DOI: 10.19729/j.cnki.1673-5188.2018.04.002
http://kns.cnki.net/kcms/detail/34.1294.TN.20181015.1722.002.html, published online October 15, 2018

Survey of Attacks and Countermeasures for SDN

BAI Jiasong [1,2,3], ZHANG Menghao [1,2,3], and BI Jun [1,2,3]
(1. Institute for Network Sciences and Cyberspace, Tsinghua University, Beijing 100084, China;
2. Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China;
3. Beijing National Research Center for Information Science and Technology (BNRist), Tsinghua University, Beijing 100084, China)

Abstract

Software defined networking (SDN) has attracted significant attention from both academia and industry by its ability to reconfigure network devices with logically centralized applications. However, some critical security issues have also been introduced along with the benefits, which put an obstruction to the deployment of SDN. One root cause of these issues lies in the limited resources and capability of devices involved in the SDN architecture, especially the hardware switches lied in the data plane. In this paper, we analyze the vulnerability of SDN and present two kinds of SDN-targeted attacks: 1) data-to-control plane saturation attack which exhausts resources of all SDN components, including control plane, data plane, and the in-between downlink channel and 2) control plane reflection attack which only attacks the data plane and gets conducted in a more efficient and hidden way. Finally, we propose the corresponding defense frameworks to mitigate such attacks.

Keywords

SDN; indirect/direct data plane event; data-to-control plane saturation attack; control plane reflection attack

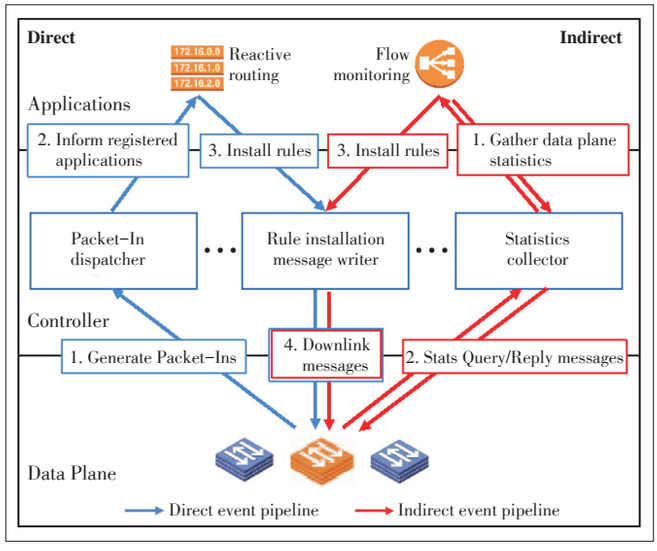
1 Introduction

Software defined networking (SDN) has enabled flexible and dynamic network functionalities with a novel programming paradigm. By decoupling the control plane from the data plane, control logics of different network functionalities could be implemented on top of the logically centralized controller as "applications". Typical SDN applications are implemented as event-driven programs, which receive information directly or indirectly from switches and distribute the processing decisions of packets to switches accordingly. These applications enable SDN to adapt to the data plane dynamics quickly and make the responses according to the application policies timely. A wide range of network functionalities are implemented in this way, allowing SDN-enabled switches [1] to behave as firewall [2], load balancing [3], L2/L3 routing, and so on.

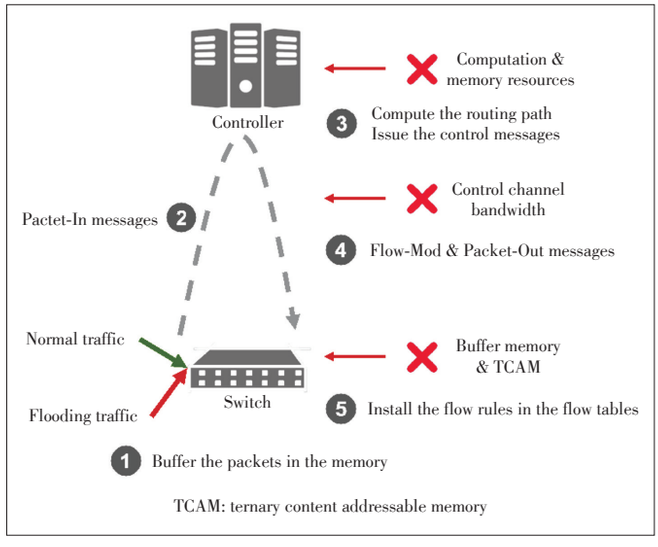
While the decoupling paradigm has enabled unprecedented programmability in networks, it also becomes the vulnerability of SDN infrastructure. The typical SDN infrastructure consists of three major components: the control plane, the data plane, and a control channel, where the two planes can communicate through standard protocols. To express the logics of control applications, control messages are generated in both the two planes and transferred through the channel. By triggering numerous control messages in a short time, attackers can paralyze the SDN infrastructure by exhausting the available resources of all three components. In particular, the control message processing capability on switches proves to be the bottleneck of the infrastructure, which is constrained by the wimpy central processing units (CPUs) , limited ternary content-addressable memory (TCAM) [4], [5] update rate and flow table capacity due to financial and power consumption reasons. These limitations have slowed down network updates and hurt network visibility, which further constrains the control plane applications with dynamic policies significantly [6].

The applications enable a network to dynamically adjust network configurations based on certain data plane events as illustrated in **Fig. 1**. These events can be categorized into the following two types: direct data plane events (e.g., Packet-In messages) and indirect data plane events (e.g., Statistics Query/Reply messages). In the first case, the controller installs a default table-miss flow rule on the switch. Arriving packets which fail to match any flow rule are forwarded to the control plane for further processing. In the second case, the controller installs a counting flow rule on the switch to record the statistics of arriving packets and periodically polls the flow counter values. A large number of control plane applications combine these two kinds of events to compose complicated network functions.

This work was supported in part by the National Key R&D Program of China under Grant No. 2017YFB0801701, the National Science Foundation of China under Grant No. 61472213 and CERNET Innovation Project (NGII20160123).



▲Figure 1. Architecture and event pipelines of current software-defined networking.



▲Figure 2. Adversary model of the data-to-control saturation attack.

From our previous study, we find that flow rule update messages from the SDN control plane will be triggered by both kinds of events, which can be exploited by an intentional attacker. In this article, we present two kinds of attacks, i.e., the data-to-control saturation attack [7], a dedicated Denial-of-Service (DoS) attack against SDN essentially, and the control plane reflection attack [8], which can be further categorized into the table-miss striking attack and counter manipulation attack by the type of applied events. Furthermore, we propose the defense frameworks to mitigate these two attacks. In the following, we illustrate the details of two types of attacks in Sections 2 and 3, present the corresponding defense frameworks in Sections 4 and 5, and conclude this article and make some discussion in Section 6.

2 Data-to-Control Saturation Attacks

Intuitively, an attacker could commit the data-to-control saturation attack by producing a large number of short-flows by controlling a number of zombie hosts in an SDN-enabled network. The attack traffic is mixed with benign traffic, making it difficult to be identified. With the reactive routing and fine-grained flow control mechanism taken by the existing mainstream SDN controllers, the unmatched packets in the data plane would be delivered to the controller directly and processed by the corresponding applications. As a result, the data plane, the control channel and the control plane would quickly suffer from the attack, and soon the SDN system could not provide any service for benign traffic.

We start from a simplified motivating scenario to illustrate how an adversary attacks the SDN infrastructure. As depicted in **Fig. 2**, when a new packet arrives at a switch where there is no matching flow entry in the local flow tables, the switch will store the packet in its buffer memory and send a Packet-In message to the controller. The message only contains the packet header if the buffer memory is not full, but will contain the whole packet when the buffer memory is full. After the controller receives the message, it computes the route and takes the corresponding actions on the switches through control messages including Flow-Mod and Packet-Out. Then the switches parse the packets and install the flow rules in the capacity-limited flow tables. The attacker can exploit the vulnerability of this reactive packet processing mechanism by flooding malicious packets to the switches. The header fields of these packets are filled with deliberately forged values that it is almost impossible for them to be matched by any existing flow entries in the switches. After that, numerous table-misses are triggered, and a large number of packet-in messages are flooded to the controller, making the entire SDN system suffer from resource exhaustion. In this adversary model, all three levels of SDN resources are compromised.

3 Control Plane Reflection Attacks

Compared with saturation attacks, control plane reflection attacks are much hidden and sophisticated. It does not target at the controller, nor the end host, but it utilizes the limited processing capability of downlink messages in the SDN-enabled hardware switches and easily gain much more prominent effects than saturation attacks.

A general procedure of control plane reflection attacks consists of two phases, i.e., the probing phase and triggering phase. During the probing phase, an attacker uses several kinds of probe packets to learn the conditions that application adopts to issue new flow rule update messages. Upon the information obtained, the attacker can carefully craft the patterns of attack packet stream to trigger numerous flow rule update mes-

sages in a short interval to paralyze the hardware switches.

3.1 Table-Miss Striking Attacks

The table-miss striking attack is an enhanced attack vector from the saturation attack. Instead of leveraging a random packet generation method to commit the attack, a striking attack adopts a more accurate and cost-efficient manner by utilizing probing and triggering phases.

The probing phase is to learn the confidential information of the control plane to guide the patterns of attack packet streams. The attacker could first probe the use of direct data plane events by using various low-rate probing packets with deliberately faked headers. By sending these probing packets and observing the response accordingly, the round trip time (RTT) could be obtained. If the first packet has a longer RTT, we can conclude that it is directed to the controller while the others are forwarded directly to the data plane. This indicates that the specific packet header matches no flow rule in the switch. Then the attacker could change one of the header fields with the variable-controlling approach. Within limited trials (42 in the latest OpenFlow specification), the attacker was able to determine which header fields were sensitive to the controller. Then the attacker could deliberately craft attack stream based on probed grains to trigger the expensive flow rule update operations.

3.2 Counter Manipulation Attacks

The counter manipulation attack is based on indirect data plane events and much more sophisticated compared with abovementioned attacks. In order to accurately infer the usage of indirect data plane events, three types of packets are required, i.e., timing probing packets, test packets and data plane streams.

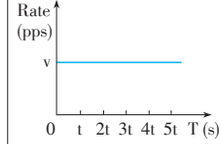
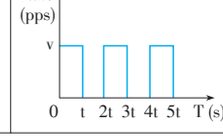
Timing probing packets are used to measure the work load of software agent of a switch, inspired by time pings in [9]. Three properties should be satisfied. First, they should go to the control plane by hitting the table-miss flow rule in the switch, and trigger the operations of corresponding applications. Second, each of them must evoke a response from the network to compute the RTT. Third, they should be sent in an extremely low rate (10 packets per second (pps) is enough) and put as low loads as possible to the switch software agent. There are many options for timing probing packets, e.g., Address Resolution Protocol (ARP) request/reply, Internet Control Message Protocol (ICMP) request/reply.

Test packets are used to strengthen the effect of timing probing packets by adding extra loads to the software agent of the switch. We consider test packets with a random destination IP address and the broadcast destination Media Access Control (MAC) address is an ideal choice. By hitting the table-miss entry, each of them would be directed to the controller. Then the SDN controller will issue Packet-Out message to forward the test packet directly. As a result, the aim of burdening switch software agent is achieved.

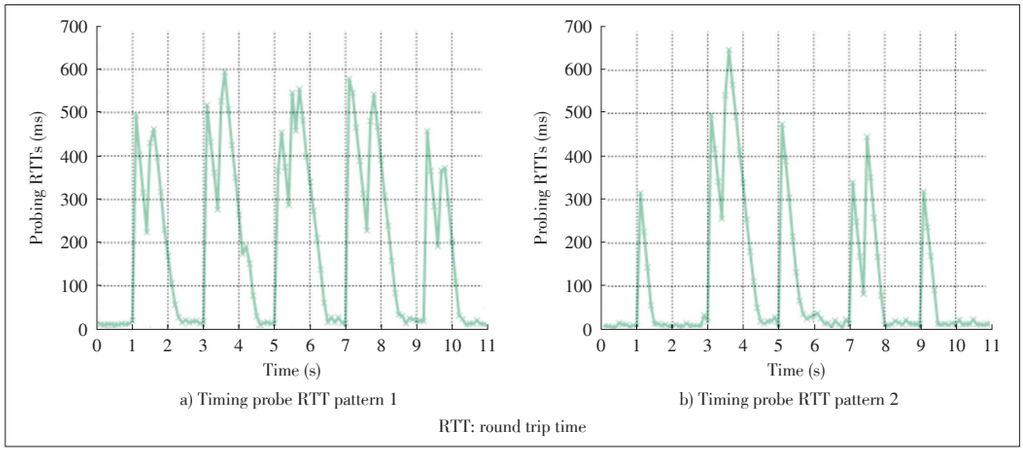
A data plane stream is a series of templates, which should go directly through the data plane to obtain more advanced information such as the specific conditions for indirect event-driven applications. We provide two templates here, as shown in **Fig. 3**. The first template has a steady rate v and packet size p, which is mainly used to probe volume-based statistic calculation and control method. The second has a rate distribution like a jump function, where three variables (v, t, p) determine the shapes of this template as well as the size of each packet, which is often used to probe the rate-based strategy.

The insight of the probing phase of counter manipulation attacks lies in that different downlink messages have diverse expenses for the downlink channel. Among the interaction approaches between the applications and the data plane, there are mainly three types of downlink messages, i.e., Flow-Mod, Statistics Query, and Packet-Out. Flow-Mod is the most expensive one, Statistics Query comes at the second and Packet-Out is rather lightweight. The latencies of timing probing packets will vary when the switch encounters different message types. Thus, the attacker could learn the type of message issued by the control plane. As for indirect data plane events, the statistic queries are usually conducted periodically by the applications. As a result, each of these queries would incur a small rise for the RTTs of timing probing packets. If a subsequent Flow-Mod is issued by the controller, there would be a double-peak. Based on the double-peak phenomenon, the attacker could even infer what statistic calculation methods the application is taking, such as volume-based or rate-based. With several trails of two templates above and the variations of v and p in a binary search approach, the attacker could quickly obtain the concrete conditions (volume/rate values, packet number/byte-based) that trigger the expensive downlink messages. The confidential information, such as the query period and exact conditions, helps the attacker permute the packet interval and packet size of each flow. By initiating a large number of flows, Flow-Mod of equal number would be triggered every period, making the hardware switch suffer extremely.

We use a simplified example (**Fig. 4**) to illustrate the attack.

Template Name	Coordinate Axis	Variables
Data plane stream with steady rate		v, p
Data plane stream with 0–1 rate		v, t, p

▲**Figure 3. Templates for a data plane stream.**



◀Figure 4.
Timing-based patterns for the counter manipulation attack.

If an attacker obtains a series of successive double-peak phenomenon (Fig. 4a) with the input of data plane stream template 1, where v is a big value, and obtains a series of intermittent double-peak phenomenon (Fig. 4b), where v is also a significant value. The attacker could determine that packet number volume-based statistic calculation approach is sensitive to stream with a high pps. With the variations of v and p, the critical value of volume can be inferred to help conduct the attack.

4 FloodShield: Defending Data-to-Control Plane Saturation Attacks

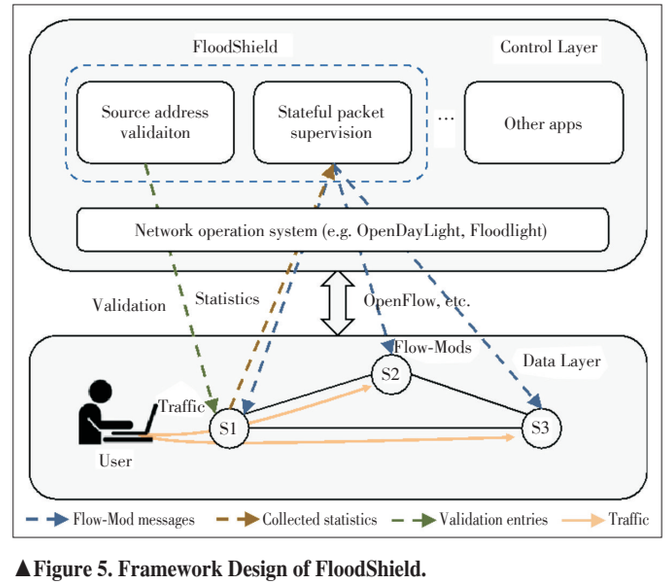
Floodshield [7] is a SDN defense framework against the data-to-control saturation attacks by combining two modules, i.e., source address validation and stateful packet supervision. The former validates the source addresses of the incoming traffic and filters the forged packets directly in the data plane, since attackers tend to commit attacks with a forged source address to hide the locations of attack sources. Based on it, the last module monitors the packet states of each real address and performs network service differentiation according to the evaluation scores and network resource usage.

As depicted in **Fig. 5**, the source address validation module works when a host connects to the SDN-enabled network. By snooping the address assignment mechanism procedure, the module maintains a global Binding Table at the controller to record the mapping between end hosts and their IP addresses. Based on the table, the module then takes advantage of the multi-table pipeline of OpenFlow to install filter rules in table 0 and install normal flow rules in the following tables. Packets with forged IP addresses are dropped in table 0 while trusted packets are directly forwarded to the non-filter flow tables.

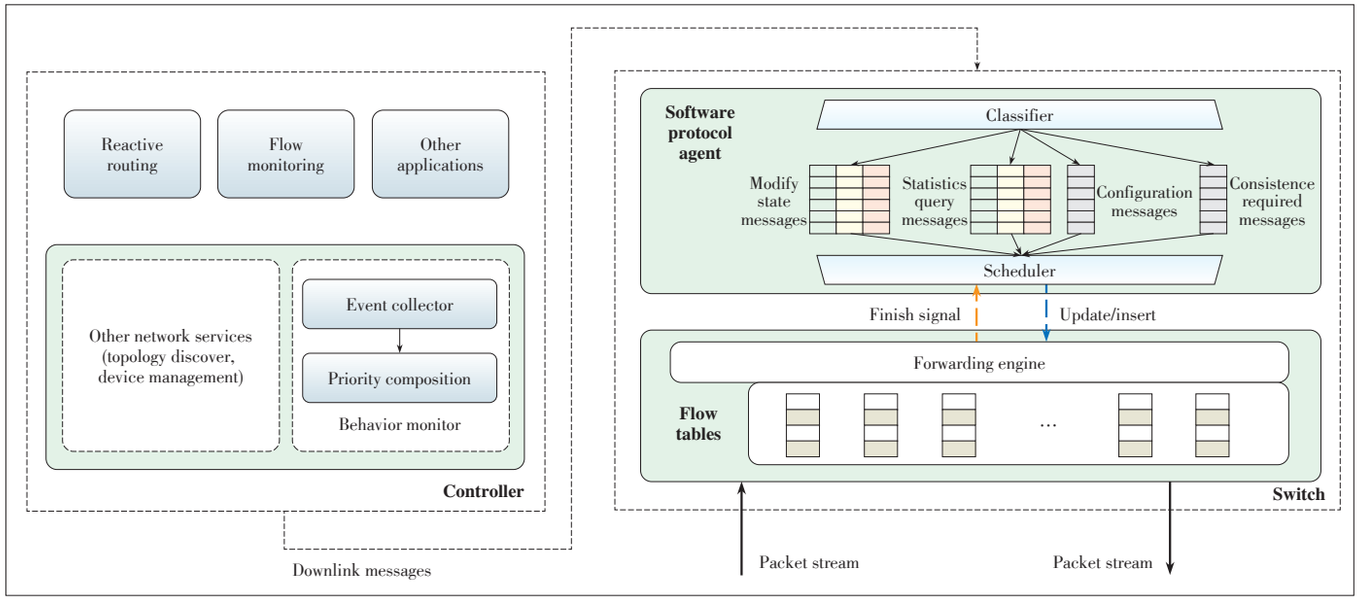
Since packets with real source addresses could also be harnessed to conduct attacks, a stateful packet supervision module is introduced to distinguish flows by traffic features and achieve differentiated services for different user dynamically. The module takes packet-in rate and average flow length as two metrics to evaluate user behavior. Users are divided into three levels according to their evaluation scores and allocated with different priorities. Flows with a high priority are processed as usual while those with a lower priority are limited on the rate or even dropped.

5 SWGuard: Defending Control Plane Reflection Attacks

The basic idea of SWGuard [8] is to discriminate good from evil, and prioritize downlink messages with discrimination results. SWGuard introduces a multi-queue scheduling strategy to achieve different latency for different downlink messages. The scheduling strategy is based on the statistics of downlink messages during the last period, which takes both fairness and efficiency into consideration. When the downlink channel is becoming congested, the malicious downlink messages are inclined to be put into a low-priority scheduling queue and the requirements of good messages are more likely to be satisfied. As shown in **Fig. 6**, SWGuard mainly redesigns two compo-



▲Figure 5. Framework Design of FloodShield.



▲Figure 6. Framework Design of SWGuard.

nents of SDN architecture. On the switch side, it changes the existing software protocol agent to multi-queue based structures. On the controller side, it adds a Behavior Monitor module as a basic service which assigns different priorities to different messages dynamically.

SWGuard redesigns the software protocol agent of the existing switch to prioritize the downlink messages. Since different types of downlink messages have diverse requirements, SWGuard summarizes the downlink messages into four categories: 1) Modify State Messages, 2) Statistic Query Messages, 3) Configuration Messages, and 4) Consistency Required Messages. It also designs a Classifier to classify the downlink messages into different queues accordingly. The first two types are related to behaviors of hosts and applications which are sensitive to latency and order, so a multi-queue is allocated for each. The latter two types inherit from the original single queue. With messages in the queues, a Scheduler is designed to dequeue the messages with a time-based scheduling algorithm. For queues with the highest priority are dequeued immediately, messages are dequeued immediately as they arrive. However, for queues with lower priority, different time interval is added to messages before dequeued.

To distinguish different downlink messages with different priorities, SWGuard proposes the novel abstraction of Host-Application Pair (HAP) and use it as the granularity for monitoring and statistics. Packets are recorded for each application of each user. Assuming there are K applications in the control plane, and N hosts in the data plane, packets should be categorized into $K \times N$ groups. SWGuard is designed as attack-driven. When the number of downlink messages in a period is less than a threshold, all packets are allocated with the highest priority. When the reflection attacks are detected, the SWGuard starts to calculate the penalty coefficient for each HAP by comparing their required resources with their real resource occupation. According to the coefficient, downlink messages are enqueued into queues with different priorities. Besides, multi-queues based software protocol agent may violate the consistency of some messages, which need to be sent in a particular order for correctness reasons. To address this issue, a coordination mechanism between the Behavior Monitor and Classifier in software protocol agent is designed.

6 Conclusions

While SDN has offered new opportunities to network automation and innovations, it has also introduced new security concerns. Securing the network infrastructure is crucial to the promotion and adoption of SDN. In this article, we review two SDN-targeted attacks, data-to-control saturation attacks, and control plane reflection attacks, along with the corresponding defense frameworks, FloodShield and SWGuard. The two attacks are both targeted at limited resources of SDN infrastructure, especially resources and limited processing capability of the data plane. Since hardware switching systems share many common designs like TCAM-based flow table, the SDN-targeted attacks also provide new perspectives to the security of other emerging architecture, e.g. the programmable data plane [10].

References

[1] N. McKeown, T. Anderson, H. Balakrishnan, et al., "OpenFlow: enabling innovation in campus networks," *ACM SIGCOMM Computer Communication Review*, vol. 38, no. 2, pp. 69–74, 2008. doi: 10.1145/1355734.1355746.
[2] A. K. Nayak, A. Reimers, N. Feamster, and R. Clark, "Resonance: dynamic access control for enterprise networks," in *Proc. 1st ACM Workshop on Research on Enterprise Networking*, Barcelona, Spain, 2009, pp. 11–18. doi: 10.1145/1592681.1592684.

[3] R. Miao, H. Zeng, C. Kim, J. Lee, and M. Yu, "Silkroad: making stateful layer-4 load balancing fast and cheap using switching ASICS," in *Proc. Conference of the ACM Special Interest Group on Data Communication*, Los Angeles, USA, 2017, pp. 15–28. doi: 10.1145/3098822.3098824.
[4] A. R. Curtis, J. C. Mogul, J. Tourrilhes, et al., "Devoflow: scaling flow management for high-performance networks," *ACM SIGCOMM Computer Communication Review*, vol. 41, no. 4, pp. 254–265, 2011. doi: 10.1145/2043164.2018466.
[5] A. Wang, Y. Guo, F. Hao, T. Lakshman, and S. Chen, "Scotch: elastically scaling up SDN control- plane using vswitch based overlay," in Proc. *10th ACM International on Conference on Emerging Networking Experiments and Technologies*, Sydney, Australia, 2014, pp. 403–414. doi: 10.1145/2674005.2675002.
[6] X. Jin, H. H. Liu, R. Gandhi, et al., "Dynamic scheduling of network updates," in *ACM SIGCOMM Computer Communication Review*, Chicago, USA, 2014, pp. 539–550. doi: 10.1145/2619239.2626307.
[7] M. Zhang, J. Bi, J. S. Bai, et al., "FloodShield: securing the SDN infrastructure against denial-of-service attacks," in *17th IEEE International Conference on Trust, Security and Privacy in Computing and Communications (TrustComm18)*, New York, USA, 2018, pp. 687–698. DOI:10.1109/TrustCom/BigDataSE.2018.00101.
[8] M. H. Zhang, G. Y. Li, L. Xu, et al., "Control plane reflection attacks in SDNs: new attacks and countermeasures," in *21st International Symposium on Research in Attacks, Intrusions and Defenses (RAID18)*, Heraklion, Greece, 2018, pp. 161–183.
[9] J. Sonchack, A. Dubey, A. J. Aviv, J. M. Smith, and E. Keller, "Timing-based reconnaissance and defense in software-defined networks," in *Proc. 32nd Annual Conference on Computer Security Applications*, Los Angeles, USA, 2016, pp. 89–100. doi: 10.1145/2991079.2991081.
[10] P. Bosshart, D. Daly, G. Gibb, et al., "P4: programming protocol-independent packet processors," *ACM SIGCOMM Computer Communication Review*, vol. 44, no. 3, pp. 87–95, 2014. doi: 10.1145/2656877.2656890.

Manuscript received: 2018-06-19

Biographies

BAI Jiasong (bjs17@mails.tsinghua.edu.cn) received his B.S. degree from Department of Computer Science and Technology, Tsinghua University, China in 2017. He is currently a master student in Department of Computer Science and Technology, Tsinghua University. His research interests include SDN, NFV and programmable data plane.

ZHANG Menghao (zhangmh16@mails.tsinghua.edu.cn) received his B.S. degree from Department of Computer Science and Technology, Tsinghua University, China in 2016. He is currently a Ph.D. student in Department of Computer Science and Technology, Tsinghua University. His research interests include the availability and security of SDN and NFV.

BI Jun (junbi@tsinghua.edu.cn) received his B.S., C.S., and Ph.D. degrees from Department of Computer Science, Tsinghua University, China. He is currently a Changjiang Scholar Distinguished Professor and the Director of Network Architecture Research Division, Institute for Network Sciences and Cyberspace, Tsinghua University. He is also the Director of the Future Network Theory and Application Research Division at Beijing National Research Center for Information Science and Technology. His current research interests include Internet architecture, SDN/NFV, and network security. He successfully led tens of research projects, published over 200 research papers and 20 Internet RFCs and drafts, and also holds 30 innovation patents. He received the National Science and Technology Advancement Prizes, the IEEE ICCCN Outstanding Leadership Award, and Best Paper awards. He is the co-chair of the AsiaFI Steering Group and the Chair of the China SDN Experts Committee. He served as the TPC co-chairs of a number of Future Internet related conferences or workshops/tracks at INFOCOM and ICNP. He served on the Organization Committee or Technical Program Committees of SIGCOMM, and ICNP, INFOCOM, CoNext, and SOSR. He is Distinguished Member of the China Computer Federation.