FIRM: A Function-Independent Rule Management System for Mobile Security Function Chaining

Guanwen Li*, Bohao Feng, Huachun Zhou, and Guanglei Li
School of Electronic and Information Engineering
Beijing Jiaotong University, Beijing, 100044, China
{guanwen_li, bohaofeng, hchzhou, guangleili}@bjtu.edu.cn

Abstract

With the development of SDN/NFV technologies in 5G, the mobile security function chaining is supposed to meet the rapidly growing security requirements for mobile traffic. However, there are still many challenges to be addressed. Among them, how to correctly configure an ordered set of security functions is an important topic, but does not draw much attention from researchers. Thus, in this paper, we propose a framework of function-independent rule management (FIRM) with the corresponding security rule specification, which decouples the security rules with the specific functions and simplifies related configurations. The purpose is to alleviate misconfigurations of security rules related functions. We built the FIRM prototype and conduct several experiments. The related experim ent results confirm its availability and superiority.

Keywords: Security Function Chaining, Security Rule Specification, Function-independent Rule Management

1 Introduction

According to the Cisco Visual Networking Index (VNI) 2017 [6], there will be 5.5 billion global mobile users and generate 49EB mobile traffic per month by 2021. Moreover, as the rapid development of mobile communications, there is an explosive requirement growth of higher bandwidth, lower latency, and increased security. To meet these requirements, the concept of fifth generation mobile networks (5G) is proposed. However, 5G targets a variety of mobile use cases with a variety of specific security requirements, which needs flexible security mechanisms. Therefore, in such the architecture, with the key technologies Software Defined Networks (SDN) [14] and Network Function Virtualization (NFV) [5], required security functions are easily composed on demand in mobile security function chains to protect mobile traffic.

Mobile security function chaining allows the network security administrator composes common used basic network security functions flexibly for different mobile security requirements. There are some researches about the arrangement of security functions in the academia, such as [13] and [2]. However, the network security not only relies on the arrangement of security functions, but also the configuration of security rules. In fact, the misconfiguration of security rules is common and serious. For example, [1] finds that many firewalls deployed by experts are misconfigured because of the complexity of the manual configurations for firewalls. Due to the mobile security function chaining, the configuration of security rule is more complex because security rules may belong to many different types of security functions. One of the effective ways to alleviate the misconfiguration is to simplify the configuration by decoupling security rules with specific functions and actualizing automatic security rule management.

IT CoNvergence PRActice (INPRA), volume: 5, number: 4 (December 2017), pp. 1-11

*Corresponding author: Beijing Jiaotong University, No.3 Shangyuancun, Haidian District, Beijing 100044, China, Tel: +86-130-1118-1332

Thus, in this paper, we propose a function-independent rule management (FIRM) system with its corresponding specification of security rule configurations. The function-independent rule specification is used to describe security rules regardless of the configurations of specific security functions. Based on the proposed FIRM system, the configuration of security rules is automatic and simple for network security administrators. Therefore, the misconfiguration of security rules can be alleviated by the automatic and function-independent FIRM system.

On the other hand, based on SDN/NFV technologies, the mobile security function chain is usually built on the cloud datacenters at the edge of the network. However, in terms of the limited computing resources and low latency requirements for mobile users, a mobile security function chain may be deployed on distributed cloud datacenters. In other words, the mobile traffic may pass through several different cloud datacenters. Therefore, we propose the framework of FIRM system by taking distribution management into the consideration.

The contributions of this paper are as follows: Firstly, to decouple security rules with specific security functions, we present a function-independent security rule specification and describe it with YANG modeling language for automatic configurations. Secondly, we propose the framework of FIRM system, including its control plane and data plane. Thirdly, we present a use case of mobile security function chaining and verify the proposed FIRM system on our testbed.

The rest of this paper is organized as follows: In section 2, we review related works about security function chaining, especially security rule management. In section 3, we briefly introduce the mobile security function chaining. In section 4, we outline a function-independent security rule specification. In section 5, we describe the framework design of FIRM for mobile security function chaining. In section 6, we present a use case and the verification with the testbed. In section 7, we make a summary and discuss the future work.

2 Related Work

Since IETF Service Function Chaining (SFC) working group [11] was established, there are some researches about security function chaining in academia. For example, [13] presents a SFC based architecture for on-demand security services. [2] concludes the general network defense patterns with security function chaining.

However, most of the researches focus on optimizing the composition of security functions in a security function chain, such as [20]. Moreover, [15] proposes a framework for security function collaboration to mitigate large scale attacks. It designs a distributed mitigation system for dynamical security requirements, but has no consideration on security rules. [3] takes policy enforcement into considerations and proposes an architecture for permission control policy enforcement to solve the trust problem. Nevertheless, it cannot assign the security polices across different domains.

The most closed work is [18], which proposes a global security policy enforcement architecture for federated cloud environment. While, there are two differences from this paper. First, our proposed FIRM system and function-independent security rule specification can decouple security rule with specific security function and actualize automatic configuration, so that the misconfiguration of security rules can be alleviated. Second, there is a consideration about the characteristic of mobile traffic which may across many different domains in this paper. In a word, our proposed FIRM adapts to simplifying the security rule configuration and reducing misconfigurations for mobile security function chaining.

Therefore, we present the FIRM system with its function-independent security rule specification for mobile security function chaining in this paper.

3 Mobile Security Function Chaining

The core idea of mobile security function chaining is derived from service function chaining. However, in order to meet the security requirements of mobile users, required security functions and corresponding security rules should be considerate at the same time. Therefore, in this section, the architecture of mobile security function chaining is introduced, which consists of security function composition and security rule configuration.

For security function composition, there are four network elements, including classifier, security function (SF), security function forwarder (SFF) and SFC controller. Classifier is used to classify the mobile traffic based on pre-defined security classification criteria. With the classifier, there is an unique label called Service Path ID (SPI) to identify the mobile traffic. SF processes the traffic according to security rules and the order of SFs is denoted as a label called Service Index (SI). SFF is used to steer network traffic to next SFF or SF in sequence. SFC controller is in charge of planning the composition order of security functions based on the classification result from classifier.

For security rule configuration, there are some security functions and a security controller. The security rule is configurated by network security administrator on the security controller. The security function receives the security rule from security controller and carries it out.

To create a mobile security function chain for mobile traffic, SFC Controller analyzes the security requirement and creates an abstract security function composition at first. At the same time, according to the collected information of network topology, SFC Controller chooses required SFs and SFFs and obtains their location information. Then, the abstract security function composition is assigned to chosen SFs and SFFs. After that, SFC Controller tells the classifier the routing path of the received mobile traffic. Based on the location information of SFs, the Security Controller issues security rules configurated by the network security administrator to corresponding security functions. Once the mobile traffic pass through these security functions. the security rules are enforced.

Moreover, according to the latency requirement of mobile traffic, different security functions should be deployed close to the mobile user and the server separately. Thus, the mobile security function chain should be deployed across multiple SFC domains. Based on the Hierarchical Service Function Chaining (hSFC) [9] architecture, the mobile security function chain can be divided into several mobile security function sub-chains for different domains. Each domain is regarded as a sub-domain and has its own classifier, which is called Internal Boundary Node (IBN). In each domain, the mobile traffic enters and exits the sub-chain by its IBN. Besides, there is a logical top domain, which consists of all IBNs, some top-level SFFs and a special top-level classifier. In the top domain, each IBN is regarded as a SF.

For example, Figure 1 shows a mobile security function chain across through two domains. With the classification from CF0, the whole chain is divided into sub-chain1 and sub-chain2 with corresponding SPIs. After the classification, the traffic is steered to SFF1. Then, SFF1 forwards the traffic to IBN1 which is regarded as SF1 in the top domain. When the traffic enters the Domian1 by the IBN1, it passes follow sub-chain1 and finally returns to the IBN1. Next, the SFF1 forwards the traffic to SFF2. The traffic processing on SFF2 is similar to the SFF1. After the traffic returns from Domain2, it is forwarded to SFF3.

In summary, different from the service function chaining, the mobile security function chaining focuses on security and latency requirements for cross-domain mobile traffic.

4 Security Rule Specification

In this section, a function-independent security rule specification is defined to describe all the security rule possible in the security function chaining. First of all, a general security rule definition is proposed

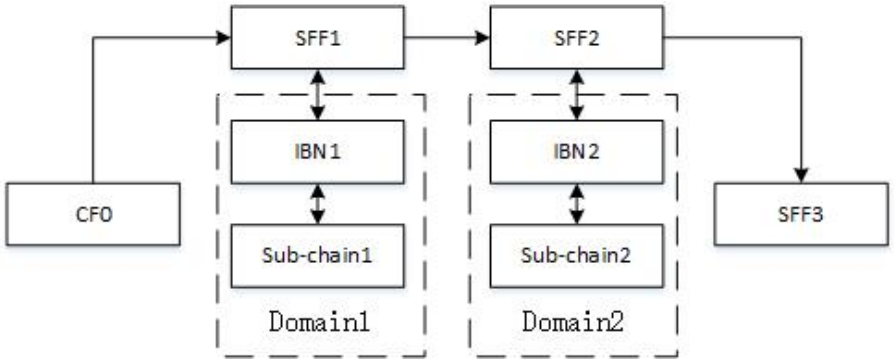

Figure 1: an example of hierarchical service function chaining

to decouple the security rule with the specific security function. Then, to actualize the automatic security rule configuration with NETCONF protocol, an example YANG modules is given for the configuration of IDS rules.

4.1 Security Rule Definition

In our notation, R denotes the security rule which consists of the type of security function, the id, the matching field, the action and the priority.

$$R(type, id, match, action, priority) \tag{1}$$

Because of the diversity of security functions and their corresponding security rules, the format of the matching field R.match is a matching set as follow:

$$R.match = \{srcmac, dstmac, srcip, dstip, proto, srcport, dstport, app\} \tag{2}$$

where srcmac and dstmac denote the source and destination MAC addresses, scrip and dstip denote the source and destination IP addresses, proto denotes the protocol type of transport layer, srcport and dstport denote the source and destination ports, and app denotes the protocol type of application layer.

In terms of typical security functions, the available value of R.action is an action set as follow:

$$R.action = \{accept, drop, log, alert, pass, translate, \cdots\} \tag{3}$$

There are common used actions of various security functions, such as firewall, intrusion detection system (IDS), network address translation (NAT) and etc. For instance, the “accept” and “drop” are often used for firewall rules, while the “alert”, “pass” and “log” can be used for IDS rules.

4.2 YANG Module Example

In order to actualize automatic configuration, security rules are enforced by using the IETF standardized NETCONF [10] protocol with its specific data modeling language YANG [4]. For example, a YANG module for IDS is shown in Figure 2.

It shows a full YANG module for IDS. In line 1, the module name is defined as security-rule. In the header information (line 2-4), there are namespace statement and prefix statement. To identify the type of security function, this prefix statement is labeled as ids. In the linkage statements (line 5-10), some neccessary linkages are imported, including ietf-inet-types, ietf-yang-types and tailf-common. Next, in the revision history (line 11-12), there are the version information and its description. The final and

```
1  module security-rule{
2  namespace "http://iplab.com/security-
3  policy";
4  prefix ids;
5  import ietf-inet-types {
6    prefix inet;}
7  import ietf-yang-types {
8    prefix yang;}
9  import tailf-common {
10   prefix tailf;}
11 revision 2017-05-27 {
12   description "Test";}
13 container rule{
14   tailf:callpoint hcp;
15   list ids-classification-rule {
16     key rule-name;
17     max-elements 64;
18     leaf rule-name {
19       type string;
20       mandatory true;}
21     leaf rule-id {
22       type uint32;
23       mandatory true;}
24     container condition {
25       leaf-list traffic-char {
26         type string;
27         min-elements 0;
28         max-elements 256;}}
29     container action {
30       container action-type {
31         leaf accept {
32           type boolean;}
33         leaf drop {
34           type boolean;}}}}
35     leaf rule-priority{
36       type uint32;
37       mandatory true;}}}}
```

Figure 2: an example YANG module for IDS

important part is the container called rule (line 13-37). Because the security rule consists of the type of security function, the id, the matching field, the action and the priority, the list ids-classification-rule id is defined with the following 5 parts. The first part (line 18-20) is leaf rule-name, which represents the name of the security rule. The rule-name is the key leaf in the list, which is a string used to identify this security rule. The second part (line 21-23) is leaf rule-id, a random assigned integer when the rule is created. The third part (line 24-28) is container condition, which represents the matching field. There are different patterns of matching field for different types of security function. In this instance, matching field of IDS is represented as leaf-list traffic-char. The fourth part (line 29-34) is container action representing a specific action. The last part (line 35-37) is leaf priority, which can be calculated by the SPI and the SI of the corresponding security function in this security function chain.

5 Security Rule Management

In this section, the framework of the FIRM system is presented, especially the design of the security controller. As shown in Figure 3, the framework comprises control plane and data plane.

5.1 Control Plane

On the control plane, the Web UI is designed to human-faced security rule configuration, which is independent from the specific security function. When the network security administrator wants to deploy or update a security rule, he/she should login in the website with the authorization at first.

Then, the web page jumps into a security rule configuration page. The configuration page is used to receive human-read security rule. For example, when the network security administrator wants to add a firewall rule to the mobile security function chain, he/she should choose the function type as firewall and fill in the rule name, the matching field (such as source and destination IP address), and the action (accept or drop) on this page.

Afterwards, the web engine translates the received security rule into a kind of machine-read data structure, including the type of security function, the rule name, the rule id, the matching field, the action

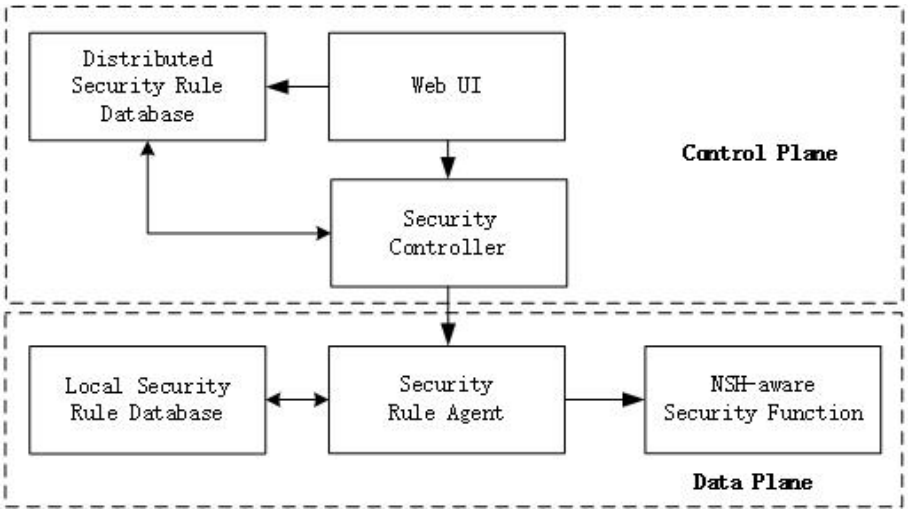

Figure 3: the framework of security rule management

and the rule priority. At the same time, the web engine generates a random id for this rule. The rule priority is calculated based on the SPI of the security service chain and the SI of this security function which can be obtained from the external SFC Controller.

Next, the data structure is stored into a Distributed Security Rule Database, which can be shared with other security controllers in different SFC domain.

The design of Security Controller is shown in Figure 4. There is a Listener running all the time. When the operation of data storage is finished, the security controller is activated to read the security rule from the database and prepare to translate it into the form of proposed security rule specification.

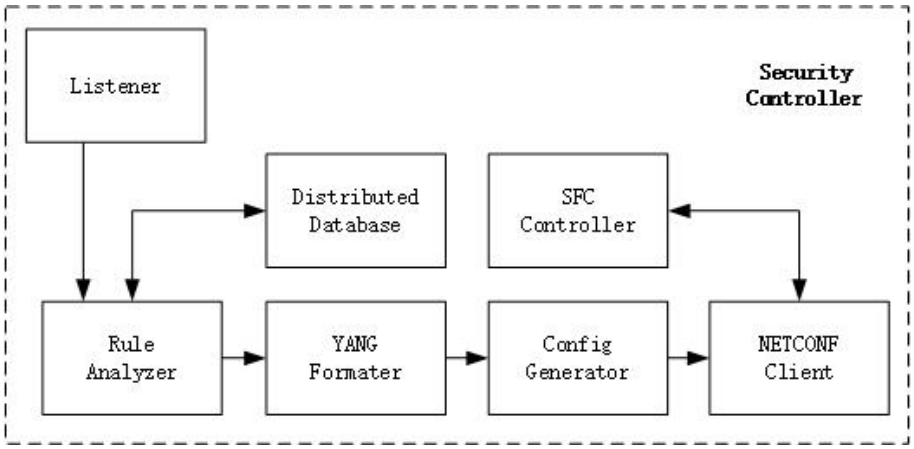

Figure 4: the design of Security Controller

According to the complexity and diversity of the matching field in a security rule, the Rule Analyzer formats keywords (such as source and destination addresses and protocol type) in a matching field based on the type of the security function.

Later, the YANG Formatter translates the composition of type of security function, the id, the matching field, the action and the priority with YANG model and calls Config Generator to save the security rule as a XML format file.

Finally, the Security Controller issues this security rule by the NETCONF Client with the location

information from the external SFC Controller. There is a connection between control plane and data plane with NETCONF protocol.

5.2 Data Plane

On the data plane, there is a security rule agent running on each security function node. The security rule agent not only receives the security rule from control plane, but also sets the security rule to the chosen security function.

The design of security rule agent is based on a NETCONF Server. With the predefined YANG modules, the security rule agent can resolve the received security rule with NETCONF protocol.

After the resolving of the security rule, it is stored into a Local Security Rule Database as a backup.

Finally, the resolved security rule is translated into the format which can be read by the specific security function and written to the local security function configuration.

6 Implementation

6.1 Use Case

The Mobile-Edge Computing (MEC) [17] is one of the most important parts for future mobile network architecture, which deploys as many as possible mobile network services and applications on the data-center at the edge of network. Taking computing resources and geographic factors into consideration, these cloud datacenters are often located closed to the user side and the server side respectively, providing higher bandwidth and lower latency for mobile traffic.

As shown in Figure 5, there are two cloud datacenters providing security functions for mobile traffic. The first datacenter DC-1 is next to the user mobile network, while the second datacenter DC-2 is next to the server. There is a connection between two datacenters across through the Internet. According to the security requirement and guaranteed performance, the whole mobile security function chain is divided into two security function sub-chains distributed in different datacenters. To protect user privacy, there is a composition of firewall and IDS deployed in DC-1, which belongs to the first sub-chain. To defense the malicious attack from the Internet, there is another composition of firewall and IDS deployed in DC-2, which belongs to the second sub-chain. There is a global FIRM system is deployed across different domains but not illustrated in the figure.

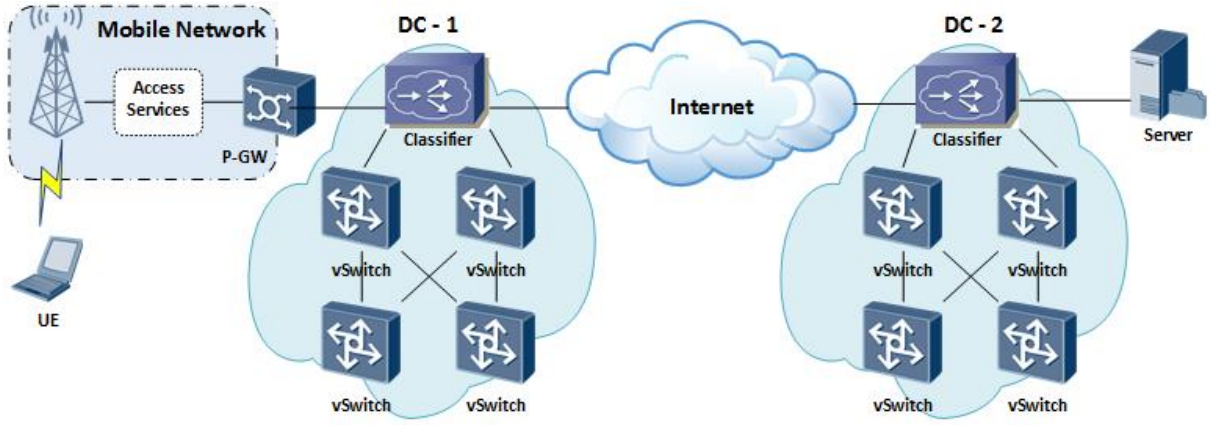

Figure 5: the use case for mobile security function chaining

We implement these two cloud datacenters based on OpenStack [8] testbed. To simplify the experimental topology, each datacenter consists of one gateway and four virtualized switches. The gateway is regarded as the classifier in each datacenter, which is used to classify mobile traffic and steer them into the security function sub-chain. Each virtualized switch is regarded as a Security Function Forwarders (SFF), which can be deployed required security functions. With Docker virtualization technologies [12], we deploy the security functions in the form of container image, including firewall image and IDS image. In this case, the firewall image is made based on iptables [16], and the IDS image is made based on Snort [7]. Additionally, refer to the definition of [19], the flow in the mobile security function chain is encapsulated by NSH. With NSH-aware security functions, the NSH flow can be processed as same as normal IP flow natively. However, there is no existed security function configuration supporting NSH flow. Thus, in order to enforce the security rule in NSH-aware security function, we also extend the rule sets of the firewall and the IDS.

6.2 Verification

6.2.1 Firewall Rule Configuration

Figure 6 shows the Web UI of firewall rule configuration whose function type is chosen as firewall. On that page, the network security administrator can customize the rule name, matching field (including source address, destination address and protocol), and action. At that time, the security rule is independent from the final configuration of the firewall.

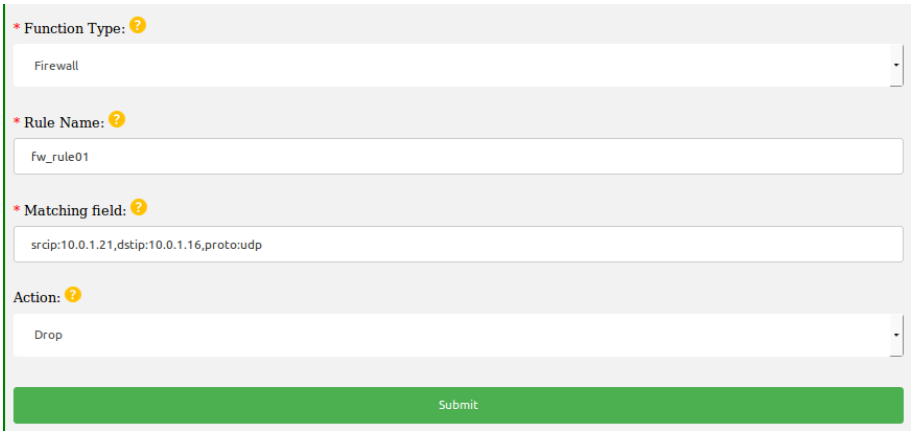

Figure 6: the Web UI of firewall rule configuration

After the web page is submitted, the security rule is processed by Security Controller and assigned to the corresponding security function. Finally, the received firewall rule shown in Figure 7 is add to the iptables.

```
Chain INPUT (policy ACCEPT)
target     prot opt source               destination
DROP       udp  --  10.0.1.21           10.0.1.16
```

Figure 7: the configured firewall rule

6.2.2 IDS Rule Configuration

Similar to the firewall, Figure 8 shows the Web UI of IDS rule configuration whose function type is chosen as IDS. Because the matching field of IDS is more complex than that of firewall, the web page allows default configurations for the matching field. At that time, the security rule is independent from the final configuration of the IDS.

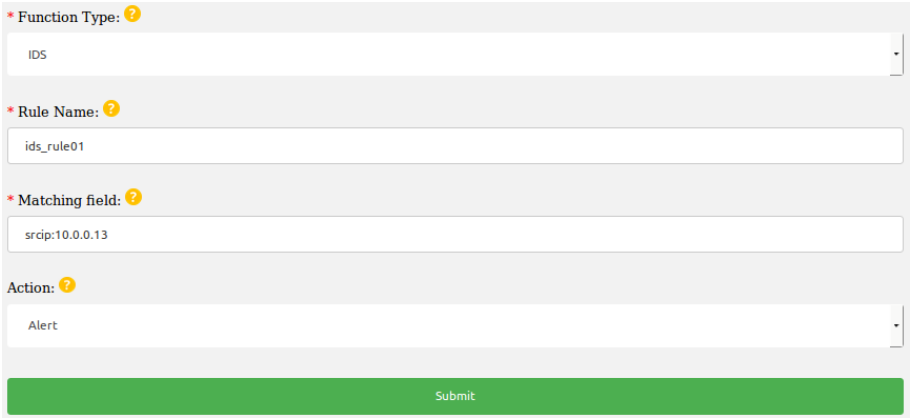

Figure 8: the Web UI of IDS rule configuration

After the web page is submitted, the security rule agent receives the IDS configuration. Finally, the new IDS rule shown in Figure 9 is add to the extended rule set of Snort.

```
#------------
# TEST RULES
#------------
#
alert udp 10.0.0.28/32 any -> 10.0.0.27/32 any (msg:"Outer IP!";sid:10000001;)
alert ip 192.168.0.10/32 any -> 192.168.1.23/32 any (msg:"Inner IP!";sid:10000002;)
alert ip 10.0.0.13/32 any -> any any (sid:1495847132;)
```

Figure 9: the generated IDS rule set

To summarize, we implement the function-independent configuration of security rules on our testbed and validate our proposed the FIRM system by experiments.

7 Conclusion

In this paper, we present a function-independent security rule specification to describe different kinds of security rule in a mobile security function chain with YANG modeling language for automatic configuration. We also design the framework of our proposed function-independent rule management (FIRM) system for mobile security function chaining including both control plane and data plane. A related prototype is implemented and experiment results confirm its availability and superiority.

However, the propped FIRM still has some limitations. For instance, the collaboration of different security rules in a mobile security function chain may lead to security rule anomalies. Therefore, an effective approach is required to resolve the possible conflicts between two more security rules with different types of security functions. Furthermore, although the automatic security rule configuration is actualized, the security rules still depend on the network security administrators. Thus, a way to decide required security rule automatically by the characteristics of mobile traffic is also needed.

Acknowledgments

This paper is supported by NSAF under Grant No. U1530118, National High Technology of China ("863 program") under Grant No. 2015AA015702, NSFC under Grant No. 61602030, National Basic Research Program of China ("973 program") under Grant No. 2013CB329101 and the Fundamental Research Funds for the Central Universities under Grant No. 2017YJS001.

References

[1] S.-S. Alireza, J. Yosr, P. Makan, and C. Mohamed. A quantitative study of firewall configuration errors. *Computer*, 37(6):62–67, June 2004.

[2] S.-S. Alireza, J. Yosr, P. Makan, and C. Mohamed. Efficient provisioning of security service function chaining using network security defense patterns. *IEEE Transactions on Services Computing*, PP(99):89–103, January 2016.

[3] V. Anh-Vu and K. YoungHan. A network service permission control platform for service function chaining. In *Proc. of the 2017 International Conference on Information Networking (ICOIN'17), Da Nang, Vietnam*, pages 151–156. IEEE, January 2017.

[4] M. Bjorklund. Yang - a data modeling language for the network configuration protocol (netconf). IETF RFC 6020, October 2010. `https://tools.ietf.org/html/rfc6020`.

[5] M. Chiosi et al. Network functions virtualisation–introductory white paper. Technical report, The European Telecommunications Standards Institute, 2012.

[6] Cisco. Cisco visual networking index: Global mobile data traffic forecast update, 2016–2021. Technical report, CISCO, 2017.

[7] Cisco. snort, 2017. `https://www.snort.org` [Online; Accessed on October 3, 2017].

[8] R. C. Computing. Openstack, 2017. `https://www.openstack.org` [Online; Accessed on October 3, 2017].

[9] D. Dolson, S. Homma, D. Lopez, M. Boucadair, D. Liu, T. Ao, and V. Vu. Hierarchical service function chaining (hsfc). Technical report, January 2017. `http://www.ietf.org/internet-drafts/draft-ietf-sfc-hierarchical-04.txt` [Online; Accessed on October 3, 2017].

[10] R. Enns. Netconf configuration protocol. IETF RFC 4741, December 2006. `https://tools.ietf.org/html/rfc4741`.

[11] J. Halpern and C. Pignataro. Service function chaining. IETF RFC 7665, October 2015. `https://tools.ietf.org/html/rfc7665`.

[12] D. Inc. Docker, 2017. `https://www.docker.com` [Online; Accessed on October 3, 2017].

[13] C. Li-Der, T. Chia-Wei, H. Yu-Ki, C. Kuo-Chung, O. Tsung-Fu, and Y. Chia-Kuan. A security service on-demand architecture in sdn. In *Proc. of the 2016 International Conference on Information and Communication Technology Convergence (ICTC'16), Jeju, South Korea*, pages 287–291. IEEE, October 2016.

[14] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner. Openflow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review*, 38(2):69–74, March 2008.

[15] D. Migault, M. A. Simplicio, B. M. Barros, M. Pourzandi, T. R. M. Almeida, E. R. Andrade, and T. C. M. B. Carvalho. A framework for enabling security services collaboration across multiple domains. In *Proc. of the 37th International Conference on Distributed Computing Systems (ICDCS'17), Atlanta, Georgia, USA*, pages 999–1010. IEEE, June 2017.

[16] netfilter core team and contributors. iptables, 1999-2014. `http://www.netfilter.org/projects/iptables/index.htm` [Online; Accessed on October 3, 2017].

[17] M. Patel, B. Naughton, C. Chan, N. Sprecher, S. Abeta, A. Neal, et al. Mobile-edge computing – introductory technical white paper. Technical report, The European Telecommunications Standards Institute, 2014.

[18] M. Philippe, D. Laurent, A. Amel, D. Sebastien, L. Anna, and V. Massimo. End-to-end security architecture for federated cloud and iot networks. In *Proc. of the 2017 IEEE International Conference on Smart Computing (SMARTCOMP'17), Hong Kong, China*, pages 1–6. IEEE, May 2017.

[19] P. Quinn and U. Elzur. Network service header. Technical report, July 2017. `http://www.ietf.org/internet-drafts/draft-ietf-sfc-nsh-16.txt` [Online; Accessed on October 3, 2017].

[20] V. P. Trung, B. Nguyen, Khac, K. Youngpin, L. Hyun-Jin, and P. Minho. Optimizing resource allocation for elastic security vnfs in the sdnfv-enabled cloud computing. In *Proc. of the 2017 International Conference on Information Networking (ICOIN'17), Da Nang, Vietnam*, pages 163–166. IEEE, January 2017.

Author Biography

Guanwen Li received the B.S. and M.S. degrees in Software and Information science from Iwate Prefectural University in 2004 and 2006, and Ph.D. degrees in the same University in 2010. Currently he is an assistant professor in the Ibaraki University. His research interests include Web Geographic Information System for local governments, Disaster Management System, Safety Confirmation System, Regional Disaster Prevention Planning, Virtual Reality and Tele-Immersion. He is a member of IEEE, Virtual Reality Society of Japan (VRSJ).

Bohao Feng received the B.S. degree in Engineering from Ibaraki University in 2014. Currently he is taking a master's course at Graduate School of Science and Engineering, Ibaraki University. His research interests include Web Application System, Geographical Information System and Crisis Management.

Huachun Zhou received the B.E. and M.E. degrees from Iwate Prefectural University in 2010 and 2012, respectively. From April 2012, he works at Iwate Monozukuri Software Integration Technology Center at Iwate Prefectural University, Iwate, Japan. And now, he is taking a doctor's course at Graduate School of Software and Information Science, Iwate Prefectural University. His research interests include Tiled Display System, Disaster Prevention Information System and Geographical Information System. He is a member of Information Processing Society of Japan (IPSJ) and Virtual Reality Society of Japan (VRSJ).

Guanglei Li received the B.S. degrees from University of Tennessee in 1994, M.S. degrees in Software and Information science from Iwate Prefectural University in 2003, and Ph.D. degrees in the same University in 2011. Currently he is an associate professor in the Saitama Institute of Technology. His research interests include Cognitive Wireless Networks, QoS, and Heterogeneous Network. He is a member of IEEE, Information Processing Society of Japan (IPSJ), and Institute of Electronic and Communication Engineering in Japan (IEICE).