Design of a Network Scan Defense Method by Combining an SDN-based MTD and IPS

Shoya Chiba*, Luis Guillen†, Satoru Izumi‡, Toru Abe*§, and Takuo Suganuma*§
*Graduate School of Information Sciences, Tohoku University, Sendai, Japan
†Research Institute of Electrical Communication, Tohoku University, Sendai, Japan
‡National Institute of Technology, Sendai College, Sendai, Japan
§Cyberscience Center, Tohoku University, Sendai, Japan
Email:{shoya.chiba, lguillen}@ci.cc.tohoku.ac.jp, izumi@sendai-nct.ac.jp, {beto, suganuma}@tohoku.ac.jp

***Abstract*—This paper proposes a Software-Defined Network (SDN)-based Moving Target Defense (MTD) mechanism to protect the network from potential scans. The proposed mechanism can work in combination with an IPS without affecting its normal behavior. To do so, an SDN controller changes the packets' headers passing through switches using virtual IP addresses while the operation of IPS continues monitoring the devices' actual IP addresses. Preliminary results in an emulated environment show that it is possible to achieve a seamless collaboration between the MTD and IDS to detect low and high-rate scans.**

***Index Terms*—MTD, IPS, Network Scan, SDN**

I. Introduction

To prevent and protect from the ever-increasing cyber-attacks, several defense technologies such as Intrusion Prevention System (IPS) and Firewall have been proposed. In particular, the most frequently occurrences are Advanced Persistent Threat (APT) attacks such as ransomware [1]. These attacks are performed for various purposes, such as kidnapping the data stored in servers or hosts in the network and requesting money to release them. The attackers who break into the network, and then scan it to search for any useful information, including the status of the hosts/servers, their configuration, among others. For example, Wannacry [2], one of the most well-known ransomwares, performs network scans after infection. If these scans are successful, the attackers may be collecting the information necessary to proceed to a higher level attack of internal resources. Moreover, it can also be a precursor to the spread of infection. Therefore, preventing such scans is necessary to prevent the attackers from obtaining information and executing attacks.

In traditional networking, IPSs, such as Snort [3], are used as a defensive and preventive measure against scans. An IPS can detect and block suspicious activity such as scanning that occurs in the network. However, IPS's rules are mainly set for detecting high-rate scans, but they are not effective when detecting low-rate scans. For instance, attackers might use slow scans to circumvent detection by scanning a single IP address every certain time, or use other typical scan patterns that pings a single destination every second [4].

As a complementary mechanism, Moving Target Defense (MTD) [5] has proven effective to deal with this problem. The overall idea in MTD is to protect from the attacks by frequently moving the targets of those attacks. As a result, the information obtained by the attacker scanning at low-rate is rendered useless in a short time. However, the process of changing the address one after another may affect the IPS for monitoring of hosts inline. To avoid this undesirable effect and prevent all types of scans, using only MTD, the frequency of moving the target will need to be increased. However, this will certainly increase the load on the management.

Moreover, Software Defined Network (SDN) [6] has become more widespread, moving from the tightly-coupled control-data plane to a separated planes. This allows to add innovative and flexible ways to manage a network. As a result of this shift on the networking management, there have been also proposals to implement MTD mechanisms using SDN [7], [8]. However, as explained above, the interaction with other network elements such as IPS or Firewalls is still an open issue.

Therefore, in this study, we propose a method for configuring MTD that can transparently interact in combination with IPS. Specifically, the proposed MTD is realized using SDN, where the controller implements the MTD functions to change the IP address of the packet passing on an SDN switch so that it does not interfere with the operation of the IPS. The combined usage of these elements will allow the detection both low and high-rate scans. We conducted initial experiments in an emulated environment, and preliminary results show that the proposed mechanism is feasible and effective compared to existing approaches.

II. Related Work

An MTD that changes the IP address at regular intervals has been proposed for obfuscation of scan in [7]. This method is designed for OpenFlow (OF)-based Networks, and its main feature consist of rewriting the IP address of the received packet header received at the OF switch. Therefore, to realize the MTD strategy, the authors change the real IP address (*rIP*) to a different virtual IP address (*vIP*). This mapping between *vIP* and *rIP* is updated at regular intervals, making the information obtained by low-rate scan invalid since; it is updated by the time the scan runs again. However, this method notifies the *vIP*s using a DNS, as shown in Figure 1. Thus, an attacker can obtain the *vIP* from the DNS, in which case, the

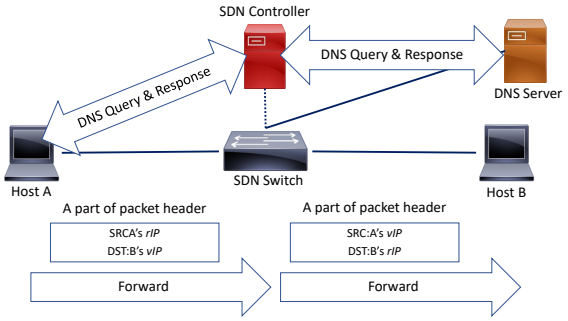

Fig. 1. Overview of OF-RHM [7]

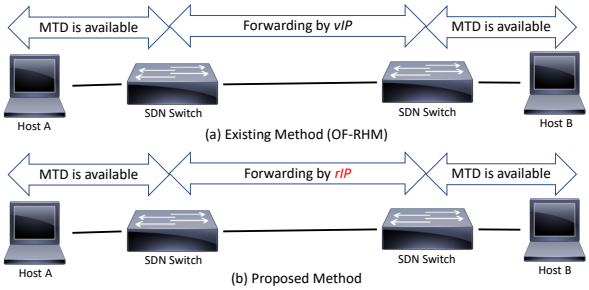

Fig. 2. Overview of the proposed method

attack is performed on the upper layer. Normally, an inline IPS will be used in this case to detect intrusions. However, since this method rewrites the header when the switch forwards the packet, the mechanism is not compatible with the IPS. For example, even if an IPS detects the attack and performs the corresponding blocking procedure according to its rule-based policy, it makes it difficult to compatibilize with the MTD strategy, as the the addresses changes and the rule is not matched.

Other adverse effects on security mechanisms are explained in [9]. In this work, the authors argue that IP address randomization MTDs should be applied only where they do not affect the security mechanisms already deployed. Otherwise, it cannot be guaranteed that the deployed security mechanism will work properly, and in the worst case, the existing security mechanism will be removed.

In [8], the authors present a method to apply MTD efficiently by using IDS. This method uses the IDS to identify suspicious hosts while at the same time it also randomizes the IP addresses for obfuscations. Then, only the ports of suspicious hosts are blocked. However, the mechanism to change the IP address of MTD is the same as [7], and the effect on IPS installed inline is not considered. In addition, when MTD and IPS are used for defense of scans, it is necessary to determine the address change interval and scan detection time slot according to the attacker's strategy. For example, in order to avoid the effect of MTD on IPS, if the IPS alone prevents both normal and slow-rate scans, it is necessary to set a long detection time slot, which increases false detection rate and detection delay. On the other hand, if the MTD alone prevents both low- and normal-rate scans, it is necessary to set a short address change interval. This increases the load on the SDN controller that manages the MTD, and increase the communication delay. Therefore, multi-faceted protection using MTD and IPS together shown in this work is ideal; but, even if a scan from a malicious host is detected, continuous monitoring will be performed and the communication may not be blocked.

III. Proposed Anti-Scan MTD Mechanism Transparent to IPS

A. Overview of the proposal

As explained in Section II, it is difficult for MTD and IPS to properly determine the address change interval and scan detection time slot. Therefore, it is not realistic to prevent all scans and attacks with only an MTD strategy, the same is true when using IPS alone. In actual operation, complementary interaction with different security solutions is required. However, the above-mentioned properties of MTD may adversely affect this interaction. For this reason, we cannot deploy IPSs, FWs, or other network elements in their normal states in a network using MTD.

Therefore, in this study, we implemented an SDN-based MTD method that provides address randomization to end hosts while performing transparent address change of packets for IPS. Hence, IPS can be observed based on *rIP* that does not change over time. In addition, this proposal can be used as an effective anti-scan method even when it is deployed in combination with an IPS.

Figure 2 shows a comparison of the proposed and existing methods. As observed,in Figure 2 (a) one of the existing methods [7], packets traverse between switches based on their *vIP*s. However, the *vIP*s change frequently; therefore, even if an IPS is placed between switches, the traffic cannot be monitored correctly. On the other hand, in our proposed method Figure 2 (b), switches transfer packets based on *rIP*, and uses the *vIP*s only between end hosts.

In the proposed mechanism, an IPS can monitor traffic based on the *rIP*, therefore traffic can be continuously blocked without being affected by the MTD strategy. Moreover, IPS can detect high-rate scans that are completed within the MTD interval and block compromised hosts without interference, making it possible to take measures against both low- and high-rate scans.

B. Packet rewriting procedure

When a host communicates within the network, it has to firstly resolve the IP address using ARP, then it needs to obtain the MAC address, and finally sends the IP packet. Therefore, the MTD mechanism requires rewriting the headers of ARP

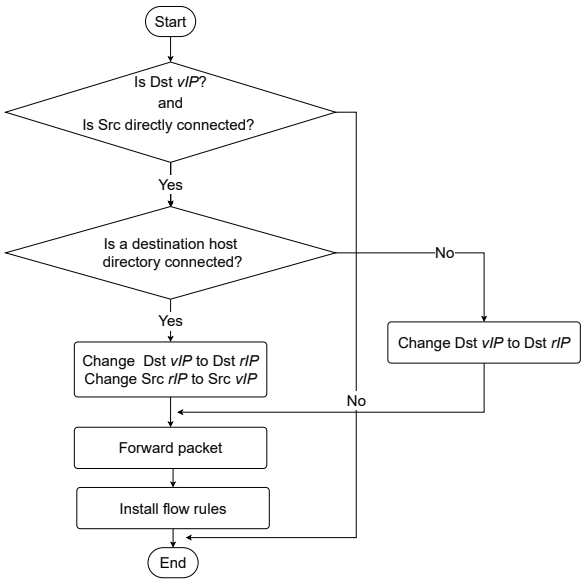

Fig. 3. Flowchart of *vIP* packet rewriting

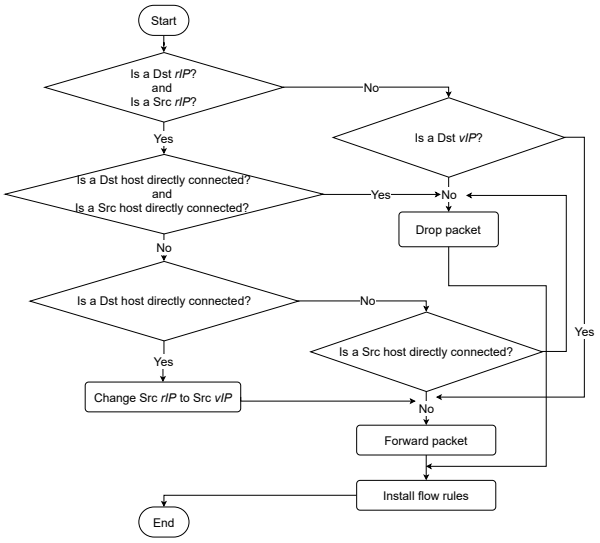

Fig. 4. Flowchart of *rIP* packet rewriting

packets and IP packets. In the particular case of an SDN-based MTD, when an SDN switch receives ARP packets, it rewrites the source/destination IP addresses. Similarly, when it receives IP packets, it rewrites their IP addresses.

In the proposed method, it is necessary to rewrite the destination and source to *rIP* when transferring packets between switches. Therefore, the controller installs a flow rule on each switch to perform this task, as described in Figure 3 and 4.

First, Figure 3 shows a flowchart with the procedure to handle the packets sent from a host directly connected to the switch to a *vIP*. As observed, if the switch receives a packet from a host connected to it and the destination of the packet is a *vIP*, then the controller checks whether the host corresponding to the destination *vIP* is directly connected to the switch. In case the hosts are directly connected via the same switch, the controller installs flow rules to rewrite the destination *vIP* to *rIP* and the source *rIP* to *vIP* to forward the packet. On the other hand, when hosts are not directly connected in the same switch, then the controller installs flow rules that the switch rewrites *vIP* to *rIP* and transfers it to the next switch.

Next, Figure 4 shows the flowchart to handle part of the packet sent from another switch. As observed, when the switch receives a packet of *rIP* for both the source and destination, if the destination host and the source host are connected via the same switch, then the packet is dropped. This is because the MTD of this proposal does not allow connection with *rIP*. However, when the packet was received from another switch and the destination host was connected to that switch, the controller installs flow rules to rewrite the source *rIP* to *vIP*, forwards it to that host, and installs it on the switch. Finally, if the destination host is not connected to the switch, it installs flow rules to only route the the packets in their current state.

By performing this process in cooperation with the SDN switch and the controller, the MTD mechanism can be applied between end hosts while maintaining transparency to the IPS deployed inline.

C. Interaction between the Proposed MTD with an IPS

In the proposed mechanism, the destination and source IP addresses of the traffic flowing between the switches are represented by the *rIP*, that does not change over time. Therefore, IPS can be placed between these switches and operated in a complementary manner with MTD.

For example, if the MTD address change interval is 60 seconds, the attacker must complete the scan and move to the attack phase within that time frame. Therefore, we set the scan detection unit time of IPS within the same time-frame (e.g., 60 seconds), and add a rule to isolate the host when a scan is detected. As a result, if an attacker performs a high-speed scan, it will be blocked by the IPS, and if a low-speed scan is performed, the reconnaissance activity will be hindered by the MTD.

Note that, since the notification mechanism of the *vIP* change is based on [7], it is possible that the DNS reconnaissance attack reaches the target host without affecting the MTD. Moreover, when using the method proposed in [7] together with an IPS, the attack packets can be blocked. However, the traffic from the IP-based blockage of hosts is temporarily suspended until its *vIP* changes. By contrast, in the proposed mechanism, when the IPS detects an attack packet, both the traffic and the compromised host can be blocked without interruption.

D. Implementation

To deploy the proposed mechanism in an SDN-based environment, the following functions are required in the controller:

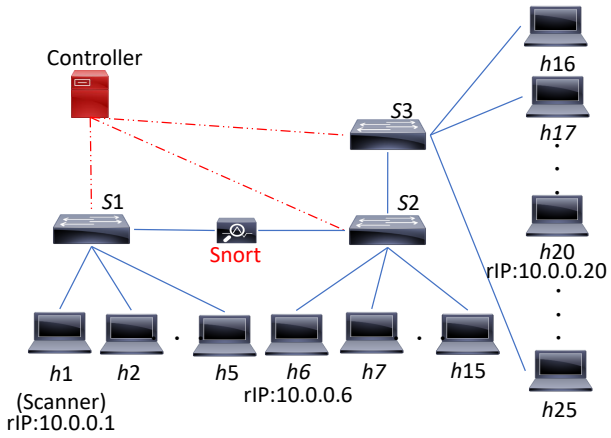

Fig. 5. Topology used in the experiments

1) Address change mechanism to realize an MTD transparent to IPS
2) *vIP* change notification mechanism for the DNS
3) A mechanism to prevent ongoing communication disconnection due to *vIP* change

For the current implementation, we use Ryu SDN Framework [10] as the OpenFlow controller. Moreover, to test the interaction with a real IPS, we used Snort to monitor traffic.

IV. Evaluation

A. Experimental Setup

To test the proposed approach, we conducted various experiments to:

1) Evaluate the effectiveness as a scan countermeasure
2) Verify the appropriate procedures to block the malicious host without being affected by the MTD
3) Evaluate the impact on the traffic delay and throughput

To evaluate these items, we built the topology shown in Figure 5 using the network emulator Mininet 2.3 [11], deployed in an Ubuntu virtual machine hosted on a Windows PC. As observed, this topology consists of 25 hosts, 1 IPS (Snort 2.9), and 3 switches. In addition, the bandwidth of all links is set to 1000 Mbps. The machines specifications are as follows:

- PC
  - OS: Windows10 21H1
  - CPU: Core-i7 9700 8 CORE
  - RAM: 64GB
- Virtual Machine
  - OS: Ubuntu 20.04LTS
  - CPU: 2 CORE vCPU
  - RAM: 4GB

Snort 2.9 is installed and executed in the Ubuntu virtual machine to simulate the behaviour of an inline IPS, which is then executed in a Mininet host.

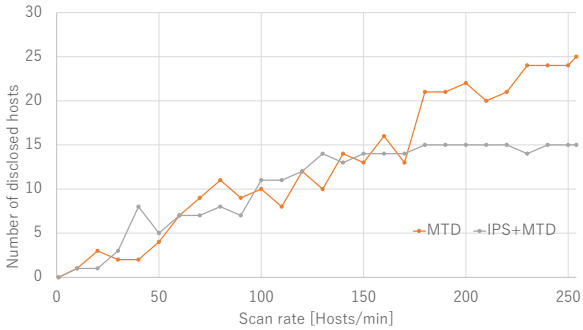

Fig. 6. Number of disclosed host per approach

B. Methodology

Initially, we performed an ICMP scanning from one of the hosts ($h1$) in the topology shown in Figure 5.

Then, we compare the results of the detection when 1) both MTD and IDS are installed and running, and 2) when only the MTD was installed as a countermeasure.

The scan rate was set from 1[Hosts/min] to 254[Hosts/min]. Also, Snort was programmed to detect and block 10 ICMP packets to other hosts per minute as a scan.

Under those conditions, we measured the effects of the scans completed within the address change interval, and the number of active hosts disclosed in a one-minute scan attempt.

C. Results

1) Disclosed Host Per Approach: As observed in Figure 6, when the IPS and MTD are used together, the number of disclosed hosts is limited to 15 because the scanning activity is restricted by the IPS, based on the scan rate. On the other hand, when using only MTD, when the scan rate increases, the number of hosts disclosed by scanning also increases. It is worth noting that, most hosts can be scanned when the rate exceeds 200. Therefore, we can conclude that, the combined use of IPS with MTD based on the proposed method or existing method will improve the effectiveness of scan countermeasures.

2) IPS Transparency: Next, we tested whether the IPS can block the host without being affected by MTD in the proposed method. To do so, we used the topology in Figure 5, and confirmed the blockage of the communication by the IPS between the hosts using ICMP. In this experiment, Snort was placed between switches $S1$ and $S2$ (in Fig. 5) as the IPS needs to detect the ICMP communication to confirm the scan. Initially, Snort has a blocking rule for hosts that attempt to connect to others more than 10 times in 60 seconds. Therefore, if the scanner in $h1$ pings to the host connected to $S3$ for more than 10 times, it will be blocked.

At this time, with the existing method, packets whose destination and source are represented by their *vIP* passes on to Snort, hence the IPS set a rule with the *vIP*.

However, on the proposed method, the rule is set to the *rIP* in Snort; thus, the traffic will pass.

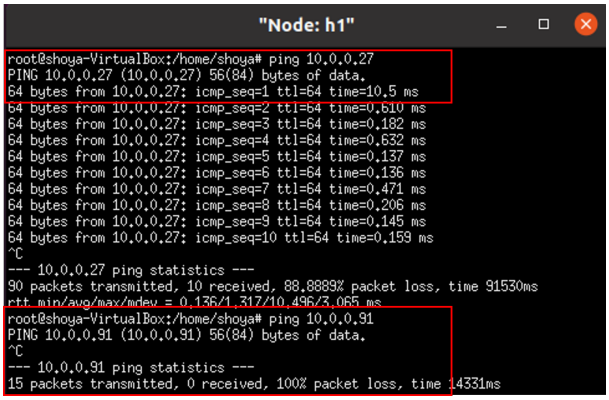

Fig. 7. Ping output of the proposed method when using an IPS

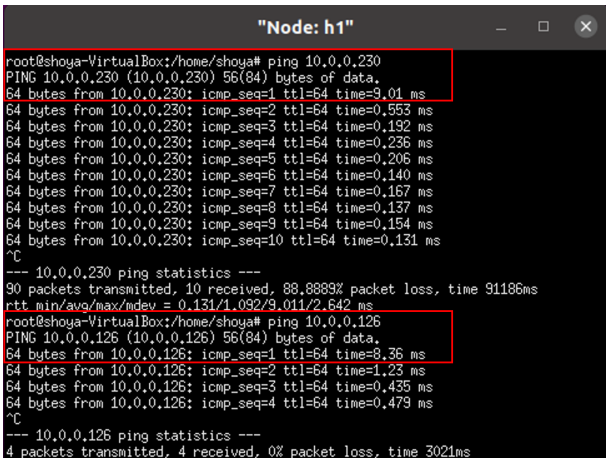

Fig. 8. Ping output of existing method [7] when using an IPS

Note that, in this paper, the notification mechanism of *vIP* by the DNS was not implemented. Therefore, a new *vIP* will be assigned based on the mapping table of *vIP* and *rIP* of the controller. However hosts cannot get a new *vIP*. Based on this premise, the address change interval was set to 120 seconds, so that the MTD can manually get the operation time responding to the *vIP* change. As observed in Figure 8, the communication is confirmed again to the new *vIP* by referring mapping table of the controller.

A sample output ICMP command in the proposed and in the MTD-only approaches is shown in Figure 7 and Figure 8 respectively. Figure 7 shows that $h1$ sent to ICMP packets to *vIP* 10.0.0.27 assigned to h20, the communication was cut off when the ICMP packet has transmitted 10 times. Then, 120 seconds later, $h20$ get a new *vIP* address 10.0.0.91, $h1$ sends ICMP packets by ping command to 10.0.0.92. However, there was no response from $h20$, indicating that communication was continuously blocked even after the address was changed.

On the other hand, Figure 8 shows that $h1$ sent ICMP packets to *vIP* 10.0.0.230 of $h20$, when the ping was sent 10 times it was blocked. This is the same behavior as the proposed method. However, when a new *vIP* is assigned to $h20$ after 120 seconds from the start of the experiment, the

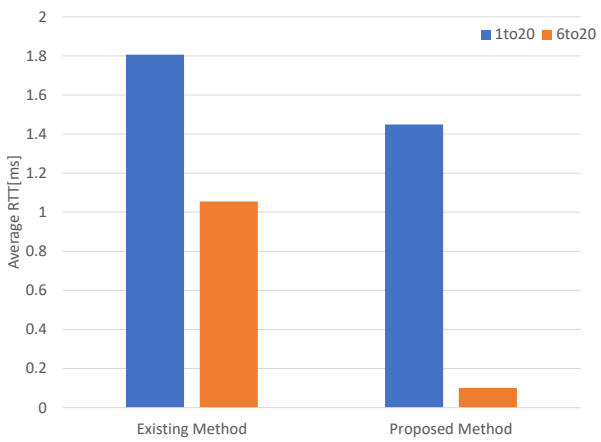

Fig. 9. Average RTT per approach

newly sent ICMP packets from $h1$ to 10.0.0.126, unlike the proposed method, there is a response from 10.0.0.126. This is because, both the destination and the source IP address of the packet passing through the IPS are represented by *rIP* in the proposed method.

From these results, we can conclude that when an IPS is used together with the MTD proposed in [7], the IPS functions such as traffic monitoring and blocking are affected by address changes. By contrast, with the proposed method, these functions are not affected; therefore, it is considered possible to detect high-speed scans and attacks and block compromised hosts.

Furthermore, the information obtained by the low-rate scans in shown on the left side of Figure 6, are reset each time the address is changed by MTD. Therefore, the combined use of MTD and IPS based on the proposed method is considered to be useful as a countermeasure for scans that can handle both low-rate scans and high-rate scans.

3) Communication Delay and Throughput: Finally, we evaluated the effect of using the proposed method on the communication delay and throughput. For this experiment, we used two controllers, one based on the existing method [7] as explained in Figure 1 and one based on the proposed method. We used the ping command to measure the delay and iPerf command to measure the throughput between $h1$ and $h20$ where the traffic pass through IPS on the topology of Figure 5. Moreover, we also measured the same items for $h6$ to $h20$ where traffic does not pass through the IPS.

Figure 9 shows the average value of the RTT when the ICMP Echo Request and Echo Reply are exchanged 10 times using the ping command.

When going through the IPS, the average RTT value of the existing method is 1.8 milliseconds, and the average RTT value of the proposed method is about 1.4 milliseconds. The RTT was even smaller for the ping between $h6$ and $h20$ that did not go through IPS, and in either case, no significant delays were measured that affected the actual operation; in fact, the RTTs were no higher than 1000 milliseconds.

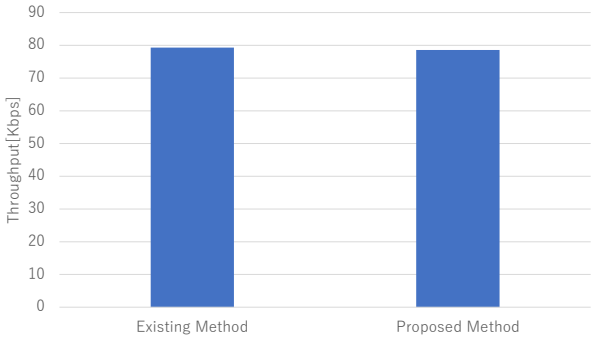

Fig. 10. Average throughput per approach ($h1$ to $h20$)

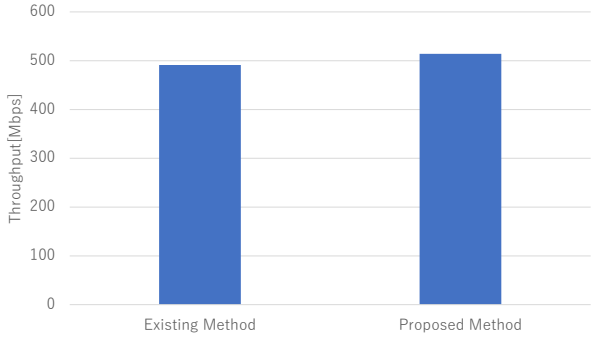

Fig. 11. Average throughput per approach ($h6$ to $h20$)

Regarding the throughput, as observed in Figure 10 and Figure 11, we measured the throughput from $h1$ and $h6$ to $h20$ respectively for both approaches. We found out that there is a big difference between these two cases. In the first case ($h1$-$h20$), which goes through the IPS, as shown in Figure 10, the throughput was below 80 Kbps with either of the controllers.

On the other hand, when the traffic did not go over the IPS ($h6$-$h20$); as shown in Figure 11 the throughput between $h6$ and $h20$ using either method was around 500 Mbps.

We are currently investigating the reasons and possible measures to solve this issue. However, we believe that this difference might be due to the limitations of running Mininet in conjunction with inline Snort in the same VM. For example, if $h1$ uses SCP command or FTP command to $h20$, $h1$ may not complete its tasks such as copying files under this throughput (About 80 Kbps). Therefore, it is difficult to test with more realistic environments. Nevertheless, it is expected that by using the proposed approach the performance will not be severely affected.

V. Conclusion

In this paper, we designed and implemented an MTD mechanism that is transparent to IPS by forwarding packets based on *rIP* between switches in SDN-environments. Using the proposed approach along with an IPS, through extensive experimentation, we showed that it is possible to preventing low-rate scans and high-rate scans.

Moreover, preliminary results show that the proposal is also effective to apply countermeasures for various rates of scan, by deploying the MTD as a low-speed scan in combination with the IPS to monitor/block high-speed scans. Finally, regarding the delay and throughput of the transmissions, the results show the throughput is greatly affected by the presence of an IPS in the experimental environment.

As a future work, we plan to implement all the remaining proposed MTD functions, since only one of the functions was implemented for this article as a proof-of-concept (i.e., address change function). Also, we plan to perform a more detailed evaluation of the MTD system. Specifically, we will evaluate in more detail the impact on the application layers and the effect as a scan countermeasure. Finally, we also plan to conduct performance tests in real environments to assess its effectiveness and behaviour in an actual deployment.

References

[1] A. Alshamrani, S. Myneni, A. Chowdhary, and D. Huang, "A Survey on Advanced Persistent Threats: Techniques, Solutions, Challenges, and Research Opportunities," IEEE Communications Surveys and Tutorials, vol. 21, no. 2, pp. 1851–1877, 2019, doi: 10.1109/COMST.2019.2891891.

[2] M. Akbanov, V. G. Vassilakis, and M. D. Logothetis, "WannaCry ransomware: Analysis of infection, persistence, recovery prevention and propagation mechanisms," Journal of Telecommunications and Information Technology, no. 1, pp. 113–124, 2019, doi: 10.26636/jtit.2019.130218.

[3] Snort - network intrusion detection & prevention system.https://www.snort.org/.(Accessed on 05/20/2021).

[4] NMAP.org. Timing and performance —nmap network scanning. https://nmap.org/book/man-performance.html.(Accessed on 05/20/2021).

[5] U.S. National Science and Technology Council, "Trustworthy cyberspace: Strategic plan for the federal cybersecurity research and development program," Federal Cybersecurity: Strategy and Implementation for Research and Development, pp. 69–99, 2011.

[6] Software-Defined Networking (SDN) Definition, Open Networking Foundation, https://opennetworking.org/sdn-definition/.(Accessed on 05/28/2021).

[7] J. H. Jafarian, E. Al-Shaer, and Q. Duan, "OpenFlow random host mutation: Transparent moving target defense using software defined networking" Proceedings of the 1st ACM International Workshop on Hot Topics in Software Defined Networks, pp. 127–132, 2012, doi: 10.1145/2342441.2342467.

[8] Y. Shi, H. Zhang, J. Wang, F. Xiao, J. Huang, D. Zha, H. Hu, F. Yan and B. Zhao, "CHAOS: An SDN-Based Moving Target Defense System," Security and Communication Networks, vol. 2017, 2017, doi: 10.1155/2017/3659167.

[9] B. Van Leeuwen, W. M. S. Stout, and V. Urias, "Operational cost of deploying Moving Target Defenses defensive work factors," Proceedings - IEEE Military Communications Conference MILCOM, vol. 2015-Dec, pp. 966–971, 2015, doi: 10.1109/MILCOM.2015.7357570.85

[10] Ryu SDN framework.https://ryu-sdn.org/.(Accessed on 05/21/2021).

[11] Mininet: An instant virtual network on your laptop (or other pc) - mininet. http://mininet.org/. (Accessed on 05/21/2021).