ICIC Express Letters
Volume **11**, Number **1**, January **2017**

ICIC International ©2017 ISSN 1881-803X
pp. 135–141

END INFORMATION HOPPING FOR ACTIVE CYBER-DEFENSE BASED ON SDN

Leyi Shi[1,*], Qin Li[1], Zijing Cheng[2], Zhiyu Xue[1] and Xianyong Lv[1]

[1]College of Computer and Communication Engineering
China University of Petroleum (East China)
No. 66, Changjiang West Road, Qingdao 266580, P. R. China
[*]Corresponding author: shileyi@upc.edu.cn; stoneglad@hotmail.com

[2]State Key Laboratory of Space Ground Integrated Information Technology
Beijing Satellite Information Engineering Research Institute
No. 82, Zhichun Road, Beijing 100086, P. R. China
linuxdemo@126.com

Received June 2016; accepted September 2016

Abstract. *Concerning the static, hysteresis and passivity of traditional network defense capability, this paper has presented an end information hopping method based on OpenFlow protocol, combining with the advantages of flexibility and easy-control in software defined network (SDN). The data streams are diverse by constantly changing the IP address, port number, protocol etc. to bewilder the attackers. The hopping method based on SDN is described in detail, including two critical issues: dummy address mapping and synchronization. The prototype implementation based on open source floodlight has been carried out, which is deployed as the testbed. Our analysis reveals that, the method in this paper can effectively improve the security defense performance of network.*
Keywords: End information hopping, SDN, OpenFlow protocol, Active cyber-defense, Security

1. **Introduction.**

In recent years, many network security incidents happened and brought serious disasters to global Internet users. The security defense capability of information system faces severe challenges. Confronting the ever-changing technology of hackers, the active cyber-defense technology emerged, including moving target defense (MTD), mimic security defense (MSD), end-hopping, etc.

The active cyber-defense technology is the current hot topic. MTD is widely used to protect the identity of nodes for MANETs [1], defense cross site scripting attacks [2], and protect the privacy of users in cloud infrastructures [3]. Wu described the main content and characteristics of MSD [4]. Shi et al. firstly proposed the conception of end-hopping [5], and then it was used to defense the denial of service (DoS) attacks in [6,7].

The APOD project [8] and DyNAT [9] based on traditional network architecture are prone to service interruptions, complex configurations and time delay [10], so they are characterized by inefficiency and instability. The OF-RHM proposed in [11] is unit random and cannot satisfy random hopping well. Moreover, the method only suits for LAN-level. In this paper, we propose a method of end information hopping based on SDN, which separates the control plane from data plane [12]; therefore, we can operate the data flow conveniently rather than to change the configuration of clients in traditional architecture. For external communications, internal end information maps to different one in every short time, including IP address, port number, protocol, timestamp, etc. So it achieves multivariate random and the data flow between communicators shows a pattern of diversification. It can enhance the confidentiality and availability of system.

The remainder of the paper is organized as follows. Section 2 gives a description of our method in detail. In Section 3, we analyze the system service performance while employing the end-hopping method, and then valuate the defense ability of the system. Finally, we conclude the paper and discuss the future works in Section 4.

2. Hopping Framework Based on SDN.

Our primary goal is to design a method which can deal with complex and volatile attacks actively. A common technology is the end information hopping. We take the advantage of flexibility in SDN to realize fast mutation. There are two critical issues: dummy address mapping and synchronization which are described in Section 2.2 and Section 2.3 separately.

2.1. End-hopping framework.

We assume that there are two networks net_1 and net_2. They connect with each other. Each network is a traditional LAN. Network net_1 has a set of hosts $\{h_1, \ldots, h_n\}$. Network net_2 has a set of hosts $\{h_{n+1}, \ldots, h_{n+m}\}$. The network boundaries of net_1 and net_2 are composed by OpenFlow switches OF_s_1 and OF_s_2. Suppose internal networks of net_1 and net_2 were safe and the attackers were external attackers.

Firstly, we list some notations used in this paper and give their meanings in Table 1, while the end-hopping framework is shown in Figure 1.

TABLE 1. Notations

Notation	Meaning
h_timeout	the valid time of a flow table entry
r_SrcIP	real source IP address of a packet
r_DstIP	real destination IP address of a packet
d_SrcIP	dummy source IP address of a packet
d_DstIP	dummy destination IP address of a packet
r_SrcPort	real source MAC address of a packet
r_DstPort	real destination MAC address of a packet
d_SrcPort	dummy source MAC address of a packet
d_DsPort	dummy destination MAC address of a packet

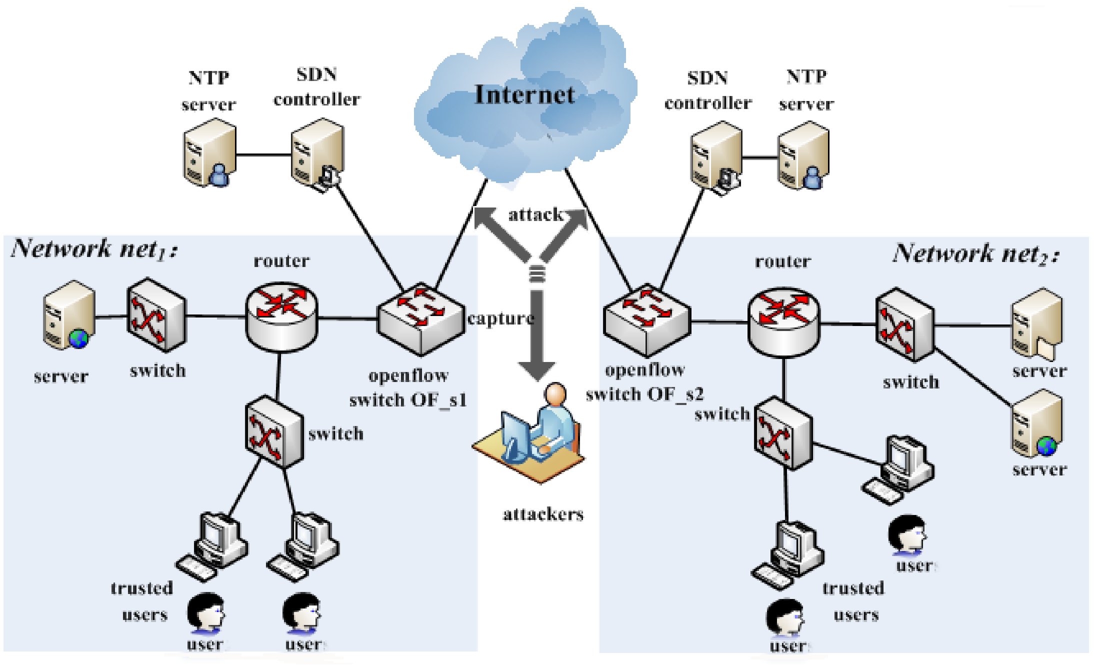


FIGURE 1. End hopping framework based on OpenFlow

The communications in this architecture can be divided into two types: the first one is that both the two terminals are in a same LAN net_1 or net_2, while the second type is opposite. We focus on the latter. In this case, the SDN controller maps the real end information to dummy information and the map relationship changes all the time.

2.2. **Dummy map mechanism to confuse the attacker.** Sniffer or listening is always the first step in network attacks. Attackers use sniffer tools to ascertain the IP addresses or port numbers of the target network. Once the attackers get the end information, they can launch a series of attacks. Our first work is using dummy map mechanism, in which we map the real end information to dummy one, to hide the significant end information. The changing elements include IP address, port number, protocol, etc. So it is multivariate random and has much capacity for change. Therefore, it has great randomicity and the attackers cannot obtain the real end information, or even launch attack. The process of dummy map is described as follows, as shown in Figure 2.

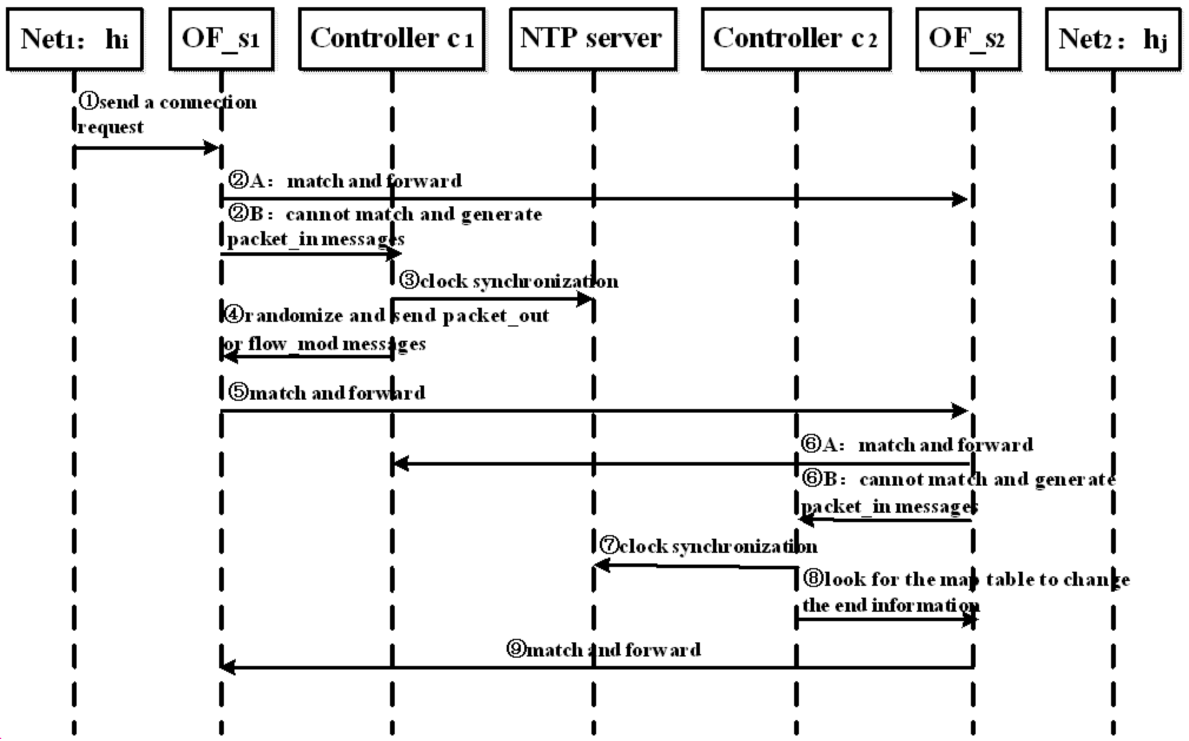

Net1: hi
OF_s1
Controller c1
NTP server
Controller c2
OF_s2
Net2: hj
①send a connection request
②A: match and forward
②B: cannot match and generate packet_in messages
③clock synchronization
④randomize and send packet_out or flow_mod messages
⑤match and forward
⑥A: match and forward
⑥B: cannot match and generate packet_in messages
⑦clock synchronization
⑧look for the map table to change the end information
⑨match and forward

FIGURE 2. The process of dummy map

(1) A terminal node h_i in net_1 sends a connection request to the node h_j in net_2.
(2) When the boundary OpenFlow switch OF_s_1 receives the message, it looks for its flow table. If there was an entry to match the packet, it would forward directly. Otherwise, it would generate a packet_in message and send it to the SDN controller to deal with.
(3) SDN controller gets the clock from NTP server and adjusts its own time.
(4) SDN controller uses random election algorithm to choose end information like *d_SrcIP* and *d_SrcPort*, and then sends a packet_out or flow_mod message to the OF_s_1; meanwhile, the 4-*tuples* composed by *r_SrcIP*, *r_SrcPort*, *d_SrcIP*, and *d_SrcPort* are stored in the map table.
(5) OF_s_1 generates a new rule to forward the packets with the same characteristics.
(6) When OF_s_2 receives the packet, it will perform like the process (3).
(7) SDN controller gets the clock from NTP server and adjusts its own time.
(8) The controller seeks its map table to map the dummy end hopping information to the real one.
(9) OF_s_2 generates a new rule to forward the packets with the same characteristics.

When h_j receives the packet, it will send response packets to h_i. Then this process will be performed again.

2.3. **Hopping and synchronization issue.** We not only use dummy end information to communicate with each other, but also change the map relationships all the time. End information hopping method makes the data flow just like some changing and irregular packets for attackers.

(1) Hopping mechanism. We use the end-hopping method to protect our system, so hopping is the fundamental thought of this method.

In this work, our main thought is to change the flow table entry by changing the map relationship. There are two situations in which the flow table entry will be removed: the former is the end of a timer, and the latter relies on the SDN controller. Each flow table entry has an idle_timeout timer and a hard_timeout timer. The idle_timeout timer calculates the time of no matching traffic, while the hard_timeout timer calculates the time after the entry is inserted into the table. OpenFlow switch will delete the flow table entry automatically once the timer reaches deadline. In addition, the controller can also send commands to remove the flow table entries. The flow table will not have matching entry after it is deleted. So when new packet comes, it will perform the dummy map again, and then change the map relationship.

(2) Synchronization issue. For all the hopping systems, synchronization is a permanent issue. The end information of terminals in our end-hopping system is changing all the time, so it needs reasonable synchronization mechanism to ensure the normal data communication.

NTP is a protocol to ensure the same time. It can estimate the round-trip time (RTT) delay of the packet in the network and the time deviation of the computer independently. In addition, it can realize high-accuracy computer synchronization. In our work, we use precision clock synchronization mechanism based on NTP protocol to ensure communicating parties with the same time.

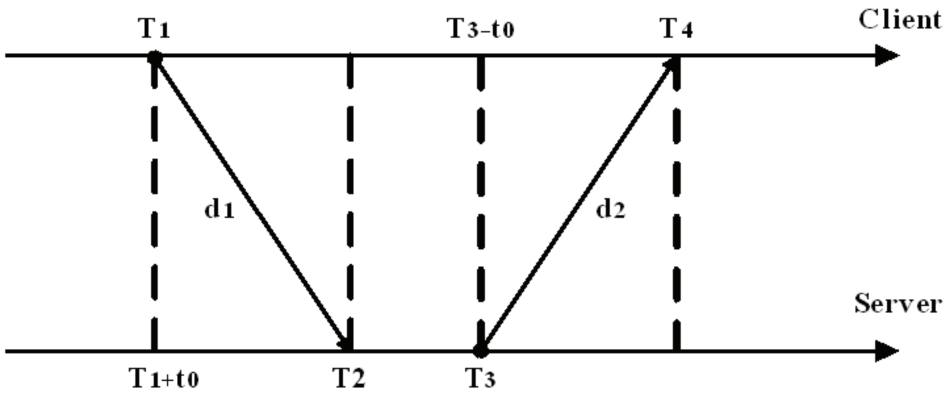
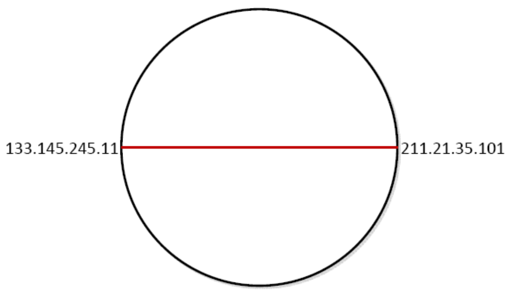


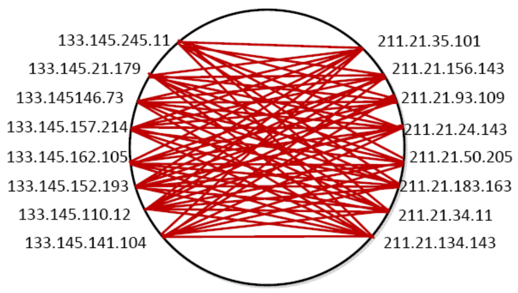
Figure 3. Theory of the NTP synchronization

3. **Experimental Evaluations.** At first, we use Linux system Ubuntu 14.04 LTS, and install the open source floodlight as the SDN controller to implement the prototype system based on OpenFlow protocol. Then the Mininet 2.2.1 is used to simulate the network scene. Finally, the system is used to analyze and evaluate the performance of end information hopping method based on SDN.

3.1. **End-hopping performance.** The main part of our method is to realize end information hopping all the time. To evaluate the end information hopping performance of our system, we use network protocol analyzer software to capture network packets and the results are shown in Figure 4. As we can see, the end information is changing all the time



(a) Traffic mapping (no hopping)

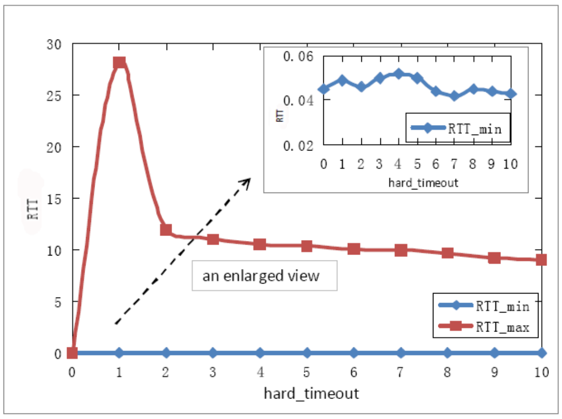


(b) Traffic mapping (hopping)

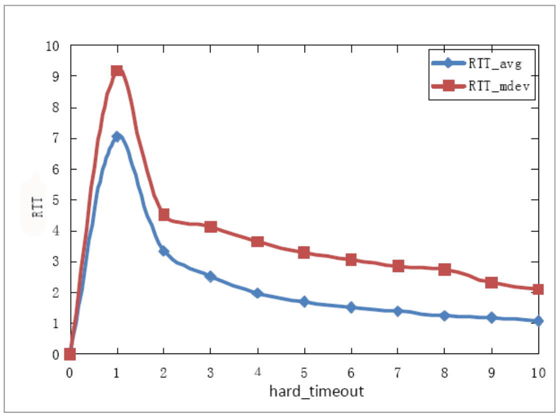
Figure 4. Network traffic with hopping or not

Table 2. Performance test data for different hopping rates

hard_timeout	0	1	2	3	4	5	6	7	8	9	10
RTT_min	0.045	0.049	0.046	0.050	0.052	0.050	0.044	0.042	0.045	0.044	0.043
RTT_max	0.089	28.186	11.944	11.092	10.565	10.409	10.096	10.007	9.727	9.251	9.051
RTT_avg	0.062	7.050	3.344	2.521	1.979	1.697	1.520	1.390	1.247	1.182	1.081
RTT_mdev	0.006	9.183	4.511	4.118	3.642	3.294	3.069	2.844	2.740	2.319	2.091



(a) *RTT_min* and *RTT_max*



(b) *RTT_avg* and *RTT_mdev*

Figure 5. Packets captured with hopping

in Figure 4(b), while Figure 4(a) is static. At the same time, we send 10000 packets to test the system, and it has no packets loss. So the end-hopping system based on SDN can guarantee accuracy, validity and completeness compared to the traditional architecture.

3.2. **QoS test.** We test the web quality of service when our system changes the end information at different speeds and the results are shown in Figure 5. The parameter *hard_timeout* stands for the speed of hopping. The smaller *hard_timeout* is, the faster the end information hops, but 0 is infinity and means the system is without hopping.

(1) According to the enlarged view of the minimum of RTT (*RTT_min*), we can see that the *RTT_min* line is smooth. It virtually unchanged, no matter what the *hard_timeout* is. The reason may be that the flow table has an entry to match the date stream, and then the packets will forward directly. The RTT is the smallest at this point.

(2) As we can see in Figure 5, the maximum, average and mean deviation of RTT decrease as *hard_timeout* increases except when the *hard_timeout* equals 0 and 1. When *hard_timeout* equals 0, the end information is static and the flow table is infinity, so the RTT is basically stable and small. When *hard_timeout* equals 1, the end information

changes quickly, so it always needs to communicate with SDN controller to deal with the packet_in messages. Then the *RTT_max*, *RTT_avg* and *RTT_mdev* reach the peak.

3.3. **Anti-attack test.** We do experiment on our system to test the anti-attack ability. In this part, the tool hping3, which is a command-line oriented TCP/IP packet assembler/analyzer, is used to simulate the DoS attacker. hping3 sends SYN packets to connect to the server at high speed rate, and Figure 6 shows the number of loss packets when the *hard_timeout* is different. We use the sectional image to deal with the Y-axis for the significant differences between the lost packets.

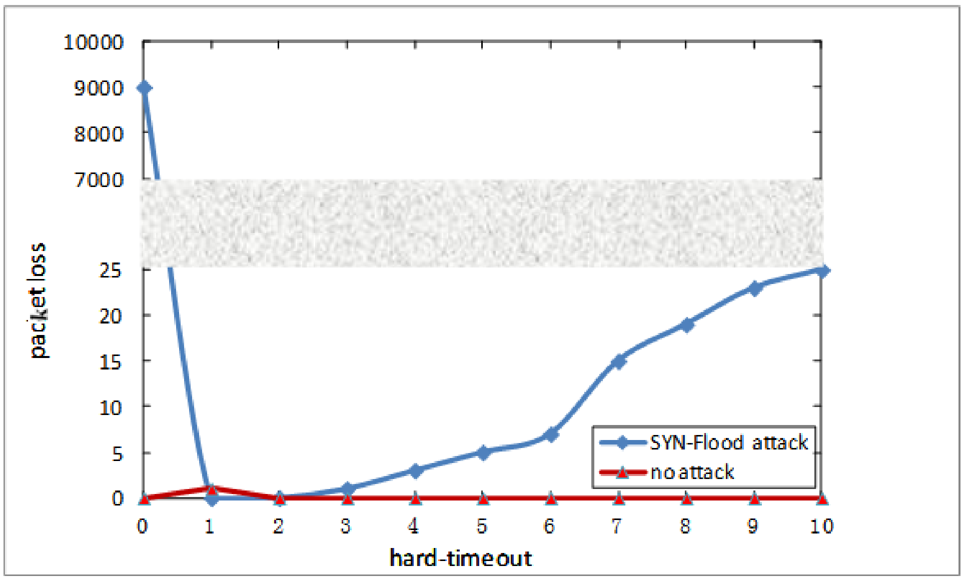


FIGURE 6. The number of loss packets

The results are as expected. (1) When the end information is static, the server goes into crash quickly. (2) The number of loss packets increases with the *hard_timeout* increases.

4. **Conclusions and Future Works.** Our work has focused on the security defense problem of end information hopping based on SDN architecture. We designed a dummy map mechanism applied on OpenFlow network. In order to prove the feasibility and effectiveness, we designed and implemented an end information hopping system based on SDN. The experiments and simulation results show that the end-hopping method based on SDN can achieve high hopping speed rate and defense DoS attack to a certain degree. We may reasonably conclude that, taking the advantages of SDN, the active cyber-defense method in this paper can effectively improve the network security defense performance. In some networks with high-level security requirements, it can be another defense line, in addition to firewall and IDS. As for the future works, we will continue to study the random mapping mechanism and efficient synchronization methods. We expect to improve the system defense ability further and analyze the validity in more detail theoretically.

Acknowledgments. This work is supported by the National Natural Science Foundation of China (NSFC) under grant No. 91438117 and grant No. 91538202.

REFERENCES

[1] M. Albanese, A. D. Benedictis, S. Jajodia et al., A moving target defense mechanism for manets based on identity virtualization, *2013 IEEE Conference on Communications and Network Security*, pp.278-286, 2013.

[2] J. Portner, J. Kerr and B. Chu, *Moving Target Defense Against Cross-Site Scripting Attacks (Position Paper)*, Springer International Publishing, Switzerland, 2015.

[3] P. Wei, F. Li and X. Zou, *Moving Target Defense for Cloud Infrastructures: Lessons from Botnets*, Springer New York, New York, 2014.

[4] J. Wu, MSD in network space (in Chinese), *Security Science and Technology*, pp.4-9, 2014.

[5] L. Shi, C. Jia and S. Lv, Research on end hopping for active network confrontation, *Journal on Communications*, vol.29, no.2, pp.106-110, 2008.

[6] T. Siva and E. S. P. Krishna, Controlling various network based ADoS attacks in cloud computing environment: By using port hopping technique, *International Journal of Engineering Trends and Technology*, vol.4, no.5, pp.2099-2104, 2013.

[7] R. P. Kumar, J. Babu, T. Gunasekhar et al., Mitigating application DDoS attacks using random port hopping technique, *International Journal of Emerging Research in Management & Technology*, vol.4, no.1, pp.1-4, 2015.

[8] M. Atighetchi, P. Pal, F. Webber et al., Adaptive use of network-centric mechanisms in cyber-defense, *Proc. of the 6th IEEE International Symposium on Object-Oriented Real-Time Distributed Computing*, pp.183-192, 2003.

[9] D. Kewley, R. Fink, J. Lowry et al., Dynamic approaches to thwart adversary intelligence gathering, *Proc. of the DARPA Information Survivability Conference & Exposition II*, pp.176-185, 2001.

[10] B. Zhuge, B. Wang, Y. Wang et al., Architecture of SDN applications based on software defined price, *Telecommunications Science*, vol.31, no.8, pp.1-11, 2015.

[11] J. H. Jafarian, E. A. Shaer and Q. Duan, OpenFlow random host mutation: Transparent moving target defense using software defined networking, *Proc. of the 1st Workshop on Hot Topics in Software Defined Networks*, pp.127-132, 2012.

[12] N. McKeown, Software-defined networking, *Proc. of the INFOCOM Key Note*, http://infocom2009.ieee-infocom.org/technicalProgram.htm, 2009.