AS-Based Accountability as a Cost-effective DDoS Defense

Daniel R. Simon Sharad Agarwal David A. Maltz
Microsoft Research

***Abstract*— Defenses against botnet-based distributed denial-of-service (DDoS) attacks must demonstrate that in addition to being technically feasible, they are also economically viable, particularly when compared with the two most widely deployed defenses—simple massive overprovisioning of resources to absorb and handle DDoS traffic, and "scrubbing" of incoming traffic by the victim's ISP. We argue that the key to cost-effective handling of DDoS attacks on a network such as the Internet is *accountability*, meaning that the sources of all traffic can be accurately and reliably identified, and receivers can effectively block traffic to them from any source.**

We propose a simple approach to directly providing accountability within a group of ASes. It combines strict ingress filtering on all edge traffic with an AS-based infrastructure that allows hosts to request that traffic to them from specific other hosts be blocked at the source. We also propose using the previously proposed "evil bit" in IP headers to allow a group of ASes that implement accountability to collectively reduce the impact of DDoS attacks originating outside their portion of the Internet. Finally, we present evidence for the economic competitiveness of our approach, compared with the current default approaches of massive overprovisioning and ISP scrubbing.

I. Introduction

With the advent of the underground botnet market, denial-of-service (DoS) attacks have graduated from amateur recreation to for-profit crime. Major content providers and server operators currently spend millions of US dollars each on resources to cope with attacks that cost less than $1000 each to launch [21]. So long as this disparity remains, there will be an incentive to use DoS attacks to extort "protection money" or harm business competitors.

DoS attacks work by diverting the target's resources away from dealing with legitimate traffic. In particular, a *network-layer DoS attack* attempts to exhaust the target's network capacity, flooding it with so many attack packets that legitimate traffic might never even reach the target. To be effective, then, a defense against network-layer DoS must involve the network itself.

The simplest defense against this attack, which is also the current default solution for actual or potential DoS targets, is extreme overprovisioning of bandwidth, enabling the target to absorb DoS traffic while still serving legitimate clients. A related solution is to have the target's ISP "scrub" the traffic before it reaches the customer, filtering out enough to avoid overwhelming the customer's bandwidth. These approaches can both be quite costly, given that large botnets can generate stunningly huge volumes of traffic. But they also have a number of attractive features:

- **Incremental deployability:** individual customers are able to implement them and see benefits, without every user or AS on the Internet having to participate.
- **Independence of infrastructure hardware:** they require no changes to deployed router hardware throughout the Internet.
- **Backward compatibility:** traffic among non-deployers does not get the benefit of deployers' resiliency against DoS attacks, but is otherwise unaffected.
- **Economic viability:** they can be provided at a cost that DoS targets are willing to pay.

This last property is crucial. A more sophisticated network-based defense against DoS attacks must involve finding a way to distinguish legitimate traffic and concentrate resources on it. Incorporating such a defense into the network will inevitably impose a cost. If that cost is greater than the cost of the aforementioned current practices, then it is highly unlikely to displace them. Conversely, if it is considerably cheaper than these practices, and can also match their other desirable properties, then it should be able to supplant the current solutions-of-choice in the marketplace.

In this paper, we describe a set of techniques that together satisfy these criteria. Our approach is to create an architecture for *accountability*, inside which hosts receiving unwanted packets can cause these packets to be filtered at the source. The elements of this architecture include:

Deployment of general and on-demand ingress filtering. We present a practical approach to ingress filtering deployment, thus ensuring that DoS targets can distinguish network-level DoS traffic from legitimate traffic based on its (accurate) source address. We then provide a mechanism for allowing DoS targets to request that traffic sent to them from a particular source be filtered at the source.

Distinguishing traffic by the accountability of the source AS. Crocker [11] proposed using a bit in the packet header to indicate whether or not the packet originates from "well-maintained machines." We extend this idea by using a bit in the packet header to distinguish the traffic of ASes that deploy our accountability architecture from the traffic of those ASes that do not. Thus our system helps create a working ecosystem out of pairwise trust among providers, within which untrusworthy traffic from outside the ecosystem can be recognized and filtered as necessary. (In deference to an "April's fools" joke RFC by Bellovin [5], we refer to the bit in the packet header as the "evil-bit."[1])

[1]Steve Bellovin proposed the addition of an "evil bit" to the IP packet header: all applications and hosts with malicious intent would be required to set this bit to 1, and all routers would preferentially drop evil traffic.

In addition to presenting our proposal, we will offer a detailed analysis of its economic viability, arguing that its deployment would be significantly less costly than long-term continued reliance on the current default alternatives. We believe this is the first such analysis of a proposed solution to the DoS problem.

Section II discusses related work and contrasts it with our own. Section III explains and justifies our definition of the DoS problem and Internet accountability. Section IV describes how accountability can be achieved among a club (not necessarily a clique) of ASes with pairwise trust. Sections V to VI explain how accountability can be used to block DoS traffic, and how to deal with ASes who misbehave within the club of accountable ones. Finally, Section VII examines the economics of accountability, compared with the current state of affairs.

II. Related Work

Analyses of botnets [10], [32] consistently mention DoS as one of the major uses to which botnets are put. Measures against botnet-based DoS can be either preventive or reactive. Preventive measures [15], [6], [28] aim at finding and disabling DoS perpetrators before they attack. Reactive measures come into play when prevention has failed and a DoS attack is under way, necessitating a defense of some kind.

To date, DoS defense has followed one of two approaches. The first approach involves applying enough resources to absorb the attack – for example, placing "scrubber boxes" between the attackers' traffic and the targeted servers and provisioning these scrubbers with sufficient resources and algorithms to separate the attack traffic from the desired traffic. This approach and its costs are explored in more detail in Section VII.

The second approach is to change the Internet's architecture so that DoS traffic is easily identified and dealt with. Since our approach is also architectural, we briefly review the three major types of proposed anti-DoS architecture.

Source Identification: Source identification mechanisms, such as ingress filtering [14], PKI-based IPSec [20] and IP Traceback [29], provide a method by which DoS targets can distinguish attack traffic from other traffic based on its source. However, source identification mechanisms, by themselves, provide little benefit until they are deployed universally — since attackers can simply avoid their range of effectiveness — and hence do not justify the cost of partial deployments. Moreover, IP addresses alone are in some cases too transient to be useful source identifiers. Finally, the ability to identify DoS traffic once it arrives at its target doesn't by itself protect the target entirely — for example, its last-hop bandwidth may be overwhelmed regardless.

Pushback: "IP Pushback" [18] involves in-band signaling among adjacent routers near an attacked end host that specific links are being overwhelmed, presumably by DoS traffic. Because it does not attempt to identify DoS traffic except in the aggregate, Pushback can only achieve partial filtering of unwanted traffic and risks dropping much wanted traffic as collateral damage. Moreover, Pushback requests require an authentication infrastructure, lest they themselves become a tool for DoS attacks, and they may also require router hardware changes. Finally, reflection attacks [27] (see Section IV-E), in which the attacker rapidly shifts the apparent source of the DoS traffic by bouncing it off any of the innocent hosts in the Internet, can render Pushback ineffective.

Packet-carried Capabilities: In capability-based systems [34], [2], [35], routers generate per-path labels, which are then combined to form a "capability" for that path. Recipients relay the capability to each welcome sender on request, and routers give preferential treatment to packets carrying a valid capability. There are three drawbacks to this architecture. First, the capability request process is itself not protected by the capability mechanism, and is therefore vulnerable to flooding attacks. (In [35], the statistical identification methods of [33] are used to reduce but not eliminate this problem.) Second, routers throughout the network must have their hardware modified to maintain expensive per-capability state to prevent capabilities from being overused. Finally, the architecture inherently limits identification to network location alone, rather than a more persistent attribute. Hence in some cases (e.g., mobile hosts like laptops), the identification will be transient and easy to circumvent.

A filtering architecture not entirely dissimilar to ours has recently been proposed [3]. However, it posits filters at arbitrary points in the network, and replaces ingress filtering with a randomized path-marking technique (which leaves open the possibility of reflection attacks). Filter requests in that architecture are validated using a TCP-style handshake, whose initiation can itself be used as a DoS-attack mechanism. Another, similar, proposed general filtering architecture [4] piggybacks on the routing infrastructure, and addresses neither authenticity of filtering requests, nor source-address spoofing of attack traffic, nor reflection attacks.

Crocker [11] proposed that ISPs decide whether or not their customers run well-managed networks, marking packets from well-managed networks with a DiffServ code point entitling them to preferential treatment over poorly-run networks during times of overload. We use a variant of Crocker's idea to distinguish packets by originating AS. We then add the ability for traffic within accountable ASes to be blocked on a per-customer basis. Together these enable the AS' decision regarding whether a particular customer is "well-managed" to be effectively automated, as described in Section V.

III. Accountability and (D)DoS

A. The Economics of (D)DoS

The goal of a DoS attack is to impair the target's functioning by forcing it to spend resources (bandwidth,

processing time, or storage space) handling the attacker's traffic. If the resources needed to handle the attacker's traffic are a large enough fraction of the target's total available resources that the remainder is insufficient to handle legitimate traffic, then the attack succeeds.

Once a DoS attack is in progress, the target's only defense is to attempt to distinguish between DoS traffic and legitimate traffic — typically, based on its source — and reduce the cost of dealing with DoS traffic. The attacker's resources are measured in the number of hosts the attacker has at its disposal, since that number gives a rough approximation of the resources that the attacker can muster, assuming that the attacker controls a large botnet consisting of compromised end hosts. [2]

At the application layer, numerous end host-based strategies may be available to the target for distinguishing DoS sources from others. For example, the target may use a handshake (such as the TCP handshake) to identify hosts by network location, and detect and blacklist hosts that use up abnormally large amounts of the target's resources. Or the target may attempt to distinguish hosts controlled interactively by humans from those generating traffic automatically [19], under the assumption that all legitimate traffic to the application will be in the former category. The attacker may in turn attempt to counter each of these defenses, of course—say, by taking advantage of DHCP to change network addresses frequently, or by attempting to simulate interactive human controllers in software.

However, vulnerability to application-layer DoS is ultimately completely application-dependent. For example, if the application is so vulnerable to DoS attacks that a single client can launch a successful DoS attack against it by issuing a legitimate sequence of application-layer requests, then the only feasible defense is to fix the application. Conversely, a sufficiently well-designed application may be completely invulnerable to application-layer DoS, in the sense of being capable of handling all application traffic that can possibly fit through its network connection.

On the other hand, even before traffic reaches the application layer, it must first pass through the network layer, where its processing consumes resources — especially bandwidth — *irrespective* of the destination application. If the attacker can cause enough resource consumption at the network layer, then application-level anti-DoS measures will be useless. And in practice, it is common for botnets to flood a target with many times more traffic than even peak loads of legitimate traffic ever reach.

For example, a large website or host may plan for a peak ("flash crowd") load of 100,000 "hits" per second of presumably legitimate traffic, with each hit requiring roughly 5KB of transmission. This level of traffic can be handled by a 4Gbps connection. However, a botnet of 4,000 nodes, each using a 1Mbps broadband connection, can completely saturate a 4Gbps line. Such botnets are by no means uncommon, and are reputedly available for rental at a rate of 10 cents per bot, or $400 [21].

Hence, unless the target can distinguish and block DoS traffic *in the network layer*, DoS defense will require, at the very least, a large outlay for surplus network capacity and per-packet processing or upstream "scrubbing". (We discuss these costs in detail in Section VII.) Minimizing these network-layer costs is thus a key to effective-yet-economical DoS defense.

B. Defining Accountability

In light of the need, noted above, to *distinguish* and *block* DoS traffic in the network layer, we define accountability in a network as having two components:

1) *Identification*: the originators of traffic can be identified by some *persistent attribute*—that is, one that is relatively difficult to create, re-create or change. The originator's IP address itself might be difficult to create or change, for example—or it might be easy to create or change, but reliably associated, at any given moment, with another more permanent attribute (e.g., legal name or credit card number).
2) *Defensibility*: destinations are able to prevent traffic from a source with a particular address or persistent attribute from affecting their use of the network. Note that defensibility requires, but does not necessarily follow from, identification: a network that doesn't provide identification cannot provide defensibility—since the latter requires that traffic be distinguishable by originator—but a network that provides identification can still fail to provide defensibility (and hence, full accountability).

The reason why identification must be by a persistent attribute should be clear from the definition of defensibility: the latter is useless if the originators of unwanted traffic are able to escape identification merely by changing the attribute that identifies them. For example, if a traffic originator's IP address is easily and quickly changed—as it can be, in some circumstances—then identifying him or her by IP address alone does little to help stop the unwanted traffic. The originator can, after all, resume sending the traffic using a different IP address.

[2] It is a common misconception that unwanted traffic on the Internet is purely a product of end host insecurity. However, that is not the case. Suppose, for instance, that a solution to the insecure end host problem were suddenly installed on every host connected to the Internet. One property of that solution, of course, would be that the legitimate users of that host could then run any software they pleased — say, the latest version of SETI@home [1] — completely safely, without any risk of compromising their own data or other software on the host. Likewise, a simple "DDoS@home" application could (and no doubt soon would) be distributed, which, in return for performing some pleasing service to the user — showing attractive images of some sort, for instance — would borrow some of the end host's spare CPU cycles and bandwidth, for use in the distributor's "virtual botnet". The user would be completely unharmed—protected by bulletproof end host security — and may or may not even know, or care, how his or her (cost-free) spare cycles and bandwidth are being used. The distributor's botnet would be as large and effective as before, and ready to continue its DDoS attacks. Clearly, then, DDoS attacks do not depend solely on the ubiquitous availability of insecure end hosts.

Conversely, if senders of traffic *can* be identified by a persistent attribute, then a fairly simple policy can be formulated regarding acceptable consumption of network-layer resources by a sender. Senders who violate this policy would be deemed to be DoS attackers, and their traffic would be blocked. A DoS attack is thus reduced to the equivalent of a transient "flash crowd": each attacker can only consume roughly as much of the target's resources as a "normal" sender would.

IV. Making the Internet Accountable

This section outlines the five steps involved in creating accountability among a group of ASes, where peering ASes have a contractual basis for trust, shared keys, and agreement on operating procedures, but non-peered ASes in the group need not have any trust or shared keys with each other. This assumption is no stronger than the kind upon which the Internet already relies, where peered ASes must agree on the meaning and use of BGP community strings for interdomain routing to function properly [30].

A. Step 1: Identifying the customer

Accountability is based on the network having the ability to associate the source IP address of a packet with a particular customer. We define the *customer* of an AS as an entity to which the AS has issued one or more subnets of IP addresses, agreeing to deliver traffic to those IP addresses to that entity. (We call an AS with customers an ISP.) Some customers, such as DSL customers, may have only a single IP address in their subnet. Others (e.g., small companies) might have many addresses.

To track the association between source IP address and customer, we require ISPs to upgrade their customer relationship management (CRM) systems so that each customer record stores not only the billing information of the customer, but also the IP addresses assigned for use by that customer. The most common methods for connecting to an ISP already create such an association, and the few methods that do not meet it are easily changed to ones that do. For example, companies connecting over a leased-line (e.g., a T1) can be associated when the leased-line is physically connected. DSL customers connect to their ISP through a Broadband Remote Access Server (BRAS) that identifies and authenticates the customer by either the Permanent Virtual Circuit (PVC) or the user name and password of the PPPoE connection over which they connect. Cable companies authenticate their users at the Cable Modem Termination System (CMTS) by the cable loop and modem MAC address over which they connect.

For ISPs that dynamically assign customers IP addresses, either through DHCP or by employing NAT and assigning customers different port numbers behind a single IP address, the ISP will need to retain DHCP and NAT logs for some limited period of time. With these logs, the combination of an IP address, port number(s) and time precisely identifies an individual customer.[3]

B. Step 2: Per-customer ingress filtering

For defensibility to be possible, a host receiving a packet must be able to extract enough information from the packet to uniquely identify the customer that sent the packet. To ensure that the source IP address of a packet can be traced back to the customer that sent it, participating ISPs deploy strict ingress filtering [14] over all their customers. This means that as a customer connects, the router configuration management system will configure a packet filter on the interface/port to which the customer has connected that drops any packet with a source IP address outside the subnet(s) assigned to the customer.

C. Step 3: Defensibility through at-source filtering

We implement defensibility by creating a system of Filter Request Servers (FRSes) that take a target host's request that traffic to it from an IP Address be stopped, and relay the request to the FRS in the ISP where the customer using that IP address is connected. Since the customer's immediate ISP is the only entity that can easily map an IP address to a customer, it is uniquely capable of implementing filtering based on persistent attributes, regardless of any tricks the customer might use to evade it. For example, should the customer change its IP address (say, if it is connecting via a "hotspot" network), the FRS can ensure that filters requested against the customer are updated with its new address, and move with it.

The technical difficulty in implementing near-the-source filtering is in verifiably conveying filtering requests from target hosts to source hosts' ISPs. If it were possible to spoof, forge or replay filter requests, then the filtering mechanism itself could be used for DoS attacks. Section V describes our design for such a verifiable system.

Performing filtering in the ISP actually connected to the source has several additional advantages: downstream ASes are spared the unwanted traffic; multiple paths, requiring multiple filters, are less likely to exist at the filtering point; and habitual offenders — presumably botnet participants — can be recognized and dealt with.

Cases where the "customer" consists of a large number of hosts may be handled in one of two ways. If the customer — say, a university or enterprise — has a relatively well-managed network, and wants to avoid all-or-nothing filtering, then it can deploy its own FRS, and become the equivalent of an ISP serving its end hosts as if they were individual customers. Thus accountability will

[3]We do assume that customers cannot obtain a new address via DHCP, or reuse a NAT port number, in periods shorter than the maximum round trip time plus network time synchronization. However, this assumption is typically valid, as network time sync under 100 ms is easy to obtain and round trip times more than a few seconds are rare. In any case the period of reuse is under the ISP's control and not the customer's, so a participating ISP can choose appropriate values.

be preserved, and a single rogue end host can be filtered without interrupting service to the customer's other hosts.

Alternatively, the customer — say, a public wireless "hotspot" — may be unwilling to take the trouble to police its hosts. In that case, its best course of action is simply to have its ISP treat it as if it were a separate ISP that has not deployed our Internet accountability system. We explain next how our system deals with traffic from such ISPs.

D. Step 4: Dealing with the outside world

Not all ISPs will implement Internet accountability, and yet communication between hosts in participating and non-participating ISPs must continue as least as well as it does today. Packets carried by a non-participating ISP may have spoofed source addresses and will almost certainly not be impeded by filter requests, so there must be some mechanism beyond the FRS to deal with them. Our solution is to leverage the existing relationships among ASes. Participating, hence accountable, ASes will configure their border routers to set the "evil bit" on any packet entering from a non-participating, unaccountable, AS. Packets entering from other participating ASes will have their evil bit left unchanged. With this simple change, customers of an accountable ISP receiving a packet with the evil bit clear will know that the source address of the packet can be tied to a customer, and therefore that traffic from that customer will stop in response to a filter request. Packets with the evil bit set, on the other hand, are not necessarily malicious, but they may be, and moreover their source addresses may have been forged.

To prevent unaccountable traffic from overwhelming a host, router, or link, accountable ISPs and servers preferentially drop packets with the evil bit set when an overload condition exists. The evil bit and this priority scheme can be implemented using the DiffServ [7] hardware and configuration commands that already exist in essentially all ISP routers. The evil bit could either be assigned from the six bits in the DiffServ Code Point (DSCP) [24] or represented by a well-known value of the DSCP.

Otherwise-innocent customers of unaccountable ISPs would be unaffected by this scheme, *except* when trying to communicate with a host under DDoS attack. In that case, the customer would be completely cut off — its packets having been dropped — whereas a customer with a fully accountable path to the host would likely still be able to get through. While minor, this advantage could serve as an incentive to ISPs to embrace accountability for the sake of their customers.

E. Step 5: Stopping Reflection attacks

In a "reflection attack" [27], the attacker, instead of sending packets directly to the target, sends packets to arbitrary hosts that cause those hosts to send packets in turn to the target. In the simplest example, the attacker sends TCP SYN packets to an arbitrary host, with the target's IP address as the source address. The host then sends its SYN-ACK packets to the target. The result is that the attacker is subjected to the same volume of traffic, but from a far larger number of sources.

In a universally accountable Internet, ingress filtering would make source address spoofing, and hence reflection attacks, impossible. On *partially* accountable networks, however—such as the Internet after partial deployment of accountability—reflection attacks provide a means by which attackers in unaccountable parts of the network can "launder" their attack traffic of the evil bit that would normally be attached to it by reflecting it off hosts in accountable networks. Targets of a reflection attack in an accountable network could request filters against the reflectors, but the attack traffic can easily dodge this filtering by choosing different reflectors.

A simple solution to this problem is for hosts to ensure that every response to an incoming packet preserves the "evil bit state" of the packet to which it is responding. For example, every TCP SYN-ACK should bear an evil bit with the same value as its corresponding TCP SYN. Thus, reflection attack traffic is rendered effectively no different from direct attack traffic, and can be filtered or rate-limited along with all other unaccountable traffic during an attack. For protocols where the kernel cannot associate outgoing packets with the incoming packets that caused them (e.g., DNS, which uses UDP transport), it will fall to the application to carry over the evil bit state of the incoming request to the outgoing response. This can be achieved by extensions to the socket API so that evil bit state can be queried when a packet is received and set when a packet is sent.

Of course, this solution requires that hosts be upgraded to recognize and preserve evil bit state. Like accountability itself, this upgrade will at best be deployed slowly across Internet hosts, presumably bundled together with other updates and bug fixes to minimize distribution costs. Hence ISPs that deploy accountability will have to deal with hosts that haven't yet been upgraded. As an interim measure, ISPs can set the evil bit on *all* traffic (including, obviously, reflected traffic) originating at hosts that haven't yet been upgraded. Fortunately, testing whether a host has upgraded is easy: for each protocol vulnerable to reflection attacks (TCP, ICMP, and so on), and each port number, the ISP sends an initial packet to the host being tested, with its evil bit set. The host "passes" the test if and only if the response to these probes always has the evil bit set. Hosts running unpatched kernels or applications that are used as reflectors may also end up with many filters being requested against them, which will also result in the host having its traffic marked with the evil bit by the ISP. This is fair, as unpatched hosts used as reflectors *are* a source of DoS traffic, even though an indirect one.

While the idea of updating end-host network stacks may seem onerous, our observation is that reflection attacks

represent a fundamental problem. By their nature, reflectors allow an attacker to cause an innocent host to send traffic to a target of the attacker's choosing. This means that *any* approach to controlling DoS will involve either (1) not responding to any traffic with an untrustworthy source address, or (2) implementing inheritance of some property of the incoming packet in the response packet. But the first option would prevent hosts using one accountability scheme from communicating at all with hosts under a different accountability scheme, and hence destroys backward compatibility, leaving option 2 as the only acceptable one.

The next section describes the FRS in more detail, examines attacks on the system, and explains how untrustworthy ASes are discovered and evicted.

V. Filter Propagation

Our solution requires a complete system for relaying filter requests among accountable ASes, to effect defensibility. In this section, we give a detailed design of such a system.

A. Goals and Assumptions

The system must have the following properties:

- *Effectiveness:* The system should be effective in blocking all traffic traveling (solely) through participating ASes, from a particular source to a destination that has requested that traffic from that source be blocked.
- *Security:* A request should only be honored if it actually emanates from the IP address that is requesting the blocking.
- *DoS Resilience:* The mechanism itself should be robust to DoS attacks of any kind.
- *Deployability:* The system should avoid prohibitively complicated or expensive infrastructure (such as a global PKI), or out-of-band mechanisms in the normal case.

Note that the first goal is quite limited, since it says nothing about traffic passing through non-participating ASes.

B. The System

1) Filter Request Servers: Every accountable AS and ISP must run a Filter Request Server (FRS). Customers request a filter by sending their *own* ISP's FRS a *filter request* consisting of the IP address whose traffic should be blocked, the time the offending traffic was received, the source/destination port numbers (if any), and for how long the traffic should be blocked. The time and port numbers may be needed by the FRS at the traffic's source to match the traffic to a customer if DHCP or NAT are being used. Since the ISP running the FRS is accountable, it will already be enforcing ingress filtering on its customers, and the FRS can easily identify the customer making the request.

2) Filter Request Processing: First, the FRS determines if that customer is entitled to issue a filter request. By default, customers should have very limited ability to do so, since they should have little need. The ISP may forbid most of its customers to issue filter requests at all, or it may allow them to issue requests at a very low rate, in response to harassment from individual users.

However, on request, and presumably upon paying a mild premium, a customer—typically, a server operator—would be allowed to issue filter requests on a large (that is, essentially unlimited) scale. Customers of this type would also be allowed, if they desire, to issue filter requests from a single IP address on behalf of a block of IP addresses belonging to the same customer—perhaps only from the particular designated IP address of a relatively secure host (to minimize the threat of a single host compromise resulting in a flood of requests).

3) Filter Request Forwarding: When two peered ASes agree to implement accountability with each other, they tell each other the IP addresses of their FRSes.

Upon authorizing an incoming request, the FRS then determines, based on the IP address specified as the traffic source, a "next hop" AS to which to forward the request. By running an iBGP session with any local router, the FRS can obtain this "next hop AS" information. The FRS then forwards the request to the next-hop AS's FRS, appending the correct source address from which the sending FRS received the request. With the source address verification provided by ingress filtering, the next-hop FRS can verify that the request has indeed been forwarded from a neighboring AS's FRS. (For extra security, the FRSs can also share an IPSec SA. The overhead will be small, given the small number of neighboring ASes for a given AS, but ought not to be strictly necessary.)

The request builds up a chain of addresses as it travels, starting with the originator of the request, and continuing with a sequence of AS' FRS IP addresses. When the request reaches the FRS of the ISP that owns the IP address in question, the ISP acknowledges the request, by passing the acknowledgment back along the same chain of FRSs as the one the request took. This backtracking mechanism prevents asymmetric routes from causing acknowledgments to pass through non-participating ASes.

The FRS in the ISP of the offending traffic source uses the IP address, time and port number listed in the request to determine which customer's traffic needs to be filtered, and then translates that customer identity into a filter location and source/destination IP addresses to filter. The FRS then contacts the AS's router configuration system, requesting the application of the appropriate filter at the appropriate location. As with filter requests themselves, the instructions from the local FRS can be authenticated based on their originating IP address—assumed reliable, given full source address verification—or based on an IPSec SA, for extra security.

Filter requests may include an expiration time—say, a day, or a week, or a month—to allow for the possibility of the target "reforming" (for example, repairing a compromised end host that had been incorporated into a botnet). The requested duration of a filter would likely be chosen

by the requesting customer based on the frequency and volume of unwanted traffic emanating from the source being filtered.

ISPs may also take additional measures beyond application of the filter. For example, if a particular customer repeatedly triggers numerous filters against itself, thus imposing extra costs on the ISP, then the ISP may respond by imposing extra limitations on the customer. These might include rate-limiting of traffic; imposing extra charges; or, more positively, offering to sell root-kit scrubbing or computer maintenance services to the customer. The effort required from an ISP to respond to filters against a customer is bounded, because if a customer has more filters requested against than the ISP has resources to handle (e.g., due to router limitations), then ISP can set the "evil" bit on all packets emanating from the customer.

Consider the example shown in Figure 1. ASes 1, 2, 3 and 4 are all accountable and are interconnected as shown. AS1 is a large DSL provider. Its customers connect to the network through a DSLAM (DSL Access Multiplier). The DSLAM connects to the rest of the network via router R1. R2 and R3 provide connectivity to other ASes. Since this AS is accountable, it runs an FRS, which communicates with the local CRM (Customer Relationship Management) system to access customer information, and has the credentials to implement filters at routers and DSLAMs. AS2 and AS3 are large transit providers and do not have any end-hosts. For accountability, they need to only house FRSs. AS4 is a large web-hosting provider and has a server W1 which is under attack. W1 sends a filter request to FRS-4. FRS-4 has an iBGP peering session with R6 and learns about BGP routes to the offending IP address, which resides in AS1. It decides to forward the filter request to AS2. It looks up AS2 in its list of FRS's that it knows about and sends the request to FRS-2. FRS-2 similarly then forwards the request to FRS-1. FRS-1 looks up the IP address in the local network routing protocol, such as OSPF or IS-IS, and finds out that the DSLAM is assigned that address. It looks up the relevant records in the CRM and finds that the offending address is currently used by a customer with a DSL subscription, which happens to be the home with E1 and E2. Since the CRM only knows about a single account and single IP address on that DSLAM port, it cannot distinguish between E1 and E2, the FRS implements a filter at the DSLAM for that port. The filter blocks any traffic from E1 or E2 to W2. If the home with E1 and E2 had an FRS that it had registered with FRS-1, then FRS-1 would have forwarded the request to this home FRS, which would then have to determine which of E1 and E2 sent the offending traffic, perhaps by looking at the source port number and timestamp in the filter request.

Note that while FRS-4 had forwarded the request to FRS-2, it could have instead picked FRS-3. The choice of path does not matter as long as a chain of trusted FRSs is traversed to FRS-1. If AS1 was not an accountable network, traffic from it would most likely be marked as evil, and then in that case W1 would not generate a filter request to FRS-4 – instead it would consider requesting rate limiting of evil bit traffic. If AS1 did not have an FRS and AS2 and AS3 nonetheless allowed traffic from AS1 to not be marked as evil, it would be their responsibility to implement the filter at R4 and R5.

C. Attacks

1) Spoofing Attacks: As long as participating ISPs perform source address verification, customers of participating ISPs cannot launch spoofing attacks. Non-participating ISPs can, of course, be the source of arbitrarily large amounts of spoofed traffic. But traffic from non-participating ISPs will be marked with the evil bit, and, as mentioned above, we assume that such traffic will necessarily require massive, relatively indiscriminate filtering in the event of a DoS attack. (e.g., a lower queuing priority or collective rate-limiting). The end result is that traffic from non-participating ISPs to servers in participating ISPs will see performance roughly equivalent to today, where DoS traffic can swamp out desired traffic. Traffic between hosts in participating ISPs should see DoS-free performance, as servers can protect themselves from DoS.

2) DoS Attacks: The filter request system must itself be robust to DoS attacks of any kind, otherwise it will suffer from the very problem it is trying to solve. The following are the three major classes of DoS Attack and how the FRS deals with them.

Overloading the FRS with requests, to prevent it from accepting requests from customers under DoS attack. The FRS itself is capable of causing routers to block traffic, and can therefore block attack traffic at the source on detection. This applies both to attackers from within the same ISP, who can be filtered directly, and those whose requests come via other ISPs' FRSs.

Requesting so many filters for fellow DoS attackers that the ISP refuses to apply more filters to them, allowing them to launch DoS attacks with impunity. All routers have limits to the number of individual packet filters they can apply to traffic flowing through them, so at some point more filters may be requested against a customer's addresses than the router can implement. Rather than allowing such a user to send traffic with impunity, the ISP would remove all the filters on the customer and instead set the evil bit on all the customer's traffic. This allows the ISP to control its resource outlays and upholds the principle of accountability at the same time.

Requesting numerous filters for innocent customers, with the goal of causing them to be identified as habitual DoS attackers themselves. The application of the filters themselves is harmless, since the target was presumably uninterested in sending traffic to the attacker anyway. The goal of such a "framing attack" would be to cause unwarranted ISP action against the targeted customer. However, an ISP can easily check to see if a customer is being framed by examining the

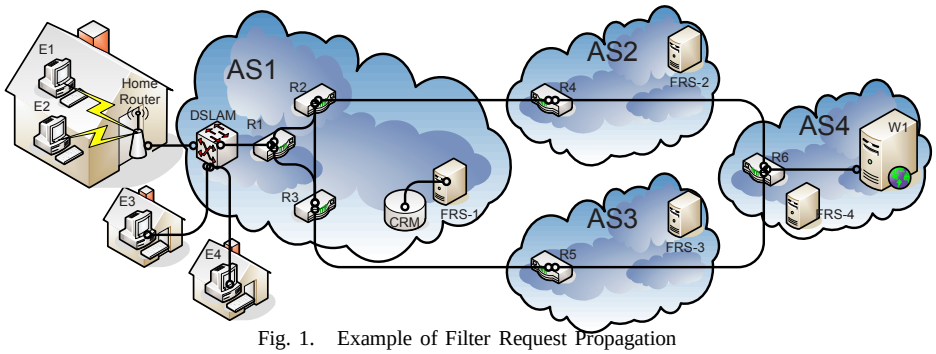

Fig. 1. Example of Filter Request Propagation

packets sent by the customer before taking action - if there are no packets from the customer to the filter requester, there is a framing attack underway.

A compromised server cutting off traffic to itself. If a server itself is compromised and requests long-lived filters for sources whose traffic is actually desired, there is a possibility of blocking wanted traffic even after the compromise has been repaired. However, the server's local FRS will have copies of every filter request it sent while compromised, and these filters can be recalled. Part of the process of recovery from the compromise would be revocation of those filters. Issuing and processing revocations of filters would be roughly identical to the issuing and processing of the filters themselves.

VI. ISP (or Other AS) Misbehavior

The greatest threat to our accountability scheme is a an ISP that claims to be following the accountability regimen — thus exchanging traffic without the evil bit set — while in fact not implementing either identification (i.e., ingress filtering) or defensibility (i.e., on-request filtering).

Detecting an ISP that fails to apply on-demand filtering is easy, as other ISPs' customers will immediately notice that their filter requests are being ignored, and can raise the alarm that that ISP is not honoring its agreements. The more difficult case is that of an ISP who fails to honor its commitment to apply proper ingress filtering to its customers. In that case, its customers may launch DDoS attacks that do not implicate the ISP—indeed, they may implicate other ISPs.

Cases of ISP malfeasance at ingress-filtering are likely to be rare (as malicious BGP misconfiguration is rare), but require out-of-band methods to resolve. For example, if an ISP receives complaints that another ISP is failing to filter attack traffic (claiming to be) from its customers, it can sample the traffic arriving from peer ASes, find the ASes from which the attack traffic is arriving, and demand that those ASes deal with the problem—either by properly filtering customers, or by enforcing contractual terms to ensure that upstream ASes do likewise. Those ASes in turn can follow the same procedure, and eventually an AS that fulfills its obligations will confront one that doesn't. The former, having not been satisfied with the latter's effort to stop the flow of DoS traffic, will then simply evict the AS from the accountability club by setting the evil bit on all traffic from the evicted ISP.

This approach is very different from statistical approaches to DDoS defense, such as IP traceback. In IP traceback, every DoS attacker has to be identified individually, by IP address, for each target in turn, so that its traffic can be ignored. In our approach, ASes only need identify sources at the granularity of the offending AS, and only need to identify one consistently unfiltered attack traffic source, once, to implicate an AS. Finally, beyond simply blocking traffic from a particular attacker, the community of network operators has recourse — an exposed AS can be designated as unaccountable and its traffic treated accordingly. Moreover, such identification may include penalties specified in the ISP's peering agreements. If these penalties are severe enough, then we can expect that their deterrent effect will make the need for this kind of sampling-and-tracking procedure quite rare.

VII. Economic Viability

In this section, we assess the economic viability of our proposal, by comparing the cost of its deployment with the costs currently incurred by DoS victims.

A. Economic Cost of DDoS: Price of conventional defenses deployed within an AS

A conventional tactic that some enterprise networks use for DoS defense is the combination of "scrubbing" devices with heavy over-provisioning of WAN connectivity. A scrubber is a network device that is typically placed at the edge of the enterprise network, facing the WAN connection. The scrubber inspects all inbound traffic and filters out "malicious" packets and connections. However, this can only protect connectivity within the enterprise network – since the WAN connection can still be flooded, the enterprise network. Thus typically the WAN connection it is heavily over-provisioned to match the expected attack volume, and enough scrubbers are deployed to handle the size of the connection. An example of a scrubber device is the Cisco Guard XT 5650, which can scrub traffic at a line rate of 1Gbps and retails for roughly $68,000 [8].

Instead of keeping the scrubber device continually in the path between the local network and the WAN, and thus paying a latency penalty, it can be coupled with

a Cisco XT 5600 Traffic Anomaly Detector, which will detect a potential DoS situation and then shunt traffic to the 5650. A Cisco XT 5600 retails for about $45,000 [9]. The administrators of an enterprise network would estimate the volume of the largest attack that they wish to defend against in multiples of OC-12s, and then lease that much connectivity from multiple ISPs. They would then couple each OC-12 (622 Mbps) with one Cisco XT 5650 and possibly an XT 5600. An OC-12 from a "tier-1" ISP will cost about $30,000 per month [17], [31]. The initial setup cost for a leased line is about $6,000 [23]. Thus for an organization wishing to defend itself against an attack stream of X Mbps for Y years, the total cost will be about $ ((30000 * 12) Y + (45000 + 68000 + 6000)) X / 622 which is about $(580 X Y + 190 X).

B. Economic Cost of DoS : Price of defense services offered by ISPs

An alternative strategy available to enterprise networks today is to take advantage of scrubbing services provided by large ISPs such as MCI [26] and Saavis [16]. The MCI service costs between $2,000 per month for a T-3 line (45Mbps) and $69,000 per month for an OC-48 line (2.448Gbps). This is in addition to the cost of raw IP connectivity, which we previously quoted as $30,000 for an OC-12 (622Mbps). Setup fees range from $200 to $2,500. Thus for an organization wishing to defend itself against an attack stream of X Mbps for Y years, the cost will be about $12 X Y * 69000 / 2448 which is about $(338 X Y), plus the setup fee.

C. Economic Cost of Accountability: Implementing ingress filtering and deploying FRSs

Estimating the cost in dollars that an ISP would have to pay to upgrade its network to support our system is not easy. The bulk of the expense will not be the new equipment and software that must be purchased — an FRS is, after all, no more complicated than a load-balanced cluster of machines running a database application. Rather, the majority of the cost will come from integrating the FRS into the ISP's existing systems. New fields will need to be added to the databases in the Customer Relationship Management (CRM) systems. New software interfaces to the router configuration management systems will need to be coded. Customer service personnel will have to be trained on new procedures and customer documentation updated. Activities like these are carried out continuously at major ISPs, using internal staff and external contract consultants, but ISPs rarely disclose the resulting costs.

However, we can bound the costs of implementing accountability by analogy to two network upgrades for which there is data in the public record. In 2002, cellular carriers in the US were required by the Federal Communications Commission (FCC) to upgrade their networks to support Number Block Pooling and Local Number Portability (a.k.a. wireless number portability). Number Block Pooling increases the granularity of phone number allocations, roughly analogous to Classless Interdomain Routing (CIDR). Local Number Portability allows customers to keep their phone numbers, even if they change carriers. Each of these upgrades required changes far more extensive than would be necessary to implement accountability.

Sprint Wireless operates a network supporting 13M subscribers, and estimated the cost of its Number Block Pooling upgrade at $59M. Implementing Local Number Portability required changes to over 70 systems inside the carrier, from their routing architecture to their external websites, and was estimated to cost $36M [13]. Leap Wireless, which provides service in dozens of cities across the US, estimated the costs of number pooling at $6M, and local number portability at an additional $2M [12]. They have a subscriber base of 1.62M as of September 2005 [22]. The cost estimates by both Sprint and Leap include all aspects of the network upgrades, including cost to update CRM software, cost for upgrading network elements, costs for new documentation and training for both network and customer support personnel, etc..[4]

According to the US Congressional Budget Office, the 30 OECD nations have 55.8M broadband subscriptions [25], and about 17M websites. Thus, if we use the cost of Number Block Pooling as representative of the cost of implementing accountability for all the broadband connections in the OECD nations, we arrive at about $188M to $231M. If we instead rely on Local Number Portability, we arrive at $63M to $141M.

An additional cost occurs in upgrading end host stacks to respond to evil bit traffic with the evil bit set in their replies. Since our accountability system allows incremental deployment, this update to end host OSes and reflection-susceptible applications can also occur incrementally. This update can be included with any more critical updates and does not require special user attention or intervention, thus minimizing the cost for the user.

D. Comparing the Economic Cost of DDoS to the Economic Cost of Accountability

Recall that we've estimated the rough cost of conventional protection for an organization against an X Mbps DDoS attack for Y years to be somewhere between $(338 XY) and $(580 XY + 191 X). We assume that organizations acquiring this protection will expect to be protected against a DDoS attack consisting of up to 50,000 broadband hosts, each with a minimum of 128Kbps upstream bandwidth—that is, an attack stream of up to 6.4Gbps. If we consider DoS defense over 3 years, the cost of conventional protection for 1 organization is roughly $6.5M to $12M. We estimate the cost for implementing accountability to be in the range $63M to $231M. Hence if there are at least 36

[4] Also noteworthy is that Sprint's filings argued against requiring carriers to implement these network upgrades, while Leap's filings argued in favor. This suggests that these estimates do bracket the true costs.

networked organizations that are afraid of being attacked by a large botnet army, it will be cheaper for the entire Internet to implement accountability.

While the ISPs willing to spend money to defend themselves against DoS are different from the ISPs that tend to serve compromised end hosts, we believe that competitive pressure among ISPs that serve large organizational customers will eventually drive them towards the less expensive option of subsidizing the widespread implementation of accountability, in order to allow them to undercut competitors offering the more expensive alternatives.

VIII. Conclusions

This paper proposes a simple approach to blocking DoS attacks, based on the concept of accountability. It allows hosts to stop receiving undesired traffic, and ISPs to identify habitual offenders within their network. Our system employs an "evil bit" label on packets, to allow a subset of the Internet to implement accountability and still obtain some relief from DoS. It thereby solves the problem of incremental deployment that ingress filtering and other DoS prevention techniques have faced in the past.

Our system does require some changes to ISP operations, and thus introduces costs, but since the current Internet is prone to DoS attacks, any solution will require change, and will therefore have a cost component to it. We have made rough estimates of the cost of deploying accountability, and compared it to the amount that organizations are willing to spend today to defend themselves against attacks. Our analysis shows that if there exist today at least a few organizations (say, 36 large corporations) that wish to defend themselves against large botnet attacks for a few years, then implementing our system for the entire Internet will cost less than the currently available alternatives. We believe that far more than 36 such corporations exist on the Internet today.

References

[1] D. P. Anderson, J. Cobb, E. Korpela, M. Lebofsky, and D. Werthimer. SETI@home: An experiment in public-resource computing. *Communications of the ACM*, 45(11), November 2002.

[2] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet denial-of-service with capabilities. In *Proc. ACM HotNets-II*, November 2003.

[3] K. Argyraki and D. Cheriton. Active internet traffic filtering: Real-time response to denial-of-service attacks. In *USENIX*, 2005.

[4] H. Ballani, Y. Chawathe, S. Ratnasamy, T. Roscoe, and S. Shenker. Off by default! In *Proc. ACM HotNets-IV*, November 2005.

[5] S. Bellovin. The security flag in the IPv4 header. RFC 3514, 2003.

[6] J. Binkley and S. Singh. An algorithm for anomaly-based botnet detection. In *Proc. Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet*, 2006.

[7] S. Blake, D. Black, M. Carlson, E. Davies, and Z. W. W. Weiss. An architecture for differentiated services. RFC 2475, 1998.

[8] CNET Shopper prices for Cisco Guard XT 5650. http://shopper.cnet.com/Cisco_Guard_XT_5650_security_-appliance/4014-3243_9-31208569.html.

[9] CNET Shopper prices for Cisco Traffic Anomaly Detector XT 5600. http://shopper.cnet.com/Cisco_Traffic_Anomaly_Detector-_XT_5600_security_appliance/4014-3243_9-31208808.html.

[10] E. Cooke, F. Jahanian, and D. McPherson. The zombie roundup: Understanding, detecting, and disrupting botnets. In *Proc. Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet*, 2005.

[11] S. D. Crocker. Protecting the Internet from distributed denial-of-service attacks: A proposal. *Proceedings of the IEEE*, 92(9):1375–1381, September 2004.

[12] Reply comments of Leap Wireless. WT Docket No. 01-184, October 2001.

[13] Reply Comments of Sprint PCS. WT Docket No. 01-184, October 2001. http://gullfoss2.fcc.gov/prod/ecfs/retrieve.cgi?-native_or_pdf=pdf&id_document=6512767909.

[14] P. Ferguson and D. Senie. Network ingress filtering: Defeating denial of service attacks that employ IP source address spoofing. Internet RFC 2827, 2000.

[15] F. Freiling, T. Holz, and G. Wicherski. Botnet tracking: Exploring a root-cause methodology to prevent distributed denial-of-service attacks. In *Computer Security - ESORICS 2005: 10th European Symposium on Research in Computer Security*, 2005.

[16] C. Garretson. Savvis launches security service. Network World, December 2005. http://www.networkworld.com/news/2005/121905-savvis.html.

[17] A. Helland. The economics of large-scale data transfer: the benefits of fibre channel and SONET for high-performance, cost-effective data transport through WAN - storage networking. Computer Technology Review, December 2002.

[18] J. Ioannidis and S. M. Bellovin. Implementing pushback: Router-based defense against DDoS attacks. In *Proc. ISOC Symp. on Network and Distributed Systems Security*, 2002.

[19] S. Kandula, D. Katabi, M. Jacob, and A. Burger. Botz-4-sale: Surviving DDoS attacks that mimic flash crowds. In *NSDI*, 2005.

[20] S. Kent and R. Atkinson. Security architecture for the internet protocol. Internet RFC 2401, 1998.

[21] J. Layden. Phatbot arrest throws open trade in zombie PCs. The Register, May 2004. http://www.theregister.co.uk/2004/05/12/phatbot_zombie_trade/.

[22] Leap investor relations website. http://phx.corporate-ir.net/phoenix.zhtml?c=95536&p=irol-irhome.

[23] R. A. Mortimer. Investment and cooperation among internet backbone firms. UC Berkeley Economics, 2001. PhD Dissertation.

[24] K. Nichols, S. Blake, F. Baker, and D. Black. Definition of the differentiated services field (DS field) in the IPv4 and IPv6 headers. RFC 2474, 1998.

[25] U. C. B. Office. Economic and Budget Issue Brief, February 2004. http://www.cbo.gov/showdoc.cfm?index=5082&sequence=0.

[26] D. Pappalardo. MCI offers network protection service. Network World, June 2005. http://www.networkworld.com/news/2005/060605-mci-security.html.

[27] V. Paxson. An analysis of using reflectors for distributed denial-of-service attacks. *ACM CCR*, 31(3), July 2001.

[28] A. Ramachandran, N. Feamster, and D. Dagon. Revealing botnet membership using dnsbl counter-intelligence. In *Proc. Usenix Workshop on Steps to Reducing Unwanted Traffic on the Internet*, 2006.

[29] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical network support for IP traceback. In *ACM SIGCOMM*, August 2000.

[30] J. Stewart. *BGP4: Inter-Domain Routing in the Internet*. Addison-Wesley, 1998.

[31] T1-shopper.com prices for Verio connectivity. http://www.t1shopper.com/carriers/verio/.

[32] R. Vogt, J. Aycock, and M. Jacobson. Army of botnets. In *Proc. ISOC Symp. on Network and Distributed Systems Security*, 2007.

[33] A. Yaar, A. Perrig, and D. Song. Pi: A path identification mechanism to defend against DDoS attacks. In *Proc. IEEE Symposium on Security and Privacy*, May 2003.

[34] A. Yaar, A. Perrig, and D. Song. SIFF: a stateless internet flow filter to mitigate DDoS flooding attacks. In *Proc. IEEE Symposium on Security and Privacy*, May 2004.

[35] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *ACM SIGCOMM*, August 2005.