AN ELASTIC-HYBRID HONEYNET FOR CLOUD ENVIRONMENT

Nguyen Khac Bao, Sung Won Ahn, Minho Park

Department of Information Communication, Materials, and Chemistry Convergence Technology, Soongsil University, Seoul 156-743, Korea

ABSTRACT

When low-interaction honey net systems are not powerful enough and high-interaction honey net systems require a lot of resources, hybrid solutions offer the benefit's of both worlds. Affected by this trend, more and more hybrid honey net systems have been proposed to obtain wide coverage of attack traffic and high behavioral ideality in recent years. However, these system themselves contain some limitations such as the high latency, the lack of prevention method for compromised honey pots, the waste of resources and the finger printing problem of honey pot that hinder them to achieve their goals. To address these limitations, we propose a new honey net architecture called Efficient Elastic Hybrid Honey net. Utilizing the advantages of combining SDN and NFV technologies, this system can reduce the response time for attack traffic, isolate compromised honey pots effectively, defeat the finger printing problem of honey pots, and optimize the resources for maintenance and deployment. Testing our system with real attack traffic, the results have showed that Efficient Elastic-Hybrid Honey net system is not only practical, but also very efficient.

INDEX TERMS

Honey net, Honey pot, Elastic, Hybrid, Software defined Networking, Network Function Virtualization

1. INTRODUCTION

Since 1990s, honey pots [1] have been used to cope with various security threats by observing and understanding the exploits, methods, and strategies used by attackers. Throughout the years, a lot of honey pots have been proposed and developed to capture, analyze, and ultimately react against these new types of attacks [12-16]. These honey pots can be classified by the level of interaction [2]. High-interaction honey pots allow attackers to interact with a real operating system running some vulnerable services with few restrictions. By using high-interaction honey pots, we can obtain the information of how the honey pots were being probed, and misused, as well as the motivation of the attackers. On the other hand, low-interaction honeypots just provide limited functionalities to attackers by using emulated operating systems and services. However, low-interaction honey pots can deal with the large of different types of network traffic in a short time.

Honey net, a network of honey pots, was first introduced in 1999 by a long-term project called Honey net Project. Since then, some researchers have proposed several hybrid honey net systems

Dhinaharan Nagamalai et al. (Eds) : CSEIT, NCS, SPM, NeTCoM - 2018
pp. 117–127, 2018. © CS & IT-CSCP 2018 DOI : 10.5121/csit.2018.81809

[6-9] which can utilize the both advantages of low and high interaction honey pots. However, there are four main limitations in the existing honey nets: (1) the high latency of the system, (2) the finger printing problem of honey pots, (3) the large amount of resources required for deployment and maintenance, and (4) the lack of prevention mechanism when honey pots are compromised by attackers. Because the existing hybrid honey net systems just adopted both types of honey pots they still have the same problems of honey pots. The fingerprinting issue is an example of the problems. So far, a wide range of fingerprinting techniques have been developed to detect the existence of honey pots. In [3], the authors revealed that most low-interaction honey pots can be easily fingerprinted. For example, two low-interaction honey pots called Kippo and Kojoney which always return a hardcoded timestamp when attackers access these systems. In addition, low-interaction honey pots can be detected by checking their environmental variables. Relying on the operating system types and these services running on them is also a honey pot fingerprinting technique that skilled attackers have always used [4]. In the current honey nets, most of the authors did not take into account the compromised problem of the high-interaction honey pots. Therefore, when these honey pots are compromised by attackers, they can be used to attack production servers or internal hosts in the network. In term of resource efficiency, these existing honey net systems may bring resource waste if there is no attacker. In other words, they waste a lot of resources to run these honey nets without attackers.

To overcome these limitations of the existing systems, in this paper, we propose an SDN and NFV based honey net, called Efficient Elastic-Hybrid Honey net (E2H2). In our system, each type of honey pots deals with different phases of attacks to maximize the advantages of both low-interaction honey pots and high-interaction honey pots. Initially, only a low-interaction honey pot with less used resources runs. All of the discovering attacks are taken by this low-interaction honey pot to get the wide coverage of attack traffic. Along with these reconnaissance attacks, high-interaction honey pots are created in a form of virtual machines to interact with the attacker in the specific attacks to obtain the deep understanding of the attacker's behaviors. The number of high-interaction honey pots increases proportional to the number of the attacks. When the specific attacks end, the system collects all log files in the high-interaction honey pots, and destroys the virtual machines of the high-interaction honey pots to save resources.

To defeat finger printing techniques, E2H2 system uses the same information of OS types and number of services in high interaction honey pots and virtual hosts created by the low interaction honey pot. Hence, when switching the connections from the low-interaction honey pot to the new high-interaction honey pot the attackers hardly discover the changes. Moreover, the abundant number of high-interaction honey pot images are created in NFV platform to confuse even skilled attackers. Currently, the existing honey net systems have to process all of attack traffic then depends on the capabilities of the honey pots to find appropriate ones. By leveraging SDN technology, E2H2 system can easily observe all connections belonging to the existing the high-interaction honey pots in the network. Thus, any abnormal action of these honey pots raises an alarm to indicate the compromised problem has occurred. These compromised honey pots will be isolated and disconnected from the network immediately for offline analyses.

This paper is organized as follows. The related work is reviewed in Section 2. In Section 3, we provide a full presentation of Efficient Elastic-Hybrid Honey net system. In Section 4, serveral experiment results are discussed. Section 5 concludes this paper.

2. RELATED WORK

A. HYBRID HONEYNET ARCHITECTURE

Along with the development of honey pots, a lot of researchers have been interested in developing a hybrid honey net system. Bailey [6] is the first person who tried to integrate low and high-interaction honey pots to solve the trade-off problem between two types of honey pots. In his system, low-interaction honeypots used as sensors to collect information to get the wide coverage of attack traffic. Their approach can reduce the number of the high-interaction honey pots in a network while still get the wide coverage of different attack traffic. However, it increases the burden for the low-interaction honey pots and raises the fingerprinting issue of honey pots.

In 2013, VMI-Honey mon [8] appeared as a hybrid honey net system which solves the routing problems when using multiple identical high-interaction honey pot clones. Since these high-interaction honey pots share the same MAC and IP addresses, they have to put each clone into separate network bridges. They used ip tables to forward incoming connections to the low-interaction honeypots and then queued them. Thus, this can increase the latency of system response time and decrease the opportunities to successfully lure attackers.

Based on Bailey work, in 2015, FAN [7] proposed a dynamic hybrid honeynet with two main modules: decision engine and redirection engine. The author used a server acts as a gateway, which gets the attackers requests, forwards messages to the low-interaction honeypots, and then redirects connections to the high-interaction honeypots. As VMI Honeymon, this system also has the high latency since their server has to modify TCP headers of all response packets time by time. Moreover, by using an additional server, this system consumes more resources than other existing honeynet systems.

HoneyMix [9] is the latest work which related to NFV and SDN environment. This system focused on two main limitations of the existing honeynets: (1) fingerprinting problem, and (2) Gen-III honeynets only provide coarse-grained data control. By using dynamic connection selection mechanism, this system can provide various responses for attacker's requests. Hence, this kind of system might be not practical in the real world. In addition, both types of honeypots can be selected; therefore, the low-interaction honeypots get the high probability to be detected by the attacker.

B. SOFTWARE-DEFINED NETWORKING AND NETWORK FUNCTION VIRTUALIZATION

Software-defined networking (SDN) is an emerging network paradigm that provides a flexible way to control entire the network. By separating data plane and control plane, SDN allows moving part of the decision-making logic from network devices to the controllers.

Network Function Virtualization (NFV) [10] aims at providing network functions such as gateways, firewall... in software, which can be run on commodity hardware in data centers.

3. ELASTIC HYBRID SYSTEM

Normally, before performing a specific attack, attackers always try to discover the network for obtaining the useful information. Thus, network reconnaissance is the primary and initial step of any advanced and persistent attack. Performing network recon naissance, attackers can(1) discovering and enumerating active hosts, (2) detecting the current OS, open ports and related application names in these hosts, (3) discovering vulnerabilities in each host.

Based on the obtained information of active hosts, the attacker can perform further attacks such as flooding attack, brute force, SQL injection, etc. Therefore, the normal attack process can be divided into two phases:

1) Reconnaissance attack: Attackers try to find out the active hosts inside the network and the services, which are running on these host, by using some attack tools such as Nmap, Zenmap [4]. After discovering an active IP address, Nmap can issue a reverse-DNS query to obtain the domain name from the host's address.

2) Specific attack: Based on the information from reconnaissance attack, attackers will use some dedicated tools to attack some given services

A. SYSTEM MODELING

Utilizing the SDN technology, the network inside E2H2 system is controlled by SDN controllers to get more flexible and reduce the computational cost when steering network traffic. Beside that, with the help of NFV technology, we can create or remove instances as well as network between them quickly and easily. Basically, E2H2 system consists of five main components:

• Selecting services provides an active way to deal with scanning attacks. By choosing a random number, this module synchronizes with the low-interaction honeypot to give some available services for attackers. This number also relates to the image id in the NFV platform which will be used to initiate a new high-interaction honeypot later.

• Forwarding engine keeps an important role in E2H2 system. When the attacker sends a lot of scanning traffic for discovering network, forwarding engine redirects the traffic to the low-interaction honeypot which opened some services based on random number created by selecting services. All of the new traffic from the attacker will be forwarded to a recently created high-interaction honeypot to get deeper attacker's behaviors learning.

• Observation engine checks the connections between the attacker and the high-interaction honeypot periodically. It sends a message to add/removing engine to start removing this high-interaction honeypot. The handle honeypot module will use NFV APIs to remove the high-interaction honeypot for resource restoration.

• Adding/Removing engine works as a bridge between SDN controllers and NFV orchestrator. It receives messages from selecting services and observation module then sends add/remove requests to handle honeypot engine for further processes.

• Handle honeypot engine is located in NFV orchestrator. Whenever receiving requests from Add/Removing engine, handle honeypot engine uses all open APIs of NFV platform to initiate or remove instances

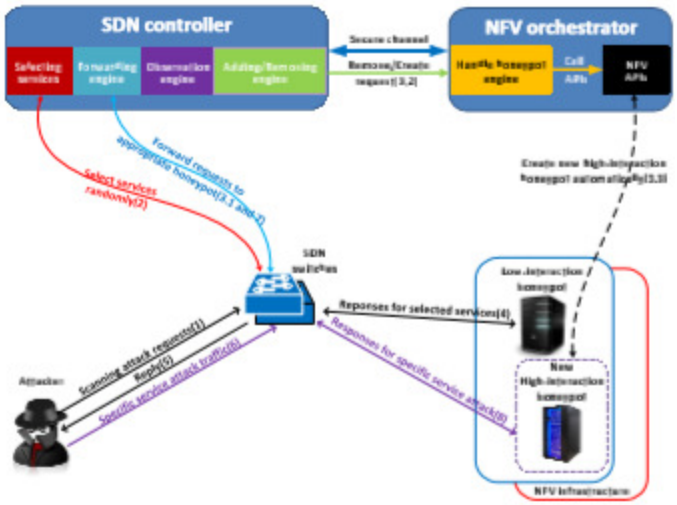

Fig. 1. Efficient Elastic-Hybrid Honeynet architecture

B. LOGIC SYSTEM WORK FLOW

In E2H2 system, there is only one low-interaction honeypot which runs all the time to deal with network reconnaissance attacks. All high-interaction honeypots will be created to handle further attacks from the adversary.

Figure 1 shows an use case of how E2H2 system works. When the attacker performs scanning attack to find out active hosts and opening ports, the selecting services module randomly creates a number which related to image id in NFV orchestrator. With NFV platform, we can make a large number of images manually by using CD or DVD ISO files. Each image corresponds with an OS type and some fixed services running on it. By this way, a honeynet with abundant different services was made for attackers to attack to. The relationship between virtual hosts created by the low-interaction honeypot and image list of NFV infrastructure is showed in Figure 2.

After creating random number, the forwarding engine selectively forwards all of the attacker's requests which related to designated available services to the low-interaction honeypot. Since these requests are belong to the scanning attack, the forwarding engine just sets a small value of hard timeout for all of them. Beside that, a historic used source port list is created by the forwarding engine to save all source ports in the reconnaissance attack. In the existing honeynet architecture, they have to receive all attacker's scanning traffic and then provide the responses back to the attacker based on the ability of the current honeypots.

Along with this step, Adding/Removing engine sends the creation request to handle honeypot module in NFV orchestrator. This means a new high-interaction honeypot is creating while the low-interaction honeypot is responding to the attackers scanning requests. Figure 3 gives the straight view of our mechanism for traffic redirection.

When the attacker receives the responses from the low-interaction honeypot, he/she knows which ports are opening so he/she will choose one of these ports to perform further attacks such as flooding attack, brute force, SQL injection, etc. These attacks can be performed by using some tools e.g., bonesi, loic or sqlmap. By checking the used source ports and the relation between used source ports and the source port of new flow, the system can know when the second phase of the attack starts.

The observation engine inspects the connections between the attacker and the high-interaction honeypots periodically. If the connections are closed, observation engine will collect all log files in high-interaction honeypot then send to offline server for deep analyses. After that, it sends a message to Add/Removing engine to delete this high-interaction honeypot. The handle honeypot module calls NFV APIs to remove the high-interaction honeypot instance and get back the resources. In addition, the observation engine also takes into account the infection of the compromised high-interaction honeypots. According to the authors of the paper [5], the higher the interaction level, the higher the possible misuse. Proving the real environments for the attackers is the both advantage and disadvantage of high-interaction honeypots. In E2H2 system, all of the high-interaction honeypots are supervised by the observation engine. If any connection with internal hosts is found, this high-interaction honeypot is identified as a compromised honeypot. It rapidly be disconnected from the network for offline investigation, then another similar one will be created to replace.

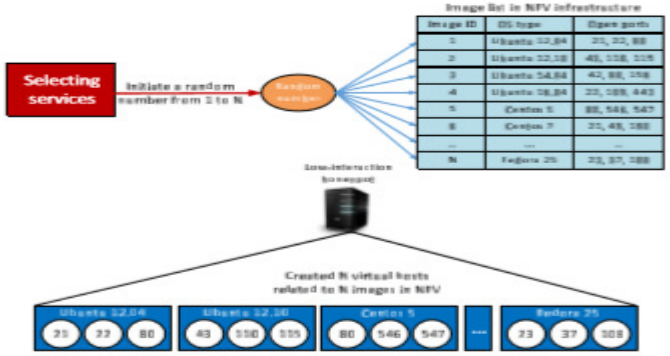

Fig. 2. Relation between two types of honeypots in E2H2 system

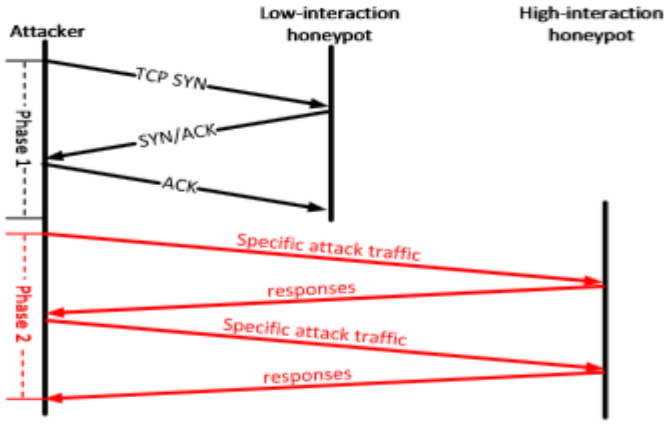

Fig. 3. Illustration of the traffic redirection mechanism

C. RESOURCE OPTIMIZATION MODEL

In the normal existing honeynet architecture, all the honeypots have one common problem, i.e., they are worthless if there is no attacker. As long as attackers do not send any packet to the honeynet, the system wastes a lot of resources for running these honeypots. Low-interaction honeypots can typically be deployed with fewer resources because they do not fully offer the real services and they also incur less risk. Resource optimization is an important problem that we want to solve since it is the limitation of the existing honeynet architecture.

Normally, the used resources of honeypots is vary depended on the state of them. Under the attack, a honeypot uses more resources than it is in normal state. Assume that we have N honeypots with the same type, the function of average used resource for each honeypot in different states can be defined as follows:

$$\overline{R}_{Honeypot}(nor) = \frac{\sum_{i=1}^{N} \left(P_i^{CPU}(nor) + P_i^{RAM}(nor)\right)}{N} \quad (1)$$

$$\overline{R}_{Honeypot}(atk) = \frac{\sum_{i=1}^{N} \left(P_i^{CPU}(atk) + P_i^{RAM}(atk)\right)}{N} \quad (2)$$

We assume that there are m low-interaction honeypots and n high-interaction honeypots in the normal system. Under the attack, the probability of a low-interaction honeypot will be chosen is p. With k attacks at the same time, the average used resources of honeynet in the normal system is:

$$\overline{R}_{NS}^{k} = (m-kp)\overline{R}^{LIH}(nor) + [n-k(1-p)]\overline{R}^{HIH}(nor) + kp\overline{R}^{LIH}(atk) + k(1-p)\overline{R}^{HIH}(atk) \quad (3)$$

	Normal	Attack
Avg. CPU (LIH)	1.13 (%)	2.03 (%)
Avg. RAM (LIH)	12.24 (%)	13.15 (%)
Avg. CPU (HIH)	5.27 (%)	7.18 (%)
Avg. RAM (HIH)	22.23 (%)	25.74 (%)

Table I Average Used Resources Of Lih And Hih In Different States

In E^2H^2 system, there is only one low-interaction honeypot which always runs. High-interaction honeypots will be created depend on the number of attacks. Therefore, with k attacks at the same time, the average used resources of honeynet in E2H2 system is:

$$\overline{R}_{E-H}^{k} = \overline{R}^{LIH}(atk) + (k-1)\left(\overline{R}^{LIH}(atk) - \overline{R}^{LIH}(nor)\right) + k\overline{R}^{HIH}(atk) \quad (4)$$

4. EXPERIMENTS

E^2H^2 was tested using Open Stack platform and POX controller which supports Open Flow protocol version 1.0. We used two SDN-enable switches: an Open v Switch version 2.3 and HP3800 switch, along with two compute nodes in Open Stack environment. With the Mitaka Open Stack platform, we ran the Open Stack controller in a high hardware configuration machinewith32GBRAMandinteli73.4ghzQuadCoreCPU.

A. BANDWIDTH EXPERIMENT

With the help of SDN technology, we can have a better control of our network. The New mechanism can help to reduce the number of responses for the attackers requests. The nethogs tool was used for bandwidth calculation in this experiment

As the results, in Figure 4, the reduction of bandwidth in the low-interaction honeypot is significant. With our system, even receiving the large number of concurrent attacks, the increasing of bandwidth in the low-interaction honeypot is very small (≈ 33 KBytes/s). In the other existing systems, the low-interaction honeypots receive and respond to all attackers requests; therefore, it can cause an overload in low-interaction honeypots with the increasing of bandwidth is 203.943 KBytes/s.

B. RESOURCES EXPERIMENT

For collecting the largest possible amount of information including complete attack logs, data access, executed byte codes, etc, they preferred to use high-interaction honeypots rather than low-interaction honeypots in the normal existing honeynet systems. Thus, the number of low-interaction honeypots and high-interaction honeypots in the normal systems respectively are 4 and 10. Another reason is that the small number of the high-interaction honeypots can lead to the high probability that attackers can detect the low-interaction honeypots in these systems. We set the value of p is 0.7. The average used resource of them are equal. Table I shows all information related to both types of honeypots in two different states.

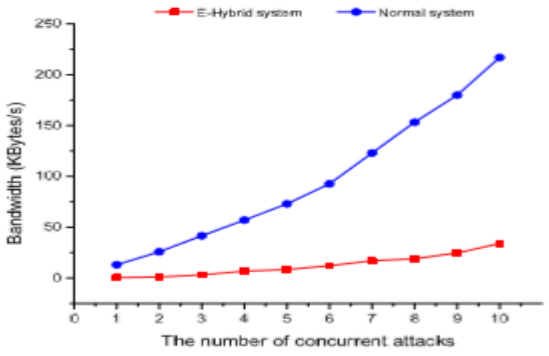

Fig. 4. Bandwidth comparison of low-interaction honeypots in two systems

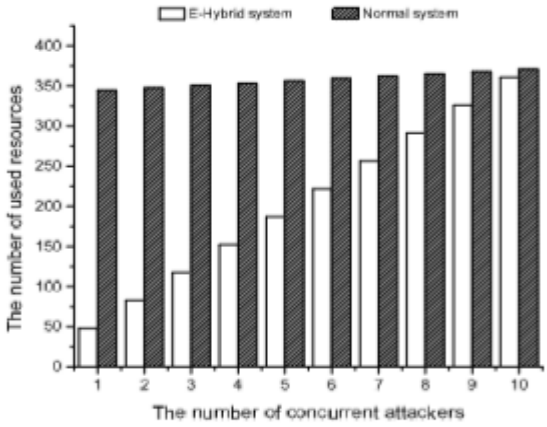

Fig. 5. Resource comparison under multiple concurrent attackers

In Figure 5, when the number of attackers attack in the same period of time is small, our system can save a lot of resources (just used ≈ 14 % in compare with the normal systems). The probability of multiple attackers attack a system at the same period of time is very small; however, even when this event can be occurred (with 10 concurrent attackers) E2H2 system still uses a less resources than the other existing ones.

C. AVERAGE RESPONSE TIME EXPERIMENT

The average response time of a system is also an important criteria to evaluate the effectiveness of a honeynet system. The experiment was performed by running the bonesi tool to generate a large number of packets in 10 seconds. We captured all the packets by using wireshark and calculate the average response time based on the time for a TCP connection establishment (three-way handshake).

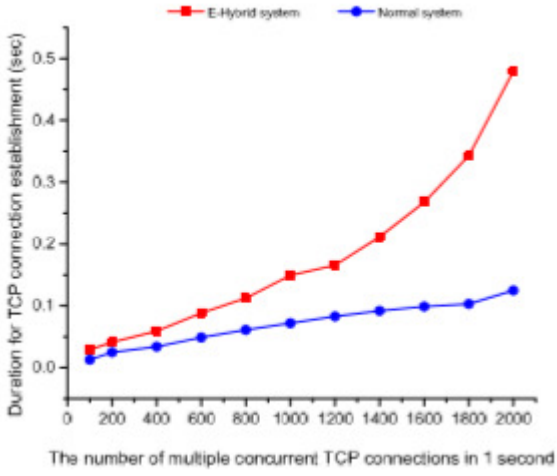

Fig. 6. Connection establishment time comparison

In E^2H^2 system, the high-interaction honeypot creation step runs along with the scanning attack process of the attacker. The main latency comes from the first data process step when we select the image id of high-interaction honeypots and choose some services which will be attacked by the attacker. In the Figure 6, that latency seems very small. Even when the attacker sends 2000 TCP connections per 1 second, the latency of our system is just 0.4 seconds. It is acceptable since this latency does not cause any suspicion to the attacker.

5. CONCLUSIONS

In this paper, we proposed Efficient Elastic-Hybrid Honeynet system, a new architecture for honeynet to enhance the efficiency of using resources. Moreover, by using different type of honeypots to deal with different phases of attacks, this system can obtain the advantages of both low-interaction and high-interaction honeypots. The system not only gains the wide coverage types of network traffic but also gets the high behavioral fidelity. Currently, while implementing E2H2 system in the real world, we have gotten some optimal results. In the future, we will try to reduce the response time of the low-interaction honeypot and deploy this system in some large-scale networks in order to make it be practical for enterprise environment.

ACKNOWLEDGEMENTS

This work was supported by Institute for Information & communications Technology Promotion(IITP) grant funded by the Korea government(MSIT) (No.2018-0-00254, SDN security technology development)

REFERENCES

[1] L. Spitzner, ”Honeypots: Catching the insider threat,” In the IEEE 19th Annual Conference on Computer Security Applications, pp.170179, 2003.

[2] Seifert, C., Welch, I., and Komisarczuk, P., Taxonomy of Honeypots. [Online]. Available: http://www.mcs.vuw.ac.nz/comp/ Publications/archive/CS-TR-06/CS-TR-06-12.pdf.

[3] Black Hat USA 2015 - Breaking Honeypots For Fun And Profit. [Online]. Available: https://www.youtube.com/watch?v=Pjvr25lMKSY.

[4] J. H. Jafarian, E. Al-Shaer, and Q. Duan, ”Adversary-aware IP address randomization for proactive agility against sophisticated attackers,” in Proceedings of the IEEE Conference on Computer Communications, pp. 738746, April, 2015.

[5]Marcin Nawrocki, Matthias Whlisch, Thomas C. Schmidt, Christian Keil, Jochen Schnfelder, ”A Survey on Honeypot Software and Data Analysis,” August, 2016.

[6] Bailey, M., Cooke, E., Watson, D., Jahanian, F., Provos, N.: A hybrid honeypot architecture for scalable network monitoring. Technical Report CSE-TR-499-04, U. Michigan, October 2004.

[7] FAN, Wenjun, et al. ”Dynamic Hybrid Honeypot System Based Transparent Traffic Redirection Mechanism,” In International Conference on Information and Communications Security, pp.311-319, 2015.

[8] Lengyel, T.K., Neumann, J., Maresca, S., Kiayias, A., ”Towards hybrid honeynets via virtual machine introspection and cloning,” In: Lopez, J., Huang, X., Sandhu, R. (eds.) NSS 2013. LNCS, vol. 7873, pp. 164177. Springer, Heidelberg (2013).

[9] HAN. Wonkyu, et al. HoneyMix, ”Toward SDN-based Intelligent Honeynet,” In Proceedings of the 2016 ACM International Workshop on Security in Software Defined Networks & Network Function Virtualization, pp.1-6, 2016.

[10] ETSI GS NFV-MAN Network Functions Virtualization (NFV); Management and Orchestration v1.1.1, Dec. 2014. [Online]. Available: http://www.etsi.org/deliver/etsi gs/NFVMAN/001 099/001/01.01.01 60/gs NFV-MAN001v010101p.pdf.

[11] 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):6974, 2008.

[12]C. Leita, V. Pham, O. Thonnard, E. Ramirez-Silva, F. Pouget, E. Kirda, and M. Dacier, The leurre. com project: collecting internet threats information using a worldwide distributed honeynet, in Information Security Threats Data Collection and Sharing, 2008. WISTDCS08. WOMBAT Workshop on, pp. 4057, 2008.

[13] V. Yegneswaran, P. Barford, and V. Paxson, Using honeynets for internet situational awareness, in Proceedings of the Fourth Workshop on Hot Topics in Networks (HotNets IV). Citeseer, pp.1722, 2005.

[14] P. Baecher, M. Koetter, T. Holz, M. Dornseif, and F. Freiling, The nepenthes platform: An efficient approach to collect malware, in Recent Advances in Intrusion Detection. Springer, pp. 165184, 2006.

[15]M. Gruber, D. Hoffstadt, A. Aziz, F. Fankhauser, C. Schanes, E. Rathgeb, and T. Grechenig, Global voip security threatslarge scale validation based on independent honeynets, in IFIP Networking Conference (IFIP Networking), pp. 19, 2015.

[16] M. Wahlisch, A. Vorbach, C. Keil, J. Sch onfelder, T. C. Schmidt, and J. H. Schiller, Design, implementation, and operation of a mobile honeypot. [Online]. Available: http://arxiv.org/abs/1301.7257.