A SDN-based IP Address Hopping Method Design

Ke Zheng[1, a*], Xin Zhao[2, b*], Xiao Li[3, c*] and Yao Zhou[2, d*]

[1, 2, 4] 3rd Department, Academy of National Defense Information,Wuhan, China

[3] Science research department, Academy of National Defense Information,Wuhan, China

[a]237422762@qq.com, [b] freedomcoco8@gmail.com, [c] zk6468521@sina.com, [d] zy6541@sina.com

Keywords: SDN; proactive defense; IP hopping; cyberspace security

Abstract. To mitigate growing cyber threats, this paper builds on previous researches on proactive defense and Software Defined Network to propose a SDN-enabled IP address hopping method design, including the basic model, detailed hopping process, key generation and SDN controller.

Introduction

A fundamental cause of growing cyber threats originates from the defender's disadvantaged position against the attacker. Firewalls, access control, data encryption and other static defense mechanisms are under increasing pressure. Researchers began to focus on Moving Target Defense [1], a proactive security measure. As an integral part of proactive defense, IP address hopping tries to hide the identity of hosts and confuse the attacker by constantly changing the IP address in a random way. This is basically an idea inspired by the frequency-hopping technique in radio communication to prevent third-party detection, interception and jamming. Meantime, Software Defined Network was proposed as an innovative network structure. Now SDN has become a focal point for many researchers in the IT community and represents one of the major trends of future information networks. As a result, we propose a SDN-based IP address hopping mechanism.

Related work

Initially, the idea of IP address hopping was proposed in the APOD program of DARPA in an effort to seek proactive defense through the alternation of IP address. Researchers of the program successfully fended off cyber-attacks by establishing a hopping tunnel and hiding the identity of the host whose IP address was randomly distributed. A LAN-based IP hopping method was introduced in [2], which focuses on updating the IP address through dynamic configuration protocol to guard against worms. But the configuration duration is time-consuming as the link needs to be re-established every time when a hopping occurs. An OpenFlow-based host mutation method was proposed in [3], and it was later analyzed using Satisfaction Model Theory (SMT) to optimize the size of IP address pool. But it concentrates on theoretical analysis instead of putting forward a practical and feasible hopping process design or effectiveness assessment.

A SDN-based IP address hopping design

Most precedent researches focus on realizing IP address hopping at the host-end, which is cost-effective and easy to deploy. But the hopping rate and performance are relatively poor as a result of limited bandwidth and computation capacity of the host itself. As such, we propose a SDN-based IP address hopping mechanism that is supported by independent switches or routers. The separation between data panel and control panel in SDN not only make the network more flexible, but also establish a logically centric and programmable controller which provides a chance for us to generate and deploy the hopping strategy from the top.

A. Hopping model. In this part, we set a typical C/S-based scenario that is SDN-enabled to illustrate how the IP address hopping mechanism works. As in figure 1, the client first needs to be authenticated before gaining access to the network. Then it sends an inquiry message to the DNS

server to acquire the address of the current address of the target server. The generation and distribution of hopping message are managed by a single hopping controller which maps out hopping strategy in view of the security situation and deploy the strategy through the north interface. The OF switches in the picture indicate switch equipment that support OpenFlow protocol.

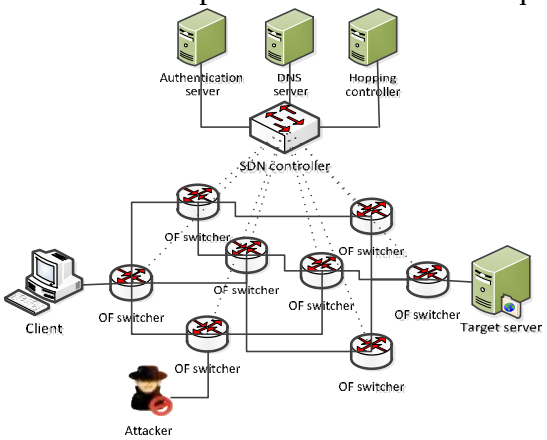

Figure 1. Illustration of SDN-based IP address hopping

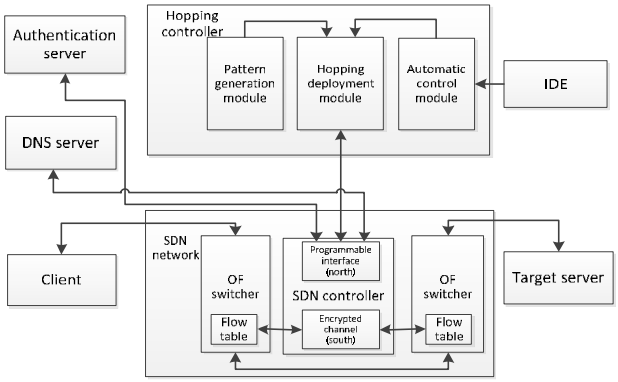

Figure 2. Structure of SDN-based port and IP address hopping

B. Basic structure. The whole system, built on a software-defined network, consists of 5 major components including the client, the target server, the hopping controller, authentication server and the DNS server, as suggested in figure 2.

C. Process design. First the client has to be verified by the authentication server before connecting to the hopping network. Then it sends a message to the DNS server to inquire the current IP of the target server. The hopping controller serves as the key in this as it generates hopping patterns and distributes them through the SDN network to achieve proactive defense. To simplify the process, we focus on the protection of the target server and assume the client is free from cyber threats. As such, we only showcase in figure 3 the hopping process design from the client to the server, instead of the other way around. The specific process is listed below.

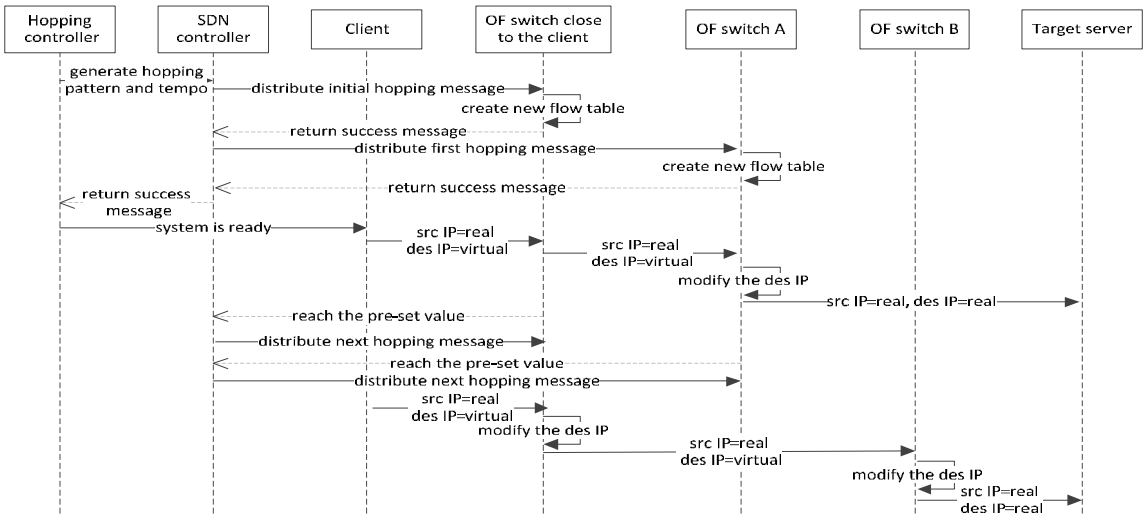

Figure 3. Hopping control process

- Upon receiving the message informing the access of new client, the hopping controller sets the hopping tempo based on the security situation and uses the random algorithm within to generate the hopping pattern from an IP address pool which is determined by the number of usable OF switches in the system. The information that contains hopping pattern and tempo is then sent to the SDN controller through the programmable interface (north);
- The SDN controller will then, through the encrypted OpenFlow-supported channel (south), distribute the hopping information to the OF switch close to the client and the OF switch A, which

corresponds to the network number of the initial hopped/virtual IP address. Any OpenFlow-enabled switch can modify the destination or source address of an IP packet in accordance with the flow table set by the SDN controller. The counter, a pre-designed mechanism in the OF switch, can serve as a trigger to the occurrence of next hopping;

- The OF switch near the client and the OF switch A will then translate the message from the SDN controller into content in a flow table, according to which packets are matched, counted and modified. A success message will be returned to the controller;
- When the whole system is ready, the client starts to send packets to the target server. Since the client has already acquire the first hopping IP (with which the OF switch A matches) of the target server by inquiring the DNS server, the destination IP of packets from the client is actually not the real IP of the target server and is labeled as virtual;
- When packets arrive, the OF switch at the client side checks the source and destination IP. If packets are confirmed to be from the client, the counter within the switch begins to count the number of packets coming through;
- According to the virtual destination IP address, packets from the switch close to the client are then transmitted to OF switch A which will check and verify the packets pursuant to the flow table within. If they match, the switch will replace the virtual destination IP with the real IP of the target server, and then start to count the number of packets. After that, those packets will be sent to the target server naturally without modification;
- When both the counters within the OF switch A and the switch near the client reach the limit, the trigger of next hopping, both switches will send message to the controller requesting for new hopping information. The controller will then distribute new hopping information, including the new virtual IP and maximum packet number, to the switch close to the client and the new OF switch B, both of which will generate a new flow table. And then the system enters the next hopping process.

D. Pattern generation. As the core of the hopping controller, the pattern generation module determines the pool of hopping IP addresses according to the useable OF switches in the network and the current security level. Relevant random algorithm and parameters are utilized to generate hopping patterns.

First, we need to identify a scope from which the hopping IP addresses are generated. According to the TCP/IP protocols, an IP address consists of two parts, the host identifier and the network number, which needs to be known firstly by the pattern generation module. To do this, the hopping controller identifies the number of useable OF switches in the network through the SDN controller, which can easily acquire the sub-network numbers these switch belong to. The hopping controller will enumerate these network numbers from 1 to n. The pool of network numbers of hopping IP addresses, that is, $N=\{ NID_1, NID_2, \ldots, NID_n \}$;

When the scope of network numbers is identified, we can easily formulate a hopping IP address as the host identifiers are pre-defined. For example, when the NIDx consists of 24 bits, that is, a C class network, then the set E of host identifiers is from 1 to 2^8. In this connection, the set H_{IP} of hopping IP $H_{IP} = \{N, E\}$. As the scope of hopping IP addresses is clear, we now move to the generation algorithm of hopping patterns. To ensure the safety and randomness of hopping patterns, the MD5 algorithm is adopted to generate hopping IP addresses using the current time T and random number R of client as inputs. And then we have $H_{IP} \leftarrow MD5(R, T)$. The random number R is pre-assigned by the hopping controller to every new client as the only identifier. And T is the current system local time. In this way, the same client uses different hopping IP addresses at different time interval; and different clients use dissimilar IP addresses at the same time interval. The hopping controller deploys the hopping information through the north interface that connects the SDN controller.

When the counter reaches the pre-set value, the next hopping is triggered. The controller uses the new system time and random identifier as inputs to generate a new MD5 result, which is then translated into hopping information.

E. SDN controller. The SDN controller needs to, 1) modify the DNS message from the DNS server to the client; 2) distribute and deploy hopping information to relevant switches.

The client sends a message to the DNS server requesting the IP of the target server. The return message from the DNS server will be modified by the SDN controller to replace the real IP of the target server with the first virtual/hopping one. The message will be finally sent to the client, which uses the virtual IP address to encapsulate its packets.

Those packets firstly arrive at the near-client switch, which has already been configured by the SDN controller and formed a new flow table as pictured in figure 4.

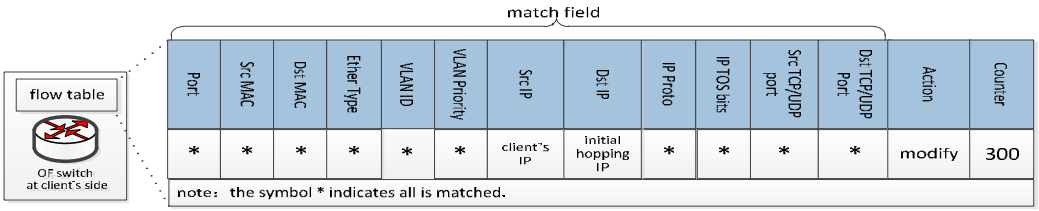

Figure 4. Flow table of the OF switch at client's side

For all packets coming through, the switch first matches its source and destination IP to check whether they are from the client. If confirmed, the switch will execute the ***modify_field*** action, that is, replacing the destination IP of the packets with the hopping IP, and then count the number.

When the counter reaches the pre-set value, the SDN controller will deploy and distribute the next hopping information to OF switches using the south interface (OpenFlow protocol).

Problems to be further solved

With the rapid development of SDN and proactive defense technology, the combination of the two will surely become a focal point of future research in the field of network security. Despite some work we may have done, there are still more to improve, including the designing of self-adaptive hopping mechanism which could automatically adjust the hopping tempo or scope to reflect changes in security level or data flow, the effectiveness analysis which involves cyber-attack experiments to demonstrate advantages of IP hoping over traditional security measures in defending networks, and the research on the protection of the hopping controller, which could face mounting threats since it now stands as the center of the whole hopping system.

References

[1] Jajodia S, Ghosh A K, Swarup V, et al. Moving target defense: creating asymmetric uncertainty for cyber threats[M]. Springer Science & Business Media, 2011.

[2] Antonatos S, Akritidis P, Markatos E P, et al. Defending against hitlist worms using network address space randomization[J]. Computer Networks, 2007, 51(12): 3471-3490.

[3] Jafarian J H, Al-Shaer E, Duan Q. OpenFlow random host mutation: transparent moving target defense using software defined networking[C]//Proceedings of the first workshop on hot topics in software defined networks. ACM, 2012: 127-132.

[4] Kampanakis P, Perros H, Beyene T. SDN-based solutions for Moving Target Defense network protection[C]//A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on. IEEE, 2014: 1-6.

[5] Wheeler B F. A Computer Network Model for the Evaluation of Moving Target Network Defense Mechanisms[J]. 2014.

[6] Krylov V, Kravtsov K. IP fast hopping protocol design[C]//Proceedings of the 10th Central and Eastern European Software Engineering Conference in Russia. ACM, 2014: 11.

[7] Wu Jiangxing. Pseudo - state calculation and the original intention and vision of mimic security defense [J]. Telecommunications Science, 2014 (7): 2-7.