Denial of Service Elusion (DoSE): Keeping Clients Connected for Less

Paul Wood, Christopher Gutierrez, and Saurabh Bagchi
School of Electrical and Computer Engineering
Purdue University
West Lafayette, IN 47907
Email: [pwood,gutier20,sbagchi]@purdue.edu

***Abstract*—Denial of Service (DoS) attacks continue to grow in magnitude, duration, and frequency increasing the demand for techniques to protect services from disruption, especially at a low cost. We present Denial of Service Elusion (DoSE) as an inexpensive method for mitigating network layer attacks by utilizing cloud infrastructure and content delivery networks to protect services from disruption. DoSE uses these services to create a relay network between the client and the protected service that evades attack by selectively releasing IP address information. DoSE incorporates client reputation as a function of prior behavior to stop attackers along with a feedback controller to limit costs. We evaluate DoSE by modeling relays, clients, and attackers in an agent-based MATLAB simulator. The results show DoSE can mitigate a single-insider attack on 1,000 legitimate clients in 3.9 minutes while satisfying an average of 88.2% of requests during the attack.**

I. Introduction

Denial of Service (DoS) attacks are a continually evolving class of attacks that seek to degrade the ability of legitimate clients to utilize computer resources. Defending against this kind of attack has traditionally been the responsibility of large network operators and internet service providers; however these attacks are increasingly impacting smaller networks or even individual users [17]. The observed trend of increasing attack size, duration, and frequency [5] point to failures in the state-of-practice to mitigate such attacks, especially on an infrastructure level. This is increasingly true with the decreased cost of launching a DoS attack combined with the high relative cost of defending against them with commercial solutions. Time-shared DDoS attacks can be had for as little as $12.99 per month [13] while defense can cost $2,000 per month [3, 4, 11] or more. As long as the economic factors favor DoS attacks, they will become increasingly common and persistent occurrences. A low cost solution is needed to prevent stifling of free speech, on the individual side, and to increase the efficiency of doing business, for small to mid-sized businesses.

In this paper, we present our beginning effort at achieving the above goal, a system called Denial of Service Elusion (DoSE). Due to our requirement for low economic cost, we limit the defense solution to methods that do not require any enhancements to the core network infrastructure. Instead, DoSE focuses on using hosting services with widespread and relatively low cost availability such as cloud computing infrastructure and content delivery networks as cornerstones to mitigate attacks. These service have become relatively inexpensive and offer "pay-as-you-go" options which allows flexibility in the mitigation technique. DoSE leverages low cost of public infrastructure-as-a-service (IaaS) cloud and content delivery networks (CDN) to meet an economical cost of roughly $30 a month for DoS protection for 1,000 clients.

The general approach in DoSE is to connect clients to relays, in an overlay network, instead of directly to a protected service so that DoS attacks cannot easily be launched directly at the service. This technique alone is not novel. However, DoSE adds in a smart management layer which acts to conceal relays from attackers and provide an attack-resistant connection establishment mechanism while most importantly minimizing costs. The relays are created on cloud infrastructure as virtual machines, so the number of relays can expand and contract easily to adapt to changing network conditions. Client-to-relay assignments are communicated over a push-based CDN system which allows for fast reassignments during attack periods, unlike traditional domain name systems (DNS), as well as enabling client-specific assignments. The clients are partitioned among relays and each new relay's address is selectively released to clients so that if an attack occurs, a set of suspect clients can be identified. The suspects are then separable from the legitimate clients, and with each attack, the suspicion set can be narrowed down. Suspicious clients can be connected to the same relay so that future attacks impact only a small subset of the users.

Prior work in this area is capable of stopping attacks but fails to address the economic considerations of DoS defense. Techniques, such as Portcullis [19] and Epiphany [9], require Internet-wide infrastructure upgrades to combat attacks. Portcullis relies on the assumption that attackers and legitimate clients have similarly-balanced computing power, which may not hold. Epiphany relies on router upgrades (to support reverse multicast) and the availability of thousands of proxy nodes to defeat DoS (according to *their* experimental setup - Section V-A). MOTAG [7, 8] and other overlay network type techniques [2, 10, 21] fail to address the resiliency of the client assignment or initial connection channel, relying instead on Portcullis[19] and other existing techniques to stop attacks on a management channel. The work in MOTAG [7] and its subsequent work [8] operate moving target defenses

using similar techniques to DOSE. These solutions are not cost-conscious, and call for 1000 active relays for example. Without cost consideration, optimization, or evaluation, they are susceptible to economics-based attacks, whereby the attack exhausts the budget of the consumer for supporting network traffic.

In terms of contributions, DOSE shows how to achieve low cost DDoS attack mitigation for small hosting clients or medium-sized organizations, which have a limited security budget that precludes them from getting a dedicated filtering network or some special arrangements from an ISP. DOSE designs a novel approach for connecting clients to relay proxies and new methods for assigning clients to relays to mitigate network layer attacks while minimizing costs. This work *does not address* attacks capable of disabling large data centers or other large infrastructure networks by well-resourced attackers, or application-layer attacks, i.e., attacks that exhaust the application's capacity by sending a large number of legitimate-looking, but computationally-expensive-to-process requests.

The rest of the paper is structured as follows. In Section II, we provide background information on the different kinds of DoS attacks. In Section III, we give a high-level view of the workings of DOSE, followed by the detailed design in Section IV. In Section VI, we lay out the economic costs of using DOSE versus two existing approaches, one from the commercial domain and the other from the research literature. In Section V, we describe our experiments with varying numbers and capabilities of legitimate and adversarial nodes and measure what fraction of the non-attack traffic can be supported when the service is under attack. In Section VI, the costs of DOSE are analyzed and Section VII presents our conclusions.

II. Background and Related Work

A. Traditional Defenses

Traditional DDoS defenses technologies rely on filtering or rate limiting traffic along different points of the network [12], such as with access control lists and firewalls. Each technique addresses a way to distinguish legitimate from malicious traffic and then provides a mechanism for increasing goodput [1] by filtering.

The foundation of filtering techniques have a major pitfall, however, because the victim has no control over routers on other autonomous systems (AS) and must rely on cooperation or ingress capacity to begin filtering. To counter this limitation, overlay networks, where traffic is routed to an intermediate server which filters before forwarding on to the destination [2, 10, 21], allow defenders to distribute filtering capacity in public clouds without relying on Internet infrastructure support. If the overlay servers are well distributed then a large filtering capacity can be established without modifying any Internet topology devices like firewalls or routers. Another capacity handing mechanism uses anycast [9] that forces traffic destined for a particular victim to be routed to several different servers or networks, each containing the same IP address, so that capacity is improved via redundancy.

B. Overlays and Moving Target Defenses

DOSE builds on two prior techniques, overlay networks and moving target defenses, to achieve low costs. Overlay networks [2, 10, 21] provide a layer of indirection to shield protected IP addresses from attacks, and moving target defense techniques control how and where these intermediate computers or relays process traffic to best curtail an attack. With modern cloud computing infrastructure, overlay networks can become elastic and nimble to cheaply dodge attacks, as shown in MOTAG [7, 8] and utilized by DOSE. This elasticity has to be carefully managed, however, to keep the cost of defense low. Simulated attacks in [8] use 1000 relays over 60 "shuffles" which each require additional virtual machines to implement. In Amazon's EC2 service this would translate to as much as 60,000 billable hours of usage, or over $700 in costs to repel a single attack. The existing moving target methods are stateless and do not account for clients leaving and joining, or the fact that non-aggressive attackers may take hours between subsequent attacks. DOSE focuses on different assignment techniques that minimize these costs by dynamically managing the expenses paid and the number of active relays for any particular defense situation.

III. DOSE Overview

In this section, we first describe the capability of adversaries that DOSE seeks to defend against. Then, we give a high level view of how DOSE appears to an end user and then what are the entities and mechanisms that are involved in providing such a view to the end user. The detailed design for each of the pieces mentioned here come in the next section.

A. Threat Model

DOSE is targeted at protecting small to medium-sized services, so the adversary is generally comprised of critics and competitors rather than highly skilled cyberwarefare attackers or those with extensive resources. DOSE is then designed to defend against attackers that are capable of disrupting communications to a few cloud virtual machines. DOSE then makes maneuvers to thwart the attacker by making the target unclear and difficult to determine if an attack was successful.

B. Workflow of DOSE

In DOSE, a client does not directly connect to the protected service; in fact the location of the protected service is kept hidden. Instead the client connects to a relay node on a public cloud infrastructure, which in turn reaches the protected service as explained in Section IV-A. We then utilize a commercial Content Delivery Network (CDN) based system to disseminate the *client-specific* relay information to each client. The dissemination of relay information to a client is done securely through an Assignment Service, with the channel between the Assignment Service and the client being made secure through a shared key that is established as a part of

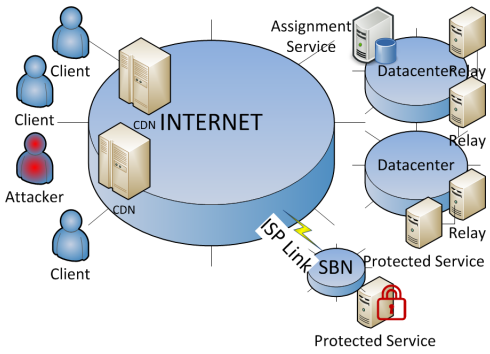

Fig. 1. Overview of Infrastructure deployed as part of DoSE: The attackers and clients exist on the Internet and are attempting to access a protected service that is either located in a data center or on a small business network (SBN).

DoSE. A malicious client cannot enumerate the list of relays, preparatory to attacking them, or connect to a relay that it has not been assigned to, as presented in Section IV-B.

Figure 1 details how the different elements of DoSE are laid out. The protected service is in a network which is potentially limited in bandwidth, while the relays are in the network of a public cloud provider (denoted as "datacenter" in the figure). The assignment service, responsible for assigning clients to relays, can be located either in the data center network or the small business network (SBN). A plausible deployment will have all the entities - the protected service, the relays, and the DoSE management service – in the cloud environment.

C. Client-Relay Assignment

The relays used in DoSE are such that whenever a relay is attacked, it can be taken offline and a new relay brought up in its place as in an elastic cloud. This allows the defender to have a seemingly endless supply of fresh relays, whose addresses are selectively released to clients. Further, DoSE keeps track of the assignment of clients to the relays so that if a relay gets overwhelmed due to traffic, a set of suspect clients can be identified. By progressively partitioning suspect clients among multiple relays, DoSE can ultimately identify persistent malicious clients as described in Section IV-C.

D. Defense Against Some Obvious Adversaries

Here we lay out some obvious ways in which an adversary can launch a DoS against the protected service or one of the entities that DoSE introduces.

An attacker identifies the IP address to which she is connected (a relay) and launches an attack at that IP, disabling it and any other clients also connected to that address. Consequently, DoSE creates new relay(s) and re-assigns clients among these new relays. The threat has several characteristics that inform DoSE's assignment strategy. Each time the client attacks, she is deemed by DoSE to have a higher risk due to prior behavior, i.e., her client being connected to the attacked node, and is therefore managed in such a way to minimize the impact of future attacks as explained in Section IV-D.

In the previous attack, the adversary can spawn a new client and try to access the protected service again, concealing her identity. This is handled by the design that a new client is considered inherently "risky". DoSE will then effectively place new clients in a kind of holding tank to establish trust. Therefore even under intense attacks, well established and well behaved clients can maintain communications with the service. In the event that an attacker is not aggressive, client connections can be consolidated to a few relays to save costs during non-attack periods and redistributed according to risk when an attack resumes.

E. Cost Minimization

Relays are only needed during attack periods which can be equated to the amount of risk present in the system. As more clients are involved in attacks, more relays become necessary to maintain communications. The number of active relays in DoSE is a function of the total connected client risk, controlled by cost parameters via a feedback controller.

IV. Detailed Design of DoSE

A. Relays

The relays are simple filter and forward firewalls operating in software on virtual machines as a part of cloud infrastructure. They are not on the direct network path from the client to the service but instead are substituted as the destination for the client and then interface with the protected service on the client's behalf. Each relay has a set of whitelist firewall rules and rate limiting for each assigned client, and the only entities in the white list are those clients that have been assigned to the relay. For each whitelisted client, there is a rate limit built in at the relay. These can be accomplished simply, for example by using iptables.

Attack traffic can come from two sources. The first source is an established client itself, in which case the attacker is known and the excess traffic is stopped by the rate limiting portion of the firewall. The second source is from a non-whitelisted address in which case it is simply dropped. In either case, however, the inbound bandwidth to the relay may become overloaded resulting in a disruption to all of the clients assigned to a particular relay. Detection of an overload is accomplished by an external watchdog, either the assignment node or some other source, by observing that round trip times have exceeded an acceptable limitation. When an overload occurs, the relay is abandoned and a new relay is established in the cloud.

How an attack against a relay node is detected is not central to DoSE, and any monitor which identifies when a relay is inaccessible to the clients is sufficient.

B. Client-Relay Assignment Infrastructure

The client-relay assignment infrastructure is responsible for communicating the assignment of a client to a relay to the client. It must provide an attack-resilient announcement of IP addresses to those clients while maintaining two properties. First, it must provide each assignment secretly, that is only the

intended client must be able to read an assignment. Second, it must be able to provide this assignment information without needing identifying information from the client.

To create attack-resilient relay announcements, the assignment is stored on a content distribution network (CDN). These networks, provided by large ISP's or data centers, store files in a distributed set of caching locations and provide only simple file transfer services (e.g. hash table lookups) which makes them difficult to target at an application layer for attacks. The content itself is replicated across several independent data centers with very fast network connections that make direct infrastructure attacks difficult. The CDN can operate as a push-only replica where the content must be pushed by the provider to the CDN, thus isolating the back end assignment infrastructure.

The goal that we seek to achieve in this protocol is to provide each client with a secure way to acquire its relay assignment using push-only files. At a high level, this is done by having each client solve a puzzle and the solution to the puzzle gives the name of a file to acquire from the CDN. Following the content in the file, the client is able to find the IP address of the relay that it is assigned to.

In the first stage, the CDN has a file with a large number of puzzles. A client fetches this set of puzzles from the CDN and decides randomly to solve one puzzle from the set. A puzzle comprises of guessing a number from a pre-specified range and solving a CAPTCHA. Thus, the Identity Establishment service (IES), which is part of DoSE's Management service, gives to the client a CAPTCHA image and an integer value "PoW difficulty". The interpretation of "PoW difficulty" is that the random number that the client has to guess will lie in [0, PoW difficulty]. The client computes the following hash H(puzzle identifier, text for the CAPTCHA, PoW guess) and sends it to the IES for verification. To perform partial verification at the client itself, the IES also provides the first half of the correct hash value to the client.

Next, the client guesses a value for the first component of the hash function ("PoW guess"), calculates the hash value, and checks if the first half of it matches with the first half of the correct hash value provided by the IES. If it does (say, outcome "A"), it uses the entire hash value as the name of the file to retrieve from the CDN. If it does not (say, outcome "!A"), the client has clearly not guessed the random number correctly. So it tries the next guess. The scheme forces the client to do some work before it gains service. Under outcome A, the client is able to acquire a file from the CDN, call this file "foo1". This file is itself encrypted using symmetric encryption and the key used is simply H(PoW guess (from the earlier hashing application) + 1). This seemingly contrived design is such that a brute force reading of files from the CDN can release the file to an adversary, but due to the encryption, it will not be able to decipher the contents of the file. The file "foo1" has a client ID and a short-term cryptographic key K_s.

The client ID maps to the name of a file also stored in the CDN, call this file "foo2". This file has the assignment of the client to a relay node, *i.e.*, reveals the IP address of the relay

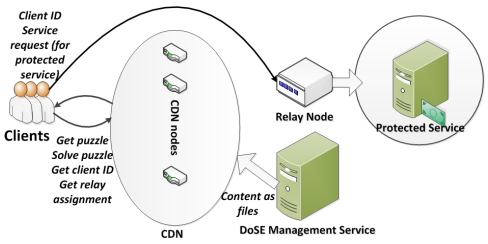

Fig. 2. The client interacts with static files stored on the CDN to retrieve an identity for obtaining future relay assignments in the event of an attack. The client selects a random puzzle from a set on the CDN and then does a proof-of-work to solve the puzzle and retrieve an initial Client ID and relay assignment to begin accessing the protected service.

to the client. During the first interaction with the relay node, made secure using K_s, the client provides its client ID to the relay node. If, due to collision of the puzzle space, the client ID has already been assigned to a previous client, the relay rejects the initial request from the client. The client then has to go back to the beginning of the process and solve another puzzle en route to getting a new client ID. If, on the other hand, the client ID is unique, then the relay node accepts the request of the client, *i.e.*, forwards it to the ultimate destination. Also, the relay provides a long-term cryptographic key K_l to the client and to the IES. When the client will be re-assigned to a new relay, say due to an attack, then it will repeat the part of the protocol starting from accessing "foo2" on the CDN. Since it has already acquired a unique client ID, it does not need to repeat the first part of the scheme. For all subsequent interactions with relay nodes, the client will use the key K_l.

The above scheme for informing a client of its relay assignment achieves the goal that it is using a vanilla CDN, which is "simply" a mechanism to distribute content (files "foo1" and "foo2" in this case). The IES of the DoSE simply pushes content to the CDN and the clients are never allowed to pull information directly from the IES. This provides a level of protection against DoS against the Management service (the IES is part of this) of DoSE.

The "PoW Difficulty" parameter allows the server to control the amount of work it will have the client do before it is allowed to connect to a relay; a higher value implies higher amount of work. The set of puzzles will be set of expire after a certain length of time, say after T time units. This is done so that a solution by one client cannot be reused forever by other clients. To reduce the likelihood of collision of the puzzle that a client solves, the number of puzzles in a set will be kept much larger than the number of new clients that are expected to connect to DoSE in time T. Finally, the client-CDN connection can be established over SSL to prevent man-in-the-middle attacks. An attacker may attempt to solve several puzzles to deny service to new clients by forcing collisions. This attack would require extensive computational resources on the attackers end, especially if the "PoW Difficulty" remains high, resulting in high expenditures for an attacker relative to the IES.

Several conditions must be maintained for this system to

be successful. With period T the content on the CDN is expired and removed. Clients arrive at a rate C_r, and puzzles are generated at a rate P_r. Since clients choose puzzles randomly, $C_r << P_r$ ensures a low chance of collision. P_r is a function of the computation and network capacity of the key management service. A modest capacity of 10 Mbit/s of network upload could provide up to 1000 puzzles per second, assuming a 1 KB image size. The PoW Difficulty parameter allows the server to control the computational advantage it has over malicious connecting clients at the expense of client connection establishment times.

C. Client-Relay Assignment Strategy Overview

At the core of DoSE is the many-to-one client-relay assignment strategy in which careful assignment reveals the attacker's identity.

In considering the client-relay assignment, we will differentiate between two categories of adversaries. The first category has the adversary connects, attacks, disconnects, and creates a new identity for itself (as explained in Section IV-B). She then launches an identical attack against the newly assigned relay. We call this type of adversary a *lone drone*. The second category has the adversary creating multiple clients concurrently, each of which gets assigned to its relay, then stays connected to the relay for an extended period, and then, in a coordinated manner, launching a DoS attack against all the relays. We call this type of adversary an *aggregate insider*.

In considering the assignment strategy, we introduce the notion of *risk* for each client. Risk is the likelihood that a client will launch a DoS attack in the future. The information items that DoSE currently uses are — length of time the client has used the service (in a well-behaved manner) and whether it is suspected to have been involved in an attack in the past. The latter factor is coarse-grained because when a relay is determined to be under attack, the suspicion falls on *all* the clients assigned to the relay.

New clients are slowly relieved of the *lone drone* risk by the function $P(\text{lone drone}|t_L) = e^{\frac{-t_L}{EXP(t)}}$ where t_L is the connection lifetime and $EXP(t)$ is the expected connection lifetime. The probability of a client being insider is more stateful and complex than that of the drone. If a set of clients is connected to a relay that gets attacked each client will have a uniform probability of being the attacker. The risk assigned to each client is then proportional number of clients connected to a relay. The statefulness of the risk assessment is maintained by recording both the clients involved in each attack and the increase in risk for each client involved in an attack. When an attacker is identified, the risk that was added for all clients implicated in attacks in which the known attacker was also a suspect, is removed. In summary, the factors that drive the risk estimation of clients is as follows:

- Each client begins with a suspicion of being a lone drone.
- As time progresses, the probability that a client is a lone drone exponentially decays.
- If a client is connected to a relay which comes under attack, its level of suspicion is incremented inversely proportionally to the number of other clients connected to the relay.
- When an attack is solved, the suspicion of clients associated with the series attacks is reduced.

Note that DoSE still allows for anonymous clients. It only requires there be a secure session between the client and the relay, but it does not require a specific identity of the client to be divulged to the service.

D. Formal Relay Assignment Strategy

The assignment strategy is designed to minimize percentage of disrupted clients which means evenly spreading risk among each relay.

With the notion of a high risk client being associated with its own relay comes the parameter *risk per relay* (RPR), which is defined as the maximum cumulative risk any relay will tolerate. The total number of relays to use is then calculated as the sum of client risks divided by the RPR. The implication is that if a set of clients connected to a particular relay is implicated in an attack and this causes the risk on each client to double, say, then the set of connected clients will be split among two relays. Another parameter of interest is called the *cumulative risk per attack* (CRPA). This is the total risk increment when a DoS incident occurs. This increment happens for all the clients connected to the attacked relay and the CRPA is divided by that number of clients to arrive at the per-client increment of risk. The CRPA is then parametrized and scaled to control, under a normal scenario, the growth rates of relays. The RPR and CRPA work together to control how the clients are spread during attacks. The higher the $\frac{CRPA}{RPR}$ ratio, the more quickly the attacker is isolated, but also higher is the number of required relays.

E. Optimizing Cost

Cost is the direct function of the number of relays utilized and can be controlled by the RPR parameter. A proportional-integral-derivative (PID) controller [6] can be used to adjust this parameter to control long term cost targets at the expense of mitigation effectiveness, specifically the integral term in the controller.

V. Experimental Results

Measuring the effectiveness of Denial of Service mitigation is a challenging task, especially when comparing techniques that operate on fundamentally different principles. Prior work on metrics for DoS attacks and defense [16] suggests using the percentage of failed transactions to establish effectiveness which we will use and denote as *PFT* for measuring the success of DoSE.

The effectiveness of different assignment strategies can be assessed without the need for real hardware implementations. The actual implementation parameters that influence the performance of DoSE can be easily measured, e.g., the time that it takes to bring up a relay, the time that it takes to create an iptable filtering rule (at a relay). These parameter values

are then fed into our simulator. For that reason, the performance parameters are modeled in an agent-based simulation to evaluate the DoSE approach. While other testbeds exist [14, 15], their goal is to assess filtering methods, which must operate on real-time traffic. In the case of DoSE, once a relay is determined to be unusable due to an attack, it is null routed and henceforth, the actual size of the attack does not have any implication on the network.

A. Management Service Overhead Analysis

The management service has three components that need analysis. The first is the CDN system itself which has been measured sufficiently by prior studies [18, 22]. The second component is the puzzle generation system. Proof-of-work systems are not new, and Portcullis [19] is one example of a network defense implementation which has done performance testing on hash functions. Hashing performance was measured on an EC2 "m1.small" instance to be 1 million hashes per 1.71 seconds which is plenty of computing capacity for large scale puzzle generation. The next component is the CAPTCHA generation. With 1 ECU, 10,000 GTT's , 4.2 KB each, were created in 5.11 seconds using libcaptcha. The final component is the assignment and client tracking system. Each session is associated with a key, assignment, and a risk value which is only a small amount of data to store. The assignment algorithm itself is simple — a client is greedily placed on the lowest-risk relay. The conclusion then is that simple EC2 instances can generate a sufficient number of puzzles to supply 10-100 new clients per second with identities.

B. Agent-Based Simulator

To model the DoS scenario, we constructed an agent-based simulator in MATLAB. The simulator consists of a framework for event scheduling and agent-to-agent communications. The agents constructed are the clients, attackers, relay nodes, the assignment service, and the end application. The client is modeled as having a request rate distribution to mimic UDP streaming applications. If the client requests exceed the capacity, then the requests will be delayed or discarded. The relay is modeled as a simple packet forwarder that wraps and unwraps packets to the service from the client as well as the reply. The attacker has the ability to disable its assigned relay node at any time, causing all traffic to be dropped by that relay. We believe this agent-based simulator is suitable for comparative evaluation of a suite of DoS protection techniques, namely, those that rely on some form of proxy or relay nodes. We therefore will be releasing this simulator for broader use.

In the rest of the section, we will refer to the terms *legitimate clients* and *malicious clients*. Where it can be used without ambiguity, when we will use the term *client*, it will refer to a legitimate client. We will also sometime use the term *attacker* which will be synonymous with malicious client.

C. Experiment 1: Single Adversary

The first experiment involves 1,000 legitimate clients and a single attacker to mimic the impact of a DDoS-for-Hire service. The simulation is initialized with all of the clients and the attacker connecting at the same time, thus having an initially uniform distribution of risk. The attacker continuously attacks, and the experiment ends once the malicious node gets identified and isolated.

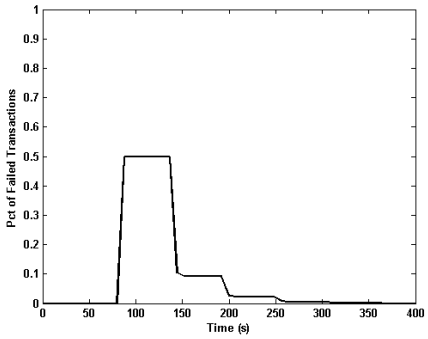

Fig. 3. Percentage of Failed Transactions with a single attacker: The attack begins at 80 seconds and is progressively mitigated with each relay-creation cycle, till the attacker is identified and neutralized at 3.9 minutes. The relay power-on time is 40 seconds, the approximate width of the steps.

Figure 3 shows the resulting average PFT for the set of connected clients across time. In this case, the minimum number of relays is two. After the initial attack, one of the two relays is disabled, resulting in a PFT of 50%. Additional relays are brought online to distribute the now high risk set of disrupted clients. This continues until 3.9 minutes into the simulation when the attacker is identified. The time needed to neutralize the malicious client depends on the cumulative risk per attack (CRPA) parameter and the time to bring a relay online (40 seconds). If the CRPA increases, the time to identify and neutralize the malicious client will decrease. However, the downside to having very large CRPA increases is that this will trigger a very high rate of relay creation. If the time to bring an instance online increases, then the time to find the malicious client will also increase. Over the time span of the entire experiment, the average PFT is 0.118, meaning that an average of 88.2% of the requests are satisfied during the mitigation period.

Another important metric is the number of relays used in the defense. Figure 4 details how the number of relays grows with time, growing from an initial 2 relays to 16 relays. The growth rate is a function of the risk assigned per attack (CRPA) along with the risk per relay (RPR) parameter. After each attack cycle, additional relays are created to reduce the impact of attacks as well as to zoom into the suspect list. The number of relays is also dependent on the number of clients connected. Figure 5 shows the number of relays in use during the same attack but with only 100 clients. In this case, the attacker is found more quickly and the overall usage is lower, growing to a maximum of 11 relays. This drives home the point that costs in DoSE are client-dependent. Note however, that there is no strict proportionality between the number of clients and the number of relays needed to mitigate the attack. The formulae used in the algorithms for assignment of clients to relays are

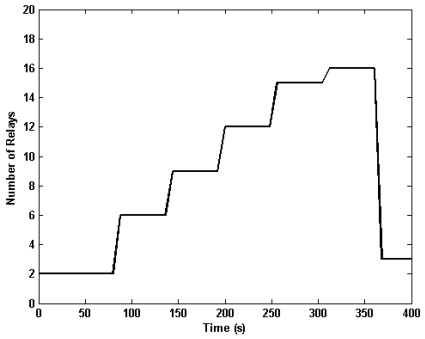

Fig. 4. Number of Relays Used: Defending against a single attacker with 1000 clients shows increased relay counts until the attacker is found.

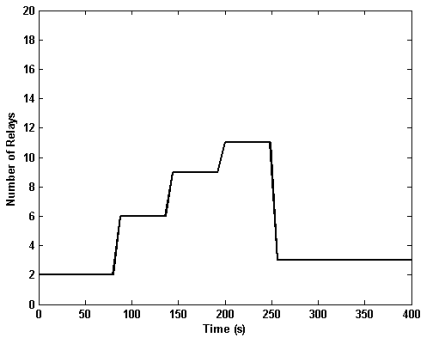

Fig. 5. Number of Relays Used: Defending against a single attacker with a smaller number of clients shows the relay count to be smaller than with 1,000 clients.

non-linear and this explains the observation.

D. Experiment 2: Streaming Attack

In the second experiment, the stress due to the attack is higher with multiple malicious clients launching coordinated attacks. We refer to this as the *Streaming Attack*. New legitimate clients connect to the service uniformly every 1.6 seconds and stay connected for 400 seconds, so that the steady-state client base is 250 clients. The attackers arrive in sets of 10 every 160 seconds and launch a simultaneous attack. Eventually, the malicious clients are identified and neutralized.

Figure 6 shows how the PFT is impacted by the streaming attack. The initial attack is more difficult to mitigate because number of active relays is low prior to any attack and the attackers begin at a time when there are not a large number of legitimate clients already connected and those that are in the system have not been connected for a long time. As the client base matures, subsequent attacks have a much lower impact on the PFT because the new clients are high risk compared to the existing legitimate ones. We see that even the spikes, which correspond to the arrival of the new batch of malicious nodes, only go up to 20% failed transactions.

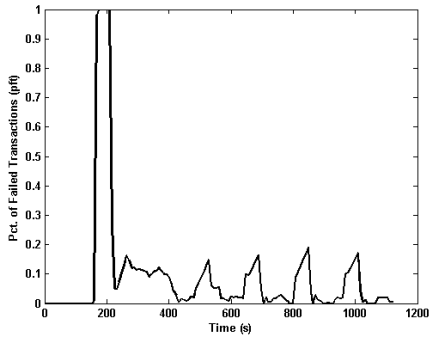

Fig. 6. Percentage of Failed Transactions (PFT) with the streaming attack: The attack begins at 160 seconds and is repeated with 10 malicious clients arriving every 160 seconds. A legitimate client arrives every 1.6 seconds and stays connected for 400 seconds. As time progresses, the risk profile of the legitimate clients decreases leading to a better isolation of the malicious nodes. Consequently, the PFT decreases compared to the initial burst of attack.

E. Experiment 3: Attackers Present at Startup

In this experiment, we evaluate the case when the malicious clients are present when the system is first brought online. The performance of DOSE is evaluated as the number of malicious clients is varied from 1 to 500, with a constant legitimate client base of 1,000. Two metrics are used - PFT and the number of relays. Both metrics are averaged over 800 seconds, by which time the last malicious client has been identified and isolated.

Figure 7 shows how many relays are used to defeat attacks of varying sizes of number of malicious clients. The overall impact is that with a large number of attackers, the number of relays required to distinguish between attackers and legitimate clients becomes large as well. With 1 malicious client, the number of relays required is 6; with 15, it is 30; and, with 500 malicious clients, the number of relays required is 154. There are two factors to note here. First, the ratio of adversarial to legitimate clients of 1:2 is rather high, and second, the growth in the number of relays is slower than the increase in attackers. Also, an insider and a legitimate client begin with indistinguishable features and history. A more likely case is that some clients will have a higher trust due to a long association with the service.

Figure 8 details the impact on the average PFT during the attacks. Expectedly, as the number of attackers increases, the PFT increases. As each attack occurs, the legitimate clients are prevented from accessing the service until a relay is assigned only legitimate clients. With this level of discrimination, the malicious client(s) can be isolated. However the convergence time to reach this point increases with the number of attackers, thus driving up the average PFT.

F. Sensitivity to CRPA

The CRPA parameter, as described in Section IV-D, controls the amount of relay growth per attacked relay. If the CRPA is 2 and the RPR is 1, then a single attacked relay will be replaced by 2 new relays. The CRPA factor trades-off between growth rate of the number of relays, cost, and speed of mitigation.

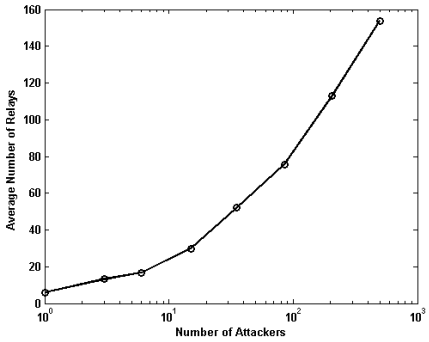

Fig. 7. Number of Relays vs Number of Malicious Nodes: The number of legitimate clients is kept fixed at 1,000 and the number of malicious clients is increased. All the clients are present at startup. The increase in the number of malicious clients is met with automatic and progressive increases in the number of relays, to isolate the malicious nodes.

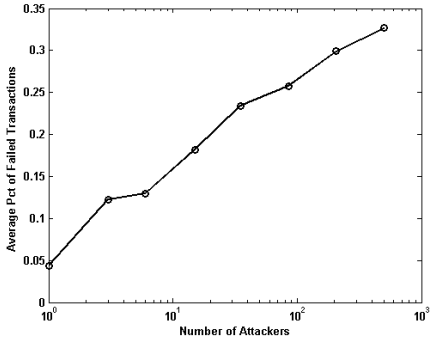

Fig. 8. PFT vs Number of Malicious Clients: The number of legitimate clients is kept fixed at 1,000 and the number of malicious nodes is increased. All the clients are present at startup. The increase in the number of attackers causes the average impact on PFT over the attack simulation window to grow as a larger-sized attack impacts more client-connected relays.

Figure 9 shows the relay growth from Experiment 1 (single attacker, Section V-C), with varying CRPA factors.

In this experiment, the attacker is flooding the relays continuously and therefore, the high CRPA value gives the best result — the time to identify the attacker is the lowest (compared to lower CRPA values) and the total cost is the same. The downside of a high CRPA value will be exposed if the attacker is more subtle, and after launching an attack (and causing a large number of relays to be created), it lies low for a while. The spurt in the number of relays does not help DoSE if condensed prior to a subsequent attack. In summary, we can say that the optimal CRPA value should be a function of the expected time interval between attacks.

G. Quantitative Comparison with Epiphany and Speak-up

We do a quantitative comparison of Epiphany [9] and Epiphany coupled with Speak-up [23] with DoSE for the setup of experiment 3. DoSE is executed and the number of relays used in DoSE is given as the number of proxies in Epiphany to compare the effectiveness of attack mitigation. To do a comparison, a distribution was created of clients

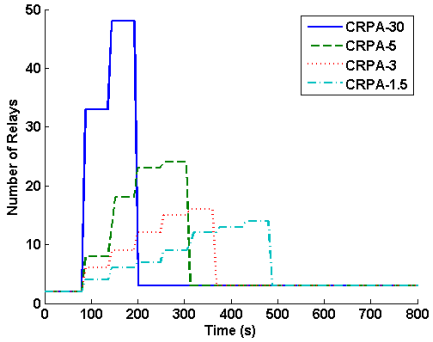

Fig. 9. Number of relays over time for varying CRPA factors: The number of relays utilized peaks to a much higher number with high CRPA values, however the attacker is found much more quickly. The factor controls the growth rate in the number of relays created in response to an attack.

and attackers on the Internet, such that 10% of the Internet is "unclean networks", where 80% of the IP addresses are attackers. The remaining 90% of the Internet is split between completely clean and partially clean networks, with the clean network accounting for 10% of the addresses and defined as having no malicious IP address. The partially clean network has 20% of its IP addresses as attackers. The design of Epiphany suggests that if a legitimate client is in the unclean network, then she will not be able to have any successful transactions. If the client is assigned to a clean network, then there will be no impacts of the attack. Therefore the percentage of failed transaction varies between 10% and 90%. The remaining clients may or may not be connected to proxies which contain attackers. If there is an attacker, then the ratio of successful transactions is the percentage of good clients connected. Thus if there are 9 clients connected to a proxy and 1 attacker, 90% of the transactions will be successful. This models the behavior of Speak-Up [23], as used by Epiphany. In the alternative mode, without Speak-Up, any client assigned to a relay with an attacker will not be able to access the protected service.

Figure 10 shows the results of the comparison. At a low number of attackers, DoSE is able to maintain the advantage. At higher numbers of attackers, Epiphany + Speak-Up is able maintain the best advantage while DoSE outperforms the pure Epiphany solution. However, as pointed out earlier, Epiphany by itself, and even more so with Speak-up, requires widespread upgrades to the basic network infrastructure.

H. Comparison with MOTAG

MOTAG [7] provides an alternative approach to client-relay assignments from DoSE. It uses a greedy algorithm to guide assignments in an attempt to save as many clients as quickly as possible. To make a comparison between DoSE and MOTAG, DoSE was run and the average number of relays used during the scenario was used as the input to MOTAG. MOTAG also relies on a set of shuffling proxies and a set of serving proxies, and legitimate clients are moved from the shuffling to the serving proxies as the attack progresses. We assume

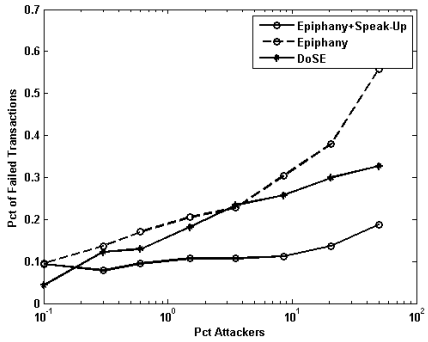

Fig. 10. Comparison of DoSE to Epiphany and Epiphany+Speakup: At low ratio of number of attackers to legitimate nodes, DoSE outperforms other solutions, while at mid to high ratios, Epiphany+Speak-up outperforms DoSE.

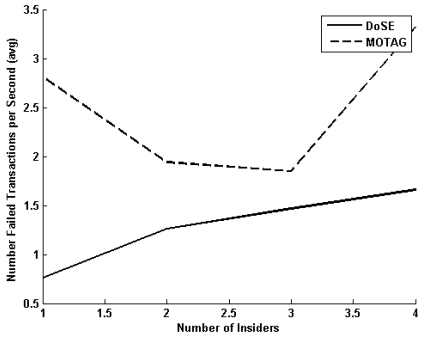

Fig. 12. Comparison of DoSE to MOTAG. DoSE is able to keep the number of failed transactions lower during the simulation because it better isolates and identifies the attacker more quickly, resulting in less clients co-located on relays with insiders.

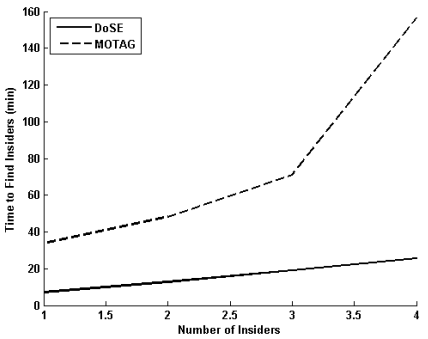

Fig. 11. Comparison of DoSE to MOTAG. The time it takes to find all of the insiders is significantly lower in DoSE, especially as the number of insiders increases. This is due to the smart relay management which divides the clients during successive attacks in DoSE while MOTAG relies on a stateless classification which lets intelligent adversaries fool the system.

a distribution of half and half between the shuffling and the serving proxies. Additionally, MOTAG provides an estimation technique for the number of insiders; here, the truth value is simply provided to MOTAG. The insider feeds the IP address of the relay to the botnet. In this evaluation, we use a rate of one IP address being revealed per minute so multiple insiders are independent sources of addresses for attack.

Figure 11 shows that MOTAG takes much longer to converge on the attackers than does DoSE. This is because MOTAG relies on stateless transitions between attacks while DoSE builds a long-standing history, through the risk value, for each client. Therefore an insider who stops attacking is not forgotten, while in MOTAG, the insiders may continue to impact legitimate clients if moved to a serving proxy. A higher time to find the attacker results in increased costs since a larger number of relays are ultimately used in defense, thus with a lower end time, DoSE costs 59% less than MOTAG. Figure 12 shows the number of failed transactions per second, averaged across the total simulation time. Since DoSE finds the attacker more quickly than does MOTAG. Less transactions fail for legitimate clients when using DoSE to defend against this attack scenario.

VI. Costs and Discussion

The cost analysis is application dependent with two components: time-of-use resources (relay instance runtime) and bandwidth-based network resources (cost to forward). Relay nodes on EC2 [20], for example, cost $8.64 a month for the smallest instance or $2.23 with spot pricing. Network resources are a function of legitimate client traffic that must be passed from the cloud to the protected service. Costs for forwarding traffic out-of-network are priced per GB and range from $0.05 to $0.12 per GB, but can be mostly avoided by co-locating the relays and protected service in the same IaaS cloud.

A comparison can be made between the cost of DoSE and existing commercial solutions. For a representative sample, Staminus Communications [4] will be used. The defensive cost is related to the capability of the adversary, since a filtering technique is used, and the cost comes to about $35 per Gbit, $175 per MPPS of attack protection. The defensive cost for an average attack of 9.7 Gbit/s, 19.8 MPPS peak [5] would be $3,465 per month.

The cost of DoSE can be compared to Epiphany [9] and MOTAG [7, 8]. For the former, a set of static proxies are needed as entry points into the Epiphany routing structure. The cost of operating a proxy for Epiphany will be 50% higher than EC2 because it is transitioning from a datacenter style location to a more end-user style location (drawing from Amazon's differentiated tiers). To estimate the router upgrade cost, if each router takes 1 man-hour to reconfigure, and there are 1 million routers in the system, then upgrading 0.01% (a parameter chosen in [9]) of the routers to support Epiphany would require 1,000 man-hours or $100,000 at $100/hour for a network engineer. If there are 10 routers between the source and destination and each router has 10 output ports enabled, the amount of bandwidth used will be 100 times the original transmission due to reverse-multicast.

For MOTAG, cost-equivalent performance comparisons can be seen in Section V-H. As tested in [8], MOTAG relied on 1000 replicas through 60 shuffles which amounts to 60,000

billable hours (at 1 hour minimums) on EC2, or over $700 to stop a single attack. With that many replicas in play, more efficient static approaches could keep clients connected.

We summarize these findings in the following table using the shorthand "R" for relay, "m" for month, and R-dependent means it depends on the number of relays.

Defense	Overhead	BW	Atk. Size
DoSE	$9/R/m	2x	Independent
Epiphany	$22/R/m+$1670/m	100x	R-Dependent
Commercial	$3,465/m	Included	100 Gbps / 20 MPPS

Another cost of using any design as in DOSE of separate entities — relay and protected service — is the latency penalty for taking alternative routes in the network. This has been measured in [7] and a round trip time penalty of 70 ms would be typical for the continental US.

The cost of DOSE for the experiments in Section V can be calculated for EC2. In the first attack example V-C, a maximum of 16 relays are used, and there are 5 rounds of relay expansion captured by the number of steps seen in the graph. Since EC2 requires a minimum billing of 1 hour for each instance started, the 5 instances that were disabled by the attacker along with the 16 relays consumed are billed for 1 hour, totaling 21 hours of billing on the smallest EC2 instance. The cost is then $0.42 for on-demand instances or $0.07 for spot instances to stop this attack. Additional costs will depend on how much Internet bound traffic passes through the relay.

VII. Conclusion

We presented DOSE as a method for mitigating network-layer Denial of Service attacks. DOSE uses cloud-based relay nodes to hide protected services from direct attack while acting as sacrificial targets. The key idea behind DOSE lies in its ability to assign relays to specific clients and by re-assigning suspect clients to a progressively smaller set of relays, to identify and isolate the malicious clients. DOSE also smartly manages the set of relays to optimize cost while allowing maximal attack mitigation. The technique is capable of quickly mitigating attacks while continually improving on the legitimate client's ability to complete transactions. We believe ours is the first technique that can achieve DoS protection for a price point that would be acceptable to medium-to-small-sized businesses, of less than $100 per month. In future work, we are looking at ways to intelligently place the management service with respect to multiple protected services and ways of customizing the CDN, without significantly increasing costs. On a larger scale, we will investigate low-cost ways of handling other kinds of attacks, through the use of virtual machines and CDNs.

References

[1] S. Agarwal, T. Dawson, and C. Tryfonas. Ddos mitigation via regional cleaning centers. Technical report, Sprint ATL Research Report RR04-ATL-013177, 2003.

[2] D. G. Andersen et al. Mayday: Distributed filtering for internet services. In *USENIX Symposium on Internet Technologies and Systems*, pages 20–30, 2003.

[3] I. CloudFlare. Cloudflare - compare plans, Oct. 2013.

[4] S. Communication. Ddos protection - secureport global, Sept. 2013.

[5] M. E. Donner. Increasing size of individual ddos attack dfine third quarter— prolexic, Sept. 2013.

[6] G. F. Franklin, J. D. Powell, and A. Emami-Naeini. Feedback control of dynamics systems. *Addison-Wesley, Reading, MA*, 1994.

[7] Q. Jia, K. Sun, and A. Stavrou. Motag: Moving target defense against internet denial of service attacks. In *ICCCN, 2013*, pages 1–9, 2013.

[8] Q. Jia, H. Wang, D. Fleck, F. Li, A. Stavrou, and W. Powell. Catch me if you can: A cloud-enabled ddos defense. In *Dependable Systems and Networks (DSN), 2014 44th Annual IEEE/IFIP International Conference on*, pages 264–275. IEEE, 2014.

[9] V. Kambhampati, C. Papadopolous, and D. Massey. Epiphany: A location hiding architecture for protecting critical services from ddos attacks. In *Dependable Systems and Networks (DSN), 2012 42nd Annual IEEE/IFIP International Conference on*, pages 1–12, 2012.

[10] A. D. Keromytis, V. Misra, and D. Rubenstein. Sos: Secure overlay services. *ACM SIGCOMM Computer Communication Review*, 32(4):61–72, 2002.

[11] LiquidWeb. Ddos prevention pricing structure, Apr. 2014.

[12] G. Loukas and G. Öke. Protection against denial of service attacks: a survey. *The Computer Journal*, 53(7):1020–1037, 2010.

[13] R. Masse. Denial of service as a service - asymmetrical warfare at its finest, Dec. 2013.

[14] J. Mirkovic, S. Fahmy, P. Reiher, and R. K. Thomas. How to test dos defenses. In *Conference For Homeland Security, 2009. CATCH'09. Cybersecurity Applications & Technology*, pages 103–117. IEEE, 2009.

[15] J. Mirkovic, A. Hussain, S. Fahmy, P. Reiher, and R. Thomas. Accurately measuring denial of service in simulation and testbed experiments. *Dependable and Secure Computing, IEEE Transactions on*, 6(2):81–95, 2009.

[16] J. Mirkovic, A. Hussain, B. Wilson, S. Fahmy, P. Reiher, R. Thomas, W.-M. Yao, and S. Schwab. Towards user-centric metrics for denial-of-service measurement. In *Proceedings of the 2007 workshop on Experimental computer science*, page 8. ACM, 2007.

[17] A. Nixon and C. Camejo. Ddos protection bypass techniques. *Black Hat USA*, 2013.

[18] E. Nygren, R. K. Sitaraman, and J. Sun. The akamai network: a platform for high-performance internet applications. *ACM SIGOPS Operating Systems Review*, 44(3):2–19, 2010.

[19] B. Parno, D. Wendlandt, E. Shi, A. Perrig, B. Maggs, and Y.-C. Hu. Portcullis: protecting connection setup from denial-of-capability attacks. SIGCOMM '07, pages 289–300, New York, NY, USA, 2007. ACM.

[20] A. W. Services. Amazon ec2 pricing, pay as you go for cloud computing services, Sept. 2013.

[21] A. Stavrou and A. D. Keromytis. Countering dos attacks with stateless multipath overlays. In *Proceedings of the 12th ACM conference on Computer and communications security*, pages 249–259. ACM, 2005.

[22] S. Triukose, Z. Wen, and M. Rabinovich. Measuring a commercial content delivery network. In *Proceedings of the 20th international conference on World wide web*, pages 467–476. ACM, 2011.

[23] M. Walfish, M. Vutukuru, H. Balakrishnan, D. Karger, and S. Shenker. Ddos defense by offense. SIGCOMM '06, pages 303–314, New York, NY, USA, 2006. ACM.