DDoS Mitigation in Non-Cooperative Environments

Guanhua Yan and Stephan Eidenbenz*

Information Sciences (CCS-3)
Los Alamos National Laboratory
Los Alamos, NM 87544, USA
{ghyan, eidenben}@lanl.gov

Abstract. Distributed denial of service (DDoS) attacks have plagued the Internet for many years. We propose a system to defend against DDoS attacks in a non-cooperative environment, where upstream intermediate networks need to be given an economic incentive in order for them to cooperate in the attack mitigation. Lack of such incentives is a root cause for the rare deployment of distributed DDoS mitigation schemes. Our system is based on game-theoretic principles that provably provide incentives to each participating AS (Autonomous Systems) to report its true defense costs to the victim, which computes and compensates the most cost-efficient (yet still effective) set of defenders ASs. We also present simulation results with real AS-level topologies to demonstrate the economic feasibility of our approach.

1 Introduction

The distributed denial of service (DDoS) attack is a formidable problem that has plagued the Internet for many years. The intent of such attacks is to exhaust the resources of a remote host or network such that the service provided to legitimate users is degraded or denied. As a response to the serious threat posed by DDoS attacks, a plethora of target-resident DDoS mitigation techniques have been proposed, including packet filtering, anti-spoofing, anomaly detection, protocol analysis, and rate limiting. Albeit commercial products providing these solutions are available, it is still a challenging undertaking for the victim to successfully defend against a large-scale DDoS attack all by itself. The reasons are two-fold. First, although the victim is at the best position to detect a DDoS attack, distinguishing traffic with good and ill intentions is not easy. A smart attacker can complicate the defense by making the attack traffic behave similar to legitimate traffic. Second, even if the victim has the perfect technology to characterize malicious traffic, it may not have the computational resources to execute it at the same speed as the arriving attack traffic. This turns the DDoS mitigation component itself into a target of a DDoS attack.

Some earlier work has suggested pushing the perimeter of DDoS defense to intermediate networks [16]. None of these techniques, however, are widely deployed, due to lack of economic and social incentives for the intermediate networks [14]. In many cases, intermediate networks do not suffer directly from a

* Los Alamos National Laboratory Publication No. LA-UR-07-3012

DDoS attack, and are thus reluctant to devote a non-trivial amount of resources to help the victim. Even during some flooding DDoS attacks in which traffic traversing through these networks increases significantly, bandwidth overprovision prepared for flash crowds helps them absorb such types of attacks.

Motivated by this observation, we develop an economically sound framework that mitigates DDoS attacks in *non-cooperative* environments. We assume that intermediate networks (or ASs) are selfish, and they are willing to defend against the DDoS attack for the victim only when they receive enough economic compensation. Put in a game theoretic context, the distributed DDoS mitigation problem triggers the following questions: *What ASs should be selected for defending against the attack*? *How much should the victim pay to each participating AS for the defense*? *How can we design a protocol that stimulates each AS to truthfully report its real cost on the defense other than an arbitrarily high value without economic basis? Can the victim impose a limit on how much money it is willing to pay for the defense in the protocol*?

Our approach to addressing these questions in an economically and game-theoretically sound manner, is as follows: Using a combination of established tools to compute DDoS attack graphs, we make ASs on attack paths report defense cost estimates. Based on these estimates, we compute the most cost-efficient set of defender ASs. Using the game-theoretic paradigm of VCG-type payment schemes ensures that each defense participant maximizes its own utility but truthfully reporting its real cost estimate. In the parlance of game theory, our protocols are provably *truthful* and also satisfy *individual rationality*. We evaluate our protocol on a real Internet-scale AS-level topology. Experimental results reveal that under DDoS attacks launched from botnets of typical sizes observed so far, the inevitable economic inefficiency of our protocol is limited within only 20% of the total payment to the participating defenders if the deployment ratio of the distributed defense scheme reaches 60%.

Related work. The concept of game theory, especially mechanism design, has been used in routing protocol design for communication networks, such as inter-domain routing [9] and Ad hoc-VCG for mobile ad-hoc networks [5]. On the other hand, Mirkovic et al. presented a comprehensive survey on solutions to defending against DDoS attacks in [14]. Huang et al. suggested that currently providing incentives for participating parties in the DDoS defense should be given higher priority than improving the defense technology [11]. Their arguments are in agreement with the motivation behind this paper.

The remainder of this paper is organized as follows. Section 2 discusses how the victim constructs the defense graph. Section 3 presents how to select defender ASs. Sections 4 and 5 introduce the payment schemes. Section 6 evaluates the economic inefficiency and Section 7 concludes this paper.

2 Defense Graph Generation

We consider an attack model in which many compromised hosts are used to launch a large-scale DDoS attack against a server providing a network-accessible service, such as online banking, online game, or database query. As the network and computation resources (e.g., bandwidth, CPUs, storage, software licenses) supporting these services are usually limited, an attacker can convene a large army of zombie machines (e.g., using a botnet) to deplete these resources and thus significantly degrade the QoS (Quality of Services) experienced by legitimate clients. We do not assume that the victim can always easily distinguish malicious traffic from benign traffic. A smart attacker can make the malicious traffic behave similar to that from a legitimate user. We assume that to achieve this, the attacker has to use real source IP addresses in his attack traffic[1].

Our work requires that the defense graph is constructed at the victim site. To do this, we leverage existing tools and methods as follows. *First*, when the target server becomes overloaded, incoming source IP addresses are randomly sampled. For each sampled packet, we use the *network-aware clustering* technique described in [12] to derive its origin AS. It has been shown that this method is able to cluster 99% of the web clients. For each clustered IP prefix, we can look up its corresponding AS number from a map, which can be constructed offline or obtained from a third party such as Team Cymru [1]. *Second*, the victim decides V_{or}, the set of origin ASs from which traffic should be inspected based on the traffic rate from each AS and how much the current traffic rate from an AS exceeds the average traffic rate observed in the past. *Third*, the victim infers AS-level paths from ASs in V_{or} to the victim network using tools such as RouteScope [13]. Combining all the AS-level paths inferred from ASs in V_{or} to the victim network, the victim can build the defense graph $G(V, E)$, in which nodes are denoted by A_1, A_2, ..., and $A_{|V|}$.

The aforementioned approach to constructing defense graphs differs from existing solutions to attack graph generation [16] that use the probabilistic packet marking (PPM) scheme. PPM, which requires intermediate routers to mark traversing packets probabilistically, has a few drawbacks here. First, it is not immune to edge faking, standard IP stack attack, and near-birthday collision attack [19]. Second, intermediate routers can erase marks inscribed by upstream ones to increase its chance being selected as a defender. Third, deriving a complete attack graph at the victim site requires all ASs to perform PPM, which is difficult from a deployment point of view.

After constructing the defense graph, the victim does the following: **(1)** contact the *defense proxy* (explained later) of each AS in the defense graph, querying the defense price if that defense proxy deploys the defense scheme requested by the victim; **(2)** select the most cost efficient set of ASs; **(3)** calculate the payment made to the owner of each AS that is selected for distributed defense; **(4)** send the payment to the owner of each AS selected for distributed defense.

Once an AS that is selected for distributed defense receives the payment, it forwards all traffic destined to the target server to the defense proxy. A defense proxy performs requested defense action by the victim such as packet filtering and anti-spoofing. To alleviate the high workload on the server, the victim can even delegate part of its service to the defense proxies. In the following sections, we explain the details of Steps (1) through (4).

[1] Actually source address spoofing is rarely used these days in DDoS attacks [18].

3 Defender Selection

Phase I: Requesting Defense Costs

After constructing the defense graph, the victim sends a request to each AS in it, asking distributed defense against the DDoS attack. We assume that an AS can forward its traffic to its defense proxy, if it has one, at two places: links connected to end hosts in its local administrative domain, and outbound links to other ASs (if the victim does not belong to the AS) or links directly connected to the victim (if the victim belongs to the AS). We use $L_{loc}(k)$ and $L_{out}(k)$ to denote these two sets of links in AS A_k respectively. We also assume that DDoS attack traffic originates only from end hosts. For ease of presentation, we let v denote the address of the target server. If AS A_k forwards traffic with destination v to its defense proxy on links in $L_{out}(k)$, it does not need to do it on links in $L_{loc}(k)$, because any locally generated traffic with destination v, if not dropped inside AS A_k, must traverse some link in $L_{out}(k)$ before reaching the victim network.

Each AS A_k in set V_{or} under request calculates two costs: one is associated with forwarding traffic with destination v from links in $L_{loc}(k)$ to its defense proxy and then performing requested action on the traffic there, and the other associated with forwarding traffic with destination v from links in $L_{out}(k)$ to its defense proxy and then performing requested action on the traffic there. Let $C_{loc}(k)$ and $C_{out}(k)$ denote these two costs respectively. If AS A_k is in set $V - V_{or}$, it only computes $C_{out}(k)$ and its $C_{loc}(k)$ is automatically 0. In cases that AS A_k does not have a defense proxy, both its $C_{out}(k)$ and $C_{loc}(k)$ are $+\infty$. We further define $\boldsymbol{C}(k)$ as a 2-tuple vector: $\langle C_{loc}(k), C_{out}(k)\rangle$, and call it the *genuine cost vector* of AS A_k. $\boldsymbol{C}(k)$ is private information owned by AS A_k, and in the parlance of game theory, it is called the *type* of AS A_k.

Here, we do not impose specific formula on how each AS should compute its genuine cost vector. Actually, such estimated costs rely on many factors such as performance degradation due to requested defense and the service level agreements (SLAs) between the AS and its customers. Instead, our protocol only assumes the existence of such costs and it works independently of how ASs estimate these costs. Moreover, we also assume that such cost estimation can be done quickly (e.g., performed automatically without human intervention) by each AS. We will explore how to achieve this in our future work.

Meanwhile, each AS A_k is requested to report its genuine cost vector to the victim. As an AS may lie about its true costs, we use $R_{loc}(k)$ and $R_{out}(k)$ to denote the two reported costs respectively. Similarly, we define, $\boldsymbol{R}(k)$, the *reported cost vector* of AS A_k, as 2-tuple vector: $\langle R_{loc}(k), R_{out}(k)\rangle$.

Phase II: Computing Defense Set

After the victim receives all the reported cost vectors from the ASs in the defense graph, it decides which ASs should be selected for defending against the DDoS attack. Recall that an AS A_k can perform the requested action on traffic going through links in either $L_{loc}(k)$ or $L_{out}(k)$. Hence, the victim should also determine traffic on which links an AS, if selected for defense, should check according to the requested action. Define a *defense set* D as a set of 2-tuple $\langle A_i, p_i\rangle$, where $A_i \in V$ and $p_i \in \{0, 1\}$. If $p_i = 0$, AS A_i performs the requested action on

$L_{loc}(k)$; if $p_i = 1$, AS A_i performs the requested action on $L_{out}(k)$. We further define $\Gamma(D)$, a set of network links, as follows:

$$\Gamma(D) = \{l \mid (\forall \langle A_i, 0\rangle \in D : l \in L_{loc}(i)) \vee (\forall \langle A_i, 1\rangle \in D : l \in L_{out}(i))\} \qquad (1)$$

In other words, set $\Gamma(D)$ consists of all the links on which traffic is inspected under defense set D. We use $P(q)$ to denote the set of links on the path of packet q. We define the *completeness* of defense set D as follows:

Definition 1. *Defense set D is complete relative to AS set S if for any packet q that originates from an AS in S and goes to the target server, there exists a link l in $P(q)$ such that $l \in \Gamma(D)$.*

From the economic point of view, it is reasonable to assume that the victim wants to find the most cost-efficient defense set that is also complete. Given a defense set D, we define its cost $\mathcal{H}(D)$ as follows:

$$\mathcal{H}(D) = \sum_{\{A_i, p_i\} \in D} R_{loc}(k) \cdot (1 - p_i) + R_{out}(k) \cdot p_i. \qquad (2)$$

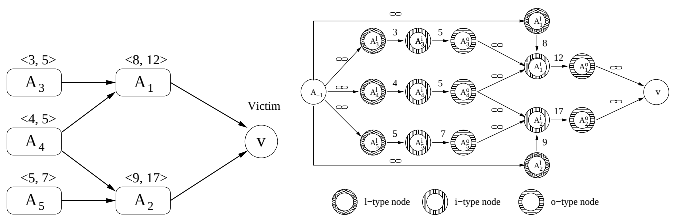
We further give the following definition:

Definition 2. *A defense set D that is complete relative to AS set S is efficient relative to the reported costs if for any other defense D' that is also complete relative to AS set S, it holds $\mathcal{H}(D') \geq \mathcal{H}(D)$.*

We now present an algorithm that computes a defense set that is complete relative to a subset of V_{or} and efficient relative to the reported costs. D_{min} is initialized to be empty. The algorithm consists of two steps:

Step 1: generate mutated defense graph G_m. We first create a new graph $G_m(V_m, E_m)$, which is called the *mutated defense graph*, based on defense graph $G(V, E)$ and reported costs from each AS in V. For each AS node A_k in V_{or}, we create three nodes in G_m, which are denoted by A_k^i, A_k^o, and A_k^l respectively; we also add directed edges (A_k^i, A_k^o) with weight $R_{out}(k)$ and (A_k^l, A_k^i) with weight $R_{loc}(k)$ to G_m. For each AS node A_j in $V - V_{or}$, we add only two nodes into G_m, which are A_j^i and A_j^o; we add directed edge (A_k^i, A_k^o) with weight $R_{out}(k)$ to G_m. For each directed edge (A_i, A_j) in the original defense graph, we add a directed edge (A_i^o, A_j^i) with weight ∞ to G_m. We also put the victim node v into graph G_m; for each AS node A_i in V that is directly connected to v, we add edge (A_i^o, v) with weight ∞ into G_m. Finally, we add to G_m a virtual source node A_{-1} and a directed edge with weight ∞ to every node A_i^l with upper index l in G_m whose corresponding AS node A_i is in set V_{or}. In G_m, we call a node with upper index i an *i-type* node, a node with upper index o an *o-type* node, and a node with upper index l an *l-type* node. Similarly, we call an edge from an i-type node to an o-type node an *i-o-type* edge, an edge from an l-type node to an i-type node an *l-i-type* edge, and an edge from an o-type node to an i-type node an *o-l-type* edge. In Fig. 1, we illustrate the mutated defense graph generated from an defense graph and the reported costs from each AS.

Step 2: compute the min-cut of G_m. We treat the mutated defense graph as a flow network with source A_{-1} and sink v; the weight of each directed link is its capacity. It is possible that on some paths from A_{-1} to v all the edges have infinite weights; this occurs when no ASs on these paths deploy



(1) Original defense graph. (2) Mutated defense graph

Fig. 1. Original defense graph and mutated defense graph. The reported cost vector of each AS is shown on the corresponding node.

defense proxies. We preprocess G_m as follows: use the DFS (Depth First Search) algorithm to traverse graph G_m by starting from sink v and traversing the graph only along edges with infinite weights in reverse direction; whenever an l-type node u is visited, we add its corresponding AS node A_k to set U and also erase the l-i-type edge associated with it from G_m. It is not difficult to see that after preprocessing G_m as described, G_m must have a min-cut with a finite capacity.

The celebrated max-flow min-cut theorem [6] states that the maximal amount of a flow is equal to the capacity of a minimal cut. To compute the min-cut of G_m, we first derive the maximum flow f_m^{max} in G_m. Given the max-flow f_m^{max}, we can obtain set E_{mc}, which contains all the cutting edges in G_m that it saturates. Then, the min-cut of G_m can be obtained by cutting all the edges in E_{mc}.

Based on E_{mc}, the victim can derive a defense set that is complete relative to AS set $V_{or} - U$ and efficient relative to the reported costs. As only l-i-type and i-o-type edges have finite capacities in G_m, E_{mc} contains only such edges. For each l-i-type edge (A_u^l, A_u^i) in E_{mc}, we add $\langle A_u, 0\rangle$ to D_{min}; For each i-o-type edge (A_u^i, A_u^o) in E_{mc}, we add $\langle A_u, 1\rangle$ to D_{min}. In the example shown in Fig. 1, the capacity of the min-cut is 29 and E_{mc} is $\{(A_1^i, A_1^o), (A_2^i, A_2^o)\}$. The final D_{min} is thus $\{\langle A_1, 1\rangle, \langle A_2, 1\rangle\}$.

Based on the property of the min-cut, we can easily establish the following theorem (proof omitted):

Theorem 1. *The defense set found by the algorithm as described is complete relative to set $V_{or} - U$ and efficient relative to the reported costs.*

4 Payments Without Reserve Price

In this section, we consider the cases in which the victim does not impose a constraint on how much money it is willing to pay for the defense. Such a constraint is sometimes called *reserve price* [7]. A naive payment scheme is that the victim pays every AS in the final defense set by its reported cost. However, in a non-cooperative computation environment, there is no reason to believe that each AS truthfully reports its true costs. To prevent cheating, we borrow the payment scheme from the Vickrey-Clarke-Groves (VCG) mechanisms [5].

We say that we *disable* AS node $A_k \in V$ from graph G_m by doing the following: if edge $(A_k^l, A_k^i) \in E_m$, we set its capacity to be $+\infty$, and if edge $(A_k^i, A_k^o) \in E_m$, we set its capacity to be $+\infty$. The new graph after disabling node A_k from G_m is denoted by $G_{m,-A_k}$. Then, for each item $\langle A_k, p_k\rangle$ in D_{min}, the payment to AS A_k's owner $\omega(A_k)$, denoted by $\eta(\omega(A_k))$, is given by

$$\eta(\omega(A_k)) = (1-p_k)\cdot R_{loc}(k) + p_k \cdot R_{out}(k) + \Upsilon(G_{m,-A_k}) - \Upsilon(G_m) \qquad (3)$$

In other words, AS A_k, if selected for defense, is paid by the difference between the cost of the most cost efficient defense set after A_k is disabled and the cost of the most cost efficient defense set without its own reported cost.

From a practical point of view, there are two problems with the above payment scheme. First, an economic entity (e.g., a big ISP) can own multiple ASs and as the above payment scheme treats each AS as an *agent* in the game, multiple agents belonging to the same economic entity can collude to achieve better economic benefits. We thus call it the *collusion attack problem.* It is also well known that the VCG-type payment scheme does not prevent collusion attacks [8]. To solve this problem, we slightly modify the above payment scheme. We treat each independent economic entity as an agent. Let Φ denote the entire set of economic entities or agents involved in the distributed defense, and $\Psi(s)$, where $s \in \Phi$, to denote the set of ASs that agent s controls. We also use $\Upsilon(G_{m,-S})$ to denote the capacity of the min-cut after disabling *all* the AS nodes in set S from graph G_m. The total payment made to agent s is given as follows:

$$\eta(s) = \sum_{\forall A_k \in \Psi(s): \langle A_k, p_k\rangle \in D_{min}} ((1-p_k)\cdot R_{loc}(k) + p_k \cdot R_{out}(k)) + \Upsilon(G_{m,-\Psi(s)}) - \Upsilon(G_m).$$

The difference $\Upsilon(G_{m,-\Psi(s)}) - \Upsilon(G_m)$ is the overpayment made to agent s (relative to its reported cost); it is also called the *premium* paid to agent s.

Second, after disabling all AS nodes belonging to an agent s, the min-cut $\Upsilon(G_{m,-\Psi(s)})$ may become infinity. This means that the victim has to pay an infinite amount of money to agent s, which is obviously not realistic. We call it the *monopolistic extortion problem.* To circumvent it, we modify the mutated defense graph G_m to satisfy the *1-agent resilience property*: for any path from node A_{-1} to victim node v, there are at least two edges that have finite weights and the nodes they are incident to belong to different agents. To achieve this property, we simply remove the first l-i-type edge on *every* path that does not satisfy this condition and put the AS node associated with that edge into set U.

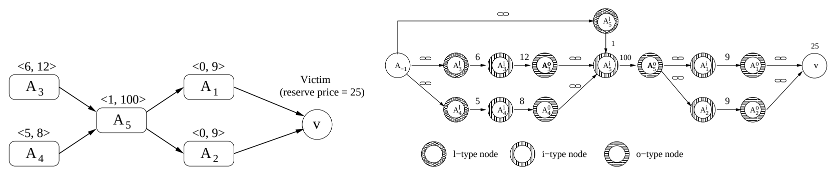
Following the example in Fig. 1, suppose there are three agents, s_1, s_2, and s_3. $\Psi(s_1)$, $\Psi(s_2)$, and $\Psi(s_3)$ are $\{A_3, A_4\}$, $\{A_1, A_5\}$, and $\{A_2\}$ respectively. To satisfy 1-agent resilience property, we remove the two edges from A_1^l to A_1^i and from A_2^l to A_2^i, and the new defense set D_{min} thus becomes $\{\langle A_3, 0\rangle, \langle A_4, 0\rangle, \langle A_5, 0\rangle\}$. The payments made to s_1, s_2 and s_3 are 12, 17, and 0 respectively.

The *utility* of an agent is defined as the payment it receives from the victim less its *true* cost. Let $u(s)$ be the utility of agent s. Following the same example, we have $u(s_1) = 5$, $u(s_2) = 12$, and $u(s_3) = 0$. In the parlance of game theory, our protocol is *truthful* (or *strategyproof*) only if *truth-telling* is a dominant strategy for each participating agent; that is to say, if an agent wants to maximize its utility, it must report truthfully its genuine cost vector for each AS that it owns, independently of any other agent's report. The individual rationality of a protocol means that an agent participates in the game only when its utility after participation is as much as that without participation. We then have the following theorem (proof provided in Appendix A):

Theorem 2. *If the victim does not have a reserve price and the protocol as described is executed, behaving truthfully is a dominant strategy and individually rational for any agent.*

5 Payments with Reserve Price

In this section we consider a more realistic setting in which the victim specifies its reserver price, the maximum amount of money it wants to pay for the distributed defense. We use P_r to denote the reserve price of the victim. P_r is the type of the victim, and the utility of the victim is P_r minus the total payments made by the victim. Let P_d be the total payment made to all the agents owning ASs in D_{min}. A naive solution to deciding whether a deal should be made between the victim and the agents owning ASs in D_{min} is: If $P_r \geq P_d$, the deal is made; otherwise, the deal fails. Although intuitively reasonable, such a decision rule can lead to cheating behavior by some agents. We use a simple example in Fig. 2 to illustrate that. Each AS belongs to a different agent. D_{min} is $\{\langle A_3, 0\rangle, \langle A_4, 0\rangle, \langle A_5, 0\rangle\}$. We thus have $\eta(\omega(A_3)) = 12$, $\eta(\omega(A_4)) = 11$, and $\eta(\omega(A_4)) = 7$. Since $P_d = 30 > 20$, the defense deal cannot be made. Now suppose that A_3 reports its cost vector as $\langle 11, 12\rangle$ instead of $\langle 6, 12\rangle$. Clearly, D_{min} does not change. $\eta(\omega(A_3)) = 12$, $\eta(\omega(A_4)) = 6$, and $\eta(\omega(A_5)) = 2$. Hence, $P_d = 20 < 25$, which makes the deal take place. The utility of A_3 changes from 0 to 6 after it cheats.



(1) Original defense graph (2) Mutated defense graph

Fig. 2. Untruthfulness with reserve price

To solve this problem, we borrow the idea of global replacement from [7]. Define set Φ_{min} as follows:

$$\Phi_{min} = \{s | \exists \langle A_k, p_k\rangle \in D_{min} : A_k \in D_{min}\}. \tag{4}$$

That is to say, Φ_{min} contains every agent that has at least one AS selected into the defense set D_{min}. Then, for *every* agent s in Φ_{min}, we disable all the ASs that it owns from graph G_m. Let the new graph be G_m^r. We then use the algorithm described in Section 3 to recompute the defense set from G_m^r. We denote the new defense set by D_{min}^r and call it the *global replacement defense set*. We define P_d^{-mc} as the total payment made to all the agents owning ASs in D_{min}^r. The decision of the victim depends on P_r and P_d^{-mc}: if $P_r \geq P_d^{-mc}$, the deal takes place and the payment to each agent is the same as computed in Section 4; otherwise, the deal does not take place. In the example, if all ASs report their genuine cost vector truthfully, we have $P_d^{-mc} = 18 < P_r = 25$; therefore, the distributed defense takes place and the payments to $\omega(A_3)$, $\omega(A_4)$ and $\omega(A_5)$ are 12, 11 and 7 respectively. If $\omega(A_3)$ falsely reports the cost vector of AS A_3 as $\langle 11, 12\rangle$, its utility remains unchanged at 6.

As the victim is only willing to pay P_d^{-mc}, the budget is no balanced if $P_d^{-mc} \neq P_d$. In the example, the victim pays only $P_d^{-mc} = 18$ for the defense, but the total payment made to all the ASs is $P_d = 30 > 18$. To overcome such a side effect, we introduce the *central bank model* [10], in which each AS has a bank account at a central bank and the bank periodically credits or debit each agent to balance its budget for the differences between P_d^{-mc} and P_d. Such a central bank can be managed by a trusted authority, such as IANA, which allocates unused AS numbers in the Internet.

Theorem 3 establishes the truthfulness and individual rationality of the protocol when the victim has a reserve price (proof provided in [20]):

Theorem 3. *If the victim has a reserve price and the protocol as described is executed, behaving truthfully is a dominant strategy and individually rational for both the victim and any participating agent.*

6 Experiments

Setup. In our experiments, we use a real Internet topology derived from BGP routing table dump files. From a BGP RIB file downloaded from the Route Views Project [2], we extract 25,193 AS numbers. The size of each AS is roughly estimated by adding the number of IPv4 addresses covered by all the prefixes announced by it. We choose a server that resides in AS 18784 as the target of a DDoS attack. We use the AS path inference service provided by the BGPVista project [3].To obtain a map from each AS to its corresponding economic entity, we collect the information of each AS from the FixedOrbit website [4]. If two ASs have close names, we assume that they belong to the same economic entity. We identify 23,075 agents, among which 99.77% have only one AS. To obtain a map from each AS to its corresponding economic entity, we collect the information of each AS from the FixedOrbit website [4]. If two ASs have close names, we assume that they belong to the same economic entity.

We also vary the sizes of the botnets that are used to launch a DDoS attack against the target server among 10^4, 10^5, and 10^6. These values are based on the sizes of today's botnets [15]. For each bot, we let its sending rate be 130kbps, according to the survey from [17]. In the experiments, we assume that bots are uniformly distributed in the IPv4 addresses owned by all the ASs under consideration. Moreover, when selecting a list of ASs from which traffic should be inspected, we do not consider normal traffic in our experiments, because this does not keep us from explaining the basic principle of our protocol. Instead, when the DDoS traffic generated from an AS exceeds a certain threshold θ, this AS is added into set V_{or}. We set θ to be 10Mbps. We assume that the genuine cost of an AS A_k on inspecting traffic on links in either $L_{loc}(k)$ or $L_{out}(k)$ is a linear function of the traffic destined to the victim through these links:

$$\begin{cases} C_{loc}(k) = a_{loc}(k) \cdot T_{loc}^v(k) + b_{loc}(k) \\ C_{out}(k) = a_{out}(k) \cdot T_{out}^v(k) + b_{out}(k) \end{cases} \tag{5}$$

where $T_{loc}^v(k)$ and $T_{out}^v(k)$ denote the total traffic rate destined to the victim v through links in $L_{loc}(k)$ and $L_{out}(k)$ respectively, coefficients $a_{loc}(k)$ and $a_{out}(k)$

are uniformly chosen between 10^{-6} and 5×10^{-6}, and constants $b_{loc}(k)$ and $b_{out}(k)$ are uniformly chosen between 100 and 500.

The reserve price of the victim P_r is given by $c_v \cdot T_v$, where c_v is 10^{-5} and T_v is traffic arrival rate at the victim site. We vary the probability with which an agent deploys defense proxies among 20%, 40%, 60%, 80%, and 100%. If an agent is chosen to deploy defense proxies, *every* AS it owns has a defense proxy.

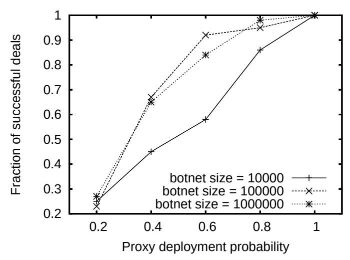


Fig. 3. Fraction of successful deals on distributed defense

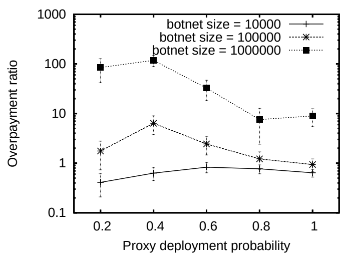


Fig. 4. Overpayment ratio with reserve price (95% conf. interval)

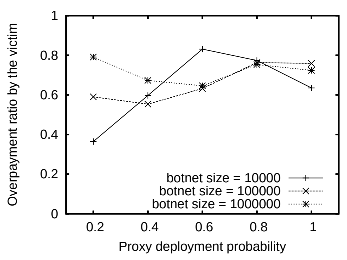


Fig. 5. Overpayment ratio by the victim with reserve price

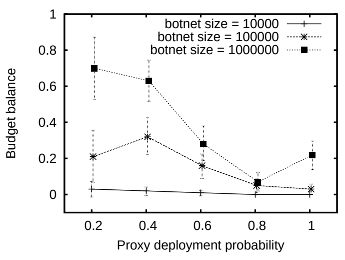


Fig. 6. Budget balance with reserve price (95% conf. interval)

Results. Due to space limitation, we present only the experimental results with reserve prices here. We refer readers to [20] for results without reserve prices. Fig. 3 depicts the fraction of successful deals between the victim and the participating agents. We note that as more agents deploy defense proxies, it is more likely that the distributed defense takes place. This is because with wider defense proxy deployment, there are more choices for the global replacement defense set, which helps reduce P_d^{-mc}.

From the economic point of view, it is interesting to know how much the victim needs to overpay the participating agents so that the latter do not cheat. Among the successful deals between the victim and the participating agents, the overpayment ratio is illustrated in Fig. 4. The overpayment ratio is defined as $\frac{P_d - Y}{P_d}$, where P_d is the total payment made to all participating defenders and Y is the true total defense cost incurred by them. The general trend of each curve is that as more agents deploy defense proxies, the overpayment ratio decreases. This is because a higher portion of agents with defense proxies provides more options if a participating agent is disabled from the mutated defense graph, which helps reduce the premium made to that agent. Although the overpayment can be very high, the victim does not need to pay as much to the participating agents because of its reserve price. Fig. 5 presents the overpayment ratio that is contributed *only* by the victim: the largest average overpayment ratio by the victim is 0.83, much less than the highest overall overpayment ratio (about 100).

The economic model incorporating the victim's reserve price requires a central bank to pay the overpayment that is not contributed by the victim. To balance its own budget, the central bank can impose a tax upon the agents [10]. Fig. 6 depicts the *budget balance*, defined as $(P_d - P_d^{-mc})/P_d$. The graph suggests that when there are not many agents deploying defense proxies and the DDoS attack is launched from a million-host botnet, the budget balance of the central bank can be as high as 70% of the total payment. However, million-host botnets are still rare so far [15], and for smaller botnets, if at least 60% of the agents deploy defense proxies, the average budget balance is below 20%.

7 Conclusion

Our work in this paper is motivated by the observation that lack of social or economic incentive is one of the root causes that no distributed defense scheme has been widely deployed against DDoS attacks. Assuming that autonomous systems behave selfishly in defending against DDoS attacks. We propose a protocol that provides incentives for these ASs to participate in cooperative defense. Our protocol is provably truthful and also satisfies individual rationality.

References

1. http://asn.cymru.com/.
2. http://www.routeviews.org/.
3. http://www.bgpvista.com/.
4. http://www.fixedorbit.com/.
5. L. Anderegg and S. Eidenbenz. Ad hoc-VCG: a truthful and cost-efficient routing protocol for mobile ad hoc networks with selfish agents. In *Proceedings of MobiCom'03*, September 2003.
6. T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein. *Introduction to Algorithms.* The MIT Press and McGraw-Hill, 2001.
7. S. Eidenbenz, G. Resta, and P. Santi. COMMIT: A sender-centric truthful and energy-efficient routing protocol for ad hoc networks with selfish nodes. In *Proceedings of IPDPS'05*, volume 13, April 2005.
8. J. Feigenbaum, A. Krishnamurthy, R. Sami, and S. Shenker. Approximation and collusion in multicast cost sharing. In *Proceedings of EC'01*, 2001.
9. J. Feigenbaum, C. Papadimitriou, R. Sami, and S. Shenker. A BGP-based mechanism for lowest-cost routing. In *Proc. of PODC'02*, 2002.
10. J. Green and J. Laffont. Incentives in public decision making. *Studies in Public Economies*, 1:65–78, 1979.
11. Y. Huang, X. Geng, and A. B. Whinston. Defeating DDoS attacks by fixing the incentive chain. *ACM Trans. on Internet Technology*, 2006.
12. B. Krishnamurthy and J. Wang. On network-aware clustering of web clients. In *Proceedings of SIGCOMM'00*, 2000.
13. Z. M. Mao, L. Qiu, J. Wang, and Y. Zhang. On AS-level path inference. In *Proceedings of SIGMETRICS'05*, Banff, Alberta, Canada, June 2005.

14. J. Mirkovic and P. Reiher. A taxonomy of DDoS attack and DDoS defense mechanisms. *ACM SIGCOMM Computer Communications Review*, 34(2), April 2004.
15. Honeynet Project and Research Alliance. Know your enemy: Tracking botnets. http://www.honeynet.org/papers/bots/, 2005.
16. S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *Proceedings of ACM SIGCOMM'00*, 2000.
17. K. K. Singh. Botnets – an introduction. http://www-static.cc.gatech.edu/classes/AY2006/cs6262_spring/botnets.ppt.
18. Summary of the initial meeting of the dos-resistant internet working group. http://www.thecii.org/dos-resistant/meeting-1/summary.html, January 2005.
19. M. Waldvogel. GOSSIB vs. IP traceback rumors. In *Proc. of ACSAC'02*.
20. G. Yan and S. Eidenbenz. Distributed DDoS mitigation in non-cooperative environments. Technical Report LAUR-07-3012, Los Alamos National Lab, 2007.

Appendix A: Proof of Theorem 2

In Eq. (3), as $\Upsilon(G_{m,-\Psi(s)})-\Upsilon(G_m)$ cannot be negative, the utility of each agent cannot be negative. Hence, our protocol is individually rational.

Now we show it is the best interest of each agent to report truthfully its genuine cost vector for each AS it owns. Let Π_{-s} denote an *arbitrary* set of cost vectors reported by all agents except agent s. We use $D^{(s)}_{min}(\Pi_{-s}, \boldsymbol{X})$ to denote the defense set found by the algorithm when all agents except s report their cost vectors according to Π_{-s} and agent s reports its cost vectors as $\boldsymbol{X}$. We also use $G^{(s)}_m(\Pi_{-s}, \boldsymbol{X})$ and $G^{(s)}_{m,-\Psi(s)}(\Pi_{-s}, \boldsymbol{X})$ to denote graphs G_m and $G_{m,-\Psi(s)}$ respectively when all agents except s report their cost vectors according to Π_{-s} and agent s reports its cost vectors according to $\boldsymbol{X}$. Without introducing any confusion, we drop input Π_{-s} in these notations. Let vector $\boldsymbol{X}^s_{rep}$ contains all the reported cost vectors by agent s, and vector $\boldsymbol{X}^s_{gen}$ contains all the genuine cost vectors of the ASs owned by agent s. We also introduce notations $\mathcal{R}(D, s)$ and $\mathcal{C}(D, s)$ as follows:

$$\mathcal{R}(D,s) = \sum_{\forall \langle A_k, p_k\rangle \in D : A_k \in \Psi(s)} (p_k \cdot R_{loc}(k) + (1-p_k)\cdot R_{out}(k))$$

$$\mathcal{C}(D,s) = \sum_{\forall \langle A_k, p_k\rangle \in D : A_k \in \Psi(s)} (p_k \cdot C_{loc}(k) + (1-p_k)\cdot C_{out}(k)).$$

Let $\Delta u(s)$ be the utility of agent s when it reports $\boldsymbol{X}^s_{rep}$ less that when it reports $\boldsymbol{X}^s_{gen}$. We distinguish the following three cases.

C1: If agent s reports $\boldsymbol{X}^s_{gen}$, some ASs in $\Psi(s)$ appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{gen})$, but if agent s reports $\boldsymbol{X}^s_{rep}$, no ASs in $\Psi(s)$ appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{rep})$. Its utility changes from a non-negative value to 0.

C2: If agent s reports $\boldsymbol{X}^s_{gen}$, no ASs in $\Psi(s)$ appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{gen})$, but if agent s reports $\boldsymbol{X}^s_{rep}$, some ASs in $\Psi(s)$ appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{rep})$. Then, $\Delta u(s) = \Upsilon(G^{(s)}_{m,-\Psi(s)}(\boldsymbol{X}^s_{rep})) - (\Upsilon(G^{(s)}_m(\boldsymbol{X}^s_{rep})) - \mathcal{R}(D^{(s)}_{min}(\boldsymbol{X}^s_{rep}), s) + \mathcal{C}(D^{(s)}_{min}(\boldsymbol{X}^s_{rep}), s))$. The second item on the right side of the equation is the capacity of a cut of $G^{(s)}_m(\boldsymbol{X}^s_{gen})$. As no ASs in $\Psi(s)$ appear in D_{min} and the first item is thus the capacity of a min-cut of $G^{(s)}_m(\boldsymbol{X}^s_{gen})$, $\Delta u(s)$ is not positive.

C3: If agent s reports $\boldsymbol{X}^s_{gen}$, some ASs in $\Psi(s)$ appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{gen})$, but if agent s reports $\boldsymbol{X}^s_{rep}$, some ASs in $\Psi(s)$ also appear in $D^{(s)}_{min}(\boldsymbol{X}^s_{rep})$. Then, $\Delta u(s) = \Upsilon(G^{(s)}_m(\boldsymbol{X}^s_{gen})) - (\Upsilon(G^{(s)}_m(\boldsymbol{X}^s_{rep})) - \mathcal{R}(D^{(s)}_{min}(\boldsymbol{X}^s_{rep}), s) + \mathcal{C}(D^{(s)}_{min}(\boldsymbol{X}^s_{rep}), s))$. The second item on the right side of the equation is the capacity of a cut of $G^{(s)}_m(\boldsymbol{X}^s_{gen})$ and the first item is the capacity of a min-cut of $G^{(s)}_m(\boldsymbol{X}^s_{gen})$, so $\Delta u(s)$ is not positive. Combining all cases from C1 to C3, we prove that our protocol is truthful. $\square$