Adaptive Countermeasure by Credibility to Defend SDN Controllers Against BlackNurse-SC Attacks

You-Chiun Wang[1,2] and Pin-Yuan Wang[1]
[1]Department of Computer Science and Engineering; [2]Information Security Research Center
National Sun Yat-sen University, Kaohsiung, Taiwan
Email: ycwang@cse.nsysu.edu.tw; snowflake120101@gmail.com

***Abstract*—*Software-defined networking (SDN)* facilitates network management by using a controller to monitor the network status and regulate switches. Owing to its central control nature, the controller often becomes the target of attacks. *BlackNurse-SC*, an emerging type of DDoS attack, drains the controller of its computing resources via ICMP error messages and causes a disruption in the SDN network. To defend the controller against BlackNurse-SC attacks, we propose an *adaptive countermeasure by credibility (ACC)* that checks if hosts have suspected attacking behavior and assesses their credibility. Low-credibility hosts will have their ICMP packets blocked as a penalty, and the penalty duration is adjusted depending on the degree of credibility. In addition, the credibility of a host can be restored when it doesn't send ICMP error messages for a while. Simulation results reveal that the ACC scheme can protect a controller from BlackNurse-SC attacks effectively.**

***Keywords*—BlackNurse, controller, credibility, ICMP, SDN.**

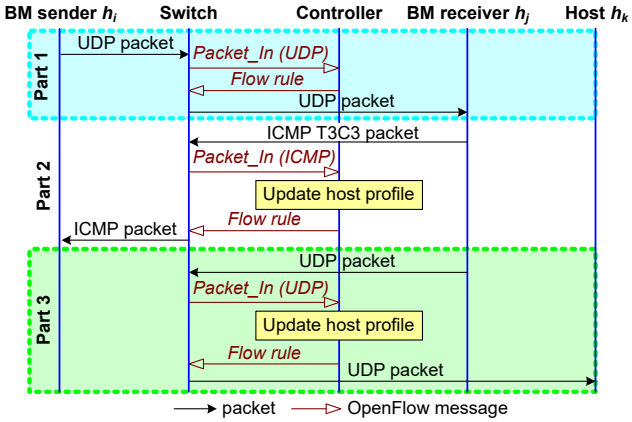


Fig. 1. Complete message flow in one iteration of BlackNurse-SC.

I. Introduction

A network can be segmented into control and data planes. The control plane supervises the network, and the data plane copes with packet forwarding. Traditionally, both planes are located in switches, making management jobs (e.g., applying new protocols or monitoring the network) cumbersome [1]. Hence, *software-defined networking (SDN)* abstracts the control plane and puts it into a controller to flexibly monitor and configure the network. The controller queries switches about their states and issues instructions to direct their operations. SDN has various applications, such as privacy preservation [2], road safety [3], data-center management [4], and smart cities [5]. It is also a key technology for B5G communication [6] and IoT service provisioning [7].

OpenFlow is widely used to implement SDN, which allows the controller to set flow rules for switches. Each switch finds appropriate flow rules in its flow table for an incoming packet. If there is a matching flow rule, the switch forwards or drops the packet based on that rule. Otherwise, the switch relays the packet that is wrapped in a *Packet_In message (PIM)* to the controller. After processing the packet, the controller assigns a flow rule to the switch [8].

However, as the controller is essential to an SDN network, it is a significant target for attacks. Recently, a novel type of attack called *BlackNurse-SC* (SC means "SDN controller") has been proposed [9], which aims to use up the controller's resources. It is a *distributed denial-of-service (DDoS)* attack evolving from a BlackNurse attack that targets the firewall. BlackNurse-SC sends packets to *botnet members (BMs)* in the SDN network, making the controller set flow rules for the packets. BMs are compromised computers and IoT devices (below, they are all referred to as *hosts*) whose security has been breached. The attacker then orders BMs to create ICMP error messages. Doing so not only makes switches send many PIMs to the controller but also forces the controller to be busy dealing with ICMP error messages and updating its topology view (frequently but incorrectly). Eventually, the controller's computing resources will be exhausted, thereby causing a denial-of-service to legitimate users.

Unlike most DDoS attacks, BlackNurse-SC does not produce a flood of packets, which would raise the difficulty of identifying the attack. Thus, this paper proposes an *adaptive countermeasure by credibility (ACC)* to defend the controller against BlackNurse-SC. The controller records a trust value for every host in the SDN network to keep evaluating the host's credibility. The more ICMP error messages a host sends, the lower its trust value. Once the trust value falls below a threshold (i.e., low credibility), the host's ICMP packets are blocked as a penalty. The threshold is adaptively adjusted based on the network status. When a host does not send ICMP error messages for some time, its credibility can be restored. Moreover, the duration of blocking ICMP packets is adjusted based on the degree of a host's credibility. Simulation results reveal that the ACC scheme can significantly reduce topology changes made by the controller and PIMs sent by switches. This indicates that ACC can efficiently resist BlackNurse-SC.

II. BlackNurse-SC Attack

BlackNurse-SC exploits ICMP T3C3 (type 3 code 3) messages, which point out that destinations cannot be reachable due to unreachable ports, to oblige the controller to frequently update its *host profile*. This profile is used for topology view,

which records the MAC address, IP address, and ingress port ID for each host that the controller learns.

BMs are classified into *senders* and *receivers*. BM receivers are in the SDN network. BM senders transmit UDP packets to all BM receivers, where these packets are non-spoofed and small-sized. Fig. 1 shows the complete message flow in one iteration of BlackNurse-SC, which contains three parts:

1. BM sender h_i transmits a UDP packet to BM receiver h_j. When the switch obtains the packet, it sends a PIM to the controller. Then, the controller finds a route to h_j and issues a flow rule to the switch. With the flow rule, the switch can forward the UDP packet to h_j.

2. After getting the UDP packet, h_j returns an ICMP T3C3 packet to the switch, which is relayed to the controller. Then, the controller does three things: 1) delete h_j's entry from its host profile to update the topology view, 2) send a *flow-mod* message to the switch to change the action of the flow rule related to h_j from 'forward' to 'drop', and 3) set a flow rule in the switch to forward the ICMP packet to h_i.

3. h_j picks a host h_k to send a UDP packet, where $h_k \neq h_i$. Again, the switch sends a PIM to the controller since it has no suitable flow rules. In this case, the controller checks its host profile and finds that there is no entry for h_j. Hence, the controller adds an entry for h_j to the host profile to update the topology view and sets a flow rule in the switch to forward h_j's packet to h_k. Moreover, the controller needs to send a flow-mod message to ask the switch to remove the drop flow rule (for h_j) which was set in Part 2.

As can be seen, switches send many PIMs to the controller. Besides, the controller repeatedly removes and adds entries for BM receivers in the host profile to update its topology view. Doing so will exhaust its computing resources.

III. Related Work

A. SDN-based Countermeasures Against DDoS Attacks

DDoS attacks usually generate many packets to consume network bandwidth, and how to apply SDN to counter these attacks has been widely discussed. The work [10] checks if a switch has more incoming UDP packets than outgoing ones, which is a sign of a UDP flooding attack. Both [11] and [12] let the controller inspect packet features (e.g., TCP flags) to identify attacks. In [13], the controller uses a nested reverse-exponential storage method [14] to record packet information and checks whether there are attack flows based on their flow sizes, IP variability, and durations. Salaria et al. [15] adopt a principal component analysis to detect attacks and convert a large set of traffic data into a small data set to facilitate the analysis. Considering that attack packets may originate from a botnet of hosts whose IP addresses are diverse, many studies [16]–[18] detect DDoS attacks by estimating the entropy of the source IP addresses of packets.

Though BlackNurse-SC belongs to DDoS attacks, it aims to paralyze an SDN controller by using only a few UDP and ICMP packets. Hence, the above countermeasures cannot be used efficiently to resist BlackNurse-SC.

B. Countermeasures Against BlackNurse Attacks

BlackNurse is a low-rate ICMP attack capable of causing denial-of-service to many commercial firewalls. As indicated in [19], this attack can generate just 50K ICMP T3C3 packets per second to let the firewall's CPU load reach almost 100%. The work [20] discusses some mitigation methods and points out their disadvantages: 1) Discarding ICMP (T3C3) packets: Doing so affects legitimate users. 2) Monitoring ICMP flows: The effect is not good, as BlackNurse is a low-rate attack. 3) Using a whitelist: This results in low scalability. 4) Upgrading the firewall's CPU: The hardware cost will increase. Hayawi et al. [21] estimate the attack's duration via a Markov chain, whose objective is to drop attack packets as early as possible. However, some normal ICMP packets are also discarded (i.e., false alarms). Both [22] and [23] apply the fission technique, which employs virtual machines or Kubernetes containers to process ICMP packets. They target mitigating the impact of BlackNurse instead of detecting and stopping the attack.

BlackNurse-SC is a variant of BlackNurse, and there is no effective defense method yet [9]. This motives us to propose the ACC scheme, which evaluates the credibility of each host and adaptively adjusts the duration of blocking ICMP packets sent from low-credibility hosts.

IV. The Proposed ACC Scheme

BlackNurse-SC relies on BM receivers in the SDN network to attack, but these BMs do not frequently send ICMP T3C3 packets. So, the difference in behavior between BM receivers and legitimate hosts is not apparent. Hence, the ACC scheme takes two measures. First, each host h_j in the SDN network is associated with a three-tuple *credibility entry* (λ_j, f_j, τ_j), where $\lambda_j \in \mathbb{Z}_0^+$ is the trust value, $f_j \in \{\text{true}, \text{false}\}$ indicates whether h_j sends ICMP T3C3 packets in the current period, and τ_j is the penalty duration (in seconds). Second, on getting an ICMP T3C3 packet sent by h_j, the controller then deletes h_j from its host profile but still keeps h_j's credibility entry. As mentioned in Section II, BM receivers will be repeatedly added to and removed from the controller's topology view to consume its computing resources. Keeping credibility entries can help the controller stably evaluate the credibility of hosts from which it has learned. When a host h_j is removed from the topology and does not appear for a long time (e.g., more than 3 hours), the controller discards h_j's credibility entry to save its memory space.

In Section IV-A, we explain how to set credibility entries for hosts. Section IV-B adjusts a host's credibility entry based on its behavior in sending ICMP T3C3 packets and disposes of the host accordingly (specifically, punishing it or restoring its credibility). We then detail the penalty mechanism for low-credibility hosts in Section IV-C. After that, Section IV-D has a discussion on the ACC scheme.

A. Setting Credibility Entries

When the controller discovers that a host h_j is added to the SDN network (according to the PIM sent from a switch) and there is no credibility entry for h_j (which implies that h_j is a *new* host), the controller assigns a credibility entry $(\lambda_j, \text{false}, 0)$ to h_j. In particular, $f_j = \text{false}$ (i.e., not sending ICMP T3C3 packets yet) and $\tau_j = 0$ (i.e., no penalty). The initial value for λ_j is set to either λ_{H} or λ_{L}, where $\lambda_{\text{H}} > \lambda_{\text{L}}$, based on the mode used:

Normal mode: There are only a few ICMP T3C3 packets sent, so we set $\lambda_j = \lambda_{\text{H}}$. Each host is allowed to send more ICMP T3C3 packets to reflect the topology's change.

Alert mode: If the rate of sending ICMP T3C3 packets significantly rises, there is a good possibility that a BlackNurse-SC attack is launched. Hence, we set $\lambda_j = \lambda_{\mathtt{L}}$ to mitigate the attacking effect of BM receivers as early as possible.

To decide the mode, we check the following condition:

$$(P_{\mathtt{max}} - P_{\mathtt{s}})/P_{\mathtt{max}} < \delta, \tag{1}$$

where $P_{\mathtt{max}}$ is the maximum number of ICMP T3C3 packets that the controller can handle per second, $P_{\mathtt{s}}$ is the number of ICMP T3C3 packets sent in a second, and $0 < \delta < 1$ is a threshold. If Eq. (1) holds, it means that the rate of sending ICMP T3C3 packets increases abnormally, so the alert mode is used. Otherwise, we employ the normal mode.

B. Assessing and Disposing of Hosts

Let $\hat{\mathcal{H}}$ be the set of hosts in the SDN network known by the controller, and T be the period length. Algorithm 1 gives the pseudocode to assess and dispose of hosts in $\hat{\mathcal{H}}$, which is performed period by period. This algorithm has two phases. Phase 1 contains lines 1–7, and phase 2 covers lines 8–15.

In phase 1, whenever a host $h_j \in \hat{\mathcal{H}}$ sends an ICMP T3C3 packet within the current period, its trust value λ_j is deducted by one and flag f_j is set to true. The code is given in lines 3–5. Notice that to avoid the value of λ_j becoming negative (where this may occur when h_j keeps sending ICMP T3C3 packets), we thus set λ_j to $\max\{\lambda_j - 1, 0\}$ in line 4. Then, once λ_j falls below a threshold $\lambda_{\mathtt{th}}$, h_j is considered to be attacking the controller. Consequently, the penalty mechanism in Section IV-C will be performed to block h_j's ICMP T3C3 packets, as shown in lines 6 and 7.

Phase 2 is carried out at the end of a period. If flag f_j is true, it implies that host h_j sent ICMP T3C3 packets during the period. Hence, we just set its flag f_j to false, and h_j has to employ the residual value of λ_j for assessment in the next period, as shown in lines 9 and 10. Otherwise, h_j behaved normally (i.e., without transmitting ICMP T3C3 packets), and its credibility can be restored. As discussed in Section IV-A, h_j's trust value λ_j is reset to $\lambda_{\mathtt{L}}$ or $\lambda_{\mathtt{H}}$ when the current mode is alert or normal, respectively, where $\lambda_{\mathtt{L}} < \lambda_{\mathtt{H}}$. The code is shown in lines 11–15. Resetting the trust value of a host (i.e., restoring its credibility) can avoid excessively blocking the host's ICMP T3C3 packets, thereby reducing false alarms.

Algorithm 1: Assessing and Disposing of Hosts

```
1  while Period T has not yet expired do
2      foreach h_j ∈ Ĥ do
3          if h_j sends an ICMP T3C3 packet then
4              λ_j ← max{λ_j − 1, 0};
5              f_j ← true;
6              if λ_j < λ_th then
7                  Punish h_j by the penalty mechanism
                     in Section IV-C;
8  foreach h_j ∈ Ĥ do
9      if f_j = true then
10         f_j ← false;
11     else
12         if mode = alert then
13             λ_j ← λ_L;
14         else
15             λ_j ← λ_H;
```

C. Penalty Mechanism

When Algorithm 1 judges that a host h_j is launching the BlackNurse-SC attack, h_j's subsequent ICMP T3C3 packets will be blocked to prevent h_j further attacking the controller. The duration τ_j of blocking ICMP T3C3 packets is decided based on the degree of h_j's credibility. In particular, if h_j's trust value is lower (which means that h_j has sent more ICMP T3C3 packets), the penalty duration becomes longer, and vice versa. To do so, we calculate the penalty duration as follows:

$$\tau_j = \left\lceil \frac{\lambda_{\mathtt{th}} - \lambda_j}{\lambda_{\mathtt{th}}/m} \right\rceil \times t_{\mathtt{base}}, \tag{2}$$

where the duration is divided into m levels ($m > 1$) and $t_{\mathtt{base}}$ is the basic time length (in seconds). By line 6 in Algorithm 1, the condition $\lambda_{\mathtt{th}} > \lambda_j$ holds, so Eq. (2) ensures that $\tau_j \geq t_{\mathtt{base}}$. Let us take an example. Suppose that $\lambda_{\mathtt{th}} = 30$, $m = 3$, and $t_{\mathtt{base}} = 15$s. According to trust value λ_j, there are three levels of penalty duration for h_j: 1) If $20 \leq \lambda_j \leq 29$, $\tau_j = \lceil \frac{30-20}{30/3} \rceil \times 15 = \cdots = \lceil \frac{30-29}{30/3} \rceil \times 15 = 15$s. 2) If $10 \leq \lambda_j \leq 19$, $\tau_j = \lceil \frac{30-10}{30/3} \rceil \times 15 = \cdots = \lceil \frac{30-19}{30/3} \rceil \times 15 = 30$s. 3) If $0 \leq \lambda_j \leq 9$, $\tau_j = \lceil \frac{30-0}{30/3} \rceil \times 15 = \cdots = \lceil \frac{30-9}{30/3} \rceil \times 15 = 45$s.

Let s_k be the switch that connects with h_j. The controller installs a flow rule in s_k to ask it to discard h_j's ICMP T3C3 packets. The hard_timeout field of this flow rule is set to τ_j. In this way, after τ_j seconds, s_k will remove the flow rule by itself and stop discarding h_j's ICMP T3C3 packets.

During the execution of the penalty, if h_j still sends ICMP T3C3 packets (these packets will be dropped), s_k then records the number $N_j^{\mathtt{pen}}$ of ICMP T3C3 packets sent from h_j. After finishing the penalty, s_k passes the information about $N_j^{\mathtt{pen}}$ to the controller, and h_j's trust value is set to $\max\{\lambda_j - N_j^{\mathtt{pen}}, 0\}$ (i.e., to replace line 4 in Algorithm 1).

D. Discussion

Let us discuss the rationale of our ACC scheme. Observe the complete message flow in Fig. 1. BlackNurse-SC exploits ICMP T3C3 packets sent from BM receivers in Part 2 to force the controller to update its host profile and issue flow rules to switches, where these rules are actually unnecessary. After BM receivers have been removed from the controller's topology view, they transmit UDP packets to make the controller repeat similar things in Part 3. Doing so will eventually use up the controller's computing resources. Therefore, ACC aims to block malicious ICMP T3C3 packets. In this way, the message flow in Part 2 will be stopped. Moreover, since these BM receivers are still kept in the controller's topology view, even if they send UDP packets in Part 3, the controller need not update the host profile and set flow rules in switches. In other words, the message flow in Part 3 can also be stopped.

In effect, BM receivers are not fixed. A legitimate host in the SDN network may become a BM receiver if it is installed with malware (e.g., a virus or a Trojan horse). On the other hand, a BM receiver can revert to a legitimate host once its malware has been removed. Hence, traditional solutions, such

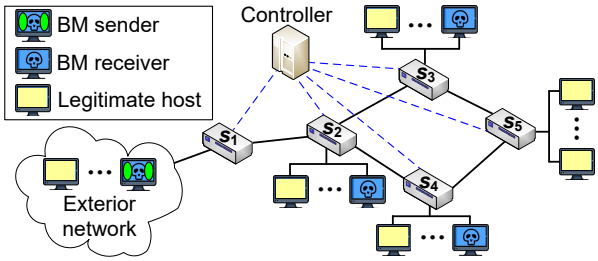


Fig. 2. Network topology used in the simulation.

as using a blacklist to record BM receivers, may not perform well. Instead, the ACC scheme assesses the credibility of each host based on its behavior in sending ICMP T3C3 packets, which is more practicable and efficient.

ACC uses Algorithm 1 to evaluate the credibility of hosts. Low-credibility hosts have their ICMP T3C3 packets blocked to stop their attack. Based on Eq. (2), the blocking duration will extend if hosts have sent more ICMP T3C3 packets. On the other hand, the credibility of some hosts can be restored when they don't send ICMP T3C3 packets for a period of time. The above design has two benefits. First, if a legitimate host *accidentally* sends too many ICMP packets, it will only be penalized for a short time, after which its credibility can be restored. In this way, we can reduce false alarms. Second, when a BM receiver keeps sending ICMP T3C3 packets, its trust value (i.e., λ_j) will be kept low for a long time. In this case, Algorithm 1 together with the penalty mechanism keeps blocking its (malicious) ICMP T3C3 packets, which prevents the BM receiver from attacking the controller.

V. Performance Evaluation

We simulate an SDN network with five switches by using Mininet [24], as Fig. 2 shows. Switch s_1 acts as a gateway to exterior networks, in which BM senders reside. Switches s_2, s_3, and s_4 connect with legitimate hosts and BM receivers. For switch s_5, it links with only legitimate hosts. To enable OpenFlow, the controller and switches are respectively implemented by the Ryu framework [25] and the Linux Open vSwitch module [26]. Moreover, we employ the Hping3 tool [27] to generate ICMP T3C3 packets.

As mentioned in Section III-B, there is no effective defense method against BlackNurse-SC yet. In the literature, a few studies (e.g., [28]) also adopt the concept of credibility for attack detection, but they block malicious packets for a fixed time. So, we propose a *static credibility-based defense (SCD)* method for comparison, which reckons the credibility of each host h_j by a trust value λ_j. The initial value of λ_j is 15, and λ_j is deducted by one whenever h_j sends an ICMP T3C3 packet. Once λ_j becomes zero, h_j is prohibited from sending ICMP T3C3 packets for $t_{\texttt{base}}$ seconds, where $t_{\texttt{base}}$ is set to 15s. After that, λ_j is reset to 15.

Since BlackNurse-SC consumes the controller's computing resources by making it frequently change the topology view, we thus measure the number of topology changes made by the controller, which is defined as the number of entries updated in the host profile. Besides, BlackNurse-SC forces switches to send many PIMs to the controller to ask for flow rules, as shown in Fig. 1. Thus, we measure the number of PIMs sent by switches. For the ACC scheme, we set its parameters as follows: $\delta = 0.5$, $\lambda_{\texttt{H}} = 60$, $\lambda_{\texttt{L}} = 45$, $\lambda_{\texttt{th}} = 30$, and $m = 5$. The simulation time is 300s.

TABLE I
Three scenarios used in experiment 1.

scenario	BM senders	BM receivers	legitimate hosts
A1	5	15	30
A2	10	10	30
A3	15	5	30

A. Experiment 1: BM Senders and Receivers

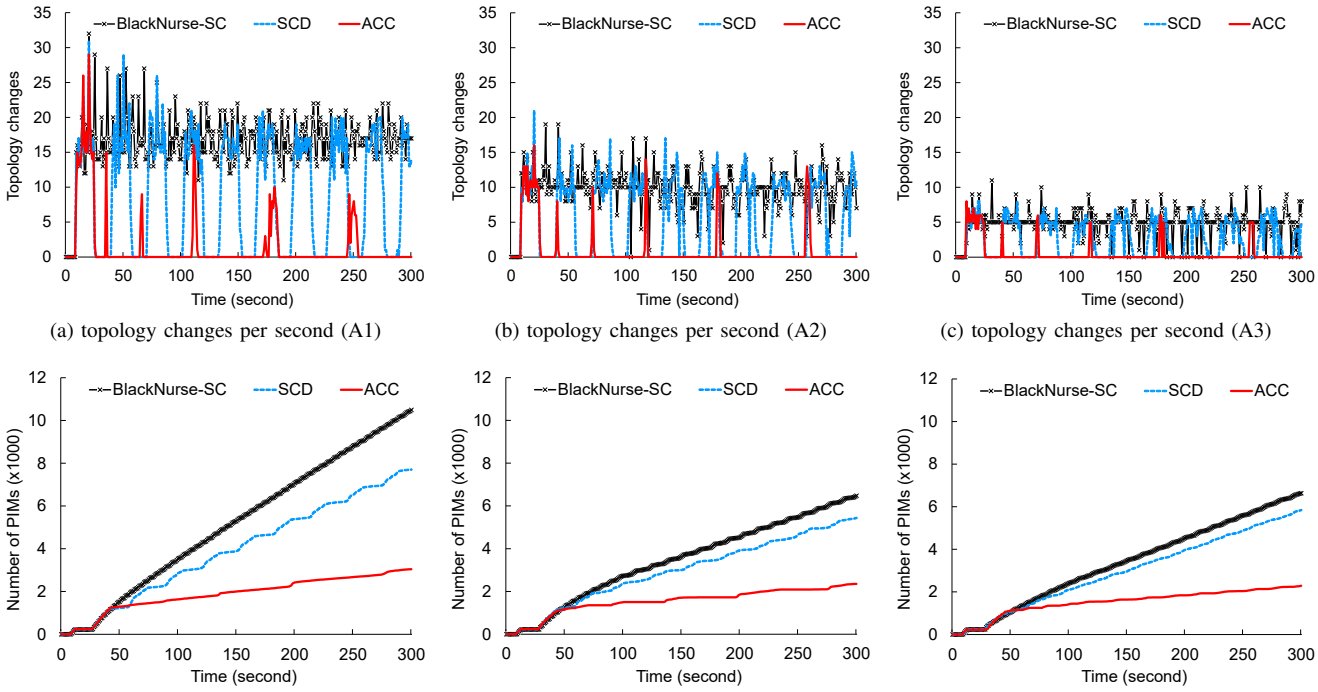
In experiment 1, we change the number of BM senders and receivers to study its effect, as shown in Table I. Beginning in the 10th second, the attacker launches one BlackNurse-SC attack per second. Following the message flow in Fig. 1, each BM sender sends UDP packets to all BM receivers to trigger the attack in every iteration of BlackNurse-SC.

Fig. 3(a), (b), and (c) give the number of topology changes per second in scenarios A1, A2, and A3. Apparently, when there are more BM receivers (e.g., scenario A1), the controller needs to perform more topology changes. The reason can be found in Fig. 1, where the controller updates its host profile (for topology changes) due to receiving ICMP T3C3 packets and their subsequent UDP packets from BM receivers. Without blocking the malicious ICMP T3C3 packets, BlackNurse-SC compels the controller to frequently change its topology view, which will eventually exhaust its computing resources. The attack's effect becomes stronger with more BM receivers. By assigning each host a trust value for evaluating its credibility, the SCD method can reduce the number of topology changes that the controller performs. However, SCD uses a static policy, where the duration to block ICMP T3C3 packets is fixed to 15s. After the penalty, the trust value of a host is always reset to 15, no matter whether the host still sent ICMP T3C3 packets during the penalty. Hence, we can observe that the number of topology changes in SCD will increase rapidly and then drop to zero periodically.

Thanks to the penalty mechanism in Section IV-C, our proposed ACC scheme punishes BM receivers for a longer time, thereby eliminating most malicious ICMP T3C3 packets. As mentioned in Section IV-D, doing so prevents the controller from updating its host profile in Parts 2 and 3 of Fig. 1. Thus, ACC significantly decreases the number of topology changes. More concretely, our ACC scheme reduces 90.95%, 89.72%, and 89.51% of the topology changes caused by BlackNurse-SC in scenarios A1, A2, and A3. As compared with the SCD method, ACC can diminish 83.20%, 82.40%, and 80.96% of topology changes in scenarios A1, A2, and A3.

Fig. 3(d), (e), and (f) show the aggregate number of PIMs sent by switches in scenarios A1, A2, and A3. Let us observe Fig. 1. For each pair of BM sender and receiver, one PIM is caused by the BM sender, and two PIMs are caused by the BM receiver. Hence, the number of PIMs reduces as there are fewer BM receivers. This phenomenon is especially manifest when there is no defense against BlackNurse-SC. By blocking some ICMP T3C3 packets, the SCD method can eliminate a part of PIMs caused by BM receivers.

ACC drops more malicious ICMP T3C3 packets than SCD, so most PIMs in ACC may be solely caused by BM senders.



(a) topology changes per second (A1) (b) topology changes per second (A2) (c) topology changes per second (A3)

(d) aggregate PIMs (A1) (e) aggregate PIMs (A2) (f) aggregate PIMs (A3)

Fig. 3. Performance evaluation in experiment 1.

That explains why the gap between SCD and ACC rises as time goes by. Specifically, the ACC scheme reduces 70.98%, 63.71%, and 65.49% of the PIMs caused by BlackNurse-SC in scenarios A1, A2, and A3. As compared with the SCD method, ACC can save 60.46%, 56.74%, and 60.83% of PIMs in scenarios A1, A2, and A3.

B. Experiment 2: Attack Interval

Then, we observe the effect of the attack interval. There are 5 BM senders, 15 BM receivers, and 30 legitimate hosts. Besides scenario A1 (whose interval is 1s), two scenarios are considered: in scenarios A4 and A5, starting from the 10th second, a BlackNurse-SC attack is launched every 5s and 10s.

Fig. 4(a) and (b) present the number of topology changes in scenarios A4 and A5. As compared with scenario A1 in Fig. 3(a), there are fewer topology changes when the interval between two attacks increases. That is because the number of attacks in the simulation diminishes. In scenarios A4 and A5, our ACC scheme eliminates 95.30% and 87.91% of the topology changes that BlackNurse-SC brings about. Besides, ACC decreases 91.88% and 83.25% of topology changes in scenarios A4 and A5, as compared to the SCD method.

Fig. 4(c) and (d) show the aggregate number of PIMs in scenarios A4 and A5. The trend is similar to that in scenario A1 in Fig. 3(d). The ACC scheme saves 72.76% and 54.19% of the PIMs caused by BlackNurse-SC in scenarios A4 and A5. Compared to the SCD method, ACC can reduce 64.42% and 46.22% of PIMs in scenarios A4 and A5.

VI. Conclusion

In an SDN network, the controller plays a key role and is easily the target of attacks. BlackNurse-SC is a DDoS attack that aims to consume the controller's computing resources. Unlike many DDoS attacks, BlackNurse-SC exploits a few ICMP T3C3 packets to keep the controller busy updating its topology view. In this paper, we propose the ACC scheme to resist BlackNurse-SC attacks, which evaluates the credibility of hosts according to their behavior in transmitting ICMP T3C3 packets. For low-credibility hosts (i.e., potential BM receivers), their ICMP T3C3 packets are blocked to avoid attacking the controller, and the blocking time can be adjusted according to the degree of credibility. When a host doesn't transmit ICMP T3C3 packets, its credibility can be restored. With Mininet simulations, we show that the ACC scheme can efficiently reduce topology changes that the controller makes and PIMs sent by switches caused by attacks.

For future work, we will study the impact of BlackNurse-SC on the distributed SDN controllers, where each controller manages a subset of switches in the SDN network [29]. In addition, it deserves further investigation on how to protect controllers against BlackNurse-SC in a multi-domain SDN-based network [30]. This may require collaboration between controllers to detect a BlackNurse-SC attack.

Acknowledgment

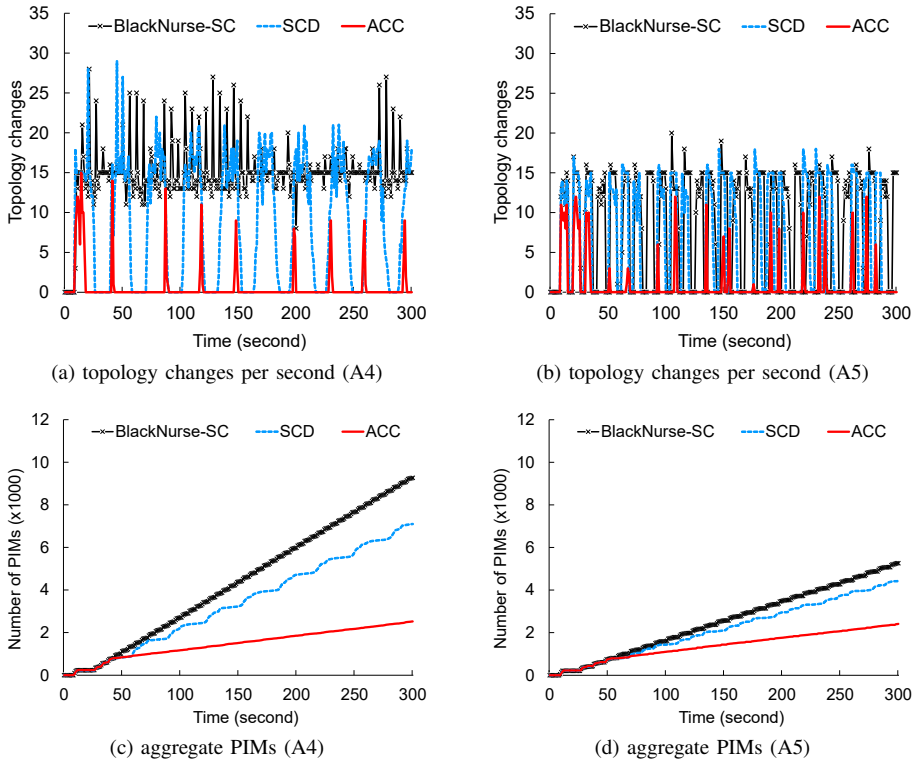
This work was supported by National Science and Technology Council, Taiwan under Grants 111-2221-E-110-023-MY2 and 112-2634-F-110-001-MBK, and Information Security Research Center at National Sun Yat-sen University.

References

[1] Y. C. Wang and S. Y. You, "An efficient route management framework for load balance and overhead reduction in SDN-based data center networks," *IEEE Transactions on Network and Service Management*, vol. 15, no. 4, pp. 1422–1434, 2018.

[2] Q. Xu, Z. Su, M. Dai, and S. Yu, "APIS: privacy-preserving incentive for sensing task allocation in cloud and edge-cooperation mobile Internet of Things with SDN," *IEEE Internet of Things Journal*, vol. 7, no. 7, pp. 5892–5905, 2020.

[3] J. V. Leon, O. G. Bautista, A. Aydeger, S. Mercan, and K. Akkaya, "A general and practical framework for realization of SDN-based vehicular networks," in *IEEE International Performance, Computing, and Communications Conference*, 2021, pp. 1–7.



(a) topology changes per second (A4)

(b) topology changes per second (A5)

(c) aggregate PIMs (A4)

(d) aggregate PIMs (A5)

Fig. 4. Performance evaluation in experiment 2.

[4] Y. C. Wang and T. J. Hsiao, "URBM: user-rank-based management of flows in data center networks through SDN," in *IEEE International Conference on Computer Communication and the Internet*, 2022, pp. 142–149.

[5] A. Sachan and N. Kumar, "SDN control-enabled and time-quantum-driven max-pressure approach for intersection management in smart city," *IEEE Systems Journal*, vol. 17, no. 1, pp. 1694–1702, 2023.

[6] S. H. A. Kazmi, F. Qamar, R. Hassan, and K. Nisar, "Routing-based interference mitigation in SDN enabled beyond 5G communication networks: a comprehensive survey," *IEEE Access*, vol. 11, pp. 4023–4041, 2023.

[7] W. Rafique, A. S. Hafid, and S. Cherkaoui, "Complementing IoT services using software-defined information centric networks: a comprehensive survey," *IEEE Internet of Things Journal*, vol. 9, no. 23, pp. 23 545–23 569, 2022.

[8] Y. C. Wang and H. Hu, "An adaptive broadcast and multicast traffic cutting framework to improve Ethernet efficiency by SDN," *Journal of Information Science and Engineering*, vol. 35, no. 2, pp. 375–392, 2019.

[9] N. Ravi and S. M. Shalinie, "BlackNurse-SC: a novel attack on SDN controller," *IEEE Communications Letters*, vol. 25, no. 7, pp. 2146–2150, 2021.

[10] Y. H. Tung, H. C. Wei, Y. W. Ti, Y. T. Tsou, N. Saxena, and C. M. Yu, "Counteracting UDP flooding attacks in SDN," *Electronics*, vol. 9, no. 8, pp. 1–27, 2020.

[11] P. Rengaraju, V. R. Ramanan, and C. H. Lung, "Detection and prevention of DoS attacks in software-defined cloud networks," in *IEEE Conference on Dependable and Secure Computing*, 2017, pp. 217–223.

[12] K. Kalkan, G. Gur, and F. Alagoz, "SDNScore: a statistical defense mechanism against DDoS attacks in SDN environment," in *IEEE Symposium on Computers and Communications*, 2017, pp. 669–675.

[13] Y. C. Wang and Y. C. Wang, "Efficient and low-cost defense against distributed denial-of-service attacks in SDN-based networks," *International Journal of Communication Systems*, vol. 33, no. 14, pp. 1–24, 2020.

[14] Y. C. Wang, Y. Y. Hsieh, and Y. C. Tseng, "Multiresolution spatial and temporal coding in a wireless sensor network for long-term monitoring applications," *IEEE Transactions on Computers*, vol. 58, no. 6, pp. 827–838, 2009.

[15] S. Salaria, S. Arora, N. Goyal, P. Goyal, and S. Sharma, "Implementation and analysis of an improved PCA technique for DDoS detection," in *IEEE International Conference on Computing Communication and Automation*, 2020, pp. 280–285.

[16] K. Kalkan, L. Altay, G. Gur, and F. Alagoz, "JESS: joint entropy-based DDoS defense scheme in SDN," *IEEE Journal on Selected Areas in Communications*, vol. 36, no. 10, pp. 1–24, 2018.

[17] M. Xuanyuan, V. Ramsurrun, and A. Seeam, "Detection and mitigation of DDoS attacks using conditional entropy in software-defined networking," in *IEEE International Conference on Advanced Computing*, 2019, pp. 66–71.

[18] C. S. Whittle and H. Liu, "Effectiveness of entropy-based DDoS prevention for software defined networks," in *IEEE International Symposium on Technologies for Homeland Security*, 2021, pp. 1–7.

[19] BlackNurse, http://blacknurse.dk/.

[20] Z. Trabelsi, S. Zeidan, and K. Hayawi, "Denial of firewalling attacks (DoF): the case study of the emerging BlackNurse attack," *IEEE Access*, vol. 7, pp. 61 596–61 609, 2019.

[21] K. Hayawi, Z. Trabelsi, S. Zeidan, and M. M. Masud, "Thwarting ICMP low-rate attacks against firewalls while minimizing legitimate traffic loss," *IEEE Access*, vol. 8, pp. 61 596–61 609, 2020.

[22] Y. Shan, G. Kesidis, D. Fleck, and A. Stavrou, "Preliminary study of fission defenses against low-volume DoS attacks on proxied multiserver systems," in *International Conference on Malicious and Unwanted Software*, 2017, pp. 67–74.

[23] A. F. Baarzi, G. Kesidis, D. Fleck, and A. Stavrou, "Microservices made attack-resilient using unsupervised service fissioning," in *European Workshop on Systems Security*, 2020, pp. 31–36.

[24] Mininet. [Online]. Available: http://mininet.org/

[25] Ryu. [Online]. Available: https://ryu-sdn.org/

[26] Open vSwitch. [Online]. Available: https://www.openvswitch.org/

[27] Hping3. [Online]. Available: http://www.hping.org

[28] Y. C. Wang and R. X. Ye, "Credibility-based countermeasure against slow HTTP DoS attacks by using SDN," in *IEEE Annual Computing and Communication Workshop and Conference*, 2021, pp. 890–895.

[29] W. K. Lai, Y. C. Wang, Y. C. Chen, and Z. T. Tsai, "TSSM: time-sharing switch migration to balance loads of distributed SDN controllers," *IEEE Transactions on Network and Service Management*, vol. 19, no. 2, pp. 1585–1597, 2022.

[30] Y. C. Wang and E. J. Chang, "Cooperative flow management in multi-domain SDN-based networks with multiple controllers," in *IEEE International Conference on Smart Communities: Improving Quality of Life Using ICT, IoT and AI*, 2020, pp. 82–86.