Telephony Denial of Service Defense at Data Plane (TDoSD@DP)

Aldo Febro
School of Computer Science
University of Hertfordshire
Hatfield, UK AL10 9AB
Email: a.febro@herts.ac.uk

Hannan Xiao
School of Computer Science
University of Hertfordshire
Hatfield, UK AL10 9AB
Email: h.xiao@herts.ac.uk

Joseph Spring
School of Computer Science
University of Hertfordshire
Hatfield, UK AL10 9AB
Email: j.spring@herts.ac.uk

***Abstract*—The Session Initiation Protocol (SIP) is an application-layer control protocol used to establish and terminate calls that are deployed globally. A flood of SIP INVITE packets sent by an attacker causes a Telephony Denial of Service (TDoS) incident, during which legitimate users are unable to use telephony services. Legacy TDoS defense is typically implemented as network appliances and not sufficiently deployed to enable early detection. To make TDoS defense more widely deployed and yet affordable, this paper presents TDoSD@DP where TDoS detection and mitigation is programmed at the data plane so that it can be enabled on every switch port and therefore serves as distributed SIP sensors. With this approach, the damage is isolated at a particular switch and bandwidth saved by not sending attack packets further upstream. Experiments have been performed to track the SIP state machine and to limit the number of active SIP session per port. The results show that TDoSD@DP was able to detect and mitigate ongoing INVITE flood attack, protecting the SIP server, and limiting the damage to a local switch. Bringing the TDoS defense function to the data plane provides a novel data plane application that operates at the SIP protocol and a novel approach for TDoS defense implementation.**

***Index Terms*—SIP, DoS, DDoS, SDN, P4, data plane**

I. Introduction

Telephony Denial of Service (TDoS) attack has the potential to disrupt telecommunication services denying legitimate users access to the service. One common method of launching a TDoS attack is by sending a flood of SIP INVITE packets to a SIP server. This attack overwhelms the system and causes a service disruption for legitimate users.

We review previous proposals in the literature from two perspectives: the attack detection method and the defense location. From the perspective of an attack detection method, previous proposals can be broadly categorized into three types: rule-based, anomaly-based and state machine-based. With rule-based detection method, specific rules are followed that are used to determine whether to make a pass or drop decision. This rule can either be static or dynamic. For example, a rule or signature can be constructed to raise the alarm when certain characteristics are met [1] [2]. For anomaly-based detection methods, a mathematical or statistical model is built that describes proper behavior for a SIP session, and when anomalies or significant deviations are detected, it raises the alarm [4]. With state machine based detection methods, the state machine for the SIP protocol is tracked and deviation from the expected state transition will raise the alarm [5] [6].

From the defense location perspective, previous proposals have been implemented either as a network-based IDS, a host-based IDS, an extension module for the SIP server software, or hybrid combination of these. With network-based IDS, a dedicated server or network appliance is installed in the network path between trusted and untrusted areas, either as a transparent layer 2 device (as a bump-in-the-wire) or as a layer 3 device. Being on the circuit, the appliance will be able to see all packets that pass through the device [1] [2]. With host-based IDS, this capability is implemented as detection and mitigation software installed on the targeted server [4]. With the extension module, this capability is implemented as an extension module of the SIP Proxy software [5] [6]. Commercial DDoS defense solution is offered as a hybrid solution between cloud-based scrubbing center in the provider's data center and DDoS appliance in the subscriber's data center.

A. Motivation and use cases

As we enter the era of IoT and edge computing, distributed devices produce a high volume of streaming data which requires substantial bandwidth and computing resources to perform attack detection and mitigation functions in the data center. Due to the cost and practical constraints, the IDS/IPS sensors are not sufficiently deployed for early detection. The first motivation of this work is to enable TDoS defense as widely as possible before TDoS packets had the chance to accumulate and become disruptive to SIP service. Edge-oriented defense approach is also proposed by [3], but this paper is focused solely on SIP packets.

The second motivation is to address recent security trends that expose the weaknesses of existing approach. These include for example, growing DDoS attack size (as demonstrated by Mirai botnet attack at 1.1 Tbps [7]), Low-and-slow application-layer DDoS attack [8], mobile botnets that consists of infected mobile phones [9], DDoS attacks that were launched from mobile phones against 911 emergency services [10], the use of SIP for botnets communication [11], etc.

Given these new concerns and trends, defense methods from previous proposals are still required, but it needs to

978-1-5386-3416-5/18/$31.00 © 2018IEEE

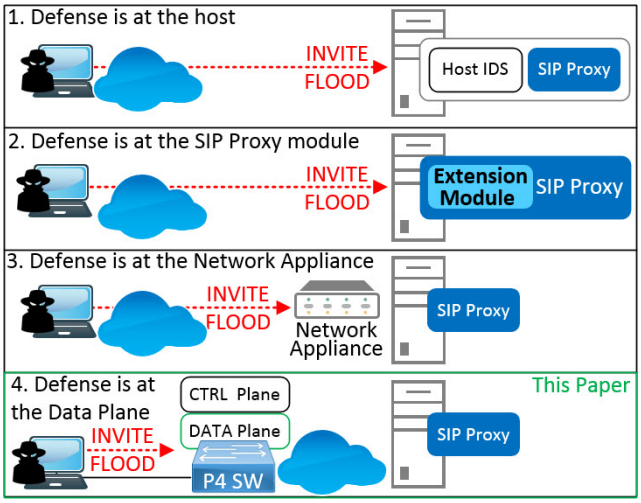

Fig. 1. Locations where typical defense is implemented (1-3) vs. this paper (4)

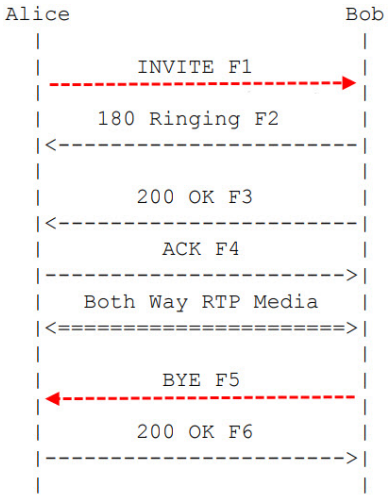

Fig. 2. Typical SIP session establishment (RFC3665/BCP75)

be implemented and deployed in more locations. Instead of relying on legacy network appliances installed at data centers, these recent security trends point to the need for detection and mitigation capabilities to be deployed as widely as possible, and placed as near as possible to the attacker. For practical reasons, this approach should be cost-effective, support incremental and non-contiguous deployment. As such, it is not meant to replace existing defense systems, rather it is designed to augment or complement.

B. Contribution

The contributions of this paper are as follows:

- *Novel data plane application that operates at SIP protocol layer*. To the best of our knowledge, data plane applications have never been applied to the DoS vulnerability of the SIP protocol.
- *Novel implementation of state-based SIP INVITE flood attack detection*. To the best of our knowledge, SIP INVITE flood attack detection have never been implemented at the data plane.
- *Novel location for SIP INVITE flood attack detection and mitigation*. To the best of our knowledge, SIP TDoS attack detection and mitigation have never been implemented at every port on a network switch.

The rest of the paper is organized as below: the proposed solution is outlined in section II, followed by section III that describes the experiments. The result from experiments are presented in section IV-A, discussion in section IV-B, followed by current limitation in section IV-C. Finally, conclusion remarks are drawn in section V.

II. The Proposed Solution

A. Data plane application for TDoS defense

SDN offers a new paradigm for networking with programmable control and data plane. This level of programmability offers a new set of capabilities that were not previously available to SIP security researchers. Considering that telephony is a time-sensitive application, it is necessary to minimize delay when processing real-time packets. With that requirement in mind, we propose to implement TDoS detection and mitigation capability in the data plane directly. When the data plane is programmable, it is now possible to perform deep packet inspection at the SIP protocol layer for every single packet. This function is necessary to verify whether or not a particular SIP session deviates from normal SIP state machine protocol. In essence, we propose to implement the functions of a traditional network appliance in the data plane and to make it available at every switch port. The contrast between previous proposals and this paper is illustrated in Figure 1.

Each rectangles in Figure 1 depicts locations where TDoS detection and mitigation are commonly implemented, i.e., as an application installed on the same host as the SIP Proxy Server, as an extension module of the SIP Proxy Server, and as a dedicated network appliance, respectively. In contrast, this proposal implements TDoS defense at the data plane as depicted in Figure 1 (bottom rectangle). In this approach, SIP TDoS defense can be made available on each port on P4 switch.

B. SIP state machine detection algorithm

The SIP protocol uses a state machine to create and terminate a SIP session. A valid and proper SIP session will consist of a pair of INVITE-and-BYE packet, whereas for a Telephony DoS attack it will have high number of INVITE packets. As per RFC 3665 or Best Current Practice 75 [12], successful session establishment looks like Figure 2 where the caller initiated the session by sending INVITE packet and the callee ended the session by sending BYE packet.

As depicted with pseudo code in Figure 3, the attack detection algorithm states that for every switch port, it cannot exceed active session limit that has been pre-set for that port. A SIP session is considered active when a port has received an INVITE packet and BYE packet has not been sent out from the

```
if (INGRESS) then
  if (packet="INVITE")and(ingressPORT="X") then
    inviteReg[X]++
  end if
  if (inviteReg[X] > inviteLimitReg[X]) then
    dropPacket
  else
    forwardPacket
  end if
end if

if (EGRESS) then
  if (packet="BYE")and(egressPORT="X") then
    byeReg[X]++
  end if
  if (inviteReg[X] => inviteLimitReg[X])and
     (byeReg[X]=> byeLimitReg[X])then
        inviteReg[X] = 0
        byeReg[X] = 0
  end if
end if
```

Fig. 3. Pseudo code for detection & mitigation

TABLE I
CUSTOMIZABLE LIMIT FOR EACH PORT

Connected device	Port#	INVITE Limit
IP phone	1	1
IoT device	2	0
Computer	3	1
–	–	–
Trunk port	24	1000

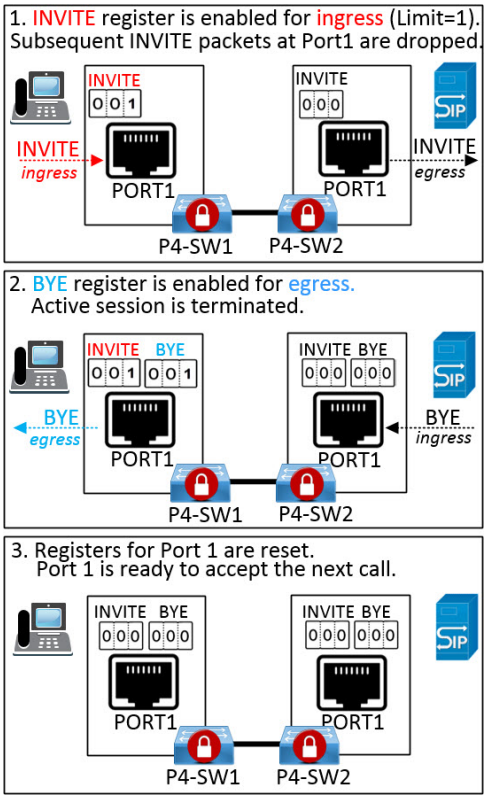

Fig. 4. SIP INVITE & BYE counters at every port. Counters are reset after completion of a valid session.

same port. In other words, the algorithm can imposes a limit to the outstanding INVITE packets. When this limit is reached, subsequent INVITE packets will be automatically dropped and a SIP session cannot be created.

During TDoS attack, the attacker keeps sending INVITE packets without following interactions as specified in the protocol specification RFC 3261 [13]. As such there will be a lot of INVITE packets and this condition is interpreted as an ongoing INVITE flood attack. In this situation, these INVITE packets are automatically dropped at the data plane.

Tracking the SIP state-machine is effective because it still works even when the source IP address is spoofed and randomized. To contrast with signature/rule-based detection method, it is hard to build effective detection rules when the source IP address is random. With the anomaly-based method, the low-and-slow attack does not trigger the alarm because it does not exhibit a different traffic pattern (e.g., sudden burst of traffic) and therefore appears as normal traffic to the anomaly-based algorithm. In contrast with the anomaly-based method, the state-machine detection method still works regardless of how slow or how fast these malicious SIP sessions were created.

C. Custom INVITE limit for each port

On a switch, each port has its own INVITE Limit depending on which device is connected to that port and the potential threat that this device is presenting (Table I). For example, the limit can be set to zero for an IoT device that is not expected to have SIP communication. This is useful to prevent an IoT device that has been infected and recruited by a botnets to use SIP as a command and control (C2) channel [11]. If the port is authorized for SIP sessions, the limit for this port can be set to one or two depending on the requirement. If this port is connected to an upstream SIP provider, then its Outstanding INVITE Limit can have a larger value.

When an INVITE packet was received by port 1, the value of INVITE register at port 1 was increased by 1 as depicted in Figure 4 step (1). Subsequent INVITE packets that were received by port 1 were automatically dropped since the Outstanding INVITE Limit for port 1 (connected to an IP phone) was pre-set to 1 (Table I). Non SIP packets were processed as normal, i.e., were routed towards the destination. When a BYE packet was about to be sent out of port 1, the value of the BYE register at port 1 was increased by 1 as depicted in Figure 4 step (2). At this time both registers (INVITE and BYE) had the value of one that signifies a completed SIP session. These registers were reset to 0 as in Figure 4 step (3) at the end of a complete session.

III. EXPERIMENTS

A. Three scenarios

To contrast the different outcome between a legacy switch and TDoSD@DP under TDoS attack situation, three scenarios were tested as depicted in Figure 5.

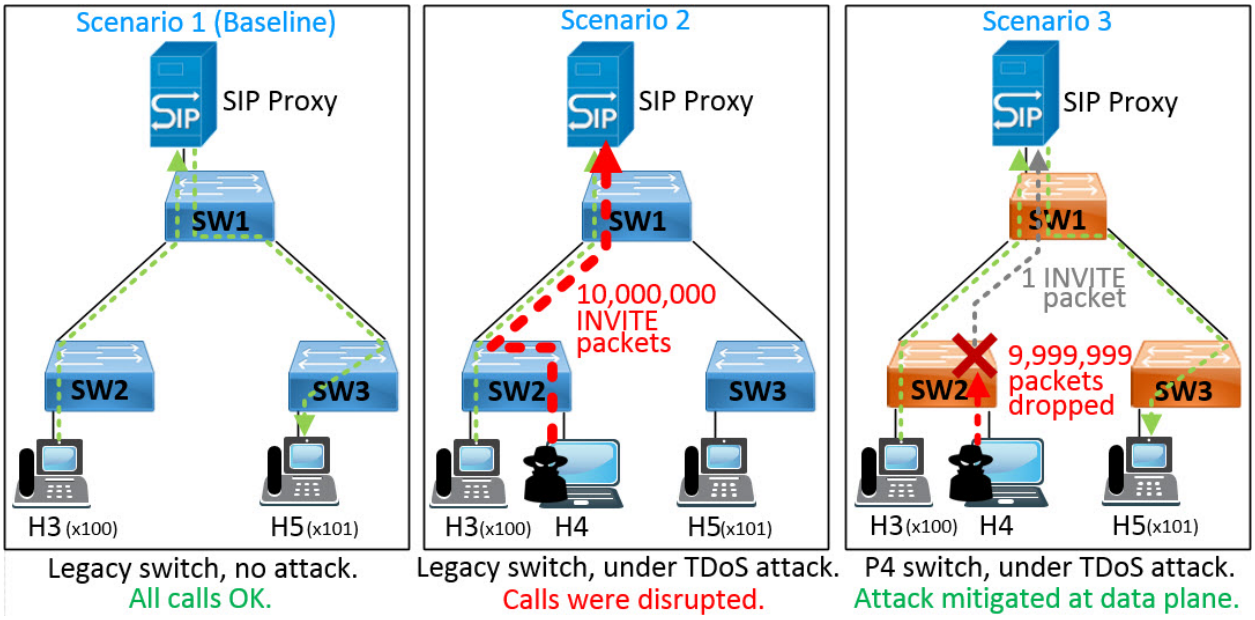

Fig. 5. Comparing the outcome between legacy vs. P4 switch, with and without TDoS attack.

- *Scenario 1: Legacy switch, no attack*. This scenario is taken as the baseline for comparison. H3 (extension 100) is calling H5 (extension 101) once per second for three consecutive minutes (a total of 180 total calls).
- *Scenario 2: Legacy switch, under TDoS attack*. TDoS attack was introduced after 60 seconds into the session. An attacker at switch 2 sent 10 million SIP INVITE packets to the SIP Proxy (which is more than enough to cause process stack overflow, i.e., 110 packets/second [14]). This scenario shows the impact of TDoS attack on the victim (SIP Proxy) and on the switch that is directly connected to the attacker.
- *Scenario 3: TDoSD@DP, under TDoS attack*. The same attack scenario as Scenario 2, but legacy switches were replaced by P4 switches that run TDoSD@DP data plane application. This scenario shows how TDoSD@DP solves the problem presented in Scenario 2.

B. Test environment

The experiment was conducted using mininet to build topology in Figure.5 to emulate a typical SIP environment. This emulated environment consists of 3 switches, 1 SIP Proxy server, 2 SIP phones (H3 and H5), and an attacker (H4). The SIP Proxy server was connected to switch 1, the caller (extension 100) and the attacker was connected to switch 2, while the callee (extension 101) was connected to switch 3. Two kinds of switches were used for comparison purposes (Open vSwitch to represent the legacy switch and bmv2 to run TDoSD@DP data plane application). For the SIP Proxy server, a popular open source Asterisk software was used. For SIP phones (extension 100 and 101), an open source SIP traffic generator called SIPp was used. For the attacker, a real TDoS tool called inviteflood [15] was used.

C. P4 program implementation

The data plane application for the SIP TDoS defense was written with the P4 $_{14}$ programming language [16] and ran on the bmv2 switch. A SIP header was defined with one field, i.e. startline, with the size of 64 bits which is sufficient to identify whether it contains an INVITE or a BYE packet. Four registers were defined: INVITE, INVITE_limit, BYE, and BYE_limit. These registers were used to hold the total number of INVITE and BYE packets coming from/to a specific port and their associated limits. Ingress and Egress port number metadata is used to access these registeres. Two tables were used: checkINVITE table that was used during ingress, and checkBYE table was used during egress. The match for checkINVITE table was based on two fields, i.e., ingress port number and SIP startline, whereas the match for checkBYE table was also based on two fields, i.e., egress port number and SIP startline. To get a match for a SIP INVITE packet on the checkINVITE table, it was pre-populated with ingress port number and hex value for SIP INVITE, e.g., `1, 0x0x494e564954452073` (to match SIP INVITE packet arrived at Port 1) and its associated action was to increment the INVITE counter. In order to get a match for SIP BYE packet, the table was pre-populated with egress port number and hex value for BYE e.g. `1, 0x425945207369703a` and its associated default action to increment the BYE counter. With the above functions, the data plane was able to track the number of SIP INVITE packets received by each port, as well as SIP BYE packets sent by each port. For IPv4 routing and switching, longest prefix match on destination IP address and setting destination MAC address was performed, similar to P4 simple router program.

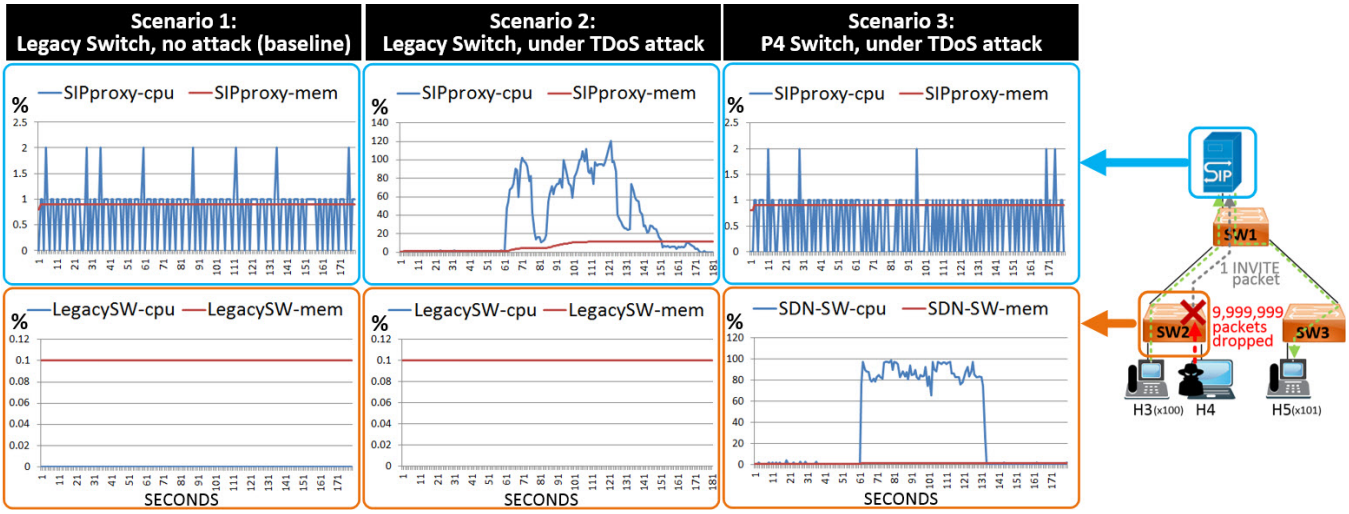

Fig. 6. Comparison of CPU & Memory utilization for SIP Proxy & SW2 between 3 scenarios. Workload is shifted to the attacker's P4 switch (SW2) instead of on the SIP Proxy server.

TABLE II
RESULTS

	Scenario.1 Non-programmable Data Plane. No attack	Scenario.2 Non-programmable Data Plane. With TDoS attack	Scenario.3 Programmable Data Plane. With TDoS attack
Virtual switch software	Open vSwitch	Open vSwitch	bmv2
Attack packets sent by attacker	None	10,000,000	10,000,000
Attack packets received by server	0	10,000,000	1
Attack packets dropped by switch	0	0	9,999,999
Server (victim) error message	None	Unable to create socket	None
Server (victim) max CPU %	2	120.5	2
Server (victim) max MEM %	0.9	11.4	0.9
Switch (defense) max CPU %	0	0	98.6
Switch (defense) max MEM %	0.1	0.1	1.3
Number of calls attempted	180	67	180
Number of successful call	180	63	180

D. Normal call emulation

To emulate a normal call, extension 100 (H3) was calling extension 101 (H5) by sending INVITE packets to the SIP Proxy Server. The SIP Proxy server recognized that the call was meant for extension 101, and created a second SIP session to extension 101. Extension 101 answered the call and then terminated the call by sending a BYE packet towards the SIP Proxy server, which then terminated the call to extension 100. It took less than 1 second to establish and terminate a call.

E. TDoS attack emulation

A TDoS attack was initiated 60 seconds after a normal call was initiated. Using the inviteflood tool, the attacker (H4) sent 10 million SIP INVITE packets to the SIP Proxy server.

IV. Result and Discussions

A. Result

The result from the 3 experiments is given in Table II. A total of 180 calls were made in all experiments. During the TDoS attack, the legacy switch passed all 10,000,000 attack packets to their destination, whereas the TDoSD@DP only let the first INVITE packet to pass through and dropped the remaining packets (9,999,999).

With the legacy switch, the SIP Proxy server received 10 million attack packets and generated an error message saying unable to create sockets. This corresponds to the low number of successful call attempt (63) reported by the caller (H3, extension 100). The SIP Proxy server stopped working as it ran out of internal resources. Whereas with TDoSD@DP, the SIP Proxy server did not generate any error message and was able to keep serving clients as normal. This condition was captured by the number of successful calls reported by H3, which was 180 (1 call per second for 3 minutes).

Next we compare the system utilization of the three scenarios. The top row in Figure 6 depicts CPU and memory utilization for the SIP Proxy server, and the bottom row depicts CPU and memory utilization of the switch that is directly connected to the attacker.

- *Scenario 1: Legacy switch, no attack*. CPU utilization of the SIP Proxy server was hovering around 1 % for most of the time, whereas memory was around 0.9 %.
- *Scenario 2: Legacy switch, under TDoS attack*. CPU

utilization of the SIP Proxy server shot up to 120.5 % (multicores server, which used the second core for additional processing capacity). The memory was hovering around 11.4 %. This was the time when the SIP Proxy server stopped processing calls as it ran out of internal resources to create sockets.

- *Scenario 3: TDoSD@DP, under TDoS attack*. The SIP Proxy did not experience performance hit. It shows similar CPU and memory usage pattern as the baseline. Instead of being processed by the SIP Proxy Server, the processing workload (attack detection and mitigation) has shifted to the TDoSD@DP. At one point, the maximum CPU utilization reached 98.6 %.

B. Discussion

The main difference between the three experiments was the type of switch being used. With the legacy switch, it simply delivered the packets towards the destination and was not even aware that the packets were part of an ongoing TDoS attack. In contrast, TDoSD@DP was able to do deep packet inspection at SIP layer, dropped attack packets and let legitimate packets to pass through. A period of high CPU load on the bmv2 switch is expected due to processing the malicious packets.

In enterprise or service provider environments, dedicated network appliances usually perform deep packet inspection for SIP traffic. This capability is typically available in a centralized scrubbing center or data center. However, it is costly to deploy these appliances widely, and it is inefficient to transport TDoS packets from multiple remote locations to the central data center, where it is eventually dropped.

With TDoSD@DP, SIP flood defense can be performed as soon as the attack packets entered the network and did not have to wait until it reaches the central data center. This approach has the advantages of mitigating the attack closest to the source of the attack. The detection and mitigation algorithm is kept simple considering limited computing resources available on access layer switch.

As for concerns about performance, two considerations might address these. First, since TDoSDD@DP is dealing with SIP DoS attack at the network edge, the size of attack traffic is considerably small and manageable for a modern switch to handle. Second, the underlying target platform (e.g., NIC, NPU, ASIC, etc.) is getting faster. For example, Netronome Agilio NIC can run at 40 Gbps, while Tofino chips can run at 6.5 Tbps.

For the recent attack use cases, TDoSD@DP is capable of dealing with various SIP TDoS attacks. For example, in Distributed TDoS attacks where IoT devices were used to launch SIP DoS attack, the INVITE limit can be set to zero, so IoT device will not be able to send INVITE packet. With the low-and-slow SIP DoS attack, TDoSD@DP limits the total number of SIP session allowed, regardless of how slow or how fast the sessions were created.

C. Limitation

With the current state of bmv2 implementation, TDoS detection and mitigation capability are limited to unencrypted packets. Detection also limited to the first eight bytes of SIP packet as P4_14 and bmv2 does not currently support multiple variable-length fields which is typical for SIP headers. As consequences, it prevents TDoSD@DP to reach granularity at individual SIP branch e.g., CSeq, Call-ID, and remote/local-tag level.

V. Conclusion

This paper proposes Telephony Denial of Service Defense at the Data Plane (TDoSD@DP), a novel data plane application that operates at the SIP protocol layer and a novel implementation of SIP DDoS defense. The experiments showed that TDoSD@DP was able to detect and mitigate ongoing SIP INVITE flood attack. TDoSD@DP provides network operators with granular control of SIP INVITE at every switch port and enables them to spread SIP DoS defense capability throughout the network rather than solely relying on legacy DoS defense appliances that are installed in regional or global data center.

References

[1] S. Ehlert, C. Wang, T. Magedanz, and D. Sisalem, "Specification-based denial-of-service detection for SIP voice-over-IP networks," *Proceedings - The 3rd International Conference on Internet Monitoring and Protection, ICIMP 2008*, pp. 59–66, 2008.

[2] A. Lahmadi and O. Festor, "SecSip: A stateful firewall for SIP-based networks," *2009 IFIP/IEEE International Symposium on Integrated Network Management, IM 2009*, pp. 172–179, 2009.

[3] M. Ozcelik, N. Chalabianloo, and G. Gur, "Software-Defined Edge Defense Against IoT-Based DDoS, " *IEEE CIT 2017 - 17th IEEE International Conference on Computer and Information Technology*, 2017.

[4] H. Sengar, H. Wang, D. Wijesekera, and S. Jajodia, "Detecting voIP floods using the hellinger distance," *IEEE Transactions on Parallel and Distributed Systems*, vol. 19, no. 6, pp. 794–805, 2008.

[5] E. Chen, "Detecting DoS attacks on SIP systems," *1st IEEE Workshop on VoIP Management and Security, 2006.*, pp. 2–7, 2006.

[6] H. Sengar, D. Wijesekera, H. Wang, and S. Jajodia, "VoIP intrusion detection through interacting protocol state machines," *Proceedings of the International Conference on Dependable Systems and Networks*, vol. 2006, pp. 393–402, 2006.

[7] A. Stavrou, J. Voas, and I. Fellow, "DDoS in the IoT," *Computer*, vol. 50, pp. 80–84, 2017.

[8] M. Shtern and M. Litoiu, "Towards Mitigation of Low and Slow Application DDoS Attacks," no. Vm, 2014.

[9] Z. Lu, W. Wang, and C. Wang, "On the Evolution and Impact of Mobile Botnets in Wireless Networks," *IEEE Transactions on Mobile Computing*, vol. PP, no. 99, pp. 2304–2316, 2015.

[10] M. Guri, Y. Mirsky, and Y. Elovici, "9-1-1 DDoS: Attacks, Analysis and Mitigation," *Proceedings - 2nd IEEE European Symposium on Security and Privacy, EuroS and P 2017*, pp. 218–232, 2017.

[11] A. Berger and M. Hefeeda, "Exploiting SIP for botnet communication," *5th IEEE Workshop on Secure Network Protocols, NPSEC'09*, pp. 31–36, 2009.

[12] S. Donovan, R. Sparks, and C. Cunningham, "[RFC3665] Session Initiation Protocol (SIP) Basic Call Flow Examples, BCP75," pp. 1–94, 2003.

[13] J. Rosenberg, A. Johnston, J. Peterson, R. Sparks, M. Handley, and E. Schooler, "RFC 3261: Session Initiation Protocol (SIP)," *Internet Engineeging Task Force*, vol. 1, no. 11, pp. 1829–1841, 2002.

[14] J. Yu, "An Empirical Study of Denial of Service (DoS) against VoIP," *Proceedings - 2016 15th International Conference on Ubiquitous Computing and Communications and 2016 8th International Symposium on Cyberspace and Security, IUCC-CSS 2016*, pp. 54–60, 2017.

[15] M. Collier, *Hacking Exposed Unified Communications VoIP Security Secrets Solutions*. McGraw-Hill Osborne Media, 2014.

[16] P. Bosshart, D. Daly, G. Gibb, M. Izzard, N. Mckeown, J. Rexford, C. Schlesinger, D. Talayco, A. Vahdat, G. Varghese, and D. Walker, "P4: Programming Protocol-Independent Packet Processors,", 2014.