Engaging Edge Networks in Preventing and Mitigating Undesirable Network Traffic

Lan Wang, Qishi Wu, Dung Dinh Luong
Department of Computer Science
University of Memphis
{lanwang, qishiwu, dluong}@memphis.edu

Abstract

We address the security vulnerabilities in existing protocols for network traffic authentication, by engaging edge networks in defending against undesirable traffic. In the proposed PATRICIA architecture, edge networks cooperate to prevent misbehaving sources from ***flooding traffic in both control and data channels****. Moreover, edge networks employ an endorsement procedure to approve data communications among local and remote hosts, hence breaking down* ***potential collusion between those hosts****. The protection mechanism is only activated for hosts under attack, therefore minimizing the control traffic bandwidth and processing overhead. Our performance evaluation demonstrates the effectiveness of PATRICIA.*

1 Introduction

The current Internet has few effective mechanisms to prevent a user from flooding network traffic, whether the user is launching a denial-of-service (DoS) attack or unintentionally misusing network resources. A colossal amount of such traffic can exhaust the network bandwidth and server resources available to well-behaved users, thus posing a significant threat to the Internet's viability (see IAB workshop report [2]). For example, a 2002 DoS attack [8] brought down 9 of the 13 DNS root servers, which were again targeted by a major DoS attack in early 2007 [6].

An effective approach to mitigating undesirable traffic should throttle the traffic at or near its *source(s)*, which means that routers on the traffic path, especially those close to the sources, need a mechanism to identify the undesirable traffic. Filtering-based mechanisms (e.g. [3]) focus on deploying per-flow filters in routers, which may not scale well during a highly distributed attack. Recent work on *capability* (e.g. [1, 11, 12]) attempts to address the traffic identification problem in a scalable manner. More specifically, a source needs to obtain a capability from a destination *before* the actual data transmissions can begin. The destination may either accept or reject the source's request, assuming that it has some means to identify undesirable sources. As a result, packets sent by those undesirable sources will not carry valid capabilities and thus will be dropped by routers. The verification of capabilities typically involves simple hashing and little state information.

As pointed out by Argyraki and Cheriton [4], however, capability-based mechanisms are subject to *denial-of-capability attacks*, i.e., attackers can flood request messages to exhaust the bandwidth allocated for control traffic, hence preventing legitimate senders from obtaining their capabilities. Another problem is *collusion*, in which hosts controlled by the same attacker issue each other capabilities so that they can flood data traffic to congest certain links. In fact, as we will explain in Section 2, flooding of control traffic and collusion among malicious hosts can also occur in filtering-based mechanisms.

We observe that, *as long as trust is placed entirely in end hosts, the above problems will exist in any scheme designed to mitigate undesirable traffic*. The trust allows any host to request or issue a capability in capability-based mechanisms and install traffic filters in filtering-based mechanisms. There is no other entity to control hosts that exploit the trust to deny other hosts' access to service. The question is what other entities should be involved in controlling those hosts. In our view, ***edge networks*** **are in the best position to control hosts who abuse the trust because these networks own the resources shared by many end hosts**. Our work therefore aims to provide the necessary *incentives*, *information*, and *mechanisms* for edge networks to actively participate in the defense against undesirable traffic. To this end, we propose an architecture named *PATRICIA*, Passive and Active Traffic Regulation Infrastructure using Covered IP Addresses, with the following three major features.

First, PATRICIA provides a cooperative mechanism for edge networks to share information about suspected misbehaving senders so that these senders' traffic can be blocked

at their source networks. More specifically, once a destination node detects a misbehaving source, the destination's network will notify the source's network, which thereafter will prevent the source from flooding any *control* or *data* traffic to the destination.

Second, PATRICIA employs an *edge-network endorsement* procedure. A source node must obtain (a) an endorsement from its own network and (b) an authorization from the destination, before it can send out any data traffic. Likewise, the destination must obtain an endorsement from its own network in order to authorize the source's data traffic. The endorsement procedure allows an edge network to defeat potential collusion between a local host and a large number of remote sources; it also limits address spoofing and flooding of control traffic.

Third, since attacks on regular hosts do not occur most of the time, we switch a host to a mode that requires active regulation of its incoming traffic only when necessary. Therefore, the majority of Internet traffic should **not** require active traffic regulation, resulting in less processing and control bandwidth overhead along the end-to-end path.

The rest of the paper is organized as follows. After discussing related work in Section 2, we describe the PATRICIA design in Section 3 and discuss deployment issues in Section 4. Simulation results are presented in Section 5. We conclude our work in Section 6.

2 Background and Related Work

Our work is partly inspired by two existing approaches that stop undesirable traffic: *packet filtering* and *capability*. Both approaches assume that the destination has some ability to identify bad senders, which may involve observing the source's application-layer behavior and/or network-layer behavior. A number of previous papers, e.g. [12], have discussed how this identification may be done.

A filtering-based mechanism, e.g. AITF [3], allows receivers to notify the network of their intent to stop certain traffic flows by installing flow-level packet filters in routers. Such mechanisms may not scale well when dealing with large-scale distributed DoS attacks, as routers may need to maintain a large number of filters and complex filtering could significantly slow down packet processing.

To address scalability in stopping undesirable traffic, Anderson *et. al.* proposed the concept of *capability* [1], which was refined in SIFF [11] and TVA [12]. A capability is a form of authorization granted by a receiver to a sender; only authorized traffic will be delivered by routers. SIFF is lightweight in that routers do not need to maintain per-flow information and only need to hash some packet header information with a local secret to verify a capability. TVA provides greater DoS protection at the cost of increased state at each hop.

Two major problems remain for capability-based mechanisms: DoS attacks against legitimate control traffic and collusion among malicious hosts as discussed in [4] and [12], respectively. *Note that these problems also exist in the filtering approach, albeit in different forms*: (a) a malicious host may flood filter requests to exhaust the memory in routers, hence denying other hosts' legitimate requests; (b) it may collude with one or more senders by not sending filters to stop them even if they are flooding traffic.

To counter the above problems, our work engages not only end hosts, but also edge networks in issuing authorizations. Moreover, each edge network maintains a blocking list (similar to a filter) of suspected misbehaving senders reported by other edge networks. To some extent, our work employs both capability and filtering while addressing their inherent problems. Another major difference between our design and previous work is the dynamic switching between passive and active traffic regulation, which reduces the overall processing and bandwidth overhead.

3 PATRICIA Design

We first present an overview of PATRICIA focusing on our design rationale. We then describe the PATRICIA control protocols, TRAP and TCEP, as well as the packet processing rules at border routers.

3.1 Design Overview

In PATRICIA, each potential source is required to obtain an *authorization* (in the form of a capability) from the destination, as well as *endorsements* from both the source's and the destination's networks, at the beginning of their session. This process is called "*active traffic regulation*" as opposed to "passive traffic regulation" in a normal situation, which does not require authorizations and endorsements. There are one or more *traffic regulators* in each edge network, whose main responsibility is to *endorse legitimate PATRICIA control messages used to obtain authorizations*. The traffic regulators use local policies and a blocking list (i.e., a list of misbehaving local hosts identified by other networks) to make endorsement decisions. If a PATRICIA control message does not contain a valid endorsement, it cannot be used to obtain authorization from the destination as routers will drop the message. Therefore, the endorsement procedure empowers edge networks to break up collusion among nodes. It also prevents a source from flooding control messages, since traffic regulators can easily impose a rate limit on the endorsement of such messages. Details can be found in the *Traffic Channel Establishment Protocol (TCEP)* described in Section 3.3.

Since the majority of Internet hosts are not under severe attacks most of the time, active traffic regulation is enabled

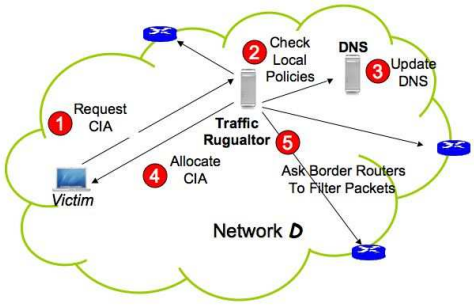

Figure 1. Control Messages in TRAP

only for nodes that are under attack or are likely to be attacked. This design consideration distinguishes PATRICIA from prior work by avoiding unnecessary delay and overhead associated with active traffic regulation. Please refer to the *Traffic Regulation Activation Protocol (TRAP)* in Section 3.2 for details.

An endorsement is in the form of a *public-key signature* on a PATRICIA control message. Therefore, only traffic regulators with the correct private key can produce a valid endorsement. We assume that all traffic regulators in an edge network share the same public/private key pair or use a group public-key scheme [5], with the shared public key signed by a globally known authority. Because each Autonomous System (AS) has only one public key and the number of ASes in the Internet is on the order of 10^4 (this number grows slowly), an edge network may simply download the entire key database from the authority and distribute them to all its border routers. *We are also investigating the feasibility of distributing the keys through BGP*. As to the overhead of public-key operations, we would like to emphasize that since active traffic regulation is turned *off* by default for regular hosts, PATRICIA control messages are expected to be a small portion of the overall traffic. Moreover, Wendlandt *et al.* have recently demonstrated that even software implementations can achieve very fast signature verification [10] (see Section 4 for more details).

3.2 TRAP

TRAP handles the dynamic switching between passive and active regulation. In passive traffic regulation, a host uses a regular IP address, allowing other hosts to send data packets directly to it. Once the host is deemed under attack, it is assigned an address from the *covered IP address* pool and enters the "active traffic regulation" mode. A covered IP address could be identified by some subset of the bits in the address, e.g. the value of the last eight bits falls in a globally known range. This allows a router to instantly recognize whether a packet is sent to a covered IP address.

Fig. 1 shows the control messages used in TRAP for requesting and assigning a covered IP address. The traffic regulator may apply local policies in the address assignment, e.g., the policy may favor hosts that offer known services. *Some hosts offering critical services (e.g. DNS servers) may be issued permanent covered IP addresses* as they are primary targets of attacks.

After granting a covered IP address to a host, the traffic regulator immediately notifies the local network's authoritative DNS servers to update the DNS record for this host – all new senders should use the new address to communicate with the host. *In addition, the host should notify all the hosts that are actively communicating with it* so that they will send requests for authorization to the new address. **If the above measures are taken, few *legitimate* senders will try to reach the old address after the address change**, due to the following reasons: (i) if this destination host is a critical server, it should use a permanent covered IP address and its users would not encounter this problem at all; (ii) if the host does not provide any service, there should not be any other legitimate hosts initiating connections to it; and (iii) if the host provides some non-critical service, the user could use a short expiration time for the DNS record. This approach has already been adopted by many users, as evidenced by the popularity of dyndns.org, which provides dynamic DNS records with an expiration time of 60 seconds.

A malicious source may flood data packets to the old address. Although this attack cannot overwhelm any host due to the address change, it may exhaust network bandwidth. Such attacks can be mitigated by several mechanisms, one of which is the lightweight bandwidth sharing policies described in Section 3.4 (we omit the other mechanisms due to space constraints; please refer to [9] for details).

3.3 TCEP

Fig. 2 illustrates the TCEP protocol that establishes a transmission channel between two hosts before their data communication begins (all their subsequent data flows use the same channel). First, the source sends an **SER (Source Endorsement Request)** message to its traffic regulator, which verifies the message's source address, e.g., by checking if it falls in the address blocks assigned to the network. Note that the two-way communication between the requester and the traffic regulator further limits address spoofing. If the address appears to be authentic, the traffic regulator checks it against a "blocking list" – a list of misbehaving local hosts identified by other edge networks; messages from any requester on this list are rejected. In addition, the network may have some local policies for granting the endorsement, e.g., a maximum endorsement frequency for each host to prevent flooding of control traffic.

If the traffic regulator decides to endorse the request, it sends back an **SEA (Source Endorsement Authorization)** message to the source, which contains a newly generated

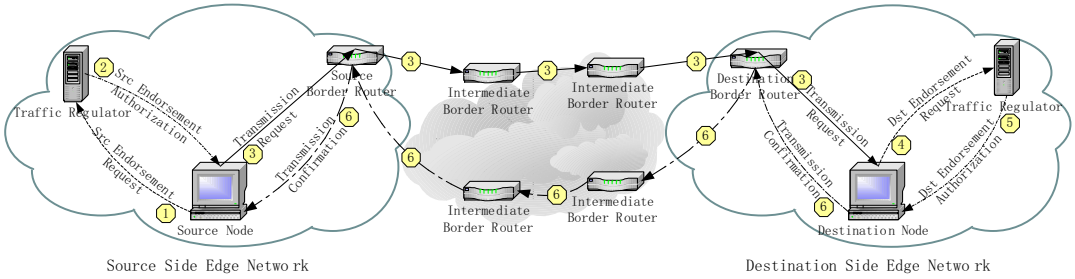

Figure 2. TCEP Message Exchange

endorsement and the original values used to compute the endorsement. The endorsement is computed over the following values uniquely identifying the requester's proposed data transmission channel: (1) source address in the SER message's IP header; (2) the proposed destination address contained in the SER message body; (3) the traffic regulator's address; and (4) a timestamp and a unique ID generated by the traffic regulator for detecting replay attacks. The endorsement is verified by all the *border routers* traversed by the *endorsed* message.

The source then sends a **TR (Transmission Request)** message to the destination with the traffic regulator's address, the timestamp, the unique ID, and the endorsement (note that the TR message's IP header contains the source and destination addresses). Along the path from the source network to the destination network, each border router performs two functions: *(a) verifying endorsement:* the router first looks up the BGP routing table to find the origin AS of the source IP address, and then retrieves the public key associated with that AS from its key database. The router also verifies if the traffic regulator address carried in the message matches one of the traffic regulators in the public-key record (in case there are multiple origin ASes for the same address); *(b) generating authenticator for subsequent data traffic:* the router computes a keyed hash over the source and destination addresses using a local secret key. The hash value is appended to the TR message and carried to the destination host. The series of hash values becomes the *authenticator* for subsequent data messages. Because the secret key in each border router changes periodically, the authenticator needs to be renewed periodically (but these renewals do not involve control messages). If the source misbehaves, the receiver will not return the new authenticator to the source, thereby preventing the source from sending more data traffic. This kind of hash-based authenticator was proposed in [1] and refined in [11] and [12].

Once the destination receives the TR message, it decides whether to accept the request. If this source has never communicated with the destination over an authorized transmission channel before, the destination may simply accept the request; otherwise, the destination can make an informed decision based on the source's past behavior. It then sends a **DER (Destination Endorsement Request)** with its decision to its local traffic regulator. The traffic regulator performs address verification and enforces local policies, e.g., how frequently a host can authorize new flows. *Such policies can prevent the destination from accepting too many transmission requests, thus breaking down collusion.*

Upon successful validity check, the traffic regulator sends a **DEA (Destination Endorsement Authorization)** message containing an endorsement to the destination host. The received endorsement and the authenticator for the data communication are placed in a **TC (Transmission Confirmation)** message and sent back to the source host. All the participating border routers on the reverse path verify the endorsement in the message. Upon the receipt of such a confirmation message, the source host extracts the authenticator and fills it in the IP header's optional portion in its subsequent data packets for routers to verify.

We now describe how the **blocking list** is established through cooperation among edge networks. A source might be identified by the destination as a potential DoS source (e.g., through a reverse Turing test), and as a result, its authenticator was not renewed. If this source tries to establish a transmission channel with the destination again, the destination can report the source to the source network through the Transmission Confirmation message. More specifically, when an endorsed TC message containing a reject decision reaches the source network's border router, the router will verify the endorsement in the message and report the source to its local traffic regulator. The traffic regulator will add the source to its *blocking list* and stop endorsing this node's later communication requests to *this particular destination*. A malicious destination cannot use this mechanism to block the traffic from this source to other destinations.

3.4 Packet Processing Rules

In PATRICIA, an ingress border router divides traffic into four classes: *Control*, *Authorized*, *Regular*, and *Demoted*. The **Control** class consists of endorsed control messages with relatively low traffic volume. This class has the highest priority to ensure timely delivery of control messages. The **Authorized** class contains data packets with

valid authenticators. This class has the second priority. The **Regular** class contains data packets that do not require active traffic regulation and also has the second priority. The **Demoted** class contains data packets without correct authenticators and has the lowest priority. To prevent collateral damage, each class of traffic receives a *dedicated* queue at each router, and the first three classes are allocated their own share of bandwidth. Note that one difference between our work and TVA is the *Regular* class, since we do not require all communications to be authorized by default.

To control flooding attacks within each traffic class, we want flows in the same queue to share the allocated bandwidth fairly, but perfect fair-queuing has extremely high overhead. Fortunately, our work does not require perfect fair-queuing since we focus on best effort traffic and can therefore use lightweight bandwidth sharing mechanisms. For example, RED-PD [7] approximates fair-queueing with low state overhead and minimal processing overhead, by maintaining the dropping probability and history of only high-bandwidth flows. We apply RED-PD on a per source AS basis for control traffic, a per-flow basis for authorized traffic, and a per-destination basis for regular traffic (see [9] for more information).

4 Deployment Issues

Wide deployment of PATRICIA is achievable for the following reasons. First, PATRICIA allows incremental deployment. Any intermediate networks that do not deploy PATRICIA will forward the control messages as regular packets. If a network does not deploy PATRICIA, it can still communicate with a PATRICIA-capable network except that its packets will be demoted to the lowest-priority queue when the destination uses a covered IP address. Second, PATRICIA is highly beneficial to edge networks that host critical services (e.g. popular web servers or high-level DNS servers), as they require stringent protection against disruptive traffic. These networks could become the early adopters. Third, due to the popularity of the early adopters' services, the deployment domain could expand to a large number of other edge networks, because the end users in those networks need to access the critical services provided by PATRICIA-protected servers *during attacks*. Fourth, Internet service providers (ISPs) are likely to support PATRICIA due to demands from their customers, since malicious traffic can be stopped earlier if the ISPs adopt PATRICIA.

PATRICIA incurs processing overhead at the regulators and border routers. However, only a small number of destinations may require active traffic regulation at any given time; most destinations are in passive mode. Moreover, advances in both software and hardware continue to enhance processing capability. In [10], the authors measured the processing overhead of the Rabin-Williams algorithm, one of

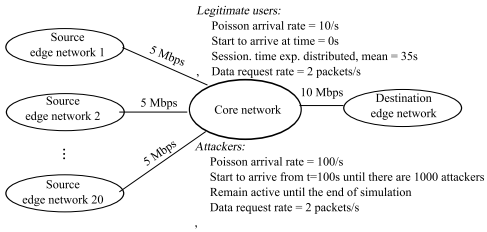

Figure 3. Network topology for simulation.

the fastest public-key signature schemes with comparable security to RSA. Their software implementation achieves 3.4 ms for signing and 61 μs for verification on a 3.2 GHz Pentium-IV PC. If 10% of a link's capacity is reserved for control messages, and a PATRICIA control message has at least 200 bytes (a Rabin-Williams signature requires 80 bytes, and the IP header requires 20 bytes), their software implementation running on the PC can handle the signature verification for a link faster than 262 Mbps. Hardware-based implementations using FPGA or ASIC can handle even faster links. For hash code verification, please refer to [12] for a detailed feasibility evaluation.

Another practical concern is that an irresponsible edge network may allow its sources to send a large number of control messages with invalid endorsements to exhaust resources in other networks, even though its border routers should have dropped the messages. In this case, the upstream ISP can easily detect invalid endorsements and "punish" the edge network by demoting its traffic. It is also hard to launch DDoS attacks this way because the attacker needs to find a large number of irresponsible edge networks.

5 Performance Evaluation

We conducted simulations with several attack scenarios and obtained similar results. We present a typical one with 20 source edge networks and 1 destination edge network as shown in Fig. 3. Legitimate senders are generated and assigned randomly to one of the source networks with a Poisson arrival rate of 10/s and an exponentially distributed lifetime of 35 seconds on average. Based on the queueing theory of M/M/inf systems, there are on average 350 legitimate senders in the network. The attack starts at time 200s and the number of attackers increases to 1000 in about 10 seconds. Once generated, these attackers do not leave the network (i.e. infinite lifetime). Both legitimate senders and attackers have the same sending rate: two 1000-byte packets/s, but as the attackers greatly outnumber the legitimate senders, they will consume most of the bandwidth if no defense mechanism is in place.

The receiver processes up to 800 data packets per second with a queue size of 200 packets. If the queue occupancy level exceeds a threshold T_h, the receiver triggers a generic attacker identification mechanism with a correct identifica-

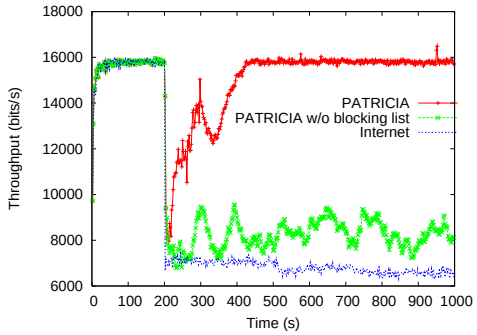

Figure 4. Throughput of legitimate senders.

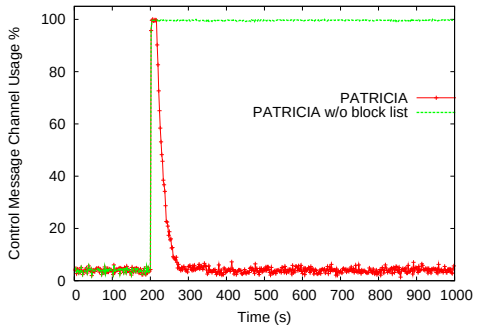

Figure 5. Congestion of control channel.

tion rate and a false alarm rate. The identification mechanism terminates when the queue occupancy level falls below a threshold T_l and the number of identified attackers per second is less than a threshold N. In our simulation, we set $T_h = 80\%$, $T_l = 20\%$, and $N = 7/s$.

Fig. 4 shows the throughput of legitimate senders in three cases: *PATRICIA*, *PATRICA without blocking list*, and *the Internet*. We have the following observations: (a) the Internet case has the lowest throughput – each legitimate sender receives a throughput of 6 - 7Kbps on average after the attack, even though their sending rate is 16Kbps; (b) PATRICIA performs the best among the three: the senders maintain a 16Kbps throughput after a short recovery period following the attack. During the recovery period, the receiver identified the attackers with a probability of 0.1 every 2 seconds. Obviously, a higher identification probability will result in a shorter recovery period; (c) without the blocking list, PATRICIA is far less effective because of the congested control channel. Even though the attackers were denied access to the authorized data channel, they kept sending request messages, thus congesting the control channel and blocking the legitimate senders from obtaining their authorizations. Fig. 5 shows that after the recovery period, PATRICIA had a non-congested control channel while the case without blocking list still suffered from the flooding of control messages. These results show that PATRICIA can effectively mitigate a DoS attack against both the control and data channels.

6 Conclusion and Future Work

We presented PATRICIA for regulating undesirable Internet traffic, where edge networks are involved in flow authorization between hosts. Simulation results show that PATRICIA protects legitimate sources against flooding of both control and data traffic. In our future work, we will conduct more extensive simulations, investigate approaches to distribute public keys, and refine the protection scheme against attack traffic to obsolete IP addresses. We also plan to develop prototypes of PATRICIA-capable hosts and routers.

Acknowledgment

We thank Brad Montgomery for his help at the beginning of this project. We are also grateful to Andreas Terzis and the anonymous reviewers for their insightful comments.

References

[1] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial-of-Service with Capabilities. In *Proceedings of Hotnets-II*, Nov. 2003.

[2] L. Andersson, E. Davis, and L. Zhang. Report from the IAB workshop on unwanted traffic March 9-10, 2006. *Work in Progress*, Feb. 2007.

[3] K. Argyraki and D. R. Cheriton. Active Internet traffic filtering: Real-time response to Denial-of-Service attacks. In *USENIX Annual Technical Conference*, 2005.

[4] K. Argyraki and D. R. Cheriton. Network capabilities: The good, the bad and the ugly. In *Proceedings of the ACM Hot Topics in Networks (HotNets) Workshop*, 2005.

[5] D. Chaum and E. van Heijst. Group Signatures. In *Advances in Cryptology - Eurocrypt '91*, pages 257–265, 1991.

[6] ICANN. Factsheet root server attack on 6 February 2007, Mar. 2007.

[7] R. Mahajan, S. Floyd, and D. Wetherall. Controlling high-bandwidth flows at the congested router. In *Proceedings of ICNP '01*, Nov. 2001.

[8] P. Vixie, G. Sneeringer, and M. Schleifer. Events of 21-oct-2002. `http://d.root-servers.org/october21.txt`, Nov. 2002.

[9] L. Wang, Q. Wu, and D. D. Luong. Engaging edge networks in preventing and mitigating undesirable network traffic. Technical Report CS-07-004, U. Memphis Computer Science Department, May 2007.

[10] D. Wendlandt, D. G. Andersen, and A. Perrig. Bypassing network flooding attacks using FastPass, 2006. `http://www.cs.cmu.edu/~dwendlan/fastpass/fastpass_short.pdf`.

[11] A. Yaar, A. Perrig, and D. Song. SIFF: A stateless Internet flow filter to mitigate DDoS flooding attacks. In *Proceedings of the IEEE Security and Privacy Symposium*, May 2004.

[12] X. Yang, D. Wetherall, and T. Anderson. A DoS-limiting network architecture. In *Proceedings of the ACM SIGCOMM '05*, pages 241–252, 2005.