

# Analyze and Determine the IP Spoofing Attacks Using Stackpath Identification Marking and Filtering Mechanism

V. Shyamaladevi<sup>1</sup> · Dr. R.S.D Wahidabanu<sup>2</sup>

<sup>1</sup> Research Scholar, K.S.Rangasamy College of Technology Tiruchengode, Tamilnadu, India  
shyamala\_msc@yahoo.com

<sup>2</sup> Research Supervisor, Government College of Engineering, Salem, Tamilnadu, India

## ABSTRACT

Distributed denial of service is a major threat to the availability of internet services. DDoS attacks are stealthy; The Internet service provider (ISP) finds it when the new technique propose a defense mechanism, stack identification and marking (SPi & M), which the previous approaches are allows the host being attacked, or its upstream ISP, to filter out attack packets and to detect spoofed source IP addresses, on a per-packet basis. This work proposed StackPi-Write ahead with a new packet marking scheme based on Pi, and new filtering mechanisms. Stack marking is the similar for TTL marking the packet's TTL to aggregate the markings from different routers; each router instead treats the IP Identification field as though it were a stack. Write ahead marking All the router needs to do is substitute its own IP address for the last-hop IP address and the next-hop IP address for its IP address when calculating the bits to mark, they need not be repeated for each forwarded packet and also develop a new filter, the *PiIP filter*, which can be used to detect IP spoofing attacks with just a single attack packet. StackPi filtering can thus defend against not only DDoS attacks, but also many IP spoofing attacks and multicast source spoofing attacks. A packet is marked deterministically by routers along its path towards the destination. Packets traveling along the same path will have the same marking so that an attack victim need only identify the StackPi marks of attack packets to filter out all further attack packets with the same marking. In this scheme almost completely eliminates the effect of legacy routers in small quantities and performs 3-5 times better than the existing system. For the filtering mechanism, derive an optimal threshold strategy for filtering with the Path identification marking. Finally, evaluate the Stack path identification's compatibility with IP Fragmentation, applicability in an IPv6 environment.

**Index Terms:** Denial-of-service, DoS, IP address spoofing, packet marking, Stack marking, Write ahead marking, Path Identifier.

## I. INTRODUCTION

### A. IP Spoofing and DDoS Attacks

Internet security is becoming of critical importance in today's computing environment, as our society, government, and economy is increasingly relying on the

Internet. Unfortunately, the current Internet infrastructure is vulnerable to attacks—in fact, malicious attacks on the Internet have increased in frequency and severity. Large scale Distributed Denial-of-Service (DDoS) attacks disrupt critical Internet services and cause significant financial loss and operational instability. One of the most difficult challenges in defending against DDoS and many other attacks is that attackers often spoof the source IP address of their packets and thus evade traditional packet filters. Unfortunately, the current routing infrastructure cannot detect that a packet's source IP address has been spoofed or from where in the Internet a spoofed IP packet has originated from. The combination of these two factors makes IP spoofing easy and effective for attacks. In fact, many different types of Internet attacks utilize spoofed IP addresses for different purposes. In a TCP hijacking attack, an attacker can inject malicious data into a TCP connection and potentially even hijack the connection if it knows the IP address, TCP port number, and correctly guesses the sequence numbers being used. Consider a *telnet* session, where the attacker inserts the UNIX command `rm -rf /`, which would delete all files of the current user. Another form of this attack is for the attacker to send TCP RST packets to close an existing connection. Such attacks can be particularly disruptive if the TCP connection is a BGP session between two routers.

### B. Proposal

We propose a new approach, called *StackPi* (short for *Stack Path Identifier*), which is the first defense mechanism that satisfies all of the above desired properties. In StackPi, as a packet traverses routers on the path towards its destination, the routers deterministically mark bits in the packet's IP Identification field. The deterministic markings guarantee that packets traveling along the same path will have the same marking. StackPi allows the victim and routers on the attack path to take a proactive role in defending against a DDoS attack by using the StackPi mark to filter out attack packets on a per packet basis. In addition, the victim can build statistics over time relating StackPi marks to IP addresses. Then if an attacker spoofs an IP address, it is likely that the StackPi mark in the spoofed packet will not match the StackPi mark corresponding to the legitimate IP address in the database, thus enabling the victim to tag packets with

possibly spoofed source IP addresses. StackPi is not only effective against large scale DDoS attacks, but also effective against other IP spoofing attacks such as TCP hijacking and multicast source spoofing attacks. Our scheme is extremely efficient and responds quickly to attacks. Proposed system develops the path identification IP filter, which can be used to detect IP spoofing attacks with a single attack packet. It also examines the conflicts between IPv4 fragmentation and path identification marking, and path identification deployment in an IPv6 environment.

II. LITERATURE REVIEW

Sung and Xu propose an altered IP traceback approach, where the victim tries to reconstruct the attack path but also attempts to estimate if a new packet lies on the attack path or not [15]. Their scheme is probabilistic and each router either inserts an edge marking for the IP traceback scheme or a router marking identifying the router. Unfortunately, their approach requires the victim to collect on the order of 105 attack packets to reconstruct a path, and once the path is reconstructed, this scheme will likely have a high false positive rate as the routers close to the victim will all lie on some attack path and frequently mark legitimate packets which will then get rejected. The original Path identification marking is based on the use of the packet's TTL field as an index into the IP Identification field where a router should add its marks. This method is not as lightweight as the Stack Path identification method. Legacy routers have a harmful affect on the original Path identification scheme because they decrement the TTL of a packet but do not add any markings. The Stack Path identification scheme is robust to legacy routers and even includes the write-ahead scheme to incorporate markings for single legacy routers in the path.

Collins and Reiter use a novel approach of combining Cisco NetFlow data from a large network with Skitter map data, to compare DDoS defense mechanisms [4]. They measure the effectiveness of path aware defense systems (Path identification and Hop-Count Filtering), as well as Static and Network-aware clustering. Recently, network capability-based systems have been proposed for DDoS defense. Machiraju et al. propose a secure Quality-of-Service (QoS) architecture that is based on network capabilities [10]. Lakshminarayanan et al. leverage the i3 infrastructure to enable a receiver to cut off unwanted senders [8]. Anderson et al. [2] present an infrastructure where the sender uses a capability to set up a path to the receiver. We subsequently proposed SIFF, a capability-based system that allows a receiver to enforce flow-based admission control [16].

Yang et al. propose a capability-based mechanism with fine-grained service levels that attempts to address the denial-of capability attack [17]. They leverage Path identification markings to filter out floods of request packets in their scheme routers attempt to provide fair sharing among capability request packets based on their Path identification markings. Path identification are complementary to capability-based systems, and can be

used to mitigate spoofing and flooding in the capability request channel.

III. SPIS WITH ROUTER WRITE AHEAD

The intuition behind the Stack marking scheme is in order to generate a path identifier that is representative of a particular path from a source to a destination in the Internet, each router along the path must contribute some small amount of information whose aggregate among the routers of the path will be the StackPi marking. We store the StackPi mark in the IP Identification field of the IP header. Because the IP Identification field is only 16 bits long, we are limited to using only 16 bits for our marking, thus allowing for 216 unique StackPi marks. Although there are likely more than 216 possible paths from any end-host in the Internet to our DDoS victim, we believe (and later show) that this level of granularity is sufficient to allow the victim of an attack to accurately filter out attack packets, while still maintaining good service for legitimate users. Stack marking is the same as that for TTL marking: in order to generate a path identifier that is SIFF, a capability-based system that allows a receiver to enforce flow-based admission control [16].

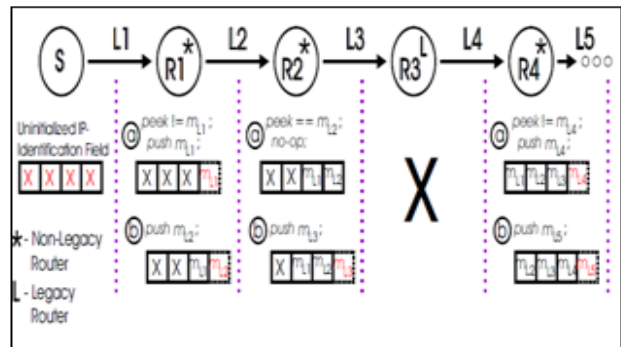


Figure 1. Stack Marking Scheme with Write-ahead Technique

representative of a particular path from a source to destination in the Internet, each router along the path must contribute some small amount of information whose aggregate among the routers of the path will be the Pi marking. However, instead of using the packet's TTL to aggregate the markings from different routers, each router instead treats the IP Identification field as though it were a stack. Upon receipt of a packet, a router shifts the IP Identification field of the packet to the left by n bits and writes its marking bits (calculated in the same way as in TTL marking) into the least significant bits that were cleared by the shifting. In other words, the router simply pushes its marking onto the stack. Because of the finite size of the Identification field, the n most significant bits, which represent the oldest mark in the packet, are lost in this process; just as in TTL marking. In fact, Stack marking and TTL marking are equivalent in the case of 100% deployment. The differences between TTL and Stack marking become evident when legacy routers are introduced into the topology.

Unlike TTL marking, which interacts poorly with legacy routers because of its reliance on the packets TTL which is modified by legacy routers, Stack marking does not rely on the TTL, and hence, has no interaction with legacy routers at all. There are no longer any marking

holes because each marking router places its mark adjacent to the last marking router's mark, in the least significant bits of the IP Identification field. Completely marking the whole field using Stack marking requires only that there be 16 non-legacy routers anywhere in the path. Stack marking eliminates the interaction between Pi enabled routers and legacy routers that is present in TTL marking. However, Stack marking is still limited in that a path which has too few marking routers will still result in end-host initialized bits arriving at the victim; which allows attackers to shift between different Pi markings.

We can add an extra step to our marking scheme to improve this situation slightly. We already assure that each router knows the IP address of the last-hop routers or hosts from which it receives packets (this knowledge is necessary for generating the marking bits of each router explained). If we also assume that each router knows the IP address of the next-hop routers or hosts to which it is forwarding packets, then the router is capable of marking the packets *on their behalf*. All the router needs to do is substitute its own IP address for the last-hop IP address and the next-hop IP address for its IP address when calculating the bits to mark (of course, the results of this calculation should also be cached so that they need not be repeated for each forwarded packet). This second marking is called *Write-ahead marking*. The benefits of Write-ahead marking are immediately evident when considering a Pi-enabled router followed by a legacy router in a path. In this case, the Pi-enabled router will mark not only for itself but also for the next hop router, the legacy router, so that its marks will be included as well.

There is, however, a slight complication with the Write-ahead scheme: what happens when two Pi-enabled routers are adjacent to each other in a path. It would be a waste of space for the second Pi enabled router to add its own mark to the packet, since the first Pi-enabled router would have added that mark already. Therefore, we are forced to change our scheme slightly to accommodate this situation. Upon receipt of a packet, a router *peeks* (the process of looking at the item in the top of the stack without modifying the stack itself) at the least significant n bits and compares the marking it finds there to what its own marking would be. If the markings are identical, then the router will assume that the last hop router is Pi-enabled and has already marked the field on the current router's behalf. In this case, the current router will skip pushing its own marking and will only push the next-hop routers.

A. Packet Filtering

In this section, we explain how a DDoS victim can use the StackPi mark to defend against a DDoS attack. StackPi is one of the first schemes that allow the victim to take a proactive role in filtering out attack packets. Because StackPi allows for per-packet filter decisions and is geared to defend against DDoS attacks and IP spoofing attacks, it is extremely important that the filters at the end-host have a low per packet computation cost, as an end-server will need to be able to filter every packet that arrives over the network.

In this section, we present two efficient packet filtering methods. Note that the filtering mechanism is not limited

to deployment only at the victim. To prevent last-hop bandwidth starvation DDoS attacks, StackPi filters can be deployed at the victim's ISP. StackPi filters can be deployed throughout the Internet to enhance the filtering capability of up stream routers in schemes such as Pushback [15]. The simplest filtering strategy for an end-host under attack is to build a list of the StackPi markings of offending packets and compare each incoming packet's marking to that list; if there is a match, then the incoming packet is assumed to be an attack packet and is dropped without further processing. The problem with this simplified filter is that it is vulnerable to marking saturation, that is, it is possible for an attacker to DoS legitimate users with the same StackPi marking by causing a server to add the attacker's StackPi marking to its drop list, even if the attacker's traffic is only a marginal portion.

$p_j$  - The total number of packets sent by entity j

$v_j$  - The total number of packets sent by j accepted by the victim.

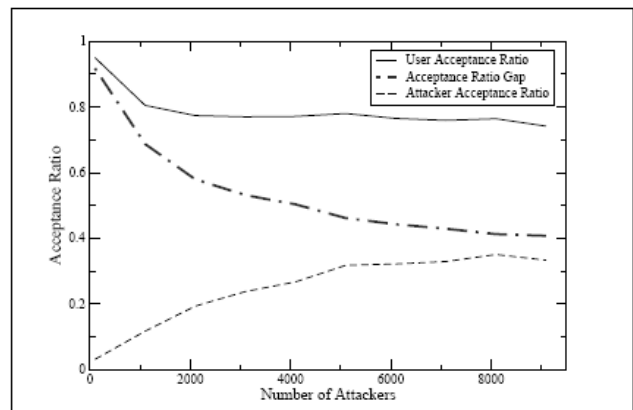


Figure 2. Optimal Threshold Filter Using Internet map

The acceptance ratio,  $a_j$  for a given entity j is defined as:

$$a_j = \frac{v_j}{p_j}$$

The goal of the victim, then, is to maximize the difference

$$\Delta = \frac{v_U}{p_U} - \frac{v_A}{p_A}$$

between the two ratios: the acceptance ratio gap is a useful metric to determine how a particular StackPi filter performs relative to no filter. Without StackPi marking and filtering, the victim can only make accept/drop decisions at random, which intuitively gives an equal user and attacker acceptance ratio, or an acceptance ratio gap of zero. The acceptance ratio gap is a useful metric to determine how a particular StackPi filter performs relative to no filter. Without StackPi marking and filtering, the victim can only make accept/drop decisions at random, which intuitively gives an equal user and attacker acceptance ratio, or an acceptance ratio gap of zero.

$$\Delta = \frac{v_U}{p_U} - \frac{v_A}{p_A} = \frac{\sum_{i=0}^{2^{n-1}-1} v_{U_i}}{p_U} - \frac{\sum_{i=0}^{2^{n-1}-1} v_{A_i}}{p_A} = \sum_{i=0}^{2^{n-1}-1} \left( \frac{v_{U_i}}{p_U} - \frac{v_{A_i}}{p_A} \right) \quad (1)$$

B. Filtering (StackPi-IP Filter)

The StackPi filtering extremely light-weight and efficient, but here it present a slightly more complex but more accurate filtering method. The filter itself is simple examine an incoming packet's StackPi mark *and* source IP address and allow access based on that tuple. Ideally, a database of legitimate users' h Stack Pi, IP i tuple will be built during times when there are few or no ongoing attacks. Any packet with a StackPi marking that does not match the StackPi marking of the same IP address in the database will be flagged as a packet with a spoofed IP address. In the case of filtering based only on StackPi markings, we have to assume that our filters get feedback from some higher layer algorithm that can classify some sampled packets as legitimate packets or attack packets, and tell the filter which StackPi markings correspond to attack traffic and should be dropped. The StackPi-IP filter does not rely as strongly on this assumption, because the StackPi-IP filter does not need to be bootstrapped with attack traffic. Quite the opposite, the StackPi-IP filter is bootstrapped during non-attack periods and identifies attack periods by an increase in the incidence of packets with spoofed IP addresses. We define the set of n distinct StackPi markings recorded for address k as {m0,m1.....;mn}. For each Stack Pi mark recorded at the victim for IP address k, there is a set of other IP addresses that also map to the same StackPi mark. If the attacker were to spoof any of these, the attack packet would be accepted by the filter. Thus, the probability of an attacker with IP address k successfully spoofing is:

$$P_k = \frac{\sum_{i=0}^n \text{unique IPs}(m_i, k)}{N} \tag{2}$$

where the unique IPs() function returns the number of unique IP addresses that map to StackPi mark mi, excluding IP address k as well as any duplicates between function calls, and N represents the number of end-hosts in the topology; which is the size of the list of possible IP addresses that the attacker can spoof. Given the probability of an attacker with a specific IP address of successfully spoofing a packet, we can now calculate the probability of an attacker with a random IP address successfully spoofing:

$$P = \frac{\sum_{k=0}^N P_k}{M} \tag{3}$$

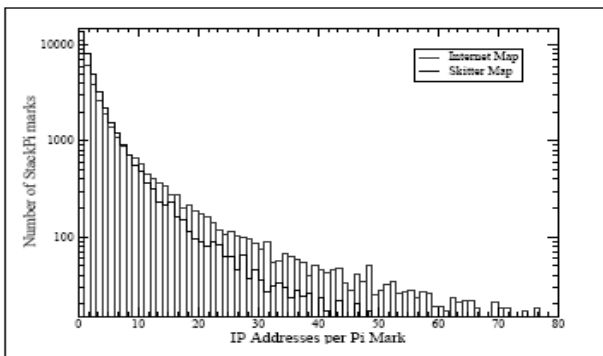


Figure 3. Histogram of the frequency of StackPi marks with a particular

number of IP addresses.

IV. PERFORMANCE EVALUATION AND DEPLOYMENT

A. Attack Model

In order for a DDoS victim to protect itself against attack packets, it must have a way to differentiate them from legitimate users' packets. Although StackPi filtering provides such a method, the filter must be initialized with the StackPi marks of known attack packets. It is outside the scope of this paper to define an algorithm for attack packet identification. In order to reflect the use of such an algorithm to boot strap our filter, we have modeled our DDoS attack simulation in two phases. In the first phase, the *learning phase*, all packets are assumed to be analyzed by the victim, using some packet identification algorithm.

In other words, the victim is given the power to differentiate between attack and user packets that arrive in the learning phase. Thus, the victim is able to generate an attacker StackPi markings list. In the second phase, the *attack phase*, we assume that the victim is no longer able to automatically differentiate attack and user packets, and thus is forced to rely on StackPi filtering to make accept and drop decisions. The learning phase of our DDoS attack lasts for 16 packets and the attack phase lasts for 10 packets. All our results are taken from the attack phase of the DDoS attack. For our experiments, we use sampled Internet topologies from Burch and Cheswick's Internet Mapping Project [14] and from the Skitter Project distributed. Both datasets are constructed by having a single host send trace route probes to a large number of destinations, listing all router IP addresses on the resulting paths. We filtered the data sets to remove all incomplete and duplicate routes (although multiple routes to the same end-host were not removed). We also removed all routes of path length shorter than seven hops. In our simulated attacks, the single trace route host is used as the DDoS victim server.

Our DDoS simulations proceed as follows: a certain number of *paths* are selected, at random, from the topology file and assigned to be either attack or legitimate user paths. Unless otherwise noted, all of the DDoS simulations select 10,000 paths for legitimate users and vary the number of attackers.

Because compromised machines and dedicated attackers generally send more traffic to the victim during a DDoS attack, in our simulations, we assume that each attacker sends 10 times more packets than each user. When we refer to the duration, in packets, of a phase of the DDoS attack, we do it from the users' point of view. Thus, in the learning phase, each user sends 16 packets and each attacker sends 160 packets. Like wise, for the attack phase, each user sends 10 packets and each attacker sends 100 packets. Our results are the averages of 3 runs of an attack with a particular topology and particular set of parameters.

B. StackPi in IPv6 environment

Although the Pi scheme has been specifically designed for deployment in IPv4, its principal ideas are equally applicable in an IPv6 environment. The IPv6 protocol

does not support en-route packet fragmentation, and thus does not have an equivalent field to the IP Identification field of IPv4. There are, however, two possibilities for marking space in IPv6: in the *flow identification* field or in a *hop by-hop* option. The advantage of marking in the flow identification field of the header is that because the field is part of the standard header, router markings will not add to the packet's size (which might cause the packet to exceed the MTU of an intermediate network and be dropped). The flow identification field is 20 bits in length, which allows more routers to include their markings in each Pi-mark. Of course, this is not the purpose that the flow identification field was meant to serve [17]. The other option is to include the Pi marking in a hop by-hop option inserted by the first Pi enabled router in the path. The benefit of this approach is that the length of the option need not be limited to 20 bits, as is the flow identification field. However, inserting such an option into the packet may cause it to exceed the MTU of a link somewhere along the path. In either case, DDoS protection is a critical feature that should be present at the network level, and IPv6's current limited deployment makes it a good candidate for modification to include the Pi scheme.

#### CONCLUSION

In this work, we present StackPi, a novel approach to defending against DDoS and IP spoofing attacks. The StackPi defense is split into two parts: StackPi marking and StackPi filtering. The StackPi marking scheme defines how the routers along a packet's path create a deterministic marking in the packet such that each packet traversing the same path has the same marking. StackPi marking includes a write-ahead method whereby some legacy routers' markings can be included in the StackPi mark. The StackPi filter defines how an end-host utilizes the packet markings to filter out attack traffic. We present two types of StackPi filters: threshold and IP based. Threshold filters allow the DDoS victim to minimize the false positive and false negative rates of incoming traffic. IP based filters allow the victim to detect, with high probability, a spoofed source IP address in an incoming packet. We show that StackPi provides strong protection to a DDoS victim, and shows strong incremental deployment properties. StackPi is compatible with IPv4 fragmentation and can operate in an IPv6 environment. StackPi can also be used to complement certain existing anti-DDoS schemes such as Pushback and SPIE. There are many extensions to StackPi that have yet to be explored, such as variable bit marking and the application of machine learning techniques to StackPi filtering, that promise to make StackPi a critical deterrent to today's most common Internet attacks.

#### REFERENCES

- [1] Micah Adler. Tradeoffs in probabilistic packet marking for IP traceback. In *Proceedings of 34th ACM Symposium on Theory of Computing (STOC)*, 2002.
- [2] Abraham Yaar, Adrian Perrig, and Dawn Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *IEEE Symposium on Security and Privacy*, May 2003.
- [3] SETI@home, "Search for extraterrestrial intelligence (SETI)," <http://setiathome.ssl.berkeley.edu/>, 2003.& CERT, "TCP SYN flooding and IP spoofing attacks,"
- [4] Vern Paxson, "An analysis of using reflectors for distributed denial-of-service attacks," *Computer Communication Review*, vol. 31, no. 3, 2001.
- [5] John Ioannidis and Steven M. Bellovin, "Implementing pushback: Router-based defense against DDoS attacks," in *Proceedings of the Symposium on Network and Distributed Systems Security (NDSS 2002)*, San Diego, CA, Feb. 2002, Internet Society.
- [6] L. Garber, "Denial-of-service attacks rip the internet," in *IEEE Computer*, April 2000, vol. 33.[20] J. Howard, *An Analysis of Security Incidents on the Internet*, Ph.D. thesis, Carnegie Mellon University, August 1998.
- [7] David Moore, Geoffrey Voelker, and Stefan Savage, "Inferring internet denial of service activity," in *Proceedings of the 10th USENIX Security Symposium*, Washington, D.C., Aug. 2001, USENIX.
- [8] Kihong Park and Heejo Lee, "On the effectiveness of route-based packet filtering for distributed DoS attack prevention in power law internets," in *ACM SIGCOMM '01*, 2001
- [9] A. Mankin, D. Massey, C.L. Wu, S.F. Wu, and L. Zhang, "On design and evaluation of intention-driven ICMP traceback," in *Proceedings of the IEEE International Conference on Computer Communications and Networks*, Oct. 2001.
- [10] Micah Adler, "Tradeoffs in probabilistic packet marking for IP traceback," in *Proceedings of 34th ACM Symposium on Theory of Computing (STOC)*, 2002.
- [11] Drew Dean, Matt Franklin, and Adam Stubblefield, "An algebraic approach to IP traceback," *ACM Transactions on Information and System Security*, May 2002.
- [12] Michael Goodrich, "Efficient packet marking for large-scale IP traceback," in *Proceedings of the 9th ACM Conference on Computer and Communications Security*. 2001, pp. 117–126, ACM Press.
- [13] Heejo Lee and Kihong Park, "On the effectiveness of probabilistic packet marking for IP traceback under denial of service attack," in *Proceedings IEEE Infocomm 2001*, Apr. 2001.
- [14] Stefan Savage, David Wetherall, Anna Karlin, and Tom Anderson, "Network support for IP traceback," *ACM/IEEE Transactions on Networking*, vol. 9, no. 3, June 2001..
- [15] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, and W. Timothy Strayer, "Hash-based IP traceback," in *Proceedings of the ACM SIGCOMM 2001 Conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM'01)*, Aug. 2001, pp. 3–14.
- [16] Alex C. Snoeren, Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer, "Single-packet IP traceback," *IEEE/ACM Transactions on Networking (ToN)*, vol. 10, no. 6, Dec. 2002.
- [17] Minh Sung and Jun Xu, "IP traceback-based intelligent packet filtering: A novel technique for defending against internet DDoS attacks," in *Proceedings of IEEE ICNP 2002*, Nov. 2002.
- [18] Avi Yaar, Adrian Perrig, and Dawn Song. SIFF: A Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks. In *Proceedings of the IEEE Symposium on Security and Privacy*, pages 130–143, May 2004.
- [19] Xiaowei Yang, David Wetherall, and Tom Anderson. A DoS-limiting Network Architecture. In *Proceedings of ACM SIGCOMM*, pages 241–252, August 2005.