2016 IEEE 41st Conference on Local Computer Networks

Mitigating Crossfire Attacks using SDN-based Moving Target Defense

Abdullah Aydeger*, Nico Saputro*, Kemal Akkaya*, and Mohammad Rahman†
*Dept. of Electrical & Computer Engineering,Florida International University, Miami, FL, 33174 USA
{aayde001, nsapu002, kakkaya}@fiu.edu
†Department of Computer Science, Tennessee Tech University, Cookeville, TN, USA 38505
marahman@tntech.edu

***Abstract*—Recent research demonstrated that software defined networking (SDN) can be leveraged to enable moving target defense (MTD) to mitigate distributed denial of service (DDoS) attacks. The network states are continuously changed in MTD by effectively collecting information from the network and enforcing certain security measures on the fly in order to deceive the attackers. Being motivated from the success of SDN-based maneuvering, this work targets an emerging type of DDoS attacks, called Crossfire, and proposes an SDN-based MTD mechanism to defend against such attacks. We analyze Crossfire attack planning and utilize the analyzed results to develop the defense mechanism which in turn reorganize the routes in such a way that the congested links are avoided during packet forwarding. The detection and mitigation techniques are implemented using Mininet emulator and Floodlight SDN controller. The evaluation results show that the route mutation can effectively reduce the congestion in the targeted links without making any major disruption on network services.**

Keywords: Software Defined Networking; Moving Target Defense; crossfire DDoS attacks, route mutation; data delay

I. Introduction

Recently, it has been reported that cyberattacks frequency, in particular distributed denial of service (DDoS) attacks, has continued to increase and represents more than 20 percent of all attacks [1] [2]. Such attacks cause a lot of revenue lost for companies [3]. While there have been plenty of detection and mitigation mechanisms for DDoS [4], they become obsolete due to the recent shift of the attacked targets, from directly attacking a specific target to indirect attack (e.g., [5] [6]). Moreover, the existing defense mechanisms are reactive approaches that detect and mitigate during the attacks (e.g., [7]).

Over the last few years, moving target defense (MTD) has received an increasing attention from the research community. The idea behind MTD is to dynamically change the configuration and behavior of the network in order to deceive adversaries and to make it harder for them to launch successful attacks to the networks [8]. The changes are typically controlled by an automated but centralized system. Recently, the emerging software defined networking (SDN) has been integrated with MTD [9]–[13].

In this paper, we propose an SDN-based MTD for proactive (i.e., before the attacks) and reactive (i.e., during the attacks) defense against crossfire attacks to a certain area. We implemented and evaluated the proposed approaches in a Mininet environment using SDN Floodlight controller. While we have the similar idea as in [14] that performs a proactive defense at the reconnaissance phase, in particular during the link-map construction, our proposed approaches are different from the existing works in two major ways: 1) the detection mechanism is based on the identification of source-destination pairs that are involved in *traceroute* operations and this is detected by SDN controller that has not been tried before; 2) the defense mechanism utilizes the collected information in the detection phase and applies MTD using SDN controller's ability to observe the switches continuously.

II. Related Work

The work on MTD with SDN focused on the change of different setting in the considered network such as changing the host IP addresses [9], [11], change routes randomly [10], or mutations on both the IP addresses and hosts [12]. On the other hand, [13] investigates changing the hosts IP address by involving the DNS interactions. A number of defense mechanisms exist for the link-flooding DDoS attacks are differed in a number of ways such as when they perform the detection and mitigation (e.g., proactive or reactive), how they detect the target links, and how they identify the attack agents. While most approaches are reactive [7], [15], [16], a proactive defense at the reconnaissance phase, in particular during the link-map construction [14] is proposed recently. The anomalies of path performance metrics are correlated to the *traceroute* data to infer the target links or area [16]. To identify the attack agents, enforcing to behave suspiciously by not complying to certain actions such as rerouting requests to follow the preferred paths or rate control requests is pursued [15]. However, none of the existing works considered SDN-based defense for crossfire attack mitigation.

III. Preliminaries

A. Network and Threat Model

The considered network model is an autonomous system (AS) of an Internet Service Provider (ISP) that provides internet connectivity services to customers. These customers may have their own private networks which are managed by themselves (i.e., smaller ASes). The network infrastructure for this ISP-controlled AS is assumed to be SDN-based so that the ISP can manage the traffic flows within its AS using SDN.

DOI 10.1109/LCN.2016.108

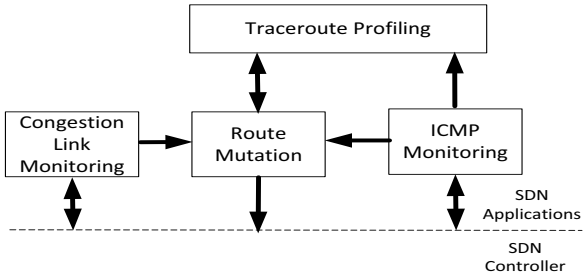

Figure 1: The proposed SDN-based MTD modules

The considered threat model is the *crossfire attack*. This indirect attack strives to build the link-map of the network using the *traceroute* messages, finds the critical links, and then floods the selected links with multiple low-rate flows from a plethora of coordinated, compromised devices to exhaust the available bandwidth on those links. In this way, an attacker can isolate a specific area within an ISP-controlled AS by performing link-flooding DDoS attacks to some targeted links so that the servers within that area are unable to provide their services.

B. Problem Definition

The problem definition can be divided into two parts, the prevention problem for the link-map construction and the detection and mitigation problem during the attack. During the link-map construction, an attacker strives to find the persistent links which becomes the candidate for the targeted link-flooding DDoS attacks. The persistent links of a route from a source to a destination are links that always present each time a pair of attack agents performs reconnaissance, while transient links of a route are links that may not always present. Typically, the transient links happen because of the implemented routing protocol operations such as load-balancing routing which may cause a route from source to destination to change every time. Therefore, at the prevention stage the problem that needs to be addressed is *how to identify and obfuscate the link map construction effort so that the cost of attacking the targeted links are high and make it less attractive for an attacker*.

During the attack, link-flooding DDoS attack is difficult to detect since an attacker uses huge amount of bots that send low-rate persistent small traffic through the targeted links. This way, the targeted links will be flooded with huge amount of low-rate small traffic from plethora of bots. To identify the bots that are sending these traffic, typically the defender will request sources to confront a certain request and when the act is suspicious (i.e., not following the request), these sources can be considered as bots and further actions can be performed for these suspects. Instead of bots detection, our goal is to use the same traceroute profiles that are collected in the prevention stage, for the detection and mitigation during the link-flooding DDoS attack stage. Hence the problem definition would be *how to do detection and mitigation against crossfire DDoS attacks using SDN by utilizing the traceroute profiles collected from the network*.

IV. Proposed SDN-based MTD Approaches

Our proposed approaches consist of two defense mechanisms: (1) obfuscating the links during the potential link-map creation of the attackers to make it harder to launch the attacks (i.e., proactive stage), and (2) detection and mitigation during the attacks (reactive stage). To perform these defense mechanisms, we relied on the abilities of SDN controller and OpenFlow protocol. Specifically, four inter-related SDN application modules are designed and deployed as depicted in Fig. 1: (1) ICMP monitoring, (2) *traceroute* profiling, (3) route mutation, and (4) Congestion-Link monitoring. The first three modules are required for the proactive stage while the last three modules are needed for the reactive stage. Note that *traceroute* uses ICMP packets. The detail of each module is explained next.

A. ICMP Monitoring Application

ICMP Monitoring continuously monitor the presence of ICMP packets by requesting every ICMP packet to be sent to the controller through *packet-in* messages of OpenFlow. On receiving these ICMP packets, the ICMP Monitoring application tries to determine whether the *traceroute* operations are present in the network by looking at the following patterns: (1) there are multiple ICMP echo packets with increasing number of TTLs from a source to a certain destination, and (2) there are multiple ICMPs with TTL exceeded information packets to specific destination followed by ICMP with destination unreachable to the same destination. When these patterns are detected, ICMP monitoring application sends the *traceroute* information to *traceroute* profiling application.

B. Traceroute Profiling Application

On receiving *traceroute* information, the proposed application builds the *traceroute* profile database that consists of the following information: source IP address, destination IP address, timestamp, the closest SDN switch to the source IP address, the closest SDN switch to the destination IP address, and all intermediate SDN switches between source and destination where this *traceroute* is detected. Note that the traffic may be generated from outside the AS, and thus the closest devices here can be switches at the boundary of the network.

Since the ICMP packets can continuously arrive at the SDN controller and the database can be updated anytime, this application will try to identify the potential target links within a T time interval when an excessive number of *traceroute* attempts are detected. *Traceroute* profiling module uses the *traceroute* profile database to identify the potential target links that can be attacked based on the number of *traceroute* that may pass through the common links within that T time interval. When such common links exist, the *traceroute* profiling module informs the route mutation module to find all alternate routes that will not pass through these potential target links, and then sets rules of the SDN switches to randomize the routes for ICMP packets from suspected sources.

When the route mutation operations are active at the SDN switches, the SDN controller can decide to terminate the route mutation on the switches when the SDN controller detects that the number of *traceroute* attempts has dropped significantly within the T time interval.

C. Route Mutation Module

From the *traceroute* profiling module, the potential target links and suspected sources can be identified. Route mutation module strives to find all possible routes from suspected host to destination by using our functions implemented on Controller. These functions are implemented at FloodLight SDN Controller. Our first function takes start and end hosts, and the potential target links that should be avoided. By using these input parameters, it finds all possible routes available. Essentially, this is providing link-disjoint routes from a source to destination.

As an example of these algorithms, by considering our experiment topology in Fig. 2, let us assume there is a congestion at the link between switches S3 and S5. First, the most source-destination packet transmission of this link will be figured out. Let it be H1 to H4. Then first function will be called with H1 as start node, H4 as the end node and S3-S5 link is given as a link not to be used. In this algorithm, all possible links that can be followed just after H1 will be considered as potential routes and recursive function will be called to see whether they can go until the destination host or not. In last function, if a route is found, it will be added to the list by verifying that it does not include any links from list of links that should not been used, link S3-S5 in our case. After running the our algorithm, it will return all possible routes which are H1 - S1 - S2 - S4 - S6 - S9 - H4 and H1 - S7 - S8 - S9 - H4 in our example.

D. Congested-Link Monitoring

Concurrently with ICMP monitoring, Congested Link Monitoring is basically performed at the SDN Controller. Whenever the SDN Controller detects a congested link, source-destination pair that is using this link will trigger route mutation for itself. The controller will consult the *traceroute* profiling and see whether there is previous history of the pair of source-destination that also sent *traceroute*. If there are, then controller will set new rules to all switches for the route picked randomly among all possible ones.

V. Experimental Evaluation

A. Environmental Setup

For our experiments, a module is implemented at FloodLight controller. The *packet-in* handling and route mitigation is performed by using this module. The *packet-in* feature of SDN enables directing the packets to the controller so that some processing can be done. We used a small network topology that has 9 Mininet switches and 4 hosts, where 2 hosts act as clients (H1 and H2) and the other hosts act as servers (H3 and H4) as shown at Fig. 2.

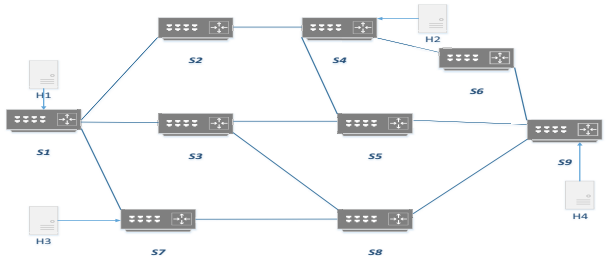

Figure 2: Our network topology

For the traceroute profiling phase, all the switches are set the rule to drop all TCP/UDP packets so that we can initially only allow ICMP packets. The switches are set with the rule for ICMP packets with output to Floodlight Controller as OpenFlow *packet-in* message and to an output to port which is defined manually. ICMP packets are generated and transferred between hosts for one minute. After this phase, real data transfers are started and maintained for 5 minutes. The Floodlight Controller set a rule for each switch so that whenever it receives a TCP/UDP packet, it will forward it to both the Floodlight Controller as *packet-in* and to the next hop according to route that is defined. FloodLight Controller checks the link usage every 10 seconds. If data link transmits more than threshold, then the route that source-destination pair is using will be mitigated.

As the baseline approach, we consider the case where we do not apply MTD and continue with the initial network configuration. We represent these cases as MTD and No-MTD in the graphs. We consider the two metrics to assess the performance of the above approaches: (1) *Average Link Usage*, and (2) *Average end-to-end Delay*.

B. Experimental Results

1) TCP Experiments: We first conducted experiments by using TCP. As can be seen from Fig. 3, when MTD takes a place, there is a significant reduction in average link usage in the network.In addition, the route mutation approach provides some load balancing among the link usages of different links. The usage in the first three cases are almost similar and only in the case of S5.P2 and S5.P3 there is a slight increase which is normal.

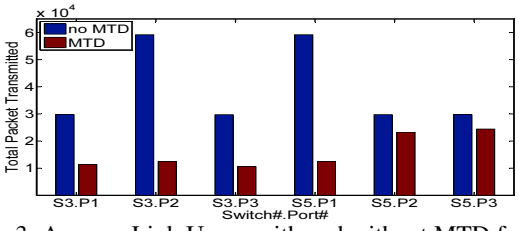

Figure 3: Average Link Usage with and without MTD for TCP Transmission. S3.P1 means port 1 at switch id 3, S5.P2 means port 2 at switch id 5 and so on.

We then looked at the average delay performance to see if the proposed MTD approach made any significant impact on this metric. The results are shown in Fig. 4. We can see that

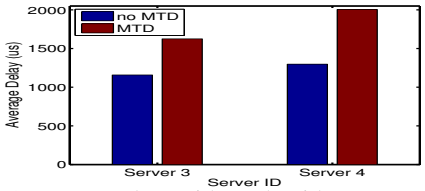

Figure 4: Average Delay with and without MTD for TCP Transmission for Server 3 (H3) and Server 4 (H4)

there is an increase in delay for the packets arriving at both servers. The reason behind this, since the routes are changed during the process, the new routes may not always be the shortest path routes in terms of packet delay. An increased number of hops, for instance, may increase the end-to-end packet delay.

2) Impact of the Frequency of Checks: Finally, we investigated if there is an impact of the frequency of running the proposed algorithms to check whether there is congestion or not. In this experiment, we changed it to 5 and 20 secs respectively. We observe that when the frequency of checking is increased, this helps to alleviate the congestion in most of the links. However, this may increase the packet delay due to randomized path selection.

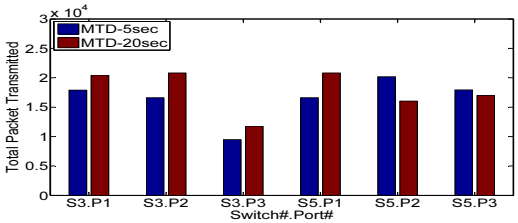

Figure 5: Average Link Usage of MTD when frequency is varied

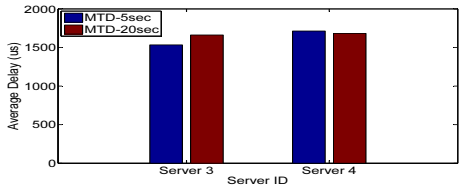

Figure 6: Average Delay of MTD for Server 3 (H3) and Server 4 (H4) when frequency is varied

VI. Conclusion

In this paper, we presented how the emerging SDN paradigm could be useful for MTD, especially to defend Crossfire attacks. Specifically, we first presented an approach to detect attacks at the planning phase by considering the traceroute messages. After this phase of traceroute profiling, we proposed an attack mitigation approach using route randomization by also utilizing the information from the profiling phase. Experimental results indicated that our proposed SDN-based MTD methodology can effectively decrease the chance of link flooding by checking each link regularly and changing routes, and thus lessening on congested links. On the other hand, because of the route updates, MTD will create a bit more delay especially for those TCP packets flowing at the same time of route mutations.

For future work, we would to evaluate the scalability of our approaches and improve the MTD mechanism by incorporating machine-learning.

VII. Acknowledgement

This work is supported in part by a seed grant from Florida Cybersecurity Center.

References

[1] VERISIGN, "Verisign distributed denial of service trends report q3 2014." [Online]. Available: http://www.verisigninc.com/assets/

[2] A. Networks, "10th annual worldwide infrastructure security report (wisr)," 2014. [Online]. Available: http://www.arbornetworks.com

[3] Incapsula, "What ddos attacks really cost businesses," 2014. [Online]. Available: http://lp.incapsula.com

[4] S. T. Zargar, J. Joshi, and D. Tipper, "A survey of defense mechanisms against distributed denial of service (ddos) flooding attacks," *IEEE Communications Surveys Tutorials*, vol. 15, no. 4, pp. 2046–2069, Fourth 2013.

[5] A. Studer and A. Perrig, "The coremelt attack," in *Proceedings of the 14th European Conference on Research in Computer Security*, ser. ESORICS'09. Berlin, Heidelberg: Springer-Verlag, 2009, pp. 37–52. [Online]. Available: http://dl.acm.org/citation.cfm?id=1813084.1813088

[6] M. S. Kang, S. B. Lee, and V. D. Gligor, "The crossfire attack," in *Proceedings of the 2013 IEEE Symposium on Security and Privacy*, ser. SP '13. Washington, DC, USA: IEEE Computer Society, 2013, pp. 127–141. [Online]. Available: http://dx.doi.org/10.1109/SP.2013.19

[7] C. Liaskos, V. Kotronis, and X. Dimitropoulos, "A novel framework for modeling and mitigating distributed link flooding attacks," in *IEEE INFOCOM*, San Francisco, CA, USA, Apr 2016.

[8] T. E. Carroll, M. Crouse, E. W. Fulp, and K. S. Berenhaut, "Analysis of network address shuffling as a moving target defense," in *Communications (ICC), 2014 IEEE International Conference on*. IEEE, 2014, pp. 701–706.

[9] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "Openflow random host mutation: Transparent moving target defense using software defined networking," in *Proceedings of the First Workshop on Hot Topics in Software Defined Networks*, ser. HotSDN '12. New York, NY, USA: ACM, 2012, pp. 127–132. [Online]. Available: http://doi.acm.org/10.1145/2342441.2342467

[10] P. Kampanakis, H. Perros, and T. Beyene, "Sdn-based solutions for moving target defense network protection," in *A World of Wireless, Mobile and Multimedia Networks (WoWMoM), 2014 IEEE 15th International Symposium on*. IEEE, 2014, pp. 1–6.

[11] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "An effective address mutation approach for disrupting reconnaissance attacks," *Information Forensics and Security, IEEE Transactions on*, vol. 10, no. 12, pp. 2562–2577, 2015.

[12] A. Chavez, W. M. Stout, and S. Peisert, "Techniques for the Dynamic Randomization of Network Attributes," in *Proceedings of the 49th Annual International Carnahan Conference on Security Technology*, Taipei, Taiwan, Republic of China, Sept. 21–24, 2015.

[13] D. C. MacFarland and C. A. Shue, "The sdn shuffle: Creating a moving-target defense using host-based software-defined networking," in *Proceedings of the Second ACM Workshop on Moving Target Defense*. ACM, 2015, pp. 37–41.

[14] T. Hirayama, K. Toyoda, and I. Sasase, "Fast target link flooding attack detection scheme by analyzing traceroute packets flow," in *Information Forensics and Security (WIFS), 2015 IEEE International Workshop on*, Nov 2015, pp. 1–6.

[15] S. B. Lee, M. S. Kang, and V. D. Gligor, "Codef: Collaborative defense against large-scale link-flooding attacks," in *Proceedings of the Ninth ACM Conference on Emerging Networking Experiments and Technologies*, ser. CoNEXT '13. New York, NY, USA: ACM, 2013, pp. 417–428. [Online]. Available: http://doi.acm.org/10.1145/2535372.2535398

[16] L. Xue, X. Luo, E. W. W. Chan, and X. Zhan, "Towards detecting target link flooding attack," in *Proceedings of the 28th USENIX Conference on Large Installation System Administration*, ser. LISA'14. Berkeley, CA, USA: USENIX Association, 2014, pp. 81–96. [Online]. Available: http://dl.acm.org/citation.cfm?id=2717491.2717497