A Dynamic Deceptive Honeynet System with A Hybrid of Virtual and Real Devices

Zheng Minjiao
School of Information and Communication, National University of Defense Technology
Wuhan, China 430010
mjzzheng@126.com

Ma Yufeng
School of Information and Communication, National University of Defense Technology
Wuhan, China 430010

Wu Bo
School of Information and Communication, National University of Defense Technology
Wuhan, China 430010

Qian Zhang
Experimental Training Base, National University of Defense Technology
Xian, China 710000
cc_2022_cc@163.com

2022 5th International Conference on Computing and Big Data (ICCBD) | 978-1-6654-5716-3/22/$31.00 ©2022 IEEE | DOI: 10.1109/ICCBD56965.2022.10080304

***Abstract*—Honeynet, as a representative network deception technology, can protect the network while attracting attackers to understand their patterns, strategies and behaviors. The new honeynet based on the latest bility, but the ability to construct honeynet scene dynamically is still a little bit insufficient at the moment. We propose a dynamic deception honeynet architecture combining virtual and real honeypot, which is used to dynamically generate a real network according to the cyber situation and the attack behavior to attract attackers. It redirects the advanced attacker to the environment that he is more interested, in order to capture the attacker and conduct further analysis. This improves the lack of dynamic and inductivity in the existing honeynet. Finally, we de-veloped the prototype system and set up the experimental environment. The result shows that the system has a high degree of intelligence and strong in-ductivity.**

Keywords—SDN, Docker, traffic migration, honeynet generation, Multi-type Honeypot

I. INTRODUCTION

Honeynet is a simulated network composed of multiple honeypots[1]. By combining multiple honeypots, a trap network similar to the real business network is formed. It comprehensively captures and monitors all incoming traffic to the architecture and detects attacks[2][3]. Honeynet network configuration is highly controllable and host functions are rich and varied. Therefore, it can collect and sample various types of attack information. The information can be used to collect the attack behaviors of attackers and update related security policies[4-6].

After years of development, there are three trends in the research and application of honeynet system. One is to enhance the simulation of honeypot network through more fine-grained business environment simulation[7][8]. The second is to analyze the attacker's behavior with Al and other new technologie[9-12]. Third, mimicry feature construction and dynamic evolution technology appeared to improve the adaptive ability of the new honeypot[13-16]. However, in the existing open honeypot schemes, the dynamic strategy is limited by the deployment scheme and other factors, and is often determined with the first deployment of honeypot network, which loses part of the dynamic. At the same time, for multi-stage advanced attacks, the ability to construct honeynet scenarios is still a little insufficient.

Therefore, this paper gives full play to the advantages of the mixture of physical and virtual honeypots, combines the technical flexibility and flexibility characteristics of SDN and container, and puts forward a technical scheme of virtual and real combination of deception honeynet system, which improves the lack of dynamic generation of existing honeynet and further improves the decoy of honeynet. Descibes as follows:

The traffic migration mechanism synchronizes and mirrors the traffic destined for the real service network to the attack filtering honey network for detection through traffic redirection. Disconnects the attack traffic from the service net-work after an attack is detected. Intelligent updates of dynamic honey network nodes, topology, and other attributes, while migrating traffic to the desired environment of the attacker.

The honeynet generation mechanism can dynamically generate virtual honey-pots containing specific services. It can unify the management and scheduling of virtual and real honeypot. The generation mechanism combined with the genera-tion mechanism can generate dynamic adjustment strategy according to the cur-rent situation of network and the result of attack perception. In this way, the deception of the attacker and the active trapping of the attack can be realized, and the resources can be saved to the greatest extent.

The structure of this paper is arranged as follows: The second section proposes a virtual and real combination based honeypot architecture, including traffic migration mechanism, multi-type honeynet generation mechanism and principle of honeynet generation mechanism. Section 3 presents experimental and test results; Finally, the fourth part puts forward some conclusions and suggestions for future work.

II. VIRTUAL AND REAL COMBINATION BASED HONEYPOT ARCHITECTURE

The virtual and real combined honeypot architecture consists of four parts: Openflow switch and management server, virtual honeypot server and real honeypot group, as shown in Fig.1. The real service network connected to the Openflow switch. Visitors access the real service network through openflow switch and security protection device such as firewall. The management server, which contains the security controller, the network controller and the policy generator components, implement the

migration and honeynet generation mechanism. The migration mechanism makes decisions on attack traffic. The honeynet generation mechanism implements a dynamic transformable multi-type honeynet that uses the openflow switch, real honeypot group and OVS, honeypot in virtual honeypot server. The real honeypots are real devices on the network. Owing to their high interactivity, they can capture high-value attacks, but they are vulnerable to attack and difficult to maintain. At the same time, the honeypot in virtual honeypot server includes containers or virtual machine forms, which can be generated on demand, flexible and scalable. It can be used to build large-scale honeypot networks and simulate various large-scale networks or systems. The migration mechanism and the honeynet generation mechanism will be introduced below.

A. Traffic Migration Mechanism

The traffic migration mechanism is designed and implemented based on SDN network framework. In this framework, the control layer, which contains SDN controller as its core component, is the operating system of the network device in the infrastructure layer. RYU, as the SDN controller, can deliver "flow table" to network device, including hardware and virtual SDN switching device, based on openflow protocol to guide data forwarding and collect network flow table information. The network device receives the flow table and performs corresponding actions by matching each entry in the flow table to implement traffic redirection.

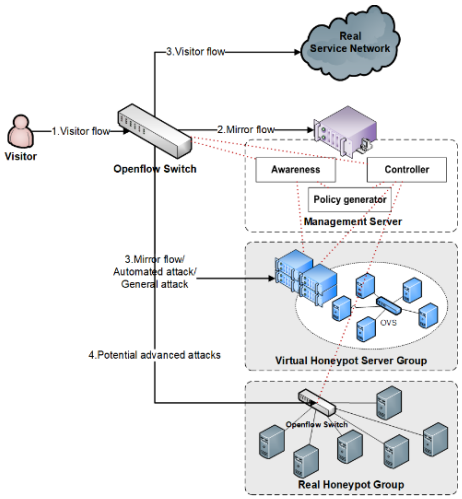


Fig. 1. Traffic migration

As shown in Fig.1, When the traffic passing through the network security device such as firewall, it reaches the openflow switch. Under the unified deployment of the SDN controller, the traffic is mirrored into the attack filtering honey network in the virtual Honeypot Server while accessing the real business network. The honeypot in the attack filtering honeypot can only perform security detection and does not interact with the traffic before the attack behavior is detected. When abnormal behavior is perceived, follow the following procedure.

The proxy module in the honeypot sends the relevant behavior information to the perception module of the controller. The perception module identifies the attack and records the event, while pushes the information to the policy generation module.

The policy generation module is responsible for analyzing the event information. If it is an attack event, the traffic is disconnected from the real service network and interacts with the honeynet. Then the policy generation module predicts the next behavior according to the event information, generate the honeynet dynamic update policy, and push it to the controller module.

The controller dynamically updates the node configuration and network topology of the honeynet based on the Dynamic Update Policy, enabling the honeynet to adapt to the attacker's behavior. The controller migrates traffic to the desired environment of the attacker, by modifying IP addresses and MAC ad-dresses corresponding to OVS or openflow switches flow tables. Meanwhile, the controller determines which honeypots to enable and what services these honey-pots contain. by modifying honeypot configuration files. For automation and general attacks, honeynets usually consist of virtual honeypots. However, for the potential advanced attackers, it may be necessary to generate a trap net combining by real and virtual honeypots.

B. Multi-type Honeynet Generation Mechanism

Multi-type honeypot generation mechanism is proposed to solve the following problems. For example, the functional services provided by honeypot have been fixed when the honeypot is built, and the honeynet only contains virtual honeypot with more flexibility or real honeypot with more simulation. In order to generate a more targeted and effective honeynet and better deal with complex and dynamic network scenarios, it is necessary to break the original static and single characteristics. Thus the honeynet has high simulation and flexibility, and can better adapt to the complex and changeable real scene.

Based on the honeypot function dynamic construction technology, multi-type honeypot unified management technology and honeynet topology generation technology, multi-type honeynet generation mechanism can dynamically generate attack filter honeynet or trap honeynet with different characteristics. These honeynets are composed of honeypots or network device in the form of different resources.

Thereof, the honeypot function dynamic construction technology, mainly re-lies on docker, dynamically construct the honeypot function through modular combination, multi-service aggregation, custom deployment and so on. Different services/software are encapsulated in different docker image layers, so honeypot can be composed of different image layers. After executing the start command, honeypot added the read-write container layer on the image layer, and used unionFS and OverlayFS technology to create the file system corresponding to the image. The startup of different services/software can also be controlled by modifying scripts/configuration files. After the container is started, execute the submit command in the container to submit the top read-write container layer to the image layer and generate a new image. Through this framework, as shown in Fig. 2, modularized dynamic management of honeypot function is realized.

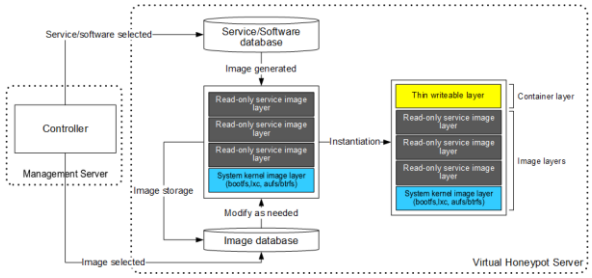


Fig. 2. The honeypot function dynamic construction technology

As the Fig.3 shown, the multi-type honeypot unified management technology can uniformly manage honeypots of different resource forms and different interaction degrees, such as real host, virtual host and containerized service. Multi-type honeypot unified management technology abstracts host resources of various types of honeypot into Pot, manages, schedules arranges trapping scenes and perceives attacks in a unified way.

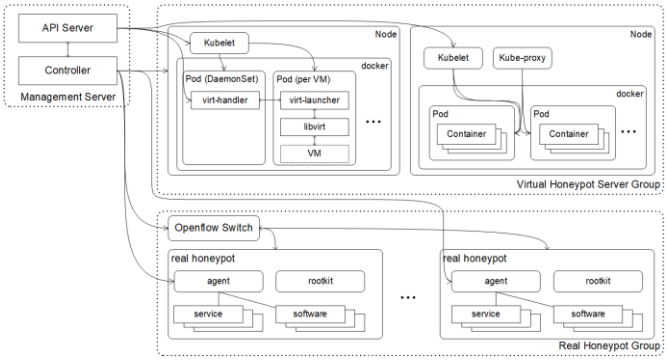


Fig. 3. The multi-type honeypot unified management technology

Dynamic honeypot function construction and multi-type honeypot unified management technology constitute the honeypot dynamic management framework. Based on this framework, the honeypot can be customized, dynamically generated and flexibly deployed, so that the honeypot can be dynamically generated at the node level. Furthermore, the honeynet topology generation technology can dynamically generate topologies.

The honeynet topology generation technology, represented in Fig. 4 below, is based on SDN framework and uses distributed RYU controller to uniformly manage virtual or real Openflow switches. The control layer consists of a central controller and several area controllers. The central controller interacts with the policy generation module to generate control commands and deliver them to area controllers. The area controller allocates infrastructure layer resources, including real switches and honeypots, and virtual switches and honeypots deployed on virtual honeypot server group. In this way, the topology is dynamically generated.

For docker-based honeynet, combined with Kubernetes technology, topology generation is more efficient. Use Calico as a networking plug-in for Kubernetes. Change the topology by modifying interfaces or configuration files to orchestrate honeypot startup, destruction, and migration operations.

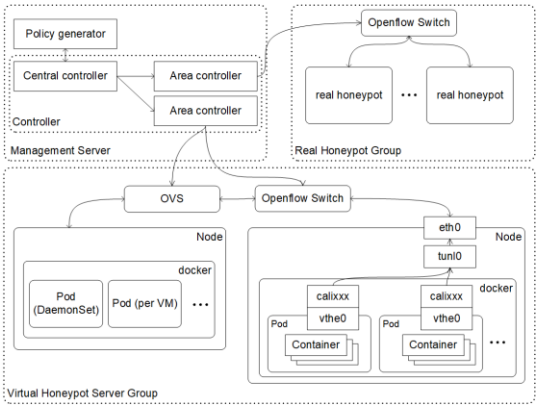


Fig. 4. The honeynet topology generation technology

C. *Principle of Honeynet Generation Mechanism*

In order to generate or dynamically change honeynet against the attack of the system, principle of honeynet generation mechanism is proposed as the Fig. 5 shown. Thereof, attack perception is the basis, which integrates a variety of perception technologies to discover the attacker's attack behavior without being aware of it. Policy recommendation is the core, which integrates the collaborative filtering algorithm based on attacker and attack behavior to predict the honeypot service that attackers may be interested in. The attack surface model is used to predict the honeypot topology attributes that attackers are interested in. To sum up, the operation vector of active trapping component is generated, and the honeypot service and network topology to be recommended are determined by comprehensive decision making method. After conflict detection, the final honey network adjustment policy is delivered, and response data is collected for model iterative update.

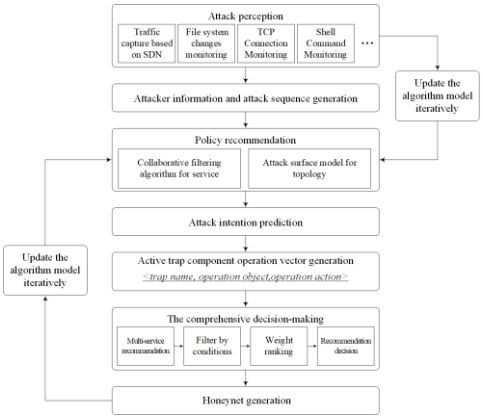


Fig. 5. The principle of honeynet generation mechanism

III. EXPERIMENT

As shown in Figure 6, an intelligent honeypot system was developed and deployed, including openflow switch, real honeypot and a server deploying management platform and virtual honeypot resources, based on the above honeypot architecture. The real service network includes network segments 10.2.1.0/24, 10.2.1.0/24, and 192.168.100.0/24. Network segments 10.2.1.0/24 and 10.2.1.0/24 can communicate with each other, but 192.168.100.0/24 cannot communicate with other two network segments.

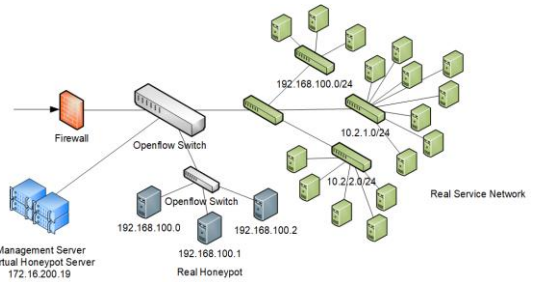
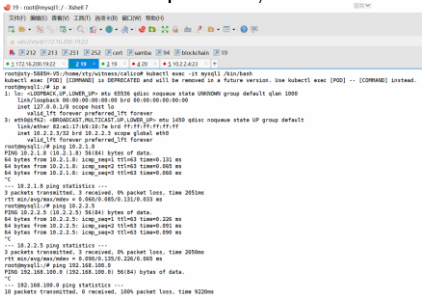


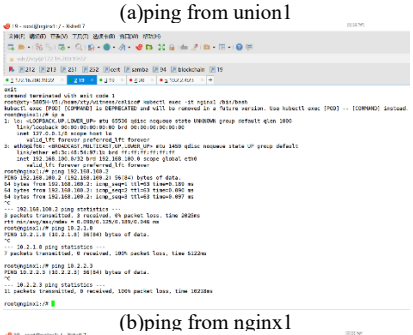
Fig. 6. The topology for experiment

Configure the honeynet topology consistent with the real service network, and deploy the corresponding honeypot service consistent with the service server.

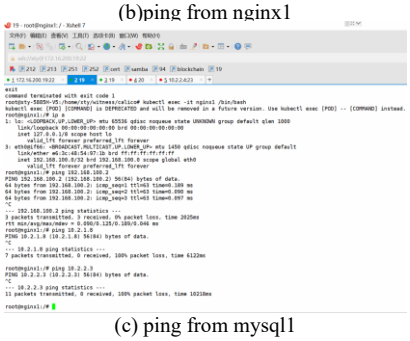
Enter honeypot node union1 (10.2.1.8), nginx1 (192.168.100.0) and mysqL1 (10.2.2.3) respectively from the system background, and use the ping command to check the connectivity between the honeypot node and other honeypot nodes. As shown in Figure 7, network segment 10.2.1.0/24 and 10.2.1.0/24 can be connected, and network segment 10.2.1.0/24, 10.2.1.0/24 cannot connect with 192.168.100.0/24. The honey network has the same interoperability as the real service network.



(a)ping from union1



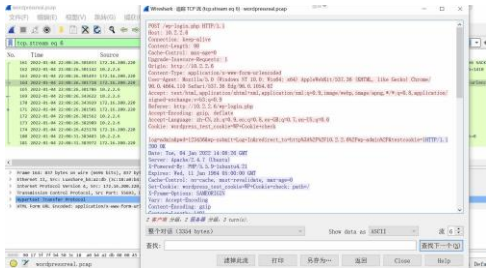
(b)ping from nginx1



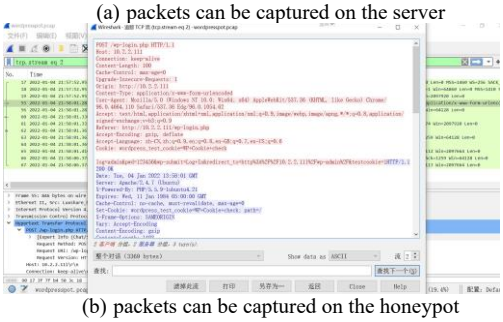
(c) ping from mysql1

Fig. 7. Ping result from different network segment

Access the wordpress service deployed on the server (10.2.2.6) in the real service network and capture packets. The same network packets can be captured on the corresponding node (10.2.2.111) in the honey network. The results are represented in Fig. 8 below:



(a) packets can be captured on the server



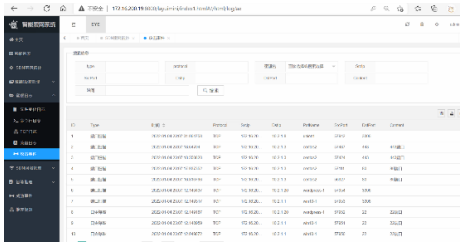
(b) packets can be captured on the honeypot

Fig. 8. Protocol reply after access simulation

The simulated attacker launches a port scanning attack, and the honeynet system successfully captures the port scanning attack and generates a new intelligent inference record. In this case, the traffic between the system and the real service network is cut off, and the honey network responds. In addition, honeypot topology has changed, honeypot Centos2(10.2.1.3) has been deleted, and a honeypot Wordpress2(10.2.2.7) has been added to network segment 10.2.1.0/24.

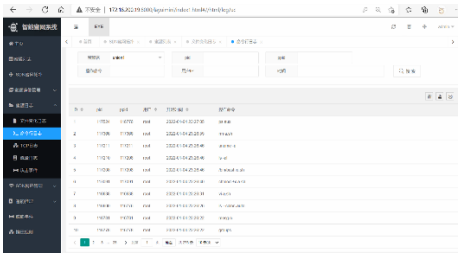
Simulated attacker launches a command blast attack. After the root login password is obtained, the simulated attacker creates files, modifies file contents, and executes file scripts. The system records the attack behaviors in the password blasting event logs, file change logs, command line logs, and TCP connection logs. The logs are shown in fig.9 below.

As shown in Fig. 10, a new intelligent inference record is generated, the honeynet topology is changed, and a new trap honeypot(10.2.2.8) containing SSH services is created now.

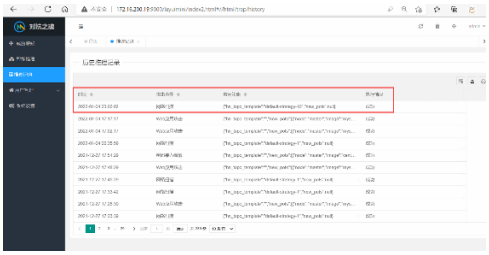


(a)events logs after the attack

(b)file change logs after the attack

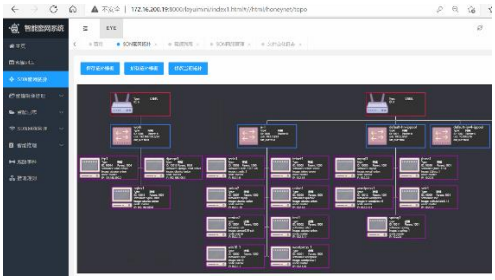


(c)command line logs after the attack

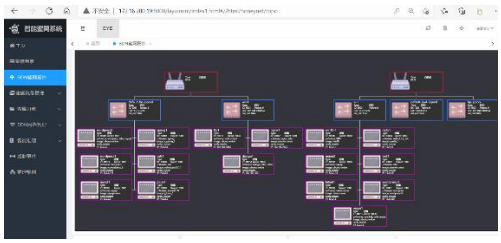


(d)policy logs after the attack

Fig. 9. Part of logs after the attack



(a)before the attack



(b)after the attack

Fig. 10. The topology of the honeynet

Through the experiment, it can be seen that the honeynet can be dynamically generated. The honeynet can mirror the traffic to the real service network in real time and sense the attack behavior. After being attacked, the honeynet can take corresponding measures, including cutting off the relationship between attack traffic and the real service network, recording the behavior data and dynamically updating the honeynet scenario according to the preset policy to trap the attack.

IV. Conclusion

Combining the characteristics of virtual honeypot and real honeypot, based on SDN and container technology, a new honeypot system architecture combining virtual and real is proposed for traffic transfer and multi-type honeypot dynamic generation. The system can dynamically generate a large and high simulation trap according to the condition of protected network. The honeynet can deceive the attacker, and dynamically adjust the network topology and the composition and type of nodes according to the behavior of the attacker in the whole attack process. It improves the traditional honeynet dynamic and inducible lack of problems. Finally, an experimental environment was built and a prototype system was developed to verify the proposed mechanism. The results show that the system has high intelligence and strong concealment.

In the future, we will further improve the mechanism of honeynet generation, improve the model in the real environment, and further strengthen the deceptive honeynet.

References

[1] PARK B, DANG S P, NOH S,et al.: Dynamic virtual network honeypot[C] In: Proceedings of the 2019 International Conference on Information and Communication Technology Convergence (ICTC). pp. 375-377 (2019) .

[2] SHI L Y, CUI Y W, HAN X, et al. Mimicry honeypot: An evolutionary decoy system[J]. International Journal of High Performance Computing and Networking. 14(2), 157-164 (2019).

[3] Lian Z, Yin X, Xi X I, et al: SDN Virtual Honeynet Based on Mimic Defense Mechanism[J]. Computer Engineering and Applications(2019).

[4] Wonkyu Han, Ziming Zhao, Adam Doupé, et al.HoneyMix: Toward SDN-based Intelligent Honeynet [C]. Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, pp.1-6 (2016).

[5] Manzano A, Naranjo J, Be Rnal I, et al: A prototype for a honeynet based on SDN[C]. Telematics & Information Systems, pp.1-8(2016).

[6] Kyung S, Han W, Tiwari N, et al. HoneyProxy: Design and implementation of next-generation honeynet via SDN[C]. 2017 IEEE Conference on Communications and Network Security.(2017).

[7] Pohl C, Zugenmaier A, Meier M, et al. B.Hive: A Zero Configuration Forms Honeypot for Productive Web Applications[C] Ifip International Information Security Conference, (2015).

[8] Dowling S, Schukat M, Barrett E: Using Reinforcement Learning to Conceal Honeypot Functionality[M]. European Conference, ECML PKDD 2018, pp.10–14, (2019).

[9] Kamel N E, Mohamed E, Lmoumen Y, et al: A Smart Agent Design for Cyber Security Based on Honeypot and Machine Learning[J]. Security and Communication Networks, 8(9), (2020).

[10] Veluchamy S, Kathavarayan R S: Deep Reinforcement Learning for Building Honeypots against Runtime DOS Attack[J]. Solid State Technology63(2), pp.576-591 (2020).

[11] Venkatesan S, Albanese M, Shah A, et al: Detecting Stealthy Botnets in a Resource-Constrained Environment using Reinforcement Learning[C]. 4th ACM Workshop on Moving Target Defense. pp.75-85(2017).

[12] Huang L, Zhu Q: Adaptive Honeypot Engagement through Reinforcement Learning of Semi-Markov Decision Processes[J]. 2019.International Conference on Decision and Game Theory for Security. Springer, Cham, pp.196-216(2019).

[13] Gao Y, Zhang G, Xing C, et al: A Multiphase Dynamic Deployment Mechanism of Virtualized Honeypots Based on Intelligent Attack Path Prediction[J]. Security and Communication Networks (2021).

[14] Pauna A, Iacob A C, Bica I: Qrassh-a self-adaptive ssh honeypot driven by q-learing[C]. 2018 12th International Conference on Communications (COMM), pp.441-446(2018).

[15] WANG Juan, YANG Hongyuan, FAN Chengyang: A SDN Dynamic Honeypot with Multi-phase Attack Response[J]. Netinfo Security, 21(1), 27-40(2021).

[16] Lian Z, Yin X, Tan R, et al. Research on SDN Virtual Honeynet for Network Attack Situation[J]. Journal of Air Force Engineering University (Natural Science Edition), (2017).