Game Theory for Strategic DDoS Mitigation

Omkar Thakoor
University of Southern California
othakoor@usc.edu

Phebe Vayanos
University of Southern California
phebe.vayanos@usc.edu

Milind Tambe
Harvard University
milind_tambe@harvard.edu

Minlan Yu
Harvard University
minlanyu@g.harvard.edu

ABSTRACT

Mitigating DDoS attacks on the networks is an important security challenge for the Internet Service Providers. Recent work has shown a flexible and elastic DDoS defense mechanism via Software Defined Networking/Network functions virtualization and such systems are showing increasingly efficient performance. However, an intelligent attacker can exploit the system by dynamically adapting the attack profile which a static defense cannot adequately handle. We propose a game theoretic model for a dynamic deceptive defense. We decompose the problem in two phases. The first phase focuses on optimising the defense while trying to deceive the attacker with dynamic adaptations in the defense strategy. The second phase consists of optimising the resource allocation in order to execute the pre-computed defence strategy. We show analytical and numerical results showing the efficiency of our computation algorithms.

KEYWORDS

DDoS; Cyber Deception; Game Theory; Resource Allocation

1 INTRODUCTION

Mitigating distributed denial-of-service (DDoS) attacks on the networks is an important security challenge for the Internet Service Providers. A rapid increase in DDoS related cybercrime has reached an estimated 20,000 daily attacks [17], moreover, the attack types and signatures constantly evolve [14, 20, 24]. Conventionally, DDoS defense has relied on using hardware resources [5, 19] which often tend to be proprietary, expensive and not flexible in scaling functioning, or network positioning. Thus, this is not effective in thwarting new attack types and high volumes. Recent work has shown a flexible and elastic DDoS defense mechanism via Software Defined Networking (SDN) [12, 16], Network functions virtualization (NFV) [9]. Such systems are showing increasingly efficient performance, specifically in handling larger volumes or new types of attacks, and are being deployed in practice more and more by leading ISP providers [2, 3, 6, 8].

We consider such a setup. The attacker aims to exhaust the network bandwidth of the victim by flexibly launching packets of several attack types (e.g., TCP SYN flood, UDP flood, DNS amplification). We assume there to be anomaly detection techniques in place which can raise flags for suspicious packets — ISPs are typically equipped with these [4]. Thereby, the suspicious traffic and its estimated volumes at different network ingresses are assumed to be the given input to the problem. Further, we assume there are pre-defined defense strategies for different attack types [11] which analyze the incoming traffic and respond accordingly. Each such strategy can be represented by a Directed Acyclic Graph where each node is a logical module that processes the suspicious packet and the edges indicate the traffic forwarding rules. This is the stage of the problem that determines the scope of this paper. The complete structural flow is described in Fig. 1. Each defense module needs to be realized via different virtual machines. The defender's goal is to find the allocation of virtual resources which can process as much suspicious traffic as possible while incurring as low latency costs as possible which result from the processing.

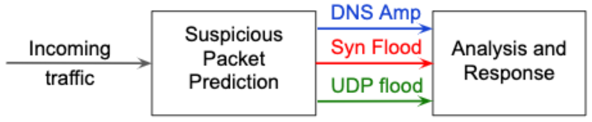

Figure 1: DDoS Defense Pipeline

While deploying a defnese strategy for a given attack mix is a relatively simpler problem, an intelligent attacker can exploit the system by dynamically adapting the attack profile which a static defense cannot adequately handle. Previous models [11] attempt to handle this issue by considering regret-minimization objectives that can be met by deploying well-studied strategies such as *Follow the perturbed leader* (FPL) [13]. Such an approach is conservative, in that it merely reacts to the attacker's adaptations, but does not aim to anticipate them and administer accordingly. This approach is particularly inadequate when the re-allocation of the defense resources to reciprocate the shifts in attack has significant delay, proving attacker's exploitative adaptions effective. To that end, we propose a game theoretic model to compute a defense strategy that is not only dynamic, but can prove deceptive to the attacker. An intuitive example of the deceptive property is to have the defenses set up so as to appear having false vulnerabilities, steering the attacker to exploit them, which can then be anticipated and effectively nullified.

Several works in the past have used Game theory for Cyber security, and in particular, Cyber deception. The Cyber Deception Game (CDG) [21] and Cyber Camaouflage Game [25] are game-theoretic models for deception via attribute obfuscation. Other works in cyber defense [1, 15, 22, 23] have adopted game theoretic models,

Proc. of the 19th International Conference on Autonomous Agents and Multiagent Systems (AAMAS 2020), B. An, N. Yorke-Smith, A. El Fallah Seghrouchni, G. Sukthankar (eds.), May 2020, Auckland, New Zealand

https://doi.org/doi

including several that aim to strategically deploy honeypots [7, 18]. A dynamic defense model in the green security domain was shown effective in [10] and lays the foundation for our solution approach.

Our contributions are as follows. We propose a model that consists of two phases. The first phase computes a dynamic defense strategy, whereas the second phase optimally deploys compute resources to implement the strategy. We show analytical results for the minimax equilibria of the static (single-stage) game and subsequently provide Mixed Integer Linear Program (MILP) formulations for computing the optimal dynamic strategies of the multiple-stage game as well as efficient and effective heuristics. Finally, we show numerical results on simulated attack data which demonstrate the efficacy of out solutions approaches.

2 DEFENSE CAPACITY COMPUTATION AND DECEPTION (BASE GAME AND REPEATED SETTING)

In this section, we abstract out the details pertaining to the VM resource allocation for handling attack traffic of different types. We model the problem as follows.

Let A denote the set of attack types. For each attack type a, we say that handling a unit of traffic takes up c_a amount of defense resources. The defender must decide the traffic volume f_a for each type that he wants to allocate resources for. Thus, the strategy can be via vector $f = (f_a)_a$. Given this strategy f, the total amount of resources used up is $\sum_a f_a c_a$; we assume limited resources for the defender and thus capping the total usage of $\sum_a f_a c_a$ by a budget C. The attacker's strategy is to launch an attack mix of different types, and is similarly defined via a vector $g = \{g_a\}_a$. The Attacker must need computational and technological resources for the attacks, which are typically limited. We assume launching a unit traffic of type a has a net cost r_a to the attacker. Thus, given his attack strategy $g = \{g_a\}_a$, his total cost is $\sum_a g_a r_a$, assumed to be capped by a budget R. Given the player strategies, the resultant reward to the attacker from a particular type of attack depends on the amount of traffic that goes through, i.e., cannot be processed by the defender, which is $\max(g_a - f_a, 0)$ for each type a. Assuming a per-unit utility of u_a and a linear utility aggregation, the total utility for the attacker is given by $U(f, g) = \sum_a u_a \max(g_a - f_a, 0)$. Since the defender's goal would be to minimize the traffic that penetrates the network, we model this as a zero-sum game.

First, we outline the extreme case when the defender's budget is so high that it is possible to set up a defense that would not allow the attacker a positive utility.

PROPOSITION 2.1. *If $C/R \geq \sum_a c_a/r_a$, the defender can manage to have zero utility, i.e. thwart the attacker completely.*

PROOF. Since the maximum volume the attacker can attack for a single type a with all his budget is R/r_a, the defender can set up a defense to handle all of it by spending $R/r_a \cdot c_a$ for it. Aggregating this for all the types, the defender can play a strategy $\{R/r_a\}_a$ with a total cost of $\sum_a R/r_a \cdot c_a$, if the total budget C allows for it, which will prevent the attacker from succeeding with any attack type. Thus, this is possible if $C \geq \sum_a R/r_a \cdot c_a$, concluding the proof. □

For the rest of the paper, we consider the more practical case that the defender's budget is not as high. First, we analyze the scenario where the defense strategy is static.

2.1 Static Play

When the defense strategy is fixed, the attacker may be able to conduct reconnaissance techniques to learn the defense and adjust the attack to be the best response. Hence, we model this as a Stackelberg game where the defender is the leader and the attacker is the follower. This scenario of static strategies is also sufficient to consider cases when the players can change strategies but do so by committing to fixed mixed strategies, as explained next.

Mixed strategies. Suppose the players are able to implement mixed strategies. As in any Stackelberg game, the follower always has a pure strategy best response. Hence, it suffices to consider only pure strategies for the attacker. Subsequently, suppose the defender plays a mixed strategy, i.e., the defender strategy f is drawn from a certain distribution P. Then, for any attacker strategy g, the attacker's expected utility $\mathbb{E}_{f \sim P}[U(f, g)]$ is greater than $U(\mathbb{E}[f], g)$ by Jensen's inequality since $U()$ is convex in f. Hence, the defender can simply play the pure strategy $E[f]$ and do at least as good. Hence, it suffices to only consider pure strategies for the players.

Next, we analyze the attacker best response. We can show that,

PROPOSITION 2.2. *Given the defender's strategy f, the attacker has a best response attacking only one type, say a^*, such that*

$$a^* \in \underset{a}{\operatorname{argmax}}\, u_a \left(R/r_a - f_a\right)$$

.

PROOF. First, note that if the attacker in his best response attacks a non-zero volume g_a for a type a, it must be that $g_a \geq f_a$ since otherwise he gets no utility from attacking type a and could instead use that budget on other types to potentially gain a higher utility. Now, say g is a best response that attacks fewest different types. To show contradiction, say $\exists a,\ a'$ s.t. $g_a,\ g_{a'} > 0$. WLOG, let $u_a/r_a \geq u_{a'}/r_{a'}$. Then, consider the budget of $R' = g_{a'} r_{a'}$ which is required to attack a'; The attacker can instead use R' to increase the attack volume for a by R'/r_a. This causes a net change of $u_a(R'/r_a) - u_{a'}(R'/r_{a'} - f_{a'})$, which is non-negative since $f_{a'} \geq 0$ and $u_a/r_a \geq u_{a'}/r_{a'}$. Hence, this modified strategy must be a best response for the attacker with fewer attacks types being attacked than g, a contradiction. Thus, there exists a best response with just one type attacked.

Subsequently, if only type a^* is attacked, the resultant attacker utility is $u_{a*}(R/r_{a^*} - f_{a^*})$. It follows, that for such a best response, $a^* \in \operatorname{argmax}_a u_a \left(R/r_a - f_a\right)$ □

Given that the attacker can best-respond to the defender's defense strategy, we consider the minimax strategy for the defender to minimize the worst-case loss. We can show that,

PROPOSITION 2.3. *The defender has a minimax strategy f^* yields a utility of λ where*

$$\sum_a c_a(R/r_a - \lambda/u_a) = C \quad \textit{and} \quad f^*_a = (R/r_a - \lambda/u_a)$$

.

Proof. For any defender strategy f, let $v_a(f) = u_a(R/r_a - f_a)$ denote the attacker utility for attacking only type a in response to f.

Now, let the defender minimax strategy be f^*. By Prop. 2.2, there exists a best response to f^* that attacks only one type, say a. Now, if $\exists a' \in A$, and $\delta > 0$ s.t. $v_a(f^*) - v_{a'}(f^*) \geq \delta$, then, the defender can play a strategy f' constructed from f^* by

(1) reducing the defense for a' by, say $\delta/2 \cdot (1/u_{a'})$ so that $v_{a'}(f')$ is higher than $v_{a'}(f^*)$ by $\delta/2$. This frees up a budget $c' = c_{a'}\delta/(2u_{a'})$.
(2) using the surplus budget c' to increase the defense for all types $a'' \in A \setminus \{a'\}$ s.t. each $v_{a''}(f')$ is lower than $v_{a''}(f^*)$ by a $\delta' > 0$, s.t. $\delta' \leq \delta/2$.

As a result, f' constructed as above still has the same attacker best response but lower attacker utility, thus a contradiction. Hence, we must have $v_a(f^*) =$ (say) $\lambda\ \forall a \in A$; λ thus being the minmax utility. This gives us the second half of the claim. Rewriting it for each a gives $f^*_a = (R/r_a - \lambda/u_a)$ and further using the defender's budget constraint gives the first half. The system of equations together give f^* and λ (the closed forms are omitted for brevity). □

2.2 Dynamic defense and Deception

Now, we consider the case when the defender can dynamically alter the defense strategy. In this case, if the defender can anticipate the attacker's adaptations, he can change his own strategy so as to exploit the attacker. As the defender varies the strategies, he can achieve deception by showing the attacker fake vulnerabilities and anticipating the subsequent attack response. This can be modeled in multiple rounds/stages. For simplicity, we assume the budget is uniformly spread across the rounds for the attack and defense capabilities respectively. Typically the budget captures the compute capacities in the network and thus, assuming an equal budget for a given time period is a reasonable assumption. For an attacker who does not adjust to the varying defense, and launches a static attack instead, the defender can simply play the minimax strategy discussed in the previous section, so as to cover the worst case. The following idea of deception works against an attacker who can perceive the defender strategy in the previous rounds, and launches an attack in the new round in response to the anticipated defender strategy estimated from the observed strategies in the previous rounds. This idea of deception demonstrated by the following example.

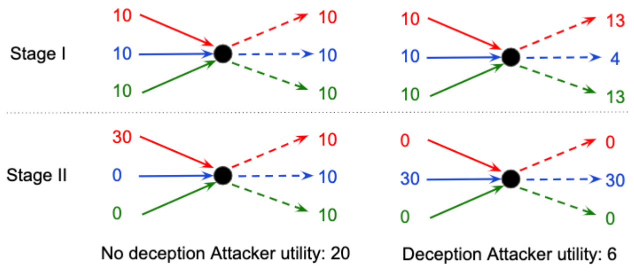
Stage I
10
10
10
10
10
10
10
10
10
13
4
13
Stage II
30
0
0
10
10
10
0
30
0
0
30
0
No deception Attacker utility: 20
Deception Attacker utility: 6

Figure 2: Example: Deception can yield better results

Consider Fig. 2 where we consider two stages (time intervals) allowing the players to adapt strategies between the stages. Let there be 3 attack types. Let the player budgets be $C = R = 30$ in each round and the per-unit costs and utilities for all attack types for both the players be 1. When the defender is not attempting any deception or dynamic adaptations, he can equally distribute the budget across types to handle attacks of all types. The attacker without any prior observation launches uniform attacks in the first round, however, randomly exploits one of the types in the next stage, which the defender cannot predict and must resort to the same uniform strategy, yielding an attacker utility of 20. On the other hand, if the defender knows the attacker's exploitative nature, he can display vulnerability for type two with a very weak defense. As the attacker attempts to exploit it however, the defender scales up the defense there to full strength, reducing the aggregate attacker utility over two rounds to 6. Thus, this example demonstrates the effect of adapting strategies dynamically and the potential for deception.

Formally, we model this as a $T(<\infty)$ round game. We assume that the attacker has a memory length Γ, and non-negative coefficients $\alpha_0, \ldots, \alpha_\tau$ summing to 1, which govern the way attacker estimates the new defender strategy given the past observations. Specifically, in round t, when the attacker has observed the defender strategy in the previous Γ rounds, he anticipates a strategy $h^t = \alpha_0 f^0 + \sum_{\tau=1}^{\Gamma} \alpha_\tau f^{t-\tau}$, where f^0 is the *uniform* strategy which we assume as the bias/noise representing the lack of certainty in the estimated future strategy. When the number of previous rounds are fewer than the memory, we treat $f^{t-\tau} = f^0$. We let the player's budgets in each rounds be denoted by R, C respectively as before. The rest of the notation is carried from the static play as well.

In any round t, the attacker estimates the defender strategy h^t, and assuming rationality, plays a best response to it. Previous works have also considered bounded rationality via SUQR models. Our formulation can be easily adapted for such settings by appropriately changing the attacker response modelling. For a rational attacker, however, we can formulate the following objective for the defender.

$$\min_{f^1,\ldots,f^T} \quad \sum_t U(f^t, g^t) \tag{1}$$

$$\text{s.t.} \quad U(h^t, g^t) \geq \max_a u_a(R/r_a - h^t_a) \ \ \forall t \leq T \tag{1a}$$

$$\sum_a f^t_a c_a \leq C, \quad \sum_a g^t_a r_a \leq R \ \ \forall t \leq T \tag{1b}$$

$$h^t = \alpha_0 f^0 + \sum_{\tau=1}^{\Gamma} \alpha_\tau f^{t-\tau} \forall t \leq T \tag{1c}$$

The objective of the optimization problem is the attacker utility aggregated over all rounds which the defender wants to minimize. Constraint (1a) ensures the attacker plays a best response g^t against his estimated defender strategy h^t in each round t. Inequalities in (1b) capture the budget constraint for the players in each round. The final inequality defines the attacker's estimate h^t of the defender strategy in each round t in terms of the previous observations.

Note that the function $U(f, g)$ is not linear. The Attacker BR constraint in the OP above can be linearized in two ways:

(1) Introduce binary variables z^t_a, and let $l^t_a = \max(g^t_a - h^t_a, 0)$. Then, using a big constant M, we add the constraints

$$\sum_a u_a l^t_a \geq \max_a u_a(R/r_a - h^t_a)$$

$$l_a^t \geq 0, \quad l_a^t \geq g_a^t - h_a^t$$
$$l_a^t \leq g_a^t - h_a^t + Mz_a^t, \quad l_a^t \leq M(1 - z_a^t)$$

(2) Introduce binary variables z_a^t, which indicate whether the attacker attacks type a (only) in that round. This is since we know that the attacker best response attacking only one type exists. Then, we add the constraints:

$$\sum_a z_a^t u_a(R/r_a - h_a^t) \geq \max_a u_a(R/r_a - h_a^t), \qquad \sum_a z_a^t = 1$$

This further leads to bilinear terms $z_a^t h_a^t$ which can be linearized (using big-M constraints).

This turns the formulation into a Mixed-integer Linear Program (MILP). Due to the integer variables and Big-M constraints, computing the solution for all T rounds at once is expensive when T is large. Hence, we propose the following heuristics.

Look and Book Heuristic. Here, we only consider ("look" at) the next few rounds, and compute an optimal solution for say n future rounds. However, since the subsequent rounds have not considered enough future rounds, we only commit to ("book") the first, say $m < n$ rounds and repeat the process after these m rounds. We note that the future rewards after these m rounds may get overestimated since they are never achieved, and to remedy this, we introduce a discount factor γ for the future rounds when computing the optimal solution.

Algorithm 1: Look and Book (LB-(m,n))

```
for $t = 1; t += m; t \leq T$
    $(f^t, \ldots, f^{t+n-1}) \leftarrow lb(t, n)$
Return $(f^1, \ldots, f^T)$
```

The function lb(τ, n) solves the following optimization problem:

$$\min_{f^\tau,\ldots,f^{\tau+n-1}} \sum_{t=\tau}^{\tau+m-1} U(f^t, g^t) + \sum_{t=\tau+m}^{\tau+n-1} U(f^t, g^t)\gamma^{t-(\tau+m)+1} \tag{2}$$

$$\text{s.t.} \quad U(h^t, g^t) \geq \max_a u_a(R/r_a - h_a^t) \ \ \forall t \leq T \tag{2a}$$

$$\sum_a f_a^t c_a \leq C, \quad \sum_a g_a^t r_a \leq R \ \ \forall t \leq T \tag{2b}$$

$$h^t = \alpha_0 f^0 + \sum_{\tau=1}^{\Gamma} \alpha_\tau f^{t-\tau} \tag{2c}$$

Fixed Sequence Heuristic. The idea here is to find a short sequence of strategies with fixed length M and require the defender to execute this sequence repeatedly. This can be computed by simply adding the following constraint to formulation (1):

$$f^t = f^{t-M} \quad \forall t \mid M < t \leq T$$

We name this heuristic FS-M corresponding to the sequence length M. This heuristic has the following approximation guarantee.

THEOREM 2.4. *Let the memory length be $\Gamma = 1$ and the bias coefficient $\alpha_0 = 0$, i.e. the attacker estimates the defender strategy $h^t = f^{t-1}$ in each round t. Then, there exists a fixed sequence strategy giving a $(1 - \frac{1}{M})\frac{Z-1}{Z+1}$ approximation to the optimal strategy profile in terms of the normalized utility, where $Z = T/M$.*

This property follows from Theorem 1 in [10] as it can be shown to hold independent of the exhaustive key differences in the models which are utility functions, strategy spaces and attacker rationality.

3 DDOS MITIGATION RESOURCE ALLOCATION

Recent works [11] have proposed DDoS mitigation techniques that involve allocating VM or hardware resources which process malicious traffic. We consider this resource allocation problem with the goal of minimizing latency, so as to execute the pre-computed strategy in the first phase. We use a running example for illustration as shown in Fig. 3.

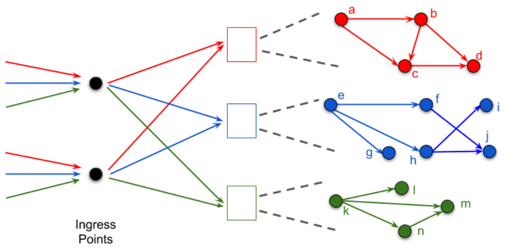

Figure 3: Traffic flow and resource allocation for DDoS mitigation

Let A denote the set of attack types. For each $a \in A$, let $\mathcal{T}_a$ denote the different types of virtual machines required to process traffic of attack type a. Each such VM type represents a distinct logical module it runs under the defense strategy corresponding to the particular attack type. In the adjoining example, we consider three attack types, i.e., $A = \{1, 2, 3\}$ with the first having six different types of virtual machines, namely the set $\mathcal{T}_1 = \{a, b, \ldots, f\}$. Further, let $\mathcal{T} = \bigcup_a \mathcal{T}_a$. (We assume that the defense modules associated with different attack types are all different, i.e. the sets $\mathcal{T}_a$ for different a are all disjoint). A VM of type t can process traffic upto F_t which is input to the problem. The *strategy graph* for this attack type is a DAG denoted by $G_a = (\mathcal{T}_a, E_a)$, where $E_a \subseteq \mathcal{T}_a \times \mathcal{T}_a$ is the set of directed edges. Fig. 4 shows a strategy graph illustration. The example in Fig. 3 shows 3 strategy graphs each corresponding to an attack type.

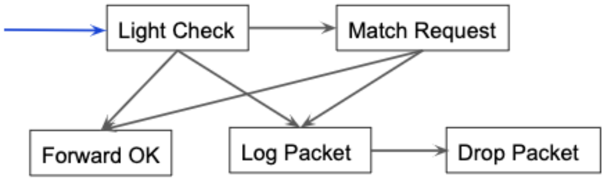

Figure 4: Strategy graph example

The virtual machines are instantiated at several servers which are placed at several datacenters. Let $\mathcal{S}$ denote the set of servers

Symbol	Meaning
$a \in A$	Attack type from the set of types
$\mathcal{T}_a$	Types of VMs for a given $a \in A$
$G_a = (\mathcal{T}_a, E_a)$	Strategy graph for $a \in A$
t_a^{in}	The VM type in $\mathcal{T}_a$ that is the source of DAG G_a
$\mathcal{T}, \mathcal{T}^{\text{in}}$	$\bigcup_a \mathcal{T}_a$ and $\bigcup_a \{t_a^{\text{in}}\}$ resp.
$p \in P$	A PoP node from the set of PoP nodes
g_{pa}	Attack volume to be handled at $p \in P$ of type $a \in A$
$\delta_{tt'}$	Estimated fraction of traffic on edge (t, t') in a strategy graph
F_t	Processing capacity of a VM of type t
C_s	Server compute capacity of $s \in S$
C_d	Datacenter uplink capacity
L_{pd}	per unit latency cost between $p \in P$ and a server at datacenter d
$L_{ss'}$	per unit latency cost between servers s and s'
α	Relative importance of inter- to intra- datacenter latency cost
λ_{pa}	The payoff from an unprocessed packet of type $a \in A$ at $p \in P$
f_{pts} or f_{es}	The total traffic on edge $e = (p, t) \in E^{\text{in}}$ that is sent to VMs of type t on server s
$f_{tt'ss'}$ or $f_{ess'}$	Total traffic on edge $e = (t, t') \in E$ that is sent from VMs of type t on server s to VMs of t' on s'
n_{st}	The number of VMs of type t at server s

Table 1: Table of notation

which can be partitioned as $\mathcal{S} = \bigcup_d \mathcal{S}_d$ where each $\mathcal{S}_d$ denotes the set of servers at a particular datacenter $d \in D$. For the MILP formulation, consider the set of variables $\{n_{st}\}$ where each n_{st} denotes the number of VMs at server s allocated to carry out the processing of type t. For any attack type a, let $t_a^{\text{in}} \in \mathcal{T}_a$ be the module in the corresponding strategy graph where the traffic is routed from a PoP node, i.e. the unique source node of the DAG G_a, and let $\bigcup_a \{t_a^{\text{in}}\} = \mathcal{T}^{\text{in}}$. In our example, $\mathcal{T}^{\text{in}} = \{a, g, l\}$. Let the set of *ingress* or PoP nodes be $\mathcal{P}$, i.e., the points where the traffic enters the network. Let E^{in} denote the edges from $\mathcal{P}$ to $\bigcup_a \{t_a^{\text{in}}\}$. The example shows two PoPs amounting to 6 edges in E^{in}.

For an edge $e = (t, t') \in \bigcup_a E_a$ and servers s, s', let $f_{ess'}$ or $f_{tt'ss'}$ denote the traffic flow along edge e from server s to s'. Similarly, for edge $e = (p, t) \in E^{\text{in}}$ and servers s, let f_{es} or f_{pts} denote the portion of traffic flow along edge e from PoP node p to server s. Let g_{pa} be the attack traffic volume that the defense strategy has to handle for attack type a at PoP p, which is given by the first phase (see Section 2). This traffic flow is demonstrated in Fig. 5.

Let $\delta_{tt'}$ be the expected fraction of traffic from t to t' which is given as input. Let L_{pd} denote the latency cost for routing the traffic from PoP p to datacenter d, and let $L_{ss'}$ denote the latency cost for routing the traffic from server s to server s' with α being the relative importance of the two latency costs. The notation has been summarized in Table 1.

[11] assumes that there are sufficient resources to handle the attack volume and given the constraint that all the suspicious traffic is to be processed, the goal is to route the traffic through VM defense resources so that the latency costs incurred are minimized. Hence we get the following MILP.

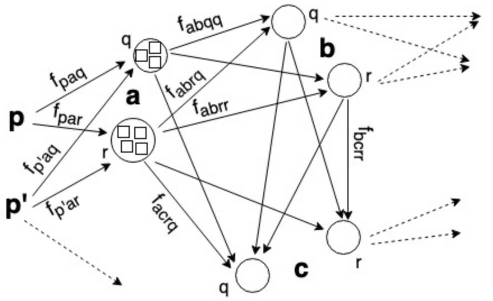

Figure 5: Traffic flow among servers: a,b,c are three VM types and q,r are servers which run VMs to implement the corresponding analysis modules

$$
\begin{aligned}
\min_{\{f_{st}\},\{n_{st}\}} \quad & L \\
\text{s.t.} \quad & L \geq \alpha \sum_{p,d} L_{pd} \sum_{a} \sum_{s \in S_d} f_{pt_a^{\text{in}}s} + \sum_{s,s'} L_{ss'} \sum_{e \in E} f_{ess'} \quad \text{(Latency)} \\
& n_{st} F_t \geq \sum_{t' : (t',t) \in E_a} \sum_{s' \in S} f_{t'ts's} \quad \forall s \forall t \in \mathcal{T}_a \forall a \\
& n_{st} F_t \geq \sum_{p} f_{pts} \quad \forall s \forall t \in \mathcal{T}_a^{\text{in}} \quad \text{(Adequate VM allocation)} \\
& \sum_{s,s'} f_{tt'ss'} = g_{tt'} \quad \forall t, t' \qquad \text{(Flow variables match strategy)} \\
& \sum_{t',s'} f_{t'ts's} = \sum_{t'',s''} f_{tt''ss''} \quad \forall t, s \qquad \text{(flow conservation)} \\
& \textstyle\sum_t n_{st} \leq C_s \quad \forall s \qquad \text{(Server compute capacity)} \\
& \sum_{p,a} \sum_{s \in S_d} f_{pt_a^{\text{in}}s} \leq C_d \quad \forall d \qquad \text{(Datacenter uplink capacity)}
\end{aligned}
$$

The various inequalities capture the model constraints as described next to them. Note that the heterogeneity of the attack traffic comes from the various ingress-'attack type' pairs, thus, when applying the strategy computation phase to the problem as analyzed in the previous section, each such ingress-'attack type' pair is regarded as an *attack type* therein.

Having analyzed both the phases, we next show numerical results on our approaches.

4 NUMERICAL RESULTS

We compare the proposed heuristic approaches with varying parameters and game settings. LB-(1,1) is the myopic strategy that maximizes the reward in each round and FS-1 is equivalent to calculating the best static defender strategy against an attacker with given memory and coefficients. For LB-(m,n) strategies in general, we tune the discount factor γ and pick the one that works the best in our comparisons.

Table 2 shows that computing the complete optimal solution quickly becomes untractable. With a cut-off time of 1 hour, most instances do not finish with 10 rounds and 4 or 5 attack types. For smaller values, the average runtime is seen to heavily rise with the the number of rounds or attack types which can be justified due to the fact that the number of integer variables scales with both of these.

		Attack types		
		3	4	5
Rounds	3	3.45	5.87	9.18
	5	18.62	32.11	48.02
	10	43.13	–	–

Table 2: Runtime of the Full solution MILP formulation (min.) as No. of Attack types and Rounds are varied

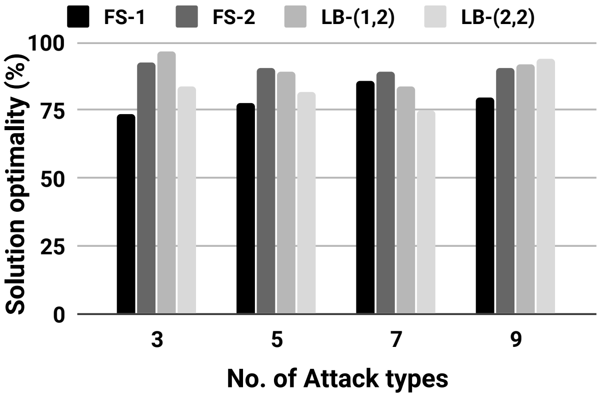

Figure 6: Optimality of different heuristics compared

		Servers		
		10	50	100
VM types	10	0.73	7.41	19.2
	20	2.54	21.02	44.12
	30	6.32	36.25	54.51

Table 3: Runtime (minutes) of the Resource allocation MILP as No. of VM types and servers are varied

Fig. 6 shows that the heuristics parametrized by small values for fast computation, produce output that is up to 90% optimal in several cases. FS-2 always produces a better quality solution than FS-1 since it encompasses the solution space of the latter. LB-(1,2) is seen to better in three of the four cases highlighting the utility in committing to a small portion of the myopic view rather than its entirety.

Finally, we show that our MILP solution for the resource allocation phase computes fast for large problem sizes. Table 3 shows that the computation runtime scales with No. of VM types and servers are varied, however, with a cut-off time of one-hour, it can handle problems of size 100 servers, 30 VM types.

5 DISCUSSION

For mitigating the dynamic DDoS attacks, we propose a model that consists of two phases. The first phase computes a dynamic defense strategy, whereas the second phase optimally deploys compute resources to implement the strategy. We show analytical results for the minimax equilibria of the static (single-stage) game and subsequently provide Mixed Integer Linear Program (MILP) formulations for computing the optimal dynamic strategies of the multiple-stage game as well as efficient and effective heuristics. Finally, we show numerical results on simulated attack data which demonstrate the efficacy of out solutions approaches.

Future efforts advancing such a model may consider additional constraints on the attacker strategy, such as attack volume bounds for avoiding detection. Another direction is to relax the assumption on the defender's precise knowledge of the attacker model coefficients, and compute robust solutions that do not rely on it, integrating possible uncertainties therein.

REFERENCES

[1] T. Alpcan and T. Başar. 2010. *Network security: A decision and game-theoretic approach*.

[2] AT&T. 2013. *AT&T Domain 2.0 Vision White Paper*. https://www.att.com/Common/about_us/pdf/AT&T%20Domain%202.0%20Vision%20White%20Paper.pdf.

[3] AT&T. 2014. *AT&T and Intel: Transforming the Network with NFV and SDN*. https://www.youtube.com/watch?v=F55pHxTeJLc#t=76.

[4] Paul Barford, Jeffery Kline, David Plonka, and Amos Ron. 2002. A signal analysis of network traffic anomalies. In *Proceedings of the 2nd ACM SIGCOMM Workshop on Internet measurment*. 71–82.

[5] Cloudflare. 2004. *Cloudflare*. https://www.cloudflare.com/ddos.

[6] John Donovan. 2014. *ONS 2014 Keynote*. http://bit.ly/1RQFMko.

[7] K. Durkota, V. Lisỳ, B. Bosanskỳ, and C. Kiekintveld. 2015. Optimal Network Security Hardening Using Attack Graph Games.. In *IJCAI*.

[8] Stuart Elby. 2012. *Verizon-Carrier Adoption of Software-defined Networking*. https://www.youtube.com/watch?v=WVczl03edi4.

[9] NFVISG ETSI. 2012. Network Functions Virtualization-Introductory White Paper. In *SDN and OpenFlow World Congress*.

[10] Fei Fang, Peter Stone, and Milind Tambe. 2015. When security games go green: Designing defender strategies to prevent poaching and illegal fishing. In *Twenty-Fourth International Joint Conference on Artificial Intelligence*.

[11] Seyed K Fayaz, Yoshiaki Tobioka, Vyas Sekar, and Michael Bailey. 2015. Bohatei: Flexible and elastic ddos defense. In *24th {USENIX} Security Symposium ({USENIX} Security 15)*. 817–832.

[12] Albert Greenberg, Gisli Hjalmtysson, David A Maltz, Andy Myers, Jennifer Rexford, Geoffrey Xie, Hong Yan, Jibin Zhan, and Hui Zhang. 2005. A clean slate 4D approach to network control and management. *ACM SIGCOMM Computer Communication Review* 35, 5 (2005), 41–54.

[13] Adam Kalai and Santosh Vempala. 2005. Efficient algorithms for online decision problems. *J. Comput. System Sci.* 71, 3 (2005), 291–307.

[14] M. S. Kang, S. B. Lee, and V. D. Gligor. 2013. The Crossfire Attack. In *2013 IEEE Symposium on Security and Privacy*. 127–141.

[15] A. Laszka, Y. Vorobeychik, and X. D. Koutsoukos. 2015. Optimal Personalized Filtering Against Spear-Phishing Attacks.. In *AAAI*.

[16] Nick McKeown, Tom Anderson, Hari Balakrishnan, Guru Parulkar, Larry Peterson, Jennifer Rexford, Scott Shenker, and Jonathan Turner. 2008. OpenFlow: enabling innovation in campus networks. *ACM SIGCOMM Computer Communication Review* 38, 2 (2008), 69–74.

[17] NETSCOUT. 2019. *NETSCOUT THREAT INTELLIGENCE REPORT*. https://www.netscout.com/threatreport.

[18] R. Pıbil, V. Lisỳ, C. Kiekintveld, B. Bošanskỳ, and M. Pechoucek. 2012. Game theoretic model of strategic honeypot selection in computer networks. *Decision and Game Theory for Security* (2012).

[19] Prolexic. 1995. *Prolexic*. http://www.prolexic.com/.

[20] Christian Rossow. 2014. Amplification Hell: Revisiting Network Protocols for DDoS Abuse.. In *NDSS*.

[21] A. Schlenker, O. Thakoor, H. Xu, F. Fang, M. Tambe, L. Tran-Thanh, P. Vayanos, and Y. Vorobeychik. 2018. Deceiving Cyber Adversaries: A Game Theoretic Approach. In *AAMAS*. http://dl.acm.org/citation.cfm?id=3237383.3237833

[22] A. Schlenker, H. Xu, M. Guirguis, C. Kiekintveld, A. Sinha, M. Tambe, S. Sonya, D. Balderas, and N. Dunstatter. 2017. Don‘t Bury your Head in Warnings: A Game-Theoretic Approach for Intelligent Allocation of Cyber-security Alerts. (2017).

[23] E. Serra, S. Jajodia, A. Pugliese, A. Rullo, and VS Subrahmanian. 2015. Pareto-optimal adversarial defense of enterprise systems. *ACM Transactions on Information and System Security (TISSEC)* 17, 3 (2015), 11.

[24] Ahren Studer and Adrian Perrig. 2009. The coremelt attack. In *European Symposium on Research in Computer Security*. Springer, 37–52.

[25] Omkar Thakoor, Milind Tambe, Phebe Vayanos, Haifeng Xu, Christopher Kiekintveld, and Fei Fang. 2019. Cyber Camouflage Games for Strategic Deception. In *GameSec*.