MTG: A High-Speed Malicious Traffic Generating Method and Implementation for Future Networks

Yiwen Le* and Jinghan He
School of Electrical Engineering, Beijing Jiaotong University, Beijing 100044, P. R. China
*Corresponding author

***Abstract*—Software-Defined Networking (SDN) has been one of the most promising candidates for future Internet architectures, and it may be one of the best candidates for the future communication system of Smart Grid. Moreover, SDN facilitates a variety of technology innovations in network security area. In this paper, a high-speed malicious traffic generating method called as MTG is proposed, to test the security of future network architectures. MTG can assemble packets and send them at a very high speed up to 10Gbps using hardware-based methods, and shape the sent traffic to follow different patterns with the help of a dynamically interval-between-packets adjusting mechanism. Moreover, the performance of MTG is evaluated. To the best of our knowledge, this is the first attempt to design a hardware-based high-speed malicious traffic generating method for SDN.**

Keywords-traffic generating method; high-speed hardware design; DDoS;SDN;smart grid

I. INTRODUCTION

With the rapid growth of networking technologies, the Internet achieves great success, where people all over the world can connect and communicate with each other and access almost everything they need in any time. However, traditional IP networks, which are the key support technologies for the Internet embedding various interdiction mechanisms such as Access Control List and packet filters, are complex and hard to manage efficiently. For example, to configure a certain high-level service police, network operators have to translate it to corresponding low-level complicated configuration commands, and then use them to configure every involved network device one after another, which consumes a lot of time and effort [1].

To solve the aforesaid issue, future networking technologies such as Software-Defined Networks (SDN) are emerging recent years. In fact, SDN has been a promising networking candidate for future Internet architecture and achieves attentions both from academia and industry [2]. SDN is a clean-slate approach to simplify policy configuration and management by decoupling the control plane of the network (that makes decision on how to handle packets) from the data plane (that includes the underlying routers and switches that forward packets following the decisions from the control plane), which makes all the control tasks of a network be embedded into a logically but not a physically centralized system, while all the routers or switches become just simple forwarding devices [3]. All these advantages benefit the development of future networking architecture such as Smart Grid [4].

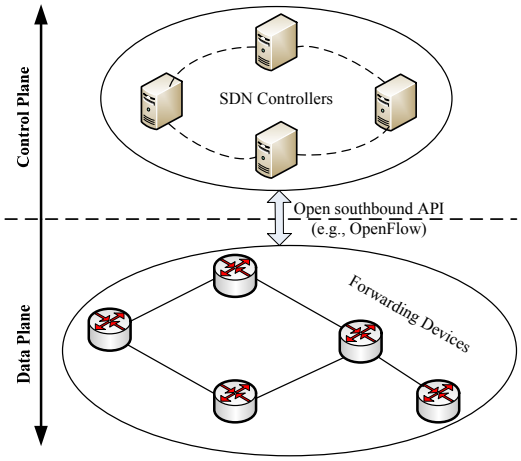

FIGURE I. A DIAGRAM OF SDN ARCHITECTURE

As shown in Figure I, the communications between the control plane and the data plane of SDN can be realized by the open southbound application programming interface (API) using special protocols such as the notable OpenFlow [5]. Usually, The controllers in SDN generate control polices and then send them to forwarding devices such as routers and switches through the southbound API to exercise direct control, while the involved forwarding devices perform corresponding actions (e.g., forwarding packets, modifying content of packets, dropping packets, sending certain packets back to controllers) according to these control polices instructed by the controllers.

Although SDN breaks the vertical integration between the control and data plane and thus facilitates police management for Internet, it indeed involves some risks for network attacks [6]. These attacks can bring in huge damage on financial institutions, government units, energy facilities and even a nation's wide infrastructure, which make them become one of the top level concerning issues in the context of SDN. In fact, a few recent papers have studied the security issues of SDN, such as how to enforce the policy management and how to detect and mitigate malicious traffic [7-19]. However, the research on a real traffic generator that can send a large volume of malicious traffic with different patterns at very high speed (e.g., 10Gbps) is still missing. Indeed, malicious traffic such as high-speed distributed denial-of-service (DDoS) attacks and scanning activities can bring in more potential damage on SDN than traditional networks, because they can disrupt network

operations by saturating the control place of SDN with large volume of malicious traffic [7].

This paper focuses on how to design a realistic malicious traffic generating approach, in order to test the security of SDN with real-world network traffic (not only simulations). The main contributions of this paper are summarized as follows:

1) A high-speed malicious traffic generating method called MTG is designed, which includes a hardware-based packets assembling mechanism to achieve high speed packet assembling, and a dynamically interval-between-packets adjusting mechanism to shape the sent packets following certain traffic patterns.

MTG is designed to perform stress tests for future networking technologies, especially for SDN. This is because SDN bring in a new network entity called controller, which makes decision on how to forward packets. SDN routers or switches will send messages to ask controllers for packet-forwarding guidance, whenever the routing information for the destination IP address of a packet cannot be found at the forwarding tables. In this case, SDN brings in new risks for attackers that they can easily send plenty of malicious packets with fake destination IP addresses that are not reachable, triggering large number of messages from SDN routers/switches to controllers to saturate its resource. Therefore, the proposed MTG is very useful for testing SDN-based protocols or SDN devices to prevent such threat, by generating plenty of different types of malicious packets at a very high speed.

2) The performance of MTG is evaluated through extensive demo experimentations, and results show that MTG can not only succeed in sending high-speed traffic with certain content at 10Gbps, but also generate different traffic patterns to simulate different kinds of network attacks.

The rest of this paper is organized as following. Section 2 introduces some closely related work. Section 3 presents the MTG. The evaluation results of the proposed MTG are given in section 4. Finally, section 5 concludes this paper.

The rest of this paper is organized as following. Section 2 introduces some closely related work. Section 3 presents the MTG. The evaluation results of the proposed MTG are given in section 4. Finally, section 5 concludes this paper.

II. RELATED WORK

There are already a diverse set of security studies emerging in SDN research area [8-20]. Most take advantage of the individual characteristic of SDN that its control place is decoupled from its data plane to enhance and improve the security of SDN-enabled networks, mainly focusing on two aspects: policy enforcement (e.g., source address validation, access control list) and malicious traffic countermeasure.

To improve the policy enforcement in SDN-based networks, the work in [8] introduces the "middlepipes" that can facilitate application developers or network administrators on new network function abstraction or new management polices implementing, without caring about the low lever details such as packets vs. requests. In [9], the authors propose the "SANE", which embeds a protection layer to govern all connectivities and make routing and access control decisions, to enhance the security of networks and simplify the management for enterprise networks that are filled with complex routing and polices. The authors in [10] design the "LiveSec" employing the access-switching layer to achieve fine-grain and effective security management. The work in [11] proposes the "VAVE" to perform the source address validation operation with the help of the OpenFlow protocol, which brings in a new method to enhance the policy enforcement in SDN. The work in [12] proposes a flow-based network access control solution called "FlowNac", to grant users who can get the right to access fine-grained certain services that are univocally defined as a set of flows. The work in [13] proposes the "CloudWatcher" which can provide monitoring services by writing a simple policy script in SDN. In addition, the authors in [14] focus on eliminating security problems of using multiple controllers to degrade the threat from malicious administrators.

To detect and mitigate malicious or anomaly traffic in SDN, the work in [15] takes advantage of the programmatic interface of SDN architecture to facilitate the handling of switch information, including both normal traffic and useless/malicious traffic. The authors in [16] use the programming ability of SDN to mitigate DDoS attacks on a per-flow level and then protect normal operation of the victim. Using SDN technologies, the work in [17] develops a proactive moving target defense architecture that mutates hosts' IP addresses unpredictably and thus hides hosts' identifiers from potential attackers. The authors in [18] focus on the security of home networks, and implement several existing prominent anomaly detection algorithms into SDN context to identify malicious activities. The authors in [19] propose the "FlowDiff", which detects malicious behavior by analyzing the control traffic between SND controllers and switches through passive measurements. Additionally, the work in [20] introduces "NetFuse" using passive control messages to protect SDN from misbehaving traffic both originated from external and internal networks.

All the work described above are making effort to enhance the security of SDN, however, none of them attempts to design high-speed malicious traffic generating mechanisms to test SDN's security in real-world environment (not only in simulations). Therefore, MTG is proposed to fill the blank in this area of SDN research, in order to accelerate the development and implementation of SDN devices and their corresponding technologies.

III. MTG DESIGN

To achieve a real malicious traffic generating scheme with a very high attacking rate (e.g., 10Gbps) for SDN security testing, the MTG embeds two key technologies: dynamically interval-between-packets adjusting mechanism (DIA) and hardware-based packets assembling mechanism (HPA).

A. DIA Design

Define abbreviations and acronyms the first time they are used in the text, even after they have been defined in the abstract. Abbreviations such as IEEE, SI, MKS, CGS, sc, dc,

and rms do not have to be defined. Do not use abbreviations in the title or heads unless they are unavoidable.

DIA generates malicious traffic with different attacking rates by sequencing data packets in different time intervals with a very fine granularity (6.44 nanosecond), which grantees the proposed MTG can achieve to sending different traffic rates and distributions to flood certain network architectures and their infrastructures (e.g., SDN and its SDN controllers or SDN switches).

Specially, DIA transforms the different time intervals to different data bytes intervals between different packets (called as gap-size in this paper) to achieve simple but accurate traffic-sending-rate adjusting. Specifically, DIA adaptively shapes the number of gap-sizes in each CPU clock of 155.25MHz according to Eq. (1), and then sequences packets to achieve different traffic-sending-rates according to Eq. (2).

$$x = a * c \tag{1}$$

In Eq. (1), a represents the line-rate of the link that is connected with some SDN switch, c represents the time instant, and x represents the gap-sizes.

$$z = \frac{a/(b + y + x)}{a/(b + y + x_{\min})} \tag{2}$$

In Eq. (2), x_{min} represents the minimum of gap-sizes, b represents the size of the preamble of each packet, y represents the size of each packet, and z represents the ratio of the real traffic rate generated by DIA and the line-rate of the link.

Eq. (2) can be translated to Eq. (3),

$$z = \frac{b + y + x_{\min}}{b + y + x} \tag{3}$$

In the DIA design, the preamble of each sent packet is fixed with 8 bytes, the minimum size of each Ethernet frame is 64 bytes and the minimum of the gap-size is 12 bytes. That is, $b = 8$, $y \geq 64$ and $x \geq x_{min} = 12$.

Thus, Eq. (3) can be further translated to Eq. (4),

$$z = \frac{20 + y}{8 + y + x}, \quad y \geq 64 \tag{4}$$

B. HPA Design

In HPA, every consumptive operation of each packet of the sent traffic generated by MTG (e.g., editing content, injecting error or malicious fragment, CRC computing) is embedded into hardware-based scheme by implementing very high speed FPGA functional modules that have the ability of parallel processing, in order to solve the high-speed packet assembling problem. HPA can assemble about 16,500 packets per second. That is, for a sent packet with minimum size of 64 bytes, the rate of HPA generating packets in real time is about $16{,}500 \times (64+12) \times 8 \approx 10$Gbps. Additionally, software-based scheme only takes in charge of the formatting of each packet, but not actually assembling them. Specially, the procedure of HPA can be described as following:

Step 1: the malicious packets which are ready to send are formatted (e.g., setting IP source addresses, filling the payload) and then cached in packet-sending buffers through software-based scheme; noting that malicious packets can bring in damage to SDN not only by injecting error fragments but also by with normal content to form flooding traffic at a very large rate;

Step 2: all the sending parameters for each sending-ready packet are cached in parameter-control buffers, to shape the traffic patterns that simulate different types of DDoS traffic (e.g., burst traffic, continuous traffic, traffic with different packet intervals, varying IP addresses, forging MAC addresses);

Step 3: According the format and parameters defined in Step 1 and Step 2, The real packets are assembled by a hardware-based parallel processing scheme, which gets the payload and prefix information from the packet-sending buffers defined in Step 1 and simultaneously gets the parameters information as well as the corresponding computing operations (e.g., injecting error fragments into the payload of some certain packets with a predefined IP source address) from the parameter-control buffers defined in Step 2; and then sending packets follows the gap-sizes cached in parameter-control buffers to generate malicious traffic with different rates and different types.

IV. PERFORMANCE EVALUATION

A. Effect of Gap-size Setting

In the proposed **MTG**, the setting of gap-size is significant for the performance of packet sending.

Figure II illustrates how different gap-sizes affects the ratio of the sending traffic rate and the line-rate of the link connected to a SDN forwarding device, with different packet sizes inside the sending traffic.

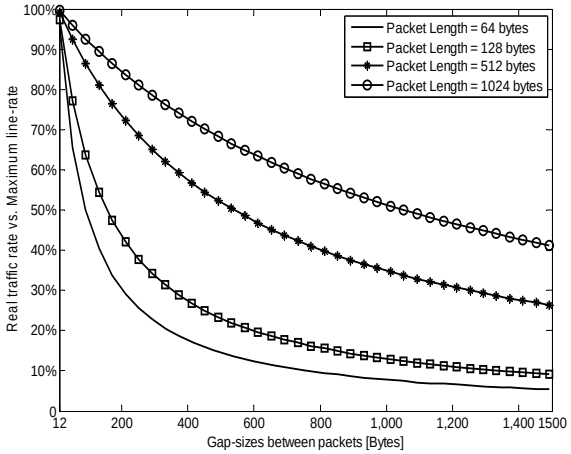

FIGURE II. DIA-GENERATED TRAFFIC RATES VS. GAP-SIZES

From Figure I, the real sent traffic rate generated by DIA is decreased with a bigger gap-size between packets, when the packets are with the same length (e.g., packet length is 64 bytes). Moreover, for the same gap-size setting in DIA, if every of the sent packets has a bigger length (e.g., 512 bytes *vs.* 128 bytes), the real sent traffic rate become higher. For the traffic where every packets inside is with the same length, DIA can achieve to shape the traffic rate from smaller than 10% to almost 100% of the maximum line-rate of a link, by adjusting the gap-size between packets.

B. Effect of Packet Length Setting

In the proposed **MTG**, setting different lengths for sending packets can affects the total rate of the out traffic.

Figure III illustrates how different packet lengths affects the ratio of the sending traffic rate and the maximum line-rate of the link connected to a forwarding device in future network, with different Gap-sizes between every two sending packets inside the sending traffic.

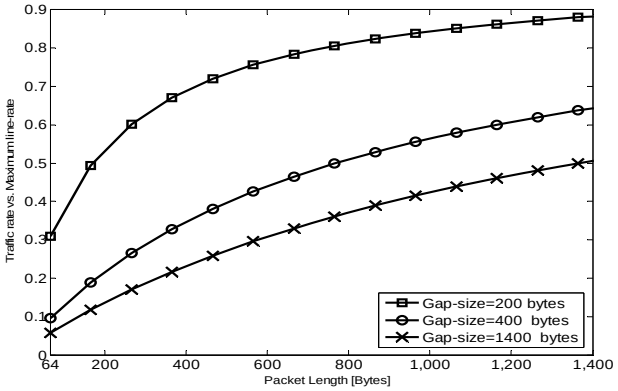

FIGURE III. DIA-GENERATED TRAFFIC RATES VS. PACKET LENGTHS

From Figure III, the real sent traffic rate generated by DIA is decreased with a bigger packet length, when the gap-size between packets is with the same length (e.g., packet gap-size is 200 bytes). Moreover, for the same packet length setting in DIA, if the gap-size between every two packets has a bigger length (e.g., 1400 bytes *vs.* 400 bytes), the real sent traffic rate become smaller. That is, with the same gap-size setting, a small packet length is better to achieve a higher rate of the sending traffic.

V. CONCLUSION

In this paper, MTG is introduced, which can assemble and send a large volume of malicious traffic at 10Gbps and with different patterns (e.g., burst traffic, continuous traffic, traffic with different packet intervals, varying IP addresses, forging MAC addresses) to damage future networks such as SDN and Smart Grid. For example, MTG can be used to test the security of SDN protocols as well as SDN controllers and forwarding devices, and can accelerate the development of SDN security technologies.

In future work, the proposed MTG will be used to test some commercial SDN controllers, routers and switches (e.g., SDN controllers from Cisco or Huawei). Moreover, SDN controller-based distributed DDoS countermeasures will be exploited, to protect SDN from high-speed malicious traffic. In addition, how to test the ability of Smart Grid when it is suffering high communication traffic jam is one of our concerns in future.

REFERENCES

[1] B. Raghavan *et al.*, "Software-defined internet architecture: Decoupling architecture from infrastructure," In *Proc.* 11*th ACM Workshop Hot Topics Netw.*, 2012, pp. 43-48.

[2] D. Kreutz *et al.*, "Software-defined networking: A comprehensive survey," *Proceedings of the IEEE*, vol. 103, no. 1, pp. 14-76, Jan. 2015.

[3] H. Kim *et al.*, "Improving network management with software defined networking," *IEEE Communications Magazine*, vol. 51, no. 2, pp. 114-119, Feb. 2013.

[4] X. He, Q. Ai, R. C. Qiu, et al., A Big Data Architecture Design for Smart Grids Based on Random Matrix Theory, IEEE Transactions on Smart Grid, vol.pp, no.99, pp.1-13, 2015.

[5] N. McKeown *et al.*, "OpenFlow: Enabling innovation in campus networks," *SIGCOMM Comput. Commun, Rev.*, vol. 38, no. 2, pp. 69-74, Mar. 2008.

[6] D. Kreutz *et al.*, "Towards secure and dependable software-defined networks," In *Proc.* 2*nd ACM SIGCOMM Workshop Hot Topics Softw Defined Netw.*, 2013, pp. 55-60.

[7] S. Shin *et al.*, "AVANT-GUARD: Scalable and vigilant switch flow management in software-defined networks," In *Proc. ACM Conf. Comput. Commun. Security (CCS)*, 2013, pp. 413-424.

[8] H. Jamjoom *et al.*, "Don't call them middle-boxes, call them middlepipes," In *Proc.* 3*rd Hot Topics Softw. Defined Netw.*, 2014, pp. 19-24.

[9] M. Casado *et al.*, "SANE: A protection architecture for enterprise networks," In *Proc.* 15*th conf. USENIX Security Symp.*, 2006, vol. 15, Article 10.

[10] K. Wang *et al.*, "LiveSec: Towards effective security management in large-scale production networks," In *Proc.* 32*th Int. Conf. Distrib. Comput. Syst (ICDCS). Workshops*, 2012, pp. 451-460.

[11] G. Yao *et al.*, "Source address validation solution with OpenFlow/NOX architecture," In *Proc.* 19*th IEEE Int. Conf. Netw. Protocols (ICNP)*, 2011, pp. 7-12.

[12] R. Raghavendra *et al.*, "Dynamic graph query primitives for SDN-based cloudnetwork management," In *Proc.* 1*st Workshop Hot Topics Softw. Defined Netw.*, 2012, pp. 97-102.

[13] S. Shin *et al.*, "CloudWatcher: Network security monitoring using OpenFlow in dynamic cloud networks (or: How to provide security monitoring as a service in clouds?)," In *Proc.* 20*th IEEE int. Conf. Netw. Protocols (ICNP)*, 2012, pp. 1-6.

[14] S. Matsumoto *et al.*, "Fleet: defending SDNs from malicious administrators," In *Proc.* 3*rd Workshop Hot Topics Softw. Defined Netw.*, 2014, pp. 103-108.

[15] R. Braga *et al.*, "Lightweight DDoS flooding attack detection using NOX/OpenFlow," In *Proc.* 35*th IEEE Conf. Local Comput. Netw. (LCN)*, Oct. 2010, pp. 408-415.

[16] K. Giotis *et al.*, "Leveraging SDN for efficient anomaly detection and mitigation on legacy networks," In *Proc.* 3*rd Eur. Workshop Softw. Defined Netw.*, 2014, pp. 85-90.

[17] J. H. Jafarian *et al.*, "OpenFlow random host mutation: Transparent moving target defense using software defined networks," In *Proc.* 1*st Workshop Hot Topics Softw. Defined Netw.*, 2012, pp. 127-132.

[18] S. A. Mehdi *et al.*, "Revisiting traffic anomaly detection using software defined networking," In *Proc.* 14*th Int. Conf. Recent Adv. Intrusion Detection*, 2011, pp. 161-180.

[19] A. Arefin *et al.*, "Diagnosing data center behavior flow by flow," In *Proc. IEEE* 33*rd Int. Conf. Distrib. Comput. Syst. (ICDCS)*, 2013, pp. 11-20.

[20] Y. Wang *et al.*, "NetFuse: Short-circuiting traffic surges in the cloud," In *Proc. IEEE Int. Conf. Commun. Security (ICC)*, 2013, pp. 3514-3518.