International Journal of Innovative Research and Scientific Studies, 9(2) 2026, pages: 35-52

ISSN: 2617-6548

URL: www.ijirss.com

Advanced EDoS eye: A pragmatic model to mitigate economic denial of sustainability attack in cloud system applying dynamic game-based decision module

Fahad Zaman Chowdhury[1], Md. Nasim Adnan[2], Md. Moradul Siddique[2,3*], Arshad Parvez[4], Mohd Yamani Idna Bin Idris[5]

[1]*Information and Communication Technology Department (ICTD), Bangladesh Bank, Dhaka-1000, Bangladesh.*
[2]*Department of Computer Science and Engineering, Jashore University of Science and Technol-ogy (JUST), Jashore-7408, Bangladesh.*
[3]*Department of Computer Science and Engineering, University of Information Technology & Sciences, Dhaka-1212, Bangladesh.*
[4]*Cloud Platform Business, Oracle Corporation, Singapore.*
[5]*Department of Computer System & Technology, Faculty of Computer Science and Information Technology, Universiti Malaya, Malaysia.*

Corresponding author: Md. Moradul Siddique (*Email: moradul_siddique@uits.edu.bd*)

Abstract

Low-rate stealthy Distributed Denial of Service (DDoS) attack results by affecting the cloud pricing model is a new form of attack called Economic Denial of Sustainability (EDoS). Existing mitigation models such as EDoS Shield, Controlled Access, and EDoS Eye can partially eliminate EDoS. Firstly, these schemes overlook the attacker's strategy profile, making their defensive system less resilient. Secondly, using predefined or static threshold values set by some schemes makes the defensive system vulnerable to more false alarms. Thirdly, they use authentication mechanisms like Graphical Turing Test, Crypto Puzzle, or similar graphic-based authentication mechanisms for each new client to identify botnets and filter out malicious traffic. These mechanisms contribute to higher response time. To successfully defend against EDoS attacks, particularly against clever rational attackers, a defender or defensive system needs to choose an optimal defense strategy considering the attacker's intent. Therefore, we propose Advanced EDoS Eye, a dynamic game theory-based model using non-cooperative, zero-sum multistage game theory in this paper. The proposed analytical Dynamic Game-based Decision Module (D-GBDM) in firewall instance generates dynamic threshold values, substantially reducing the EDoS effect's payoff in cloud computing. Overall, experimental results favour the proposed model by ensuring the Quality of Service (QoS) of the defense system in the cloud.

Keywords: Cloud computing, Game theory, Dynamic game-based decision module, Economic denial of sustainability, Nash equilibrium, Quality of service.

DOI: 10.53894/ijirss.v9i2.11243
Funding: This study received no specific financial support.
History: Received: 3 December 2025 / **Revised:** 19 January 2026 / **Accepted:** 23 January 2026 / **Published:** 13 February 2026

Competing Interests: The authors declare that they have no competing interests.
Authors' Contributions: All authors contributed equally to the conception and design of the study. All authors have read and agreed to the published version of the manuscript.
Transparency: The authors confirm that the manuscript is an honest, accurate, and transparent account of the study; that no vital features of the study have been omitted; and that any discrepancies from the study as planned have been explained. This study followed all ethical practices during writing.
Publisher: Innovative Research Publishing

1. Introduction

Cloud or utility computing is economically favorable over conventional hosting. The elasticity feature enables cloud services to allocate or deallocate cloud resources dynamically. Auto-scaling up or provisioning cloud resources is not a cost optimization technique but rather an operational requirement. In contrast, scaling down elasticity keeps the expenditure steady [1]. Moreover, enterprise-level consumers operating globally require faster end-to-end response time, which can be obtained by distributing request loads to multiple clouds in many locations [2]. In an ideal case, this new paradigm operates according to consumers' needs and demands. Thus, proper auto-scaling of cloud resources ensures end users a good experience and Quality of Service (QoS) [3].

However, both pay-as-you-go and auto-scaling features have some adverse effects. The Distributed Denial of Service (DDoS) attacker cannot turn down the cloud services due to its resource abundance, despite individual cloud consumers still being the victim of this attack [4]. An attacker in the form of DDoS floods a targeted server in the cloud with hundreds or thousands of zombie machines. As a result, these overloaded illegitimate requests trigger the auto-scaling feature of the cloud to fulfil the operational demand. The effect of such high-scale provisioning ends with an unexpectedly large bill to the victim cloud consumer due to the pay-as-you-go feature. This emerging threat is known as the Economic Denial of Sustainability (EDoS) in the literature [5]. EDoS is a kind of DDoS that manipulates vulnerable pricing models in cloud computing [6]. The enormous financial damage caused by EDoS attacks may lead many organizations towards the withdrawal of cloud services or, in the worst case, bankruptcy. It is a subtle type of attack that mostly remains unnoticed.

Therefore, existing solutions for EDoS [3, 7-17] are still in the early stages. EDoS-Shield [8, 17, 18] uses the Graphical Turing Test, a similar authentication approach, to differentiate humans from bots. The model separates humans from bots with the presumption that all EDoS attackers are bots and all humans are valid users. This predefined idea makes the model partially successful. Firstly, an EDoS attacker launches an attack strategically as well as intelligently. Hence, consideration of the attacker's profile is crucial. Secondly, if an EDoS attacker exploits the illegitimate traffic inside the cloud, how will the cloud firewall respond, and how will the firewall act with rate limiting technique? If the rate limiting technique is static, how will it detect low-rate EDoS traffic or dynamic EDoS traffic generated by the attacker, and how will cloud services ensure QoS while mitigation occurs? Thirdly, use of the Graphical Turing Test or Crypto Puzzle increases the computational time. In addition, it may also create problems for disabled and visually impaired people. Besides, the cloud itself has an authentication mechanism for some services. As a result, it is a question about how a different authentication process would integrate with the existing authentication mechanism. The literature on EDoS mitigation models needs to address these fundamental questions.

Recently, game theory has been used to mitigate DDoS attacks and was effective in some cases [19-21]. Wang, et al. [22] proposed a model based on dynamic game theory to defend botnet. Game theory is a series of analytical tools to design and formulate interaction between multiple players competing under certain conditions. A game consists of a set of strategies where the total plan of all possible actions a player can take. Strategies can be pure or mixed. A pure strategic game is like a single-shot static game where a player chooses the strategy initially and cannot alter it until the end. On the other hand, a mixed strategy or dynamic game consists of multiple stages where players can change their strategies at any point. The outcomes of the game depend on the best possible strategies taken by each player. Nash Equilibrium is one such solution that represents the steady state of the game. Nash Equilibrium can be found in a non-cooperative game involving at least two players. No player can benefit more by deviating from a state, assuming the opponent's state remains unchanged. The game can be either a zero-sum game or a non-zero-sum game. One important topic of the zero-sum game is finding the saddle point. The saddle point theorem is a single or multiple equilibrium pair(s) of strategies in a game matrix. In each equilibrium pair of the saddle point, row and column coexist in the exact location where the former has a minimum value, and the latter has a maximum value.

To the best of our knowledge, EDoS Eye Chowdhury, et al. [23] is the first game theory-based approach in literature for EDoS mitigation. Analyzing the behavior or strategic mindset is a crucial factor in real-life scenarios. In doing so, EDoS Eye analyses the attacker profile effectively by finding the optimal strategy for the defender in a zero-sum, non-cooperative static game. The optimal threshold value is obtained through Nash Equilibrium which helps reducing the EDoS attacker's payoff. In this model, all the traffic flows are determined using a primary controller named Game-based Decision Module (GBDM) which is embedded in the cloud firewall. The model also incorporates a honeypot to minimize the EDoS effect further. However, in a real-life scenario, the actions of rational attackers and rational defenders are not static. In other words, a single-shot static game is impractical in a real-life conflict between attackers and defenders. As the set of actions changes over time, dynamic game modelling is better suited for a real-life conflict. This phenomenon is primarily attributed to improving the existing Game-based Dynamic Module (GBDM) in EDoS Eye. In addition, EDoS Eye does not evaluate the response time and other parameters to ensure QoS in the cloud. The static game also limits the real-time evaluation of attackers' and defenders' strategic profiles.

In Lalropuia and Khaitan [24] the authors proposed a novel game theoretic model for interactions between EDoS attackers and defenders. The authors derive the best defense strategies, responding to EDoS attacks by computing Perfect Bayesian Nash Equilibrium (PBE), including the pooling PBE, separating PBE, and mixed strategy PBE. The model provides insights into the dynamics of EDoS attacks and offers strategies for defenders to optimize their payoff. However, due to its overly simplified approach, the model may not capture all the complexities and nuances of real-world EDoS attacks.

This paper contains six (06) sections. The architecture of the proposed Advanced EDoS Eye is presented in Section 2. Section 3 describes the game-based analysis of the Advanced EDoS Eye in two phases. Simulation results are depicted in Section 4. Section 5 demonstrates the validation part of the proposed model. Finally, the discussion and concluding remarks are presented in Section 6.

2. The Proposed Model: Advanced EDoS Eye

Figure 1 illustrates the overall architecture of the proposed Advanced EDoS Eye. The aggregate traffic flows are distributed through the load balancer. After packet inspection in firewall instances, filtered traffic reaches the target *VM*s or application instances. The communication link between the firewall and target *VM*s is prone to congestion. The topology is similar to the EDoS Eye model [23] except that it replaces the Game-based Decision Module (GBDM) with a more advanced D-GBDM module. GBDM is a static game theory-based decision module to route the traffic. On the other hand, D-GBDM is a dynamic game-theory-based decision module to route the traffic, which changes over time with respect to EDoS attacker strategy. D-GBDM identifies both the attacker and defender's optimal strategies through saddle points. Similar to the EDoS Eye model [23] a honeypot is also incorporated in the proposed model. Researchers [25-27] consider honeypots as an effective supplementary strategy in advanced intrusion detection systems, which can keep the false rate to a minimum.

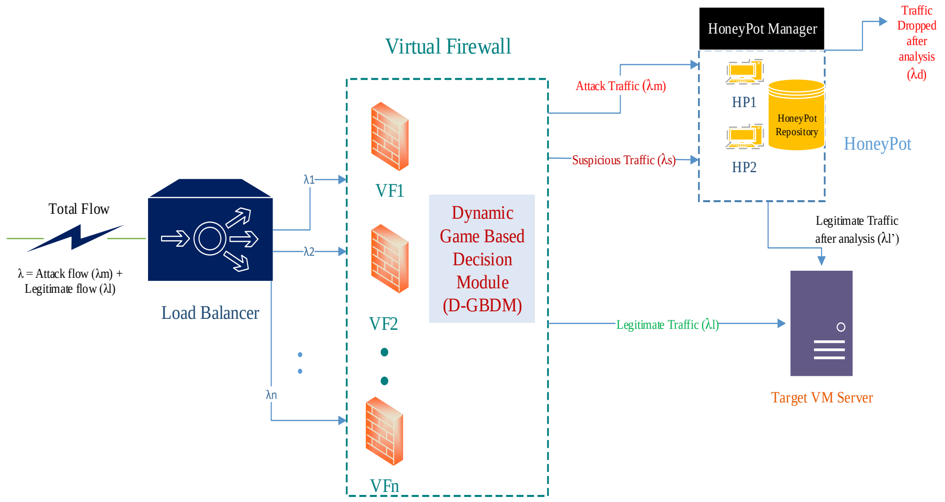

Figure 1
Topology for the proposed Advanced EDoS Eye.

2.1. Underlying Assumptions for the Traffic Flow of Advanced EDoS Eye

We propose the Advanced EDoS Eye model considering the following assumptions:

1. Attacking Nodes/*VMs*/Bots are controlled by a single attacker.
2. All the traffic per Node/*VM*/Bot represents a unique flow.
3. Unlimited Bandwidth (*BW*) is available in the cloud, especially the *BW* between edge and firewall instances.
4. Any side channel or co-resident attack is out of the scope of this research work.
5. Both valid and malicious traffic generate *UDP* traffic. i.e., we are considering application-level traffic only.
6. All incoming traffic, whether valid or malicious, enters the cloud data center following Poisson distribution.

2.2. Traffic Flow Analysis and Its Functionality

The inbound traffic flows in cloud computing typically follow the Poisson distribution [28]. In the experiment, it is considered only application-level traffic. The total flow (λ) distributed to Load Balancer (*LB*) such as $\lambda = \lambda_1, \lambda_2, \ldots, \lambda_n$ to numerous virtual firewall instances (VF_1, VF_2, …, VF_n). We introduce Dynamic Game-based Decision Module (D-GBDM) in firewall instances as shown in Figure 1. D-GBDM generates attack and optimal threshold values k_1 and k_2 respectively to limit the rate of incoming traffic. These threshold values determine traffic that is classified as legitimate (λ_l), attack (λ_m) and suspicious (λ_s). If the flow is below the optimal threshold value (k_2) then it will be directed to the target virtual instance as a legitimate flow (λ_m). Conversely, attack and suspicious traffic flows are redirected to Honeypot (*HP*) for further analysis.

Honeypot instances (*HP*$_1$, *HP*$_2$) inquire about suspicious traffic by collecting logs, including source and destination addresses, timestamps, and other identifiable attributes.

Moreover, it preserves the malicious activity information repository in Honeypot. Honeypot Manager decides which traffic will be dropped and which traffic will be redirected to destination instances i.e., target *VM*, based upon matching the traffic pattern with the repository. Usually, traffic directed to honeypots is malicious in nature [25, 26, 29]. Honeypot is designed so that attackers can conveniently exploit the vulnerabilities of this device by misunderstanding it as their target *VM* and later fall into it as a trap. As a result, malicious traffic information can easily be captured by Honeypot.

The functionality of the proposed Advanced EDoS Eye is divided into the following four processes: P_1- P_4.

P_1: Firewall Instances (*VF*s) deal with all the incoming traffic from *LB* and filter portion of (*TCP* + *UDP*) traffic based on D-GBDM generated thresholds (k_1 and k_2). D-GBDM classifies the incoming traffic such as legitimate, attack and suspicious.

P_2: If total incoming traffic (λ) < optimal threshold (k_2), then the traffic is considered legitimate (λ_m) and is directed to the target instance, i.e., *VM*. If $\lambda > k_2$ and $\lambda >$ attack threshold (k_1), then it is treated as attack traffic (λ_m). In this case, this traffic is sent to Honeypot (*HP*$_1$) for further analysis. The Honeypot repository records the attack information source for future analysis or investigation. In the proposed model, virtual Honeypot instances [30] are considered to enable the auto-scale feature to maintain the same instances before getting corrupted. Once information is collected, the *HP* manager drops the malicious traffic.

P_3: If incoming traffic (λ) > k_2 and < k_1, then D-GBDM classifies it as suspicious traffic (λ_s). It will redirect λ_s towards Honeypot (*HP*$_2$) to observe if any matching can be found with the existing attack repository.

P_4: If the suspicious traffic λ_s matches Attack Nature's (*AN*) characteristics, it will be eliminated. Otherwise, it will pass to the target instance (*VM*) as legitimate traffic (λ_l).

3. Dynamic Game Modelling

We now present the analytical framework of the Dynamic Game-based Decision Module (D-GBDM) as a key part of the proposed Advanced EDoS Eye. Firstly, D-GBDM demonstrated with formal analysis under different game scenarios, i.e., single shot static game and multistage dynamic game. In the later part of the section, the response time of the Advanced EDoS Eye has been analyzed for performance evaluation.

3.1. Dynamic Game-based Decision Module (D-GBDM) Analysis

In the proposed model, we assume the attacker is always rational, and the defender is either irrational or rational. To demonstrate the effectiveness of the D-GBDM in contrast to the GBDM, we stage a single-shot static game in Scenario 1. The focused scenario is a multi-stage dynamic game presented in Scenario 2. In a multistage dynamic game, both players exhibit maximum efficiency in dominating each other. However, firstly, we introduce the players as follows.

Rational Attacker: The rational attacker is the one who adopts her strategy to evade intrusion detection smartly so that it can inject malicious EDoS traffic to exploit the cloud utility model. The attacker has at least some knowledge of the defensive mechanism. Rational attacker always intends to maximize their payoff or incentives.

Irrational Defender: The irrational defender is indifferent toward the attacker's strategy. For example, one employs her maximum resources while defending an intermediate attacker or invests low resources while defending against full strength attacker without knowing the attacker's strategy.

Rational Defender: A rational defender can smartly adjust her defensive strategies based on the intensity of attack strength. The defender is dynamic; she can change her defense policies at any game stage. The defender has at least some knowledge of the attacker's tactic. The objective of the rational defender is to achieve the highest possible payoff throughout the game.

The proposed model considers a simple M/M/S queue best suited to the cloud. All the incoming traffic in the cloud follows Poisson distribution in general [31].

$$P_r\,[x = K] = \frac{\lambda^K e^{-\lambda}}{K!},\ K = 0, 1, 2, \dots \tag{1}$$

Equation 1 expresses the probability of K arrival for a specified interval. λ is the aggregate incoming traffic arrival rate.

S represents the total number of instances in the cloud. Since traffic of each instance will be distributed evenly in the load balancer, each instance receives $\lambda_i = \frac{\lambda}{S}$ traffic of i-th instance.

$$\text{The mean cloud utility per instance, } U = \frac{\sum_i^S \frac{\lambda_i}{S\mu}}{S} = \frac{\lambda}{S\mu} \tag{2}$$

where μ is the average service rate of the cloud. Hence, utility rate or busy rate can be derived as

$$\rho = \frac{\lambda}{\mu} \tag{3}$$

For the stability of the system, keeping $\rho < 1$ must be ensured. During an EDoS attack, the total incoming flow rate will become $\lambda = \lambda_l + \lambda_{m.,}$ where λ_l is legitimate traffic and λ_m is malicious traffic.

Hence, in the attack phase, the average computing resources utility on S active instances is defined as follows.

$$U = \frac{\lambda l + \lambda m}{S\mu} = U_l + U_m \tag{4}$$

The utility resources in the cloud solely exploited by the attacker will be.

$$U_m = \frac{\lambda m}{S\mu} \tag{5}$$

According to Wang, et al. [32]. EDoS falls into the amplification attack class category where the attacker invests fewer resources to maximize the damages of the target victim's resources. Since, in EDoS, the attacker's main goal is to impose a financial burden on the victim. This can be quantified as follows.

$$\text{Economic Amplification factor } (EAF) = \frac{Cost_{target}\,\lambda_{target}}{Cost_{attack}\,\lambda_{attack}} \qquad (6)$$

Where Bandwidth Amplification Factor (BAF) = $\frac{\lambda_{target}}{\lambda_{attack}}$.

In an amplification attack, attacker-generated payloads are usually lesser than those received at the victim's end. For instance, an attacker sends simple DNS look-up messages to a public DNS server using the victim's spoofed IP address, resulting in an overflow of DNS response traffic at the server end [32]. In this scenario, the attacker's payload packets are much lower compared to the victim's received payload packets.

$$\text{So, here, } \lambda_{target} = \lambda_m, \text{ which results in } EAF = \frac{Cost_{target}\,\lambda_m}{Cost_{attack}\,\lambda_{attack}} \qquad (7)$$

For a rational attacker $\frac{Cost_{target}}{Cost_{attck}} > 1$ and $\frac{\lambda_m}{\lambda_{attack}} > 1$, Hence, EAF > 1

Attacker's Dilemma*: The attacker deploys v number of nodes/instances to execute an EDoS attack. If v is high, the attacker conveniently maintains a lower attack flow rate (λ_m), leading to higher payoff or maximizing the damage to its victim. However, deploying multiple nodes/instances is expensive. Conversely, if the attacker plans to deploy fewer nodes/instances with a higher attack flow rate, the firewall will easily block it due to the rate limit. This will reduce attacker payoff.*

Defender's Dilemma: The defender needs to fix a certain threshold value to rate limit the traffic. If she sets a low threshold value (k) in the firewall, most traffic, including legitimate traffic, will be barred. This will result in more false positives. Conversely, setting a high threshold value (k) permits illegitimate traffic to pass into the firewall, resulting in more false negatives. Therefore, choosing an optimal threshold value (kopt) is essential to minimize the attacker payoff. Adjusting threshold (k) at different points of the game based on various traffic rates is even more challenging for the defender.

Consider the attacker investing v nodes to send EDoS traffic in the proposed game model. At the same time, legitimate users send u number of traffic requests. Hence, Equations 4 and 5 can be rewritten as follows.

$$U = \frac{u\,\lambda l + v\,\lambda m}{S\mu} \qquad (8)$$

$$U_m = \frac{v\,\lambda m}{S\mu} \qquad (9)$$

3.2. Scenario 1- Single shot static game

Scenario 1 is a static zero-sum game where players involved in the game, once they decide on a strategy, cannot alter it till the end of the game.

The attacker's objective is to exploit the cloud utility resources, i.e., an increase of U_m and the increase of the Economic Amplification Factor (EAF). The attacker's primary goal is to maximize the payoff while using the least number of nodes v. Hence, the attacker's cumulative payoff is quantified as follows.

$$C_A = W_m\,U_{m\,(nd)} + W_{EAF}\,EAF_{(nd)} - W_v\,v_{\,(nd)} \qquad (10)$$

Since it is a zero-sum game, the defender's cumulative payoff is exactly opposite the attacker's cumulative payoff.

$$C_D = -W_m\,U_{m\,(nd)} - W_{EAF}\,EAF_{\,(nd)} + W_v\,v_{\,(nd)} \qquad (11)$$

Here, W_m, W_{EAF}, and W_v are the weight coefficients of the players involved in the game. The suffix *nd* expresses no defense mode.

Considering the defense mode in the existing EDoS Eye [23]. The attacker's payoff is defined as follows.

$$C_A^d = W_m^d\,U_m^d + W_{EAF}^d\,EAF^d - W_v^d\,v^d - W_H^d\,H^d \qquad (12)$$

The defender payoff is defined as follows.

$$C_D^d = -W_m^d\,U_m^d - W_{EAF}^d\,EAF^d + W_v^d\,v^d + W_H^d\,H^d \qquad (13)$$

W_m^d, W_{EAF}^d, W_v^d, and W_H^d are the weight co-efficient of players involved from both parties in defense mode. H^d is the portion of traffic which goes to the honeypot.

The attacker aims to find the optimal value of λ_m and v. *In contrast, the defender's goal is to find the optimal value of the threshold (*k1, k2*) to maximize the expected payoff. The Nash Equilibrium represents the pair of strategies that give the maximum set of outcomes for both players unless one of the players deviates from her payoff matrix. Hence, in the EDoS Eye* Chowdhury, et al. [23] *i.e., GBDM, Nash Equilibrium is expressed as follows.*

.
$$C_A(v^*, \lambda m^*, k1^*, k2^*) \geq C_A(v, \lambda m, k1^*, k2^*) \quad \forall\, v, \lambda m \qquad (14)$$

$$C_D(v^*, \lambda m^*, k1^*, k2^*) \geq C_D(v^*, \lambda m^*, k1, k2) \quad \forall\, k1, k2 \qquad (15)$$

3.3. Scenario 2 - Multistage Dynamic Game

In real-life scenarios, the actions of rational attackers and rational defenders are not static. The set of actions changes with time. Thus, dynamic game modelling better suits the reality of battling against the attacker. We assume both the players are either rational or irrational. This means at a specific time; an attacker can set a rational action plan against an irrational action of a defender or vice versa. Furthermore, there might be a case where both the players' actions or strategies are simultaneously rational or irrational. Dynamic strategy deals with the change of movement of players from one stage to another in a game. This modelling allows us to dynamically add, enable, disable, and detach a security measure. It also permits reconfiguring defensive mechanisms by replacing older one's [33] according to Cao, et al. [33] Some inherent properties of the game are as follows.

1. Intentional Attack Property: In general, attackers are not random. Instead, it is driven by the attacker's intents and objectives.
2. Strategy-Interdependency Property: The effectiveness of the attack depends on how secured or protected the system is. The system's quality can only be assessed when the system is under attack.
3. Uncertainty Property: Both the attacker and defender have incomplete information about each other.

Generally, the dynamic game model is formed based on two crucial factors, i.e., state and time. The game is a battle between attacker and defender consisting of an interleaved sequence of states with corresponding actions, as shown in Figure 2.

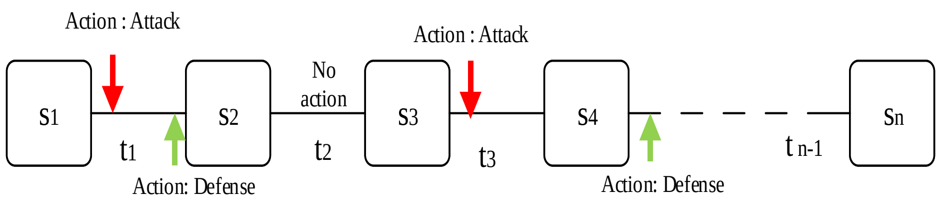

Figure 2.
States, intervals, and actions in a dynamic game.

Let state consists of a series of n number of finite smaller sub-states such as State = {*s1, s2….sn*}. The interval between two adjacent states is t_i, where t_i = {*t1, t2……tn-1*}. Only one action from each player can occur between adjacent states. Actions, each from attacker and defender, can also be allowed in the same interval between adjacent states. Moreover, an actionless interval between two states is permissible since it is not considered any legitimate traffic flow as an action. In this game, both the players have the following actions.

The attacker dynamically adjusts the number of nodes (v) and traffic rate (λm). When increasing the number of nodes, a rational attacker usually decreases the traffic rate or vice versa.

Defender can dynamically adjust the minimum rate limiting threshold parameter *k2*, i.e., increase or decrease *k2* during a game. In addition, the rational defender also applies her prediction, which is a knowledge-based probabilistic action to use appropriate steps. This prediction is a subjective parameter. In this simplified model, we quantify this parameter as $\mho$ with the name "Prediction". $\mho$ ranges from 0 to 1, where a higher value indicates more predictive to recognize the attack. Here, the defender also means the defense system or part of the defense system, such as the honeypot. Prediction $\mho$ is a parameter that follows the rule "observe then act".

In the proposed dynamic game model, the attacker's possible action set is *A* = {*A1, A2, A3, A4*} where,
A1 = Increase node (v) and increase traffic rate (λ_m)
A2 = Increase node (v) and decrease traffic rate (λ_m)
A3 = Decrease node (v) and increase traffic rate (λ_m)
A4 = Decrease node (v) and decrease traffic rate (λ_m)
Defender's possible set of action include *D* = {*D1, D2, D3, D4*} where,
D1 = Increase threshold (*k2*) without Prediction ($\mho$)
D2 = Decrease threshold (*k2*) without Prediction ($\mho$)
D3 = Increase threshold (*k2*) with Prediction ($\mho$)
D4 = Decrease threshold (*k2*) with Prediction ($\mho$)
Thus, the dynamic game matrix will be as follows:

Table 1.
Different Strategy pair matrix of attacker and defender.

Attacker (A) / Defender (D)	**D1**	**D2**	**D3**	**D4**
A1	A1D1	A1D2	A1D3	A1D4
A2	A2D1	A2D2	A2D3	A2D4
A3	A3D1	A3D2	A3D3	A3D4
A4	A4D1	A4D2	A4D3	A4D4

Table 1 demonstrates that 16 possible strategy pairs exist in this model. In each strategy pair, attack and defense strategy co-exist. Recall from the proposed analytical game model that the attacker's strategy parameter in a strategic action is *Ai* (v, λ_m), like the static model. Defender's strategic action is *Di* (*k2*, $\mho$). The new parameter $\mho$ in a defensive strategy can only be applied to rational defenders. For irrational defenders, $\mho=0$. The static game model presented earlier, $\mho$, is absent because of the irrational defender. Unlike the static model of GBDM, the D-GBDM works in the following scenario, as presented in Figure 3.

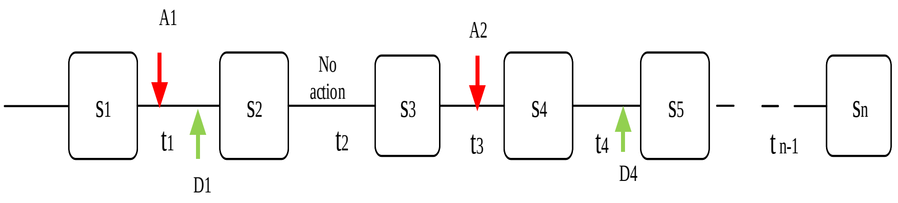

Figure 3.
A simple scenario of a dynamic game.

At interval t_1, the attacker launches an attack with the *A1* strategy, which implies the attack performs with an increasing number of nodes as well as an increasing rate of traffic. The defender takes prompt action with the *D1* strategy by seeing the attack traffic in the same interval. *D1* strategy implies increasing the threshold value to block the traffic but not analyzing the traffic nature, whether it is malicious or legitimate. So, at the t_1 interval, the strategy pair of attacker and defender is *A1D1*. An intelligent or rational attacker can observe the defensive measure after a certain interval. She decides to move her strategy from *A1* to *A2*, which suggests decreasing the traffic rate while nodes remain the same. As the defender sets a high threshold value of *k2*, the attacker intends to bypass the malicious traffic by lowering the traffic rate. The effect can be seen at interval t_3 between adjacent states *s3* and *s4*. If the defender acts rationally, she will follow the observe-then-act strategy. In the next interval, i.e., the t_4 defender switches to the *D4* strategy. *D4* implies a decrease in the threshold value *k2* with prediction $\mathcal{E}$. A good prediction $\mathcal{E}$ can differentiate malicious traffic from legitimate traffic by matching the characteristic fields of attack traffic such as source and destination address, traffic rate, traffic pattern, timestamp, etc. In this way, a rational defender can significantly reduce the false rate by taking the appropriate strategy as a reaction to the attacker's strategy.

The players' payoff in any stage s_i of this dynamic game is like their static game payoff presented in Equations 12 and 13 with slight modification. With the new parameter $\mathcal{E}$, the payoff for the dynamic game at any stage can be represented.

$$C_A{}^d(s_i) = W_m{}^d U_m{}^d + W_{EAF}{}^d EAF^d - W_v{}^d v^d - W_H{}^d H^d - W_{\mathcal{E}}{}^d \mathcal{E}^d \quad (16)$$

$$C_D{}^d(s_i) = -W_m{}^d U_m{}^d - W_{EAF}{}^d EAF^d + W_v{}^d v^d + W_H{}^d H^d + W_{\mathcal{E}}{}^d \mathcal{E}^d \quad (17)$$

Here, $W_{\mathcal{E}}$ is the weight of prediction $\mathcal{E}$.

The sub-game Nash Equilibrium exists in each stage of the game, representing the proposed D-GBDM. These equations are the ultimate game-based strategic outcome of the proposed Advanced EDoS Eye.

$$C_A s_i\,(v^*, \lambda m^*, k2^*, \mathcal{E}^*) \geq C_A s_i\,(v, \lambda m, k2^*, \mathcal{E}^*) \quad \forall\, v, \lambda m \quad (18)$$

$$C_D s_i\,(v^*, \lambda m^*, k2^*, \mathcal{E}^*) \geq C_D s_i\,(v^*, \lambda m^*, k2, \mathcal{E}) \quad \forall\, k2, \mathcal{E} \quad (19)$$

3.4. Response Time Analysis for Performance Measurement

End-to-end response time considered a crucial factor in measuring Quality of Service (QoS) should conform with Service Level Agreement (SLA). We consider the M/M/S queue in a finite space. Client and server requests follow the First Come First Serve rule (FCFS). In Al-Haidari, et al. [17] and Chandy and Sauer [34] the authors computed the average response time in a virtual cloud firewall using the M/M/s queuing model.

In the proposed model, after the even distribution in Load Balancer (*LB*), the individual arrival rate of each instance *i*, $\lambda i = \frac{\lambda na}{S}$, where λ_{na} is the total arrival rate in the non-attack case. $\lambda_{na} = u\, \lambda l$.

Assume μ is the identical service rate of each instance *i*. So, the average delay for each instance in the firewall is $\tau = \frac{1}{\mu - \lambda_i}$

The total number of service requests in S instances are $N = \sum_{i=1}^{S} \lambda_i\ \tau = \lambda na\ \tau$

Little's theorem yields the long-term response time, $Rest = \frac{N}{\lambda na} = \frac{1}{\mu - \lambda_i}$

So, the mean response time of a group of *S* instances in *VF* is as follows.

$$\overline{Res_{t\ vf}} = \frac{S}{S\mu - \lambda_{na}} \quad (20)$$

During an EDoS attack, total arrival traffic λ comprised of both legitimate traffic λ_l and attack traffic λ_m such as $\lambda = \lambda_l + \lambda_m$. If the attacker uses *v* nodes and legitimate users send *u* number of requests, then total aggregate traffic in an attack case will be $\lambda_{total} = u\lambda_l + v\lambda_m$. For simplicity, we assume, $\lambda_{total} = \psi\lambda$, where ψ is the attack strength and $\psi > 1$. Thus, the mean response time in *VF* during attack is as follows.

$$\overline{Res_{ta\ vf}} = \frac{S}{S\mu - \psi\lambda} \quad (21)$$

3.5. Response Time with the Defense Models

In Chandy and Sauer [34] the authors evaluated the response time of EDoS-Shield - a well-known EDoS mitigation model, using the M/M/1 queuing model. However, a major flaw has been observed in their computational process. Their analytical model did not consider the response time taken in verification nodes (V-nodes), which is a considerable time-intensive step. We incorporate this additional step of EDoS shield for the computation of overall response time using a similar approach and data used in Chandy and Sauer [34]. The objective is to make a fair comparison with the proposed model. The decomposition method Chandy and Sauer [34] is used in calculating the total response time where a system is segmented into subsystems.

The total end-to-end response time taken in EDoS-Shield architecture, including the time taken in V-nodes, is as follows.

$$RT_{EDoS\text{-}Shield} = (\overline{Res_{ta\ vf}} + \overline{Res_{ta\ V-nodes}}) + Link\_delay * P_a + Cloud\_end\_delay * P_a \quad (22)$$

where, $\overline{Res_{ta\ vf}} = \frac{S1}{S1\mu - \psi\lambda}$ is the mean response time taken in virtual firewall instances. S_1 is the number of virtual firewall instances.

$\overline{Res_{ta\ V-nodes}} = \frac{S2}{S2\mu - \psi\lambda}$ is the mean response time taken in verifier nodes (V-nodes). S_2 is the number of virtual instances used in V-nodes. Link delay is due to a congested link between the virtual firewall and the cloud end. Link delay is computed as below.

$Link\_delay = \frac{1-\frac{\lambda_a}{\mu_{link}}}{\mu_{link} - \lambda_a}$ where λa is the mean incoming rate after passing the virtual firewall and $\mu link$ is the link capacity between *VF* and Cloud end.

$Cloud\_end\_delay = \frac{S}{S\mu 2 - \lambda_a}$, $\mu 2$ is service rate in cloud end. *Pa* is the probability of reaching the traffic in the cloud end. Hence, the Equation 22 can be rewritten as follows.

$$RT_{EDoS\text{-}Shield} = \left(\frac{S1}{S1\mu - \psi\lambda} + \frac{S2}{S2\mu - \psi\lambda}\right) + \frac{1-\frac{\lambda_a}{\mu_{link}}}{\mu_{link} - \lambda_a} * P_a + \frac{S}{S\mu 2 - \lambda_a} * P_a \quad (23)$$

In the proposed model, the total end-to-end response time can be computed as

$$RT_{EDoS\text{-}Eye} = \overline{Res_{ta\ vf}} + \overline{Res_{ta\ Honeypot}} * P_r + Link\_delay * P_a + Cloud\_end\_delay * P_a \quad (24)$$

where, $\overline{Res_{ta\ Honeypot}} = \frac{S3}{S3\mu 3 - \psi\lambda}$, *S3* represents the number of virtual instances of honeypot and $\mu 3$ is the service rate at honeypot. *Pr* is the probability of traffic redirecting towards the honeypot from *VF*. Thus, Equation 25 represents the proposed defensive model's overall response time or latency.

$$RT_{Advanced\ EDoS\text{-}Eye} = \frac{S1}{S1\mu - \psi\lambda} + \frac{S3}{S3\mu 3 - \psi\lambda} * P_r + \frac{1-\frac{\lambda_a}{\mu_{link}}}{\mu_{link} - \lambda_a} * P_a + \frac{S}{S\mu 2 - \lambda_a} * P_a \quad (25)$$

4. Model Analysis with Simulation Outcome

4.1. Payoff and Threshold Evaluation in Game Model

The proposed model is applied in a testbed where computation and simulation are conducted using the Wolfram Mathematica 11.0 version. In defense mode, we first present Scenario 1 i.e., EDOS Eye of the previous section where the game is static. Equation 12 represents the total payoff of the attacker in defense mode for EDoS Eye. Equation 12 comprises the probabilities of allowing, dropping and redirecting traffic Chowdhury, et al. [23] which can be rewritten as follows.

$$C_A{}^{d} = W_m{}^{d} \frac{v\lambda m}{S\mu} P_a(k1, k2, v, \lambda m) + W_{EAF}{}^{d} \frac{Cost_{target}\,\lambda_m}{Cost_{attck}\,\lambda_{attack}} P_a(k1, k2, v, \lambda m) - W_v{}^{d} v^{d} - W_H{}^{d} v\,\lambda m\, P_r(k1, k2, v, \lambda m)$$

In EDoS Eye, the game is a single shot static game, and hence the traffic (λ_m) is constant, which implies the attacker only needs to manage several attacking nodes *v*. Nodes number selection and their deployment should be strategic so that the attacker can obtain maximum payoff through it. On the other hand, the defender only needs to adjust the threshold values *k1* and *k2* as a defense filter in the firewall. For simplification, let $k1 = 1.25\ k2$. It implies the defender only requires adjusting the value of *k2* in the model. We also set the weight of the honeypot, $W_H{}^d$=2. The payoff is computed based on the probability of traffic distribution according to the EDoS Eye model [23]. In this experiment, we set the limit of $k2$ = 60 Kbps and $k1 = k2$*1.25 = 75 Kbps. Traffic below or equal to *k2* will pass the firewall and reach the target server hosted in the cloud. If traffic lies between *k1* and *k2*, it will be redirected to the honeypot for deeper inspection. If $\lambda_m > k2$, then it will block this traffic. Noted that the defender can tune the threshold value *k2* dynamically. However, the defender can only set the threshold value once per game. Later, the defender will adjust threshold settings dynamically in Scenario 2, a simulation where both the players are rational in a multistage dynamic game.

In the experiment in Figure 4 if the attacker exposes all nodes (*v*) belonging to her, i.e., 80, with a maximum allowable traffic rate of 60 for this case, the receivable payoff is 6200. This is the maximum payoff. However, exposing all the nodes in an EDoS attack using a single move is not a wise choice. There is an elevated risk of losing all the investing resources of the attacker if the defender identifies and blocks it. Even a minor deviation from this stage can cause greater loss. For example, if the traffic rate is 61, the payoff will sharply fall into -1138 with these attack nodes, as the data indicates in Figure 4. The attacker's smartest move should rely on Nash Equilibrium, a pair of strategies presented in Equations 14 and 15. In this scenario, it is observed that Nash Equilibrium exists. In a zero-sum game, one way to evaluate Nash Equilibrium is to determine saddle points from a series of coordinate values.

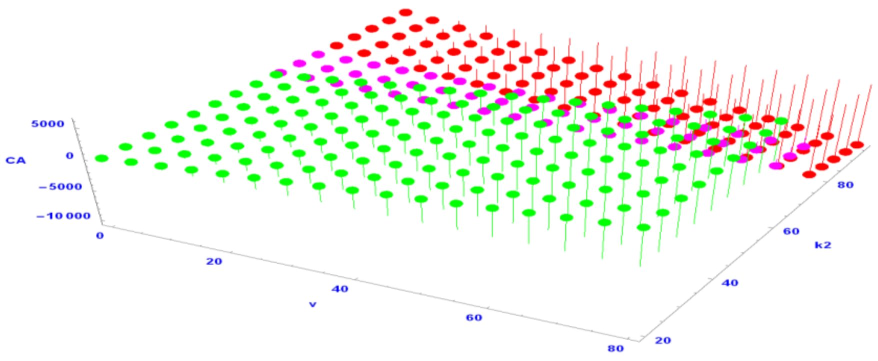

Figure 4.
Attacker payoffs for different threshold values (*k2*).

An essential condition for saddle point is the presence of payoff matrix elements. Among these elements, at least one must simultaneously contain the minimum of its row and the maximum of its column. A game might have one or more saddle points, but each must possess the same value. It is observed in Figure 5 indicated in red mark. The saddle point detects the coordinate (5, 61, -71.125), which signifies the optimal strategy of the attacker. With firewall setting threshold value (*k2*) 61, the attacker's best strategy is to invest in 5 nodes if she executes an attack with equal or slightly below this traffic rate. This gives the attacker a payoff of -71.125. In other words, if the defender sets the threshold value 61, she will get the payoff exactly opposite to the attacker, i.e., 71.125. At this stage, the attacker's deviation from this strategy will always give her a lower payoff, assuming the defender's move remains unchanged.

The proposed model illustrates a dynamic move of the attacker from *A1* to *A2* (as shown in Section 3), which means the attacker first starts the attack with an increasing number of nodes *v* as well as a higher rate of traffic λm. Defenders then choose strategy *D1*, i.e., increase the threshold value *k2* to block traffic. After *D1*, the attacker switches her move to *A2*, indicating a lower traffic rate while remaining the same attacking nodes. However, the defender sticks to the same strategy as a static game. Figure 6 depicts the attacker's payoff in this situation.

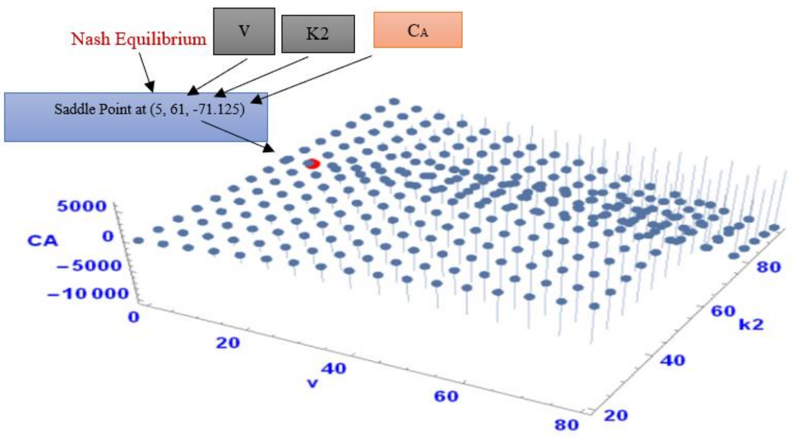

Figure 5.
The existence of saddle point (5, 61, -71.125) as Nash Equilibrium indicates in red point among attacker's payoff (*CA*) distribution.

Figure 6 shows that red marks indicate the dropped traffic the defender blocks by setting a higher threshold value. The attacker's wise move to change the traffic rate from high to low evades this threshold value and successfully bypasses the firewall as the defender sticks to the same strategy. Green marks indicate allowable traffic, which maximizes the payoff of the attacker.

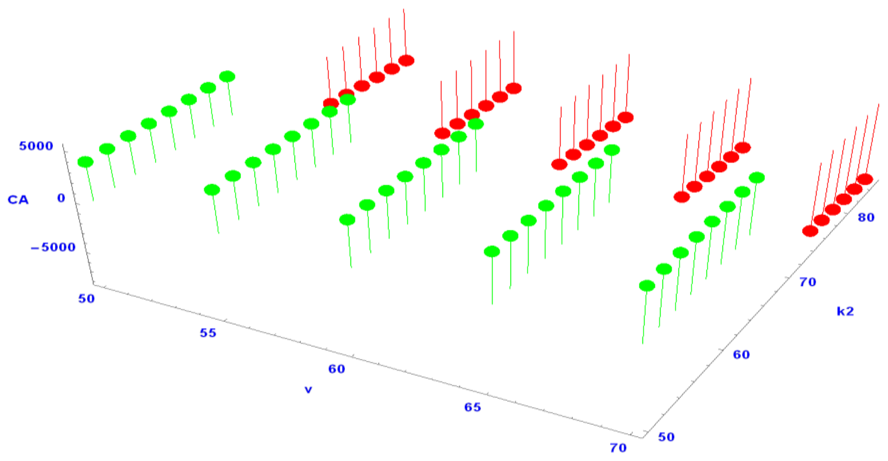

Figure 6.
Attacker payoff in a dynamic game while the defender remains static.

In the second experiment of this dynamic game, we select the *A2D4* stage, as shown in Table 2 of Section 3, where the defender rationally responds against the attacker's dynamic movement. The attacker changes her strategy to inflict low-rate stealthy traffic to evade a defender. Defender wisely follows 'observe then act' rules and applies the *D4* strategy with knowledge-based prediction *ϑ*. This move implies that the defender adjusts the threshold *k2* dynamically. Figure 7 postulates the situation.

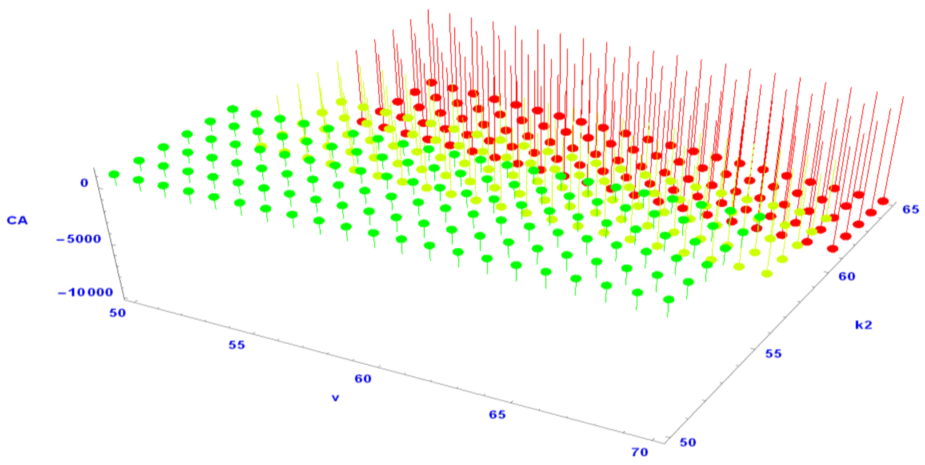

Figure 7.
Defender's wise move in the *A2D4* stage of the dynamic game.

Three saddle points are identified in various sub-states of this stage (*A2D4*) dynamic game. These saddle points are shown in Figure 8 with red point marks. Locations of these three points are (70, 55, 1335.94), (50, 60, -3868.75) and (50, 65, -7259.375). Multiple saddle points attribute the strategic changing pattern in a game stage by both the players. Each saddle point indicates the Nash Equilibrium of the corresponding strategic move in the subgame. When the attacker moves from *A1* to *A2* strategy, Nash Equilibrium at (70, 55, 1335.94) yields her maximum payoff of 1335.94 with 70 nodes and a traffic rate of 55. As the defender moves the *D4* strategy, the payoff of the attacker starts declining and reaches another equilibrium (50, 60, -3868.75). It means that if the defender, in part of the *D4* strategy, sets the threshold *k2* between 55 and 60, then the attacker would obtain a maximum payoff of -3868.75 with 50 nodes and a traffic rate of 60. Imposing more nodes and traffic rates will worsen her payoff as we see the situation in the third equilibrium at (50, 65, -7259375).

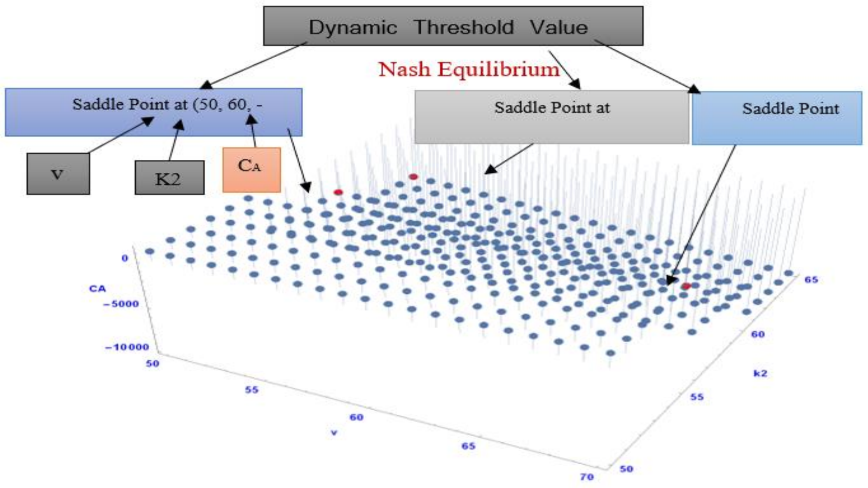

Figure 8.
Various saddle points (red mark) in the *A2D4* stage game. The points are (70, 55, 1335.94), (50, 60, -3868.75) and (50, 65, -7259.375).

If the attacker decides to follow a traffic rate of 65 to maximize her payoff, setting a threshold value *k2* between 60 and 65 by the defender would give the attacker a maximum payoff of -7259.375 with minimum nodes 50. Anything beyond that would make her payoff worse, as seen in the top left corner of Figure 8. The Nash Equilibrium at the end of the *A2D4* stage game is (50, 65, -7259.375).

Figure 9 compares various payoff matrices of attackers for different scenarios. We plot this using the traffic rate of 45-70 Kbps while attack nodes, $v = 70$. As a dynamic response, we select the *A2D4* stage of the proposed dynamic game. A dynamically adjustable threshold value with a good prediction δ can significantly lower the payoff of the attacker. For this, the defender also needs to be rational. Conversely, the static response from EDoS Eye is not as worthy as the dynamic response from Advanced EDoS Eye to efficiently mitigate EDoS attacks in cloud computing.

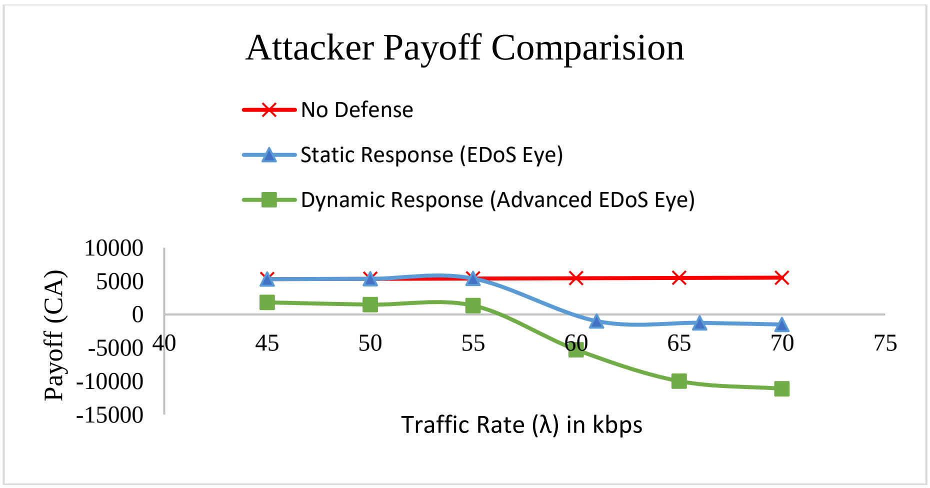

Figure 9.
Attacker payoff comparison in different game scenarios.

The results imply that, with the proper dynamic defensive response, it is possible to drastically reduce the payoff of the attacker, i.e., attacker incentives.

4.2. Response Time Evaluation

Figure 10 illustrates the mean response time taken during an EDoS attack without any mitigation approach. The firewall's mean arrival rate is blended with legitimate and attack traffic. There is no way to differentiate the attack traffic before arriving in the firewall. The response time increases exponentially when the aggregate traffic (λ) reaches near the rate of service rate (μ).

When an EDoS attack is launched, it is necessary to maintain the standard time taken by each *VF* instance according to Equation 20. The creation of duplicate *VF* instances is proportional to the increase in attack strength (ψ), as shown in Figure 11.

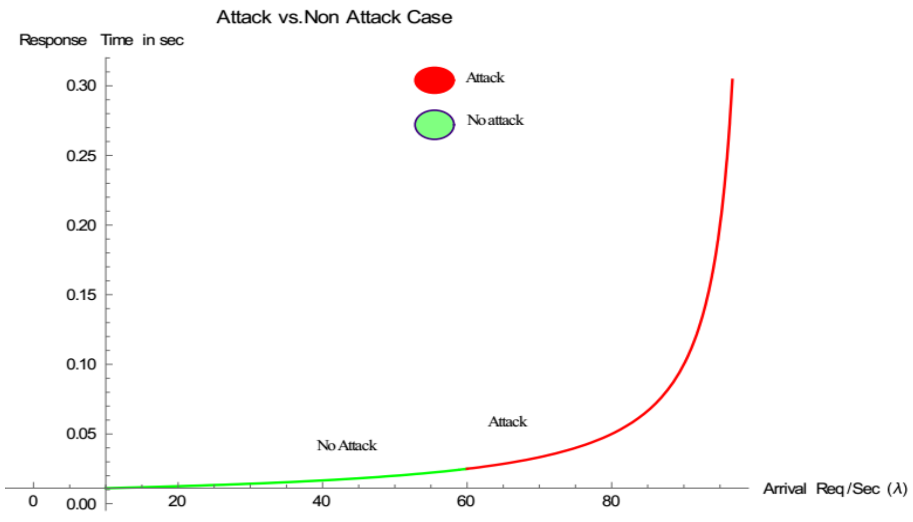

Figure 10.
The mean response time (in seconds) during attack (Without mitigation).

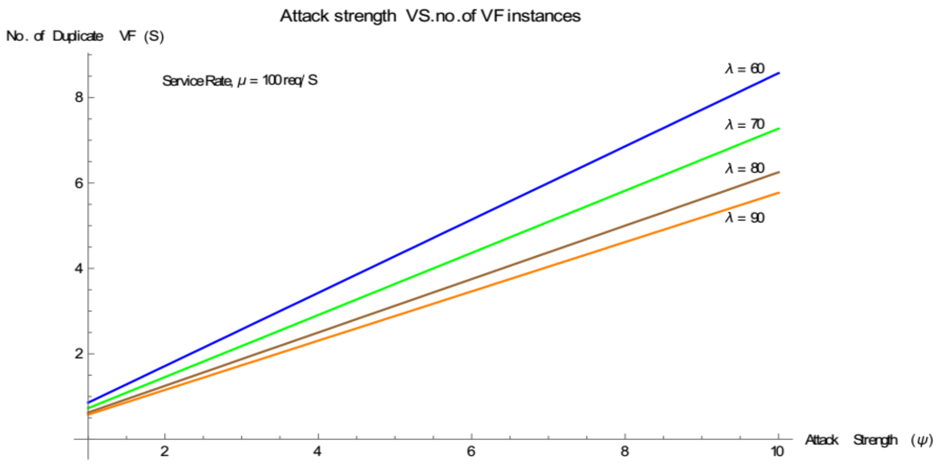

Figure 11.
The relation between attack strength and the number of duplicate *VF* instances with various arrival rates λ.

Legitimate users with guaranteed QoS should be the highest priority even before mitigation occurs. Uninterrupted services to legitimate clients must be ensured when the system is attacked. The proposed model allocates certain provisions of multiple virtual firewall instances to receive and filter large amounts of traffic. In Figure 11, we observe the different requirements of virtual firewall instances with various traffic arrival rates ranging from 60-90 req/s and a fixed service rate of 100 req/s. The requirement of a duplicate number of *VF* instances is low when we allow a high traffic rate. However, it gives lower cost to the Cloud Service Provider (CSP) with service degradation due to increasing response time. On the other hand, if we set to allow a low traffic rate, such as 60 req/s, it can maintain a Service Level Agreement (SLA) with better response time but would take more cost due to more *VF* instances. If CSP makes an SLA favorable towards Cloud Consumers (CC), then auto-scaling costs would be reduced substantially from the consumer's perspective. Thus, the impact of EDoS attacks can be minimized in this way. However, the cloud providers magnify the energy cost [35, 36] because fraudulent resource consumptions cannot be compensated. So, there needs to be a trade-off between CSP and CC in making an effective SLA.

For evaluation and comparison with EDoS-Shield, we only consider response time taken in a virtual firewall (*VF*) and mitigation process that involves the defensive architecture. The measurement of response time due to delays in both the congestion link and the cloud end is very complicated. The reason is that cloud computing architecture comprises heterogeneous devices, unlike clustering systems. Moreover, the proposed mitigation architecture has a negligible impact on the congestion link and the cloud end. Hence, the first two steps of both Equations 23 and 25 are used to evaluate the response time of EDoS-Shield and the proposed model, respectively. The remaining part of both Equations 23 and 25 are considered the same. Figure 12 shows the mean response time of the proposed Advanced EDoS Eye.

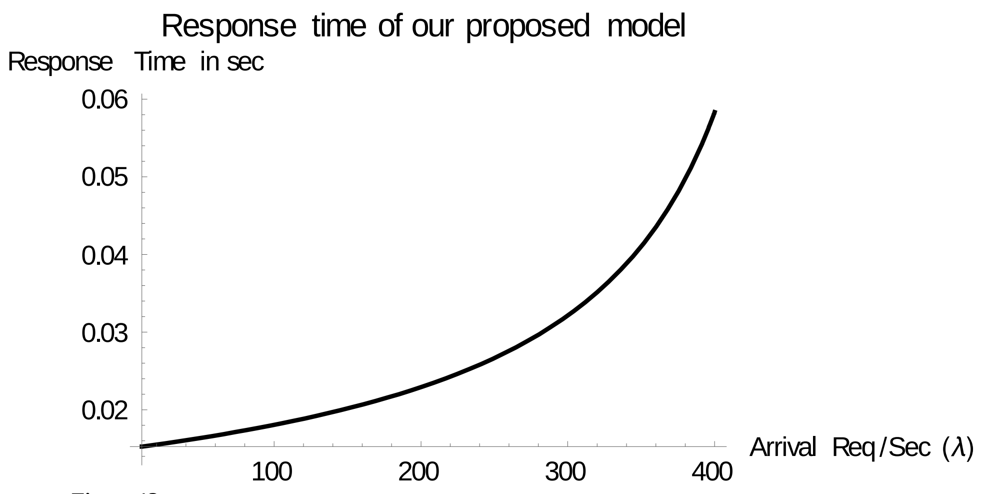

Figure 12.
The response time taken by the proposed Advanced EDoS Eye.

To measure this, we set the number of initial *VF* instances = 5, service rate =100 req/s, and rate of incoming traffic 10-400 req/s. The probability of redirecting traffic to a honeypot is 0.5.

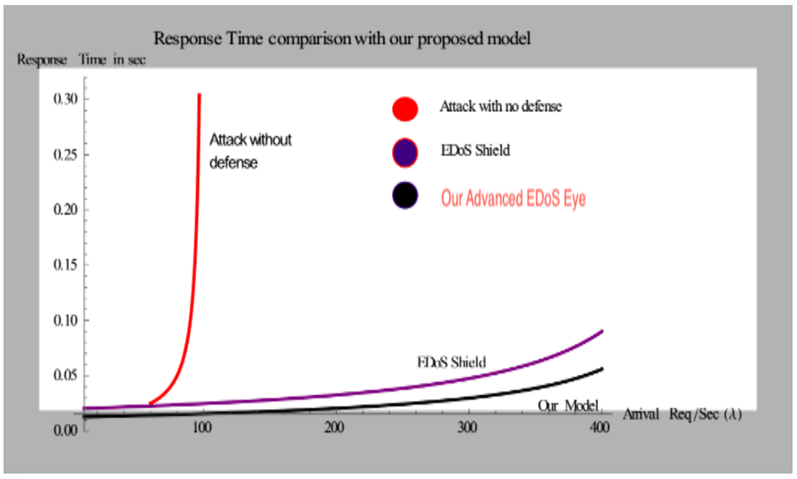

Figure 13.
Comparison of response time with EDoS-Shield and attack without mitigation.

We use the same data to compare EDoS-Shield with the proposed model shown in Figure 13. It indicates that EDoS-Shield takes a slightly longer response time than the proposed model. Mainly, it consumes extra time when verifying each new packet in the verifier node while updating the packet information list in the database.

Conversely, depending on the Dynamic Game-based Decision Module (D-GBDM) in *VF*, we redirect some packets toward the honeypot. There is no scope for a time-intensive verification test in the proposed model. This is the root cause that the proposed model outperforms EDoS-Shield by providing faster response time while ensuring QoS.

Table 2.
Comparison of proposed Advanced EDoS Eye with the EDoS-Shield and the EDoS Eye.

Models	Response Time (in sec)	Attacker Strategy Analysis	Threshold Setting	QoS during Attack
EDoS Shield	0.09 for 400 req/s	No	Static or Predefined	No
EDoS Eye	Not Evaluated	Yes	Static	No
Advanced EDoS Eye (Proposed)	0.06 for 400 req/s	Yes	Dynamic	Yes

4.3. Experimental Validation of the Advanced EDoS Eye

We conducted a simple experiment using Riverbed Modeler Academic Edition 17.5 to validate the proposed model. Discrete event simulation is used to measure the performance of the proposed model. Table 3 shows the configuration of the testbed.

Table 3.
Experimental Setup.

Setup Elements	Number
Number of Legitimate Clients	50
Number of Attack Nodes	20
Number of Routers	3 (gateway for attack traffic, gateway for legitimate traffic and edge router)
Number of Switches	3 (1 attacker switch, 1 legitimate switch and 1 server farm switch)
Load Balancer	1
Firewall	1
Number of *VM* Servers	3 (Server Type Sun Ultra 10 333MHz: 1 CPU, 1 Core(s) Per CPU, 333MHz, Solaris, System)
Communication Links	100 Base T ethernet, 100 Gbps ethernet, 10 Gbps ethernet, PPP DS1
Honeypot	1 (Server Type Sun Ultra 10 333MHz: 1 CPU, 1 Core(s) Per CPU, 333MHz, Solaris, System)
Honeypot Repository	1 (Server Type Sun Ultra 10 333MHz: 1 CPU, 1 Core(s) Per CPU, 333MHz, Solaris, System)
IP Cloud Interface	1
Traffic Type	TCP+UDP
Distribution type	Poisson + Exponential (Attacker) Poisson (Legitimate users)
Experiment Duration	25 minutes

First, we design the scenario in no defense mode to evaluate various performances as shown in Figure 14.

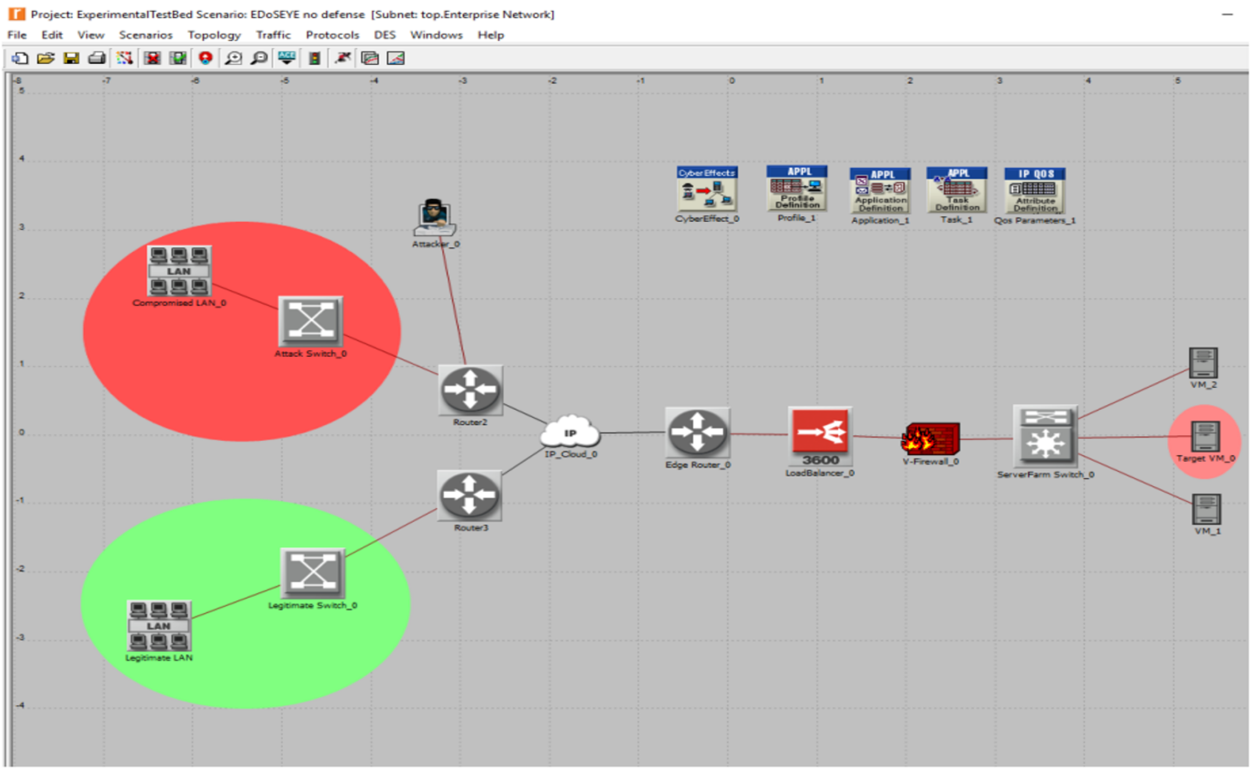

Figure 14.
EDoS attack launches in a typical cloud environment.

In the experiment's first phase, we chose a simple cloud environment with only 3 *VM*s residing on top of the hypervisor according to the proposed design shown in Section 2. The red zone indicates that the attacker compromises a LAN consisting of 20 nodes with EDoS attack scripts. The scripts contain high HTTP, high FTP and a high load of database requests. Besides EDoS attackers, we also design profiles for legitimate users, including e-commerce users, multimedia users, researchers, salesmen, etc. The legitimate users' profiles indicate the green zone in Figure 14. The EDoS attacker targets a specific *VM* to exploit the resources, termed "target *VM*", and indicated in the red mark area. The experiment has been conducted for 25 minutes.

In the second phase, we design the proposed Advanced EDoS Eye model in the riverbed modeler. The scenario of the experimental setup is shown in Figure 15.

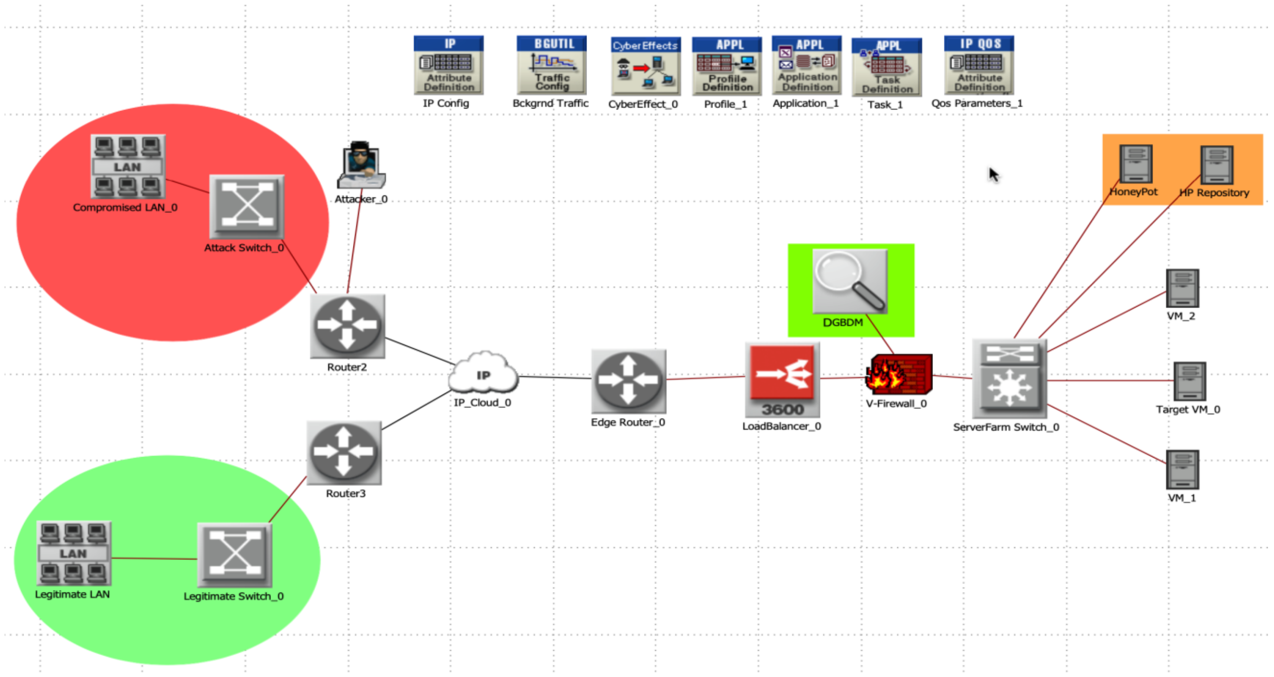

Figure 15.
Proposed Advanced EDoS Eye Model in Riverbed Modeler.

Honeypot and honeypot repository indicated both in yellow marks are the new inclusion in the scenario. In addition, D-GBDM is also incorporated into the virtual firewall, which analyses packets and generates decisions about inbound packets. The duration of the experiment was 25 minutes. The selected discrete event simulation outcomes imply positive and effective results regarding EDoS effect mitigation. Figure 16 shows a considerable reduction in CPU resource usage with the implementation of Advanced EDoS Eye.

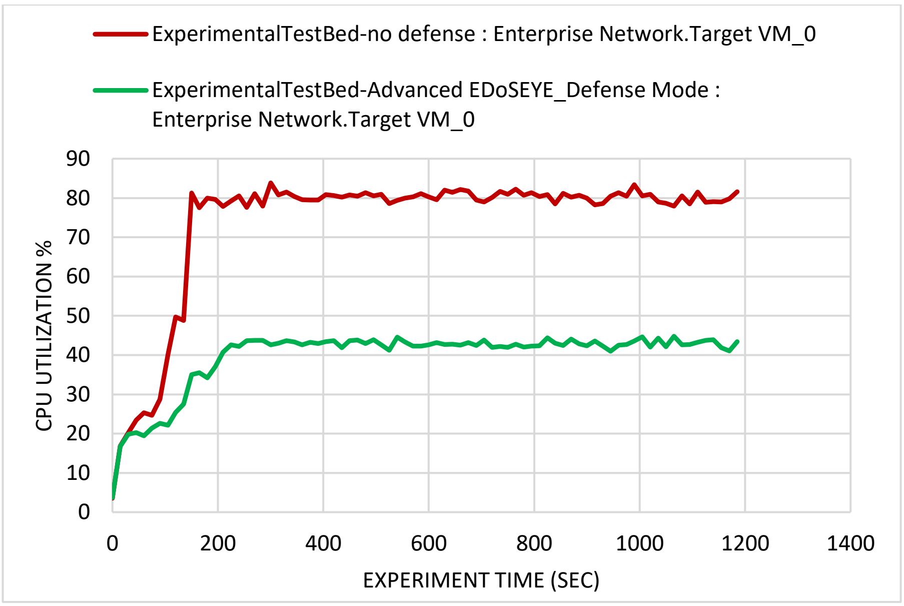

Figure16.
CPU resource usage comparison (Advanced EDoS Eye vs. No defense).

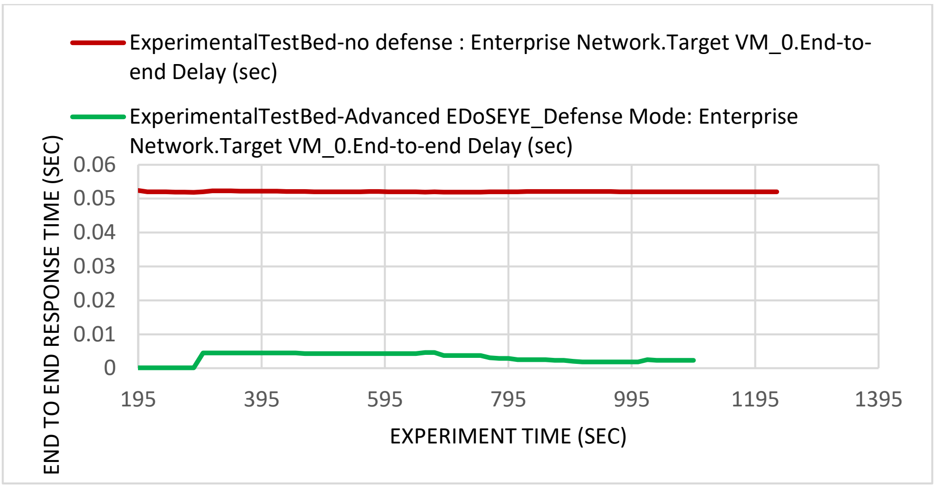

Figure 17.
Avg. end-to-end response time comparison (Advanced EDoS Eye vs. No defense).

The experiment also evaluates performance metrics such as end-to-end response time, throughput and overall HTTP response time, shown in Figures 17, 18, and 19, respectively. In all cases, Advanced EDoS Eye shows a significant performance improvement of the system.

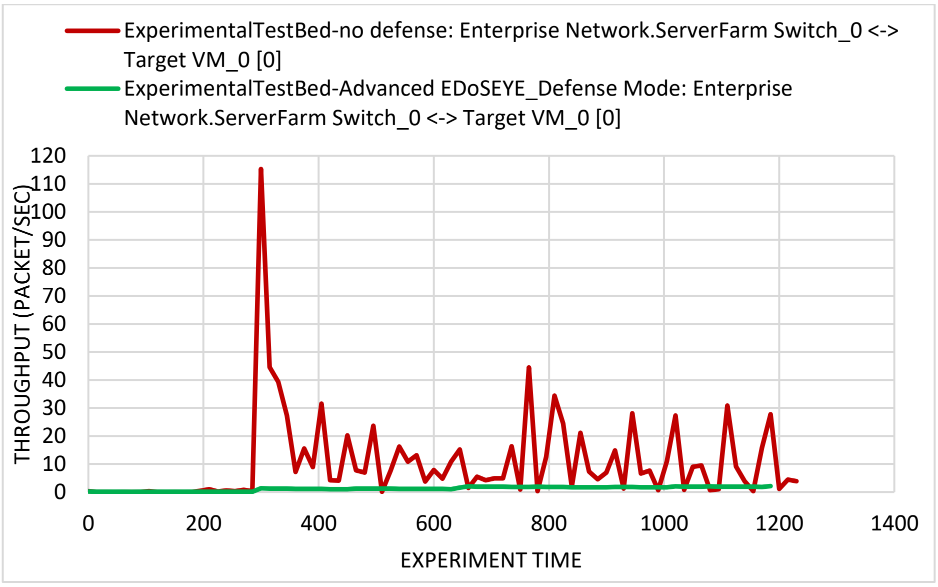

Figure 18.
Throughput comparison (Advanced EDoS Eye vs. no defense).

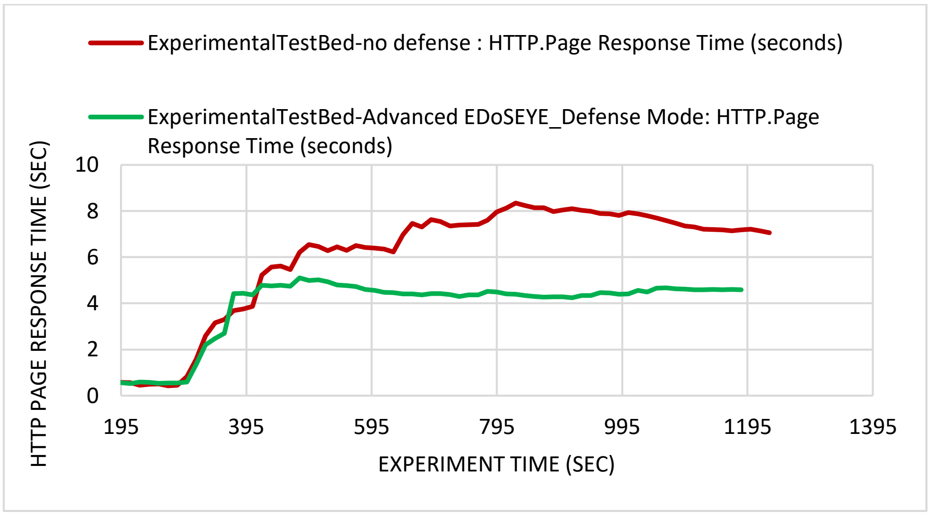

Figure 19.
Avg. HTTP page response time comparison (Advanced EDoS Eye vs no defense).

5. Conclusion

The overall experimental outcomes validate the efficiency of the proposed Advanced EDoS Eye model. The performance of the proposed Advanced EDoS eye makes a distinction which not only eliminates EDoS traffic but also improves the performance of other parameters which was indirectly affected by EDoS attack. Moreover, dynamic games have proven more effective than static games in reducing the payoff of EDoS attackers. Dynamic threshold generation in a multistage dynamic game model can successfully restrict EDoS traffic and save the target cloud consumer as a safeguard. In the simulation, we have shown that the proposed Advanced EDoS Eye outperformed the EDoS shield based on response time, attacker profile analysis and other parameters of QoS. Overall, the proposed Advanced EDoS Eye can fulfil the objectives by determining dynamic threshold, minimizing response time and ensuring QoS during the mitigation action.

References

[1] M. Armbrust *et al.*, "A view of cloud computing," *Communications of the ACM,* vol. 53, no. 4, pp. 50-58, 2010.

[2] R. Buyya, C. S. Yeo, S. Venugopal, J. Broberg, and I. Brandic, "Cloud computing and emerging IT platforms: Vision, hype, and reality for delivering computing as the 5th utility," *Future Generation Computer Systems,* vol. 25, no. 6, pp. 599-616, 2009. https://doi.org/10.1016/j.future.2008.12.001

[3] Z. A. Baig, S. M. Sait, and F. Binbeshr, "Controlled access to cloud resources for mitigating economic denial of sustainability (EDoS) attacks," *Computer Networks,* vol. 97, pp. 31-47, 2016. https://doi.org/10.1016/j.comnet.2016.01.002

[4] S. Yu, Y. Tian, S. Guo, and D. O. Wu, "Can we beat DDoS attacks in clouds?," *IEEE Transactions on Parallel and Distributed Systems,* vol. 25, no. 9, pp. 2245-2254, 2013. https://doi.org/10.1109/TPDS.2013.181

[5] C. Hoff, "Cloud computing security: From DDoS distributed denial of service to EDoS economic denial of sustainability," Rational Survivability, 2008.

[6] F. Al-Haidari, M. Sqalli, and K. Salah, "Evaluation of the impact of EDoS attacks against cloud computing services," *Arabian Journal for Science and Engineering,* vol. 40, no. 3, pp. 773-785, 2015. https://doi.org/10.1007/s13369-014-1548-y

[7] S. H. Khor and A. Nakao, "Spow: On-demand cloud-based eddos mitigation mechanism," presented at the In HotDep Fifth Workshop on Hot Topics in System Dependability, 2009.

[8] M. H. Sqalli, F. Al-Haidari, and K. Salah, "Edos-shield-a two-steps mitigation technique against edos attacks in cloud computing," presented at the In Utility and Cloud Computing (UCC), 2011 Fourth IEEE International Conference on, pp. 49-56, IEEE, Melbourne, 2011.

[9] F. Al-Haidari, M. H. Sqalli, and K. Salah, "Enhanced EDoS-Shield for mitigating EDoS attacks originating from spoofed IP addresses," in *Proceedings of the 2012 IEEE 11th International Conference on Trust, Security and Privacy in Computing and Communications (TrustCom) (pp. 1167–1174), Liverpool, United Kingdom*, 2012.

[10] M. Masood, Z. Anwar, S. A. Raza, and M. A. Hur, "EDoS armor: A cost-effective economic denial of sustainability attack mitigation framework for e-commerce applications in cloud environments," in *Proceedings of the 16th IEEE International Multi Topic Conference (INMIC), Lahore, Pakistan*, 2013.

[11] A. Koduru, T. Neelakantam, and S. M. S. Bhanu, "Detection of economic denial of sustainability using time spent on a web page in cloud," presented at the IEEE International Conference on Cloud Computing in Emerging Markets (CCEM), pp. 1-4, IEEE, Bangalore, 2013.

[12] H. Abbasi, N. Ezzati-Jivan, M. Bellaiche, C. Talhi, and M. R. Dagenais, "Machine learning-based EDoS attack detection technique using execution trace analysis," *Journal of Hardware and Systems Security,* vol. 3, no. 2, pp. 164-176, 2019. https://doi.org/10.1007/s41635-018-0061-2

[13] N. Agrawal and S. Tapaswi, "A proactive defense method for the stealthy EDoS attacks in a cloud environment," *International Journal of Network Management,* vol. 30, no. 2, p. e2094, 2020. https://doi.org/10.1002/nem.2094

[14] A. Shawahna, M. Abu-Amara, A. S. Mahmoud, and Y. Osais, "EDoS-ADS: An enhanced mitigation technique against economic denial of sustainability (EDoS) attacks," *IEEE Transactions on Cloud Computing,* vol. 8, no. 3, pp. 790-804, 2018. https://doi.org/10.1109/TCC.2018.2805907

[15] P. T. Dinh and M. Park, "Dynamic economic-denial-of-sustainability (EDoS) detection in SDN-based cloud," presented at the Fifth International Conference on Fog and Mobile Edge Computing (FMEC), pp. 62-69, IEEE, Paris, 2020.

[16] P. T. Dinh and M. Park, "R-EDoS: Robust economic denial of sustainability detection in an SDN-based cloud through stochastic recurrent neural network," *IEEE Access,* vol. 9, pp. 35057-35074, 2021. https://doi.org/10.1109/ACCESS.2021.3061601

[17] F. Al-Haidari, K. Salah, M. Sqalli, and S. M. Buhari, "Performance modeling and analysis of the EDoS-shield mitigation," *Arabian Journal for Science and Engineering,* vol. 42, no. 2, pp. 793-804, 2017. https://doi.org/10.1007/s13369-016-2331-z

[18] S. Alsowail, M. H. Sqalli, M. Abu-Amara, Z. Baig, and K. Salah, "An experimental evaluation of the EDoS-shield mitigation technique for securing the cloud," *Arabian Journal for Science and Engineering,* vol. 41, no. 12, pp. 5037-5047, 2016. https://doi.org/10.1007/s13369-016-2210-7

[19] Q. Wu, S. Shiva, S. Roy, C. Ellis, and V. Datla, "On modeling and simulation of game theory-based defense mechanisms against DoS and DDoS attacks," in *Proceedings of the 2010 Spring Simulation Multiconference, pp. 1-8, ACM*, 2010.

[20] H. S. Bedi, S. Roy, and S. Shiva, "Game theory-based defense mechanisms against DDoS attacks on TCP/TCP-friendly flows," in *Proceedings of the 2011 IEEE Symposium on Computational Intelligence in Cyber Security (CICS), Paris, France*, 2011.

[21] T. Spyridopoulos, G. Karanikas, T. Tryfonas, and G. Oikonomou, "A game theoretic defence framework against DoS/DDoS cyber attacks," *Computers & Security,* vol. 38, pp. 39-50, 2013. https://doi.org/10.1016/j.cose.2013.03.014

[22] Y. Wang, J. Ma, L. Zhang, W. Ji, D. Lu, and X. Hei, "Dynamic game model of botnet DDoS attack and defense," *Security and Communication Networks,* vol. 9, no. 16, pp. 3127-3140, 2016. https://doi.org/10.1002/sec.1518

[23] F. Z. Chowdhury, M. Y. I. Idris, L. M. Kiah, and M. M. Ahsan, "EDoS eye: A game theoretic approach to mitigate economic denial of sustainability attack in cloud computing," presented at the Proceedings of the 2017 IEEE 8th Control and System Graduate Research Colloquium (ICSGRC), Shah Alam, Malaysia, 2017.

[24] K. Lalropuia and V. Khaitan, "Game theoretic modeling of economic denial of sustainability (EDoS) attack in cloud computing," *Probability in the Engineering and Informational Sciences*, vol. 36, no. 4, pp. 1241-1265, 2022. https://doi.org/10.1017/S0269964821000334

[25] N. Weiler, "Honeypots for distributed denial-of-service attacks," in *Proceedings of the Eleventh IEEE International Workshops on Enabling Technologies: Infrastructure for Collaborative Enterprises (WETICE '02), Pittsburgh, PA, USA, pp. 109–114*, 2002.

[26] L. Spitzner, *Honeypots: Tracking hackers*. Boston, MA, USA: Addison-Wesley Professional, 2002.

[27] A. Sardana and R. Joshi, "An auto-responsive honeypot architecture for dynamic resource allocation and QoS adaptation in DDoS attacked networks," *Computer Communications*, vol. 32, no. 12, pp. 1384-1399, 2009. https://doi.org/10.1016/j.comcom.2009.03.005

[28] T. Ristenpart, E. Tromer, H. Shacham, and S. Savage, "Hey, you, get off of my cloud: Exploring information leakage in third-party compute clouds," in *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS '09), Chicago, IL, USA*, 2009.

[29] I. Mokube and M. Adams, "Honeypots: Concepts, approaches, and challenges," in *Proceedings of the 45th annual southeast regional conference, pp. 321-326, ACM*, 2007.

[30] S. Biedermann, M. Mink, and S. Katzenbeisser, "Fast dynamic extracted honeypots in cloud computing," in *Proceedings of the 2012 ACM Workshop on Cloud Computing Security Workshop, pp. 13-18, ACM*, 2012.

[31] M. Liu, W. Dou, S. Yu, and Z. Zhang, "A clusterized firewall framework for cloud computing, in communications (ICC)," presented at the 2014 IEEE International Conference on Communications (ICC), pp. 3788-3793, IEEE, Sydney (2014), 2014.

[32] H. Wang, Z. Xi, F. Li, and S. Chen, "Abusing public third-party services for EDoS attacks," in *Proceedings of the 10th USENIX Workshop on Offensive Technologies (WOOT '16), Austin, TX, USA, August*, 2016.

[33] J. Cao, K. Hwang, K. Li, and A. Y. Zomaya, "Optimal multiserver configuration for profit maximization in cloud computing," *IEEE Transactions on Parallel and Distributed Systems*, vol. 24, no. 6, pp. 1087-1096, 2012.

[34] K. M. Chandy and C. H. Sauer, "Approximate methods for analyzing queueing network models of computing systems," *ACM Computing Surveys*, vol. 10, no. 3, pp. 281-317, 1978. https://doi.org/10.1145/356733.356737

[35] M. Ficco and F. Palmieri, "Introducing fraudulent energy consumption in cloud infrastructures: A new generation of denial-of-service attacks," *IEEE Systems Journal*, vol. 11, no. 2, pp. 460-470, 2015. https://doi.org/10.1109/JSYST.2015.2414822

[36] J. Idziorek, M. Tannian, and D. Jacobson, "Detecting fraudulent use of cloud resources," in *Proceedings of the 3rd ACM Workshop on Cloud Computing Security Workshop (pp. 61-72)*, 2011.