REVIEW PAPER

A Survey: Typical Security Issues of Software-Defined Networking

Yifan Liu[1], Bo Zhao[1,*], Pengyuan Zhao[1,2], Peiru Fan[1], Hui Liu[1]

[1] Key Laboratory of Aerospace Information Security and Trusted Computing, Ministry of Education, School of Cyber Science and Engineering, Wuhan University, Wuhan 430072, China
[2] School of Cyber Security and Computer, Hebei University, Baoding 071002, China
* The corresponding author, email: zhaobo@whu.edu.cn

Abstract: Software-Defined Networking (SDN) has been a hot topic for future network development, which implements the different layers of control plane and data plane respectively. Despite providing high openness and programmability, the "three-layer two-interface" architecture of SDN changes the traditional network and increases the network attack nodes, which results in new security issues. In this paper, we firstly introduced the background, architecture and working process of SDN. Secondly, we summarized and analyzed the typical security issues from north to south: application layer, northbound interface, control layer, southbound interface and data layer. Another contribution is to review and analyze the existing solutions and latest research progress of each layer, mainly including: authorized authentication module, application isolation, DoS/DDoS defense, multi-controller deployment and flow rule consistency detection. Finally, a conclusion about the future works of SDN security and an idealized global security architecture is proposed.

Keywords: software-defined networking; network security; global security; security threat

I. INTRODUCTION

In the traditional network, the control plane and the data plane of devices are encapsulated in a box and tightly coupled with each other. Due to the interdependence and poor programmability of the devices, we cannot support new requirements quickly [1]. If the business needs to deploy a new network strategy, the configuration of the corresponding network devices (routers, switches, firewalls, etc.) will be re-edited, which is a very tedious task [2]. In the rapidly changing environment of Internet and mobile terminal network, high performance and high stability cannot meet the needs of people, but flexibility and agility are more critical.

Software-Defined Networking (SDN) provides a more innovative way to design, establish, and manage communication networks. It is a new type of network architecture proposed by the Stanford University Clean Slate Research Group [3]. SDN implements the different layers of control plane and data plane respectively; centralized control and network open programmability. In this way, SDN solves the problem of interdependence

Received: Jan. 22, 2019
Accepted: Apr. 09, 2019
Editor: Yong Cui

This survey focuses on current advances in SDN security research.

between control plane and data plane. In addition, SDN provides more programming capabilities for the network. After SDN deploying, it is not necessary to configure the routers of each node repeatedly in the network. The devices in the network are connected automatically, and users only need to define simple network rules on the controller before using [4]. The controller will play the most important role in the network, responsible for collecting and managing all network status information, which provides the possibility of network automatic management. Users can modify the built-in protocol of the router by programming to achieve better data exchange performance, and enhance the flexibility to support the rapid growth of network business needs.

SDN not only becomes a very active research field in academia, but also has successfully applied in the information and communication industry. Google deploys the B4 network and achieves an unprecedented 95% network utilization [5]. Microsoft is trying to use OpenDaylight controller to enhance the video experience of Skype users. GAP and Viptela collaborate to apply SD-WAN technology to 1,350 retail stores across the US. According to the data released by Research and Markets in Irish, SDN's turnover will grow at an annual rate of 42.3% over the next six years. By 2023, the market profits of SDN in the global operator market are expected to reach $9.5 billion [6]. SDN is called by MIT as "One of the Top Ten Innovative Technologies to Change the World"[7].

With the constant deployment and application, the defects of SDN in dealing with security threats are gradually exposed. Its security issues are closely related to its own characteristics. According to our research, the following aspects cause the SDN to be vulnerable to attack:

(1) Centralized control and un-enough security protection mechanism, which make SDN controller become an external malicious attack target.

(2) Complex interaction of various application programs. There is a high coupling between applications, which causes flow rules to conflict easily.

(3) Lack of adequate application authorization and authentication mechanism, makes it vulnerable to malicious application attacks.

(4) Not enough security and encryption measures in the communication process between control layer and data layer. Flow rules are easy to suffer malicious tampering during the process of publishing.

In general, SDN lacks of sufficient multi-level protection mechanism. SDN security research is still in the infancy and attention to uniform standards is insufficient [8]. In the past few years, researchers have focused their attention on the development of SDN functions, such as SDN resource scheduling and rule delivery. They did not attach importance to the SDN security. A complete SDN security protection strategy has not been proposed yet.

In this paper, the typical security problems of SDN are comprehensively reviewed, from the basic technology to the security problems of each layer in SDN. The main contributions of this paper are as follows:

- A review of SDN architecture and work process; an introduction of the new security threats due to SDN's characteristics.
- Analysis of SDN typical security issues from north to south; a summary of attacking objects and the cause of the problem.
- A review of existing solutions and make a summary table, which analyzes the latest research progress of each layer and the defects of each method, and then summarize these into a table.
- Conclude the major research field that can be considered in the future, and propose an idealized global security architecture.

The remainder of the paper is organized as follows. Section 2 introduces the typical architecture of SDN and analyzes its security threats. Section 3 summarizes the typical security issues at each layer of SDN. Section 4 compares the existing solutions and analyzes the defects of each method. Section 5 reviews the possible future research directions and proposes an idealized global security solution.

And conclusion of this survey is given in Section 6.

II. Architecture and Threat Analysis

2.1 Architecture of SDN

Compared with the traditional network, SDN proposes the control and forwarding separation strategy to decouple the data plane from the control plane. The control plane implements the logical centralized control by the controller and unified management of the distributed network. The data plane provides a programmable interface, which can provide users a complete set of APIs. Users can use the APIs to program, configure, control, and manage the network on the controller. The "three-tier, two-interface" architecture is shown in figure 1, including application layer, control layer, data layer and northbound-southbound interface.

The application layer includes various applications that can implement network functions. The application programs call the SDN controller layer through the northbound interface. In this way, users can implement configuration, management, and control of the data plane devices. The control layer manages the infrastructure of the network and can select multiple controllers flexibly as needed. Controller is the brain of the entire SDN network, responsible for collecting and managing all network status information. The control layer manages the data layer through the southbound interface, publishes behavior commands such as forwarding or discarding of data packets. The data layer includes basic installations such as software-based and hardware-based devices, performs specific data processing by receiving instructions from the control layer. The data layer collects information such as network configuration and runtime status and feeds it back to the controller.

2.2 Threat analysis

SDN implements the different layers of control plane and data plane respectively. The control plane formulates the flow rules and the data plane only responsible for forwarding data packets according to the flow rules. At the same time, the controller can obtain the status of each device through the southbound interface to establish a global view of the network. The separation of SDN architectures greatly enhances business flexibility, but also introduces new security threats. The new security threats introduced by SDN are mainly reflected in the following two aspects:

(1) Selective safety device permission

SDN is a flow rule driven network, the physical security device does not have discretion. Whether and when the data packets transit the security device is determined by the flow rule. In SDN, an attacker can bypass the security devices, resulting in the failure of pre-deployment security measures.

(2) Automatic global view obtaining

In SDN, the controller, as the command and control center of the entire network, can establish a global view of the network. It can obtain various status information of the network in real time. The security posture of the network can be easily obtained from the controller. Therefore, attackers can obtain the global view of the network directly in this convenient way. With the network situation, the attacker will wait for an opportunity to launch a large-scale attack.

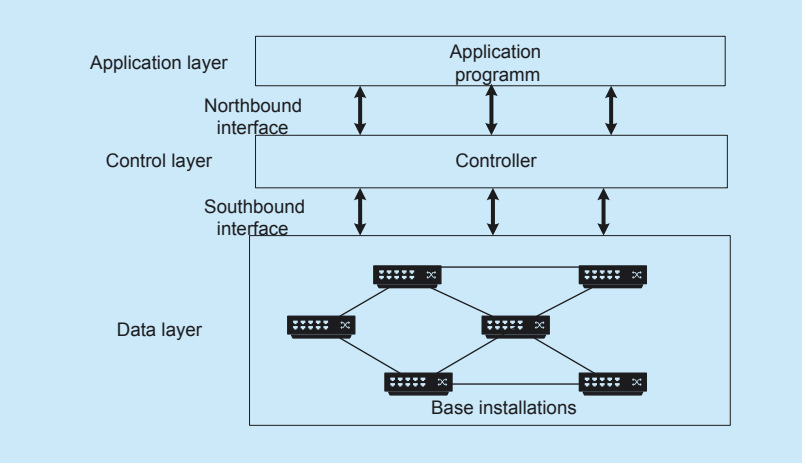


Fig. 1. *Architecture of SDN*

III. Challenges in the Evolution of SDN Architecture

For the various security problems existing in SDN, domestic and foreign researchers think that the typical security issues are mainly manifested in the following aspects: malicious applications, controller's vulnerability, legitimacy and consistency of the flow rules, the northbound interface standardized issue, communication security of the southbound interface and other aspects of the preliminary study and analysis. We analyze the typical security problems and the attack objects. The results of the analysis are shown in Table I.

From the above table, it can be seen that attack objects of different layers are different. Due to the clear multi-layer architecture of SDN, it is necessary to classify the security threats at different layer. We will discuss the existing security issues at each layer from north to south in this section.

3.1 Application layer

The application layer consists of various types of applications. In SDN, flow rules can be formulated by the administrator, OpenFlow applications, security service applications or other third-party applications. The controller will deliver the flow rules to the relevant switches and network devices in the data plane after generating. Typical security issues at the application layer are reviewed from the following aspects:

(1) Malicious application

In SDN, various types of applications bring powerful dynamism to the SDN architecture and improve the flexibility of network management. However, the switches and network devices at the data layer completely trust the flow rules delivered by the controller and execute without consideration. Once a malicious application participates in the formulation of the flow rules, it will bring unpredictable damage to SDN [9]. Therefore, malicious applications or external malicious attacks are the main security threats. Malicious applications such as viruses, Trojans, worms can attack SDN through software agents, and the cost of attack is relatively small.

(2) Authorized authentication

There is a lack of sufficient trust mechanism between the control layer and the application layer. We do not have enough trust assessment and management mechanism to verify whether the application is secure. The applications' own vulnerabilities may be exploited by the attackers to generate malicious flow rules. In addition, there are some defects in the application's role authentication and access control. Access control is critical to protecting system security [10]. We must ensure that system resources are not used or accessed illegally. Currently, there is no complete application authorized authentication module under the SDN environment. The controller is directly exposed to the attacker. Through the malicious application programs, the attacker can change the network deployment strategy, tamper with the switch flow table or steal privacy data, which can disrupt the availability of the SDN network.

(3) Flow rules conflict

There are a large number of third-party open-source applications in SDN. A variety of applications not only bring challenges to application configuration management, but also produce a higher degree of coupling. Flow rules and security policies generated by the application may appear to compete, cover, and

Table I. *Typical security issue and attack objects.*

Layer	Security issue	Attack object
Application layer	Malicious application	Code integrity
	Authorized authentication	Application software
	Flow rules conflict	Flow rules
Northbound interface	Northbound interface standardization	Controllers
Control layer	DoS/DDoS attacks	Controllers
	Hijacked/Rogue controller	Controllers
	Poor controller deployment	Controllers
Southbound interface	Communication security	Privacy data
Data layer	Authorized authentication	Basic equipment
	Legality of flow rules	Flow rules
	Consistency of flow rules	Flow rules
	DoS/DDoS attacks	Flow tables
	Side channel attacks	Privacy data

conflict. The above problems may cause network running chaos, invalid network security policies, and even some serious problems such as network crashes.

3.2 Northbound interface

At present, the largest security problem of the northbound interface is the standardization problem [11-13]. Due to the diversity and constant update of SDN applications, there are no uniform provisions on the methods of authorization and authentication. Compared with the southbound interface between the control layer and the data layer, the northbound interface trust relationship between the control layer and the application layer is more fragile. Attackers can exploit the openness and programmability of the northbound interface to launch attack. For example, they can access some important resources in the control layer, change the network status or occupy the controller resources. Therefore, it is imperative to solve the northbound interface standardization problem.

3.3 Control layer

At control layer, controller is the brain of SDN and the weakest link in the security chain. Since SDN manages networks through controllers, an attacker who compromises the controller will be able to control the entire network. Finally, it brings unpredictable damage to SDN. The controller is the main attack target in recent years. Typical security issues at the control layer are reviewed from the following aspects:

(1) DoS/DDoS attacks

DoS/DDoS attacks are the main attack mode to reduce the availability of SDN. The attacker creates a series of illegal accesses to impose excessive load on the controller, which causes the system resources unavailable to legitimate users [14]. When a new packet arrives at the switch, if there are no matching flow rules in the flow table, the entire packet or header will be forwarded to the controller, and the controller formulates the corresponding flow rules. Therefore, some attackers use the switches at the data layer to initiate a large number of false request messages, which will cause the controller to be overloaded and interrupt the legitimate switch request service. SDN centralized control makes the network resources more limited, and it is easier for attackers to initiate DoS/DDoS attacks.

(2) Hijacked/Rogue controller

The attacker can use the admin station to hijack the controller, thus causing the user's legitimate request to be rejected. In some severe cases, the attacker can use physical or logical method to destroy the controller and modify the network. SDN administrators can use upper-layer applications to configure and manage the network flexibly, but this programmable feature provides a new attack interface for unauthenticated applications. Due to the centralized control of SDN, attacks can quickly spread to the entire network [15], which is extremely harmful.

(3) Poor controller deployment

When the control layer is dominated by network attacks, it is easily overloaded. Because of the single-point failure and the insufficient capacity in large-scale networks, many researchers devote themselves to multi-controller deployment schemes. But under normal circumstances, a multi-controller deployment scheme always reassigns failed controller loads to another controller [16]. Once the controller deploys improperly, it may easily burden the controller load beyond its capacity, and cause a company-level failure [17]. In addition, a multi-controller deployment scheme will divide the network into different sub-networks. Different subnets will lead to network state aggregation consistency issues [18] and privacy problems [19]. The controller is also responsible for collecting the global network state diagram, if the state is inconsistent, it may lead to policy conflicts, such as a firewall security policy conflict [20].

3.4 Southbound interface

The southbound interface security is mainly caused by the leak of OpenFlow protocol [21, 22], specially refers to communication issues.

OpenFlow uses SSL/TLS protocol to encrypt data, but the SSL/TLS protocol is not secure [23, 24]. And since OpenFlow 1.3.0 version, TLS has been set to optional, which means we can use the channel without any security measures. In fact, the southbound interface faces eavesdropping, controller spoofing, data leakage and other security threats [25].

3.5 Data layer

The data layer is composed of switches and other basic devices. It is mainly responsible for data processing, forwarding, discarding and status collecting. It trusts the flow rules delivered by the controller. Typical security issues at the data layer are reviewed from the following aspects:

(1) Authorized authentication

As a matter of fact, data layer lacks of an effective authentication mechanism between the basic equipment and the controller. Therefore, there may be some problems such as identity impersonation and illegal access. A malicious switch may generate forged or false network data flow, manipulate or check the contents of data packets, and discard legal data packets. It will damage the integrity of the data and affect the availability of the data plane.

In addition, if the switch establishes a connection with the controller without authentication, the switch may be controlled by a malicious controller, which may cause the flow table information to be tampered with, resulting in leakage and other hazards. Out-of-order control instructions may cause confusion in switch flow table. This will directly increase security risks.

(2) Legality and consistency of flow rules

Legitimacy and consistency of the flow rules is one of the main problems at data layer. The legality of flow rules refers to malicious or incorrect flow rule injection. The consistency of flow rules mainly includes three aspects [26]. During the generation process, multiple applications cause conflicts or override flow rules. During the release process, transmission delay or malicious tampering causes the flow rules inconsistent between the controller and switches. During the update process, updating causes the synchronization of flow rules between different switches. In SDN network, failure of network nodes, traffic load transfer, or network maintenance may cause the flow rules to be updated, which make the packets see inconsistent network "views". If the data packets follow the rules of the new and old configuration mixed, it may occur problems such as black hole nodes, loop paths, or network congestion [27].

(3) DoS/DDoS attacks

The flow table space in the data plane is limited. Under normal circumstances, the flow table size of the switch can meet the data packet forwarding requirements. However, in a DoS/DDoS environment, the attacker creates a series of illegal accesses, and the flow table space is squeezed by invalid traffic rules [28]. A lot of flow table resources will be consumed, and normal flow rules do not have

Table II. *Causes of typical security problems.*

Layer	Security issue	Cause			
		SDN architecture	Access control	Encryption measures	Malicious attack
Application layer	Malicious application				√
	Authorized authentication		√		
	Flow rules conflict	√			
Northbound interface	Northbound interface Standardization	√			
Control layer	DoS/DDoS attacks	√			√
	Hijacked/Rogue controller		√		√
	Poor controller deployment	√			
Southbound interface	Communication security		√	√	
Data layer	Authorized authentication		√		
	Legality of flow rules		√		
	Consistency of flow rules	√			
	DoS/DDoS attacks	√			√
	Side channel attacks				√

enough space to process. DoS/DDoS attacks can degrade network performance.

(4) Side channel attacks

SDN carries private and confidential information all the time [29]. In SDN, the process attributes (like time attribute) of each execution action are different. By using side channel attacks, an attacker can infer network-related state information (such as flow table information) by testing the execution time of the specific type of data packet. Therefore, the flow table may cause data leakage problems. Although side channel attacks do not directly affect the availability, confidentiality or integrity of data, they can trigger further attacks.

To analyze the typical security issues at each layer of SDN, we find the main causes of the problems are SDN's own architecture, external malicious attacks, insufficient access control and encryption measures. We list the causes of security problems in Table II.

IV. Defense Technology and Challenges for SDN Security

Based on the research of typical security issues in section III, we can see that the centralized control and programmable features of the SDN provide attackers with powerful attack angles and convenient attack channels. With the increasing use of SDN, security issues are becoming more and more prominent. The main existing security defense technologies of SDN are shown in Table III.

There is no recognized solution for the northbound interface standardization problem and the southbound interface communication security problem. Therefore, we only discuss existing defense solutions from application layer, control layer, and data layer. We analyze the deficiencies of each method.

4.1 Application layer

(1) Authorized authentication module

For the lack of effective trust mechanism at the application layer, researchers recommend adding an authentication module. Through this method, the access of malicious nodes is prevented, and the application layer security is effectively improved.

FortNOX [30] and SE-Floodlight [31] both implement a role-based authentication module, which divides the role to provide different permissions for different applications. Both of them use digital signature technology to identify authorized applications and reject the unauthenticated request. However, neither of them can achieve fine-grained access control. They only classify all applications into system administrators, security applications, and security-independent OpenFlow applications.

OperationCheckPoint [32] extends the authorized permission system. Before the controller initializes the application program, OperationCheckPoint presets a series of permissions. Each application is assigned a unique ID. Before the application is accessed, OperationCheckPoint will execute the permission check. If the application request exceeds its permission, it will be considered invalid. This method adopts a fine-grained management method for application authority and improves the flexibility of management. However, OperationCheckPoint stores the program ID and access authority in a management table. Once the table is leaked or tampered, the authorization will be invalidated.

Table III. *Existing security defense technologies.*

Layer	Security Defense Technologies	Security Problems
Application layer	Authorized authentication module	Authorized authentication
	Application isolation	Malicious application
	Flow rules monitor	Flow rules conflict
Control layer	Traditional scheme enhancement	DoS/DDoS attacks
	Connection migration mechanism	
	System robustness	
	Flow priority configuration	
	User requests analysis	
	Threat modeling analysis	
	Flow rules analysis module	Flow rules conflict
Data layer	Network error detection	Configuration error
	Flow rules classification	Legality of flow rules
	Formal mathematical analysis	Legality of flow rules
	Consistency detection	Consistency of flow rules
	Authorization authentication module	Authorized authentication

PermOF [33] proposes that exposing all rights of OpenFlow to each application has a great security risk. Therefore, PermOF analyzes the access authority of the application. Based on minimizing access authority of the application, PermOF introduces an access layer, which can limit an illegal application access to the controller's kernel resources directly. PermOF achieves fine-grained authentication and management of access authority, but it takes a lot of processing time and efficiency of the system.

(2) Application isolation

The researchers found that the main reason for malicious applications affecting the system is that the high coupling between application and operating system. Researchers try to isolate applications and control layer, keep the application running independently. In this way, we effectively enhance the application layer security.

Rosemary [34] proposes to separate the application from the kernel completely. The resource manager limits each application process to one application area, and restricts each regional resource. Each request corresponds to an independent process. Rosemary implements a sandboxed design, it combines authentication with public and private key to check whether the application is legal. However, we also need pay attention to Rosemary's own system security. If Rosemary itself is penetrated, it will also cause great harm. In addition, Rosemary is computationally intensive and increases system load, which reduces the system performance.

LegoSDN [35] considers the impact of failure application. It proposes an isolation mechanism between the control layer and the application layer, just like Rosemary. The main difference between them is that LegoSDN bundles all applications on one plane. By establishing a global network system and combining with network logs, LegoSDN supports atomic updates and efficient rollback. It is a very sensible thought to isolate the potential malicious application before it is destroyed. However, LegoSDN cannot fully mitigate attacks.

(3) Flow rules monitor

Concerned the conflict between the new flow rules and the old flow rules, Porras et al. [30] propose to monitor the flow rules. For example, FortNOX monitors and compares the newly inserted flow rules. If there is a conflict, the new one will be dropped and added to a local cache. Administrators can create a set of hard-coded rules to override any dynamically created rules. In this way, we can prevent malicious applications from adding spurious flow rules to the controller. However, this mechanism also prevents legitimate security applications which rely on dynamically modifying network traffic to function properly.

4.2 Control layer

(1) DoS/DDoS defense

The DoS/DDoS attacks at the control layer have caused a great deal of damage, and the high difficulty of solving the problem has attracted the attention of researchers. DoS/DDoS attacks are the focal point of security research. The existing solution is mainly processed after the attack initiated, and belongs to the passive defense scheme. The typical security methods include the following aspects:

(i) Traditional scheme enhancement

Some researchers improve the traditional DoS/DDoS defense technology. The traditional abnormal traffic detection scheme is realized by information collection, feature extraction and classification identification.

Early DDoS Detection [36] uses the entropy change of the destination IP to detect DDoS attack. This method has fast speed and low overhead, and the scale of detection data packets is small. But the detection based on the destination IP address alone is easily bypassed.

SmartSec [37] is a smart security mechanism that monitors flow changes by reusing asynchronous messages on the control link. It distinguishes between new flow attacks and normal flow bursts by checking the hit rate of flow entries.

(ii) Connection migration mechanism

Avant-Guard [38] introduces a connection migration module to extend the OpenFlow data plane. By detecting and filtering invalid TCP session information, Avant-Guard will transfer the illegal and forged flow request information, and only legitimate flow request information will be sent to the controller. However, this method is only applicable to TCP-SYN attack. In actual environments, it needs to be deployed together with other defense mechanisms. The configuration will become cumbersome and consume system resources.

LineSwitch [39] overcomes Avant-Guard's buffer-saturated and TCP port limitations. It uses a probabilistic proxy and blacklist to proxy the first connection from an IP address, while subsequent incoming connections from the same IP address are proxied with a specific probability. But LineSwitch needs to modify the switch, which is troublesome.

SLICOTS [40] blocks malicious hosts by monitoring the number of ongoing TCP requests. It monitors all TCP handshake procedures and sets thresholds. When the number of half-open TCP connections exceeds the threshold, SLICOTS issues a flow rule to block malicious clients. Similar to Avant-Guard, this solution is only applicable to TCP-SYN attacks.

SHDA [41] monitors incomplete HTTP requests on the web server. It calls the controller when the number of requests exceeds a predefined threshold. If the processing time of the request exceeds the predefined time period, the source of the request will be regarded as a malicious attacker, and the system will prohibit the data packet sent by the same source on the switch. Similar to the above method, SHDA is only feasible for Slow HTTP DDoS attack.

(iii) System robustness

FloodGuard [42] transfers the attack traffic to an independent cache outside the data plane. It uses a round-robin scheduling algorithm to process multiple types of cache queues, which can effectively reduce the attack intensity. FloodGuard makes up for the lack of Avant-Guard and can defense against more types of DoS/DDoS attacks.

MLFQ [42] enforces fair sharing of the SDN controller resources through multi-layer queues. The queues can be dynamically extended and aggregated based on the controller load. However, the controller cannot limit flooding requests, so MLFQ cannot protect data from traffic congestion attack.

MultiSlot [44] proposes a multi-queue SDN controller scheduling algorithm based on time slice allocation strategy. This method adopts different time slice allocation strategies according to the strength of the DDoS attack. When the attack strength is different, the time slice is different, which uses an equation to calculate. After calculating, MultiSlot uses the controller to schedule flow requests from different switches, including normal switches and attacked switches. The disadvantage of MultiSlot is that legitimate traffic on the attacked switch will also be delayed.

(iv)Flow priority configuration

L Wei et al. [45] propose a request priority scheme to evaluate the trust value of each node and store the results in different priority queues. Suspicious requests are still served, but low-priority flows will be delayed. The node trust value increases over time and will return to the higher priority queue. Currently, this mechanism performs optimally.

(v) User requests analysis

Dao et al. [46] propose to analyze the source IP address of data packets. If the source IP address is abnormal, it will be recorded as malicious, and the controller will discard the request from this address. However, this solution is vulnerable to real factors. Once the source address is marked as a malicious address, the controller will deny the request from that IP, possibly causing the real request from the non-malicious user to be dropped.

(vi) Threat modeling analysis

The threat analysis model runs ahead of the SDN network deployment to evaluate potential DoS/DDoS threats in the system. Kloti et al. [47] combine the STRIDE threat monitoring model with the attack tree technology to model the OpenFlow system. By analyzing the data flow diagrams and simulation attacks of

the OpenFlow system, we can characterize the potential threats of DoS/DDoS attacks in the SDN architecture in advance.

(vii) Network behavior verification

We can verify network behaviors to detect DoS/DDoS attacks, such as link latency measurement, available bandwidth measurement, and network topology discovery.

FlowTrace [48] measures the link delay by locating each forwarding path of the stream. The collector collects the flow entries and builds the virtual flow table passively. If the flow entries change, the virtual table can be updated. The calculator simulates the forwarding action of each switch based on the virtual flow table to calculate flow path. The monitor measures the delay of the stream by inserting a temporary flow rule along the determined path. FlowTrace requires two rounds to measure latency and increase system load.

CSMABw [49] provides a versatile way to calculate end-to-end ABw (available bandwidth) in SDN. Firstly, it discovers the network topology map, and then polls the OpenFlow switch counter with the FlowStatsReq message. CSMABw calculates the current bandwidth load $b_{(i)}t$ by $b_{(i)}t = n_{(i)}t - n_{(i)}\left(t-T\right)/T$, and estimates the available bandwidth of any link in the network as $a_{(i)}t = c_i - b_i$. But CSMABw has a large cycle impact.

The OpenFlow Discovery Protocol (OFDP) is the most popular topology discovery method in SDN [50]. In OFDP V2 [51], it limits the ability of each switch can only send one LLDP packet, and then transmits LLDP packet to all available ports for network topology discovery. Although there is a significant performance improvement over the OFOD protocol, this method still consumes a lot of computing resources and takes a lot of time.

Qiu et al. [52] propose the Global Flow Table (GFT) mechanism in advance. GFT can provide the path of all streams in the SDN network, and records the input and output port of the stream with the entry time and the end time. Otherwise, the field information of the data packet (such as source IP, source MAC address, etc.) will be recorded. Based on [52], Qiu et al. [53] propose a CMD algorithm by analyzing the process and principle of MITM attack. It detects MITM attacks based on the topology and connections of the network flow, which does not require analysis of the network packets' contents.

(2) Flow rules analysis module

FloodGuard[42] contains an active flow rule analysis module. In the case of an attack, FloodGuard can generate corresponding flow rules based on the running logic of the controller and its application. In this way, FloodGuard can ensure the execution of basic network policies. FloodGuard does not modify the data plane, which is beneficial for SDN to deploy. However, it needs to execute a lot of algorithms, which will greatly increase controller load and reduce system performance.

FLOVER [54] is combined with Yices SMT solver. Flover uses assertion sets and modular theory techniques to detect whether dynamic streaming techniques conflict with existing security technologies. Flover can only judge the conflict of flow rules, so it needs to be combined with other measurements in fact. Each flow rule request in the network need to perform non-bypass attribute verification. Therefore, Flover need to support the batch processing mode to improve the controller's response time.

(3) Multi-controller deployment

Due to the single point of failure caused by centralized control and the lack of processing capacity of single controller in large-scale networks, many researchers have proposed a multi-controllers deployment scheme. The current deployment methods are mainly divided into two types: flat control and hierarchical control. In the flat control mode, each controller manages its own network. Each of controller has the same status, but it is easy to cause a waste of resources and increase the system load. Different operators may not be able to communicate equally between different domains. In the hierarchical control mode, the local controller is responsible for the respective

network, and the global controller is responsible for the local controller. The interaction between controllers is accomplished through the global controller, but the attacker may still launch an attack on the global controller, such as DoS/DDoS attacks.

CPCC [55] utilizes the controller load as an element and introduces an efficient algorithm to solve the problem of controller placement. It uses the integer programming model SP to find the minimal number of controllers with a specified radius r. In this way, CPCC reduces the number of required controllers, the load of the maximum-load controller, and the radius stretches. But it is inefficient in the presence of time-varying request arrivals.

DBCP [56] uses a density-based switch clustering algorithm to divide the network into several sub-networks. Each sub-network has a controller, and the size of sub-network is determined by the capacity of the deployed controller. When the main link is broken, the switch can choose a random backup link, which can result in unnecessary network delays.

Zhong et al. [57] propose to implement controller placement through neighborhood and minimum coverage, and try to use less controllers to achieve network reliability and low latency. But the survivability of the backup controller and the control link is poor in the case of regional fault.

Li et al. [58] propose a multi-controller architecture with Byzantine fault tolerance mechanism. When one controller fails, the other controller takes over the network and cuts off the former connection. However, this method has low performance and is only applicable to small networks.

At present, there is no security scheme that can integrate the key security functions, trust models and technical arbitration of SDN controller.

4.3 Data layer

(1) Network error detection

Network error detection is realized by detection algorithm. NICE [59] gives a test scheme based on symbolic execution model to check whether the upper application produces inconsistent network state. FlowChecker [60] uses a binary decision graph to test configuration errors within a single switch's internal flow table. Anteater [61] uses static analysis to diagnose network configuration problems. However, these methods are non-real-time solutions, which have large processing delay and high demand. They cannot fundamentally eliminate the impact of configuration conflicts to the network.

(2) Flow rules division

Dividing the application permissions based on entity's role and priority is the main method to solve legitimacy and consistency of the flow rules. The permission of the flow rules is divided by techniques such as digital signature, role division, and function classification.

Perm-guard [62] uses digital signatures to implement flow rules permission setting. If a controller or application wants to change flow rules, it needs to authenticate itself to the central authority. If the signature is incorrect, the controller or application will be considered illegal. Although this scheme cannot prevent an attacker from attempting to hijack a controller or create a malicious controller, it rejects malicious changes to the network structure by pushing rules to data paths in the network.

FortNox [30] classifies the entities involved in the formulation of SDN flow rules. FortNox validates the flow rule's legitimacy and rating before it is written into the flow table. However, dividing the participating entities into three categories is too simple to implement fine-grained access control.

(3) Consistency detection

Flowvisor [63] uses the virtualized sharing technology to divide the entire network into slices. Each slice is controlled as a separate application, and different slices isolates naturally. Flowvisor prevents flow rules conflict by different slices.

NeSMA [64] is a framework which supports network-level state awareness technology. Sun et al. process the complex network-level state in NeSMA through a combination of se-

quential and parallel methods. The data plane checks the network status periodically and reports to the controller when the state transitions. NeSMA can achieve more advanced state awareness by exploring more granular states. However, different state retrievals have different effects on system resource.

Reitblatt et al. [65] propose a packet integrity detection mechanism, which is achieved by combining the flow rules with the version number. During the update, the controller installs a new rule and retains the old one. Once all the packets of the old policy leave the network, the controller will delete the old rule from all switches. However, the cost of this solution is too large, and we need multiple configurations to achieve complete control. Based on this, Reitblatt proposes a flow consistency detection mechanism by combining version number and timeout. The controller sets the rule timeout for the old configuration and installs the new configuration with low priority. This solution cannot deal with the flow involving multiple entries, and the application environment of this method is relatively simple.

(4) Authorized authentication module

AuthFlow [66] uses a RADIUS server to authenticate the host identity. AuthFlow performs access control for each host based on its level of authority by mapping host credentials to the set of flows. AuthFlow ensures that packets containing the incoming ports of other devices are not transmitted until both devices are successfully authenticated. AuthFlow guarantees low overhead and implements a fine-grained access control.

In addition, we can use quantum cryptography to design an ultra-lightweight integrity verification, such as recently proposed in [67-69]. Quantum passwords are more secure than traditional passwords, but their implementation may be more expensive. Moreover, in cryptography, quantum cryptography is a new technology, although they have possible technical advantages, but the new approach will need to be tested over time.

(5) DoS/DDoS defense

Jia et.al [70] consider the DDoS attack in the data layer can be transformed into the flow table optimization problem. They propose the K Similar Greedy Tree (KSGT) algorithm to maximize the number of flows in the network, which can improve the efficiency of the flow table in SDN.

We summarize the above security methods as shown in Table IV.

These security technologies have mitigated SDN security threats to a certain extent, but there are still many security problems need to be solved in the future. We need to focus on more aspects.

V. Future Research Directions

As one of the hottest and most promising technologies, SDN takes advantage of control and forwarding separation to make up for the shortcomings of traditional networks scalability, flexibility and effectiveness. SDN has received extensive attention from academia and industry, and it has been widely used in data centers [71,72], cloud computing [73], wireless LAN [74], smart grid [75], smart home [76], and other application scenarios [77]. However, SDN is not mature at present, and the centralized control architecture has brought a series of new security issues. Existing solutions are not sufficient to completely alleviate security issues in SDN. In the future, we need to focus on the following aspects.

(1) Controller DoS/DDoS attacks detection and precaution

SDN uses centralized control to manage the entire network and concentrates the "wisdom" of the network on the controller. As a result, it is necessary to continuously research and design new DoS/DDoS detection and prevention technologies based on the architecture features of SDN. At present, there are problems such as large overhead on collecting information, insufficient detection precision, and insufficient speed for the controller to defend against DoS/DDoS attacks.

(2) Controller scalability and cross-domain communication

Table IV. *Summary of security methods.*

Layer	Method	Solution	Implementation	Defect
Application layer	Authorized authentication module	FortNox	Role-based authentication	Coarse granularity
		SE-Floodlight	Role-based authentication	Coarse granularity
		OperationCheckPoint	Permission preset	Table security
		PermOF	Minimize access	High overhead
	Application isolation	Rosemary	Separate applications and kernel	High overhead
		LegoSDN	Separate applications and kernel	Cannot fully mitigate attacks
	Flow rules monitor	FortNOX	Monitor and compare inserted flow rules	Reduce flow rule flexibility
Control layer	Traditional scheme enhancement	Lightweight-DDoS Detection	Extract DDoS traffic characteristics	Large cycle length impact
		SmartSec	Monitor flow changes	
	Connection migration mechanism	Avant-Guard	Connection migration module	Only feasible for TCP-SYN attack
		LineSwitch	Probabilistic proxy and blacklist	Modify the OF switch
		SLICOTS	Monitor the number of TCP Connection requests	Only feasible for TCP-SYN attack
		SHDA	Monitor the number of HTTP connection requests	Only feasible for Slow HTTP DDoS attack
	System robustness	FloodGuard	Independent cache	Extra cache space
		MLFQ	Multi-layer fair queues	Congestion control
		MultiSlot	Time slice allocation	Delayed legal traffic
	Flow priority configuration	FlowRanger	Node trust evaluation	High overhead
	User requests analysis	source IP address analysis	Analyze source IP address	Deny non-malicious user
	Threat Modeling Analysis	Security analysis	STRIDE and attack tree technology	Prior knowledge learning
	Network behavior verification	FlowTrace	Forwarding path delay	High overhead
		CSMABw	End-to-end ABw	Large cycle length impact
		OFDP	LLDP packet	High overhead
		OFDP V2	LLDP packet	High overhead
		CMD algorithm	Global Flow Table	Only feasible for MIMT attack
	Flow rules analysis module	FloodGuard	Flow rule analysis module	High overhead
		Flover	Yices SMT solver	Batch processing mode
	Milt-controller deployment	CPCC	Utilize the controller load	Inefficient time-varying requests
		DBCP	Density-based clustering algorithm	Network delay
		Zhong et al.	Min-cover algorithm	Lower survivability of backup controllers and control link
		Byzantine-resilient	Multi-controller architecture	Only feasible for small networks
Data layer	Network error detection	NICE	Symbolic execution	High latency
		FlowChecker	Binary decision graph	High latency
		Anteater	Static analysis	High latency
	Flow rules division	Perm-guard	Digital signatures	Cannot find malicious or hijacked controller
	Consistency detection	Flowvisor	Virtualized sharing technology	High overhead
		NeSMA	Network level state awareness	Different effects of different state retrieval
		Packet integrity detection	Combine matching rules with version number	High overhead
		flow consistency detection	Combine version number with rule timeout	Unable to handle multi-portal traffic
	Authorized authentication module	AuthFlow	Verify the host identity	High latency
	DoS/DDoS defense	Flow table optimization	K Similar Greedy Tree (KSGT) algorithm	Limited flow table space

To alleviate the single point of failure and poor scalability problem of SDN, OpenFlow1.3 added a multi-controller deployment strategy. However, multiple controllers may be distributed in different autonomous domains, and operations such as identity switching and resource scheduling need to be performed between different controllers. How to ensure the secure and real-time communication between multiple controllers will be an important issue to be resolved in the future of SDN security.

(3) Application authentication issues

Malicious application problems are one of the major security issues in SDN. To avoid deploying malicious or damaged applications, we should establish a trusted connection between application layer and control layer. In addition, we need to authenticate the entity's identity before exchanging control messages.

(4) Northbound interface standardization

With the continuous development of SDN technology in recent years, OpenFlow has become the de facto protocol standard of the southbound interface. However, the standardization of northbound interface protocol still faces many problems. The complexity of controller types, diversification of operating systems and completion of functional composition limit the portability and scalability of the northbound interface. Therefore, the standardization of northbound interface will be an important part of the SDN security work in the future.

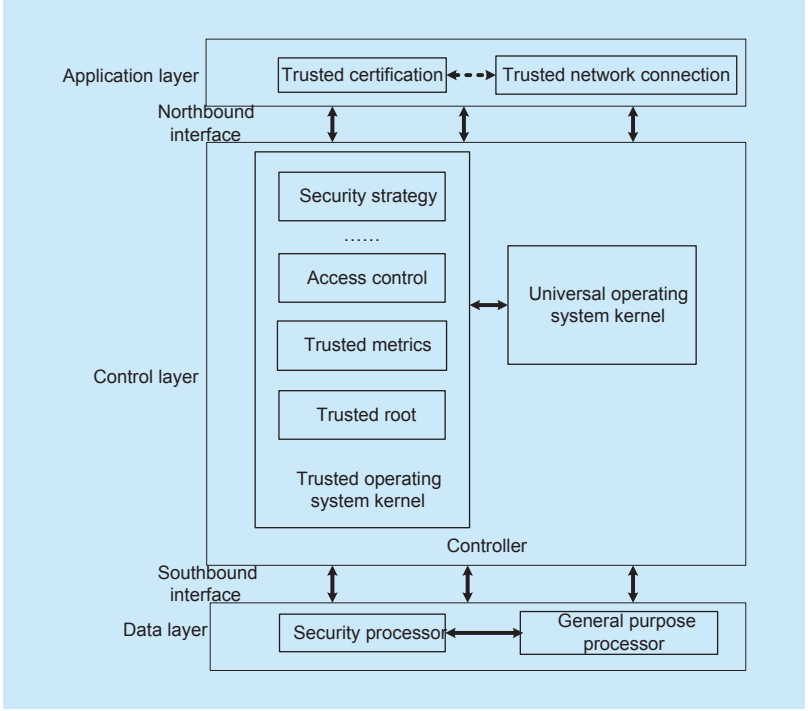


Fig. 2. *Idealized global security architecture.*

(5) Global Security Architecture

Through the previous threat analysis and current situation research in section 3 and 4, we can find that the SDN security problem is not a single-tier problem. SDN security threat involves each layer of the entire system architecture. It is far from enough to do a research on a single layer. As we can see from the previous work, we need a system perspective to solve the security problem of SDN. Currently, there is no satisfactory solution.

An idealized global security solution needs to cover hardware, operating systems, software and other aspects. SDN requires three layers and two interfaces to perform their respective duties and integrate into each other as a whole. We believe that we can use the idea of trusted computing to achieve an idealized global security solution. In 2006, China developed the first trusted PDA prototype system [78], which borrowed from the TCG specification. We achieve the academic thought of "Trust ≈ Dependability + Security" [79] to establish a relatively complete set of trust mechanisms from underlying hardware, operating system, and software. It is feasible to combine the idea of trusted computing with software and hardware to build a secure, trusted, globally integrated security solution.

Our idealized global security architecture is shown in figure 2. At the data layer, an independently designed security coprocessor is required to process the encryption and decryption operations. In order to guarantee access of confidential data, we encrypt the data transmission process and use a hardware isolation mechanism to build an isolated area in the memory. At the control layer, we implement secure boot by adding a trusted root, which can against various types of attacks such as code injection in software systems. Secure boot digitally signs the startup image of each

stage and checks the initial stage of the controller. It can ensure that the control layer code is not tampered. Only correctly signed kernels can be loaded into memory by the bootloader after a real-time trusted measurement. At the same time, the trusted execution environment is deployed at the control layer, which is responsible for sending upper-level security instructions to the data layer and returning the results to the control layer. Finally, the application layer should build a trusted network connection to perform identity authentication on various components such as terminals and platforms. After passing authentication, it measures the trusted status of the terminal platform. If the measurement result meets the security policy, the terminal is allowed to access. Otherwise, the terminal will be connected to the specified isolation area to repair and upgrade.

The global security architecture is the key point to ensure the safe and stable operation of the SDN. It is necessary to build an information security proactive defense system and information security governance system. According to the national relevant safety regulations, we need to design an overall safety protection scheme that meets the three-level system safety requirements.

VI. Conclusion

SDN separate control plane from data plane has brought a tremendous transformation in network technology. In the initial stage of SDN technology, performance was the primary consideration, while security is ignored. At present, SDN are widely used in emerging network application scenarios such as cloud computing, data centers, enterprise networks etc. With the gradual expansion of SDN applications, SDN security concerns are also increasing. This survey focuses on current advances in SDN security research. Firstly, we analyze the architecture and security threats of SDN. Then, we discuss deeply the existing security solutions and deficiencies. Finally, we propose an idealized security global solution and the direction of future SDN security development.

In the era of complex software design, people are faced with ever-expanding functional requirements and frequent security attacks. How to ensure the security of SDN from bottom to top is a huge challenge for researchers. When building a secure and trusted SDN system architecture, researchers should combine the most advanced technology such us artificial intelligence, data mining and deep learning. We can apply these technologies to achieve automatic software behavior measurement, active defense attack, discover intelligent vulnerability and other aspects.

We hope the article can make beneficial explorations for SDN research and industry development.

Acknowledgements

The authors would like to thank anonymous reviewers for their detailed comments. This work was supported by the Wuhan Frontier Program of Application Foundation (No.2018010401011295) and National High Technology Research and Development Program of China ("863" Program) (Grant No. 2015AA016002).

References

[1] D. Kreutz, F.M.V. Ramos, P.E. Veríssimo, et al., "Software-Defined Networking: A Comprehensive Survey", *Proceedings of the IEEE*, vol.103, no.1, 2015, pp. 14-76.

[2] S. Sezer, S. Scotthayward, P.K. Chouhan, et al., "Are we ready for SDN? Implementation challenges for software-defined networks", *IEEE Communications Magazine*, vol.51, no.7, 2013, pp. 36-43.

[3] Stanford University. Clean slate program[EB/OL]. http://cleanslate.stanford.edu/. 2006.

[4] F. Hu, Q. Hao, K. Bao, "A Survey on Software-Defined Network and OpenFlow: From Concept to Implementation", *IEEE Communications Surveys & Tutorials*, vol.16, no.4, 2014, pp. 2181-2206.

[5] Jain S, Kumar A, Mandal S, et al., "B4: experience with a globally-deployed software defined wan[M]// ACM SIGCOMM Computer Communication Review. ACM, 2013:3-14.

[6] Research and Markets[EB/OL]. https://www.researchandmarkets.com. 2018.

[7] MIT Technology Review. Breakthrough technologies, TRI0: Software-defined networking[EB/

OL]. http://www2.te-chnologyreview.com/article/412194/tr10-software-defined-networking/. 2009.

[8] S. Shin, L. Xu, S. Hong, et al., "Enhancing Network Security through Software Defined Networking (SDN)", *Proc. ICCCN*IEEE, 2016.

[9] Z. Fu, J. Li, "High Speed Regular Expression Matching Engine with Fast Pre-Processing", *China Communications*, vol.16, no.2, 2019, pp.177-188.

[10] W. John, A. Kern, M. Kind, et al., "Splitarchitecture: SDN for the carrier domain", *IEEE Communications Magazine*, vol.52, no.10, 2017, pp. 146-152.

[11] F. Klaedtke, G.O. Karame, R. Bifulco, et al., "Access control for SDN controllers", *Proc. ACM SIGCOMM Workshop on Hot Topics in Software Defined NETWORKING*, 2014, pp. 1325-1335.

[12] Y.E. Oktian, S.G. Lee, H.J. Lee, et al., "Secure your Northbound SDN API", *Proc. International Conference on Ubiquitous & Future Networks*, 2015, pp. 919-920.

[13] C.R. Vasconcelos, R.C.M. Gomes, A.F.B.F. Costa, et al., "Enabling high-level network programming: A northbound API for Software-Defined Networks", *Proc. 31st International Conference on Information Networking (ICOIN)*, 2017, pp. 662-667.

[14] J. Spooner, S.Y. Zhu, "A Review of Solutions for SDN-Exclusive Security Issues", *International Journal of Advanced Computer Science & Applications*, vol.7, no.8, 2016, pp. 113-122.

[15] H. Li, P. Li, S. Guo, et al., "Byzantine-resilient secure software-defined networks with multiple controllers", *Proc. IEEE International Conference on Communications*, 2014, pp. 695-700.

[16] T. Hu, P. Yi, JH. Zhang, et al., "Reliable and Load Balance-Aware Multi-Controller Deployment in SDN", *China Communications*, vol.15, no.11, 2018, pp.184-198.

[17] G. Yao, J. Bi, L.Guo, "On the cascading failures of multi-controllers in Software Defined Networks", *Proc. IEEE International Conference on Network Protocols*, 2014, pp. 1-2.

[18] E. Al-Shaer, S. Al-Haj, "FlowChecker:configuration analysis and verification of federated openflow infrastructures", *Proc. ACM Workshop on Assurable and Usable Security Configuration*, 2010, pp. 37-44.

[19] M. Nandhini, B. Praveenkumar, "An Implementation of Public Key Infrastructure Using Wireless Communication Networks", *International Journal of Grid & Distributed Computing*, vol.8, no.3, 2015, pp. 35-41.

[20] E. Al-Shaer, Q. Duan, S. Al-Haj, et al., "SensorChecker:reachability verification in mission-oriented sensor networks", *Proc. ACM International Workshop on Mission-Oriented Wireless Sensor NETWORKING*, 2013, pp. 51-56.

[21] M. Feng, Z. Xu, C. Wang, et al., "SDN-based Satellite Networks and Southbound Interface Protocol Extension", *Radio Communications Technology*, vol.43, no.5, 2017, pp. 19-23.

[22] S. Hyun, J. Kim, H. Kim, et al., "Interface to Network Security Functions for Cloud-Based Security Services", *Proc. IEEE Communications Magazine*, vol.56, no.1, 2018, pp. 171-178.

[23] F. Giesen, F. Kohlar, D. Stebila, "On the security of TLS renegotiation", *Proc. ACM Sigsac Conference on Computer & Communications Security*, 2013, pp. 387-398.

[24] H.E. Tschofenig, T. Fossati , "Transport Layer Security (TLS)/Datagram Transport Layer Security (DTLS) Profiles for the Internet of Things", *Physiological Reviews,* vol.66, no.4, 2016, pp. 1121-1188.

[25] K. Benton, L.J. Camp, C. Small, "OpenFlow vulnerability assessment", *Proc. ACM SIGCOMM Workshop on Hot Topics in Software Defined NETWORKING*, 2013, pp. 151-152.

[26] M. Wang, J. Liu, J. Chen, et al., "Software defined networking: Security model, threats and mechanism", *Journal of Software*, vol.27, no.4, 2016, pp. 969-992.

[27] K. Wu, J. Liang, S.C. Lee, et al., "Efficient and Consistent Flow Update for Software Defined Networks", *IEEE Journal on Selected Areas in Communications*, vol.36, no.3, 2018, pp. 411-421.

[28] B. Yuan Bin, D. Zou, S. Yu, et al., "Defending against Flow Table Overloading Attack in Software-Defined Networks", *IEEE Transactions on Services Computing*, vol.12, no.2, 2016, pp. 1-13.

[29] L. Schehlmann, S. Abt, H. Baier, "Blessing or curse? Revisiting security aspects of Software-Defined Networking", *Proc. International Conference on Network and Service Management*, 2015, pp. 382-387.

[30] P. Porras, S. Shin, V. Yegneswaran, et al., "A security enforcement kernel for OpenFlow networks", *Proc. the first workshop on Hot Topics in Software Defined Networking*, 2012, pp. 121-126.

[31] S. Cheung, M. Fong, P. Porras, et al., "Securing the Software-Defined Network Control Layer", *Proc. Proceedings of the 2015 Network and Distributed System Security Symposium (NDSS)*, 2015, pp. 1-15.

[32] S. Scotthayward, C. Kane, S. Sezer, "OperationCheckpoint: SDN Application Control", *Proc. International Conference on Network Protocols*, 2014, pp. 618-623.

[33] X. Wen, Y. Chen, Hu Chengchen, et al., "Towards a secure controller platform for openflow applications", *Proc. ACM SIGCOMM Workshop on Hot Topics in Software Defined NETWORKING*, 2013, pp. 171-172.

[34] S. Shin, Y. Song, T. Lee, et al., "Rosemary:A Robust, Secure, and High-performance Network Operating System", *Proc. 21st ACM Conference*

on Computer and Communications Security, 2014, pp. 78-89.

[35] B. Chandrasekaran, T. Benson, "Tolerating SDN application failures with LegoSDN", P*roc. The Workshop on Hot Topics in Software Defined NETWORKING*, 2014, pp. 235-236.

[36] S.M. Mousavi, M. Sthilaire, "Early detection of DDoS attacks against SDN controllers", *Proc. International Conference on Computing, NETWORKING and Communications*, 2015, pp. 77-81.

[37] T. Xu, D. Gao, P. Dong, et al., "SmartSec: A Smart Security Mechanism for the New-Flow Attack in Software-Defined Networking", *Proc. IEEE 85th Vehicular Technology Conference (VTC Spring)*, 2017, pp. 1-7.

[38] S. Shin, V. Yegneswaran, P. Porras, et al., "AVANT-GUARD:scalable and vigilant switch flow management in software-defined networks", *Proc. ACM Sigsac Conference on Computer & Communications Security*, 2013, pp. 413-424.

[39] M. Ambrosin, M. Conti, F.D. Gaspar, et al., "LineSwitch: Tackling Control Plane Saturation Attacks in Software-Defined Networking", *IEEE/ACM Transactions on Networking*, vol.25, no.2, 2017, pp. 1206-1219.

[40] R. Mohammadi, R. Javidan, M. Conti, "SLICOTS: An SDN-Based Lightweight Countermeasure for TCP SYN Flooding Attacks", *IEEE Transactions on Network & Service Management*, vol.14, no.2, pp. 487-497.

[41] K. Hong, Y. Kim, H. Choi, et al., "SDN-Assisted Slow HTTP DDoS Attack Defense Method", *IEEE Communications Letters*, vol.22, no.4, 2018, pp. 688-691.

[42] H. Wang, L. Xu, G. Gu, "FloodGuard: A DoS Attack Prevention Extension in Software-Defined Networks", *Proc. Ieee/ifip International Conference on Dependable Systems and Networks*, 2015, pp. 239-250.

[43] P. Zhang, H. Wang, C. Hu, et al., "On Denial of Service Attacks in Software Defined Networks", *IEEE Network*, vol.30, no.6, 2016, pp. 28-33.

[44] Q. Yan, Q. Gong, F.R. Yu, "Effective software-defined networking controller scheduling method to mitigate DDoS attacks" *Electronics Letters*, vol.53, no.7, 2017, pp. 469-471.

[45] L. Wei, C. Fung, "FlowRanger: A request prioritizing algorithm for controller DoS attacks in Software Defined Networks", *Proc. IEEE International Conference on Communications*, 2015, pp. 5254-5259.

[46] N.N. Dao, J. Park, M. Park, et al., "A feasible method to combat against DDoS attack in SDN network", *Proc. International Conference on Information NETWORKING*, 2015, pp. 309-311.

[47] R. Kloti, V. Kotronis, P. Smith, "OpenFlow: A security analysis", *Proc. IEEE International Conference on Network Protocols*, 2014, pp. 1-6.

[48] S. Wang, J. Zhang, T. Huang, et al., "FlowTrace: measuring round-trip time and tracing path in software-defined networking with low communication overhead", *Frontiers of Information Technology and Electronic Engineering*, vol.18, no.2, 2017, pp. 206-219.

[49] M. Péter, A. Botta, G. Aceto, et al., "Challenges and solution for measuring available bandwidth in software defined networks" *Computer Communications*, vol.99, 2017, pp. 48-61.

[50] F. Pakzad, M. Portmann, W.L. Tan, et al., "Efficient topology discovery in software defined networks", *Proc. International Conference on Signal Processing & Communication Systems*, 2014.

[51] S. Khan, A. Gani, A.W.A. Wahab, et al., "Topology Discovery in Software Defined Networks: Threats, Taxonomy, and State-of-the-Art", *IEEE Communications Surveys & Tutorials*, vol.19, no.1, 2017, pp. 303-324.

[52] X. Qiu, K. Zhang, Q. Ren, "Global Flow Table: A convincing mechanism for security operations in SDN", *Computer Networks*, vol.120, 2017, pp. 56-70.

[53] K. Zhang, X. Qiu, "CMD: A convincing mechanism for MITM detection in SDN", P*roc. IEEE International Conference on Consumer Electronics*, 2018.

[54] S. Son, S. Shin, V. Yegneswaran, et al., "Model checking invariant security properties in OpenFlow", *Proc. IEEE International Conference on Communications*, 2013, pp. 1974-1979.

[55] G. Yao, J. Bi, Y. Li, et al., "On the Capacitated Controller Placement Problem in Software Defined Networks", *IEEE Communications Letters*, vol.18, no.8, 2014, pp. 1339-1342.

[56] J. Liao, H. Sun, J. Wang, et al., "Density cluster based approach for controller placement problem in large-scale software defined networkings", *Computer Networks*, vol.112, 2017, pp. 24-35.

[57] Q. Zhong, Y. Wang, W. Li, et al., "A min-cover based controller placement approach to build reliable control network in SDN", *Proc. Network Operations & Management Symposium*, 2016, pp. 481-487.

[58] H. Li, P. Li, S. Guo, et al., "Byzantine-resilient secure software-defined networks with multiple controllers", *Proc. IEEE International Conference on Communications*, 2014, pp. 695-700.

[59] M. Canini, D. Venzano, P. Peresini, et al., "A NICE way to test openflow applications", *Proc. Proceedings of 9th Usenix Symposium on Networked Systems Design & Implementation*, 2013, pp. 1-14.

[60] E. Al-Shaer, S. Al-Haj, "FlowChecker:configuration analysis and verification of federated openflow infrastructures", *Proc. ACM Workshop on Assurable and Usable Security Configuration*, 2010, pp. 37-44.

[61] S. Son, S. Shin, V. Yegneswaran, et al., "Model

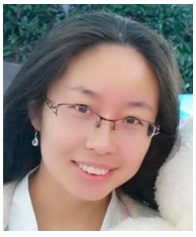
checking invariant security properties in OpenFlow", *Proc. IEEE International Conference on Communications*, 2013, pp. 1974-1979.
[62] M. Wang, J. Liu, J. Chen, et al., "PERM-GUARD: Authenticating the Validity of Flow Rules in Software Defined Networking", *Proc. IEEE, International Conference on Cyber Security and Cloud Computing*, 2016, pp. 1-17.
[63] S. Azodolmolky, R. Nejabati, S. Peng, et al., "Optical FlowVisor: An OpenFlow-based optical network virtualization approach", *Proc. Optical Fiber Communication Conference and Exposition*, 2013, 1-3.
[64] C. Sun, J. Bi, H. Hu, et al., "NeSMA: Enabling network-level state-aware applications in SDN", *Proc. IEEE International Conference on Network Protocols*, 2016.
[65] M. Reitblatt, N. Foster, J. Rexford, et al., "Consistent updates for software-defined networks: change you can believe in!" *Proc. ACM Workshop on Hot Topics in Networks*, 2011, pp. 1-6.
[66] D.M.F. Mattos, O.C.M.B. Duarte, "AuthFlow: authentication and access control mechanism for software defined networking", *Annals of Telecommunications*, vol.71, no.11-12, 2016, pp. 607-615.
[67] A. Farouk, J. Batle, M. Elhoseny, et al., "Robust general N user authentication scheme in a centralized quantum communication network via generalized GHZ states", *Frontiers of Physics*, vol.13, no.2, 2018, pp. 13-30.
[68] Y. Mao, B. Wang, C. Zhao , et al., "Integrating quantum key distribution with classical communications in backbone fiber network", *Optics Express*, vol.26, no.5, 2018, pp. 6010-6020.
[69] A. Aguado, V. Lopez, J. Martinez-Mateo, et al., "Hybrid Conventional and Quantum Security for Software Defined and Virtualized Networks", *Journal of Optical Communications and Networking*, vol.9, no.10, 2017, pp. 819-825.
[70] X. Jia, Y. Jiang, Z. Guo, et al., "Reducing and Balancing Flow Table Entries in Software-Defined Networks", *Proc. Local Computer Networks*, 2016.
[71] Z. Zali, MR. Hashemi, l. Cianci, et al., "A Controller-Based Architecture for Information Centric Network Construction and Topology management", *China Communications*, vol.15, no.7, 2018, pp.131-145.
[72] Y. Shen, P. Samadi, K. Bergman, "Autonomous network and IT resource management for geographically distributed data centers", *IEEE/OSA Journal of Optical Communications & Networking*, vol.10, no.2, 2018, pp. A225-A231.
[73] H. Yang, J. Zhang, Y. Ji, et al., "Experimental demonstration of multi-dimensional resources integration for service provisioning in cloud radio over fiber network", *Scientific Reports*, vol.6, no.1, 2016, doi: 10.1038/srep30678.
[74] L. Ma, XM. Wen, LH. Wang, et al., "An SDN/NFV Based Framework for Management and Deployment of Service Based 5G Core Network", *China Communications*, vol.15, no.10, 2018, pp.86-98.
[75] D.A. Chekired, L. Khoukhi, H.T. Mouftah, "Decentralized Cloud-SDN Architecture in Smart Grid: A Dynamic Pricing Model", *IEEE Transactions on Industrial Informatics*, vol.14, no.3, 2018, pp. 1220-1231.
[76] K. Xu, X. Wang, W. Wei, et al., "Toward software defined smart home", *IEEE Communications Magazine*, vol.54, no.5, 2016, pp. 116-122.
[77] Y. Zhang, M. Chen, N. Guizani, et al., "SOVCAN: Safety-Oriented Vehicular Controller Area Network", *IEEE Communications Magazine*, vol.55, no.8, 2017, pp. 94-99.
[78] B. Zhao, H. Zhang, J. Li, et al., "The System Architecture and Security Structure of Trusted PDA", *CHINESE JOURNAL OF COMPUTERS*, vol.33, no.1, 2010, pp. 82-92.
[79] H. Zhang, B. Zhao. Trusted Computing[M]. Wuhan University Press, 2011.
[80] T. Wang, H. Chen, G. Cheng, "Research on software-defined network and the security defense technology", *Journal on Communications*, vol.38, no.11, 2017, pp. 133-160.
[81] Z. Shi, G. Zhu, "Research on security of software-defined networking", *Journal of Computer Applications*, vol.27, no.1, 2017, pp. 75-79.
[82] C. Zhang, Y. Cui, H. Tang, et al., "State-of-the-Art Survey on Software-Defined Networking (SDN)", *Journal of Software*, vol.26, no.1, 2015, pp. 62-81.
[83] H. Zhang, Z. Cai, Q. Liu, et al., "A Survey on Security-Aware Measurement in SDN", *Security & Communication Networks*, 2018, doi: 10.1155/2018/2459154.
[84] M.H.H. Khairi, S.H.S. Ariffin, N.M.A. Latiff, et al., "A Review of Anomaly Detection Techniques and Distributed Denial of Service (DDoS) on Software Defined Network (SDN)", *Engineering Technology & Applied Science Research*, vol.8, no.2, 2018, pp. 2724-2730.
[85] Y. Zhang, L. Cui, W. Wang, et al., "A survey on software defined networking with multiple controllers", *Journal of Network & Computer Applications*, vol.103, 2018, pp. 101-118.

Biographies

Yifan Liu, PhD candidate in the Wuhan University. Her main research interests include information security and trusted computing. Email: lyf_2017@whu.edu.cn

Bo Zhao, received his PhD degree in information security from the Wuhan University. Professor. His main research interests include information security and trusted computing. Email: zhaobo@whu.edu.cn

Pengyuan Zhao, PhD candidate in the Wuhan University. His main research interests include information security and trusted computing.



Peiru Fan, PhD candidate in the Wuhan University. Her main research interests include cloud computing and trusted computing.

Hui Liu, PhD candidate in the Wuhan University. His main research interests include image encryption and trusted computing.