IMPROVED DEEP REINFORCEMENT LEARNING BASED DEFENSE SYSTEM FOR DISTRIBUTED DENIAL OF SERVICE ATTACK

[1]A. SeemaM.Sc(CS)., B.Ed.,
M. Phil Scholar,
Department of Computer Science,
Dr. SNS Rajalakshmi College of Arts and Science,
Coimbatore – 49.

[2]Dr. P. Shivaranjani M.Sc(CS)., M.Phil., Ph.D.
Academic Coordinator,
Department of Graphic & Creative design and Data Analytics,
Dr. SNS Rajalakshmi College of Arts and Science,
Coimbatore – 49.

ABSTRACT

Edge computing is the standard integration of cloud and fog computing which has limited access because of the development of Internet-of-Things (IoT) devices and large amount of data are loomed at the edge nodes in the network. Various features of edge computing including on-demand activities, pricing strategies, fast resistance etc., make it more susceptible to different kinds of Distributed Denial of Service (DDoS) attack. It is because of the direct involvement of the edge nodes on the network to enable decisions in real time without a huge amount of information being broadcast by the cloud or edge servers. Therefore, it is more important for identifying and defending these attacks in the network with the help of effective methods. One of the effective methods for defending DDoS attack is Deep Reinforcement Learning (DRL) where the network structure and network traffic information were analyzed using Q-learning to defend the DDoS attack. In DRL, perceptual aliasing occurs when many states share nearly identical features of network structure and traffic information. A consequence of perceptual aliasing is that the DRL agents struggle to learn the relationship between features (network structure and traffic) and utility of particular actions (deployment of classifier in edge server). In order to handle the perceptual aliasing in DRL based DDoS defense system, a Newtonian Action Advice (NAA) is introduced in this paper. In NAA, an advice is provided to take proper action i.e., deploying a classifier in an edge server. Moreover, policy strategy in DRL is improved by engaging combined policy gradient with DRL. It used a value function approximation with the policy which enables to process with continuous action spaces. Based on the Q values, the policy gradient deploys a classifier in the edges for defending DDoS attack.

Keywords: Edge computing, cloud computing, internet-of-things, Distributed Denial of Service, Newtonian Action Advice, combined policy gradient with deep reinforcement learning.

DOINumber: 10.48047/NQ.2022.20.20.NQ109047 **NeuroQuantology2022;20(20): 443-450**

1. INTRODUCTION

A huge volume of data is generated with the development of Internet-of-Things (IoT) where a large number of devices are connected to the internet. This exponential development of data have added immense simplicity to the lives of individuals posed significant challenges to current internet services, in particular network security. Distributed Denial of Service (DDoS) [1] is the most serious danger in the modern network. DDoS targets the availability of a system or service. DDoS attack will leads to financial loss user's dissatisfaction and loss of trust. DDoS attacks are more difficult to defend. It is due to DDoS attack traffic is attack traffic caused by network system vulnerabilities and huge volume of legitimate traffic that meets the system regulation. Even though these legitimate traffic sill has features that prove it to be attack traffic, the system cannot intercept it using conventional methods.

DDoS attack can be defended effectively by using a classifier which intercepts traffic and it cannot be deployed on routers where the traffic is concentrated. The place of the DDoS defense strategy is also important [2-4]. The network is considered as a tree with the target server as the root. The traffic passes from leaf nodes and is gathered at the root node. Generally, a classifier was deployed on server side or redirected the combined traffic to the cloud server for classification. Due to the computational complexity related to deploy a defense strategy, this is susceptibility for DDoS attacks that created a huge volume of traffic which may lead to protected services degradation. Hence, an approach to categorize traffic at distributed points is a better for DDoS attack mitigation. It categorized the traffic previous to it was combined by deploying a classifier upstream or minimize the downstream traffic quantity.

A novel DDoS defense system called Deep Reinforcement Learning (DRL) [5] was suggested which shifting cloud filtering activities to edge servers. First, edge servers were implemented at different router positions and executed classifiers for filtering the traffic sent via it. To lessen the attack traffic, reverse the client traffic and minimize inspection delays, an enhanced deep reinforcement learning model was designed for balancing the implementation of computing resource and tasks assignment in which graph neural network was applied for coding the system architecture data transfer as vector and traffic data to input into Q-network for acquiring the assignment decisions. The deep reinforcement learning used to learn the best combination of a network structure with varying traffic by learning the greedy algorithms. In the deep reinforcement learning, an agent is one who takes decision about learning based on reward and punishments. Based on Q-learning, the classifier was deployed on the edge servers and it inspected the traffic flow for DDoS defense system. However in DRL, the perceptual aliasing occurs when many states share nearly identical features which affect the performance of DDoS attack defense system.

So in this paper, an Improved deep Reinforcement Learning for DDoS Defense System (IRLDDS) is proposed to handle the perceptual aliasing in DRL and improves the performance of DDoS attack defense system. Newtonian Action Advice (NAA) algorithm is introduced in IRLDDS where an advice is obtained from an interaction to deploy a classifier at a node. In addition to this, a policy gradient is combined with DRL to enhance the performance of mitigation process. It used a value function approximation with the policy which enables to process with continuous action spaces. According to the Q-values, the policy gradient deploys a classifier in the nodes to defense the DDoS attack in the network.

2. LITERATURE SURVEY

3. PROPOSED METHODOLOGY

In this section an Improved Deep Reinforcement Learning for DDoS Defense System (IRLDDS) is described in detail. A network is considered as a directed tree-like graph $G = (V, E)$ with each vertex $v \in V$ denotes a router where defense strategies

deploy and every edge $e \in E$ is the path in the routing table. The root r of the graph G is the protection target. A classifier can be deployed on any node in graph G. A classifier is described as a 3-tuple $n = (t, j, m)$ which represents the efficiency of a defense strategy. The parameter $t(j) \in [0,1]$ denotes the proportion of legacy traffic that classifier can interdicted. N represents the set of all optional classifiers. P is defined as the collection of options on node v, where $o_i \in \{0,1\}$ and $o_i = 1$ denotes the i-th classifier is chosen. The percentage of legal traffic captured for node v is denoted as $t_v = \prod_{i=1}^{|P_v|}(t_i)^{p_i}\left(j_v = \prod_{i=1}^{|P_v|} j_i^{p_i}\right)$. The parameter $q \in \{0,1\}$ represents the penalty factor caused by delay for detecting and $q_v = \prod_{i=1}^{|P_v|}(q_i)^{p_i}$ is the factor on node v. A defender strategy $D = \langle P_v \rangle$ is an allocation of every options in the nodes set V. All leaf nodes are traffic sources. Traffic is the addition of all traffic linked to this leaf node which is represented as $H = \{v | v \in V, D_v = \phi\}$ as the collection of all leaf nodes and D_v as the collection of child nodes of the node v. A parameter $a\_t_v^{in}$ denotes the attack traffic that passes into the n which is given as follows,

$$a_{t_v^{in}} = \begin{cases} a, & v \in J \\ \sum_{i \in D_v} a_{t_v^{out}}, & v \notin J \end{cases} \quad (3.1)$$

In Eq. (3.1), a denotes the traffic coming from outside the system and $a_{t_v^{out}}$ denotes the traffic which passes out from the node i, i.e., $a_{t_v^{out}} = t_v a_{t_v^{in}}. a_{j_v^{in}}$ has same description as,

$$a_{j_v^{in}} = \begin{cases} a, & v \in J \\ \sum_{i \in D_v} a_{j_v^{out}}, & v \notin J \end{cases} \quad (3.2)$$

The alteration in traffic replicates the effect of the deployment strategy. Apparently, the traffic that passes into the protected targets are $a_{t_v^{out}}$ and $a_{j_v^{out}}$. Eliminating attack traffic, reserving legal traffic as much as possible and minimizing inspection delays are

$$\tau_v^{(a+1)} \leftarrow F\left(P_v, \left\{\tau_e^{(a)}\right\}_{e \in D_v}; \vartheta\right) \quad (3.4)$$

In Eq. (3.4), D_v denotes the set of children of node v in graph G, and F denotes a nonlinear function such as neural network. The starting embedding $\tau_v^{(0)}$ at each node is 0. three objectives which are considered for DDoS defense system. Once the traffic that passes into the network is identified, the target is a function corresponded to the defense strategy. Assume $B_1(D)$ is the whole flows that reach the protection target i.e., $B_1(D) = a_{t_b^{out}}$. $B_1(D)$ is the traffic that ate wrongly interdicted, i.e., $B_2(D) = \sum_{v \in J} a_{j_v^{in}} - a_{j_b^{out}}$. For the node v, its penalty caused by delay for detecting is $q_v^i\left(a_{t_v^{in}} + a_{j_v^{in}}\right)$, and the whole penalty $B_3(D) = \sum_{v \in V} q_v\left(a_{t_v^{in}} + a_{j_v^{in}}\right)$. The whole goal B is given as follows,

$$B(D) = \alpha B_1(D) + \beta B_2(D) + \gamma B_3(D) \quad (3.3)$$

In Eq. (3.3), α, β and γ represent the weights of three goals. Apparently, the defender need to identify a best strategy P to minimize the whole goal B.

3.3.1 Graph Embedding

A method based on Graph Convolutional Network (GCN) and improved reinforcement learning is designed which learn the structural information of the graph and finds a best strategy to minimize the whole goal. The network information is extracted by using GCN and it is combined with the traffic information to obtain the node vector. The improved reinforcement learning gets the node vector as input and it returns Q-value as output. The 2-tuple (v^*, i^*) with the smallest 1-value is chosen for deployment. Initially, the graph embedding network calculate a p-dimensional feature embedding τ_v for every node $v \in V$. A recursive method is used to compute the node representation τ_v. In every iteration of computation, the node v obtains information from its children nodes and computes the newest τ_v as,

According to Eq. (3.4), it is it is known that the node embedding update process is based on the graph topology and the embedding from the prior round. A node only obtains the detail from their children nodes. Apparently, only if

there are enough iterations the node can obtain all structural information of a sub tree which rooted at it. A parameter F is designed to update a p-dimensional embedding τ_v as,

$$\tau_v^{(a+1)} \leftarrow relu\left(\theta_1 P_v + \theta_2 \sum_{e \in D_v} \tau_v^{(a)}\right) \quad (3.5)$$

In Eq. (3.5), $\theta_1 \in \mathbb{R}^{p \times |P_v|}$ and $\theta_2 \in \mathbb{R}^{p \times p}$ are the model parameters and $relu$ denotes the rectified linear unit ($relu(x) = \max(0, x)$) applied element wise to its input. The 0-1 vector P_v denotes the option of the node v.

3.3.2 Parameterizing Q-Function

$$Q(p(D), v; \Theta) = \theta_3 \left[relu([\theta_3 \textstyle\sum_{e \in V} \tau_v, \theta_3 \tau_v]), a_{t_v^{out}}, a_{j_v^{out}}\right] \quad (3.6)$$

In Eq. (3.6), $\theta_3 \in \mathbb{R}^{|P_v| \times (2p+2)}$ and $\theta_4, \theta_5 \in \mathbb{R}^{p \times p}$ are the model parameters. $[.,.]$ is the concatenation operator. The output of Eq. (3.6) is the $|P_v|$-dimensional vector, that is the Q-value of all classifier on node v. It is more complex to learn parameters $\Theta = \{\theta_i\}_{i=1}^{5}$ because of the lack of training labels. Hence, these parameters are learned using improved deep reinforcement learning.

3.3.3 Improved deep Reinforcement Learning

The improved deep reinforcement learning is used to learn the best combination of a network structure with varying traffic. In reinforcement learning, an agent ought to take actions in an environment in order to maximize the rewards. Here, the agent ought to deploy a classifier in the edge server with minimum whole goal. In the reinforcement learning, there may be high amount of perceptual aliasing which occurs when many states shares nearly similar features. A consequence of perceptual aliasing is that the deep reinforcement learning agents struggle to learn the relationship between features (network structure and traffic) and utility of particular actions (deployment of classifier in edge server).

This problem is resolved by introducing Newtonian Action Advice (NAA) which enables a deep reinforcement learning agent to learn from action advice. NAA persists advice input across the time steps. A friction parameter specifies a count of time steps during which the action advice is recited by the agent back to itself. A force model allows each piece of advice to be generalized through time. An agent in the improved deep reinforcement learning will continue their deploying a classifier at a particular position until the friction causes the agent to resume the normal exploration. If advice was followed in one state, the NAA will cause the agent to follow the same advice if the state is seen in the future.

The embedding is used to compute the Q-function. τ_v is used for node v and the addition of all nodes embedding $\sum_{v \in V} \tau_v$ denotes the partial solution $par(D)$. The traffic information is also required to compute the Q-value of every classifier of each node. So, the Q-function $Q(p(D), v; \Theta)$ is given as follows:

The agent, state, advice, action and rewards in the improved deep reinforcement learning are defined as follows:

- Agent: At every time step, the agent listens for advice (which defines the movement of the deployment of classifier). If advice is given, the agent updates its internal advice dictionary. Then agent then chooses and deploys a classifier at the edge servers with minimum whole goal B, by receivinbg a reward and updates the leaning policy.
- State: A state S is a collection of present options of all nodes. As the embedding τ_v has already has the option information, the state is represented as p-dimensional space, $\sum_{v \in V} \tau_v$.
- Advice: It includes the new (S,Adv) pair to the agent's dictionary and assigns a parameter that will define the action selection procedure to follow the new advice for deploying a classifier at the edge servers.

- Action: The action (v, i) is a 2-tuple, where v is a node of G without a classifier deployed and i is a classifier. The actions are denoted as their related p-dimensional node embedding τ_v. Initially, the action process checks whether the advice has lately been given and should still be followed. If the advice is being generalized through time and the new (state, advice) pair has not been included to the dictionary, it will be included at this time. When the state is revisited in the future, the recent advice given for a previous state will be applied as if it had been given for this state, too. The timer that keeps track of the friction parameter threshold is updated. When the timer indicates that the advice has been followed for long enough, parameters will be reset so the agent will return the Q-learning action for the next time step. When the advice has earlier been given for this state, the agent must select between advice and Q-learning suggestion. Because of the number of classifier is determined, the outcome of the Q-function is the collection of Q-value of every classifier.
- Transition: For a given action (v, i), the node v set to use the i-th classifier.
- Rewards: The improved deep reinforcement learning learns the greedy algorithm, so the reward function $r(S, Adv, v, i)$, that the action (v, i) is taken at state S based on advice Adv, is described the same way as it, i.e.,

$$r(S, Adv, v, i) = B\big(S \cup (Adv, v, i)\big) - B(S) \quad (3.7)$$

and $B(\emptyset) = \alpha B_1(S)$.

- Policy: The policy is based on Q - function which is a deterministic greedy policy $\pi(Adv, v, i|S) \coloneqq \arg min_{v \notin S,i}\, Q(S, Adv, v, i)$ will be used. The policy strategy in deep reinforcement learning is improved by engaging policy gradient with Q-learning. The estimation of Q using the policy is given as follows:

$$Q^{\pi}(S, Adv, v, i) = \alpha(\log \pi(S, Adv, v, i) + H^{\pi}(S)) + V(S) \quad (3.8)$$

In Eq. (3.8), V denotes the value of state S.

$$H^{\pi}(S) - \sum_{(v,i)} \pi(v, i) \log(S, Adv, v, i) \quad (3.9)$$

It is known that the fixed point the Bellman residual will be small for small α, the parameters Θ is updated to reduce the Bellman residual in a fashion similar to Q-learning i.e.,

$$\Delta\Theta \propto \underset{(S,Adv,v,i)}{\mathrm{E}} \big(\mathcal{T} * Q^{\pi}(S, Adv, v, i) - Q^{\pi}(S, Adv, v, i)\big) \nabla_{\Theta} \log \pi\,(S, Adv, v, i) \quad (3.10)$$

In Eq. (3.10), $\mathcal{T}$ denotes the Bellman operator. Selecting an action (Adv, v, i) relates to adding a node of G to the current partial solution and getting a reward $r(S, Adv, v, i)$. A n-step Q-leaning method is used to learn the approximate optimal solution under different traffic.

4. SIMULATION RESULT

A simulation program is designed to meet the experimental requirements for simulating a DDoS attack on a single target. It regards the internal processing of the router as a black box,only paying attention to the traffic size, omitting the specificdetails that do not impair the experimental evaluation. Specifically, the network is seen as a tree-like graph with each routeras a node. For each node, the size of the incoming and outgoing traffic and the properties of the classifier are considered. In the simulation program, leaf nodes generate traffic from external systems, and the traffic of other nodes flows from their children. The traffic which flows from the node is related to incoming traffic and the properties of the classifier. Here, the average traffic is considered instead of instantaneous rate. Each leaf node receives trafficat a constant rate, and the legal and attack traffic rates areevenly selected in [0; 30] and [30; 300]. The

effectiveness of the DRL and IRLDDS are tested in terms of approximation ratio and

4.1 APPROXIMATION RATIO

The approximation ratio measures the quality of optimal solution obtained by DRL and IRLDDS. A definition of approximation ratio of a solution Sol to a problem instance G is defined in the traffic state T is

$$A(G,T) = \frac{o(G,T)}{B(G,T)} \qquad (4.1)$$

In Eq. (4.1), $o(G,T)$ denotes the objective value and $B(G,T)$ denotes the best solution.

Table 4.1 shows the approximation ratio of DRL and IRLDDS for different number of nodes.

Table.4.1 Comparison of Approximation ratio

Number of nodes	DRL	IRLDDS
5	0.98	0.92
10	1	0.95
15	1.1	0.96
20	1.1	0.95
30	0.99	0.94
50	0.98	0.94

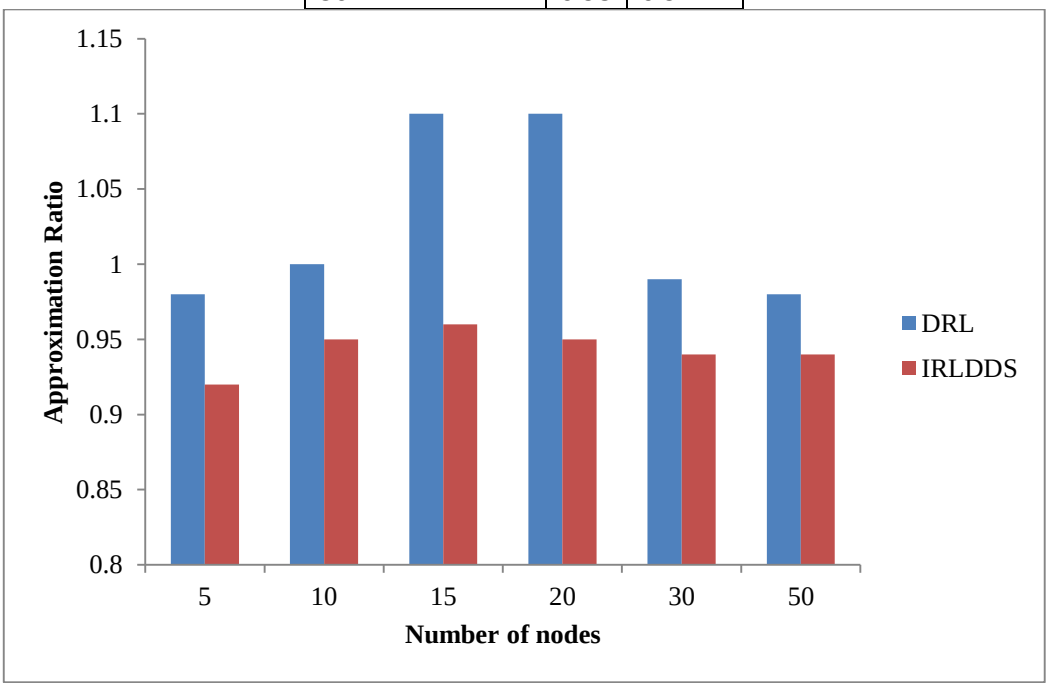

Figure.4.1 Comparison of Approximation ratio

Figure 4.1 shows the approximation ratio of DRL and IRLDDS schemes for different number of nodes. X axis denotes the number of nodes and Y axis denotes the approximation ratio. The approximation ratio of IRLDDS is 13.64% less than DRL to defend DDoS attack when the number of node 20. From this analysis, it is proved that the proposed IRLDDS scheme has low approximation ratio than DRL scheme for defending the DDoS attack in a network.

4.2 Running Time

Running time denotes the amount of time taken to defend the DDoS attack by DRL and IRLDDS schemes. Table 4.2 shows the running time of DRL and IRLDDS to defend the DDoS attack.

Table.4.2 Comparison of RunningTime

	DRL	IRLDDS

eISSN1303-5150

Running Time	20	14

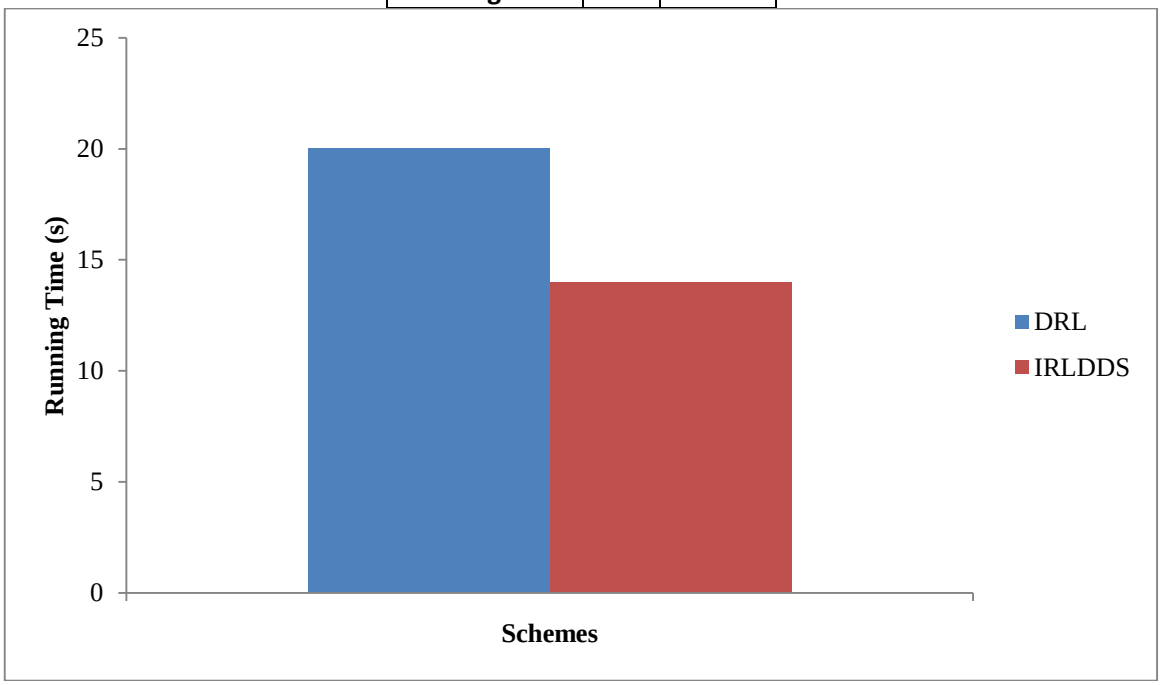

Figure.4.2 Comparison of Running Time

Figure 4.2 shows the running time of DRL and IRLDDS schemes for 250 nodes in a network. X axis denotes the schemes and Y axis denotes the running time in seconds. The running time of IRLDDS is 30% less than DRL for defending the DDoS attack. From this analysis, it is proved that the proposed IRLDDS scheme has less running time than DRL scheme for defending the DDoS attack in a network.

5. CONCLUSION

In this paper, an IRLDDS is proposed to defend DDoS attack in a network effectively. It handles the perceptual aliasing in deep reinforcement learning by using NAA and combined policy gradient with Q-learning. NAA gets an advice from an interaction to deploy a classifier at a node. The action i.e., the deployment of classifier is performed based on the advice. The combined policy gradient with deep reinforcement learning used a value function approximation with the policy which enables to process with continuous action spaces. Based on the Q values, the policy gradient deploys a classifier in the edges to defense the DDoS attack in the network. The simulation result prove that the proposed IRLDDS scheme has better approximation ratio and running time than DRL scheme for defending the DDoS attack in the network.

References

[1] Bhardwaj, A., Mangat, V., Vig, R., Halder, S., & Conti, M. (2021). Distributed denial of service attacks in cloud: State-of-the-art of scientific and commercial solutions. *Computer Science Review*, *39*, 100332.

[2] Manavi, M. T. (2018). Defense mechanisms against distributed denial of service attacks: a survey. *Computers & Electrical Engineering*, *72*, 26-38.

[3] Carlin, A., Hammoudeh, M., &Aldabbas, O. (2015). Defence for distributed denial of service attacks in cloud computing. *Procedia computer science*, *73*, 490-497.

[4] Tariq, U., Malik, Y., & Abdulrazak, B. (2012). Defense and monitoring model for distributed denial of service attacks. *Procedia Computer Science*, *10*, 1052-1056.

[5] Zhang, H., Hao, J., & Li, X. (2020). A method for deploying distributed denial of service attack defense strategies on edge servers using

reinforcement learning. *IEEE Access*, *8*, 78482-78491.