“© 2021 IEEE. Personal use of this material is permitted. Permission from IEEE must be obtained for all other uses, in any current or future media, including reprinting/republishing this material for advertising or promotional purposes, creating new collective works, for resale or redistribution to servers or lists, or reuse of any copyrighted component of this work in other works.”

CARS: Dynamic Cyber-attack Reaction in SDN-based Networks with Q-learning

Hai Hoang Nguyen*, Tri Gia Nguyen†, *Senior Member, IEEE*,
Dinh Thai Hoang‡, *Member, IEEE*, Duc Tran Le§, and Trung V. Phan**, *Member, IEEE*
*Vietnam - Korea University of Information and Communication Technology, The University of Danang, Vietnam
†FPT University, Danang 50509, Vietnam
‡School of Electrical and Data Engineering, University of Technology Sydney, NSW 2007, Australia
§University of Science and Technology, The University of Danang, Danang, Vietnam
**Chair of Communication Networks, Technische Universität Chemnitz, 09126 Chemnitz, Germany

***Abstract*—In this paper, we propose a dynamic cyber-attack reaction system based on Q-learning, namely CARS, to effectively defeat cyber-attacks in Software-Defined Networks (SDN). In particular, we first examine a cyber-attack reaction system that operates at the SDN control plane. Then, we propose a dynamic cyber-attack reaction solution to maximize the attack defense performance while minimizing the negative influence on benign traffic forwarding in the data plane. Next, we model the cyber-attack reaction system based on a Markov decision process (MDP) and formulate its optimization problem. Afterward, we develop a Q-learning based cyber-attack reaction control algorithm to solve the optimization problem, obtaining the optimal cyber-attack reaction policy. As our case study on denial-of-service (DoS) attacks, the obtained results verify that CARS can effectively prevent malicious packets from reaching the victim server in all DoS attacks, i.e., approximately 80% of abnormal packets are dropped. In addition, by implementing the optimal cyber-attack reaction policy, CARS can significantly reduce the ratio of QoS (Quality-of-Service) violated traffic flows compared to two existing solutions, i.e., GATE (by approx. 66%) and GTAC-IRS (by approx. 75%).**

***Index Terms*—Cyber-attack Reaction System, Q-learning, Denial-of-Service attacks and Software-Defined Networking.**

I. Introduction

Software-Defined Networking (SDN) [1] is an innovative networking model that departs the control from the data plane. Specifically, the SDN controller holds a global network view and performs forwarding decisions, while the SDN switches only operate traffic forwarding in the data plane. In addition, the communication between the control and data plane is managed via the OpenFlow protocol [1], which allows an SDN switch to perform the traffic forwarding task using flow-tables where flow rules can be dynamically added/removed/updated by the SDN controller. Therefore, numerous complicated network control and management tasks, e.g., network security, have been achieved by utilizing the SDN architecture [2]. However, several security vulnerabilities exist in the SDN-based networks, and it is challenging to tackle anomalies due to new attack surfaces at exposed networking layers [3]. Fortunately, with the recent adoption of Machine Learning (ML) to Software-Defined Networking, many critical security problems, e.g., denial-of-service (DoS) attacks, have been addressed [4]. Nevertheless, most researchers have paid attention to the evolution of intelligent intrusion detection systems. Meanwhile, only a few studies [5], [6] have discussed cyber-attack reaction systems that protect SDN-based networks by choosing proper policies to tackle malicious traffic flows [7].

For illustration, the authors in [5] propose a distributed game-theoretic cyber-attack reaction design based on an actor-critic model, namely GTAC-IRS, for SDN-based wireless IoT networks. Concretely, intrusion detection and reaction tasks are assigned to IoT devices by a centralized controller. Further, by utilizing a low-dimensional response matrix, the computational effort for intrusion detection and the training time of detection algorithms can be diminished. Nonetheless, it is challenging to deploy the GTAC-IRS in practice for the following reasons. Firstly, the delegation of attack detection and reaction tasks to IoT devices places enormous pressure on computational and resource-constrained IoT devices. Secondly, the GTAC-IRS module at each IoT device requires monitoring the packet reception rate of its neighbors, which raises a critical security issue among IoT devices. Finally, there are no collaborations between GTAC-IRS engines; thus, the attack detection and reaction performance might be limited, particularly for unknown cyber-attacks.

Likewise, Zolotukhin *et al.* [6] propose a reinforcement learning-based approach for attack mitigation in SDN-enabled networks. Specifically, an intelligent agent is developed to learn and find an optimal ensemble of virtual security functions, e.g., intrusion detection systems, proxies, and firewalls, which are then implemented in a sequence to filter the network traffic flows on the way from a source to a destination. Thereby, the anomaly detection and mitigation performance can be improved by installing an appropriate order of virtual security functions. Nevertheless, this solution [6] might be ineffective in large-scale networks with a large number of clients in the data plane, leading to a huge number of implemented virtual security functions.

As stated in [7], for a cyber-attack reaction system it is essential to optimally select policies which can effectively defend against abnormal activities. The authors in [8] propose a solution, namely GATE, in which the response policy selection is performed by relying on four factors: attack

** Corresponding author

damage, deployment cost, negative impact on QoS, and security benefit. Next, a single-objective optimization problem is formulated by applying the return-on-response-investment index, which is then solved by a proposed genetic algorithm with a three-dimensional encoding approach. In particular, the three dimensions include response policies, defense points, and deployment orders, respectively. Nevertheless, for medium or large-scale networks, e.g., the AttMpls network topology [9] with 25 nodes, the GATE can be ineffective due to a massive number of response meta-policies for the central cyber-attack reaction system.

Therefore, this paper proposes a dynamic cyber-attack reaction solution based on Q-learning, namely CARS, to defeat cyber-attacks in SDN-based networks effectively. In particular, we first study a cyber-attack reaction system associated with an intrusion detection system (IDS) placed on top of the SDN control plane. Subsequently, we propose a dynamic cyber-attack reaction solution to maximize the attack defense performance while minimizing the negative impact on the benign traffic forwarding performance in the data plane. Precisely, we model the cyber-attack reaction system based on a Markov decision process (MDP) approach and formulate its optimization problem. Next, we propose a Q-learning based cyber-attack reaction control algorithm, which solves the optimization problem to achieve the optimal cyber-attack reaction policy.

As our case study, we evaluate the CARS solution in denial-of-service (DoS) attacks in comparison with two other existing solutions, namely GATE [8], and GTAC-IRS [5]. The obtained results confirm that CARS can dynamically deploy the optimal cyber-attack reaction policy to tackle malicious traffic flows while minimizing the negative impact on the forwarding of legitimate traffic flows. Specifically, CARS can effectively prevent the malicious packets from arriving at the victim server in two DoS attack scenarios, i.e., approximately 80% of attack packets are dropped on average. Further, by implementing the optimal cyber-attack reaction policy, CARS can bypass flow-table overflows at SDN switches and link congestions. Thereby significantly lessening the ratio of QoS violated traffic flows by around 66% and 75% on average compared to GATE and GTAC-IRS, respectively.

This paper is organized as follows. Section II describes in detail the operational workflow and the objectives of the CARS solution in SDN-based networks. Section III provides a model of the control system based on a Markov decision process, and formulates the related optimization problem. Section IV presents the application of the Q-learning algorithm in solving the optimization problem. Section V shows the results of the performance evaluation considering three DoS attack scenarios. Section VI summarizes the findings of our study and outlines some ideas for future research.

II. System Model

Given an SDN-based network represented by an undirected graph $\mathbb{G} = (\mathcal{N}, \mathcal{E})$, as illustrated in Fig. 1, where $\mathcal{N} = \{N_i\}$ denotes the set of SDN switches, and $\mathcal{E} = \{l_{ij}\}$ is the set of links between the SDN switches in the data plane. In this

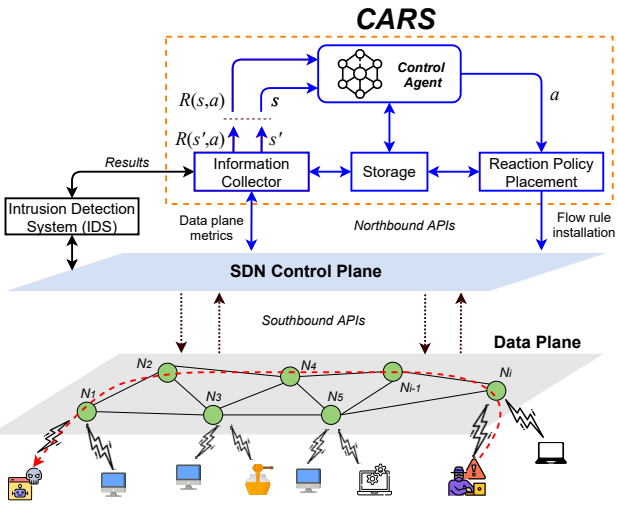


Fig. 1. CARS solution in an SDN-based network.

study, we propose a dynamic cyber-attack reaction system, called CARS, for SDN-based networks. As illustrated in Fig. 1, an intrusion detection system (IDS) and the CARS locate on top of the control plane to detect and react against anomalies in the data plane, respectively.

A. System Operation

Regarding anomaly detection at the IDS, the traffic information is regularly observed from the control plane and then analyzed by a detection engine where malicious traffic behavior can be identified through intelligent methods, e.g., a deep learning-based approach [4]. If an anomaly is recognized, the detection result and the attack information are transferred to the CARS.

For the CARS operation, we propose a dynamic cyber-attack reaction system consisting of the following modules: an information collector, a control agent, a data storage, and a reaction policy placement. Specifically, whenever the IDS detects an anomaly, the attack information, i.e., the attack type τ, the IP addresses of the attacker IP_A and the victim IP_V, and the malicious traffic path $\mathcal{P}_A$ (dashed red line in the data plane in Fig. 1), is transferred to the CARS. Precisely, the path $\mathcal{P}_A$ consists of a set of switches and links, referred to as $\mathcal{N}_A$ and $\mathcal{L}_A$, respectively, where $\mathcal{N}_A \subseteq \mathcal{N}$ and $\mathcal{L}_A \subseteq \mathcal{E}$. Meanwhile, the information collector periodically gathers the following metrics: the flow-table utilization f_i concerning the flow rule quantity in every SDN switch N_i on the path $\mathcal{P}_A$, the link utilization u_{ij} of every link l_{ij} on path $\mathcal{P}_A$, and the average throughput Ω (Mbps) of all SDN switches that are not on the path $\mathcal{P}_A$.

Based on the above attack information and the gathered metrics, the control agent defines an appropriate reaction policy to defend against the attack effectively and minimize adverse effects on traffic forwarding in the data plane. Next, the reaction policy placement module maps the policy into flow rules that are consequently installed in flow-tables of selected SDN switches to prevent malicious traffic. Details of the switch selection for reaction policy implementation

are provided in Section III (Action Space). Afterward, by using northbound APIs, the flow rule installation request is forwarded to the SDN control plane to implement the reaction policy in the data plane. Lastly, the control agent gathers the IDS results and the data plane metrics to evaluate the reaction policy's effectiveness in the subsequent observation. The above processes are periodically repeated to investigate different policies and observations.

B. System Objectives

Our ultimate intention is to maximize the attack defense performance of the CARS while minimizing the adverse effects on the benign traffic forwarding process. To accomplish this goal, we examine the following sub-objectives:

Attack damage minimization: The more malicious packets are dropped before reaching the victim since the anomaly is identified, the less harm the victim and the network infrastructure suffer; therefore, the total number of malicious packets, $\mathcal{M}$, that is dropped by the switches should be maximized.

Negative impact minimization: Whenever a response policy is activated, it can negatively impact the legitimate traffic forwarding process. In this paper, we consider the reduction of the average throughput $\Delta\Omega$ of all switches that are not on the attack path $\mathcal{P}_A$.

Accordingly, the objective function of the CARS is defined as follows:

$$\min\left(\frac{\eta}{|\mathcal{N}_A|\times\sum_{N_i\in\mathcal{N}_A}\mathcal{M}_i}+\delta\Delta\Omega\right),\ \text{s.t.}\begin{cases} f_i<1.0,\forall N_i\in\mathcal{N}_A,\\ u_{ij}<1.0,\forall l_{ij}\in\mathcal{L}_A,\end{cases} \tag{1}$$

where η and δ denote the normalization factors of $\mathcal{M}$ and $\Delta\Omega$, respectively and $|\mathcal{N}_A|$ is the number of SDN switches on the path $\mathcal{P}_A$. $f_i < 1.0$ shows that there is no flow-table overflow at any switch N_i [10] while $u_{ij} < 1.0$ represents that legitimate packets can be successfully forwarded over links on $\mathcal{P}_A$. For our convenience, $\Theta = \frac{\eta}{|\mathcal{N}_A|\times\sum_{N_i\in\mathcal{N}_A}\mathcal{M}_i}+\delta\Delta\Omega$ is used in the paper remaining.

Theoretically, the control agent might get closer to its objectives by performing different reaction policies in the data plane. Nevertheless, it is challenging to obtain an optimal reaction policy at a time step because of the high number of traffic flows in the data plane [5], [7].

III. Problem Formulation

To achieve the objectives in (1), we adopt a Markov decision process (MDP) based approach [11] to represent the CARS operation, which allows the control agent obtain optimal actions based on its observations to maximize its average long-term reward. The MDP is defined by $<\mathcal{S},\mathcal{A},\mathcal{R}>$ where $\mathcal{S}$ is the state space, $\mathcal{A}$ is the action space, and $\mathcal{R}$ is the immediate reward function.

State Space: We define the state space of the system as follows:

$$\mathcal{S}\triangleq\left\{(\tau,\varrho,\zeta,\upsilon,\psi):\upsilon=\frac{\sum_{N_i\in\mathcal{N}_A}f_i}{\varrho},\psi=\frac{\sum_{l_{ij}\in\mathcal{L}_A}u_{ij}}{\zeta}\right\}, \tag{2}$$

where τ is the attack type, $\varrho = |\mathcal{N}_A|$ is the number of SDN switches on path $\mathcal{P}_A$, $\zeta = |\mathcal{L}_A|$ is the number of links on path $\mathcal{P}_A$, υ is the average flow-table utilization at the switches on path $\mathcal{P}_A$, and ψ is the average utilization of the links on path $\mathcal{P}_A$. Note that, in this paper, we consider one digit in the decimal part of υ and ψ, which means that $\upsilon,\psi\in\{0.0,0.1,0.2,0.3,0.4,0.5,0.6,0.7,0.8,0.9,1.0\}$. In addition, in case there is no attack, then τ=0, $\varrho = |\mathcal{N}|$, and $\zeta = |\mathcal{E}|$. A state is hence defined a $s = (\tau,\varrho,\zeta,\upsilon,\psi)\in\mathcal{S}$.

Action Space: Whenever an attack is identified, the control agent is required to select an appropriate reaction policy - otherwise, it does nothing. Consequently, the action space can be represented as

$$\mathcal{A}\triangleq\begin{cases} a:a\in\mathcal{W},\ \text{if } \tau\neq 0,\\ \text{do nothing},\ \text{if } \tau=0,\end{cases} \tag{3}$$

where $\mathcal{W}$ is a list of available reaction policies, which is described in detail in our case study in Section V because the list of reaction policies can be diverse for different cyber-attacks. As stated in [8], a reaction policy can be implemented at different defense points in the network. In this study, for policies that require to add/update flow rules, e.g. for blocking an IP address, the reaction policy is deployed at a switch $N_i\in\mathcal{N}_A$, where either the flow-table of the subsequent switch $N_j\in\mathcal{N}_A$ is the most utilized or the subsequent link $l_{ij}\in\mathcal{L}_A$ has the highest utilization u_{ij}. As a result, both the flow-tables of switches and the links on path $\mathcal{P}_A$ can be protected from overflow and overload cause by malicious traffic, respectively.

Immediate Reward Function: After performing a reaction policy to defeat the recognized attack, the control agent gains an immediate reward proportional to the attack defense performance, which is denoted as $\frac{1}{\Theta}$. However, if any switch flow-tables get overflowed or any links reach their bandwidth limit, the control agent should be punished by decreasing its reward. Moreover, if there are no IDS attack warnings, the control agent takes a zero reward. Therefore, the immediate reward of the CARS is defined as follows:

$$\mathcal{R}(s,a)=\begin{cases} \frac{1}{\Theta},\ \text{if } \tau\neq 0 \text{ and } f_i<1.0,\forall N_i\in\mathcal{N}_A \text{ and } u_{ij}<1.0,\\ \forall l_{ij}\in\mathcal{L}_A,\\ \frac{1}{\Theta}-\sum f_i-\sum u_{ij},\ \text{if } \tau\neq 0 \text{ and } f_i=1.0,\exists N_i\in\mathcal{N}_A\\ \text{ or } u_{ij}=1.0,\exists l_{ij}\in\mathcal{L}_A,\\ 0,\ \text{if } \tau=0.\end{cases} \tag{4}$$

In this study, we aim to achieve the optimal cyber-attack reaction policy, $\pi^*(s)$, that maximizes the average long-term reward of the CARS:

$$\max_{\pi}\mathfrak{R}(\pi)=\sum_{t=1}^{\infty}\mathbb{E}\big(\mathcal{R}(s_t,\pi(s_t))\big),\quad \text{s.t. } f_i<1.0,\forall N_i\in\mathcal{N}_A \text{ and } u_{ij}<1.0,\forall l_{ij}\in\mathcal{L}_A, \tag{5}$$

where $\mathfrak{R}(\pi)$ represents the average reward of the control agent under the policy π and $\mathcal{R}(s_t,\pi(s_t))$ denotes the immediate reward under the policy π at time step t.

Note that, in this paper, for any time step t we examine a cyber-attack that stems from a single client and is targeted to a single server. We further develop new algorithms to defeat attacks originating from multiple clients within one observation period (time step) in our future work.

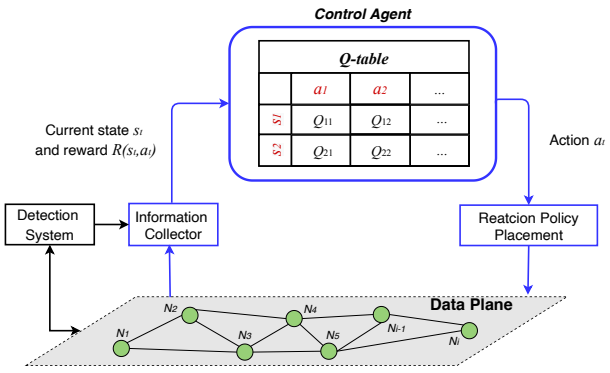


Fig. 2. Q-learning based cyber-attack reaction control in SDN.

IV. Q-LEARNING BASED CYBER-ATTACK REACTION CONTROL IN SDN-BASED NETWORKS

In this section, we propose a Q-learning based cyber-attack reaction control algorithm to solve the optimization problem stated in (5). Precisely, as illustrated in Fig. 2, a Q-table is deployed to store all state-action pairs of the data plane, and a control agent can learn from its decisions at each time step or learning iteration [11]. In addition, the Q-learning algorithm can obtain the optimal control policy $\pi^*(s)$ after a certain number of learning iterations [11]. In addition, the expected return for the control agent at state s under policy π is referred to as $\vartheta_\pi(s):\mathcal{S}\longrightarrow\mathbb{R}$, which is determined as

$$\begin{aligned}\vartheta_\pi(s) &= \mathbb{E}_\pi\left[\sum_{t=0}^{\infty}\gamma\mathcal{R}(s_t,a_t)|s_t=s\right]\\ &= \mathbb{E}_\pi\left[\mathcal{R}(s_t,a_t)+\gamma\vartheta_\pi(s_{t+1})|s_t=s\right],\forall s\in\mathcal{S},\end{aligned} \tag{6}$$

where $\gamma\in[0,\ 1]$ is the discount factor that represents the importance of the long-term reward [11]. To acquire the optimal policy $\pi^*(s)$ at a state s, the optimal action can be obtained by applying the optimal value function as follows:

$$\vartheta^*(s)=\max_a\left\{\mathbb{E}_\pi\left[\mathcal{R}(s_t,a_t)+\gamma\vartheta_\pi(s_{t+1})|s_t=s\right]\right\},\forall s\in\mathcal{S}. \tag{7}$$

Hence, for all state-action (s,a) pairs, the optimal Q-functions can be defined as

$$Q^*(s,a)\triangleq\mathcal{R}(s_t,a_t)+\gamma\mathbb{E}_\pi\left[\vartheta_\pi(s_{t+1})\right],\forall s\in\mathcal{S}. \tag{8}$$

Therefore, the optimal value function $\vartheta^*(s)$ are determined as $\vartheta^*(s)=\max_a\{Q^*(s,a)\}$. Moreover, by deploying different actions a to the data plane, the optimal Q-function value for all state-action (s,a) pairs can be achieved. In particular, the Q-function is updated at each iteration by the following function:

$$\begin{aligned}Q_{t+1}(s_t,a_t) &= Q_t(s_t,a_t)+\alpha[\mathcal{R}(s_t,a_t)\\ &\quad+\gamma\max_a Q_t(s_{t+1},a_{t+1})-Q_t(s_t,a_t)],\end{aligned} \tag{9}$$

where $s_t\in\mathcal{S}$, $a_t\in\mathcal{A}$, and $Q_t(s_t,a_t)$ is the Q-value for a state-action pair (s_t,a_t), and $\mathcal{R}(s_t,a_t)$ is the immediate reward obtained from the data plane at iteration t, and $\alpha\in[0,1]$ is the learning rate. Furthermore, we use the ϵ-greedy scheme [11] to relieve the exploration and exploitation. Finally, Algorithm 1 gives details of our Q-learning based cyber-attack reaction control algorithm.

Algorithm 1 Q-learning based Cyber-attack Reaction Control

1: Initialize a Q-table, and initialize a Q-table entry for a state-action pair (s,a) $\forall s\in\mathcal{S}$, $a\in\mathcal{A}$.
2: Initialize values of α, γ, and ϵ, and T (terminal condition in an episode), respectively.
3: **for** episode $\phi\in\{1,2,...,\phi_{max}\}$ **do**
4: **for** $t\in\{1,2,...,T\}$ **do**
5: Given current state s_t.
6: Select a reaction policy a_t with probability ϵ, and implement a_t at the data plane.
7: Observe a new state s_{t+1} and calculate an immediate reward $\mathcal{R}$.
8: Update the Q-table entry for $Q(s_t,a_t)$ using Eq. (9).
9: Update $s_t\longleftarrow s_{t+1}$.
10: Go to next episode if $t=T$.
11: **end for**
12: **end for**
13: **Output** $\pi^*(s)=\arg\max_a Q^*(s,a)$.

TABLE I
PARAMETER SETTINGS OF THE Q-LEARNING ALGORITHM

Discount factor γ	0.99
Epsilon-greedy policy ϵ [start, end]	[1.0, 0.1]
Learning iteration t (seconds)	5.0
Iterations in an episode T	100
Learning rate α	0.6

V. PERFORMANCE EVALUATION

A. Environment Setup

To evaluate the performance of the CARS, we emulate an SDN-based network by using the MaxiNet tool [12]. The network topology is adopted from a real-life network example, namely AttMpls [9], and comprises 25 OvS (Open vSwitch) switches ($|\mathcal{N}|=25$) and 57 links. Specifically, 10 virtual container-based clients, one Web server and one honeypot are connected to every OvS switch. All OvSes are under the supervision of an ONOS SDN controller. The emulation operates on a physical machine with an AMD Ryzen 7 3800X CPU with clock speed 8x3.9GHz, 32 GB RAM, and an NVIDIA GeForce RTX 3060 GPU. The CARS modules are located at another physical machine with the same configuration.

B. Denial-of-Service Attacks and Reaction Policies

Denial-of-Service (DoS) attacks are the most famous ones and significantly challenging for mitigation. Accordingly, in this paper, we consider two well-known denial-of-service attacks, namely:

TCP SYN flood: The attacker tries to open as many TCP connections as possible, leading to a flow-table overflow at SDN switches on the attack path and an overload of the victim server. According to [10], the flow-table capacity of an OvS is approximately 3,000 flow rules in an emulation setup.

Link layer flood: The attacker intends to overload the network links by continuously sending packets with a massive payload on the attack path, e.g., Ping-of-dead, making authentic traffic unable to communicate with the victim server.

With respect to DoS detection, we use a deep learning-based DoS detection approach for the IDS (see Fig. 1). Table II represents the list $\mathcal{W}$ of all reaction policies for the two mentioned DoS attacks, i.e., $\mathcal{W}$=$\{a_1, a_2, a_3, a_4, a_5, a_6, a_7\}$. In addition, Table I shows the hyperparameters of the Q-learning algorithm used in the CARS.

C. Traffic Generation Strategies

Regarding traffic generation, all clients randomly access Web servers, making background traffic. Meanwhile, the Hping3 tool is installed on clients to generate TCP SYN and ICMP flood attacks to Web servers and overload the network links. Hence, a client can simultaneously produce both legitimate and attack traffic flows. Furthermore, as discussed in Section III, at a time step, we consider a DoS attack from a single client to a single Web server. At each step, the client who generates attack traffic and the Web server is randomly selected. Lastly, we evaluate the CARS utilizing two attack traffic scenarios: TCP SYN flood and Link layer flood.

D. Comparable Solutions

We compare the CARS with the GATE [8] and the GTAC-IRS [5] concerning the DoS attack defense performance and the ratio of QoS violated traffic flows. In particular, details of the GATE and the GTAC-IRS are as follows:

Centralized cyber-attack reaction solution: In GATE [8], the cyber-attack reaction system is located at a central server, which is similar to CARS. Accordingly, to represent the GATE operation, we employ the same network setup with 25 OvSes, as explained in Section V-A, and place the GATE modules on a physical machine. Moreover, the meta-policies are formed by mixing 3 reaction policies (i.e., *blockIP-1min*, *limitRate*-50%, and *doNothing*) with 25 defense points (OvSes). That means when an attack is recognized, 75 meta-policies could be performed by the central server to defeat the attack.

Distributed cyber-attack reaction solution: In GTAC-IRS [5], the detection and reaction tasks are distributed and locally operated at IoT devices. Therefore, for the SDN setup discussed in Section V-A, we used the Snort software [13] on clients to reflect the primary concept of GTAC-IRS, i.e., devices can recognize anomalies by monitoring and intervening in the local traffic. Moreover, if the Snort puts an alert, one out of the 3 reaction policies, i.e., *blockIP-1min*, *limitRate*-50%, and *doNothing*, can be performed at the client to defeat the malicious traffic.

E. Results Analysis

1) Convergence of Q-learning algorithm: Firstly, we investigate the convergence of the Q-learning algorithm used in the CARS for two different DoS attack scenarios. As illustrated in Fig. 3, the Q-learning algorithm can obtain a significant reward (i.e., approximately 1.0) for all two scenarios after one hundred thousand learning iterations and maintain this learning performance in the remaining, which means that the optimal cyber-attack reaction policy is found. Specifically, it needs about 125,000 learning iterations for the TCP SYN flood.

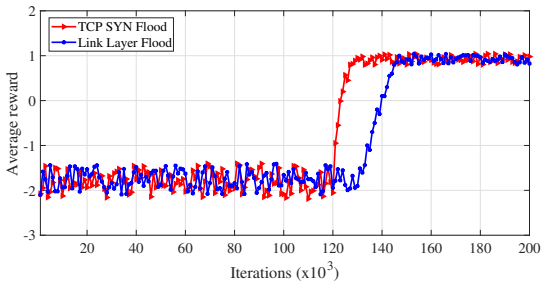
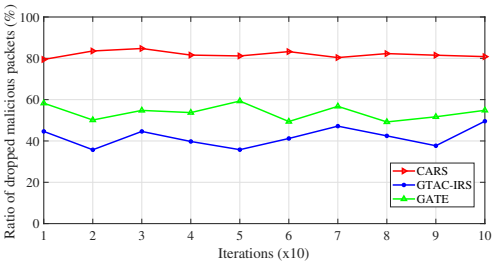
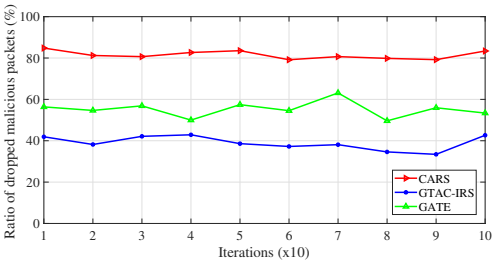


Fig. 3. Convergence of the Q-learning algorithm in the CARS.



(a) TCP SYN flood



(b) Link layer flood

Fig. 4. Ratio of dropped malicious packets since a DoS attack is detected.

In comparison, it requires approximately 145,000 learning iterations for the link-layer flood to obtain the optimal reaction policy. To sum up, the Q-learning based cyber-attack reaction control algorithm, i.e., Algorithm 1, can efficiently solve the optimization problem in (5).

2) DoS attack defense performance: We examine the attack defense performance achieved by CARS, GATE, and GTAC-IRS by measuring the ratio of discarded malicious packets since an attack is discovered. As exhibited in Fig. 4, CARS can efficiently stop malicious packets from reaching the victim in all two DoS attack scenarios, i.e., approximately 80% of attack packets are dropped on average. For GATE and GTAC-IRS, the rate of dropped abnormal packets is significantly lower, i.e., 30% and 45% on average, respectively. In particular, for the GATE, due to the many meta-policies, the genetic algorithm-based reaction policy selector could not find a set of best policies. Accordingly, GATE cannot effectively block the attack packets from entering the victim server. In the case of GTAC-IRS, the Snort at the client picks one of three reaction policies (including the *doNothing* policy) to counter the abnormal packets, leading to a significant number of attack packets that are delivered to the victim. In summary, by employing the optimal cyber-attack reaction policy, the CARS

TABLE II
RESPONSE POLICIES FOR DOS ATTACKS

Policy	Notation	Meaning
blockIP-1min	a_1	Drop all incoming packets with the attacker's IP address for 1 minute
blockIP-3min	a_2	Drop all incoming packets with the attacker's IP address for 3 minutes
limitRate-50%	a_3	Reduce the rate of incoming packets from the attacker by 50% compared to the current rate
removeMaliciousFlows	a_4	Remove all abnormal traffic flow rules at SDN switches on the attack path
toHoneyPot	a_5	Redirect the attack traffic flows to a predefined honeypot
reRoute	a_6	Redirect the attack traffic flows to another path in the data plane
doNothing	a_7	Do nothing

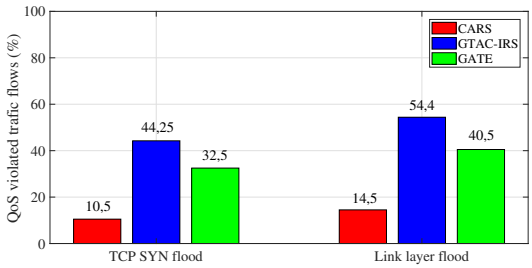


Fig. 5. Ratio of QoS violated traffic flows.

outperforms the GATE and the GTAC-IRS considering the attack defense performance.

3) Ratio of QoS violated traffic flows: Finally, we evaluate the performance of CARS by measuring the ratio of QoS violated legitimate traffic flows for two DoS attack scenarios. As stated in [10], a traffic flow from a source to a destination is identified as a QoS violated flow in the case at least one of QoS requirements is not fulfilled. Hence, we consider the end-to-end delay and the packet loss probability as QoS metrics for benign traffic flows. As shown in Fig. 5, CARS is superior to GATE and GTAC-IRS solutions considering the number of QoS violated legitimate traffic flows. Specifically, for the two DoS attack scenarios, by applying the optimal cyber-attack reaction, CARS can reduce the ratio of QoS violated traffic flows by approximately 66%, and 75% on average compared to GATE and GTAC-IRS, respectively. Furthermore, as addressed in Section 3, the CARS implements policies to drop/limit/redirect the malicious traffic flows at an SDN switch, where either the flow-table of the next switch on the attack path or the following link is most utilized. Accordingly, no flow-table overflow and link overload problems are recognized, significantly increasing the probability of successfully forwarding benign packets. By selecting random actions (e.g., *doNothing*), GATE and GTAC-IRS enable the malicious traffic flows to consume most of the network resources, i.e., the flow-table utilization of SDN switches and the link capacity utilization. Consequently, only a portion of legitimate packets is forwarded successfully and in time in the data plane. To sum up, the CARS outperforms the other solutions concerning the ratio of QoS violated traffic flows.

VI. CONCLUSION

In this paper, we introduce a novel dynamic cyber-attack reaction solution based on Q-learning, namely CARS, to effectively defeat cyber-attacks while reducing the negative impact on benign traffic forwarding in SDN-based networks. To maximize the attack defense performance, we have developed a Q-learning based cyber-attack reaction control algorithm to obtain the optimal cyber-attack reaction policy. For our case study on TCP SYN flood and Link layer flood attacks, the obtained results prove that CARS can effectively prevent malicious packets from reaching the victim while significantly lessening the ratio of QoS violated traffic flows compared to GATE and GTAC-IRS. As our future study, we intend to develop new cyber-attack reaction algorithms to defend against different cyber-attacks simultaneously.

ACKNOWLEDGMENT

This research is funded by Vietnam National Foundation for Science and Technology Development (NAFOSTED) under Grant 102.01-2019.322.

REFERENCES

[1] W. Xia, Y. Wen, C. H. Foh, D. Niyato, and H. Xie, "A survey on software-defined networking," *IEEE Communications Surveys Tutorials*, vol. 17, no. 1, pp. 27–51, 2015.
[2] P. Tsai, C. Tsai, C. Hsu, and C. Yang, "Network monitoring in software-defined networking: A review," *IEEE Systems Journal*, vol. 12, no. 4, pp. 3958–3969, Dec. 2018.
[3] A. L. Buczak and E. Guven, "A survey of data mining and machine learning methods for cyber security intrusion detection," *IEEE Communications Surveys Tutorials*, vol. 18, no. 2, pp. 1153–1176, 2016.
[4] Y. Zhao and et. al., "A survey of networking applications applying the software defined networking concept based on machine learning," *IEEE Access*, vol. 7, pp. 95397–95417, 2019.
[5] B. Wang and et. al., "Game-theoretic actor–critic-based intrusion response scheme (gtac-irs) for wireless sdn-based iot networks," *IEEE Internet Things J.*, vol. 8, no. 3, pp. 1830–1845, 2021.
[6] M. Zolotukhin and et. al., "Reinforcement learning for attack mitigation in sdn-enabled networks," in *2020 6th IEEE Conf. on Network Softwarization (NetSoft)*, pp. 282–286, 2020.
[7] P. Nespoli and et. al., "Optimal countermeasures selection against cyber attacks: A comprehensive survey on reaction frameworks," *IEEE Commun. Surveys Tuts.*, vol. 20, no. 2, pp. 1361–1396, 2018.
[8] Y. Guo and et. al., "Decision-making for intrusion response: Which, where, in what order, and how long?," in *2020 IEEE Int. Conf. on Communications (ICC)*, pp. 1–6, 2020.
[9] S. e. a. Knight, "The internet topology zoo," *IEEE J. Sel. Areas Commun.*, vol. 29, no. 9, pp. 1765–1775, 2011.
[10] T. Phan and et. al., "Deepmatch: Fine-grained traffic flow measurement in sdn with deep dueling neural networks," *IEEE J. Sel. Areas Commun.*, pp. 1–1, 2020.
[11] R. S. Sutton and et. al., *Introduction to Reinforcement Learning*. Cambridge, MA, USA: MIT Press, 1st ed., 1998.
[12] P. Wette, M. Draxler, and A. Schwabe, "Maxinet: Distributed emulation of software-defined networks," in *2014 IFIP Networking Conference*, pp. 1–9, June 2014.
[13] Docker-Snort, "Docker-snort: A free and lightweight network intrusion detection system (nids) software." https://github.com/dnif-archive/docker-snort, May 2021.