrium surface for SDN states. After being trained, the cusp catastrophe model can be utilized to infer whether an SDN is under DoS attack. The experimental results demonstrate that the method can effectively and timely perceive SDN-DoS attacks, not only in simple networks but also in larger enterprise networks.

Keywords DoS attacks, software defined network, flow features, cusp model, equilibrium surface

Citation Guo Y, Miao F, Zhang L C, et al. CATH: an effective method for detecting denial-of-service attacks in software defined networks. Sci China Inf Sci, 2019, 62(3): 032106, https://doi.org/10.1007/s11432-017-9439-7

1 Introduction

Software defined networks (SDNs) represent one of the most widely used software-based network architecture and have loosely coupled control and data planes. They support centralized network state control, and their underlying facilities are transparent to upper layer applications [1,2]. Their programming flexibility facilitates automatic network management and control, thus mitigating existing issues in the Internet, such as inflexibility in network reorganization and resource expansion. Accordingly, SDN-related technologies and businesses have rapidly developed over the past few years and have been widely deployed in various network environments, such as backbone networks, data centers, enterprise networks, and mobile networks.

The centralized control mechanism and open programming interface of SDNs have increased the flexibility of network management and operation. However, they also provide new and greater opportunities for network attacks [3]. Specifically, the centralized management structure causes all network "intelligence" to concentrate on the controllers. Once some controllers fail or their service capacity declines, the performance of the global network is seriously affected. Currently, the most common and effective attacks on SDNs are denial-of-service (DoS) or distributed denial-of-service (DDoS) attacks from the data plane. Henceforth, these will be called SDN-DoS attacks. The attacks presented in [4, 5] are two

* Corresponding author (email: nongfu@live.cn)

Table 1 Typical SDN-related DoS attack detection mechanisms

Corresponding scenario	DoS detection method	Basic principle
DoS detection in SDN	AvantGuard [8]	Using SYN proxy based module to verify the legality of each flow.
	FloodGuard [9]	Utilizing the real-time rate of PACKET_IN messages and the infrastructure (controller memory and CPU) to identify potential flooding attacks.
	Entropy-based methods [10, 11]	Identifying attacks by comparing the values of selected flow features with their preset values.
	SOM-based methods [12, 13]	Using the self-organizing map machine learning technique for detecting SDN-aimed DoS attacks.
SDN for DoS detection	Fresco [14, 15]	Using the OpenFlow technology as a flow regulation tool to monitor traffic.
	VAVE [16]	Utilizing the SDN architecture to validate source addresses.
	Bohatei [17]	Using the flexibility of SDN to steer suspicious traffic through the defense VMs while minimizing user-perceived latency and network congestion.

typical SDN-DoS attacks. They are based on sending a large number of carefully constructed data flows to a target SDN. As the OpenFlow [6] switches in the target SDN have no flow table entry to match those malicious flows, they will send PACKET_IN messages to SDN controllers to obtain suitable forwarding rules. That is, the controllers should generate and publish new rules to the switches by sending PACKET_OUT messages. Obviously, if there is an overly large number of un-matched flows, both the storage and computing resources of the controllers will be rapidly consumed, as well as the connection resources between the controllers and the switches. As a result, most normal forwarding requests will be rejected. In this case, the SDN is in a denial-of-service state.

There are several effective DoS/DDoS attack detection solutions for traditional networks. However, little research has been conducted on SDN-DoS attack detection [7]. Typical SDN-related DoS attack detection mechanisms are summarized in Table 1. AvantGuard [8] introduced an SYN proxy based module to verify the legality of each flow based on TCP handshake. FloodGuard [9] used both the real-time rate of PACKET_IN messages and the infrastructure (such as buffer memory and CPU) to identify flooding attacks based on a certain anomaly threshold. Giotis et al. [10] proposed an entropy-based anomaly detection method implementing the sFlow technology to collect traffic. Mousavi et al. [11] proposed using the central control of SDNs for attack detection and introduced an entropy-based solution. These entropy-based methods have the advantage of flexible configuration and convenient computation; however, expert experience is required for determining the threshold and assigning weight values to key factors, which puts high demands on users. Braga et al. [12] used a self-organizing map (SOM) machine learning technique for detecting SDN-DoS attacks. Yao et al. [13] proposed an SDN-DoS attack detection method based on object features using the growing hierarchical SOM (GSHOM) technology. Common problems in SOM-based methods are slow convergence and long training time under suboptimal conditions. Shin et al. [14,15] used the OpenFlow technology as a flow regulation tool to monitor traffic in cloud networks. Yao et al. [16] utilized the SDN architecture to validate source addresses. Fayaz et al. [17] used the flexibility of SDNs to steer suspicious traffic through the defense virtual machines (VMs) while minimizing user-perceived latency and network congestion. These studies address traditional network security threats using SDN-related technologies rather than focus on security issues in the new network paradigm.

In the present study, the principles and characteristics of SDN-DoS attacks are first analyzed, and it is demonstrated that there are certain obvious catastrophe characteristics when an SDN suffers a DoS attack. Subsequently, using an idea from catastrophe theory in system state transition modeling, an SDN-DoS attack detection method is proposed that is called CATH. As shown in Figure 1, certain statistic attributes of SDN flow information are selected as detection features (DFs). They can effectively represent the catastrophe state of an SDN. According to the number of state variables and control variables obtained from detection features, the cusp model is used to establish a catastrophe equilibrium surface for SDN

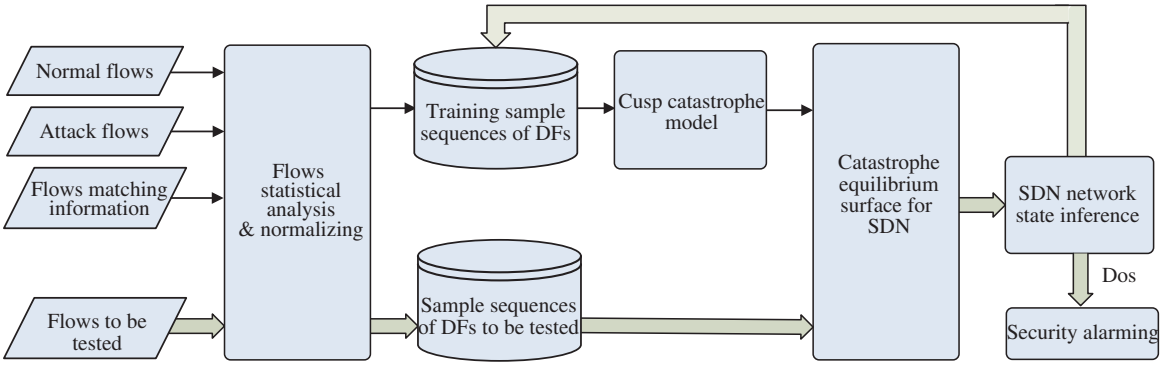

Figure 1 (Color online) Workflow of CATH.

states, including the normal and the attack states. After being trained, the cusp catastrophe model can be used to infer whether the network is under attack, thus allowing real-time detection of SDN-DoS attacks.

The remainder of the paper is organized as follows. In Section 2, a brief analysis of SDN-DoS attacks is provided. In Section 3, the feasibility of utilizing catastrophe theory to detect SDN-DoS attacks is demonstrated. The CATH method, which can effectively perceive SDN-DoS attacks, is presented in Section 4. In Section 5, two experiments are conducted to verify the effectiveness of the proposed method. Section 6 concludes the paper.

2 Analysis of SDN-DoS attacks

As a current key protocol of SDNs, OpenFlow defines the communication standard between SDN controllers and switches. The switch flow tables, which are used to indicate the forwarding paths, are distributed from the controllers through the OpenFlow protocol. It can be said that all SDN "intelligence" is concentrated on the controllers. If they fail or their service capacity declines, the network performance is seriously affected.

DoS attacks are one of the most common and serious threats in the Internet [18]. They can be carried out using a variety of techniques, such as SYN flood, smurf, and ping-of-death. However, they have the same goal, that is, to render target hosts or networks unable to provide normal services [19, 20]. For example, to conduct an SYN flood attack, attackers should send a large number of SYN TCP attack packets directly to the target (may be a host or a network), thus exhausting the connection resources of the target because it should wait for an excessively large number of TCP semi-connections. However, these target-oriented DoS attacks are no longer effective against SDN controllers because the controller is transparent and invisible to network users. Switches are the only intermediaries between users and an SDN, and the ports of a switch connected to users are isolated from the ports connected to the controllers. Therefore, users cannot directly send flows to the controllers, i.e., the controllers are not reachable by users.

When an OpenFlow switch receives network flows, it performs the matching process according to the local flow tables. If the match is not successful, the switch extracts the header of the flow and encapsulates it as a PACKET_IN message that is sent to the controller immediately. If the flow table or the cache of the switch is full, the entire received packet is encapsulated as a PACKET_IN message and is then sent to the controller. Utilizing the characteristics of the above mechanism, an attacker can generate a large number of attack flows and send them to the target SDN, as shown in Figure 2. The controller should use most of its resources to handle those malicious flows. Thus, the service capability of the controller substantially declines, and a DoS attack is successfully implemented on the target network.

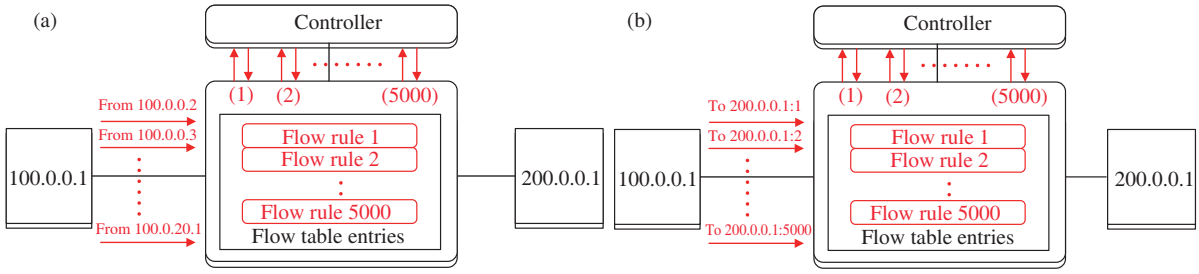

Figure 2 (Color online) SDN-DoS attacks based on (a) forged source IP and (b) port transformation.

3 Feasibility of catastrophe theory for detecting SDN-DoS attacks

A catastrophe is a process that leads from quantitative changes to qualitative changes. There are various catastrophes in real life, such as rock ruptures, bridge collapses, and tire bursts. Catastrophe theory is a methodology for studying catastrophe phenomena and their regularity in nonlinear systems. In this section, a brief introduction to catastrophe theory is provided, and catastrophe characteristics of SDNs when suffering DoS attacks are analyzed, so that the feasibility of catastrophe theory for detecting SDN-DoS attacks may be demonstrated.

3.1 Catastrophe theory

Catastrophe theory is a branch of bifurcation theory in the study of dynamical systems [21, 22]. It is concerned with the case that the long-run stable equilibrium can be identified with the minimum of a smooth and well-defined potential function $F(X, B)$, where $X = (X_1, X_2, \ldots, X_n)$ is an n-dimensional state variable and $B = (B_1, B_2, \ldots, B_m)$ is an n-dimensional control variable of the system. The state variable reflects the external changes of system behavior, whereas the control variable controls the intrinsic nature of the system [22]. It can also be said that the change of the state variable describes the interaction between control variables.

If $F'(X, B) = 0$, an equilibrium surface is obtained consisting of all critical points. If $F''(X, B) = 0$, the equation of the singular point set corresponding to the equilibrium surface can be obtained. Furthermore, if $F'(X, B) = 0$ and $F''(X, B) = 0$ simultaneously, a bifurcation set is obtained, which is the projection of the singular point set on a particular control space. In catastrophe theory, the bifurcation set is used to determine whether catastrophes have occurred in a system.

The potential function of a catastrophe model may be highly complicated. However, if the number of control variables is at most four, there are at most seven types of catastrophes, namely, fold catastrophe, cusp catastrophe, swallowtail catastrophe, butterfly catastrophe, hyperbolic umbilic catastrophe, elliptic umbilic catastrophe, and parabolic umbilic catastrophe [21].

3.2 Catastrophe characteristics of SDN flows

As catastrophe theory is capable of explaining and predicting various catastrophe phenomena, it has been applied in several fields, such as physics, biology, medical engineering, and social science. For example, the cusp catastrophe model can be used to analyze the phase transition law of eutectoid steel.

There is no difference between SDNs and traditional networks in terms of main functions and workload. The traffic carried by an open SDN is the same as the global Internet traffic, which presents several catastrophe characteristics. Although certain statistical attributes of network traffic, such as the average packet number in a flow, the percentage of correlative flow, the number of source IP address, and the number of ports, frequently fluctuate in a certain period, they vary within a small range and have a stable and continuous trend, implying that the network is in an equilibrium state. In this case, even if there are some disturbances, their impact will soon disappear.

The traffic of any open network is constantly moving, as it is in close contact with other networks. In terms of system methodology, the moving traffic can be treated as a complex dynamical system. With

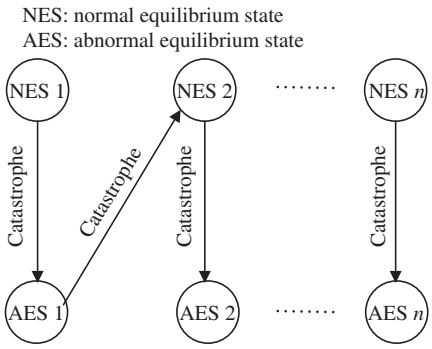

Figure 3 Catastrophe between normal/abnormal equilibrium state.

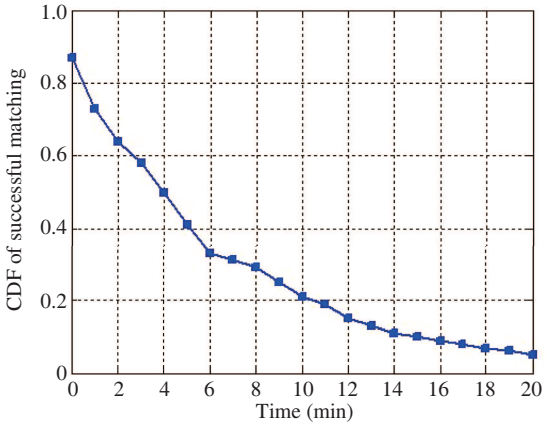

Figure 4 (Color online) CDF of the flow table matching rate in a DoS attack.

the occurrence of various events, a network shifts from one equilibrium state to another. As shown in Figure 3, an abnormal event transfers the network from a normal equilibrium state (NES) to an abnormal equilibrium state (AES). As an SDN-DoS attack is to occupy service resources of the target network and to render it unable to respond to normal requests, when a network is under SDN-DoS attack, it is said to be in the DoS state. Once the anomaly disappears, the network returns to another normal equilibrium. Different types of anomalies would take the network to different equilibrium states. The procedure of equilibrium transformation is not a slow and gradual change but a non-stationary and discontinuous catastrophe process.

Guo et al. [23] and Shin et al. [8] analyzed the change of several statistical attributes of network traffic in various cases. They drew the conclusion that regardless of network state, certain statistical attributes of the traffic, namely, the packet number in a single flow, the source IP growing speed, and the port generating speed, fluctuate within a small range, whereas they greatly vary in different states. Considering that the normal state (NS) and the DoS state (DS) are two different equilibrium states of an SDN, there must be a qualitative change when the network transitions between the two states. As shown in Figure 4, the flow table matching rate of an SDN decreases exponentially from the NS to the DS. CDF is the abbreviation of cumulative distribution function. It is also verified that the transition is sudden and dramatic.

This analysis implies that in an SDN, both the NS and the DS are stable equilibrium states. The transition between them is a process of nonlinear and non-stationary qualitative change. All these characteristics are similar to a catastrophe process in catastrophe theory. Therefore, utilizing catastrophe theory to detect the SDN-DoS attack is quite sensible.

4 SDN-DoS attack detection method

As the transformation of an SDN from the normal state to the DoS state is a typical catastrophe, an SDN-DoS attack detection method based on catastrophe theory is proposed. It is called CATH, and in this section, its principles and workflow are presented in detail.

4.1 Detection features and normalization process

The features that can accurately represent the network state are selected to form the detection feature set. Thus, SDN-DoS attack detection can be achieved by tracing the changes of detection features.

(1) Detection features. As analyzed in Section 2, to perform an effective DoS attack, attackers should carefully schedule their own computing and bandwidth resources, so that a large number of attack flows into the target SDN may be rapidly assembled. The values of several statistical attributes of a network

under DoS attack are quite different from those in the normal state. Those attributes reflecting the catastrophe process from a normal state to a DoS state will be used as SDN-DoS attack detection features to jointly describe the operating state of an SDN.

Definition 1. The size of a single flow (SSF) describes the size of every flow entering the SDN. It includes two attributes, the number of packets in a single flow (NPF) and the number of bytes of a single flow (BSF). Moreover, the average NPF (ANPF) and the average BSF (ABSF) are defined, namely, $\mathrm{ANPF} = \sum_{i=1}^{\mathrm{FlowNum}} (\mathrm{NPF}_i)/\mathrm{FlowNum}$ and $\mathrm{ABSF} = \sum_{i=1}^{\mathrm{FlowNum}} (\mathrm{BSF}_i)/\mathrm{FlowNum}$, where FlowNum is the number of flows entering the network during the sampling period, NPF_i is the number of packets in flow i, and BSF_i is the number of bytes of flow i. In the following, the variable γ will be used to represent SSF, namely, $\gamma = \{\mathrm{ANPF}, \mathrm{ABSF}\}$.

To carry out an SDN-DoS attack more effectively, attackers increase the rate of generating attack flows, usually by reducing the number of packets in every flow. In addition, as the number of bytes of every single attack flow has little effect on the attack performance, the flows used to carry out SDN-DoS attacks are always considerably smaller than normal flows. Thus, if a network suffers a DoS attack, both ANPF and ASPF are significantly smaller than in the normal state.

Definition 2. The address growing speed (AGS) consists of two attributes: source IP growing speed (IPGS) and port generating speed (PGS). They are defined by IPGS = SrcIP_Num/interval and PGS = PortNum/interval, where SrcIP_Num and PortNum represent the number of source IP addresses and the number of ports, respectively, that appear in a sampling period, and interval is the length of a sampling period. Similarly, in what follows, a single variable will be used to represent AGS, namely $\rho = \{\mathrm{IPGS}, \mathrm{PGS}\}$.

By analyzing the traffic collected when the SDN runs normally, it is seen that both the number of source IP addresses and the number of ports are stable, exhibiting only a small fluctuation. However, in an SDN-DoS attack, attackers construct and send a large number of attack flows to the target SDN network. The common method for constructing attack flows is to generate packets with different source IP addresses and port numbers. Thus, when a network is attacked, the number of source IP addresses and ports grow rapidly. That is, IPGS and PGS in the DoS state are significantly larger than in the normal state.

Definition 3. The flow table matching ratio (MR) is the proportion of flows that can be matched directly by switch flow tables. That is, MR = MatchNum/FlowNum, where MatchNum is the number of matched flows, and FlowNum is the total number of flows entering the network during a sampling period. In the following, the variable λ will be used to represent MR.

The main goal of an SDN-DoS attack is to induce controllers to frequently generate and send new flow rules to switches; thus, the attacker should construct and send large numbers of unmatched flows to the target SDN. In this case, MR would decline substantially. Namely, if an SDN is under SDN-DoS attack, MR will obviously be smaller than in the normal state.

There are several features that can characterize the SDN catastrophe between the normal state and the DoS state. A larger number of features results in a more accurate catastrophe model but also in higher overall computational overhead and thus reduced detection efficiency. Based on the statistical analysis of a large amount of historical traffic data, three features are selected, namely, SSF, AGS, and MR, to jointly describe the catastrophe of the SDN.

(2) Normalization process. The value ranges and dimensions of the three detection features are quite different, and a feature with large value range has great influence on the calculation. To unify the value ranges of different features, normalization in [0,1] is performed. As these three values do not have probability distribution characteristics, the Min-Max scaling method is adopted to linearly normalize the raw data and achieve equal scaling. Specifically, all detection features should be normalized, including ANPF and ABPF in SSF, IPGS and PGS in AGS, and MR. The process is described as $x' = \frac{x - x_{\min}}{x_{\max} - x_{\min}}$, where x' is the normalized data point, x is the raw data point, and $x_{\max}$ and $x_{\min}$ are the maximum value and the minimum value of the raw dataset, respectively.

Furthermore, the catastrophe progression method is utilized to fuse SSF and AGS into one variable,

as each of them contains multiple sub-features. If a feature contains two sub-features, the cusp normalization formula is usually selected for the fusion process, whereas if a feature can be decomposed into three sub-features, the dovetail formula should be selected. As SSF has two sub-features (ANPF and ABSF), and ANPF has greater potential for characterizing SDN-DoS attacks, the cusp normalization formula is adopted, namely, $\gamma = \min(\sqrt{\text{ANPF}'}, \sqrt[3]{\text{ABSF}'})$, where ANPF' and ABSF' are the normalization values of ANPF and ABSF, respectively. AGS also contains two sub-features (IPGS and PGS), and IPGS has greater characterization potential; thus, the cusp formula can be used as well, i.e., $\rho = \min(\sqrt{\text{IPGS}'}, \sqrt[3]{\text{PGS}'})$, where IPGS' and PGS' are the normalization values of IPGS and PGS, respectively.

4.2 Cusp catastrophe model

In catastrophe theory, the system state is described by the potential function, and estimating the system state can be reduced to computing the minimal value of the potential function, whose type is determined by the number of its control and state variables [21].

(1) Catastrophe potential function. The factors that may cause network catastrophe are referred to as control or external variables, whereas factors that may vary during a catastrophe are called state or internal variables. If a network suffers SDN-DoS attacks, it receives a large number of abnormal flows (unmatched flows). That is, the transition from the normal state to the DoS state is due to the large-scale abnormal incoming traffic, resulting in a dramatic decrease of the flow table matching rate. Therefore, the traffic features are external factors that lead to a state catastrophe, and the flow table matching rate is a varying factor. Thus, the two traffic features (SSF and AGS) are selected as control variables, and MR as a state variable. According to Thom's research in [21], the cusp catastrophe potential function that is suitable for this case has the form $F(x) = x^4 + aux^2 + bvx$, where x is a state variable, u and v are control variables, and a and b are two parameters. The equilibrium surface of the catastrophe model is described as follows:

$$M : F'(x) = 4x^3 + 2aux + bv = 0, \tag{1}$$

and the surface equation of the singular set S can be described as

$$S : F''(x) = 6x^2 + au = 0. \tag{2}$$

Eliminating the variable x by combining (1) and (2), we obtain an expression for the bifurcation set. It consists of the critical points of the equilibrium surface. The bifurcation set is also a control space, where the network state catastrophe takes place:

$$\text{Bs} : 8a^3u^3 + 27b^2v^2 = 0. \tag{3}$$

The geometric structure of the cusp model is shown in Figure 5. The upper part is the equilibrium surface of the cusp catastrophe model. It consists of an upper leaf (A), a middle leaf (B), and a lower leaf (C). A represents the normal stable state, C represents the DoS equilibrium state, and B is the unstable position, namely, the inaccessible region of the network. If the network transits from point p of A to point q of C, a sudden jump occurs, which is a so-called catastrophe process. The lower part of the figure refers to the bifurcation set, which is governed by the control variables u and v. In fact, the bifurcation set is the projection of the cusp catastrophe manifold on the u-v plane. From the projection plane, it can be seen that the trajectory from p to q passes through the bifurcation set (Bs) curve.

(2) Estimating model parameters. As the solution of a potential function may be positive or negative, a translation transformation of the sample data should be performed to easily distinguish the normal state and the DoS state. Let the normalized sample sequences $\widetilde{U} = \{\gamma^k\}, \widetilde{V} = \{\rho^k\}$, and $\widetilde{X} = \{\lambda^k\}$ be defined, where $k = 1, 2, \ldots, N$ and N is the length of a sample sequence. The translation transformation of the sample sequences is as follows.

The translated sequence of the state variable $\widetilde{X} = \{\lambda^k\}$ is $X = \{(\lambda^k - x_{\text{avg}}) | k = 1, 2, \ldots, N\}$, where x_{avg} is the average value of $\widetilde{X}$, namely, $x_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} \lambda^i$. For the two control variables, their sample

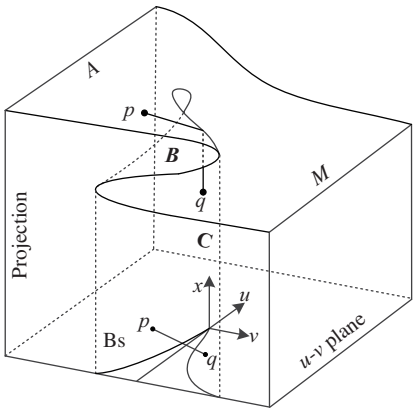

Figure 5 Geometry of the cusp catastrophe model.

sequences $\widetilde{U} = \{\gamma^k\}$ and $\widetilde{V} = \{\rho^k\}$ can be translated into $U = \{(\gamma^k - u_{\max})|k = 1, 2, \ldots, N\}$ and $V = \{(\rho^k - v_{\text{avg}})|k = 1, 2, \ldots, N\}$, where $u_{\max} = \max(\gamma^1, \gamma^2, \ldots, \gamma^N)$ and $v_{\text{avg}} = \frac{1}{N}\sum_{i=1}^{N} \rho^i$.

a and b are two parameters of the cusp model. Their optimal values are obtained by determining the extremum of multiple functions. Specifically, the least square fitting method is used. $J(a, b)$ is defined as the square sum of M and Bs, namely,

$$J(a,b) = \sum_{i=1}^{N} \{[4(X_1^i)^3 + 2aU_1^iX_1^i + bV_1^i]^2 + [4a^3(U_1^i)^3 + 27b^2(V_1^i)^2]^2\}. \tag{4}$$

Furthermore, the partial derivatives of $J(a, b)$ with respect to a and b are obtained and set equal to zero:

$$\begin{cases} \partial J(a,b)/\partial a = 0, \\ \partial J(a,b)/\partial b = 0. \end{cases} \tag{5}$$

That is,

$$\begin{cases} \sum\limits_{i=1}^{N} \{U^iX^i[(X^i)^3 + aU^iX^i + bV^i] + 12a^2(U^i)^3[4a^3(U^i)^3 + 27b^2(V^i)^2]\} = 0, \\ \sum\limits_{i=1}^{N} \{[(X^i)^3 + aU^iX^i + bV^i]V^i + 54b(V^i)^2[4a^3(U^i)^3 + 27b^2(V^i)^2]\} = 0. \end{cases} \tag{6}$$

Eq. (6) is a system of nonlinear equations with two variables. Substituting all sequences (X^i, U^i, V^i) of the training sample set into (6), we obtain six complex solutions and three real solutions for the two parameters (a and b). As X^i and U^i are real numbers, by (2), a and b cannot be complex. Therefore, the six complex solutions for a and b should be ruled out. The real solution that minimizes $J(a, b)$ is the optimal solution.

4.3 Network state inference

After the values of the model parameters have been obtained, the cusp catastrophe detection model can be defined. Hereafter, the model for inferring the state of a testing dataset is utilized. The specific inference procedure is as follows.

Step 1. The testing dataset is normalized and translated in time order. The processed dataset is denoted by $\{(x_{\text{test}}^i, u_{\text{test}}^i, v_{\text{test}}^i)|i = 1, 2, \ldots, m\}$.

Step 2. It is determined whether the projection of a testing sample on the u-v plane is on the left side of the curve Bs, which is defined in (3), namely whether $u_{\text{test}}^i > -\sqrt[3]{27a^2(v_{\text{test}}^i)^2/8b^3}$ and $v_{\text{test}}^i < 0$. If the two conditions are met, the state is normal. Otherwise, if $u_{\text{test}}^i > -\sqrt[3]{27a^2(v_{\text{test}}^i)^2/8b^3}$ and $v_{\text{test}}^i > 0$, which represents that the projection of the testing sample on the u-v plane is on the right side of the curve Bs, then the state is abnormal.

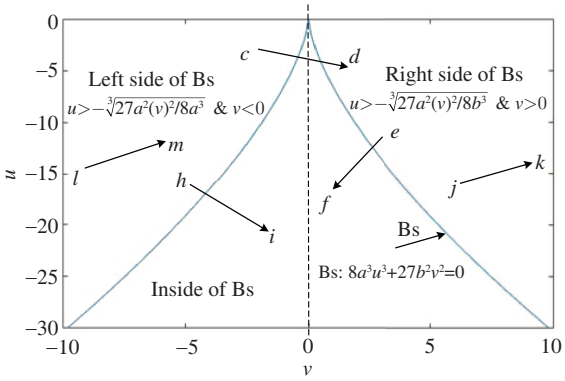

Figure 6 (Color online) Criteria for determining network states.

Step 3. If neither of the two conditions in step 2 is satisfied, the point lies on or inside the curve, and the previous dataset $(x_{\text{test}}^{i-1}, u_{\text{test}}^{i-1}, v_{\text{test}}^{i-1})$ should be used to decide whether $u_{\text{test}}^{i-1} > -\sqrt[3]{27a^2(v_{\text{test}}^{i-1})^2/8b^3}$ and $v_{\text{test}}^{i-1} < 0$. If these two conditions are satisfied, then the network is in normal state; otherwise, it is under SDN-DoS attack.

Step 4. The inference result is put into the network state base in time order; this is necessary for subsequent detection.

In the following, Figure 6 is used as an example to illustrate the inference procedure of the CATH method. There are ten points, c, d, e, f, h, i, j, k, l and m, which are the projections of ten samples onto the u-v plane. $c \to d, e \to f, h \to i, j \to k, l \to m$ represent the sample points of two consecutive periods. The trajectory of $l \to m$ does not pass through the bifurcation curve Bs and lies on its left side. This indicates that both are normal. The trajectory of $c \to d$ passes through the curve Bs, which indicates network catastrophe from the normal state to the DoS state. The trajectory of $j \to k$ is on the right side of the curve Bs, which indicates that the network is in the DoS state at these two consecutive times. The trajectories of $e \to f$ and $h \to i$ pass through the curve Bs, and both points f and i are inside the curve; thus, the previous samples should be combined to complete the inference. As h corresponds to the normal state, it can be inferred that i is in the normal state as well. As e corresponds to the DoS state, it can be inferred that f is also in the DoS state.

5 Experiments and evaluation

To verify the capacity of CATH to detect SDN-DoS attacks and its adaptability to networks of different scale, two experiments are conducted in this section. One is performed on a simple network simulating a small office/home office (SOHO) network, whereas the other experiment is conducted on a complex network simulating a large scale enterprise network.

5.1 Experiment setup

(1) Network emulator. In the following experiments, Mininet[1)] was used as the network emulator. The code developed in Mininet can be easily moved to real production networks. POX [24] was selected for the controllers, and all algorithms were implanted as POX applications written in Python. Open virtual switch (OVS) [2)] was selected as the network switch, and all switches were assumed to communicate with the controllers through the OpenFlow protocol.

To simulate SDNs of different scale, a simple network and a complex network were constructed. Their topologies are shown in Figure 7. The simple network included three switches, whereas the complex network included eight switches.

1) Mininet. http://mininet.org.
2) Open Vswitch. http://openvswitch.org.

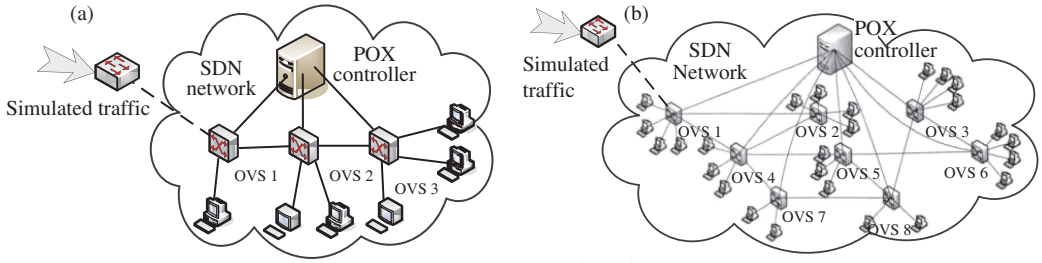

Figure 7 (Color online) Topology of (a) the SOHO network and (b) the larger scale enterprise network.

(2) Traffic generation. To evaluate CATH's performance in a wide range of network environments, benign traffic was collected at two different locations in the network of NDSC (Chinese National Digital Switching System Engineering and Technological Research Center). Initially, the performance was evaluated in a low network traffic environment by utilizing packets traced from our laboratory. Thereby, the network traffic conditions of a SOHO environment could be simulated. For further evaluation of the proposed method, higher traffic was also used to simulate a larger enterprise environment. Hence, network traffic was captured from a part of the NDSC network center. These trace files were employed in the experiments to evaluate the accuracy and detection capacity of the proposed method.

Tcpreplay[3)] is a network tool with the ability to replay the captured traffic at the speed it was captured. It was used in the experiments to replay the captured packet trace files, injecting the generated traffic to a specific Ethernet port. In addition, Scapy[4)] was employed for the attack traces. It is a powerful tool for packet generation and forging. It was used to generate attack traffic with a predefined set of destination IP addresses and port numbers, and a random and constantly varying set of source IP addresses and port numbers.

5.2 Ability to detect SDN-DoS attacks

The application of CATH to detect SDN-DoS attacks is first explained through the experiment with the simple network and the related analysis. In the experiment with the complex network, the detection ability of CATH will be further evaluated.

(1) Attack detection in the simple network. Once the SDN simulation network had run stably for some time, capturing of the real time traffic began. The background traffic was the normal network traffic collected from our lab network (50 Mbps) and replayed by Tcpreplay. It consisted approximately 65% of TCP traffic, 20% of UDP traffic, 5% of ICMP traffic, and 10% of other traffic. The SDN-DoS attack traffic was generated by Scapy and included three types of DoS attack traffic (TCP-SYN-Flood-based, UDP-Flood-based, and ICMP-Flood-based). To simulate different attack rates, attacks of 4000, 4500, and 6000 p/s were carried out.

The background traffic and the attack traffic were sampled in a mixed manner. As there is usually a large amount of background traffic in a real network environment, the sampling period was extended to increase the scale of background traffic for more realistic experimental conditions. The sampling period of background traffic T was gradually extended from 1 to 5 s by increments of 1 s, and the sampling period of attack traffic t was fixed at 0.2 s. After the raw traffic was sanitized, the normal and the attack sample sets of the three detection features were obtained through further statistical analysis. To take into account the accuracy and performance of the detection, 1200 consecutive samples were included in each sample set.

Figures 8–10 show the trends of ANPF, PGS, and MR for the training sample set (600 normal samples and 600 attack samples), where T is 2 s and the attack rate is 4000 flows/s. The trends of ABSF and AGS are not given because they are similar to those of ANPF and PGS, respectively. It can be seen that in the attack period, the value of ANPF is between 1 and 2, which is significantly smaller than in the

3) Tcpreplay. http://tcpreplay.synfin.net.
4) Scapy. http://www.secdev.org/projects/scapy.

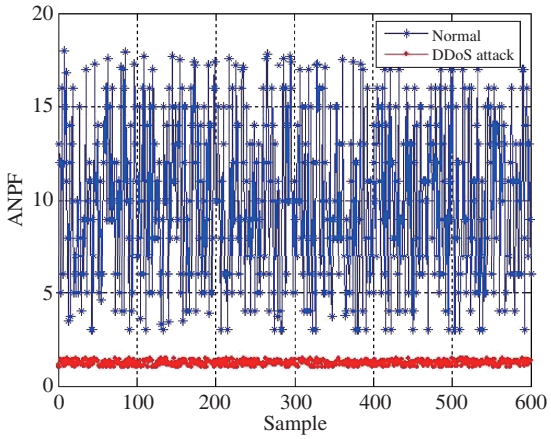

Figure 8 (Color online) ANPF of normal and attack traffic.

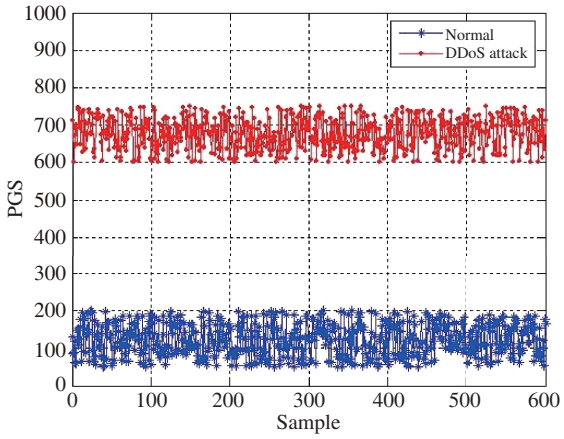

Figure 9 (Color online) PGS of normal and attack traffic.

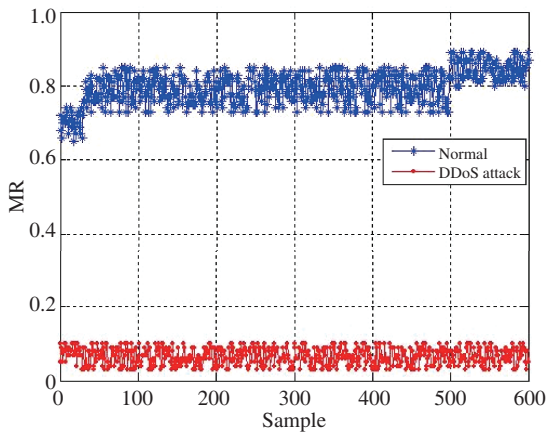

Figure 10 (Color online) MR of normal and attack traffic.

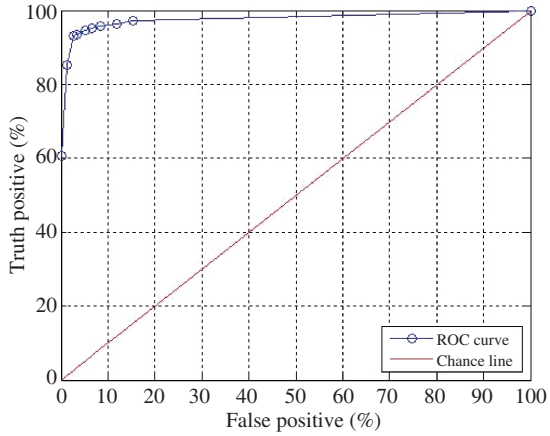

Figure 11 (Color online) ROC curve for the simple network.

normal period. Even if the sampling interval of the background traffic is 100 times as large as that of the attack traffic, PGS in the attack interval is at least 600, obviously larger than in the normal interval. Moreover, MR is at most 0.2, which is significantly smaller than in the normal interval. Consequently, there must have been a large number of abnormal flows entering the SDN network in the attack phase. Most of these flows have one packet, and their source IP addresses and port numbers are random and rarely repeated, resulting in a dramatic drop in the flow matching rate of the SDN switches.

The training sample traffic is used to train the catastrophe model for computing the model parameters. After being trained, the model can be utilized to detect the unlabeled traffic. The flow statistical analysis and normalizing (FSAN) module is called to obtain the values of the three detection features for each testing sample. Subsequently, each sample of the feature dataset is projected onto the u-v plane and taken as the input of the network state inference module. In this step, the side of the curve Bs on which the projection point lies determines whether a testing sample is in the DoS state. The ROC curve in Figure 11 shows the sensitivity and accuracy of CATH. An ROC curve is a plot with the false positive (FP) rate on the X-axis and the true positive (TP) rate on the Y-axis. The area under the curve reflects CATH's sensitivity. As can be seen, the curve is close to the Y-axis and the point (0, 1). This indicates that FP is low and the detection capacity is high.

(2) Experiment in the complex network. In this experiment, the background traffic was collected from the NDSC network center (100 Mbps). It consisted approximately 75% of TCP traffic, 10% of UDP traffic,

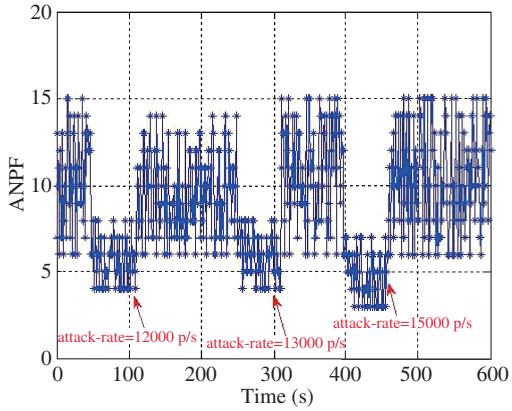

Figure 12 (Color online) ANPF of detected traffic when t= 0.5 s.

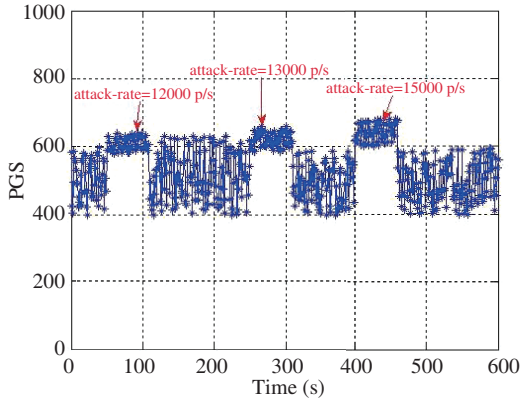

Figure 13 (Color online) PGS of detected traffic when t= 0.5 s.

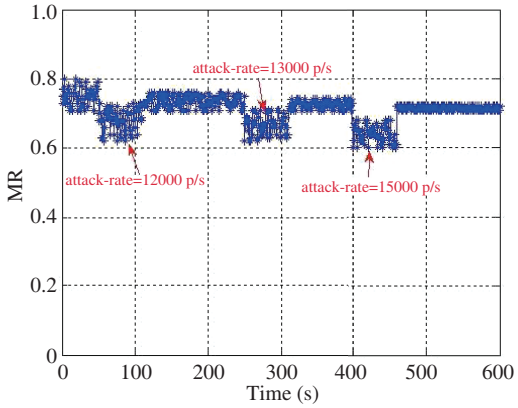

Figure 14 (Color online) MR of detected traffic when t= 0.5 s.

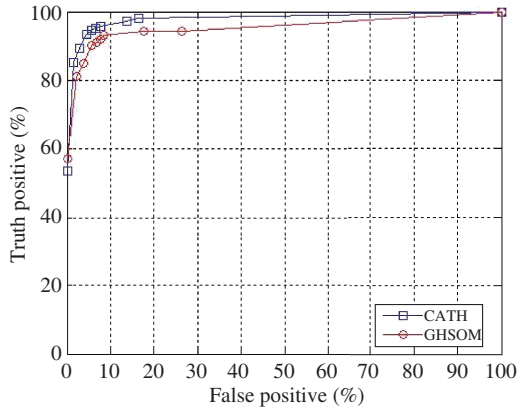

Figure 15 (Color online) ROC curves of CATH and GHOSM.

5% of ICMP traffic, and 10% of other traffic. The SDN-DoS attack traffic included TCP-SYN-Flood-based, UDP-Flood-based, and ICMP-Flood-based DoS attack traffic generated by Scapy. The variation of attack rates was achieved by carrying out attacks of 12000, 13000, and 15000 p/s. The background traffic and the attack traffic were sampled in a mixed manner as before.

However, the sampling period of attack traffic t was gradually extended from 0.1 to 0.5 s by increments of 0.1 s, and the sampling period of background traffic T was fixed to 2 s. Accordingly, five sample traffic sets were collected, each containing 1200 samples. Figures 12–14 show the trends of ANPF, PGS, and MR, respectively, for the detected traffic (mixed attack and background traffic) as the attack rate ranged from 12000 to 15000 p/s and t was 0.5 s. As the scale of the background traffic was significantly larger than that of the attack traffic, it was not easy to clearly determine by a single feature whether attack was carried. However, using the CATH method with multiple features, satisfactory detection results were obtained, as shown in Table 2. Obviously, higher attack rates yielded better detection results.

To illustrate the difference of CATH with other detection methods, such as those proposed in [13, 25, 26], the same background traffic and attack traffic were used as input, and the detection results were analyzed. In particular, a method based on growing hierarchical self-organizing maps (GHSOM), which was proposed in [13] and is widely used for anomaly detection based on machine learning, was compared with the proposed method. It applies the GHSOM algorithm to the analysis and detection of SDN-DoS attacks. Figure 15 shows the detection ROC curves of CATH and GHSOM in the above scenario. It is obvious that the detection accuracy of CATH is higher than that of the GHSOM-based method, particularly when FP is slightly higher than 3% and TP is up to 90%.

Table 2 Detection results at different attack rates

Attack-rate (p/s)	TP or FP	Detection result (%)				
		$t = 0.1$ s	$t = 0.2$ s	$t = 0.3$ s	$t = 0.4$ s	$t = 0.5$ s
12000	TP	91.4	92.1	92.8	93.5	93.9
	FP	4.89	4.87	4.73	4.23	3.97
13000	TP	91.6	92.4	93.5	94.1	94.7
	FP	5.02	4.13	4.21	4.78	4.80
15000	TP	93.7	94.2	94.9	95.1	96.2
	FP	3.76	3.51	3.41	3.64	2.97

It should be noted that in general, the network traffic has several phase characteristics. For example, the scale and the protocol type of the traffic significantly vary with time. Consequently, the normal value and the fluctuation range of detection features are different in different periods. To reduce the FP of CATH, it is necessary to dynamically update the catastrophe model. In particular, the first-in first-out (FIFO) strategy was adopted. The network administrator sets the update cycle ($T_{\text{update}} = \varepsilon$) of the catastrophe model according to the traffic characteristics of the local network. Once the update time t is up, the FSAN module is called to obtain training traffic samples collected during $(t-\varepsilon, t)$. As storing an overly large number of samples may reduce detection efficiency, new samples are added to the training set by the FIFO method. Then, the catastrophe potential function can be regenerated. The detection model can thereby be dynamically adjusted in real time and thus perceive new characteristics of the SDN-DoS attack detection features.

5.3 Computational complexity

The workflow of CATH for detecting SDN-DoS attacks consists of two stages: model training (estimating model parameters) and network state inference. Model training is an off-line process that can be performed in the idle time of the controller or in other network devices; thus, the time overhead of this process can be ignored. State inference can be divided into two procedures. The first is to perform a statistical analysis of the unlabelled network traffic, and the other is to infer the network state. The state inference process, as described in Section 3, requires only simple numerical comparisons; thus, its time overhead can be ignored as well. By contrast, the computational complexity of calculating the values of detection features is high. By the definition of detection features in Section 3, the computational complexity of calculating each eigenvalue is linearly related to the number of flows. That is, the computational complexity is $O(n)$. In the two experiments, by recording the time from the feature statistic process to the state inference process, it is seen that the computation overhead is indeed linearly related to the number of flows, and the required time is always in seconds.

6 Conclusion

With the rapid development of the Internet and the explosive growth of network users, the traditional network architecture faces many issues. To address network imperfections, several new network architectures have been put forward. Among them, the idea of separation between control and forwarding has attracted particular attention. A representative network structure is the SDN, which has been highly expected in the past few years. The centralized control mechanism and the open programming interface of SDNs not only increase the flexibility of network management and operation but also provide new and greater opportunities for network attacks. Currently, the most common and effective attack against SDNs is the DoS attack. As there are several differences between SDNs and traditional networks, particularly in network architecture and data forwarding technology, existing DoS attack detection methods have obvious deficiencies in terms of accuracy and efficiency.

In this study, an SDN-DoS attack detection method was proposed (called CATH) based several detection features for describing the network state catastrophe between the normal and the DoS state. Certain

statistic features of SDN flow information that have high potential to represent the state catastrophe of the SDN are first selected as detection features. Subsequently, following an idea from catastrophe theory, and according to the number of state variables and control variables selected from the detection features, the cusp model is used to establish a catastrophe equilibrium surface for SDN states. After being trained, the cusp catastrophe model can be used to infer whether the network is in the attack state, thus realizing real-time SDN-DoS attack detection. The experimental results showed that the CATH method can timely and effectively perceive SDN-DoS attacks, not only in simple networks but also in larger enterprise networks. As the comprehensiveness of the sample set has an important influence on the accuracy of the proposed method, the focus in future work will be on obtaining more complete and representative training sample sets. Additionally, it is difficult for network administrators to set the update frequency of the cusp detection model; thus, it is necessary to develop a method for obtaining phase characteristics of the local network traffic. Certainly, the prevention of SDN-DoS attacks deserves further study.

Acknowledgements This work was supported by National Natural Science Foundation of China (Grant Nos. 61402525, 61402526, 61502528), Key Scientific Research Projects of Henan Province Education Department (Grant No. 18A520004), and Henan Province Science and Technology Projects (Grant No. 182102310925). We also thank Zhong HUA for interesting and helpful discussion on the ideas presented here.

References

1 Nunes B A A, Mendonca M, Nguyen X N, et al. A survey of software-defined networking: past, present, and future of programmable networks. IEEE Commun Surv Tut, 2014, 16: 1617–1634
2 Kreutz D, Ramos F, Verissimo P, et al. Software-defined networking: a comprehensive survey. Proc IEEE, 2015, 103: 14–76
3 Kreutz D, Ramos F, Verissimo P. Towards secure and dependable software-defined networks. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hong Kong, 2013. 55–60
4 Shin S, Gu G F. Attacking software-defined networks: a first feasibility study. In: Proceedings of the 2nd ACM SIGCOMM Workshop on Hot Topics in Software Defined Networking, Hong Kong, 2013. 165–166
5 Kandoi R, Antikainen M. Denial-of-service attacks in OpenFlow SDN networks. In: Proceedings of IFIP/IEEE the 1st International Workshop on Security for Emerging Distributed Network Technologies (DISSECT), Ottawa, 2015. 1323–1326
6 McKeown N, Anderson T, Balakrishnan H, et al. OpenFlow: enabling innovation in campus networks. ACM SIGCOMM Comp Commun Rev, 2008, 38: 69–74
7 Yan Q, Yu F R, Gong Q, et al. Software-defined networking (SDN) and distributed denial of service (DDoS) attacks in cloud computing environments: a survey, some research issues, and challenges. IEEE Commun Surv Tut, 2016, 18: 602–622
8 Shin S, Yegneswaran V, Porras P, et al. Avant-guard: scalable and vigilant switch flow management in software-defined networks. In: Proceedings of ACM SIGSAC Conference on Computer & Communications Security, Berlin, 2013. 413–424
9 Wang H P, Xu L, Gu G F. FloodGuard: a DoS attack prevention extension in software-defined networks. In: Proceedings of the 45th Annual IEEE/IFIP International Conference on Dependable Systems and Networks (DSN), 2015
10 Giotis K, Argyropoulos C, Androulidakis G, et al. Combining OpenFlow and sFlow for an effective and scalable anomaly detection and mitigation mechanism on SDN environments. Comput Netw, 2014, 62: 122–136
11 Mousavi S M, St-Hilaire M. Early detection of DDoS attacks against SDN controllers. In: Proceedings of 2015 International Conference on Computing, Networking and Communications, Garden Grove, 2015. 77–81
12 Braga R, Mota E, Passito A. Lightweight DDoS flooding attack detection using NOX/OpenFlow. In: Proceedings of the 35th Annual IEEE Conference on Local Computer Networks, Denver, 2010. 408–415
13 Yao L Y, Dong P, Zhang H K. Distributed denial of service attack detection based on object character in software defined network. Chin J Electron Inform Tech, 2017, 39: 381–388
14 Porras P, Shin S, Yegneswaran V, et al. A security enforcement kernel for OpenFlow networks. In: Proceedings of SIGGCOMM 1st Workshop on HotSDN. New York: ACM, 2012. 121–126
15 Shin S, Porras P, Yegneswaran V, et al. Fresco: modular composable security services for software-defined networks. In: Proceedings of NDSS, 2013. 1–15
16 Yao G, Bi J, Xiao P Y. Source address validation solution with openflow nox architecture. In: Proceedings of the 19th IEEE International Conference on Network Protocols, Vancouver, 2011. 7–12
17 Fayaz S K, Tobioka Y, Sekar V, et al. Bohatei: flexible and elastic DDoS defense. In: Proceedings of the 24th USENIX Conference on Security Symposium, Washington, 2015. 817–832
18 Mirkovic J, Reiher P. A taxonomy of DDoS attack and DDoS defense mechanisms. Comput Commun Rev, 2004, 34: 39–53
19 Zargar S T, Joshi J, Tipper D. A survey of defense mechanisms against distributed denial of service (DDoS) flooding attacks. IEEE Commun Surv Tut, 2013, 15: 2046–2069

20 Huang Y, Geng X J, Whinston A B. Defeating DDoS attacks by fixing the incentive chain. ACM Trans Inter Tech, 2007, 7: 5
21 Thom R. Structure stability, catastrophe theory, and applied mathematics. SIAM Rev, 1977, 19: 189–201
22 Stamovlasis D. Catastrophe theory: methodology, epistemology, and applications in learning science. In: Complex Dynamical Systems in Education. Berlin: Springer, 2016. 141–175
23 Guo R, Yin H, Wang D, et al. Research on the active DDoS filtering algorithm based on IP flow. In: Proceedings of IEEE 5th International Conference on Natural Computation, 2009. 628–632
24 Gude N, Koponen T, Pettit J, et al. NOX: towards an operating system for networks. Comput Commun Rev, 2008, 38: 105–110
25 Rauber A, Merkl D, Dittenbach M. The growing hierarchical self-organizing map: exploratory analysis of high-dimensional data. IEEE Trans Neural Netw, 2002, 13: 1331–1341
26 Ashraf J, Latif S. Handling intrusion and DDoS attacks in software defined networks using machine learning techniques. In: Proceedings of IEEE 2014 National Software Engineering Conference (NSEC), Event-Karachi, 2014. 55–60