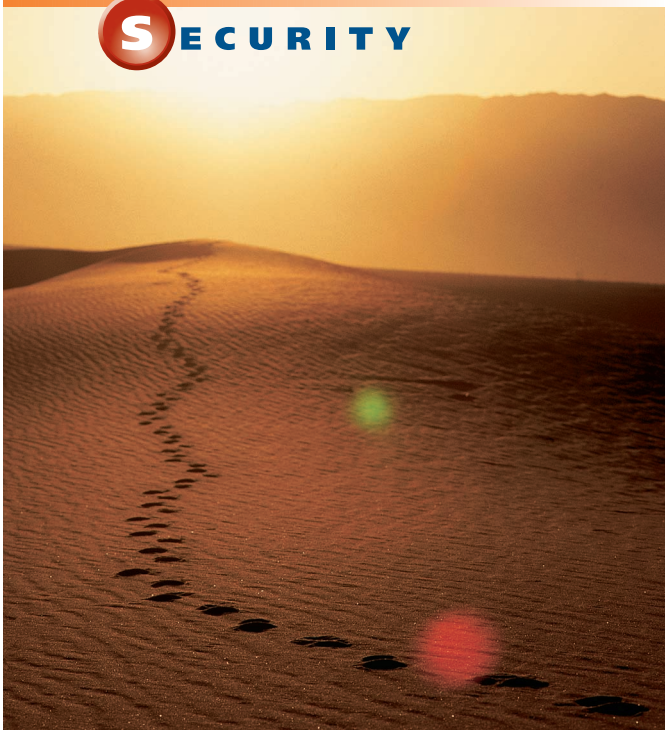
SECURITY



Technical, Legal, and Societal Challenges to Automated Attack Traceback

Susan C. Lee and Clay Shields

In physical-world crime, the hypothesis that the criminal always leaves something at the crime scene—and takes something of the crime scene away with him—is the basis for modern forensics. Careful forensics can uniquely tie fingerprints, DNA, hair, and even such minutia as fibers and pollen to an individual and a place. Voiceprints can identify anonymous callers; and paper, handwriting, or typewriter characteristics tie an anonymous letter to its writer. In contrast, network attacks leave no trace evidence; hackers carry out these attacks using electronic packets that have no identifying characteristics except those voluntarily supplied by the sender.

Tracing network attack depends on following a convoluted trail of packets and computer sessions back to their origin—a complex, multifaceted problem.

Those investigating attacks carry out attack traceback almost entirely through a manual process of human interactions and a perusal of records, logs, and other sources. Systems to do much of the legwork of attack traceback or provide law enforcement with evidence of an attack's origin are not available today.

BASIC PROBLEM

Ideally, a traceback system would identify the human responsible for the attack. In this discussion, however, we limit traceback to determining the computer host that is the source of an attack. Once a traceback system identifies an attacker host, investigators can apply traditional methods of crime investigation or intelligence gathering to assign responsibility to an individual.

To analyze the traceback problem, we must understand how attackers hide their identity. Most often, they compromise the two components of identity on the Internet.

Internet Protocol address

The first of these two components, Internet Protocol (IP) address, is what Internet software uses to direct packets—the basic unit of communication on networks—to the computer indicated by the sender. Each packet contains two such addresses. One is the packet's intended destination; the other is the packet's source (W.R. Stevens, *TCP/IP Illustrated, Volume 1*, Addison-Wesley, Reading, Mass., 1994). The basic assumption underlying IP design was the goodwill of the user: No one would send a message without wanting a proper reply. Therefore, in processing a packet or message, information about the source essentially remains unused until the item reaches its destination. For this reason, attackers can forge a packet's

1520-9202/02/$17.00 © 2002 IEEE

source address (set it to that of another computer or even a nonexistent computer), but the packet will still reach its destination. Thus, one way of concealing identity on the Internet is to simply forge source addresses, as Figure 1 shows.

Forging, or spoofing, an address in a one-way communication is as simple as putting any desired address in the source address field. Spoofing the source of a two-way communication is more difficult, because the victim will address its responses to the forged source address, not to the attacker. However, even if the attacker cannot see the packets sent by the victim to the forged source, the attacker can still carry out a simple, predictable two-way communication "in the blind." To do so, the attacker must guess the TCP (Transmission Control Protocol) sequence numbers in the victim's response packets; however, some operating-system implementations use easily guessable number sequences in network communication.

Attackers can use IP address forging to manipulate an innocent (uncompromised) host into attacking a victim. The attacker host sends a packet designed to elicit a response to a *reflector* host. If the attacker spoofs the victim's source IP as the packet source, then the reflector will innocently direct its response toward the victim, as Figure 2 shows. The response packet(s) constitute the attack. At the victim, the attack appears to come from the reflector. At the reflector, initiating packets appear to come from the victim. The attacker sits off to the side, seemingly uninvolved.

The possibility of source IP address forging makes the first problem in attack traceback learning the true identity of the upstream host in all the host-to-host communications that are part of the attack path.

Figure 1. Simple forging of IP source address.

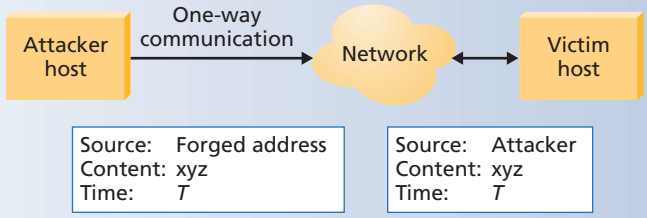


Figure 2. Use of a reflector host to hide identity.

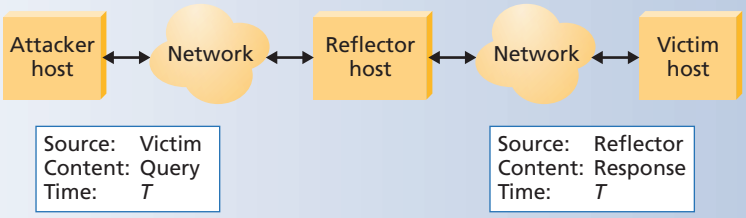
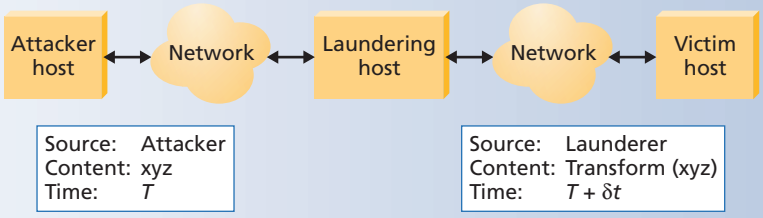


Figure 3. Effect of laundering on attack packets.



User account

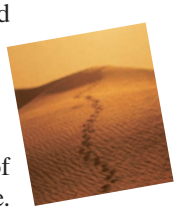
The second basic component of Internet identity is the user account—for example, e-mail or remote access. A generic account consists of a made-up name for the user (user ID) and some authentication material (typically a password). The system uses the authentication material to ensure that the person supplying the user ID is the person to whom the account belongs.

There are, however, many ways to create an untraceable user account. For example, by learning the user ID and password of an innocent user, an attacker can masquerade as another person while perpetrating his crimes. By using any one of several exploits to gain administrative privileges, an attacker can create a new account on any machine.

Attackers use such stolen or phantom accounts to launder packets before they reach a victim (Y. Zhang and V. Paxson, "Detecting Stepping Stones," *Proc. 9th Usenix Security Symp.*, Usenix Assoc., San Diego, Calif., 2000). When laundering takes place, the laundering host actually receives and processes the attacking host's packets, transmitting other packets toward the victim, as Figure 3 shows. This process changes the source address to that of the laundering host, and can also give the laundered packets different content and/or timing from that of the attacker's original packets. In these ways, attackers can use laundering hosts to disguise their identity.

Attackers basically use two types of laundering hosts: stepping stones and zombies.

Stepping stones. First, the attacker can use an intermediate host as a stepping stone. Once logged into the stepping-stone host, the attacker can launch an attack, and any traceback will not lead to the attacker, but to an innocent or phantom user on the stepping stone. Attackers typically conduct a classic penetration attack through multiple stepping-stone hosts—they use Telnet or some other common program to log on to host A, and then use host A to log on to host B, and so on. The typed commands contained in the packets eventually delivered to the victim will have been received, repacketized, and retransmitted through several stepping-stone hosts.



Attackers basically use two types of laundering hosts: stepping stones and zombies.

Zombies. The second kind of laundering host is a zombie. By definition, a zombie fundamentally transforms and/or delays the attacker's communications before they continue down the attack path. For example, after compromising the zombie host via some exploit, the attacker can install a Trojan program—timed to execute minutes, days, or even weeks after the attacker's contact—to attack the intended victim. Or a single packet sent by the attacking host can trigger a planted attack script that sends multiple packets to the victim. Distributed denial-of-service attacks use zombie hosts in this way (S. Dietrich, N. Long, and D. Dittrich, "Analyzing Distributed Denial of Service Attack Tools: The Shaft Case," *Proc. LISA 2000 System Administration Conf.,* Usenix Assoc., San Diego, Calif., 2000).

TECHNICAL ASPECTS OF AUTOMATED TRACEBACK

Automating attack traceback poses significant technical challenges. Researchers are actively exploring traceback solutions. Each of these will have different characteristics and uses.

Characterizing traceback results

We can characterize an attack traceback result using three parameters: precision, accuracy, and timeliness. Precision measures the exclusivity of the traceback result: A traceback can identify an attack source as one particular host (precise) or simply as a host in some particular country (imprecise). Accuracy measures a traceback's correctness; that is, given that a traceback results in identification, how likely is it that this identification is correct? Timeliness is a measure of when an investigator can obtain a traceback result. Some solutions might only give a result while an attack is in progress; others might require data that is only available after an attack. Some solutions depend on data that has a limited lifetime, and investigators must apply them within a specific time following the attack.

For attack reaction—stopping an ongoing attack—a traceback solution must operate in real time and have high accuracy. If the solution inaccurately identifies an attack's source, no action taken against that source can stop the attack. In this case, stopping the attack might not require high precision. For example, if investigators can track an ongoing attack to some particular input port on a router, a certain LAN (local area network), or even a domain, they can use filtering to mitigate the attack's effect.

To prevent a future attack, investigators need only complete traceback in time to take preventive measures before the next attack. In this case, precision must be higher than for attack reaction. For example, it is acceptable to filter traffic from a fairly large segment of the Internet for a short time during an attack, but such an approach is unacceptable in the long term. If a traceback solution can identify an attack more precisely, however, more preventative options are available. For example, if traceback can identify the attack as originating from some network under a single administration, the evidence could persuade system administrators to institute stricter security measures.

To establish liability, traceback need only occur in a time frame consistent with the existence of ephemeral data and the statute of limitations that applies to each particular case. Liability might not require high precision; in fact, traceback to the first entity with deep pockets—a large corporation's intranet or a large Internet service provider (ISP)—is best in this case. Criminal prosecution will require precise identification of one or more individual attackers. A high probability of accuracy could be enough to obtain a favorable verdict in either case.

Ongoing research

For the foreseeable future, traceback tools will require supplemental, unautomated intervention by humans. In particular, people must resolve administrative barriers to traceback and also supply supplemental information—such as work records, interviews, telephone records, and so on—unavailable to automated processes. Still, automated traceback systems, such as those described next, could assist humans in narrowing down candidate attack sources.

Packet source identification

The problem of identifying the actual host that is the source of any given packet (given the possibility of IP source forging) is tantamount to tracing the packet's passage backwards through the network's switching fabric. Given enough resources, this is the one fundamental traceback problem that has a guaranteed technical solution. Suppose, for example, that you could instantly replace every routing device in use today with one that places its own unique router ID in a list contained in each packet it receives. This way, the destination host would receive a packet containing the packet's entire route.

The practical drawbacks to this scheme are immediately obvious: It adds to the routing overhead and requires changes to all routing device hardware, the network protocol, and the packet's minimum size. So current research on route traceback focuses on addressing these practical difficulties. Researchers have developed three distinct approaches.

Overloading. One approach overloads some field already present in the IP packet header with route identification material. This approach essentially encodes a unique route through a potentially large number of devices within a strictly limited number of bits. Furthermore, it must do so in a secure (unspoofable) way. Only the fact that each router connects to only a few other routers makes the problem potentially solvable. One scheme uses a code based on the IP addresses of the routing devices that sequentially handle a packet (D. Song and A. Perrig, "Advanced and Authenticated Marking Schemes for IP Traceback," *Proc. IEEE Infocom 2001*, IEEE Press, Piscataway, N.J., 2001). This code goes into the IP identification field. Using this approach, a destination host can reconstruct a packet's route through up to 32 devices with reasonable computational efficiency and precision (W. Lee and K. Park, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack," *Proc. IEEE Infocom 2001*, IEEE Press, Piscataway, N.J., 2001).

Adding router IDs to packets has practical drawbacks: It requires changes to devices, network protocols, and a packet's minimum size.

Trace packet. The second approach requires routing devices to emit a secondary trace packet for each packet they handle. The destination host collects both the original packet and all the associated trace packets, and can use them to reconstruct the original packet's route. The advantage is that the trace packet can contain unambiguous, authenticated identification for the originating router. The clear disadvantage is an enormous increase in network traffic.

In a scheme of this type proposed by Steve Bellovin, the routing devices emit a trace packet on a probabilistic basis (about 1 in 20,000 for Bellovin's scheme), minimizing the increase in traffic (Steve Bellovin, "ICMP Traceback Messages," Internet draft, Internet Engineering Task Force, Washington, D.C., Mar. 2000). Such a scheme could most certainly trace a large stream of packets continuing for a considerable time (as during a `SYN` flood denial-of-service attack stream, for example).

Query. The last approach requires a destination host to query its immediately connected routing device, asking "Have you seen this packet?" Any routing device that has handled the packet in question will respond positively and also repeat the query to all connected upstream routers. Examination of the positive and negative responses yields the packet route. If this approach only traces suspicious packets, the bandwidth overhead might be small. Such an approach does require routing devices to store information on all packets they handle for some period of time. Because of processing and memory limitations, this time period is likely to be quite short, limiting this approach to a near-real-time traceback.

One implementation of a scheme that uses this approach is the Intrusion Detection and Isolation Protocol (IDIP) developed by a collaboration among Network Associates, Boeing, the University of California at Davis, and Silicon Defense. The US Defense Advanced Projects Research Agency sponsored this work (Dan Sterne and colleagues, "Autonomic Response to Distributed Denial of Service Attacks," *4th Int'l Symp. Recent Advances in Intrusion Detection RAID 2001*, Springer-Verlag, Heidelberg, Germany, 2001, pp. 134-149).

Identifying packet streams

Once you identify the source of an attack packet stream as a stepping-stone or zombie host, an effective traceback method must answer the following question: Out of all the packet streams that have come into this host, which one caused the output (attack) stream?

Of the two cases, the stepping-stone host is easier to trace. Because the stepping stone acts as a mere conduit, it has received the exact packets it sends within a very short time. So it is possible to identify the upstream host by matching the two communication streams—the ones into and out of the stepping stone. Automated traceback through stepping-stone hosts amounts to determining if two packet streams, viewed at different points in the network, are essentially the same stream.

The zombie host represents the far more general causality problem. Here, the upstream communication that resulted in the attack is not similar in content or connected in time to the communications downstream from the zombie. All that investigators know is that, at some point in time, a communication received by the zombie caused the observed outgoing packet stream.

Stream matching

Stream matching seeks to identify attack sources by comparing streams of packets coming into and leaving a stepping-stone host. These techniques focus on matchings based on one of two characteristics: packet contents or interpacket timing.

Content matching. In the mid-1990s, Stuart Staniford-Chen developed a content-matching scheme while at the University of California at Davis. Called thumbprinting, this scheme divides the stream into discrete time intervals and creates digests (such as hashes) of packets within each interval (Stuart Staniford-Chen and L.T. Heberlein,

"Holding Intruders Accountable on the Internet," *Proc. 1995 IEEE Symp. Security and Privacy*, IEEE CS Press, Los Alamitos, Calif., 1995, pp. 39-49). This method compares two streams by computing the similarity of their stream digests. Staniford-Chen showed that similar thumbprints are far more probable to represent the same stream than two random streams. Although the scheme works quite well for unencrypted streams, interim encryption makes stream matching by this method impossible.

Interpacket-timing matching. In ongoing research at Purdue, using interpacket timing to match streams has shown some success. Despite random network delays, the characteristic give-and-take of network connection protocols and human-computer interactions generates a timing thumbprint. The timing thumbprints of a single stream viewed at two points in the network are more similar than the timing thumbprints of unrelated streams, though congestion in the network adds potentially significant noise to the comparison.

Other researchers have also suggested a second method of stream matching by interpacket timing. In this approach, researchers actively perturb a stream's timing at some location in the network and search streams at other points in the network for matching perturbations.

Stream matching must either take place in real time or the stream matching systems must store information on matching streams. Storage limitations will affect the timeliness of stream-matching techniques.

Determining causality

Without markings, content, or timing to connect streams, automatically finding the attacking host upstream of a zombie will be difficult. Automatically determining causality will certainly require access to logs on the zombie host (if they exist) to find the immediate cause of the output stream (such as a Trojan program or script) and the trigger (either via remote command or chronological trigger). Investigators might use this information to select a time window in which to look for the source of the true causal event. Investigators, for example, could examine the remote connections established within the time window. Clearly, the longer the time window, the larger the set of possible sources will be, and the less likely it will be for the system to have retained relevant logs. For the foreseeable future, the use of zombie hosts will limit the potential for automated traceback systems.

LEGAL CHALLENGES

Unlike passive defense or detection, attack traceback enters a realm in which researchers must account for legal considerations. Even when liability and prosecution are not traceback goals, laws to protect privacy could limit the technical solutions. The three US federal laws that dictate legal considerations for traceback are the Electronic Communications Privacy Act, ECPA (18USC2701); the Wiretap Act (18USC2511); and the Trap and Trace Act (18USC3121). Unfortunately, legislators did not write any of these statutes with computer networks specifically in mind, so the meaning of their provisions to computer networks must be interpreted and tested in court. To date, insufficient case law exists to provide firm guidance.

Content-based traceback and the law

The type of data used in traceback and the means used

to collect it all have legal implications. For example, information gleaned from packet headers alone is fair game for traceback; legally, there is no expectation of privacy for packet headers. In contrast, packet contents are legally protected, and a traceback solution that uses packet contents could require lengthy and difficult legal procedures to obtain permission for its use.

Gray areas exist for which the statutes themselves provide no guidance. For example, the content thumbprinting technique described earlier uses digests of contents. Technically, the law protects the privacy of the original packet contents; the privacy of a digest, however, is currently undecided. The law also protects the name and address of an ISP subscriber, but law enforcement could obtain permission to access such information using a more simple procedure than for content information.

Data collected "in the normal course of business" is admissible, but the exact boundaries of "normal" are a gray area.

The law also distinguishes between

- data that is merely collected versus data disclosed to others,
- voluntary versus legally compelled data disclosure, and
- access to stored data versus collection of data in real time.

The legal meaning and impact of these distinctions also vary, depending on whether the data is used in a civil action or a criminal prosecution. It is well beyond the scope of a technical article to explore these ramifications thoroughly. However, two examples will serve to illustrate the complexity and ambiguity of the legal situation.

What data can you legally collect? In a state of affairs worthy of the novel *Catch-22*, evidence gathered in anticipation of litigation might not be admissible as evidence. Data collected "in the normal course of business" is admissible, but the exact boundaries of "normal" are a gray area currently underexplored in case law. For example, is traceback information admissible if company policy mandates its routine collection? What if the anticipation of litigating cases of intrusion drives the policy? What if the data collection is not continuous but automatically triggered by intrusive events? What if the trigger is not automatic but instead requires an administrator's intervention? These and other related questions are currently open to interpretation.

What type of questions can you ask? Traceback solutions that use querying could also conflict with the law. The ECPA makes it illegal for any government agency (not just those involved in law enforcement) to obtain electronic information from nongovernmental entities without legal process (requesting warrants and so on).

Suppose a government agency's system initiated or passed on a traceback query; it is possible to interpret the law as forbidding this altogether. Another interpretation is that the query must identify its source as governmental or nongovernmental so that receiving devices can decide whether or not to respond. Still another interpretation is that as long as the traceback confines the query to "Have you seen this traffic?" it is legal, but if it asks "Where does this traffic come from?" it is not.

There are many other legal impediments to automated traceback not related to the type of data. For example, another legal difficulty with querying systems is that eventually, in court, some human must testify as to the continuity of the query for each administrative domain it passes through. This requirement alone makes prosecuting cases based on a query-type traceback extremely expensive and difficult. There are also legal implications to taking an active approach to traceback. For example, actively perturbing a communications stream—one technique discussed earlier—could violate the law, especially if the perturbed stream originates outside the administrative domain in which the investigators apply the active perturbation.

Additional technical requirements

Legal uses of traceback results impose additional technical requirements on traceback systems. Traceback solutions must incorporate features that allow for time synchronization of events recorded at distant locations. Secure logging is necessary to protect evidence from attack in court. It would be easier to build a legal case if systems captured all pertinent traceback information in a single place, rather than assembling it from multiple logs and data files. Finally, because legal machinery moves slowly, a long retention time for records is extremely important.

SOCIETAL CHALLENGES

Perhaps even more daunting than the legal implications are the societal barriers to traceback. A surprising amount of suspicion drives interactions among government and commercial entities, and among commercial enterprises. This lack of trust makes privacy of information a higher priority than attack traceback for many enterprises. As described earlier, traceback will require additional infrastructure and cooperation among entities sharing a network. An important question for traceback is what business models are likely to cause the compliance needed to perform traceback.

There are two potential drivers toward increased cooperation for traceback. The first is increasing government regulation that could, to some extent, force cooperation. The second is the increasing cost of attacks, which could

provide an economic motive for increased cooperation. Part of the attack's cost will be liability for the attack's results. Customers could sue companies for loss of privacy if attackers compromise their private information. E-businesses could sue ISPs to recover the cost of lost business during denial-of-service attacks. Legally, businesses might escape responsibility for harm resulting from occurrences that they could not reasonably anticipate. Normally, criminal acts fall in this unanticipatable category; however, some case law has already determined that network attacks are so common that companies should anticipate them.

Eventually, the direct cost of attacks and the threat of liability for attacks could create demand for an attack insurance industry. Once insurance companies become involved, they will have a cross-enterprise incentive for attack traceback to allow for cost recovery. Premium incentives and conditions of insurance could dictate adoption of standard attack traceback tools and techniques.

To provide insurance against attack profitably, however, insurance companies must have actuarial data. Thus, like so many other information assurance problems, the ultimate solution to attack traceback could rest on the definition of appropriate metrics and collection of data over a broad cross section of society.

It will be some time before automated traceback systems become available to aid network users in finding, stopping, and perhaps prosecuting attackers on the Internet. Even when such systems are available, they will not be a panacea. They will require system administrators to keep the logs, records, and other information that traceback systems need to work. They will require cooperation among industry, service providers, network administrators, network users, and law enforcement to be effective, just as manual traceback requires today. We can begin now to establish these record-keeping standards and to build the cooperative processes that an automated system can use to advantage. ■

Susan C. Lee *is chief scientist for information operations at The Johns Hopkins Applied Physics Laboratory. Contact her at sue.lee@jhuapl.edu.*

Clay Shields *is an assistant professor of computer science at Georgetown University in Washington, D.C. Contact him at clay@cs.georgetown.edu.*

For further information on this or any other computing topic, visit our Digital Library at http://computer.org/publications/dlib.