Attack Diagnosis: Throttling Distributed Denial-of-Service Attacks Close to the Attack Sources

Ruiliang Chen and Jung-Min Park
Bradley Department of Electrical and Computer Engineering
Virginia Polytechnic Institute and State University
Blacksburg, VA 24061
{rlchen, jungmin}@vt.edu

Abstract— **Attack mitigation schemes actively throttle attack traffic generated in Distributed Denial-of-Service (DDoS) attacks. This paper presents *Attack Diagnosis* (AD), a novel attack mitigation scheme that combines the concepts of Pushback and packet marking. AD's architecture is inline with the ideal DDoS attack countermeasure paradigm, in which attack detection is performed near the victim host and attack mitigation is executed close to the attack sources. AD is a reactive defense that is activated by a victim host after an attack has been detected. A victim activates AD by sending AD-related commands to its upstream routers. On receipt of such commands, the AD-enabled upstream routers deterministically mark each packet destined for the victim with the information of the input interface that processed that packet. By collecting the router interface information recorded in the packet markings, the victim can trace back the attack traffic to the attack sources. Once the traceback is complete, the victim issues messages that command AD-enabled routers to filter attack packets close to the source. The AD commands can be authenticated by the *TTL* field of the IP header without relying on any global key distribution infrastructure in Internet. Although AD can effectively filter traffic generated by a moderate number of attack sources, it is not effective against large-scale attacks. To address this problem, we propose an extension to AD called *Parallel Attack Diagnosis* (PAD) that is capable of throttling traffic coming from a large number of attack sources simultaneously. AD and PAD are analyzed and evaluated using a realistic network topology based on the Skitter Internet map. Both schemes are shown to be robust against IP spoofing and incur low false positive ratios.**

I. Introduction

Large-scale, high-profile Distributed Denial-of-Service (DDoS) attacks have become common recurring events that increasingly threaten the proper functioning of the Internet. Despite a significant breadth of research into countermeasures, DDoS attacks still remain a major threat today.

Defending against DDoS attacks is challenging for two reasons. Firstly, the number of attack sources (i.e., zombies) involved in a DDoS attack is very large. Even if the volume of traffic sent by a single zombie is small, the volume of aggregated traffic at the victim host could be overwhelming. Secondly, zombies usually spoof their IP addresses, which makes it very difficult to trace the attack traffic back to the zombies. Though *ingress filters* [3] have been deployed in many subnets to prevent IP spoofing, their effectiveness is somewhat limited because many subnets have not implemented them. Moreover, ingress filtering does not prevent *subnet spoofing* (i.e., IP spoofing within a subnet).

DDoS attack countermeasures can be categorized into four classes: prevention, detection, mitigation, and traceback. Mitigation techniques actively thwart DDoS attacks by filtering or rate limiting attack packets. A typical attack mitigation scheme usually consists of two modules: an *attack detection* module and an *packet filtering* module. The attack detection module is used to extract the characteristics of attack packets, such as source IP addresses or marked IP header values [13], [15], [16]. After the characteristics have been summarized, this information is used by the packet filtering module to filter malicious packets. The mitigation schemes presented in [7], [13], [14], [15], [16] place the two modules at the same location, i.e., either close to the victim or close to the attack source. However, placing the two modules at the same location limits their effectiveness. Attacks can be most effectively detected at the victim end where all the attack packets can be observed readily. In contrast, it is most effective to filter attack packets as close to the attack sources as possible. Therefore, in the ideal attack countermeasure paradigm, the attack detection module is placed at (or near) the victim and the packet filtering module is placed as close to the attack sources as possible. For more details on this paradigm, see [2].

Schemes proposed in [6], [9], [17] support this paradigm. In these schemes, when attacks are detected downstream close to the victim, the upstream routers close to the attack sources filter attack packets using summarized "attack signatures" sent by the detection module. This technique, however, has two drawbacks. The first drawback is the difficulty of securely forwarding the attack signatures to the upstream routers. The schemes proposed in [9] and [17] require a global key distribution infrastructure for authenticating and verifying the attack signatures. Such an infrastructure is costly to deploy and maintain. *Pushback* [6] takes a more practical approach. It uses a hop-by-hop transmission method for forwarding attack detection information which is authenticated using the *TTL* field of the IP header. The second drawback is the difficulty of creating an accurate attack signature that can be used to distinguish attack traffic and legitimate traffic. Because the zombies can spoof source IP addresses, vary the *protocol* field in the IP header, and change transport layer port numbers during an attack, these fields may not be a valid attack signature. Thus, the only absolutely reliable information in a packet is the destination IP address. However, packet filtering based on only the destination IP address will throttle the legitimate traffic as well.

In this paper, we propose *Attack Diagnosis (AD)*, a novel attack mitigation scheme that combines the concepts of Pushback and packet marking. The execution of AD can be summarized by four steps: (1) An Intrusion Detection System (IDS) installed at the victim (or at its firewall) detects an attack; (2) The victim instructs the upstream routers to start marking packets with traceback information; (3) Based on the traceback information extracted from collected packets, the victim reconstructs the attack paths; and (4) The victim instructs the appropriate upstream routers to filter attack packets.

Although AD can effectively thwart attacks involving a moderate number of zombies, it is not appropriate for large-scale attacks because its *attack mitigation delay* is too large.

Here, attack mitigation delay refers to the time interval between the start of a synchronized DDoS attack and the start of the attack-traffic filtering process. To address this problem, we propose an extension to AD called *Parallel Attack Diagnosis (PAD)* that can throttle traffic coming from a large number of zombies simultaneously. PAD significantly reduces the attack mitigation delay at the cost of increased false positives.

Our approach has the following noteworthy features: (1) The traceback information is directly utilized to filter attack traffic close to the attack sources; (2) AD and PAD are reactive defenses that incur no network overhead when a network is not under attack; (3) AD and PAD are robust against IP spoofing and incur low false positive ratios; and (4) PAD provides a "tunable" parameter that enables the network to adjust the attack mitigation delay and the false positive ratio.

The rest of the paper is organized as follows. In the next section, we describe AD and PAD in detail. In Section III, we discuss practical considerations. The simulation results are given in Section IV. We provide an overview of related work in Section V and conclude the paper in Section VI.

II. Attack Diagnosis and Parallel Attack Diagnosis

A. Assumptions

We assume the following network environment. Every host, either a client or a server, is connected to its local *edge router*. Edge routers are in turn interconnected by *core routers*. The server being attacked is called the *victim*. A recent study [4] has shown that 95% of the routes observed in the Internet have fewer than five observable daily changes. So we make the reasonable assumption that every route from a client to the victim is fixed during the timeframe of interest. We also assume that Internet routers are not compromised.

We use the term *false negative* to denote a zombie machine whose attack packets have not been filtered, and use the term *false positive* to denote a legitimate client whose packets have been incorrectly throttled.

Like other packet marking based mitigation schemes [13], [16], we assume the existence of an IDS module installed at the victim (or at its firewall), which is able to identify and collect malicious packets.

B. Overview of AD and PAD

We will illustrate the principle ideas behind AD and PAD using Fig. 1. This figure shows an upstream tree of victim V. In the figure, some of the router interfaces are labeled with a locally unique number that identifies that interface port. We call this number the *port identifier (PID)*. PID is locally unique in the sense that interfaces of two different routers can have the same PID, while the interfaces of a single router are assigned non-repeating PIDs. A typical router has multiple interface ports and can forward packets to a specified interface port using its switching fabric.

In most cases, a PID of a router can be used to uniquely identify a router or a host that is connected to it. However, when an interface port is connected to multiple hosts via a broadcast link-layer channel (such as in a LAN), a PID cannot be used to uniquely identify a host. An example of such a case is shown in Fig. 1. In the figure, interface x of router F is connected to multiple clients through a LAN. In this case, router F maintains a *virtual PID table* that maps each "virtual" interface to a MAC address. More precisely, the table maps a "virtual PID" to every MAC address that the router observes coming through interface x. For example, in Fig. 1, MAC

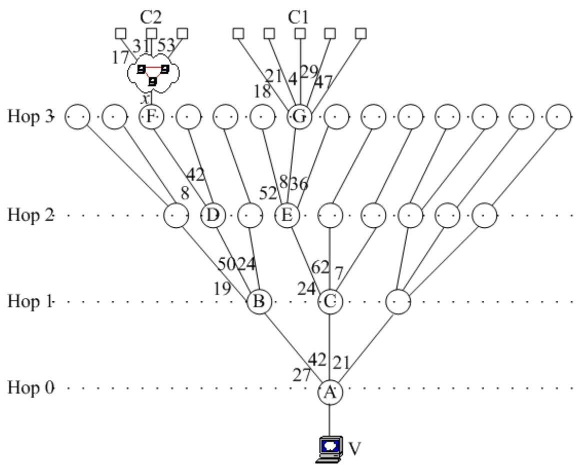

Fig. 1. The upstream tree of victim V.

address of $C2$'s Ethernet adapter is mapped to the virtual PID 31.

Since a PID is locally unique within a router, a string of PIDs can be used to uniquely identify the path from a server to a client with overwhelming probability. For example, in Fig. 1, the string 42-24-8-4 (i.e., the PIDs corresponding to each hop from hop 0 to hop 3) represents the path from V to $C1$. If $C1$ is a zombie, this string would correspond to an attack path. Hence, constructing a PID string corresponds to reconstructing an attack path in this instance. AD uses an iterative process to construct such a PID string, starting with PID 42. Details of the attack path reconstruction process are explained in Subsection II.C. Once the reconstruction of the attack path is complete, the router closest to the zombie (i.e., router G) filters all packets destined for V at interface 4.

Although AD is capable of throttling malicious traffic coming from a modest number of zombies, it does not reconstruct attack paths fast enough to effectively mitigate attack traffic coming from a large number of zombies. PAD solves this problem by dealing with multiple zombies simultaneously.

C. Attack Diagnosis

To support AD, a router needs to mark packets with its PIDs and other traceback-related information. For this purpose, we overload the *Identification* field and one reserved bit in the IP header. All probabilistic packet marking schemes overload the *Identification* field. The justification for overloading this field is based on the fact that IP fragments constitute a very small proportion of the actual Internet traffic (less than 0.25%) [10]. The marking fields of AD and PAD are shown in Fig. 2. We use a 5-bit *hop-count* field, a 6-bit *PID* field, and a 6-bit *XOR* field. The *hop-count* field records the number of hops from the router (that marks the *hop-count* field) to the victim. Five bits are allocated for this field because the vast majority of routes in the Internet have fewer than 32 hops [4]. The *PID* field of a given packet records the PID of the router's input interface port that processed the packet. In Subsection III.A, we discuss why six bits are allocated. The *XOR* field of a packet records the value obtained by taking the XOR (exclusive OR) of PID values. AD does not use the *XOR* field, but PAD uses it for distinguishing different attack paths.

A router interface can be set to either of two marking modes. A router interface is said to be in the Active Deterministic Marking Mode (ADMM) for V, if it processes every packet destined for V as follows: 1) sets the *hop-count* field to zero and 2) copies its PID to both the *PID* and the *XOR* fields.

A router interface is in the Passive Deterministic Marking Mode (PDMM) for V, if it processes every packet destined for V as follows: 1) increases the *hop-count* field by one and 2) computes the bit-by-bit XOR value of its *PID* and the *XOR* field value and writes the result back to the *XOR* field.

When the IDS installed at the victim detects an attack, the AD process is triggered. The victim begins the process by sending a "Diagnose-All-Interfaces" (DAI) request to its immediate edge router. This request packet should have the *TTL* field set to 255 so that the receiving router will be able to verify that the packet came from a host one hop away. Refer to Fig. 1 for an example. The path $C1$-G-E-C-A is assumed to be the attack path of a single zombie, $C1$. The execution of AD is summarized in the following steps:

(1) First V sends a DAI request to router A. Reacting to the request, A sets the marking mode of all of its input interfaces to ADMM for V. Also, A sends a status packet to V to notify V that it has begun marking packets. Now, every packet arriving at V has its *hop-count* field marked as zero and its *PID* field marked as the PID value of A's interface that processed the packet. If the IDS is able to identify attack packets, then V is able to identify the input interface of A that processed the malicious packets by observing their markings. Since the packets are marked deterministically, every malicious packet received by the victim is marked.

(2) After V has observed that the attack traffic is coming from the interface of A with PID 42, it sends a "Diagnose-Individual-Interface" 42 (DII-42) request to A. The DII request is accepted because A just received a DAI request for V. Otherwise this request should be ignored. A executes two steps to respond to the DII request. First, it sets the interface of PID 42 to PDMM while leaving the other interfaces unchanged. Second, A sends a DAI request to the neighbor router connected via interface 42, namely router C in our running example. Again, this request packet's *TTL* field is set to 255.

(3) Router C executes the same steps that A did when it received the same request from V. Again, a status packet is sent to V. This packet notifies V of C's IP address and the fact that C is marking packets. Now, packets received by V with their *hop-count* fields marked as one are all coming from C. Using this information, V is able to identify the input interface of C that is processing the attack traffic, which is interface 24 in the example.

(4) After identifying the interface in the previous step, V sends to C a DII-24 request. C executes the same procedures executed by A in Step 2.

(5) The procedures described in the previous steps are iterated until router G receives a DII-4 request. After this request is received, G begins to filter all the packets destined for V that is processed by interface with PID 4. If interface 4 is connected to multiple hosts via a broadcast link-layer channel, then G refers to the virtual PID table and filters frames based on the source MAC address. Router G also sends a status packet to V to notify V that it has started the filtering process.

The first four steps constitute the traceback phase, and the last step is the filtering phase. The five steps constitute a round of diagnosis. Using the steps described above, AD can effectively filter attack packets generated by $C1$. Note that in the last step, we have ignored the possibility of spoofed MAC addresses. If spoofed MAC addresses are suspected, then router G would need to filter all packets processed by interface 4.

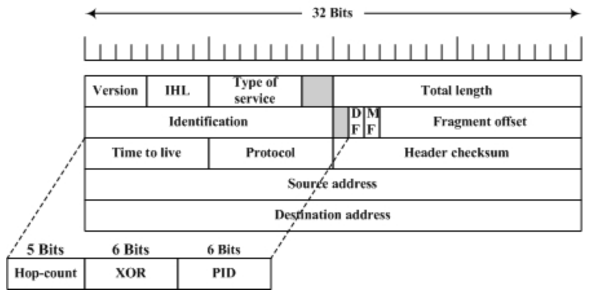

Fig. 2. The marking fields for AD and PAD.

Under the following assumptions, one can show that AD incurs no false negatives or false positives: (1) the zombies are constantly sending attack packets during the AD process; (2) the IDS installed at the victim can correctly identify malicious packets; and (3) the MAC addresses are not spoofed.

The legitimacy and integrity of the DII and DAI control messages need to be assured in order for AD to work properly. To prevent the tampering of these control messages, the messages are forwarded hop by hop from one router to its neighbor router, and the *TTL* field of the corresponding packets is set to 255. However, this mechanism is not sufficient to ensure the integrity of all control messages. In particular, an adversary located in the same subnet as the victim may send a forged DAI request to the victim's edge router, which cannot be detected by checking the *TTL* field nor can it be prevented using ingress/egress filters. In this case, the DAI message needs to be authenticated, thus requiring the victim and its edge router to share a secret (authentication) key. In practice, it is appropriate for ISPs to provide the AD service to their clients for a fee. If an ISP provides such a service, its clients can register for the AD service and acquire the authentication key during the registration process. We will discuss other deployment issues in Subsection III.D.

D. Parallel Attack Diagnosis

AD traces back and throttles the traffic of one zombie at a time. After the traffic of one attack source has been throttled, the victim initiates the same process to throttle other attack traffic, and this process gets repeated. It is obvious that this technique is too slow to defend against a large-scale DDoS attack in which thousands of zombies generate attack traffic synchronously. In such a case, the victim may be inundated with millions of attack packets before AD brings about any noticeable effect. Therefore, we introduce a "parallelized" version of AD—Parallel Attack Diagnosis (PAD)—that can handle multiple attack paths simultaneously. The primary difference between PAD and AD is that in PAD when a node sends a DII request to a router, it can specify more than one PID. The router that receives the DII request changes the marking mode of the appropriate interfaces (which are specified by the PIDs) to PDMM and sends DAI requests to the upstream neighbor routers that are connected to those interfaces. If the router is an edge router connected to hosts, then it simply begins to filter attack packets at the appropriate interfaces.

To function properly, PAD needs to distinguish distinct attack paths that are being traced back simultaneously. To illustrate this point, we refer back to Fig. 1. Assume that there are two zombies, $C1$ and $C2$, attacking the victim together. After sending a DAI request to A, V observes that there are

TABLE I
USING THE *XOR* FIELD TO DIFFERENTIATE DISTINCT PATHS.

Hop-count:	*	PID	XOR
0	A	42 [101010]#	42 [101010]
	A	27 [011011]	27 [011011]
1	C	24 [011000]	50 [110010] ($\oplus$24 = 42)
	B	50 [110010]	41 [101001] ($\oplus$50 = 27)
2	E	08 [001000]	58 [111010] ($\oplus$08 = 50)
	D	08 [001000]	33 [100001] ($\oplus$08 = 41)
3	G	04 [000100]	62 [111110] ($\oplus$04 = 58)
	F	31 [011111]	62 [111110] ($\oplus$31 = 33)

* The farthest router from V that marks packets in ADMM.
# The binary string inside [...] represents the binary equivalent.

two interfaces, 42 and 27, receiving malicious packets. If V decides to "diagnose" the two interfaces in parallel, it sends a "DII-42, 27" request to A. Then, A will change the marking mode of the two interfaces to PDMM, and send DAI requests to both B and C. In response, B and C will set the marking mode of all their interfaces to ADMM. This scenario raises an important question: when parsing through packets that have the *hop-count* field set to one, how can V determine that interface 50 belongs to B and interface 24 belongs to C? This question can be answered by using the *XOR* field. Recall that the *XOR* field of a packet contains the result of taking the XOR of all PIDs whose corresponding interfaces marked that packet along an attack path. Hence,

$$\begin{aligned} & XOR(i) \oplus PID(i) \\ & = PID(0) \oplus ... \oplus PID(i) \oplus PID(i) \\ & = PID(0) \oplus PID(1) \oplus ... \oplus PID(i-1) \\ & = XOR(i-1), \end{aligned} \qquad (1)$$

where $XOR(i)$ and $PID(i)$ denote the bit strings of the respective marking fields of a packet at the instant that the packet is marked in ADMM i hops away from the victim, and $\oplus$ denotes the XOR operation. Using (1), the victim can group the PIDs that constitute an attack path. In the running example, PID 50 at hop 1 is grouped with PID 27 at hop 0 because $41 \oplus 50 = 27$, where 41 is the value of the *XOR* field. In the same manner, PID 24 at hop 1 can be grouped with PID 42 at hop 0. This process is repeated for every hop until a DII request is received by the edge routers, G and F. In response, the routers filter packets based on the last-hop PID specified in the DII message. Table I shows how the PIDs can be grouped to form an attack path. The shaded rows correspond to the attack path of $C1$, and the non-shaded rows correspond to the attack path of $C2$.

E. Analysis of False Positives in PAD

If PAD handles q zombies in a single round of diagnosis, then it is expected to be roughly q times faster than AD. However, diagnosing multiple zombies in parallel comes at a cost—traffic from a legitimate client may get throttled. The reason is that during the traceback phase, the *XOR* and *PID* fields of the packets coming from a legitimate client may coincide with those of the packets coming from a zombie. For example, a PID string corresponding to a legitimate client's path, P_{10}-P_{11}-P_{12}, can incur a false positive when the substring P_{10}-P_{11} is a substring of a zombie's path and there is another zombie's path P_{20}-P_{21}-P_{22} under diagnosis, which satisfies two conditions: (1) $P_{22} = P_{12}$ and (2) $P_{10} \oplus P_{11} = P_{20} \oplus P_{21}$. The first condition leads to a collision of the *PID* fields and the second condition results in a collision of the *XOR* fields. Generalizing the above argument, a false positive occurs when each link of a legitimate client's path (that is not shared by any attack paths) collides with a zombie's attack path. A collision means that the *XOR* and *PID* values of two different packets traveling two different paths are the same when the traceback process has arrived at a particular link. It can be shown that allocating PIDs randomly minimizes the chance of collision. Hence, we assume that each router assigns PID values to its interfaces randomly from 0 to 63. If we assume that attack traffic is being traced back to q zombies in a single round, then the probability that a link of a legitimate client's path (that is not on any zombie's path) collides with a zombie's path is:

$$P_C = \frac{\sum_{k=1}^{\min(q,2^{12})} k \begin{pmatrix} 2^{12} \\ k \end{pmatrix} \left(\frac{1}{2^{12}}\right)^q M(k)}{2^{12}}, \qquad (2)$$

where

$$M(k) = \begin{cases} 1 & (k = 1) \\ k^q - \sum_{j=1}^{k-1} \begin{pmatrix} k \\ j \end{pmatrix} M(j) & (2 \leq k \leq 4096). \end{cases}$$

In (2), P_C can also be understood as the expected ratio of the 2^{12} combinations that is covered by q independent random 12-bit numbers. If a legitimate client's path has m links that are not on any zombie's path, then its false positive probability is:

$$P_{FP} = P_C^m. \qquad (3)$$

The value given by (3) represents the false positive probability for a single round of diagnosis. Because q zombies are diagnosed per round, PAD executes a total of N_a/q rounds (assuming $q|N_a$), where N_a is the total number of zombies. Therefore, the probability of incurring a false positive is:

$$P_{FP-PAD} = 1 - (1 - P_{FP})^{\frac{N_a}{q}}. \qquad (4)$$

For example, if $N_a = 1000, q = 100$, and $m = 2$, we obtain $P_{FP} = 0.059\%$ and $P_{FP-PAD} = 0.59\%$. In reality, the value of m is not fixed and is hard to estimate. In Section IV, we will plot (3) and (4) as a function of q using simulation data.

III. Practical Considerations

A. Support for Highly Connected Routers

We have allocated six bits for a PID. This is enough for most of Internet routers. As the study of CAIDA's (Cooperative Association for Internet Data Analysis) Skitter project shows [11], 98.5% of Internet routers have fewer than 64 working interfaces. Another Internet topology study in [1] shows that this percentage is even higher. However, for the other routers with more than 64 working interfaces and the edge routers connecting more than 64 hosts in an edge network, six bits are apparently not enough. This problem can be solved by adding another six bits to a PID and enhance the packet marking procedure. In the enhanced procedure, a router acts as if it was two routers connected in serial. The router splits the PID into two 6-bit PID fragments and associates them with different hops. For example, if B in Fig. 1 uses 12-bit PIDs, it associates the six most significant bits with hop 1 and the six least significant bits with hop 2. In each hop diagnosis, B still only marks six bits, but two hop diagnoses reveal the complete 12-bit PID. As a result, D is considered to be located at hop 3 and F in hop 4. With the same principle, additional 6-bit fragments can be added to the PID at the expense of more hop diagnoses required. The above marking procedure performed by a highly connected router is transparent to the victim (i.e., the victim follows the same procedures for AD or PAD).

B. Security Properties

We have shown that the request messages used in AD and PAD are secure because the core routers only accept requests from neighboring routers and the edge routers are responsible for authenticating the original requests. Therefore, forgery of diagnosis requests is thwarted. Besides request message forgeries, other attacks are possible. For instance, an attacker may attempt to forge information in the marking fields of packets. Fortunately, such attacks—although they may be effective against probabilistic packet marking schemes [10]—are not effective against AD or PAD. Since AD and PAD use deterministic packet marking, a forged marking will be overwritten by intermediate routers. In a different type of attack, an adversary may attempt to circumvent AD or PAD by enabling the zombies to send packets intermittently so that a diagnosis process would halt at some intermediate hop. Such an attack would be less harmful since the victim is less likely to be inundated with attack packets that are sent intermittently. A possible solution to such an attack is to store state information on the progress of an ongoing diagnosis process so that an unfinished diagnosis can be continued later on without having to start a new diagnosis process. However, the problem of timing a retry to maximize the chance of marking the attack traffic remains an open problem.

C. Router Overhead

Although the packet marking procedure required by AD and PAD adds additional router overhead, it is well within the capability of conventional routers. For instance, *input debugging* [12]—the functionality that can determine the interface that processed a particular packet—is widely supported by today's routers. Furthermore, the routine tasks of looking up routing tables and updating packets' *TTL* and *Checksum* fields are not much different from the required operations of the proposed marking procedure. More importantly, since AD and PAD are reactive defense mechanisms, it is unlikely that a router will received a large number of simultaneous requests for packet marking. This ensures that a router will not be overburdened with packet marking tasks the vast majority of the time. To reduce router overhead, a router should set a *timeout period* for every packet marking request. If, after responding to a victim's initial diagnosis request, a router does not receive any further requests before the timeout period expires, then it exits the marking mode for that particular victim. This will enable the router to recycle the resources that have been allocated for AD or PAD-related operations requested by that victim.

D. Deployment Considerations

As we have seen, the effectiveness and security of AD and PAD largely depend on their wide deployment. Since the instantaneous wide deployment of a new scheme is not possible in most cases, considerations for gradual deployment must be given. We expect that the tier-1 ISPs (such as MCI, Sprint, etc., see page 35 of [5] for more detail) would have the strongest motivation to implement AD and PAD. By implementing the defense mechanisms, they are able to provide DDoS countermeasure services to lower tier ISPs, helping them filter malicious packets at the core network. Since the core network is closer to the attack sources than the victim, DDoS attacks can be mitigated more effectively. The lower tier ISPs can in turn provide the DDoS defense service to even lower tier ISPs, and those ISPs to end hosts. In the process, a mutual trust relationship can be built between neighboring ISPs so that a diagnosis request sent from one ISP's network is accepted by another ISP's network. This system of trust may also be realized by utilizing secure signaling systems such as the Real-time Inter-network Defense (RID) [8]. In this way, AD and PAD can be gradually spread across the Internet.

IV. Simulation and Results

To evaluate the performance of AD and PAD, we have simulated them on realistic network topologies. Specifically, we used the Internet map created from CAIDA's Skitter project [11]. Based on the provided undirected link data, we first chose a router with a degree of two as the victim's edge router, and then randomly chose many distinct routes originating from it.

A study in [4] has revealed that a typical hop count distribution from clients to a server can be approximated using a Gaussian distribution, with a mean of 16.5 and a standard deviation of 4. In our simulations, we use this distribution for generating route hop counts. The upper and lower bounds of the hop counts are set to 32 and 1, respectively.

Figs. 3 and 4 show the simulation results for PAD. Fig. 3 shows the false positive ratio in a single diagnosis round, P_{FP}, as a function of q. We fixed the number of total clients at 10,000 and increased the number of zombies from 1 to 4,100. P_{FP} is computed as the number of the legitimate clients that are throttled incorrectly divided by the total number of legitimate clients. Fig. 4 shows P_{FP-PAD} as a function of q. In this simulation, the number of zombies N_a is fixed at 8,000. Every data point in Figs. 3 and 4 is the average of three independent simulation results. These results indicate that the false positive ratio incurred by PAD remains very low even when q is relatively large. For instance, when q = 2000, the false positive ratio for each diagnosis round is only 1%.

To further investigate the tradeoff relation between the attack mitigation delay and the false positive ratio, we conducted a different series of simulations. In these simulations, we measure the time interval from the beginning of a synchronized DDoS attack to the moment that all the packets generated by the zombies are throttled. We denote this time as T. In the calculation of T, the small processing delay incurred in identifying distinct attack paths is ignored. In addition, we assume that during the traceback process, a delay of 32ms is incurred per hop and the IDS installed at the victim can identify malicious packets and read their marking field values instantly without any delay. Although these assumptions are by no means realistic, they allow us to obtain qualitatively meaningful results that provide some insight on the tradeoff relation between the attack mitigation delay and the false positive ratio. Results are shown in Table II. Each entry in the table shows the average of 10 independent experiments. In each experiment, 2000 zombies were randomly chosen from 5000 clients in the Skitter map. The value of q was set as 200, 500, and 1000 respectively. The table clearly reveals the tradeoff relation that exists between the false positive ratio and the attack mitigation delay: AD with the longest attack mitigation delay has a zero false positive ratio, while PAD with the shortest attack mitigation delay (q = 1000) has the highest false positive ratio. In practice, a victim can adjust the value of q to achieve the appropriate balance between the two performance measures.

V. Related Work

There have been many attack mitigation schemes proposed in the literature. The schemes presented in [7], [13], [14], [15], [16] all place the attack detection module and the packet filtering module at the same location. According to the

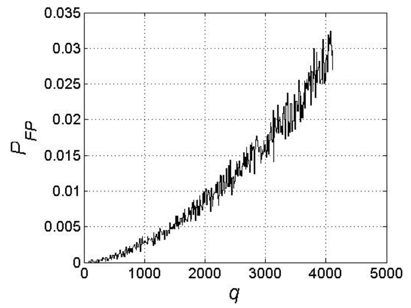

Fig. 3. P_{FP} vs. q in Skitter Internet map.

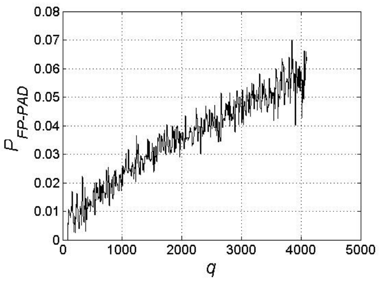

Fig. 4. P_{FP-PAD} vs. q in Skitter Internet map (N_a = 8,000).

TABLE II
THE FALSE POSITIVE RATIO VS. ATTACK MITIGATION DELAY (2000 MALICIOUS CLIENTS OUT OF TOTALLY 5000 CLIENTS, SKITTER MAP).

	P_{FP-PAD}	T (sec.)
AD	0%	11644
PAD (q = 200)	0.161%	171.14
PAD (q = 500)	0.415%	70.36
PAD (q = 1000)	0.590%	35.07

location of the modules, those schemes can be divided into two categories: victim-based filtering and source-based filtering.

Schemes proposed in [13], [14], [15], [16] use victim-based filtering. For instance, NetBouncer [14] maintains a legitimate client's list at the victim end. It mitigates attacks by prioritizing the packets sent from the clients in the list. The filtering schemes in [13], [15], [16] rely on packet marking to filter attack traffic. The markings of the received packets are used by the victim to distinguish legitimate packets from attack packets.

D-WARD [7] is representative of a source-based filtering scheme. It is designed to be installed at the edge router of a source network. It prevents the hosts from launching attacks by suppressing any host that sends a much heavier volume of traffic than it receives.

The schemes proposed in [6], [9], [17] place the attack detection module near the victim and execute packet filtering close to the attack sources.

In [17], Zhang and Dasgupta propose to build protection at the transport layer of the Internet. The border routers in Autonomous Systems (AS) are upgraded to "hardened routers" so that they can detect attacks and signal upstream hardened routers to filter attack traffic. The DDoS defense COSSACK [9] installs a system called a *watchdog* in every edge router. When the victim-end watchdog detects an attack, it multicasts attack notification to the watchdogs at the source networks so that they can suppress the attack close to the source.

Unlike the above two schemes, Pushback [6] adopts the hop-by-hop transmission of control messages. The authenticity of control message packets is ensured using the *TTL* field. This scheme uses the "aggregate-based congestion control" (ACC) module for local congestion detection and high-bandwidth traffic throttling. If a router cannot adequately throttle an aggregate by itself, the Pushback module requests the router's upstream routers to rate-limit the aggregate together.

VI. CONCLUSION

We proposed novel approaches for DDoS attack mitigation called AD and PAD. AD integrates the concepts of Pushback and packet marking. AD's framework supports the placement of the attack detection module at the victim end and the placement of the filtering module close to the attack sources. The packet marking procedure specified by AD provides the upstream routers (close to the attack sources) with an attack signature which enables them to distinguish the paths of legitimate traffic and attack traffic. Because AD cannot effectively thwart large-scale DDoS attacks, we proposed an extension to AD called PAD. Unlike AD, PAD is capable of tracing back and mitigating attack traffic from multiple zombies simultaneously. Our analysis and simulation results indicate that AD and PAD are secure, affordable, and incurring small false false positives.

REFERENCES

[1] B. Al-Duwairi and T.E. Daniels, "Topology based packet marking," *The 13th International Conference on Computer Communications and Networks (ICCCN)*, Oct. 2004, pp. 146-151.

[2] R.K.C. Chang, "Defending against Flooding-based Distributed Denial-of-service Attacks: a Tutorial," *IEEE Communications Magazine*, Vol. 40(10), Oct. 2002, pp. 42-51.

[3] P. Ferguson and D. Senie, *Network Ingress Filtering: Defeating Denial of Service Attacks Which Employ IP Source Address Spoofing*, RFC 2267, Jan. 1998.

[4] C. Jin, H. Wang, and K. G. Shin, "Hop-Count Filtering: An Effective Defense against Spoofed DoS Traffic," *The Tenth ACM International Conference on Computer and Communications Security (CCS)*, Oct. 2003, pp. 30-41.

[5] J. Kurose and K. Ross, *Computer Networking: a Top-down Approach Featuring the Internet, Third Edition*, Addison Wesley, 2004.

[6] R. Mahajan, S. M. Bellovin, S. Floyd, J. Ioannidis, V. Paxson, and S. Shenker, "Controlling High Bandwidth Aggregates in the Network," *Computer Communications Review*, 32(3), Jul. 2002, pp. 62-73.

[7] J. Mirkovic, G. Prier, and P. Reiher, "Attacking DDoS at the Source," *The 10th IEEE International Conference on Network Protocols*, Nov. 2002, pp. 312-321.

[8] K. M. Moriarty, *Distributed Denial of Service Incident Handling: Real-Time Inter-Network Defense*, Internet Draft, draft-moriarty-ddos-rid-06.txt, Feb. 2004.

[9] C. Papadopoulos, R. Lindell, J. Mehringer, A. Hussain, and R. Govindan, "Cossack: Coordinated Suppression of Simultaneous Attacks," *DARPA Information Survivability Conference and Exposition*, Vol. 1, Apr. 2003, pp. 2-13.

[10] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Practical Network Support for IP Traceback," *The conference on Applications, Technologies, Architectures, and Protocols for Computer Communication (SIGCOMM)*, Aug. 2000, pp.295-306.

[11] CAIDA's Skitter project web page, Available at: http://www.caida.org/tools/measurement/skitter/index.xml.

[12] R. Stone, "CenterTrack: An IP Overlay Network for Tracking DoS Floods," *The 2000 USENIX Security Symposium*, July 2000, pp. 199-212.

[13] M. Sung, J. Xu, "IP Traceback-based Intelligent Packet Filtering: a Novel Technique for Defending against Internet DDoS Attacks," *IEEE Transactions on Parallel and Distributed Systems*, Vol.14(9), Sep. 2003, pp. 861-872.

[14] R. Thomas, B. Mark, T. Johnson, and J. Croall, "NetBouncer: Client-legitimacy-based High-performance DDoS Filtering," In *DARPA Information Survivability Conference and Exposition*, Vol. 1, Apr. 2003, pp. 14-25.

[15] A. Yaar, A. Perrig, and D. Song, "Pi: a Path Identification Mechanism to Defend against DDoS Attacks," *The 2003 IEEE Symposium on Security and Privacy*, May 2003, pp. 93-107.

[16] A. Yaar, A. Perrig, and D. Song, "SIFF: a Stateless Internet Flow Filter to Mitigate DDoS Flooding Attacks," *The 2004 IEEE Symposium on Security and Privacy*, May 2004, pp. 130-143.

[17] S. Zhang and P. Dasgupta, "Denying Denial of Service Attacks: a Router Based Solution," *The 2003 International Conference on Internet Computing*, Jun. 2003.