Single-Packet IP Traceback

Alex C. Snoeren, *Student Member, IEEE*, Craig Partridge, *Fellow, IEEE*, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, *Member, IEEE*, Beverly Schwartz, Stephen T. Kent, and W. Timothy Strayer, *Senior Member, IEEE*

***Abstract*—The design of the IP protocol makes it difficult to reliably identify the originator of an IP packet. Even in the absence of any deliberate attempt to disguise a packet's origin, widespread packet forwarding techniques such as NAT and encapsulation may obscure the packet's true source. Techniques have been developed to determine the source of large packet flows, but, to date, no system has been presented to track individual packets in an efficient, scalable fashion. We present a hash-based technique for IP traceback that generates audit trails for traffic within the network, and can trace the origin of a single IP packet delivered by the network in the recent past. We demonstrate that the system is effective, space efficient (requiring approximately 0.5% of the link capacity per unit time in storage), and implementable in current or next-generation routing hardware. We present both analytic and simulation results showing the system's effectiveness.**

***Index Terms*—Computer network management, computer network security, denial of service (DoS), IP traceback, network fault diagnosis, wide-area networks (WANs).**

I. Introduction

TODAY'S Internet infrastructure is extremely vulnerable to motivated and well-equipped attackers. Tools are readily available, from covertly exchanged exploit programs to publicly released vulnerability assessment software, to degrade performance or even disable vital network services. The consequences are serious and, increasingly, financially disastrous. While distributed denial-of-service (DDoS) attacks, typically conducted by flooding network links with large amounts of traffic, are the most widely reported, there are other forms of network attacks, many of which require significantly smaller packet flows. In fact, there are a number of widely deployed operating systems and routers that can be disabled by a single well-targeted packet (e.g., the Teardrop attack crashes versions of Microsoft Windows with one packet [1]). To institute accountability for these attacks, the source of individual packets must be identified.

Unfortunately, the anonymous nature of the IP protocol makes it difficult to accurately identify the true source of an IP datagram if the source wishes to conceal it. The network routing infrastructure is stateless and based largely on destination addresses; no entity in an IP network is officially responsible for ensuring the source address is correct. Many routers employ a technique called *ingress filtering* [2] to limit source addresses of IP datagrams from a stub network to addresses belonging to that network, but not all routers have the resources necessary to examine the source address of each incoming packet, and ingress filtering provides no protection on transit networks. Furthermore, spoofed source addresses are legitimately used by network address translators (NATs), Mobile IP, and various unidirectional link technologies such as hybrid satellite architectures.

Accordingly, a well-placed attacker can generate offending IP packets that appear to have originated from almost anywhere. While techniques such as ingress filtering, which suppresses packets arriving from a given network with source addresses that do not properly belong to that network, increase the difficulty of mounting an attack, transit networks are dependent upon their peers to perform the appropriate filtering. This interdependence is clearly unacceptable from a liability perspective; each motivated network must be able to secure itself independently.

Systems that can reliably trace individual packets back to their sources are a first and important step in making attackers (or, at least, the systems they use) accountable. There are a number of significant challenges in the construction of such a tracing system including determining which packets to trace, maintaining privacy (a tracing system should not adversely impact the privacy of legitimate users), and minimizing cost (both in router time spent tracking rather than forwarding packets, and in storage used to keep information).

We have developed a *Source Path Isolation Engine* (SPIE) to enable IP *traceback*, the ability to identify the source of a particular IP packet given a copy of the packet to be traced, its destination, and an approximate time of receipt. Historically, tracing individual packets has required prohibitive amounts of memory; one of SPIE's key innovations is to reduce the memory requirement (down to 0.5% of link bandwidth per unit time) through the use of Bloom filters [3]. By storing only packet digests, and not the packets themselves, SPIE also does not increase a network's vulnerability to eavesdropping. SPIE therefore allows routers to efficiently determine if they forwarded a particular packet within a specified time interval while maintaining the privacy of unrelated traffic.

The rest of this paper examines SPIE in detail. We begin by defining the problem of IP traceback in Section II, and articulate the desired features of a traceback system. We survey previous work in Section III, relating their feature sets against our

Manuscript received September 29, 2001; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor V. Paxson. This work was supported by the Defense Advanced Research Projects Agency (DARPA) under Contract N66001-00-C-8038. A preliminary version of this paper was presented at ACM SIGCOMM'01, San Diego, CA, August 2001.

A. C. Snoeren is with the MIT Laboratory for Computer Science, Cambridge, MA 02139 USA, and with BBN Technologies, Cambridge, MA 02138 USA (e-mail: snoeren@lcs.mit.edu).

C. Partridge, C. E. Jones, F. Tchakountio, B. Schwartz, S. T. Kent, and W. T. Strayer are with BBN Technologies, Cambridge, MA 02138 USA (e-mail: craig@bbn.com; cej@bbn.com; ftchakou@bbn.com; bschwart@bbn.com; kent@bbn.com; strayer@bbn.com).

L. A. Sanchez is with Megisto Systems, Inc., Germantown, MD 20874 USA (e-mail: lsanchez@megisto.com).

Digital Object Identifier 10.1109/TNET.2002.804827

requirements. Section IV describes the digesting process in detail. Section V presents an overview of the SPIE architecture, while Section VI offers a practical implementation of the concepts. Section VII provides both analytic and simulation results evaluating SPIE's traceback success rates. We discuss the issues involved in deploying SPIE in Section VIII before concluding in Section IX with a brief look at future work.

II. IP Traceback

The concept of IP traceback is not yet well defined. In an attempt to clarify the context in which SPIE was developed, this section presents a detailed and formal definition of traceback. We hope that presenting a strawman definition of traceback will also help the community better evaluate different traceback schemes.

In order to remain consistent with the terminology in the literature, we will consider a packet of interest to be nefarious, and term it an *attack packet*; similarly, the destination of the packet is a *victim*. We note, however, that there are many reasons to trace the source of a packet; many packets of interest are sent with no ill intent whatsoever.

A. Assumptions

There are several important assumptions that a traceback system should make about a network and the traffic it carries.

- Packets may be addressed to more than one physical host.
- Duplicate packets may exist in the network.
- Routers may be subverted, but not often.
- Attackers are aware they are being traced.
- The routing behavior of the network may be unstable.
- The packet size should not grow as a result of tracing.
- End hosts may be resource constrained.
- Traceback is an infrequent operation.

The first two assumptions are simply characteristics of the Internet Protocol. IP packets may contain a multicast or broadcast address as their destination, causing the routing infrastructure to duplicate them internally. An attacker can also inject multiple identical packets itself, possibly at multiple locations. A tracing system must be prepared for a situation where there are multiple sources of the same (identical) packet, or a single source of multiple (also typically identical) packets.

The next two assumptions speak to the capabilities of the attacker(s). An attacker may gain access to routers along (or adjacent to) the path from attacker to victim by a variety of means. Further, a sophisticated attacker is aware of the characteristics of the network, including the possibility that the network is capable of tracing an attack. The traceback system must not be confounded by a motivated attacker who subverts a router with the intent to subvert the tracing system.

The instability of Internet routing is well known [4] and its implications for tracing are important. Two packets sent by the same host to the same destination may traverse wildly different paths. As a result, any system that seeks to determine origins using multipacket analysis techniques must be prepared to make sense of divergent path information.

The assumption that the packet size should not grow is probably the most controversial. There are a number of protocols today that cause the packet size to grow, for example technologies that rely on packet encapsulation, such as IPsec and mobile IP. However, increasing the packet size causes MTU problems and increases overhead sharply (each byte of additional overhead reduces system bandwidth by about 1%, given the average packet size of about 128 B). A recent study by the Cooperative Association for Internet Data Analysis (CAIDA) [5] found that packet encapsulation (and the resulting growth in packet size) is the single largest cause of fragmentation on the Internet. It follows that an efficient traceback system should not cause packet size to grow.

We assume that an end host, and in particular the victim of an attack, may be resource poor and unable to maintain substantial additional administrative state regarding the routing state or the packets it has previously received. This assumption comes from the observed rise in special purpose devices such as microscopes, cameras, and printers that are attached to the Internet yet have few internal resources other than those devoted to performing their primary task.

The final assumption that traceback queries are infrequent has important design implications. It implies queries can be handled by a router's control path, and need not be dealt with on the forwarding path at line speed. While there may be auditing tasks associated with packet forwarding to support traceback that must be executed while forwarding, the processing of the audit trails is infrequent with respect to their generation.

B. Goal

Ideally, a traceback system should be able to identify the source of any piece of data sent across the network. In an IP framework, the packet is the smallest atomic unit of data. Any smaller division of data (a byte, for instance) is contained within a unique packet. Hence, an optimal IP traceback system would precisely identify the source of an arbitrary IP packet. Any larger data unit or stream can be isolated by searching for any particular packet containing data within the stream.[1]

As with any auditing system, a traceback system can only be effective in networks in which it has been deployed. Hence, we consider the source of a packet to be one of the following:

- the ingress point to the traceback-enabled network;
- the actual host or network of origin;
- one or more compromised routers within the enabled network.

If one assumes that any router along the path may be co-opted to assist in concealing a packet's source, it becomes obvious that one must attempt to discern not only the packet's source, but its entire path through the network. Because subverted routers can fabricate trace information, the path can only be guaranteed to be accurate on the portion from the victim to the a source or subverted router, whichever comes first. While subverted routers may attempt to conceal their identity by appending additional sources further upstream, the subverted routers themselves must

[1]Indeed, we would argue that it is desirable to trace the individual packets within a stream because the individual packets may have originated at different sites (meeting only at the victim) and are likely to have followed different paths through the network.

still appear as a node in the trace. We consider subverted routers that attempt to conceal the true source of a packet as co-conspirator and, therefore, attack sources themselves.

Hence, we are interested in constructing an *attack path*, where the path consists of each router traversed by the packet on its journey from source to the victim. Each node in an attack path either forwarded the packet or lies upstream of a subverted router that did. Further, since multiple, indistinguishable packets may be injected into the network from different sources in the general case, a traceback system should construct an *attack graph* composed of the attack paths for every instance of the attack packet that arrived at the victim.

If routers are subverted, they may provide misinformation to the traceback system, causing the attack graph to contain false positives; that is, the attack graph may implicate sources that did not actually emit the packet. We argue these false positives are unavoidable consequence of admitting the possibility of subverted routers. An ideal traceback system, however, produces no false *negatives* while attempting to minimize false positives; it must never exonerate an attacker by not including the attacker in the attack graph.

Further, when a traceback system is deployed, it must not reduce the privacy of IP communications. In particular, entities not involved in the generation, forwarding, or receipt of the original packet should not be able to gain access to packet contents by either utilizing or as part of participating in the IP traceback system. An ideal IP traceback system must not expand the eavesdropping capabilities of a malicious party.

C. Transformations

It is important to note that packets may be modified during the forwarding process. In addition to the standard decrementing of the time to live (TTL) field and checksum recomputation, IP packets may be further transformed by intermediate routers. Packet *transformation* may be the result of valid processing, router error, or malicious intent. A traceback system need not concern itself with packet transformations resulting from error or malicious behavior. Packets resulting from such transformations only need be traced to the point of transformation, as the transforming node either needs to be fixed or can be considered a co-conspirator (source). A complete traceback system should trace packets through valid transformations back to the source of the original packet.

Valid packet transformations are defined as a change of packet state that allows for or enhances network data delivery. Transformations occur due to such reasons as hardware needs, network management, protocol requirements, and source request. Based on the transform produced, packet transformations are categorized as follows.

1) *Packet Encapsulation*: A new packet is generated in which the original packet is encapsulated as the payload (e.g., IPsec). The new packet is forwarded to an intermediate destination for de-encapsulation. This is also known as *tunneling*.

2) *Packet Generation*: One or more packets are generated as a direct result of an action by the router on the original packet (e.g., an ICMP Echo Reply sent in response to an ICMP Echo Request, or packet duplication in IP Multicast). The new packets are forwarded and processed independent of the original packet. (A large number of *reflector* attacks utilize such transforms to hide their source [6].)

Common packet transformations include those performed by RFC 1812-compliant routers [7] such as packet fragmentation, IP option processing, ICMP processing, and packet duplication. Network address translation (NAT) and both IP-in-IP and IPsec tunneling are also notable forms of packet transformation. Many of these transformations result in a loss of the original packet state due to the stateless nature of IP networks.

A recent CAIDA study of wide-area traffic patterns found that less than 3% of IP traffic underwent common transformation and IP tunneling [8]. While this study did not encompass all forms of transformation (NAT processing being a notable omission), it seems safe to assume that packet transformations account for a relatively small fraction of the overall IP traffic traversing the Internet today. However, attackers may transmit packets engineered to experience transformation. The ability to trace packets that undergo transformation is, therefore, an essential feature of any viable traceback system.

III. Related Work

There are two approaches to the problem of determining the route of a packet flow: one can audit the flow as it traverses the network, or one can attempt to infer the route based upon its impact on the state of the network. Both approaches become increasingly difficult as the size of the flow decreases, but the latter becomes infeasible when flow sizes approach a single packet because small flows generally have no measurable impact on the network state.

Route inference was pioneered by Burch and Cheswick [9] who considered the problem of large packet flows and proposed a novel technique that systematically floods candidate network links. By watching for variations in the received packet flow due to the restricted link bandwidth, they are able to infer the flow's route. This technique requires considerable knowledge of network topology and the ability to generate large packet floods on arbitrary network links.

One can categorize auditing techniques into two classes according to the way in which they balance resource requirements across the network components. Some techniques require resources at both the end host and the routing infrastructure, others require resources only within the network itself. Of those that require only infrastructure support, some add packet processing to the forwarding engine of the routers while others offload the computation to the control path of the routers.

A. End-Host Storage

Some auditing approaches attempt to distribute the burden by storing state and performing computation at the end hosts rather than in the network. Routers notify the packet destination of their presence on the route. Because IP packets cannot grow arbitrarily large, schemes have been developed to reduce the amount of space required to send such information. Recently proposed techniques by Savage *et al.* [10] and Bellovin [11] explore in-band and out-of-band signaling, respectively.

Because of the high overhead involved, neither Savage *et al.* nor Bellovin attempt to provide audit information for every