

# Single Packet IP Traceback in AS-level Partial Deployment Scenario

Chao Gong<sup>†</sup>, Trinh Le<sup>\*</sup>, Turgay Korkmaz<sup>\*</sup>, Kamil Sarac<sup>†</sup>

<sup>\*</sup> Department of Computer Science, University of Texas at San Antonio

6900 North Loop 1604 West, San Antonio, TX 78249, USA

Email: {tle,korkmaz}@cs.utsa.edu, Tel: +1-210-458-4436

<sup>†</sup> Department of Computer Science, University of Texas at Dallas

2601 N Floyd Road, Richardson, TX 75080, USA

Email: {cxg010700,ksarac}@utdallas.edu, Tel: +1-972-883-2185

## Abstract

Tracing IP packets to their sources, known as IP traceback, is an important task in defending against IP spoofing and DoS attacks. Log-based IP traceback technique is to log packets at routers in the network and then determine the network paths which packets traversed using data extraction techniques. The biggest advantage of log-based IP traceback is the potential to trace a single packet. Tracing a single packet in the Internet using log-based IP traceback involves cooperation among all Autonomous Systems (AS) traversed by the packet. The single packet traceback process may not reach the packet origin if some AS on the forwarding path does not support IP traceback. IP traceback mechanisms are deployed within each AS independently. It is not reasonable to assume all ASes begin to support the same IP traceback mechanism in a short period of time. In this paper, we study the effectiveness of log-based IP traceback in tracing a single packet under the environment where not every AS supports log-based IP traceback. We propose a scheme to conduct the single packet traceback process in AS-level partial deployment scenario. We evaluate the performance of single packet IP traceback in AS-level partial deployment scenario based on our scheme through simulation.

**Keywords:** IP traceback, Single packet IP traceback, Denial-of-Service (DoS) defense.

# Single Packet IP Traceback in AS-level Partial Deployment Scenario

Chao Gong<sup>†</sup>, Trinh Le<sup>\*</sup>, Turgay Korkmaz<sup>\*</sup>, Kamil Sarac<sup>†</sup>

<sup>\*</sup> Department of Computer Science, University of Texas at San Antonio, USA

<sup>†</sup> Department of Computer Science, University of Texas at Dallas, USA

**Abstract**—Tracing IP packets to their sources, known as IP traceback, is an important task in defending against IP spoofing and DoS attacks. Log-based IP traceback technique is to log packets at routers in the network and then determine the network paths which packets traversed using data extraction techniques. The biggest advantage of log-based IP traceback is the potential to trace a single packet. Tracing a single packet in the Internet using log-based IP traceback involves cooperation among all Autonomous Systems (AS) traversed by the packet. The single packet traceback process may not reach the packet origin if some AS on the forwarding path does not support IP traceback. IP traceback mechanisms are deployed within each AS independently. It is not reasonable to assume all ASes begin to support the same IP traceback mechanism in a short period of time. In this paper, we study the effectiveness of log-based IP traceback in tracing a single packet under the environment where not every AS supports log-based IP traceback. We propose a scheme to conduct the single packet traceback process in AS-level partial deployment scenario. We evaluate the performance of single packet IP traceback in AS-level partial deployment scenario based on our scheme through simulation.

## I. INTRODUCTION

The goal of IP traceback is to trace the path of an IP packet to its origin. The most important usage of IP traceback is to deal with certain denial-of-service (DoS) attacks, where the source IP address is spoofed by attackers. Identifying the sources of attack packets is the first step in making attackers accountable. In addition, figuring out the network path which the attack traffic follows can improve the efficacy of defense measures such as packet filtering as they can be applied further from the victim and closer to the source.

Based on the vulnerability that is exploited, DoS attacks can be classified as *brute-force* and *semantic* attacks. Brute-force attacks work by flooding some limited resource with large amounts of traffic, thereby preventing legitimate users from accessing that resource. Semantic attacks exploit some specific feature or implementation bug of operating systems or routers to disable the services with one single or a few packets. An IP traceback approach that can track an individual packet is a must for defending against semantic DoS attacks.

Log-based IP traceback technique [1] is able to trace a single IP packet. The basic idea is to log packets at routers in the network and then derive the network path using some kind of extraction techniques. During the process of tracing an IP packet, routers are queried to glean the path information, starting from the packet destination back to the source. Historically, packet logging was thought impractical because of enormous storage space for packet logs. Snoeren *et al.* [2] developed Source Path Isolation Engine (SPIE), a log-based IP traceback architecture wherein routers record packet digests in

a space-efficient data structure, bloom filter [3]. SPIE reduces the storage overhead of packet logging significantly. Moreover, improvements on SPIE have been proposed to further reduce the storage overhead [4], [5], [6]. Tracing a single IP packet using log-based IP traceback technique is becoming feasible in practice.

Any modification of Internet routers will not be deployed within all Autonomous Systems (AS) simultaneously, and can not be finished in a short period throughout the Internet. It is unrealistic to assume all ASes begin to support a specific IP traceback mechanism in a short period of time. In log-based IP traceback, tracing a single packet involves interactive cooperation with all ASes traversed by the packet en route from source to destination. The traceback process may halt prematurely because of lack of support in some ASes.

In this paper, we first propose a scheme to conduct single packet traceback under the environment where not every AS supports log-based IP traceback. With our scheme, the traceback process still has chance to proceed when encountering ASes not supporting log-based IP traceback. We then explore the effectiveness of the proposed scheme using extensive simulations. The results show that log-based IP traceback still functions when it is partially deployed in the Internet. More importantly, these results allow us to characterize the trade-offs between the success rate in tracing individual packets and protocol overheads (or operation costs).

The remainder of this paper is organized as follows. Section II puts our work in the context of related work. Section III describes our scheme conducting single packet traceback in AS-level partial deployment scenario. Section IV evaluates the performance of single packet IP traceback in AS-level partial deployment scenario based on our scheme. Finally, we conclude our work with a brief look at future work in Section V.

## II. RELATED WORK AND BACKGROUND

In this section, we first present an overview of the related work in IP traceback, then we take a close look at SPIE, a feasible implementation of log-based IP traceback.

### A. Previous Work

Motivated by the increasing frequency of DoS attacks and demand for Internet forensic analysis, many IP traceback approaches have been proposed.

Bellovin [7] introduced ICMP traceback. The principle idea is that routers select packets with low probability (1/20,000), and then send ICMP packets including the contents of sampled

packets and local path information to the same destinations as the selected packets.

Burch *et al.* [8] proposed controlled flooding. In controlled flooding, victims reconstruct attack paths by selectively flooding network links and monitoring the change of incoming traffic. In the same paper, the authors mentioned the idea of IP traceback based on marking packets, either probabilistically or deterministically, with the identification information (e.g., IP addresses) of routers they pass through.

Usually, the mark overloads a rarely used field in IP packet header, i.e., 16-bit IP identification field. Because the marking space in IP header is too small to record the entire path, routers probabilistically decide to mark packets so that each marked packet carries only partial path information. The network path can be reconstructed by combining a modest number of packets containing mark. This approach is known as IP traceback based on probabilistic packet marking (PPM) [9], [10]. The PPM approach can only trace the traffic composed of a number of packets because of its probabilistic nature.

IP traceback based on deterministic packet marking either releases the constraint on the marking space [11] or carries the assumption that routers at specific locations (e.g., ingress routers) are traceback enabled [12]. The major application of deterministic packet marking is to create a common path signature for all packets traversing the same network route, for the purpose of filtering out attack traffic at the victim [13], [14], [15].

Sager [1] suggested the idea of tracing packets by logging packets at routers in the network and then using extraction techniques to derive paths which packets traversed. This approach has the potential to track a single IP packet. In order to decrease the required storage space, logging should be done in a space-efficient manner. Source Path Isolation Engine (SPIE) [2] is an architecture implementing log-based IP traceback. In SPIE, routers store packet digests, instead of packets themselves, in a space-efficient data structure, bloom filter [3]. In this way, the storage overhead is reduced significantly (down to 0.5% of the total link capacity per unit time). At routers with high speed links, the storage requirement of 0.5% of the total link capacity per unit time may be still prohibitive. Enhancements on SPIE have been proposed to further reduce the storage overhead [4], [5], [6].

### B. Single Packet IP Traceback

Tracing a single packet with log-based IP traceback technique is becoming feasible along with the appearance of SPIE and its enhanced versions. In the rest of the paper, we use “SPIE” and “log-based IP traceback” interchangeably.

SPIE is deployed within an AS. The SPIE system in an AS consists of three major architectural components: data generation agent (DGA), SPIE collection and reduction agent (SCAR), and SPIE traceback manager (STM).

The DGA is associated with SPIE-enhanced router. Its functionality is to record packet digests. For each arriving packet, the DGA uses the first 24 invariant bytes of the packet (20-byte IP header with 4 bytes masked out plus the first 8 bytes of

payload) as input to the digesting function. The 32-bit packet digest is stored into a time-stamped digest table which is realized with bloom filter. The digest table is paged out before it becomes saturated, preventing unacceptable false-positive rates. If a packet undergoes transformation at a router, the DGA at that router records related transformation information into the *transform lookup table* [2]. The SCAR is responsible for a particular region of AS network. Upon request, the SCAR produces an attack path in its region by querying DGAs in the region. The STM controls the whole SPIE system. When the STM receives a request for tracing an IP packet, it dispatches the request to SCARs and grafts attack paths from SCARs together to form a complete attack path through its AS.

One AS’s STM must dispatch queries to adjacent ASes’ STMs in order to complete an inter-AS traceback process. The traceback process can not proceed if the adjacent AS which the attack path traverses does not support SPIE.

SPIE and its enhanced versions do not touch the issue on how to conduct single packet traceback in the context where log-based IP traceback is deployed in only a subset of ASes in the Internet. In this paper, we propose a scheme to conduct single packet traceback and, based on that scheme, evaluate the performance of single packet IP traceback in AS-level partial deployment scenario.

## III. SCHEME TO CONDUCT SINGLE PACKET IP TRACEBACK IN AS-LEVEL

We propose a scheme to conduct single packet IP traceback in AS-level partial deployment scenario. First, a STM dispatches queries to STMs of the adjacent ASes. If the adjacent AS which the attack path traverses does not support SPIE, the STM will not get any positive feedback. In this case, the STM will dispatch queries to remote (more than 1-hop away) ASes’ STMs. In this way, the traceback process still has chance to proceed when encountering ASes not supporting SPIE.

### A. Assumptions

Besides the assumptions listed in [2], there are some more assumptions that motivate and constrain our design:

- 1) ASes in the Internet may or may not deploy SPIE.
- 2) If an AS deploys SPIE, the STM of that AS is able to produce the complete attack path through the AS upon request. That is possible even if the DGA is not deployed at each router in the AS [2].
- 3) If an AS does not deploy SPIE, other ASes cannot know any information about packets’ paths in that AS.
- 4) Packet transformation does not occur in the AS not supporting SPIE. Packet transformation is unlikely to occur in reality. A recent study by CAIDA on wide-area traffic patterns shows that less than 3% of IP traffic undergoes transformation [16].
- 5) Packet digests are cached at routers for a period which is long enough for a traceback process to finish.

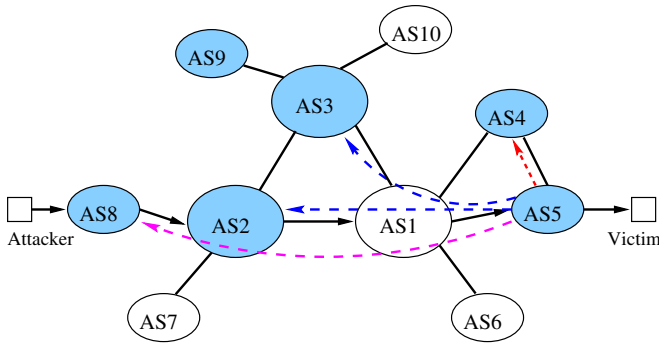


Fig. 1. Traceback in AS-level partial deployment scenario. AS2, AS3, AS4, AS5, AS8, and AS9 deploy SPIE while the other ASes do not. The STM of each AS dispatches queries by two levels. The solid arrows in the figure represent the attack path, and the dashed arrows represent traceback queries.

### B. Design

All SPIE-deployed ASes exchange SPIE deployment information with each other. We propose to utilize the BGP protocol as the vehicle to distribute the SPIE deployment information. SPIE-deployed ASes advertise their support for SPIE in a BGP attribute in the network route advertisement. Hence, each AS is able to know which ASes deploy SPIE, how many hops (in AS-level) they are far away, and their respective STMs. The STM of each AS keeps track of those information.

The STM of each AS is able to dispatch queries to other ASes' STMs *level-by-level*. It first sends queries to its level-1 SPIE-deployed AS neighbors (adjacent ASes, 1-hop away from the current AS). If the attack path cannot be reconstructed, it sends queries to its level-2 SPIE-deployed AS neighbors (2-hop away from the current AS), and so on. This process continues until the STM gets the origin of attack packet or it cannot trace anymore. The number of the levels to which the STM can dispatch queries depends on the implementation of each AS. We will talk about it later in Section III-C.

Our scheme is illustrated in Figure 1. An attack packet from the attacker in AS8 to the victim in AS5 follows the AS-level path AS8-AS2-AS1-AS5. Once the victim in AS5 has detected the attack, it issues traceback request to the STM of AS5. After reconstructing the attack path in AS5, the STM of AS5 sends queries to its level-1 SPIE-deployed AS neighbors, namely AS4. AS4 returns *negative reply message*, indicating that the attack packet did not traverse AS4. After receiving the negative reply from AS4, the STM of AS5 then sends queries to its level-2 SPIE-deployed AS neighbors, namely AS2 and AS3. Once AS2 has identified that the attack packet had traversed through it, AS2 returns *promising reply message* to AS5. The promising reply message includes information on the attack path in AS2 and all the SPIE-deployed AS neighbors of AS2 within 2 hops away (AS2 dispatches queries by 2 levels). They are, AS3 and AS8, which are the level-1 SPIE-deployed AS neighbors of AS2, plus AS9, AS4, and AS5, which are level-2 SPIE-deployed AS neighbors. Therefore, AS5 knows that the attack packet passed through AS2. Because AS5 has contacted AS3, AS5 continues the traceback process by sending queries

```

1 /* N is the furthest level to which the STM can dispatch queries */
2 FOR each traceback request DO
3   IF the request is from a STM in other ASes THEN
4     /* current AS may be an intermediate AS in the attack path */
5     process the traceback in its own AS
6     IF the attack path traverses its AS THEN
7       return promising reply message
8     ELSE IF the attack path ends within its AS THEN
9       return positive reply message
10    ELSE
11      return negative reply message
12    END IF
13  END IF
14  IF the request is from a victim within its AS THEN
15    /* the current AS is the destination AS of the attack path */
16    process the traceback in its own AS
17    IF the attack path traverses its AS THEN
18      cache the attack path in current AS
19      DO dispatch queries to SPIE-deployed neighbors level-by-level
20        IF receive promising reply message THEN
21          cache the returned attack path
22          stop current round of level-by-level queries
23          start new round of level-by-level queries
24        END IF
25        IF receive positive reply message THEN
26          graft all the cached and returned attack paths
27          return the entire attack path to the victim
28        END IF
29      UNTIL reach level-N or cannot proceed anymore
30      graft all the cached attack paths
31      return the entire attack path to the victim
32    ELSE IF the attack path ends within its AS THEN
33      return the entire attack path to the victim
34    ELSE
35      inform the victim the traceback process failed
36    END IF
37  END IF
38 END FOR

```

Fig. 2. Traceback procedure at STM.

only to AS8, the level-1 SPIE-deployed AS neighbor of AS2. After processing the query in its network, AS8 finds that the source of the attack packet is within its network and then returns *positive reply message* to AS5. The positive reply message includes the attack path in AS8 and indicates the end of traceback process. Finally the STM of AS5 grafts the attack path in AS5 and the attack paths returned from AS2 and AS8 to get the entire attack path.

Figure 2 shows the pseudocode of our scheme to conduct single packet IP traceback.

### C. Discussion

In the implementation of our scheme, one important factor is the trade-off between the traceback efficacy and the bandwidth overhead. The higher the number of levels to which each AS can dispatch queries in traceback process, the higher the probability to reach the origin of attack packet, especially when the deployment ratio of SPIE in the network is rather low. However, higher number of levels to dispatch queries results in more query messages being transmitted among ASes.

In reality, packets may undergo transformation when passing through ASes not supporting SPIE with very low probability. Hence, increasing the level to dispatch queries in the traceback process may not guarantee better performance.

Since STMs can get the SPIE deployment information of remote ASes from nearby ASes, STMs do not need to maintain the SPIE deployment information for the ASes beyond the furthest level to which they can dispatch queries. So the maintenance of SPIE deployment information at STMs is manageable. During a traceback process, intermediate STMs do not need to keep track of the status of ongoing traceback processes. Hence, our scheme is scalable in terms of the additional overhead imposed on STMs.

#### IV. PERFORMANCE EVALUATION

Simulation is conducted to evaluate effectiveness of log-based IP traceback for tracing a single packet in AS-level partial deployment scenario. Our goal is to investigate whether we can trace a single packet back to its origin under the environment where not every AS supports log-based IP traceback. Hence, our simulation ignores the false positive rate (FPR) in the traceback process. The metrics used to evaluate the performance are traceback success rate and the average number of queries issued by STMs. A single packet traceback process is successful if the origin of the packet is figured out. During a traceback process across ASes, STMs must dispatch queries to other ASes' STMs.

##### A. Simulation Setup

We considered various topologies; but, due to space limitations, we report the results for the following network topologies. The first one is a network topology modified from ANSNET [17] by inserting additional links, which is shown in Figure 3. The second one is a 3040-node network topology created by INET, an AS-level Internet topology generator [18]. The last one is a  $30 \times 30$  mesh topology. In simulation, we use  $30 \times 30$  mesh network topology as a special case to see what happens in a regular graph. Each node in topology represents an AS. We omit the intra-domain IP traceback process inside each AS. Each AS can be viewed as a black box with the ability to answer queries with 100% accuracy.

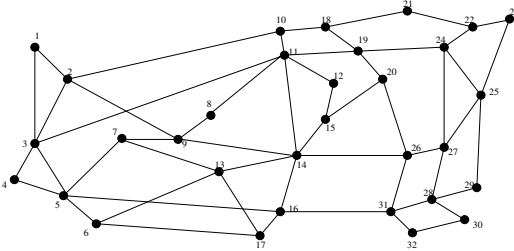


Fig. 3. Modified ANSNET network topology.

For each topology, we randomly choose 10%, 20%, ..., 100% ASes in the topology to set them to support log-based IP traceback, respectively. For each deployment ratio, we randomly choose 10000 (attacker, victim) pairs. The attacker is one AS chosen from all ASes in the topology, and the victim is a different AS chosen from all ASes supporting log-based IP traceback. For each (attacker, victim) pair, we compute the

shortest path from the attacker to the victim and set that path as the attack path.

In simulation, each STM can dispatch queries to the same number of levels. We set the number of traceback levels to be 1, 2, 4, and 8, respectively. For each setting, we execute simulation for all (attacker, victim) pairs and then compute the success rate of traceback processes and the average number of queries. When the number of traceback levels is 1, the traceback process is actually a “traditional” traceback process without using our scheme.

##### B. Simulation Results

Figure 4 shows the results of traceback success rate in the simulations. We see some common trends in all three network topologies. The more ASes to support IP traceback in the network, the higher chance to discover attackers. It seems that unless at least 40% of ASes deploy IP traceback, there is no significant chance to detect attackers. From other perspective, we need around 70% deployment in order to have 50% chance of detecting an attacker. Increasing the number of traceback levels from a small value to a medium value (e.g., from 1 to 2) results in big increments in the success rate. However, that is not true when increasing the number of traceback levels from a medium value to a large value (e.g., from 4 to 8). In the modified ANSNET topology and INET topology, 8-level traceback results in insignificant or no increments in the success rate compared to 4-level traceback. We have the same observation in  $30 \times 30$  mesh network when more than 80% of ASes deploys IP traceback. We conclude that we do not need to set the number of traceback levels to be very large in order to get an acceptable traceback success rate.

Figure 5 shows the average number of queries issued during traceback process. We see that the higher the number of traceback levels, the more the average number of queries. In the modified ANSNET topology and INET topology, the difference between 8-level traceback and 4-level traceback is small, while that difference is large in  $30 \times 30$  mesh network. We think the network topology, particularly node degree, affects that. When the number of traceback levels is 1, the average number of queries increases as the deployment ratio rises. When the number of traceback levels is larger (e.g., 4 or 8), the trend is different. Initially, the average number of queries increases as the deployment ratio grows up; it reaches the highest point when the deployment ratio is in the range of 50% and 70%; then it starts decreasing. Our explanation for this behavior is that the traceback process often terminates in the first few steps when the deployment ratio is low, and often proceeds without need to dispatch queries to remote ASes when the deployment ratio is high.

#### V. CONCLUSION AND FUTURE WORK

Partial deployment is a critical issue in any proposal that involves adding new functions to the Internet at the network layer. In this paper, we have investigated the effectiveness of single packet traceback under partial deployment of log-based IP traceback mechanism at AS level. Accordingly, we

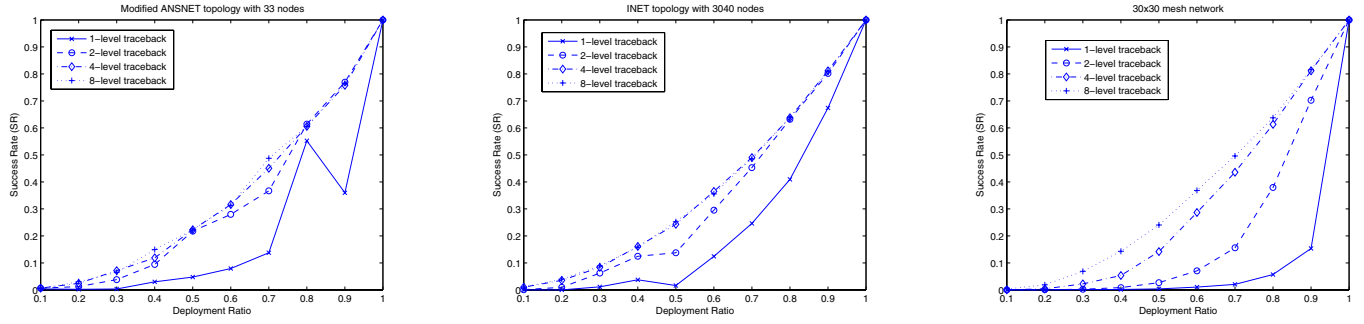


Fig. 4. Traceback success rate.

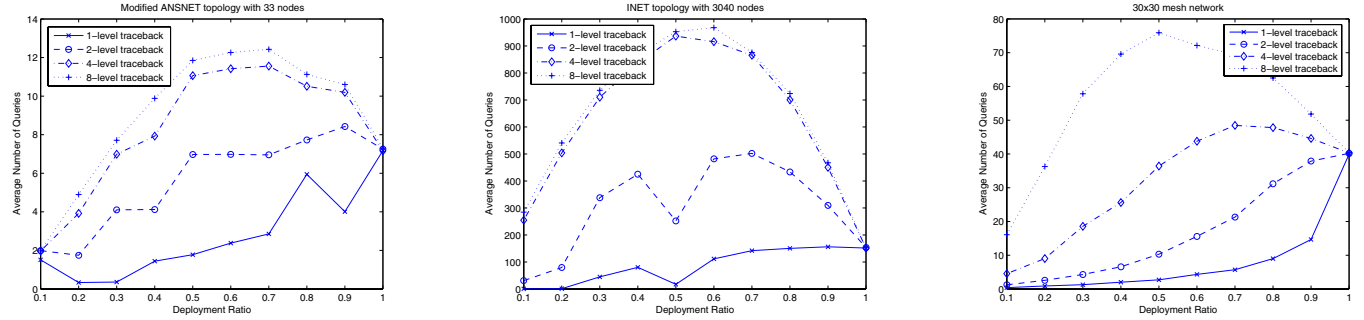


Fig. 5. Average number of queries.

have proposed a scheme and explored its performance in AS-level partial deployment scenario. Using extensive simulation, we have shown that log-based IP traceback still functions in AS-level partial deployment scenario. We have explored the relationship between the level of log-based IP traceback scheme deployed in the network and the success rate of finding out attackers. We have also characterized the trade-offs between the success rate in tracing individual packets and protocol overheads.

In our current work, we have ignored the false positive rate and packet transformation. In the future work, we plan to consider these issues and evaluate their impact on the success rate in tracing a single packet under AS-level partial deployment scenario. In addition, we have not considered partially identified attack paths in the simulations. Given that the information on partial attack paths is valuable for defense measures (e.g., packet filtering), we also plan to further investigate the rate and depth of partially identified attack paths.

#### ACKNOWLEDGMENTS

We would like to thank the anonymous reviewers for their insightful comments.

#### REFERENCES

- [1] G. Sager, "Security fun with OCxmon and cflowd," in *Internet 2 working group meeting*, November 1998.
- [2] A. Snoeren et al., "Single-packet IP traceback," *IEEE/ACM Transactions on Networking*, vol. 10, no. 6, pp. 721–734, December 2002.
- [3] B. Bloom, "Space/time trade-offs in hash coding with allowable errors," *Communications of ACM*, vol. 13, no. 7, pp. 422–426, July 1970.
- [4] T. Lee, W. Wu, and W. Huang, "Scalable packet digesting schemes for IP traceback," in *Proc. of IEEE International Conference on Communications*, Paris, France, June 2004.
- [5] J. Li, M. Sung, J. Xu, L. Li, and Q. Zhao, "Large-scale IP traceback in high-speed Internet: Practical techniques and theoretical foundation," in *Proc. of IEEE Symposium on Security and Privacy*, Oakland, USA, May 2004.
- [6] C. Gong and K. Sarac, "IP traceback based on packet marking and logging," in *Proc. of IEEE International Conference on Communications*, Seoul, Korea, May 2005.
- [7] S. Bellovin, "ICMP traceback messages." Internet draft: Draft-bellovin-itrace-00.txt, March 2000.
- [8] H. Burch and B. Cheswick, "Tracing anonymous packets to their approximate source," in *Proc. of the 14th USENIX Systems Administration Conference*, New Orleans, USA, December 2000.
- [9] S. Savage, D. Wetherall, A. Karlin, and T. Anderson, "Network support for IP traceback," *IEEE/ACM Transactions on Networking*, vol. 9, no. 3, pp. 226–237, June 2001.
- [10] A. Yaar, A. Perrig, and D. Song, "FIT: Fast Internet traceback," in *Proc. of IEEE INFOCOM*, Miami, USA, March 2005.
- [11] B. Al-Duwairi and T. Daniels, "Topology based packet marking," in *Proc. of International Conference on Computer Communications and Networks*, Chicago, USA, October 2004.
- [12] A. Belenky and N. Ansari, "IP traceback with deterministic packet marking," *IEEE Communications Letters*, vol. 7, no. 4, pp. 162–164, April 2003.
- [13] A. Yaar, A. Perrig, and D. Song, "Pi: A path identification mechanism to defend against DDoS attacks," in *Proc. of IEEE Symposium on Security and Privacy*, Oakland, USA, May 2003.
- [14] Y. Kim, J. Jo, and F. Merat, "Defeating distributed denial-of-service attack with deterministic bit marking," in *Proc. of IEEE GLOBECOM*, San Francisco, USA, December 2003.
- [15] Z. Gao, N. Ansari, and K. Anantharam, "A new marking scheme to defend against distributed denial of service attacks," in *Proc. of IEEE GLOBECOM*, Dallas, USA, November 2004.
- [16] S. McCreary and K. Claffy, "Trends in wide area IP traffic patterns: A view from Ames Internet exchange," in *ITC Specialist Seminar on IP Traffic Modeling, Measurement and Management*, Monterey, USA, September 2000.
- [17] D. E. Comer, *Internetworking with TCP/IP*, vol. I. Prentice Hall, Inc., third ed., 1995.
- [18] INET Topology Generator. <http://topology.eecs.umich.edu/inet/>.