

Perimeter-Based Defense against High Bandwidth DDoS Attacks

Shigang Chen Qingguo Song

Department of Computer & Information Science & Engineering

University of Florida

Gainesville, FL 32611

{sgchen, qsong}@cise.ufl.edu

Abstract

Distributed denial of service (DDoS) is a major threat to the availability of Internet services. The anonymity allowed by IP networking, together with the distributed, large scale nature of the Internet, makes DDoS attacks stealthy and difficult to counter. To make the problem worse, attack traffic is often indistinguishable from normal traffic. As various attack tools become widely available and require minimum knowledge to operate, automated anti-DDoS systems become increasingly important. Many current solutions are either excessively expensive or require universal deployment across many administrative domains. This paper proposes two perimeter-based defense mechanisms for Internet service providers (ISPs) to provide the anti-DDoS service to their customers. These mechanisms rely completely on the edge routers to cooperatively identify the flooding sources and establish rate-limit filters to block the attack traffic. The system does not require any support from routers outside or inside of the ISP, which not only makes it locally deployable but also avoids the stress on the ISP core routers. We also study a new problem of perimeter-based IP traceback and provide three solutions. We demonstrate analytically and by simulations that the proposed defense mechanisms react quickly in blocking attack traffic while achieving high survival ratio for legitimate traffic. Even when 40% of all customer networks attack, the survival ratio for traffic from the other customer networks is still close to 100%.

Keywords: Network-Level Security and Protection

I. INTRODUCTION

A. Background

The goal of a DoS (denial of service) attack is to completely tie up the resources of a server, which prevents legitimate users from accessing the service. A successful DoS attack achieves two objectives:

overpowering the victim and concealing the attacker's identity. To overpower the victim, the attacker needs a strategy that small resource consumption at the attacker side causes much larger resource consumption at the victim side. For example, a small packet generated by the attacker causes a buffer space to be held for an extended period of time T at the victim. While the attacker can generate a large number of packets during T , the buffer space at the victim is going to overflow, which underlines the SYN flooding attack [1], [2], [3] and the connection table overflow attack. To conceal the attacker's identity, forged source addresses are often used in the packets sent from the attacker. In a DDoS (distributed denial of service) attack, multiple malicious hosts launch a coordinated offense against one victim, which increases the resources for the offense while making it harder to track down the attacker(s). Moore, Voelker, and Savage's work demonstrated that DoS attacks were widespread on the Internet. By using a novel traffic-monitoring technique, called "backscatter analysis", they observed 12,805 attacks on over 5000 distinct Internet hosts belonging to more than 2000 distinct organizations during a three-week period [4].

To mitigate DDoS attacks, much of the current research focuses on anti-spoofing such as ingress filtering [5], route-based packet filtering [6], and various IP traceback protocols [7], [8], [9], [10], [11], [12], [13]. Their effectiveness often depends on a universal deployment on the Internet. With a partial deployment, source-address spoofing remains feasible.

Sung and Xu pointed out that, while many existing techniques focus on tracking the locations of the attackers after-the-fact, little is done to mitigate the effect of an attack while it is raging on [14]. An intelligent packet-filtering solution based on IP traceback was proposed. The victim constructs an attack graph from the received traceback marks. Based on the graph, it identifies the "infected" edges and the "clean" edges with the former more likely to carry attack traffic. It then informs a line of defense to preferentially filter out packets from the "infected" edges based on the marks in the packet header. Note that the IP-traceback function must be implemented on the routers outside of the defense line to mark the packets before they reach the line.

Mahajan et al. proposed the aggregate-based congestion control (ACC) to rate-limit attack traffic [15]. The congested router starts with local rate limit, and then progressively pushes the rate limit to some

neighbor routers and further out, forming a dynamic rate-limit tree, which can be expensive to maintain. Every router in the tree performs filtering based on its share of rate limit, which is necessary because not all routing paths are always covered by the leaves of the tree. All routers in the tree measure the traffic arrival rates, which are propagated upstream towards the congested router, allowing it to know the total arrival rate and decide whether to continue the rate limiting.

B. Our Contributions

This paper proposes a class of perimeter-based defense mechanisms, which allows Internet service providers (ISP) to provide an anti-DDoS service to its customers. The edge routers of an ISP form a perimeter separating the customer networks from the rest of the Internet (Fig. 1). Our first contribution is to study how to turn the ISP perimeter into a defense barrier against DDoS attacks. Depending on how the edge routers communicate with each other, we present two defense mechanisms, DPM (defense perimeter based on multicast) and DPIT (defense perimeter based on IP traceback). Our second contribution is to design an IP traceback scheme that is deployed *only along a perimeter* to suit the perimeter-based defense solutions. This traceback scheme is more practical as it can be locally deployed; it is also more efficient than the existing ones as it specializes to the task of identifying the entry points instead of the paths of an DDoS attack. Our third contribution is to provide an evaluation framework to study the perimeter-based defense analytically and by simulations. Several performance metrics are proposed and studied.

The main difference between this work and [14] is that our defense perimeter is *self-complete* while their line of defense is not. In [14], the routers on the line of defense (also called perimeter) perform packet filtering, but it requires support from inside the perimeter and outside the perimeter. Inside the perimeter, the victim constructs the attack graph, identifies the “infected” edges, and informs the packet-filtering routers about these edges. Outside the perimeter, the Internet routers must support the IP-traceback scheme proposed in [14]. Our defense perimeter does not require any assistance from the victim except *optionally* to signal the occurrence of an attack. It does not require any assistance from outside the perimeter either. One of our proposed defense mechanisms also uses IP traceback, which is however deployed locally

on the perimeter only.

Differing from Pushback [15], we study a one-dimension defense perimeter instead of a two-dimension defense tree. It requires a new set of techniques to realize the similar goals. We believe a perimeter-only solution is more appealing due to administration and performance reasons. Some advantages of the *self-complete perimeter-based defense* are listed below.

- *Defense at the border and efficiency at the core:* In order to reach a customer network of an ISP, any attack traffic must enter the ISP first by passing an edge router. Hence, the perimeter is the earliest location of defense. By stopping an attack before it enters the ISP, the perimeter-based defense not only mitigates the attack but also minimizes the resources consumed by the attack traffic. Since the defense mechanisms are implemented solely at the edge routers, the core is kept simple and stateless.

- *Separation of attack traffic and legitimate traffic:* The further away the attack traffic is from the victim, the less it is mixed with the legitimate traffic. Hence, it is advantageous to perform blocking at the furthest possible locations, which reduces the collateral damage of blocking legitimate traffic.

- *Single administrative domain:* All edge routers of an ISP are under the same administrative control, which simplifies the deployment in terms of management, policies, and politics.

- *Extensibility:* The implementation of the perimeter-based defense by one ISP *does not require any assistance* from the other ISPs. On the other hand, we allow ISPs to cooperate for better performance. If neighboring ISPs both implement the perimeter-based defense, they can work together to extend the defense across multiple administrative domains. This provides a positive feedback and encourages ISPs to follow when more and more peer ISPs have implemented the system.

The rest of the paper is organized as follows. Section II provides the models of ISP and DDoS attacks. Section III and Sections IV propose two perimeter-based defense mechanisms. Section V shows how the neighboring ISPs cooperate to defend against DDoS attacks. Section VI presents the simulation results. Section VII addresses the limitation of the proposed defense system. Section VIII draws the conclusion. The proofs for the theorems can be found in Appendix A.

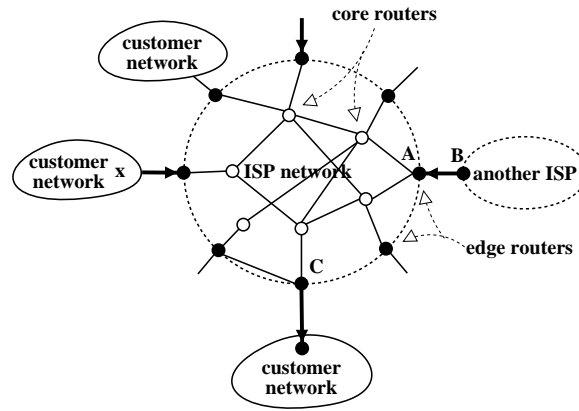


Fig. 1. Edge routers form a defense perimeter against DDoS.

II. MODELS

A. Internet Service Providers

Most businesses, institutions, and homes access the Internet via ISPs. An ISP network interconnects its customer networks, and routes the IP traffic between them. It also connects to other ISPs to provide the access to the rest of the Internet. Two neighbor ISPs may sign a contract to be each other's customers. As shown in Fig. 1, an ISP network has two types of routers: *edge routers* and *core routers*. An edge router has at least one direct connection to a customer network. It can be a BGP router connecting to an enterprise network, a router responsible for the regional cable Internet access, or a router at a local office for residential DSL or dialup Internet access. The core routers do not have direct connections to any customer networks. They route traffic between edge routers. To maximize the efficiency and improve the scalability, a common design philosophy is to push the complexity of network functions (e.g., packet classification, filtering, etc.) towards the edge while keeping the core simple and free of state information about specific traffic flows.

B. High Bandwidth DDoS Attacks

The goal of a high-bandwidth DDoS attack is to send a large amount of traffic to exhaust a target resource so that legitimate users cannot access the resource. The resource may be link bandwidth, buffer space, or processing capacity. The offending traffic can be characterized as an *aggregate* of packets [15]. A traffic

aggregate is defined by matching the fields in the packet headers (at IP, UDP, TCP, and/or application layers) against a set of values. For example, the traffic aggregate for a SYN attack consists of all SYN packets to a destination address/port pair, and the aggregate for a smurf attack consists of all ICMP echo-reply packets from a subnet to a destination address.

During a DDoS attack, the attack traffic is often indistinguishable from the legitimate traffic, which makes it difficult to block the attack traffic while letting the legitimate traffic through. Indiscriminate random dropping reduces the attack traffic to an acceptable level, but also blocks the legitimate traffic proportionally.

We assume all attack traffic is generated by some compromised hosts on peer customer networks, or generated from other locations on the Internet and then routed over via the neighboring ISPs. We do not address the problem that some ISP routers are compromised and then used to generate attack traffic or assist the attack in other ways.

C. Rate-Limit Filters

A rate-limit filter is a tuple $\langle \alpha, r \rangle$, where α is a traffic aggregate and r is the rate limit for α . After being installed at an edge router, it requires the router to shape/police the arrival traffic of α so that the acceptance rate is bounded by r .

A token-bucket implementation for rate-limit filters is presented below. Let s be the bucket size. Each filter is assigned two variables: c is a counter and t is a timestamp, which are initialized to be zero and the current system-clock value, respectively. If a packet that matches the filter is received, the following algorithm is executed.

RateLimit(a received packet that matches α)

- (1) $c \leftarrow \min\{c + (\text{the current clock value} - t) \times r, s\}$
- (2) $t \leftarrow \text{the current clock value}$
- (3) **if** ($c \geq \text{the packet size}$)
- (4) accept the packet

- (5) $c \leftarrow c - \text{the packet size}$
- (6) **else**
- (7) drop the packet

Consider a customer network x where some malicious hosts launch attack (Fig. 1). Because the attack traffic from x is mixed with the legitimate traffic from x , when the edge router of x attempts to block most attack traffic, it will block most legitimate traffic as well. Clearly, the rate-limit filtering is not designed to save the legitimate traffic from x . Instead it is to save the legitimate traffic from other customer networks that do not harbor attacking hosts.

D. Perimeter-Based Defense

The perimeter-based defense has two major tasks. The first is to identify the attack aggregates, and the second is to identify the flooding sources and install appropriate rate-limit filters on the edge routers connecting to the flooding sources.

The first task may be carried out automatically by special software [15] or manually by system administrators. We want to stress that an *attack aggregate* is not the collection of attack packets, but rather a traffic aggregate that *contains* the attack packets (as well as the legitimate packets of the same kind). It is difficult, if not impossible, to separate attack packets from legitimate packets of the same kind. However, it is a much easier job to identify an aggregate that is broader, including but not only including the attack packets. A straightforward one is “all IP packets to the server (or the subnet) under attack”, which can be reported by the server (or the router to the subnet) under attack, or by a properly-configured firewall on the routing path. This paper addresses how to block the attack packets in an attack aggregate but save the legitimate packets in the same aggregate. While the above aggregate is already quite narrow, more specific attack aggregates can be identified for specific types of attacks. For example, a SYN attack can be detected by the server whose SYN queue is kept full, or by an edge router (or firewall) configured with a security policy that measures the SYN rate and triggers a warning when the rate is beyond the server’s capacity (e.g., 10,000 SYN per second). In this case, the attack aggregate is “all SYN packets to the server”, including

both malicious and legitimate SYNs, which may be indistinguishable from each other.

This paper focuses on the second task: Assuming that the attack aggregate and the desirable rate for the aggregate are known, the problem is how to bring the total traffic volume to the desirable level, minimizing the overhead and the reaction time while maximizing the survival ratio of legitimate traffic. The proposed defense system will only interfere with the attack aggregates and attempt to save the legitimate packets within them. Our discussions will be centered around the algorithmic and protocol aspects of the perimeter-based defense for an ISP. The engineering issues of planning, auto-deployment, provisioning, and management are beyond the scope of this paper. It is conceivable that the security management tools from the major vendors (e.g., Cisco VMS, Netscreen-Global Pro, and Checkpoint Provider-1) can be extended to assist the deployment of perimeter-based defense in a large ISP.

In the text of this paper, the (ISP-side) edge routers are the default place for implementing the perimeter-based defense. However, the actual deployment can be done on edge firewalls or even dedicated devices behind the edge routers. This does not change the fundamentals of the proposed defense system but allows the flexibility of adding resources to handle large customers.

E. Terminology

Consider an aggregate α . A packet belonging to α is called an α packet. The *arrival rate* of α at an edge router is defined as the total size (number of bytes) of α packets received by the router from outside of the ISP per unit of time; the *acceptance rate* is defined as the total size of α packets that are forwarded by the router into the ISP network per unit of time. The acceptance rate may be smaller than the arrival rate due to rate-limit filtering. The edge router connecting with the destination network of α is called the *exit router* of α ; the link between the exit router and the destination network is called the *exit link*. The actual data rate of α observed on the exit link is called the *exit rate*, denoted as $A(\alpha)$.

In the description of our system, it is not important to distinguish between multiple attackers and a single attacker launching the offense from multiple Zombies. When we say “multiple attackers”, we mean either of the two cases.

III. DEFENSE PERIMETER BASED ON MULTICAST (DPM)

A. Overview

Our first perimeter-based defense mechanism is called DPM, in which the edge routers of an ISP form a designated, exclusive multicast group. The address of the group is local to the ISP and any external join requests must be rejected.

Given a DDoS attack, the edge router connecting to the victim network is responsible of coordinating the defense and is thus called the *coordinator*. The coordinator initiates the defense process after it receives an anti-DDoS request, consisting of the description of the attack aggregate α and the desirable rate $D(\alpha)$ of the aggregate. The anti-DDoS request may be generated by the server under attack, by the management console from the customer/ISP side, or by firewalls, routers, and intrusion detection devices based on pre-configured policies. The coordinator starts DPM by instructing the other edge routers to install rate-limit filters for the attack aggregate α . It then monitors the exit rate of α and periodically multicast a new base rate to the other edge routers, which update their rate limits for α according to the received base rate. This process repeats until the exit rate converges to the desirable rate.

During the attack, the exit rate $A(\alpha)$ to the victim can be much higher than the desirable rate $D(\alpha)$. The goals of DPM are 1) to reduce the exit rate to a range close to the desirable rate, i.e., $(1-\tau)D(\alpha) \leq A(\alpha) \leq (1+\tau)D(\alpha)$, where τ is a small constant (e.g., 5%), and 2) to minimize the amount of legitimate traffic that is mistakenly blocked by the edge routers. The victim's capacity is expected to be at least $(1+\tau)D(\alpha)$.

B. Detailed Description

The coordinator relies on a single type of multicast messages to communicate with the other edge routers. The message format is RATE(α, r), where r is the base rate that the edge routers should enforce on the attack aggregate α . After receiving this message, an edge router creates a rate-limit filter for α if it does not have one, and then sets the rate limit of the filter based on r .

The legitimate traffic may have different arrival rates at different edge routers. In Fig. 1, under normal conditions, the traffic entering the ISP network from the edge router A is likely to be heavier than from

other edge routers. Traffic-differentiation policies can be used to capture such differences. A traffic-differentiation policy assigns a coefficient (≥ 1) to an aggregate, which can be as broad as all TCP traffic or as narrow as HTTP traffic to a specific server. Each edge router is configured with a list of policies, which are updated by the ISP based on traffic statistics. When an edge router receives $\text{RATE}(\alpha, r)$, it checks whether there is a policy that matches α . If there is, the router sets the rate limit of α to be $c \times r$, where c is the coefficient of the policy. If there is not, the router sets the rate limit to be r .

DPM consists of two phases. The first phase rapidly reduces the exit rate $A(\alpha)$ below the desirable level $D(\alpha)$. The second phase iteratively improves $A(\alpha)$ to be close to $D(\alpha)$.

- *Phase One:* The coordinator starts with a multicast message $\text{RATE}(\alpha, D(\alpha))$, which sets the base rate to be $D(\alpha)$ at all edge routers. After that, the coordinator continuously measures the exit rate $A(\alpha)$. It reacts to the new value of $A(\alpha)$ periodically after each time interval T , which should be greater than the maximum round-trip delay between any two edge routers and should allow enough time for the new rate limits to take effect at the edge routers. For instance, a reasonable value for T can be half a minute, as the actual Internet round-trip delay is typically in tens or hundreds of milliseconds.

The coordinator maintains a variable r_p , which is the base rate that it sends out in the previous RATE message. After each time interval T , it takes one of two actions. If $A(\alpha) > (1 + \tau)D(\alpha)$, the coordinator calculates $r = r_p \times \min\{D(\alpha)/A(\alpha), 1/2\}$, and sends a $\text{RATE}(\alpha, r)$ message, which reduces the rate limits of α at all edge routers at least by half. If $A(\alpha) \leq (1 + \tau)D(\alpha)$, it enters the second phase.

- *Phase Two:* After each time interval T , if $(1 - \tau)D(\alpha) \leq A(\alpha) \leq (1 + \tau)D(\alpha)$, no action is taken. If $A(\alpha) < (1 - \tau)D(\alpha)$ or $A(\alpha) > (1 + \tau)D(\alpha)$,¹ the coordinator calculates $r = r_p \times D(\alpha)/A(\alpha)$. If $r \leq D(\alpha)$, it sends a $\text{RATE}(\alpha, r)$ message; otherwise, it takes no action.²

In order to prevent attackers from injecting forged RATE messages, all edge routers must block any external packets whose destination is the multicast address responsible for the communication between the

¹Traffic fluctuation may send $A(\alpha)$ above $D(\alpha)$ again.

²Suppose all attack traffic withdraws in the middle of Phase two. If the total arrival rate of legitimate traffic is smaller than $(1 - \tau)D(\alpha)$, $A(\alpha)$ will be always smaller than $(1 - \tau)D(\alpha)$ and the coordinator will keep sending RATE messages with increasing r values. Hence, the coordinator should stop once r reaches $D(\alpha)$.

edge routers.

C. Removal of Rate-Limit Filters

Each rate-limit filter has an expiration time, after which the filter should be removed. The lifetime of a rate-limit filter must be sufficiently long to cover the duration of a typical DoS attack; however, it should not be too long, causing unnecessary overhead to the edge router even after the attack stops. A reasonable default lifetime can be one day. An edge router may keep a log of recently expired filters. If an attack aggregate appears repetitively in the log, the lifetime of a newly created filter for the aggregate should be enlengthed, proportional to the number of expired filters for the same aggregate. On the other hand, if a rate-limit filter does not cause any packet to be dropped after its creation, it can be removed sooner (e.g., after 10 minutes). Hence, filters can be removed quickly from the edge routers that do not receive any attack traffic, while those filters that effectively block the attack traffic are left in the system. An exception is that a filter should be kept by an edge router if it is repeatedly reinstalled after removal.

In an unsynchronized attack, a malicious host (with delayed action) may participate in the attack after DPM completes and the rate-limit filter at its edge router has been removed (e.g., after 10 minutes of no packet match). In this case, the victim will be flooded and subsequently trigger DPM again, which starts Phase one from the base rate that was left off previously. The new RATE messages will be received by all edge routers and their rate limits will be adjusted appropriately according to the protocol of DPM.

D. Analysis

We study two important performance metrics: *convergence time* and *survival ratio*. The former answers the questions of whether the system will stablize and how long it takes to block out the attack traffic. The latter answers the questions of how well the system controls collateral damage and what percentage of legitimate traffic will survive the rate limiting after the system converges. We only consider the survival ratio among customer networks that do not send attack traffic. Proofs for all theorems in this and next sections can be found in Appendix A.

Theorem 1: If the arrival rates of an aggregate α remain steady at all edge routers and the total arrival

rate is larger than $D(\alpha)$, then it takes DPM at most $(\lfloor \log_2 \frac{C \times n}{1+\tau} \rfloor + 1 + \lfloor -\log_{1-\tau} 2 \rfloor)T$ time to stabilize the exit rate such that $(1 - \tau)D(\alpha) \leq A(\alpha) \leq (1 + \tau)D(\alpha)$, where C is the maximum coefficient for α at any edge router and n is the number of edge routers.

Example 1: Suppose $n = 1000$, $C = 100$, $\tau = 5\%$, and $T = 30$ seconds. The maximum convergence time is $(\lfloor \log_2 \frac{C \times n}{1+\tau} \rfloor + 1 + \lfloor -\log_{1-\tau} 2 \rfloor)T = (17 + 13)T = 30T = 15$ minutes. It takes no more than $17T = 8.5$ minutes to reduce the exit rate such that $A(\alpha) \leq (1 + \tau)D(\alpha)$. Then it takes no more than $13T = 6.5$ minutes to improve the exit rate to within $(1 \pm \tau)D(\alpha)$. The above is the worst-case time. In our simulation, the average convergence time for this configuration is less than 4 minutes.

Theorem 2: The survival ratio of legitimate traffic after DPM converges is no less than

$$\frac{\sum_{e \in E - E_k} \min\{r(e), \frac{1-\tau}{\sum_{e' \in E} c(e')} c(e) D(\alpha)\}}{\sum_{e \in E - E_k} r(e)}$$

where E is the set of all edge routers, E_k is the set of edge routers that receive attack traffic, $r(e)$ is the arrival rate of legitimate traffic at e , and $c(e)$ is the coefficient at e , $\forall e \in E$.

The above theorem gives a lower bound on the survival ratio. Our simulation shows that the actual survival ratio is close to one even when 40% of customer networks carry attack traffic.

Corollary 1: Suppose the total arrival rate of legitimate traffic is no more than $D(\alpha)$ and $c(e) \propto r(e)$, $\forall e \in E$. The survival ratio of legitimate traffic after DPM converges is no less than $1 - \tau$.

The above corollary can be easily derived from Theorem 2, given the assumptions that $\sum_{e \in E} r(e) \leq D(\alpha)$ and $c(e) \propto r(e)$, $\forall e \in E$. It means that, if $c(e)$ is configured to be proportional to $r(e)$, then the worst-case success ratio only depends on τ , which is a configurable system parameter.

We have analyzed the worst-case performance with respect to the system parameters. In particular, the worst-case survival ratio is in proportion to $1 - \tau$ (or even less sensitive to τ due to the min operation). A smaller τ means slower convergence time and higher survival ratio. An adaptive mechanism may be employed to make tradeoff between convergence time and survival ratio. Suppose an ISP wants the average convergence time to be bounded by a target value. The ISP keeps track of the convergence times for the past attacks. If the average convergence time is larger than the target, τ is gradually increased. If the

average convergence time is smaller than the target, τ is gradually reduced such that the best survival ratio can be achieved, given the constraint for the convergence time.

IV. DEFENSE PERIMETER BASED ON IP TRACEBACK (DPIT)

A disadvantage of DPM is that, even when there is only one flooding source, the rate-limit filters are temporarily placed on all edge routers though most are removed after a short period of time since they do not cause any packet to be dropped. In this section, we propose an alternative approach, called DPIT, which generates less rate-limit filters at the cost of additional overhead at the coordinator. DPIT finds the flooding sources by IP traceback.

A. *Perimeter-Based IP Traceback Problem*

The problem is for the coordinator to determine where each received packet enters the ISP perimeter. This allows the coordinator to estimate the acceptance rate of each edge router based on the received packets. The coordinator then informs the edge routers with excessive acceptance rates to perform rate-limit filtering. To do this, it must also determine the IP addresses of the edge routers.

Unlike traditional IP traceback schemes that are deployed across the Internet, the design philosophy of perimeter-based defense requires that the deployment is restricted to the perimeter, i.e., on the edge routers only. It does not require the external Internet routers, the internal core routers, or the victims to implement the traceback functions.

B. *Background*

There exist numerous IP traceback proposals [7], [8], [9], [10], [11], [12], [13], [16]. Some (e.g., Pi [16]) are not suitable for our problem because they do not determine the router addresses; the traceback marks carry only address hashes, from which addresses cannot be recovered. Some others can solve our problem, but not optimal because they are designed to construct an attack tree, which is a harder problem than finding the attack entry points along a perimeter. These traceback schemes incur considerable computation overhead [10], storage overhead [12], or communication overhead [17] to keep track of the routers that each

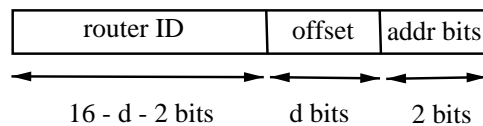


Fig. 2. 16-bit traceback mark of Solution 2

packet traverses. Take [10] as an example. The router addresses are encoded in the 16-bit IP identification field. Each packet can only carry a fragment of an address. The receiver has to piece up the fragments correctly to identify individual routers and form an attack tree. To accomplish this, hash bits are also carried in the IP identification field, and exhaustive combinations among the fragments are performed to identify those combinations with matching address bits and hash bits. In its original design, the computation is supposed to be done by an *end system*, which can be a designated machine. In addition, it can focus on a narrow band of traffic, and there is no real-time requirement. In our problem setup, however, such computation burden can be destructive because it is done by a router, specifically, the coordinator.

In the following, we propose three perimeter-based IP traceback solutions. The first two are simpler but have certain requirements on the system. The third one removes those requirements.

C. Solution 1

The simplest solution is to assign a unique ID to each edge router. An edge router replaces the IP identification field with its ID before accepting a packet into the ISP. This approach requires each edge router to keep a mapping database between IDs and IP addresses. Note that there is not just a single coordinator; any edge router can be a defense *coordinator* when its customer network is under attack. An ISP may have tens of thousands of edge routers. Whenever a new edge router is added or the address of an existing one is changed, the mapping databases at all edge routers must be updated, which is undesirable from the management point of view, considering that an ISP may have many separate management teams, each for a different geographical region or business domain. To solve this problem, we introduce two fully-distributed multi-field solutions that do not maintain the mapping database at any edge router.

D. Solution 2

The structure of the 16-bit traceback mark is shown in Fig. 2. Suppose the ISP possesses an address space of 2^x , e.g., $x = 24$ in case of a Class A network. Since $(32 - x)$ address bits are fixed, only x bits need to be encoded. These bits are partitioned into 2-bit fragments. Each packet carries one fragment, which is selected randomly with equal probabilities. There need $d = \lceil \log_2 \frac{x}{2} \rceil$ offset bits to identify the location of the fragment in the address. The rest $(16 - d - 2)$ bits store the router ID. When the coordinator receives the marked packets, it can group the address fragments based on router IDs, without performing exhaustive combinations.

A problem is that the number of router-ID bits places an upper bound (2^{16-d-2}) on the number of edge routers. A simple solution is to loosely synchronize the system clocks of all edge routers. Let Δ be the maximum clock skew. The edge routers are placed into groups; the group ID and the router ID together uniquely identify a router. Let g be the number of groups and P be a time period larger than 2Δ . Starting from time $t = 0$, during each period $t \in [iP, (i + 1)P), \forall i \in \mathbb{Z}$, only the routers in the $(i \bmod g)$ th group perform traceback marking, while the other routers simply set the router-ID bits to be all zeros, a reserved ID that should not be assigned to any router. The coordinator examines the traceback marks with non-zero router IDs during $iP + \Delta < t < (i + 1)P - \Delta$ for the $(i \bmod g)$ th group.

E. Solution 3

Solution 2 has two requirements: unique router-ID assignment and loosely-synchronized system clocks. Our third solution removes these requirements. The construction of the traceback mark is shown in Fig. 3. The hash field carries the hash value of a router address, e.g., by selecting $(16 - d - 3)$ bits from MD5(address). From the received traceback marks, the coordinator maintains a table that maps each hash value to a set of address fragments and their offsets. Given a hash value, if there is exactly one fragment for each offset, then there is no address conflict; if there are more than one fragment for an offset, then two or more router addresses have the same hash value. These conflicting addresses may share some identical fragments but differ in the other fragments (called *conflicting fragments*), which we must separate

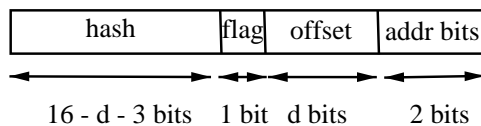


Fig. 3. 16-bit traceback mark of Solution 3

in order to construct the correct router addresses. The fragments from the same address (router) have the same arrival frequency at the coordinator. We have the following two cases.

Case 1: If the packet rates from the conflicting addresses (routers) are different, we can easily separate the conflicting fragments into different groups based on their arrival frequencies, where each group corresponds to one router address.

Case 2: If the packet rates from the conflicting addresses (routers) are too close to tell the difference, the conflict is resolved by the flag bit. Each edge router sets the flag bit of the traceback mark with certain probability, which is randomly chosen but may change periodically. While the fragments from the same address have the same *flagged* arrival frequency at the coordinator, those from different addresses most likely have different *flagged* frequencies, which allows the coordinator to set them apart.

There may exist rare cases where even the flagged frequencies cannot resolve the hash conflict. Now because the coordinator cannot identify the individual conflicting addresses, it is not able to inform those routers to install rate limits when they carry attack traffic. If this happens, the coordinator has to act on behalf of those routers by filtering packets that carry the unresolved hash values in their traceback marks.

F. Identifying and Blocking DDoS Attack Sources

The coordinator starts the defense by inspecting the traceback marks of all α packets. Using the perimeter-based IP traceback, it can determine the addresses of the edge routers from which it receives α packets and estimates the data rate from each router. The coordinator creates a table, called *DPIT table*, which stores the estimated acceptance rates of α at the edge routers. Each table entry consists of three elements: the IP address e of an edge router, the estimated acceptance rate r_e , and the estimated coefficient c_e (initialized to be one).

At the end of each time interval T , if $A(\alpha) \leq (1 + \tau)D(\alpha)$, no action is taken by the coordinator. If

$A(\alpha) > (1 + \tau)D(\alpha)$, the coordinator calculates a base rate r such that

$$(1 - \tau)D(\alpha) \leq \sum_{e \in E} \min\{c_e r, r_e\} \leq D(\alpha) \quad (1)$$

where E is the set of edge routers in the table. This can be done by a binary-search-like algorithm.

CalcBaseRate(E)

(1) $l \leftarrow 0$; $h \leftarrow D(\alpha)$; $r \leftarrow (l + h)/2$

(4) **while** ($\sum_{e \in E} \min\{c_e r, r_e\} < (1 - \tau)D(\alpha)$

(5) **or** $\sum_{e \in E} \min\{c_e r, r_e\} > D(\alpha)$)

(6) **if** ($\sum_{e \in E} \min\{c_e r, r_e\} < (1 - \tau)D(\alpha)$)

(7) $l \leftarrow r$

(8) **else**

(9) $h \leftarrow r$

(10) $r \leftarrow (l + h)/2$

(11) **return** r

The basic idea of DPIT is that, if every edge router e sets its rate limit to be $c_e r$, then the new acceptance rate at e will become $\min\{c_e r, r_e\}$. Assume that packet loss due to network congestion inside the ISP network is insignificant. $A(\alpha)$ is equal to the summation of the new acceptance rates of all edge routers. By Eq. (1), $(1 - \tau)D(\alpha) \leq A(\alpha) \leq D(\alpha)$.

One observation is that, if $r_e \leq c_e r$, the installation of a rate limit $c_e r$ at e is not necessary because the rate of α coming out of e is already under the limit. To install those necessary rate-limit filters, the coordinator sends a unicast RATE(α, r, c_e) message to every $e \in E$ with $r_e > c_e r$. Note that the RATE message in DPIT is different from that in DPM: it has an extra field to carry c_e and it is a unicast message. When e receives the message, if it has already a filter for α , it changes the rate limit to $c r$, where c is the coefficient of α ; otherwise, it creates a new filter for α with rate limit $c r$. If c is different from c_e (received from the RATE message), e sends a COEF(c) message back to the coordinator, which updates the table and changes the value of c_e to be c .

Because the initial value of c_e in the DPIT table is always one and may be smaller than the actual coefficient, some edge routers may set the rate limits higher (cr instead of $c_e r$), which causes more traffic to be accepted. For every subsequent time interval T , the coordinator repeats the calculation of r and sends out new RATE messages, until $A(\alpha)$ falls between $(1 \pm \tau)D(\alpha)$.

The removal of the rate-limit filters is done similarly as described in Section III-C.

G. Analysis

Theorem 3: If the arrival rates of an aggregate α remain steady at all edge routers and the total arrival rate is larger than $D(\alpha)$, then it takes DPIT at most $(\lfloor 1/\tau \rfloor + 2)T$ time to stabilize the exit rate such that $(1 - \tau)D(\alpha) \leq A(\alpha) \leq (1 + \tau)D(\alpha)$.

Example 2: For $\tau = 5\%$, the worst-case time complexity is $22T$, independent of the ISP size.

Theorem 4: The survival ratio of legitimate traffic after DPIT converges is no less than

$$\frac{\sum_{e \in E - E_k} \min\{r(e), \frac{1-\tau}{\sum_{e' \in E} c(e')} c(e) D(\alpha)\}}{\sum_{e \in E - E_k} r(e)}$$

where E is the set of all edge routers, E_k is the set of edge routers that receive attack traffic, $r(e)$ is the arrival rate of legitimate traffic at e , and $c(e)$ is the coefficient at e , $\forall e \in E$.

Corollary 2: Suppose the total arrival rate of legitimate traffic is no more than $D(\alpha)$ and $c(e) \propto r(e)$, $e \in E$. The survival ratio of legitimate traffic after DPIT converges is no less than $1 - \tau$.

V. COOPERATION AMONG NEIGHBORING ISPs

Two neighboring ISPs, if both implement DPM or DPIT, can cooperate to narrow the rate-limit filtering towards the actual attackers. Fig. 4 illustrates the idea. Consider an attack aggregate α . In ISP2, Link 6 is the exit link of α . The traffic then passes ISP1 and takes Link 1 as the exit link to reach the destination. One attacker is shown in the figure, which generates a large volume of α and causes a DoS attack at a victim host behind Link 1. Legitimate traffic of α enters ISP1 and ISP2 from other edge routers with much smaller arrival rates.

Under the attack, the victim signals router A to initiate the defense. With either DPM or DPIT, a rate-

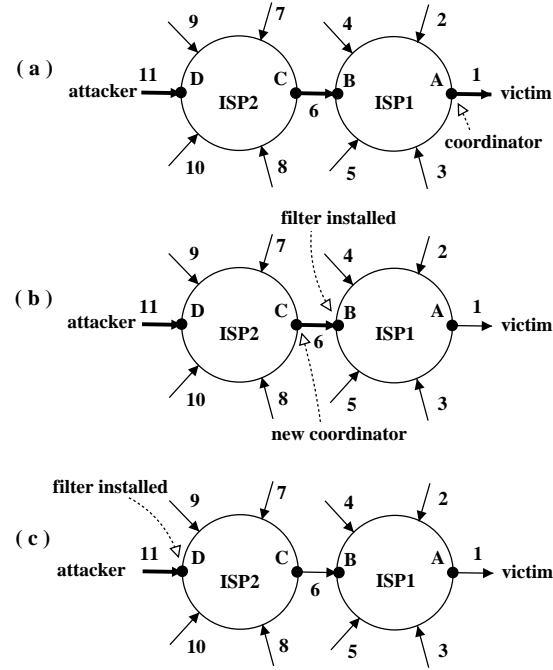


Fig. 4. cooperated defense between neighboring ISPs

limit filter will be installed at router B, which blocks most of the attack traffic. Consequently, the exit rate on Link 1 drops to the normal level, and packets of α from Links 2, 3, 4, and 5 can reach the victim host. However, packets from Links 7, 8, 9, and 10 are still mixed with the attack traffic and thus rate-limited together with the attack traffic. Router B treats itself as under a DoS attack, and the rate limit specified in the filter is the desirable rate of α from ISP2. It signals router C to initiate the defense in ISP2. A rate-limit filter is consequently stalled at router D, which reduces the exit rate on Link 6 to the desired rate and allows the packets from Links 7, 8, 9, and 10 to pass into ISP1 without being blocked. The filter at router B is then removed if it does not drop packets. The defense mechanisms implemented in ISP1 and ISP2 do not have to be the same.

A nice property of our perimeter-based defense is that *good behavior* is rewarded; ISPs that deploy the system can benefit from the existing deployment in the peer ISPs. A different rewarding property exists within an ISP. As we will see in the next section, the survival ratios for customer networks that do not generate attack traffic are close to one, while those that harbor malicious hosts may suffer. It rewards the customers that protect their hosts from being compromised.

We want to stress that the cooperation between ISPs is voluntary. Although the cooperation brings

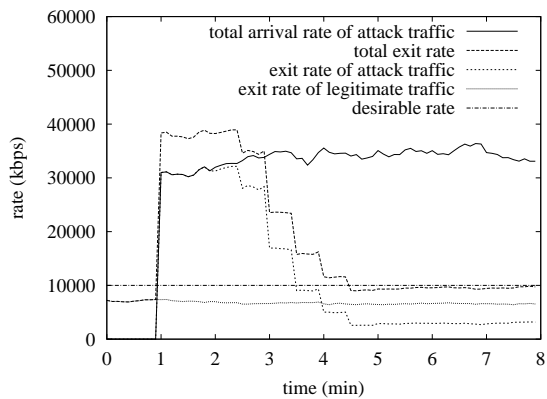


Fig. 5. DPM

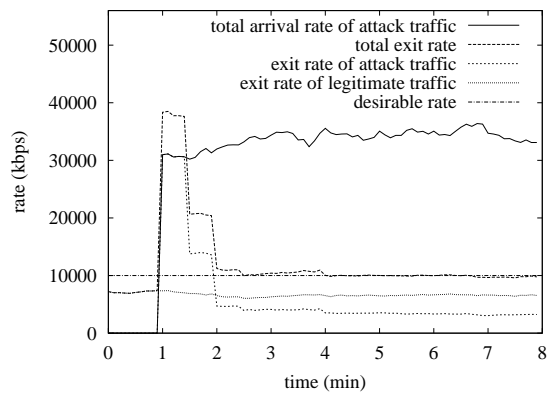


Fig. 6. DPIT

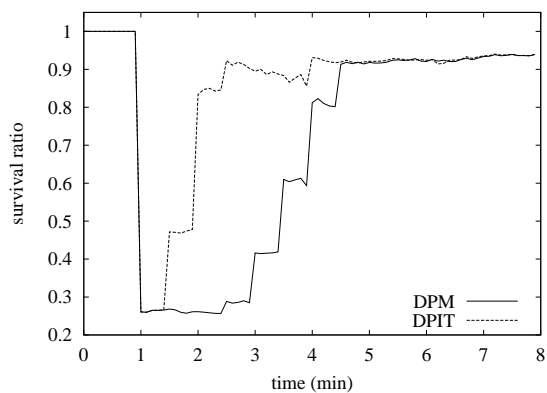


Fig. 7. Survival ratio at the server. Consider legitimate traffic from all customer networks.

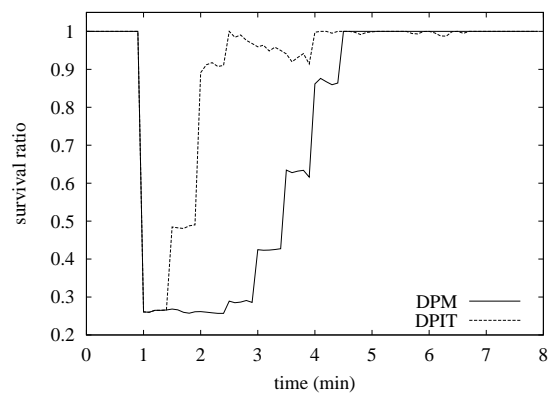


Fig. 8. Survival ratio at the server. Only consider legitimate traffic from non-attacking customer networks, i.e., those networks that do not generate attack traffic.

significant value, the defense operations within one ISP does not need the support from other ISPs. In the above example, suppose ISP2 does not implement DPM/DPIT. The attack traffic will still be blocked at B . The customers within the defense coverage, i.e., those behind Links 2-4, can still access the victim host, but the customers outside of the defense coverage, i.e., those behind Link 7-10, can no longer access the victim host. The anti-DDoS service is only offered to the customers within the defense coverage.

VI. SIMULATION

We use simulations to evaluate DPM and DPIT in terms of average convergence time and average survival ratio. The default simulation parameters are given as follows. The ISP has 1000 edge routers connecting to its customer networks. Consider a server under a DDoS attack. The desirable rate to the server

is 10,000 kbps (kilobits per second). The policy coefficients for the traffic at the edge routers follow an exponential distribution with the mean equal to five. Initially, the distribution of legitimate traffic follows the distribution of the coefficients. The average arrival rate per edge router is equal to 7 kbps. The total arrival rate of legitimate traffic is thus 7,000 kbps. The simulation is performed at ticks of 6 seconds each. The arrival rate of legitimate traffic at each edge router fluctuates up to $\pm 50\%$ over each tick. The attack traffic enters the ISP from 100 randomly selected edge routers with an average attack rate of 400 kbps, following an exponential distribution.³ Although a DDoS attack may involve a large number of compromised hosts, they typically reside on a small number of networks (such as the Feb-2000 attack on Yahoo and [18]). The default choice of 10% edge routers carrying attack traffic is not due to the limitation of DPM and DPIT. One of our simulations will scale this percentage up to 95%. The arrival rate of attack traffic at an edge router fluctuates up to $\pm 20\%$ over each tick. The time interval T for periodic update of rate limits is 30 seconds for both DPM and DPIT. $\tau = 5\%$. The default parameters are always assumed unless the figures indicate otherwise.

A. Effectiveness against DDoS

Our first simulation demonstrates that DPM and DPIT respond quickly against DDoS by establishing the appropriate rate limits at the edge routers, which block most attack traffic while allowing most legitimate traffic to pass.

Fig. 5 shows how DPM reacts to a DDoS attack. The five curves describe the total arrival rate of attack traffic, the total exit rate towards the server, the exit rate of attack traffic, the exit rate of legitimate traffic, and the desirable rate. The attack starts at time = 1 min, and the total exit rate shoots up, which floods the server. After DPM is activated, as the rate limits are decreased at each time interval, the total exit rate is reduced. The figure shows that, while more and more attack traffic is filtered, the exit rate of legitimate traffic changes little. Phase one terminates at time = 4.5 min after the total exit rate drops below the desirable rate. Then Phase two gradually improves the total exit rate to approach the desirable rate.

³Suppose the attack is launched from compromised hosts on 100 different customer networks. Some hosts may have high-speed Internet connections while some others may have low-speed links.

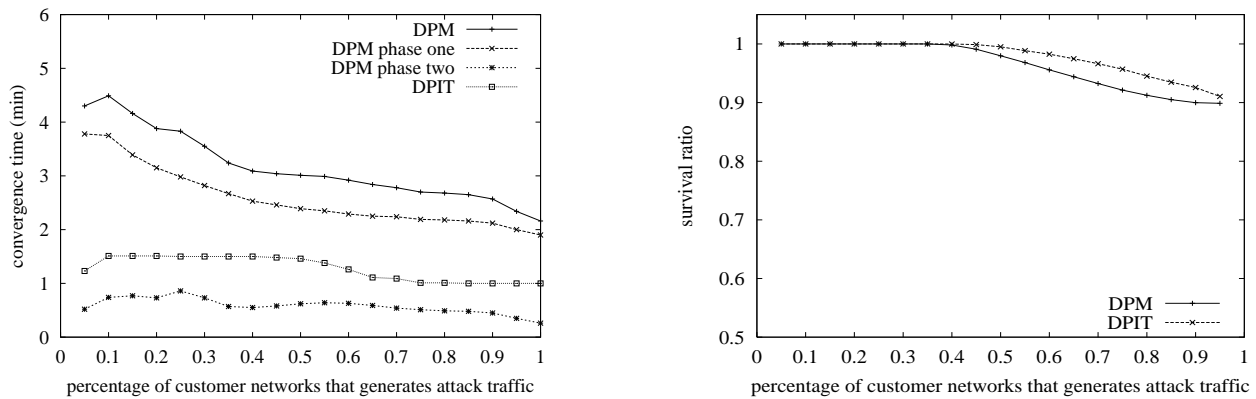


Fig. 9. convergence time and survival ratio with respect to percentage of customer networks that generate attacker traffic

Combining the two phases, the convergence time is 5 minutes (from time = 1 min to time = 6 min). DPM is reactivated at time = 7 min due to traffic fluctuation and lasts for another minute. Fig. 6 shows how DPIT reacts to the same DDoS attack. DPIT reduces the total exit rate more quickly towards the desirable rate. Its convergence time is 1.5 minutes. DPIT is reactivated at time = 4 min due to traffic fluctuation and another rate-limit update is made.

When the exit rate $A(\alpha)$ is above the server's capacity, after the traffic exits the ISP and reaches the server, the server must randomly drop the excess packets and accept the arrival traffic up to its capacity. Fig. 7 shows the survival ratio at the server, which is the percentage of all legitimate traffic that arrives at and is accepted by the server. After the attack starts, the survival ratio drops below 27% as the server drops most legitimate traffic due to overwhelming total arrival rate (total exit rate from the ISP's point of view). It then recovers quickly and stays above 92% after DPM/DPIT converges.

Fig. 8 shows that the survival ratio at the server for *non-attacking networks only* is consistently above 99.99% after convergence; for these customers, the dropped traffic can be easily handled by TCP retransmission.

B. Scaling with the Number of Attacking Customer Networks

Our third simulation demonstrates that, even when 40% of all customer networks send attack traffic, DPM and DPIT can still protect the rest customer networks from being affected.

Fig. 9 presents the average convergence time and survival ratio with respect to the percentage of cus-

TABLE I
 AVERAGE NUMBERS OF RATE AND COEF MESSAGES SENT BY DPIT PER TIME INTERVAL WITH RESPECT TO NUMBER OF EDGE ROUTERS AND NUMBER OF ATTACKING CUSTOMER NETWORKS

edge routers	attack networks	RATEs	COEFs
1000	50	81	54
5000	50	47	22
1000	100	178	113
5000	100	147	91
1000	200	317	177
5000	200	387	255

customer networks that have attackers (or compromised hosts). The left plot shows that both DPM and DPIT converge faster when there are more attackers, due to higher rate of attack traffic and thus faster reduction of rate limits according to the design of DPM/DPIT. The right plot shows that the survival ratio of legitimate traffic (at the server for non-attacking customer networks) is close to 100% unless more than 40% of all customer networks attack. It is consistently true in all our simulations that most blocked legitimate traffic comes from attacking networks.

We also performed simulations with the ISP scaling from 1,000 to 10,000 edge routers. The average convergence time and the average survival ratio are largely insensitive to the size of the ISP.

C. Communication Overhead

The communication overhead of DPM is one multicast message per time interval, and the message reaches every edge router. The average communication overhead of DPIT is shown in Table I. The number of (unicast) messages sent by DPIT per time interval is much less than the number of edge routers. The overhead is not significant considering that the time interval is set to be 30 seconds and DPIT typically converges after 2-3 intervals as shown in the previous figures. While DPIT performs consistently better than DPM in terms of convergence time, its coordinator may have to process hundreds of messages per time interval while DPM's coordinator needs to process only one. When the attack traffic comes from a small number of customer networks, DPIT is a better choice. On the other hand, when the attack traffic comes from a large number of different customer networks, DPM scales better due to less computation/storage overhead at the coordinator.

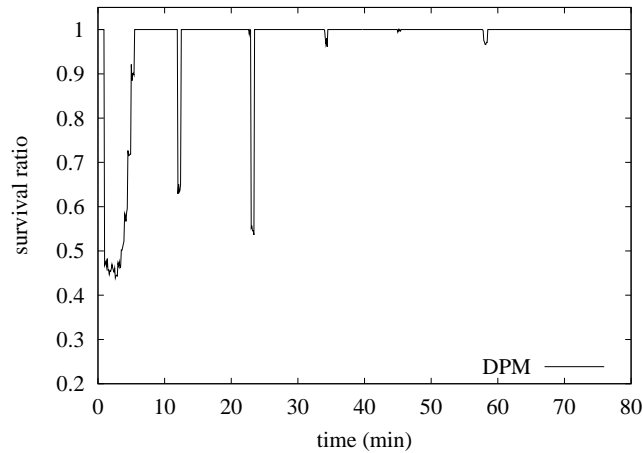


Fig. 10. survival ratio under unsynchronized attack

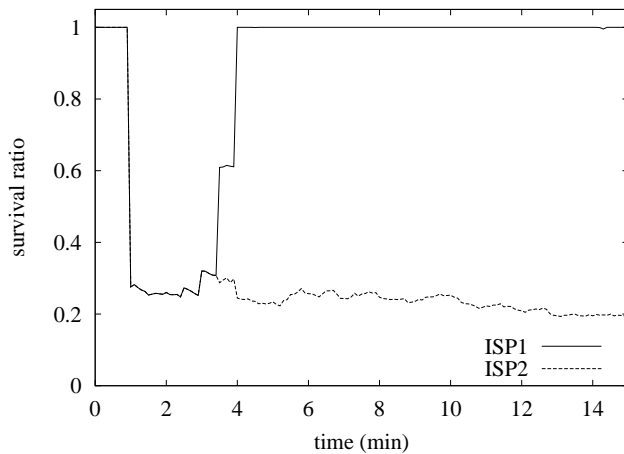


Fig. 11. ISP1 implements DPM; ISP2 does not.

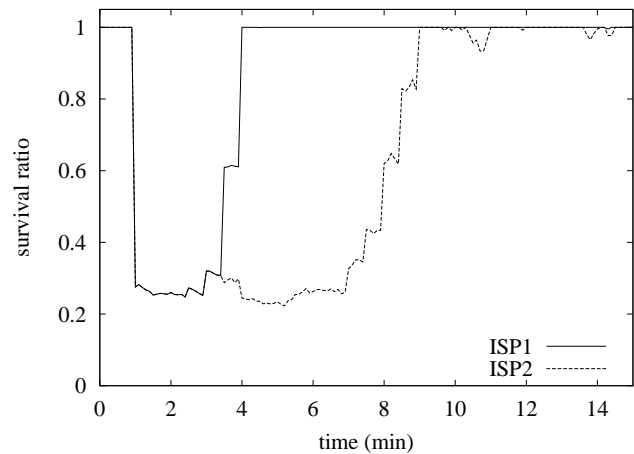


Fig. 12. Both ISP1 and ISP2 implement DPM.

D. Unsynchronized Attacks

Our fifth simulation studies the unsynchronized attacks. In the simulation, 50 customer networks start an attack at time = 1 min, 25 additional networks join the offense after each 11-minute interval at time = 12, 23, 34, 45, and 55, respectively, and 25 more networks join the offense at random times between 1 and 60 min. For DPM, an edge router will remove a rate-limit filter if it does not cause any packet to drop for 10 minutes after its creation, but will keep the filter if it is reinstalled twice after removal (Section III-C).

Fig. 10 shows the survival ratio of legitimate traffic (at the server for non-attacking customer networks) when DPM is implemented. For the initial 50 attacking networks, it takes 4.5 minutes (from time = 1 to 5.5 min) for DPM to converge and achieve close to one survival ratio. For the subsequent 25 attackers at time

= 12 (or 23) min, because the filters have been removed from most edge routers (except for the 50 initial attackers) at time = 11 (or 22) min, the attack traffic is able to enter the ISP and cause the survival ratio to drop, which triggers DPM again to restore the filters at all edge routers and return the survival ratio to about one. The convergence time is very small because the initial base rate of this DPM is the final base rate of the previous DPM (whose initial base rate is $D(\alpha)$, causing much larger convergence time). For the 25 new attackers at time = 34 (or 45, 55) min, because the edge routers keep the filters after two reinstallations at time = 12 and 23 min, most new attack traffic cannot pass their edge routers and their impact on survival ratios is small.

The simulation results for DPIT are omitted. They are comparable to those for DPM.

E. Attacks across ISPs

Our sixth simulation studies attacks across ISPs. Consider the example in Fig. 4. Suppose each ISP has 500 customer networks, of which 50 networks attack. ISP1 implements DPM. We consider two cases: a) ISP2 does not implement the perimeter-base defense (Fig. 11), and b) ISP2 implements DPM (Fig. 12). In both cases, the survival ratio for non-attacking networks in ISP1 is close to one. The survival ratio for non-attacking networks in ISP2 depends on whether ISP2 implements DPM.

VII. LIMITATION

As demonstrated by our analysis and simulations, DPM and DPIT can effectively protect the legitimate traffic from customer networks that do not carry attack traffic. However, they are ineffective for legitimate traffic from customer networks that contain compromised hosts and send attack traffic. The reason is that the legitimate traffic from those networks are indistinguishable from the attack traffic. To solve this problem, the cooperation from the customer side will be necessary, and the perimeter-based defense alone is not sufficient.

VIII. CONCLUSION

The edge routers form a natural boundary between the ISP network and the rest of the Internet. This boundary, called the ISP perimeter, can be turned into a defense barrier against network intrusions. We proposed two perimeter-based defense mechanisms, DPM and DPIT, which mitigate DDoS attacks by blocking the flooding sources while allowing most legitimate traffic to reach the destination. To the best of our knowledge, this is also the first work that studied perimeter-based IP traceback and proposed three solutions. Our analysis and simulations demonstrated that DPM and DPIT selectively block out the attack traffic and quickly converge to the desirable rate. We also discussed how neighboring ISPs can cooperate to improve the performance.

REFERENCES

- [1] C. Schuba, I. Krsul, M. Kuhn, E. Spafford, A. Sundaram, and D. Zamboni, "Analysis of A Denial of Service Attack on TCP," *Proc. of IEEE Symposium on Security and Privacy*, 1997.
- [2] J. Lemon, "Resisting SYN Flood DoS Attacks with A SYN Cache," *Proc. of USENIX BSDCON2002*, February 2002.
- [3] H. Wang, D. Zhang, and K. G. Shin, "SYN-dog: Sniffing SYN Flooding Sources," *Proc. of 22nd International Conference on Distributed Computing Systems (ICDCS'02)*, July 2002.
- [4] D. Moore, G. Voelker, and S. Savage, "Inferring Internet Denial of Service Activity," *Proc. of USENIX Security Symposium'2001*, August 2001.
- [5] P. Ferguson and D. Senie, "Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing," *IETF, RFC 2267*, January 1998.
- [6] K. Park and H. Lee, "On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets," *Proc. of ACM SIGCOMM'2001*, August 2001.
- [7] H. Burch and B. Cheswick, "Tracing Anonymous Packets to Their Approximate Source," *Proc. of USENIX LISA'00*, December 2000.
- [8] D. Schnackenberg, K. Djahandari, and D. Sterne, "Infrastructure for Intrusion Detection and Response," *Proc. of First DARPA Information Survivability Conference and Exposition*, January 2000.
- [9] R. Stone, "CenterTrack: An IP Overlay Network for Tracking DoS Floods," *Proc. of USENIX Security Symposium'00*, August 2000.
- [10] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical Network Support for IP Traceback," *Proc. of ACM SIGCOMM'2000*, August 2000.
- [11] D. Song and A. Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback," *Proc. of IEEE INFOCOM'2001*, March 2001.
- [12] A. C. Snoren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer, "Single-Packet IP Traceback," *IEEE/ACM Transactions on Networking*, December 2002. An earlier version was published in *proc. of ACM SIGCOMM'2001*.

- [13] K. Park and H. Lee, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attacks," *Proc. of IEEE INFOCOM'2001*, March 2001.
- [14] M. Song and J. Xu, "IP Traceback-based Intelligent Packet Filtering: A Novel Technique for Defending Against Internet DDoS Attacks," *10th IEEE International Conference on Network Protocols (ICNP'2002)*, November 2002.
- [15] P. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High Bandwidth Aggregates in the Network," *Computer Communications Review*, vol. 32, no. 3, pp. 62–73, July 2002.
- [16] A. Yaar, A. Perrig, and D. Song, "Pi: A Path Identification Mechanism to Defend against DDoS Attacks," *IEEE Symposium on Security and Privacy*, May 2003.
- [17] S. M. Bellovin, "ICMP Traceback Messages," *Internet Draft: draft-bellovin-itrace-00.txt*, March 2000.
- [18] S. Gibson, "The Strange Tale of the Denial of Service Attacks Against GRC.COM," <http://grc.com/dos/grcdos.htm>, 2002.
- [19] Gerald W. Gordon, "SYN Cookies, An Exploration," http://www.giac.org/practical/Gerald_Gordon_GSEC.doc.
- [20] A. D. Keromytis, V. Misra, and D. Rubenstein, "SOS: Secure Overlay Services," *Proc. of ACM SIGCOMM'2002*, August 2002.
- [21] S. Ratnasamy, S. Shenker, and I. Stoica, "Routing Algorithms for DHTs: Some Open Questions," *Proc. of 1st International Workshop on Peer-to-Peer Systems (IPTPS '02)*, March 2002.

APPENDIX A. PROOFS

Proof of Theorem 1: We first compute the maximum time for Phase one to complete. Initially the coordinator sends a $\text{RATE}(\alpha, D(\alpha))$ message, and the highest rate limit at any edge router is $C \times D(\alpha)$. After that, one RATE message is sent after each time interval T as long as $A(\alpha) > (1 + \tau)D(\alpha)$. Suppose Phase one sends m_1 RATE messages. Each RATE reduces the rate limits at all edge routers at least by half. When the highest rate limit among all edge routers drops below $\frac{(1+\tau)D(\alpha)}{n}$, we have $A(\alpha) \leq \frac{(1+\tau)D(\alpha)}{n}n = (1 + \tau)D(\alpha)$, which will cause Phase one to complete. Therefore, prior to the last RATE, the highest rate limit ($C \times D(\alpha) \times (\frac{1}{2})^{m_1-1}$) should be greater than $\frac{(1+\tau)D(\alpha)}{n}$.

$$C \times D(\alpha) \times \left(\frac{1}{2}\right)^{m_1-1} > \frac{(1 + \tau)D(\alpha)}{n}$$

$$m_1 < \log_2 \frac{C \times n}{1 + \tau} + 1$$

Since m_1 is an integer, we have $m_1 \leq \lfloor \log_2 \frac{C \times n}{1 + \tau} \rfloor + 1$. The maximum time for Phase one to complete is thus $(\lfloor \log_2 \frac{C \times n}{1 + \tau} \rfloor + 1)T$.

Next, we consider Phase two. There are two cases.

Case one: Prior to the last RATE message of Phase one, if $D(\alpha)/A(\alpha) < 1/2$, then $r = r_p \times D(\alpha)/A(\alpha)$. Since the rate limits at all edge routers are reduced by a factor of $D(\alpha)/A(\alpha)$, the exit

rate of α will be reduced *at most* by a factor of $D(\alpha)/A(\alpha)$. Hence, after the last RATE message, we must have $A(\alpha) \geq D(\alpha)$. Since it is the end of Phase one, we also have $A(\alpha) \leq (1 + \tau)D(\alpha)$. Therefore, no message will be sent in Phase two.

Case two: Prior to the last RATE message of Phase one, if $D(\alpha)/A(\alpha) \geq 1/2$, then $r = r_p/2$. Because the RATE message cuts the rate limits at all edge routers by half and $A(\alpha) > (1 + \tau)D(\alpha)$ before the message, we must have $A(\alpha) > (1 + \tau)D(\alpha)/2$ after the message.

The goal of Phase two is to send RATE messages that gradually increase $A(\alpha)$ to within $(1 \pm \tau)D(\alpha)$. Suppose Phase two sends m_2 RATE message. Each RATE message increases the rate limits by a factor of $D(\alpha)/A(\alpha)$. Consequently, the exit rate is increased *at most* by a factor of $D(\alpha)/A(\alpha)$. Hence, $A(\alpha)$ will remain smaller than $D(\alpha)$ after each RATE message. It follows that the total increase in rate limits by all m_2 RATE messages should be smaller than a factor of 2. Otherwise the rate limits would become no less than those prior to the last RATE message of Phase one, which would mean $A(\alpha) > (1 + \tau)D(\alpha)$.

Prior to each RATE message, $A(\alpha) < (1 - \tau)D(\alpha)$; otherwise, the RATE would not be sent. $r = r_p \times D(\alpha)/A(\alpha) > \frac{1}{1-\tau}r_p$. m_2 can be calculated as follows.

$$\left(\frac{1}{1-\tau}\right)^{m_2} < 2$$

$$m_2 < -\log_{1-\tau} 2$$

Since m_2 is an integer, we have $m_2 \leq \lfloor -\log_{1-\tau} 2 \rfloor$. The maximum time for Phase two to complete is thus $\lfloor -\log_{1-\tau} 2 \rfloor \times T$.

Therefore, the total time for DPM to complete is no more than $(\lfloor \log_2 \frac{C \times n}{1+\tau} \rfloor + 1 + \lfloor -\log_{1-\tau} 2 \rfloor)T$. If the total arrival rate is greater than $D(\alpha)$, DPM completes only when $(1 - \tau)D(\alpha) \leq A(\alpha) \leq (1 + \tau)D(\alpha)$.

□

Proof of Theorem 2: Let r be the last base rate that the coordinator sends before DPM converges. The rate limit at $e \in E$ is $c(e)r$. After DPM converges, if $A(\alpha) < (1 - \tau)D(\alpha)$, $r \geq D(\alpha)$ because otherwise

DPM would not have stopped. In this case, the survival ratio is 100%. If $A(\alpha) \geq (1 - \tau)D(\alpha)$, we have

$$\begin{aligned} \sum_{e \in E} c(e)r &\geq A(\alpha) \geq (1 - \tau)D(\alpha) \\ r &\geq \frac{1 - \tau}{\sum_{e \in E} c(e)} D(\alpha) \end{aligned}$$

The acceptance rate of legitimate traffic at e , $\forall e \in E - E_k$, is

$$\min\{r(e), c(e)r\} \geq \min\left\{r(e), \frac{1 - \tau}{\sum_{e' \in E} c(e')} c(e)D(\alpha)\right\}$$

Therefore, the survival ratio is no less than

$$\frac{\sum_{e \in E - E_k} \min\left\{r(e), \frac{1 - \tau}{\sum_{e' \in E} c(e')} c(e)D(\alpha)\right\}}{\sum_{e \in E - E_k} r(e)}$$

□

Proof of Theorem 3: Suppose the defense starts at time 0. Let E be the set of edge routers whose arrival rates of α is not zero. At time 0, $\forall e \in E$, the arrival rate equals to the acceptance rate, denoted as $r_{e,0}$. Suppose the exit rate stabilizes after k time intervals with $(1 - \tau)D(\alpha) \leq A(\alpha) \leq (1 + \tau)D(\alpha)$. We determine the worst-case value of k in the following.

RATE messages are sent out at time iT , $1 \leq i \leq k$. Immediately before time iT , the estimated coefficient and acceptance rate for $e \in E$ in the DPIT table is denoted as $c_{e,i-1}$ and $r_{e,i-1}$, respectively. $c_{e,i}$, $0 \leq i < k$, takes one of two values: one or the correct coefficient for α at e . The value is one until the first RATE message is sent to e and a COEF message is subsequently received from e . From then on, it becomes the correct coefficient.

At time T , a base rate r_1 is calculated such that

$$(1 - \tau)D(\alpha) \leq \sum_{e \in E} \min\{c_{e,0} \times r_1, r_{e,0}\} \leq D(\alpha) \quad (2)$$

Let $E_1 = \{e \mid r_{e,0} > c_{e,0} \times r_1, e \in E\}$. RATE messages are sent to E_1 . By Eq. (2), we have

$$\sum_{e \in E - E_1} r_{e,0} \leq D(\alpha) \quad (3)$$

At time iT , $1 < i < k$, a base rate r_i is calculated such that

$$(1 - \tau)D(\alpha) \leq \sum_{e \in E} \min\{c_{e,i-1} \times r_i, r_{e,i-1}\} \leq D(\alpha) \quad (4)$$

Let E_i be the subset of E such that $\forall e \in E_i$, (1) $r_{e,i-1} > c_{e,i-1} \times r_i$ and (2) no RATE message has been sent to e previously. $\forall e \in E_i$, $c_{e,i-1} = 1$. After RATE messages are sent, the new exit rate is noted as $A_i(\alpha)$. We must have

$$A_i(\alpha) \leq \sum_{e \in E-E_i} \min\{c_{e,i-1} \times r_i, r_{e,i-1}\} + \sum_{e \in E_i} r_{e,0}$$

because only the edge routers in E_i may set rate limits higher than $c_{e,i-1} \times r_i$ and possibly allow all arrival traffic ($r_{e,0}$) to be accepted. Since $i < k$, we also have $A_i(\alpha) > (1 + \tau)D(\alpha)$; otherwise $A_i(\alpha)$ would stabilize right away before k time intervals. Hence,

$$\begin{aligned} \sum_{e \in E-E_i} \min\{c_{e,i-1} \times r_i, r_{e,i-1}\} + \sum_{e \in E_i} r_{e,0} &> (1 + \tau)D(\alpha) \\ D(\alpha) + \sum_{e \in E_i} r_{e,0} &> (1 + \tau)D(\alpha) \\ \sum_{e \in E_i} r_{e,0} &> \tau D(\alpha) \end{aligned}$$

Since the above is true for all $1 < i < k$, we have

$$\sum_{i=2}^{k-1} \left(\sum_{e \in E_i} r_{e,0} \right) > (k-2)\tau D(\alpha)$$

Because $E_i \subseteq E$ and $E_i \cap E_j = \emptyset$, $\forall i, j$, $1 \leq i < k$, $1 \leq j < k$, $i \neq j$, we have

$$\sum_{e \in E-E_1} r_{e,0} > (k-2)\tau D(\alpha)$$

By Eq. (3), we have

$$\begin{aligned} D(\alpha) &> (k-2)\tau D(\alpha) \\ k &< \frac{1}{\tau} + 2 \end{aligned}$$

Because k is an integer, we have $k \leq \lfloor 1/\tau \rfloor + 2$. Hence, the worst-case convergence time is $(\lfloor 1/\tau \rfloor + 2)T$.

□

The proof of Theorem 4 is identical to that of Theorem 2.