IDA

INSTITUTE FOR DEFENSE ANALYSES

Techniques for Cyber Attack Attribution

David A. Wheeler

Gregory N. Larsen, Task Leader

October 2003

IDA Paper P-3792

Log: H 03-001218

This work was conducted under contracts DASW01-98-C-0067/ DASW01-02-C 0012, DASW01-04-C-003, Task BC-5-1767, for the Assistant Secretary of Defense for Networks and Information Integration Command, Control, Communications, and Intelligence (ASD(C3INII). The publication of this IDA document does not indicate endorsement by the Department of Defense, nor should the contents be construed as reflecting the official position of that Agency.

INSTITUTE FOR DEFENSE ANALYSES

IDA Paper P-3792

Techniques for Cyber Attack Attribution

David A. Wheeler

Gregory N. Larsen, Task Leader

Preface

This document was prepared by the Institute for Defense Analyses (IDA) under the task order, Computer Network Defense Assessment, in response to a task objective, to "provide technical expertise and analyses in support of the DIAP's development and evolution to enable continued improvements in the Department's IA posture." The Defense-wide Information Assurance Program (DIAP) sponsored this work.

The following IDA research staff members were reviewers of this document: Dr. L. Roger Mason, Jr., Dr. Alfred E. Brenner, Mr. Terry Mayfield, Dr. Reginald N. Meeson, Dr. Edward A. Schneider, and Dr. William R. Simpson.

Contents

Figures

Tables

Executive Summary

This paper summarizes various techniques to perform attribution of computer attackers who are exploiting data networks. Attribution can be defined as "*determining the identity or location of an attacker or an attacker's intermediary*." In the public literature "traceback" or "source tracking" are often used as terms instead of "attribution."

This paper is intended for use by the U.S. Department of Defense (DoD) as it considers if it should improve its attribution capability, and if so, how to do so. However, since the focus of this paper is on technology, it may also be of use to many others such as law enforcement personnel. This is a technical report, and assumes that the reader understands the basics of network technology, especially the Transmission Control Protocol/Internet Protocol (TCP/IP) suite of protocols.

The paper identifies the following attribution techniques:

1. Store Logs & Traceback Queries	9. Exploit/Force Attacker Self-Identification (e.g., beacons, web bugs, cookies, watermarking)
2. Perform Input Debugging	10. Observe Honeypot/honeynet
3. Modify Transmitted Messages	11. Employ Forward-deployed Intrusion Detection Systems (IDSs)
4. Transmit Separate Messages (e.g., iTrace)	12. Perform Filtering (e.g., Network Ingress Filtering)
5. Reconfigure & Observe Network	13. Implement Spoof Prevention
6. Query Hosts	14. Secure Hosts/Routers
7. Insert Host Monitor Functions (e.g., "Hack Back")	15. Surveil Attacker
8. Match Streams (via headers, content, and/or timing)	16. Employ Reverse Flow
17. Combine Techniques	

The paper also discusses a number of issues related to attribution.

This paper concludes and recommends the following:

1. There are a large number of different attribution techniques. Each technique has its strengths and weaknesses; no single technique replaces all others.

2. Attribution is difficult and inherently limited. In particular, attackers can cause attacks to be delayed and perform their attacks through many intermediaries in many jurisdictions, making attribution difficult. In some cases this can be partly countered, for example, by treating some information-gathering techniques as attacks (and attributing them), using multiple techniques, and using techniques that resist this problem (such as exploiting/forcing attacker self-identification and attacker surveillance). Nevertheless, because of the difficulty and uncertainty in performing attribution, computer network defense should not *depend* on attribution. Instead, attribution should be part of a larger defense-in-depth strategy.

3. Attribution tends to be easier against insiders or insider intermediaries.

4. Prepositioning is necessary for many attribution techniques.

5. Many techniques are immature and will require funding before they are ready for deployment. If the DoD wishes to have a robust attribution capability, it must be willing to fund its development and deployment.

6. A useful first step for the DoD would be to ***change the terrain*** of its own network. By this, we mean modify DoD computers and networks to aid attribution techniques. This includes hardening routers and hosts, so exploiting them as intermediaries is more difficult, limiting spoofable protocols, disabling broadcast amplification/reflection, and implementing network ingress filtering. Changing the terrain should also be applied to key networks the DoD relies on, to the extent possible.

1. Introduction

This paper summarizes various techniques to perform attribution of computer attackers who are exploiting data networks. Attribution can be defined as determining the identity or location of an attacker or an attacker's intermediary. In the public literature "traceback" or "source tracking" are often used as terms instead of "attribution," and in the commercial world a major interest in attribution is to counter distributed denial of service (DDoS) attacks. A taxonomy of DDoS attacks and of DDoS defense mechanisms is given in [Mirkovic]. This paper was developed by identifying and organizing the public literature available on the subject.

This paper is intended for use by the U.S. Department of Defense (DoD) in considering if and how it should improve its attribution capability. However, since the focus of this paper is on technology, the list of techniques may also be of use to many others such as law enforcement personnel. This is a technical report, and assumes that the reader understands the basics of network technology, especially the Internet's Transmission Control Protocol/Internet Protocol (TCP/IP) suite of protocols.

There are other summaries of attribution techniques, such as [Lee 2002] and Dave Dittrich's list of DDoS attacks and tools [Dittrich]. A website dedicated to surveying backtracking analysis is at Oak Ridge National Laboratory [ORNL], sponsored by the Office of Counter Intelligence of the U.S. Department of Energy, which includes the survey [Dunigan 2001]. Another website records the results of the "Attack Traceback Summit Proceedings" of September 6-8, 2000 [Purdue]; [Buchholz] includes a summary. Silicon Defense maintains a "Traceback and Related Papers Archive" [Silicon Defense]. However, these other summaries omit many attribution techniques, so making decisions solely based on them would ignore important alternatives. This paper aims to fulfill the need for a more inclusive summary of attribution techniques.

1.1 Defining Attribution

There is no universally agreed upon definition of the term attribution in the field of information assurance (IA). One dictionary defines the general term "attribution" as "to explain by indicating a cause." [Merriam-Webster 1983].

This paper defines "attribution" as "*determining the identity or location of an attacker or an attacker's intermediary.*" A resulting identity may be a person's name, an account, an alias, or similar information associated with a person. A location may include physical (geographic) location, or a virtual location such as an IP address or Ethernet address.

This definition includes intermediaries, and not just the attacker. An ideal attribution process would always identify the original attacker's identity and location. Unfortunately, clever attackers can often make themselves difficult to directly attribute (and/or providing misleading information to hide the true attacker). However, even if only an intermediary is identified, that information can still be useful. For example, blocking an attack may be more effective if an intermediary is known.

An attribution process may also provide additional information, such as the path used to perform the attack and the timing of the attack, but these cannot always be determined. In particular, it is worth noting that it can be difficult to determine by technical means the motivation for an attack.[1]

A related term is *traceback*, which will be defined in this paper as "*any attribution technique that begins with the defending computer and recursively steps backwards in the attack path toward the attacker.*" Thus, traceback techniques are a subset of attribution techniques. The term "traceback" is common in the public literature on this topic.

1.2 Rationale for Attribution

The U.S., including the DoD, is under constant network attack, and there is every reason to believe that increasingly capable and sophisticated network attacks will be perpetrated in the future. While there is a great need to harden DoD infrastructure from these attacks, passive computer network defenses cannot be, and will never be, perfect. Thus, if the DoD attempts to passively withstand all attacks, it will eventually succumb to a serious attack. As with conventional warfare, a good offense is often the strongest defense.

However, many offensive techniques, such as computer network attack, legal action (e.g., arrests and lawsuits), and kinetic energy attacks, can only be deployed if the source of the attack can be attributed with high confidence. In addition, some defensive techniques can only be employed if the defender has specific knowledge about the attacker's identity or location. Therefore, there is a need for attribution.

1.3 The Problem

In this paper, we assume that there is an adversary, attacking a system via a data network, who is potentially both intelligent and resourceful. This adversary will be termed the "attacker" in this paper. Other papers may use other terms such as "intruder" or "cracker." In this environment, the defender (also termed the victim) wants to identify or locate the attacker or at least an intermediary so a targeted response can be employed.

[1] There is ongoing work to attempt to infer the motivation / intent of attackers, based on information presented by the attack. For example, see the DARPA project "Inferring Intent of Attackers" by Peter A. Jarvis, Karen L Myers, and Teresa Lunt; more information about the project is at http://www.ai.sri.com/project/IIA.

Unfortunately, a resourceful attacker can use many approaches to make attribution difficult:

1. On an internet, data identifying the sender is normally unused while sending, so its source information can be easily forged. Forging the sender's identity in a message is called *spoofing* [Bellovin 1989]. In particular, at the Internet IP level, spoofing UDP packets is trivial. Spoofing TCP packets is slightly more difficult because of TCP protocol's design (particularly because of TCP's "sequence numbers") but it is still possible (for further discussion, see [Bellovin 1996] [Zalewski 2001]).

2. Attackers can use a "reflector host", who replies to a forged sender and thus really replies to the actual victim, hiding the attacker's location.

3. Attackers can exploit protocols in other, subtler ways to hide their identity. For example, they can set their IP packet's "time to live" (TTL) value too low and then forge the source address. A router will reply with an expired packet message to the forged source address [Templeton 2003].

4. Attackers can hide their identity and location by using a "laundering" host [Lee 2002]. A laundering host is a system that transforms data in some manner.

   - A laundering host that immediately passes that data without processing (other than repackaging the data for its new source, origin, and lower-level protocol) is termed a stepping stone. For example, if an attacker logs into system A (e.g., using ssh), and then uses system A to log into system B (e.g., using telnet), then system A is a stepping stone between the attacker and system B.

   - A laundering host that performs some more significant processing or intentionally inserts some delay is termed a zombie. In particular, note that an attacker may use a zombie to delay an attack for a long time, giving the attacker ample opportunity to escape before the attack triggers.

5. Attackers may use very fast attacks, possibly measured in milliseconds, or may distribute their attack over lengthy periods (e.g., months). This large range of timescales makes it more difficult to build effective attribution tools.

Figure 1 illustrates the attribution problem's environment. The thick lines represent local area networks, the circles represent routers, and the rectangles represent other hosts on the network. In this illustration, the attacker (on the top left of the diagram) sends an attack through a number of different hosts, which ends up at the defending host. The defender must attribute (identify or locate the attacker or at least one of the intermediaries) without misidentifying an innocent host. Although not shown in this figure, the attacker may actually control multiple intermediate systems. For example, distributed denial of service (DDoS) attacks involve a single attacker controlling a large number of intermediate systems that then attack a defender.

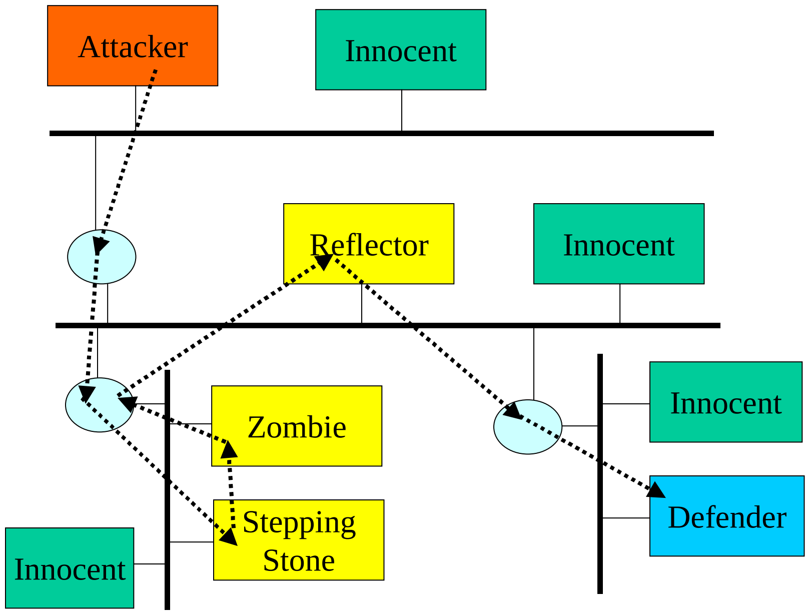


Figure 1. Attribution Problem

Modern environments often make attribution quite difficult. Typical computer network environments are *not* designed to support attribution of attackers. There are often many components in a network, making it easier for attackers to hide. Data paths may go through many systems in many countries or may be controlled by many different administrative domains, including those who may be hostile or noncooperative. Many networking capabilities unintentionally create complications for attribution, such as network address translation (NAT) that can change the sender and receiver address.

1.4 Scope

This paper is focused solely on identifying different techniques that could be used for attribution of attackers. This paper only examines attribution techniques for attackers attacking via an electronic data network (usually an Internet standards based network). Other attacks, such as physical attacks, social engineering attacks, or trusted programmers inserting malicious code into their own programs during development, are concerns but are outside the scope of this paper. This paper concentrates on approaches based on technology; non-technical approaches such as various human intelligence techniques are not the focus of this paper.

This paper does not cover identifying or locating people who are not directly attacking the defender. In particular, identifying or locating people voluntarily cooperating with each other is not covered in this paper, although some attribution techniques may also be useful in that case. [Wright 2002] describes some attribution-like techniques for anonymous peer-to-peer (P2P) networks. It also does not cover the general issue of discovering network topologies (as opposed to individual people or nodes); other resources such as the Cooperative Association for Internet Data Analysis[2] or Cyber-geography Research[3] may be useful starting points for such information.

This paper does not cover how to detect the occurrence of an attack. This paper presumes that for whatever reason, an attack has been detected. In practice, the attack might be detected by components such as an intrusion detection system (IDS), application, or firewall. See [Axelsson 2000] for a survey and taxonomy of IDSs. There are alternatives, for example, a random sample of data could be attributed to ensure that only authorized users are using the system. Indeed, the defender could treat all data as an attack unless proven otherwise, though this is unlikely to be practical in many environments.

This paper does not concern itself with determining *how* the attacker attacked. For purposes of this paper, this is considered part of "characterization," which is defined in this paper as "determining how the attacker attacked, including determining the properties, capabilities, and relative strength of an attack." The attribution process may also aid in characterizing the attack, but characterization is considered outside the scope of this paper.

After attribution, a defender may decide to perform some response to the attack specifically directed at the attacker or the attacker's intermediary. There are many response options, including host/network reconfiguration (e.g., lowering the bandwidth along some paths, disconnection of the attacker's path, transferring the connection to a decoy/honeypot, or hardening/re-installing intermediate systems), legal action, intelligence operations, computer network (counter) attack (CNA), and kinetic energy attack. Clearly, the decision of what response to make may depend on the nature of the attack, the attribution information, the confidence in that attribution, etc. Response options and decision processes are outside the scope of this paper.

There are important legal and policy issues surrounding attribution, but this is a large topic by itself and is outside the scope of this paper. [Aldrich 2002] examines some of the important legal issues involved in attribution, and notes that the law recognizes four fairly distinct roles in the area of computer network defense (CND): service provider, law enforcement, intelligence, and the warfighter. Some of these attribution techniques can only be used in certain special conditions or used a limited number of times, and their use must be carefully controlled. Some laws may need to be modified or clarified before some techniques can be used, at least in certain circumstances. Clearly, attribution techniques must be controlled in a way to ensure that their use is legal. Again, for more information on the legal issues, see [Aldrich 2002].

---

[2] http://www.caida.org

[3] http://www.cybergeography.org

Important terms in the DoD are Computer Network Attack (CNA) and Computer Network Defense (CND). Determining whether or not an attribution technique is a CNA or CND technique, and under what conditions, is not in the scope of this paper.

1.5 Generalization

To simplify and shorten this paper, general attribution techniques are discussed along with specific examples from publicly available literature. These general techniques can then be applied a number of different ways.

In particular, each technique can apply to many different network protocols. Much of the public literature on attribution focuses on the Internet Protocol (IP). One reason for this focus is that IP is central to any network based on Internet standards, so any implementation focusing on IP is useful in many circumstances. However, attribution can also be supported in other protocols, including Ethernet, Simple Mail Transfer Protocol (SMTP, the Internet standard for email), instant messaging protocols, the Dynamic Host Configuration Protocol (DHCP), and so on. Rather than re-describing the same general technique for each protocol, a single technique is discussed that may apply to many protocols. To emphasize this generality, the term “message” is used instead of “packet.” A “message” is a unit of information for the relevant protocol. Every “message” has a “message header” and “message content”:

1. The message header provides information about the message, such as the source and destination of the message. This information is used to bring the message to its intended recipient.

2. The message content contains the actual message. This content may be further broken down (e.g., Internet mail message content may have multiple MIME parts).

A “router” for a given protocol is any component that forwards messages of that protocol. For example, an Internet router is a router for IP traffic, while a Mail Transfer Agent (MTA) is a router for SMTP email.

This paper uses other means to describe the techniques in more general ways:

1. Many techniques can be implemented on the endpoints (hosts) of the communications, on the message routers, or on separate monitors that observe network traffic. These are not considered separate techniques, although the impact of different implementations may be noted.

2. Many techniques can be implemented either manually or in an automated manner. Automation of a manual technique is not considered a different technique. Note that manual techniques often fail since the speed of attacks can be far greater than a manual technique can support.

3. Many techniques that involve querying can respond with either the information being requested, or simply store the response and respond with an index to that response. The advantage of the latter approach is that the information is stored, but authentication and authorization of the person requesting the attribution information can happen separately. Since such authentication and authorization may take a long time, but the data may disappear if not stored quickly, this approach can be valuable.

2. Attribution Techniques

There are many different technical approaches that can be used to perform attribution. For purposes of this paper, these approaches have been grouped into the following seventeen techniques as shown in Table 1. The numbers with the technique names are simply identifiers; their order is unimportant.

Table 1. Attribution Techniques

Technique Name	Technique Description
1. Store Logs & Traceback Queries	Messages are logged by routers as they go through a network. Requests are traced backwards, asking each router if it has seen the message. This supports attribution of messages that were not previously identified as dangerous, but the logging routers must be pre-positioned, can have problematic costs and performance implementations, and many implementations invoke privacy concerns.
2. Perform Input Debugging	When attacked, defenders use the attack as a query to ask adjacent routers to report when they see the pattern again. If a router reports, the query is sent up to its adjacent routers, and so on. This approach is currently used against some DDoS attacks, but is fundamentally reactive and only works against attacks that continuously stream data.
3. Modify Transmitted Messages	Routers mark messages as they are transmitted so their route can be identified. This can increase bandwidth and/or decrease network performance, and can interfere with some authentication mechanisms.
4. Transmit Separate Messages (e.g., iTrace)	When routers route a message, they also send a separate message to aid in attribution. If the separate messages are sent for all messages, this could easily overwhelm network resources, but if it is only rarely done, attribution is less likely (typically only working against continuous flooding attacks).
5. Reconfigure & Observe Network	Reconfigure the network, and use the information on what (if anything) changed to backtrack to a previous step. This can be difficult to implement on large networks and create new security vulnerabilities. “Controlled flooding” can be used on networks owned by others, but can be viewed as an attack on third parties and should only be used in limited circumstances.
6. Query Hosts	Query hosts for internal state information to aid in attribution. This can be rapid, but it requires that there be a pre-existing query function. If an attacker controls the host, this may alert the attacker and make the information much less reliable.

Technique Name	Technique Description
7. Insert Host Monitor Functions (e.g., "Hack Back")	Insert querying functionality into a host that does not already provide this information (note the similarity to "Query Hosts"). A "hack back" is doing this without permission of the owner, and clearly requires significant legal control. If an attacker controls the host, this may alert the attacker and make the information much less reliable.
8. Match Streams (via headers, content, and/or timing)	Observe the streams of data entering and exiting a network or host, and determine which input streams match which output streams. This can aid attribution without needing to know the internal state of the network/host, but matching is a difficult technical problem, particularly against delayed attacks and encryption occurring inside the network/host.
9. Exploit/ Force Attacker Self-Identi-fication	Use information the attacker sends, intentionally or not, to identify the attacker. In some cases the defender can cause the attacker to send this data. When this technique works, it can directly reveal the attacker regardless of how well they hide otherwise, but many of these techniques depend on highly technical and specialized approaches (e.g., beacons, web bugs, cookies, and watermarking) that are easily foiled once an attacker knows about them.
10. Observe Honeypot/ honeynet	Honeypots/honeynets are decoy systems; anyone using them is by definition an attacker. Zombies placed in honeypots/honeynets can be revealed instantly. However, honeypots/honeynets must be monitored and analyzed (requiring significant expertise) and can only attribute attacks that go through them.
11. Employ Forward-deployed Intrusion Detection Systems (IDSs)	Place intrusion detection systems (IDSs) as close as possible to potential attackers (instead of near the defended assets). The effectiveness of this approach depends on the placement of the IDSs (they should be close to the attacker). This technique often requires significant monitoring effort, since IDSs are prone to many false positives and false negatives.
12. Perform Filtering (e.g., Network Ingress Filtering)	Filter messages so that certain links only permit messages to pass if they meet criteria that ease attribution. An advantage of the general technique is that it is often transparent to users and requires no additional storage; the information for attribution is stored in the message itself. A disadvantage of the technique is that it is primarily only useful for attribution of internal attack locations, and often only identifies a range of possible attribution values (not a specific location or identity). Often there must be multiple different paths a message can pass through, creating ambiguities that weaken the technique's effectiveness. An important approach implementing the technique is "network ingress filtering," which requires that all messages entering a network have a source address in a valid range for that network entry point. Network ingress filtering for IP is easily implemented using the existing TCP/IP infrastructure, and can be deployed incrementally (one network at a time). However, for a given network, network ingress filtering must be implemented by nearly every entry point of that network to be effective.
13. Implement Spoof Prevention	Modify protocols or their implementations to be more resistant to spoofing (forging "from" information). This greatly reduces the number of intermediate systems that need to be examined, but often protocols and/or implementations cannot be easily modified to do so.

Technique Name	Technique Description
14. Secure Hosts/ Routers	Secure hosts and routers to reduce the number of innocent intermediate systems available to an attacker. This is needed in any case for computer security, but perfect security is impractical and this does not actually perform attribution – it merely makes the problem easier to solve.
15. Surveil Attacker	Directly surveil likely or known attackers. This counters sophisticated attacker techniques, but requires pre-existing knowledge of the likely attacker's identity, and some attackers are extremely difficult to surveil.
16. Employ Reverse Flow	Specially mark data flowing back to the attacker, and then have intermediate systems detect these markings. This can trace through stepping stones, but requires detectors of these reverse flows and may be thwarted by encryption.
17. Combine Techniques	Combine more than one technique. This is more likely to succeed than any one technique, but will generally cost more to do. There is little experience in combining techniques, and remember "garbage in, garbage out."

This paper does not claim that this is an exhaustive survey of all possible attribution techniques. However, it is the most complete survey available to date, and should be useful for future work and refinement of attribution techniques. A brief taxonomy of these techniques is given in the appendix.

The following subsections describe each technique. Each subsection describes the technique, provides specific examples, and closes with a brief commentary on the technique's key advantages and disadvantages. More specific instances of a technique are called "approaches" in this paper; there may be many different approaches for implementing a technique.

2.1 Store Logs & Traceback Queries

In the "store logs & traceback queries" technique, the transmitted messages (e.g., IP packets) are logged by routers as they go through a network. The messages may also be logged by the sending and receiving hosts. A log need not store an entire message, e.g., it may store a subset of information such as only the to/from information. A log need not store every message, e.g., it may store only initial messages between parties. To trace, a requester goes backwards, querying each possible preceding router if the message or something related to the message (like a pattern or hash) went through that router. Obviously, queries using this technique can only work if the necessary information to support the query has been logged.

Figure 2 illustrates how this technique works. Presume that the routers (labeled A through D) log all messages, and the defender is attempting to track an attack backwards to the attacker. Unfortunately, the attacker is employing a zombie to hide his originating source. The defender would query router A if the attacking message went through router A; router A would reply "yes". The defender would then query routers B, C, and D, since those routers are connected to the next network. Router B would reply "no", suggesting that the attack did not go through that route. Router D would also probably reply "no", since in most cases a zombie would change the message so that the connection could not be easily determined from a log. Router C would reply "yes", suggesting that the defender query further at that point. At this point, the defender has at least identified an intermediate network, and possibly the intermediate node on the network. The defender may even be able to backtrack further through the zombie, depending on the logged information.

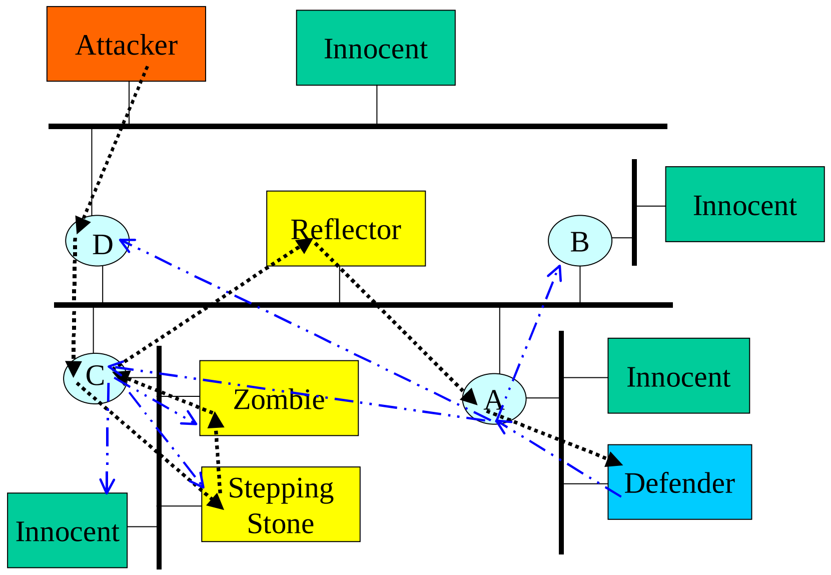


Figure 2. Store Logs & Traceback Queries Technique

This technique can be subdivided into two parts: logging and querying.

2.1.1 Logging

From the point of view of attribution, ideally every router would log every message and keep that log in perpetuity. In practice, this is undesirable: such logging may have unacceptable performance, storage space, or privacy implications. There are three ways to rectify this situation:

1. *Limit the number of messages logged.* For example, store only the data destined for an especially sensitive destination, or only packets that appear "suspicious."

2. *Limit the amount of data stored about each message.* For example, store only such as connection information (e.g., to/from information for only the initial message of a session), only to/from information from each message, only a

subset of the message (e.g., an initial fragment), only hashes of the message, or only hashes of a subset of the message.

3. *Accept the undesirable implications.* For example, buy large disk arrays or massive memory arrays to store the log data, buy faster processors (or more of them), and accept privacy risks. Clearly, limiting the number of messages logged and the amount of data stored about each message is more desirable where possible.

The criteria for selecting messages to be logged, or the amount of data to be logged in each message, could be changed dynamically. Dynamically changing these values could be accomplished by connecting these values with intrusion detection systems; see the forward-deployed IDS discussion below.

What data is stored, and how it can be retrieved, also has legal and privacy ramifications. In some circumstances, recording or retrieving information about a message (such as from/to information) may be considered different than recording or retrieving the message itself. In traditional telephone systems, it's possible to obtain (with a warrant) information on who a suspect has called (pen register) or who has received calls from the suspect (trap and trace). Warrants for pen register/trap and trace (PR/TT) information (as it's called by the law enforcement community) are often easier to acquire than recordings of the actual message traffic, which enjoys stronger legal protection.

It may be easier to store message hashes than messages themselves, since hashes are usually far smaller than the messages they hash. This is especially true in higher protocol levels, where one higher-level message may be implemented by many lower-level messages. Storing hashes instead of the actual data is probably more palatable legally and socially as well. Since the data itself isn't stored, hashing only supports attribution for data the requestor has already seen, and does not reveal the data itself. Storing hashes (instead of actual data) appears to be more akin to PR/TT data than to a recording of the message, though it is unclear if the courts will agree to this viewpoint.

One useful approach to logging events is logging authentication records for every authentication event (e.g., host login, ftp/http login, etc.), along with information such as the network address of the requestor. Reservations of resources (e.g., DHCP) are also useful for attribution. A log could be kept of every email received and sent (including the IP addresses of the other party exchanging mail, the from/to addresses, and a hash of the contents).

One problem with logging is protecting the logs themselves, particularly if an attacker gains administrative privileges over the system generating or storing the logs. A partial solution is to store logs on a separate machine from the machine performing the activities. As long as the logging system itself is secure, an attacker may be able to use other systems to append incorrect data but not remove correct data.

[Sager 1998] describes capturing flow information from Cisco routers that may be useful for attribution. This information includes (reported) source and destination IP addresses and ports, number of packets and bytes, IP protocol, and TCP flags.

One unusual solution for high volume IP packet-level logging is named Source Path Isolation Engine (SPIE). In the DARPA-sponsored SPIE approach, IP packet hashes are stored using a Bloom filter to store the information efficiently (see [Snoeren]). The SPIE approach dramatically increases the amount of data that can be logged, but even then, on high-rate routers (e.g., IP routers on an Internet backbone) this is still difficult and expensive for software-only implementations to perform in terms of memory size and memory performance. However, on lower-speed software routers (or where only a subset of packets are logged) this is not as difficult. [Sanchez 2001] describes the design of a hardware implementation of SPIE, and suggests that this would make SPIE practical at high speeds for relatively small amounts of money.[4] An Internet draft discussing traceback protocols, including SPIE, is available [Partridge 2001]. The developers of SPIE are working with the Internet Engineering Task Force (IETF) working group on traceback protocols and appear to be actively continuing the work.

Most of these approaches presume that logging is decentralized, and queries are made against the logs later. Logs are often decentralized, because the overhead of transmitting logs to a central location, and then performing analysis at a central location, does not scale well to very large networks. Nevertheless, there are those who have established centralized logging facilities for at least a portion of logging data, and then use the resulting information for attribution. For example, the Distributed Intrusion Detection System (DIDS) is a host-based approach that attempts to track all users in a network. Each monitored host sends abstracts of the audit trail to a centralized DIDS director for further analysis. Note that DIDS establishes a "Network-user ID" (NID), and audit records of session starts (logins) are sent to the central DIDS director. As a result, DIDS is able to track users moving through the network using normal logins when inside the DIDS-covered network [Snapp 1991a, 1991b, 1992, Ko].

A far more extreme version of this approach is the first approach suggested in [Arkin 2002]. In this approach, all public network traffic is logged for a period of time, and later the log of all network traffic is searched. First, a query for the network-visible pattern is requested, and then queries to find any transmission of the source code that caused the pattern are made. The presumption is that attacks tend to be tested at first in smaller regions, and so by identifying early attack tests it may be easier to identify the attacker. Searching for early versions of the source code may also aid in identifying the real attacker. Finally, profiles of attackers are built up (e.g., based on unusual approaches or rootkits), so attacks on different targets can be correlated to help identify the attacker. This approach depends on a massive storage system monitoring and logging data from

[4] Software that implements SPIE, and related papers, are available at http://www.ir.bbn.com/projects/SPIE. The software is open source software/free software under an MIT-style license. This software was developed for FreeBSD-4.3 and Linux-2.4.2, but it can monitor packets from arbitrary operating systems and the approach should easily apply to other operating systems.

the entire Internet (or a significant portion of it). The legal and social issues of such a system are not addressed in Arkin's paper.

2.1.2 Querying

Querying can be performed manually (e.g., using telephone calls and email to upstream routers) or automatically. Manual querying is currently necessary in most cases, and may always be necessary for querying in some locations not under a requestor's control. Supporting manual querying requires an efficient way to identify the point-of-contact for each router; existing databases (e.g., WHOIS for IP routers) sometimes provide this information, but techniques to ensure their validity would help, and not all protocols have a system for identifying points of contact.

However, manual querying is necessarily limited to slow human response times. If more rapid response is needed, then querying must be automated. To support automated querying, a protocol is needed to query "upstream" logging systems.

[Sterne 2001] [Schnackenberg 2000] describes the Cooperative Intrusion Traceback and Response Architecture (CITRA), based on the Intruder Detection and Isolation Protocol (IDIP), which can perform this service. Note that in CITRA's case, the "CITRA-enabled Linux routers in our testbed perform traceback by creating audit records for network flows on an ongoing basis and examining them for attack path evidence when presented with a traceback request." IDIP was developed by NAI Labs, Boeing Phantom Works, and U.C. Davis under a series of DARPA contracts.

[Sterne 2001] also references AT&T's work on "Aggregate-Based Congestion Control (ACC) and Pushback," which proposes a similar inter-router signaling protocol, and mentions other similar approaches such as Arbor Networks' and Recourse Technologies' ManHunt.

Note that some protocols that query "current state" could be modified to also examine logs instead. For example, the Session Token Protocol (STOP) [Carrier 2002], described later, could be modified to examine logs and not just the current state of the system.

[Asaka 1999a, 1999b, 1999c] takes a different approach to performing queries. Instead of sending a query to the system containing the logs, the "manger" dispatches a mobile agent called the "tracing agent" to the system where tracing is to occur. The tracing agent activates an information-gathering agent, which collects information from the system log, and investigates the point of origin of the Mark Left by Suspected Intruder (MLSI) based on accumulated data about the network connection and processes running on the system. Note that this approach is actually a hybrid of the "store logs & traceback query" technique and the "insert host monitor functions" technique discussed below, since the mobile agent is inserted into a running host to perform queries on the current state of the system. The tracing agent then repeatedly moves to the next target system on the tracing route, activating a new information-gathering agent. This approach has the advantage of reducing bandwidth use (since entire logs are not transmitted), but requires that various systems accept and execute mobile agents to examine these logs.

Note that querying could respond with an actual answer, but an alternative is to simply store the answer. If an answer is stored, a router would usually need to automatically send a “please store the information” query up further when it *has* seen matching information, to recursively acquire and store the information. There is also a need for a separate querying mechanism (which may be manual) to actually retrieve the stored information. Storing answers for later use resolves the problem that complete logs cannot usually be stored for a long time.

As with all the other entries here, querying is not limited to IP packets. Also, querying can be especially useful for authentication servers. An example would a query to ask an ISP what user is currently allocated a given IP address (if users are authenticated and then dynamically allocated an address through protocols such as DHCP).

2.1.3 Advantages and Disadvantages

This technique—logging and later querying—is easily applied to a wide variety of circumstances. The approach is widely implemented for host logins (with manual querying), and many of the references above discuss implementation approaches for implementing the technique for IP packets. Indeed, most authentication systems can easily support log & query, aiding later attribution.

A major advantage of these systems is that they support after-the-fact attribution. In other words, an attack can have already completed before the attribution process begins, and some attribution information may still be gleaned. In contrast, many other attribution techniques do not support after-the-fact attribution.

However, there are also many disadvantages. Log & query systems *must* be pre-positioned to perform logging of the relevant data before the event. Since it is difficult to determine ahead-of-time what will be relevant, this leads to storing large amounts of data in the logs about each event, in case it might be relevant later. The combination of a busy network and large amounts of data per event quickly leads to large logs, resulting in large costs (to store the data) and performance overheads.

2.2 Perform Input Debugging

Unfortunately, the term “input debugging” has acquired a number of related meanings. For purposes of this paper, the term “input debugging” describes a process where upstream routers (that is, routers one step closer to the attacker along the attack path) are given a pattern (e.g., a destination address or attack signature) by the victim, and asked to report the next time they receive messages matching that pattern. This pattern is sometimes called a *signature*. The term “input debugging” is used in the literature because the process is sometimes viewed as being similar to program debugging.

This technique is in some ways similar to intrusion detection systems, but note that in input debugging the pattern is only given after the attack has begun. Also, in input debugging, the pattern is often a characteristic that is not by itself an attack (e.g., a destination address or port). Input debugging can be done manually, but manual debugging is time-consuming. Manual input debugging is especially time-consuming if it is necessary to gain the cooperation of upstream routers outside the organization.

It is possible to do input debugging one step at a time, by going back one step to all possible routers and making an input debugging request, and when the "correct" path is located, to go back one step in that direction. An alternative is to "flood" all possible directions (say to a certain depth) with the request. Flooding, however, can overwhelm an infrastructure, and supporting such approaches can be especially hard to secure if automated.

[Schnackenberg 2000] proposed using Intruder Detection and Isolation Protocol (IDIP) to facilitate this kind of interaction between routers involved in traceback (note that IDIP can support other kinds of interaction as well). [Cisco 1] and [Cisco 2] describes Cisco router capabilities for supporting input debugging. Many Cisco routers have the "log-input" command that can aid finding one hop back, and many models also have a command "ip source-track" that enables IP source tracking on all line cards and port adapters for the IP address of the stated host. Note that to use these capabilities, the Cisco Express Forwarding (CEF) option must be enabled. This is not especially limiting for users of Cisco equipment, because most high-end Cisco routers on the Internet (e.g., running as backbones) run CEF or distributed CEF (dCEF) for performance gains. [Thomas 2001] describes how to use Cisco routers and their "NetFlow" capability to trace back to an attacker's entry point. This approach also only works well if CEF or distributed CEF (dCEF) is enabled. An older perl script named DoSTrack automates this approach on Cisco routers, but it does *not* work if CEF is enabled. DoSTrack is no longer maintained [Stone 1999].

A different approach is described in [Van 1997], which presupposes the use of an "active network." In an active network, packets can include programs for routers to run, along with regular data. In this approach, tracing back is performed by programs sent backwards on the network to the "previous" router, repeatedly going back to trace an attacker by running a program to implement the pattern matching. [Van 1997] only discusses its use in halting an attack, but it appears that it could also be used for attribution. However, the ability to cause arbitrary programs to run in routers is likely to create many new vulnerabilities. Examples of new vulnerabilities include network-wide denial-of-service if a packet storm can be started, and entire networks could be taken over if an attacker can insert malicious programs into routers. In addition, this approach is likely to severely impact performance. Thus, this approach has many dangers and may be impractical.

[Dunigan 2001] describes a small prototype testbed using separate "tracer daemons" on each (local area) network instead of tracing on the routers themselves. These tracer daemons could control their "downstream" router, and could query back one hop to the previous "tracer daemons." Communication could be secured using common protocols (such as the ssh protocol). This approach enables tracing *without* modifying the routers themselves, but the paper notes that it would be better if this functionality were integrated into the routers themselves.

[Ioannidis 2002] discusses “pushback” of aggregate-based congestion control (ACC), to rate-limit certain signatures and send those limits back. Further information is in [Floyd 2001]. The primary purpose of the approach is to support attack response, not attribution, but the system does send messages back downstream and could be modified to support attribution.

Input debugging is a common approach for handling distributed denial of service (DDoS) attacks today. Today it is often implemented manually, but manual implementation limits its scaleability. Even when automated, however, input debugging has its own disadvantages. Since this approach has no memory and is reactive in nature, it cannot find past attacks or attacks that do not continue to send data. Instead, it is only fully effective against continuously streaming attacks.

2.3 Modify Transmitted Messages

Another attribution technique involves modifying messages (e.g., packets) as they go through a network to aid attribution. Typically, this involves the various routers modifying the message to include identification of each router the message went through. In some sense, this approach is like logging, but the log entries are sent as part of the message.

There are many variations on this theme:

1. *Node append.* In the node appending approach, information is added to the message header (e.g., IP packet header) specifically identifying each router the message passed through. In short, every router adds an identifying marker saying, “I saw this packet” before retransmitting the data. This is conceptually similar to the record route option of IP. At the IP packet level this approach is an extremely expensive operation (in terms of performance); it often disables hardware support and rapidly increases packet size. Some of this approaches’ disadvantages (when applied at the IP level) are discussed in [Doeppner 2000], where it is termed “deterministic router stamping.” Thus, ways to narrowly select the relevant packets can be helpful when applying this approach at the IP layer.

   Note that unless the information is authenticated in some way, attackers can significantly weaken this approach. The Deciduous approach [Deciduous, Chang 1999, 2000, undated] uses IPSEC’s Authentication Headers (AH) to identify at least some of the routers along the way; in this case, AH is used to create an authentication mechanism.

   This approach is easier to apply at higher application protocol levels, where there is usually more data in a given message. For example, SMTP mail transfer agents (MTA) for normal Internet email already add this routing information when they forward email (though they do not normally authenticate this information).

2. *Algebraic encoding.* Existing space in a message header can be used to provide the attribution information through some sort of encoding. This is a useful technique when the message size should not be changed. In IP, the Quality of Service (QoS) area is sometimes used for this purpose since many organizations do not use the QoS information. Encoding routing information into the packet in place obviously limits the amount of information that can be inserted. [Dean 2001] discusses a technique that uses algebraic encoding; their implementation stores the encoded value in the IPv4 "fragment id" field (and thus interferes with IPv4 fragmentation). Note that the approach in [Dean 2001] emphasizes the algebraic technique, and the approach could be still be used by storing information in other locations; their generalizations involve randomization, which is similar to the PPM concepts described next.

3. *Probabilistic packet marking (PPM).* In the probabilistic packet marking (PPM) approach, a router randomly determines whether or not it should set information about the message's route into a given message. The defender can then use a set of messages to determine the route. Note that in most circumstances a number of messages must be received before attribution can be made. See [Savage 2000 and 2001] for an approach, including an encoding approach. Note that there are some special difficulties with PPM if an attacker attacks the mechanism itself, e.g., see [Park 2000]. However, attacking the mechanism itself might provide warning of an undetected attack. See also [Song 2001] for techniques to improve the reconstruction of paths and authenticate the encodings; [Lee 2001] also examines the approach.

4. *Router Stamping.* Router stamping is described in [Doeppner 2000]. It is very similar to probabilistic packet marking, in that a router randomly determines whether or not to mark a given packet. However, if a router chooses to mark a packet, it then randomly chooses one of a fixed number of slots in the packet that can be used for marking. To counter forgeries, administrators may tell a given router to change its marking for any particular target; incorrect markings are revealed as forgeries.

Note that instead of marking messages at each router, a network can be established and mark only at the entry into or exit from that network (with each entry point being marked differently).

CS3's MANAnet[5] (based on previous DARPA work) implements modifications to the IP header structure to record the paths over which packets are being transmitted (e.g., in an organization's large intranet). MANAnet assumes that a typical router has no more than 16 paths; if this is true, an entire path can be coded using 4 bits per hop, by having each router add the path on which the packet was received to the list. In addition, the IP address of the first router to admit the packet to the network is recorded in full, adding

[5] See http://ww.cs3-inc-com/mananet.html (Verified as of 8 Feb 2002)

another 4 bytes (in IPv4). If the maximum number of hops to be expected to be less than 32, the entire path can be recorded as a 20 byte expansion of the header. For the purposes of attribution, the main component is the “MANAnet Router” which implements a proprietary modified form of IP called Path Enhanced IP (PEIP); each MANAnet Router adds the place from where it got a packet to the packet before it forwards it. The approach is reviewed by [Dietrich], who notes a number of problems with the approach. This includes near-universal imposition of ingress filtering before it can even be deployed, modification of IP headers with support by all routers, a resource forecasting or reservation mechanism, and a means for determining the real source of packets. In particular, it questions whether the computational techniques can be implemented in the fast paths of backbone routers, as well as concern for a proprietary modified form of the open IP standard.

This technique has the advantage of not requiring separate logs and a separate query system, and thus attribution information is immediately available. Also, there is no need to store and manage logs.

The technique also has many disadvantages. One obvious disadvantage is that this approach can greatly increase bandwidth requirements and/or severely reduce performance. This is not only due to the increased data and processing requirements; these approaches may disable hardware and software optimizations employed by some routers. There are many situations where changing a message is impractical, for example, it may defeat authentication mechanisms since those mechanisms may detect an “unauthorized” change to the message. Ensuring that this data is correct (and not from an attacker) is also difficult. Note that this approach requires the cooperation of remote routers to add the necessary information, implying the need for a standard way to insert this data.

2.4 Transmit Separate Messages (e.g., iTrace)

An alternative similar to modifying transmitted messages is to have routers send separate messages that can be used to support attribution.

The IETF’s ICMP Traceback Working Group has been working on one such approach called “iTrace,” which sends traceback messages a small amount of the time (e.g., 1/20,000 packets); their website is at http://www.ietf.org/html.charters/itrace-charter.html. See [Bellovin 2000] where the iTrace (ICMP traceback messages) are discussed.

[Mankin 2001] describes an “intention-driven” modification to iTrace; an iTrace message is only sent towards a path that has registered an “interest”, and that interest information is shared using a minor modification of the standard Border Gateway Protocol (BGP). Indeed, the paper argues that the simple original iTrace proposal was essentially ineffective without this extension, although this modification requires more effort to implement. [Mankin 2001] also experimented with a heuristic that preferred longer paths, to increase the probability that more useful path information was sent, and the results appeared very encouraging and to improve the usefulness still further. See also [Wu 2001] for more information.

Note that these approaches could be modified to raise the probability in certain cases. For example, a router could decide that anything sent to a sensitive domain (e.g., “.smil.mil” in a faraway external router) could have a 100% chance of a trace message also being sent. Note that the trace messages could go through a different path or network (e.g., to prevent interception or to increase confidence in the trace message).

These separate messages supporting attribution could be directly received and processed by the recipient of the original message. However, they could also be observed by routers and systems observing the network.

An advantage of sending separate trace messages is that, since the original messages are not changed, many of the complications of changing a message are avoided. For example, hardware accelerators in routers often work with this technique (they are sometimes disabled if the transmitted message is modified)., and since the original message is not modified, the approach will not interfere with authentication of the original message.

However, using separate messages has disadvantages as well. Since the messages supporting tracing may be routed separately from the message being traced, extra effort is required by any implementation to associate the trace message with the message being traced. The technique could easily overwhelm network resources if a single message becomes the original message plus a message from every router the original message encountered. This is particularly true if trace messages could trigger tracing themselves; if this is possible, cascades of messages could exponentially overwhelm network resources, but if trace messages are not themselves traced, attackers may work to make their attacks look like trace messages. Practical implementations must only send separate trace messages in special cases (such as the low probabilities suggested by iTrace). In other ways, the advantages and disadvantages of transmitting separate messages are similar to modifying transmitted messages.

2.5 Reconfigure & Observe Network

Another attribution technique is to reconfigure the network, observe what changes, and use this information to identify the source or a route back to the source. The reconfiguration could be direct (e.g., changing data in a router table) or indirect (e.g., performing an action that significantly changes the network behavior).

For example, Burch and Cheswick [Burch 2000] describe a particular implementation of the technique called “controlled flooding.” Controlled flooding floods candidate upstream routers, and then watches for variations in the received packet flow to determine if that router is in the path of the attack. However, this approach could be viewed as a DoS attack on those routers. Thus, “controlled flooding” is an interesting approach but one that should only be used in specialized circumstances. One major advantage of controlled flooding over most other techniques is that it can work on routers that are *not* prepositioned or coordinated to support attribution [Savage 2000, section 2.2.2] [Savage 2001]. Also, controlled flooding is only effective for continuously flowing attacks, such as typical DDoS attacks.

Another variation on this theme is Centertrack as described by [Stone 1999]. In this approach, an overlay network is created that links all ISP edge routers to a central tracking router (or network of tracking routers). Dynamic routing is then used to redirect all packets destined for the victim so that they will then go through the tracking routers; then hop-by-hop approaches are used to find the source. Centertrack is identified as a "defunct research project" in [Dittrich].

Active networks also fit into this category. Active networks permit programs to be sent to network infrastructure components (e.g., routers), which then run and change the network behavior. Once example is [Sterne 2002], which uses the secure ANTS execution environment. The paper also discusses many other active network activities and related papers.

An advantage of this technique is that, if the network being reconfigured is large, it may be possible to very rapidly identify the attacking source. However, the technique also has disadvantages. Direct control over a large network can be difficult to implement, and can create its own security vulnerabilities. Indirect approaches (such as controlled flooding) can be viewed as an attack, and thus should only be used in limited situations.

2.6 Query Hosts

Another attribution technique is to query hosts about their state. This includes using existing services to perform such queries, or adding new services to hosts to support such queries.

An authorized administrator can use existing host tools (such as standard administrative tools) to monitor an attacker in certain circumstances. These functions may not succeed if the attacker controls the host, because the attacker may subvert the tools used to do the monitoring. This is particularly a problem for administrative tools normally on the host; an intelligent adversary is likely to know of such tools and take steps to subvert those them.

[Carrier 2002] describes Session Token Protocol (STOP), an approach for traceback *through* a host toward its sender. The STOP approach extends the standard "ident" service to permit external entities to query about active processes. Once a query is sent, the system then examines all incoming connections to that set of processes, and can recursively trace back to previous hosts. STOP has the nice property of being an upward-compatible extension to an existing service and is able to rapidly backtrack to ordinary TCP/IP connections. STOP has been implemented, but only for Unix-like systems (there is no known reason the technique could not work for other systems such as Windows). STOP intentionally stores information about the traceback, and merely returns a hash sufficient to request the real data at a future time; this is an effective technique for supporting privacy, but this approach does limit its capabilities (the requestor cannot control the recursive traceback, so an attacker may thwart simple attempts to trace back, and an attacker might easily be alerted that a traceback is occurring). It should be easy to extend STOP to immediately report its data to a trusted third party (such as a CERT), presumably using authentication and encryption. When we contacted the developers of

STOP in August 2002, we learned that there are no plans to commercialize STOP and that the authors of the paper have not worked on STOP since they wrote the paper.

An advantage of querying hosts is that it can quickly provide information. A disadvantage is that an attacker may control this information, making the information far less reliable. Also, this technique presupposes that a query function already exists. If there is no pre-existing query function, then one will need to be inserted, a technique is described in the next section.

2.7 Insert Host Monitor Functions (e.g., "Hack Back")

If a host does not include a service that provides needed attribution information, someone could add such services after an attack has been detected.

An authorized administrator can add host monitoring tools in certain circumstances, sometimes using standard administrative tools. These tools may not succeed if the attacker controls the host, because the attacker may subvert the tools used to do the monitoring. One alternative would be to clean the compromised system and add monitoring functions during that cleaning process, but this may thwart efforts to attribute an ongoing attack.

[Asaka 1999a, 1999b, 1999c] transmits mobile agents (termed "tracing agents") into systems when a trace is to occur. The agents use accumulated data about the network connections and system processes to trace backwards toward the attacker.

[Jang 2000] includes monitoring functions on a host system. If an attacker uses the host system to break into another system and acquires administrator privileges, the host system surreptitiously sends modified versions of key programs to the other system, modifying the other system to also monitor the attacker. Tracing information about the attacker is thus transmitted forward to some of the other hosts on the attack path.

A harsher and more controversial technique is to break into a host machine or series of host machines (termed by some a "hack back"), usually going backwards toward the attacker. [Staniford-Chen 1995b] reports that the U.S. Air Force has used this approach, calling it "Caller ID," to track down and arrest an intruder. Caller ID is based on the belief that if an attacker goes through intermediate systems to make an attack, there is a high probability that the intermediate systems have known vulnerabilities (including the vulnerability that the intruder used). The defender, knowing the same attack methods as the attacker does, can simply reverse the attack chain. If the attacker goes through Host_0, Host_1, Host_2, … Host_n (where Host_n is the system actually being attacked), the defender can break into Host_{n-1}, use that system state to find the next connection back, then break into Host_{n-2} and so on back toward the attacker. [Jayawal 2002] discusses some of the technological and legal issues related to "hack back".

An advantage of inserting host monitoring functions is that it can provide valuable information on the host's state. However, a disadvantage is that any attempt to insert a host monitoring function may be noticed and/or countered by the attacker. It is often best if the attacker would not notice the additional monitors when they are inserted, but it can be a challenging task to add monitors without revealing them to a capable attacker.

The "hack back" approach has many additional disadvantages. Fundamentally this involves a number of complex legal issues. [Staniford-Chen 1995b] reports that performing this activity required special permission from the Department of Justice. It is an extreme measure with many social issues, such as privacy concerns. This is especially true if the counter-attack is performed by anyone other than the host owner or authorized administrator. There is also the possibility that an intermediate system cannot be broken into (perhaps the attacker used an attack not known to the defender, or the attacker improved the security of the systems they broke into). There is also the danger of accidentally damaging intermediate systems. If hack back is implemented manually, the attacker may be gone long before the attribution can succeed. If hack back is automated, there is the danger of going awry (either by programming error or by malicious misleading data provided by an attacker). In short, hack back is an approach with a large number of important disadvantages [Staniford-Chen 1995b].

2.8 Match Streams (via Headers, Content, and/or Timing)

Instead of trying to trace through every router and host in a network, it may be possible to observe the set of streams entering and exiting some network or system and determine, externally, which input streams match which output streams. This technique is often referred to as "stream matching."

Figure 3 illustrates this technique. In this example, there are a number of data flows (A through F) that enter or leave through a few specific links or ports of a network or host. The goal is to use externally-visible information about those flows to determine which incoming flows match which outgoing flows. In this example, flow A enters the network or host and re-emerges as flow E; flow D enters and re-emerges as flow F. Not all flows need match; flow B enters but does not leave, while flow C originates from the host or network (and has no corresponding entering flow).

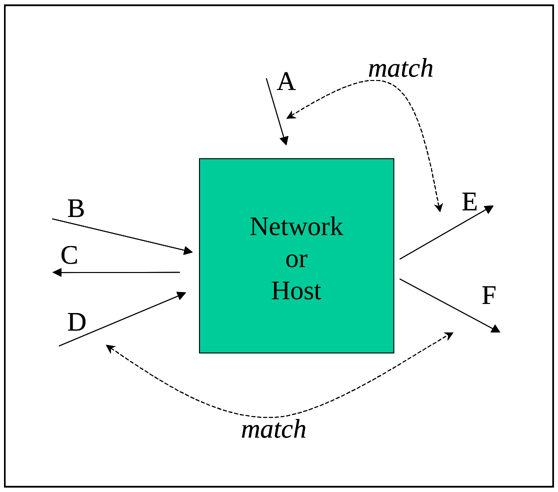


Figure 3. Stream Matching

Stream matching techniques can be further divided into techniques that examine message headers, message content, and message timing. Combinations of all three approaches are possible as well.

2.8.1 Stream Matching using Message Headers

In this approach, the headers of messages entering and exiting a network or host are examined to determine which incoming streams match which outgoing streams, and thus determine the source of a stream being examined.

[Yoda 2000] uses time stamps and headers of the packets, in particular the increase in sequence codes, to determine if one flow “matches” another flow. This approach is more effective if the intruder is manually inputting commands by hand (instead of transferring large files). [Yoda 2000] Only describes how to use the approach to match telnet or rlogin flows. The TCP sequence numbers are determined by data content length, so Yoda’s approach could be considered to involve examining message content as well. Yoda’s approach will not work if compression of the data stream occurs differently on each part of the chain, nor will it work well if link encryption is used (because the TCP headers will be encrypted).

2.8.2 Stream Matching using Data Content

In this approach, input and output data content of streams are examined to see if they match. Note that this approach will usually fail if the stream is encrypted “inside” the network/host, since the data content will be encrypted into a different value.

[Staniford-Chen 1995a, 1995b] developed a technique called “thumbprinting” that divides the stream data into discrete time intervals, creates digests of the packets within the interval, and computes the similarity of stream digests to determine if one stream matches another. [Buchholtz] reimplemented the approach and comments on it. Initial results appeared very promising.

[Mansfield 2000] discusses characterizing intrusion attempts by first identifying some specific noise or other indications of the attempt. These characterizations might include TCP-RESET packets, ICMP echo-response, or destination/port unreachable packets. The observed traffic pattern is then matched with traffic patterns collected from the connected network links. The communication between the various managers and agents is carried out using the standard SNMP management protocol. It could be argued that these are forward-deployed IDSs, but the approach emphasizes matching a particular stream of actions to an attack closer to the defender. Based the definitions used in this paper, this is considered a stream matching approach.

2.8.3 Stream Matching using Timing

In this approach, timing is used to match streams, by determining if the timing of the streams suggests a causal relation.

[Zhang 2000] discusses one approach (identifying times where there is a transition to and from an idle state), as well as comparing it to some content-based approaches. Zhang also noted that intermediate hosts are widely used for a number of legitimate purposes, so using a naïve rule like "any match is an intrusion" would result in a massive number of false positives.

[Ohta 2000] uses distributed sensors to capture error messages such as ICMP's "destination unreachable" as a way to detect network scans. The sensors then use timing correlations to determine the source network. Thus, this approach combined forward-deployed IDS sensors with matching by timing.

[Lee 2002] mentions that researchers in Purdue are looking at this approach. In some approaches, a stream's timing could be intentionally perturbed (e.g., intentionally inserting delays on a particular stream) to see if it has an effect; if there is a change, this increases the confidence of a match. [Staniford 2000] also notes the approach of matching timing, using a variety of signal analysis techniques. [Wang 2002] also discusses the approach.

Matching streams using timing can help identify stepping stones, but it is less effective for zombies. This is because a zombie's response can occur long after its command.

2.8.4 Advantages & Disadvantages

An advantage of stream matching is that it can aid attribution without requiring knowledge of the internal state of a host or network. However, actually matching streams is a difficult challenge. In particular, the technique tends to have difficulties with encryption and zombies because they hide the information used to match streams.

2.9 Exploit/Force Attacker Self-Identification

In some cases it is possible to use information the attacker sends (intentionally or not) to identify the attacker, or force the attacker to unintentionally identify themselves. This is a broad technique, with a number of specialized approaches. Some of the approaches to implementing this technique employ:

1. Data intentionally included by the attacker. For example, spam messages often include information on how to buy a product. Instead of trying to track down the system used by the spammer, it may be simpler to find the spammer (or whoever paid the spammer) by using the information for buying the product.

2. Self-identifying protocols. Many protocols and file formats include identification marks of some kind, such as the "Windows ID" [Ricciuti 1999] or CPU ID [Miles 1999].

3. Beacons. For purposes of this paper, a "beacon" is a tool, inserted by a defender into an attacker's environment, that causes the attacker to

unintentionally identify themselves when the attacker performs some action. The action is not necessarily an attack; the action simply needs to be an action that the attacker would perform. One example of a beacon is a "web bug." In a web bug, an HTML page includes a remotely-referenced transcluded[6] value such as an invisible image (typically a transparent 1x1 pixel image). If the attacker views the HTML page using a typical browser, the browser will automatically attempt to download the remotely-referenced value. The victim can then attempt to attribute any attempt to reference the transcluded image [Smith, Wheeler 2002]. Note that these techniques require that the attacker acquire the data/beacon and a triggering technique.

4. Cookies. Many protocols send information to a client, and expect the client to return that information later. For example, the HTTP protocol used by the World Wide Web includes support for "web cookies." In some cases, it may be possible for a victim to send a web cookie out to the attacker, causing the attacker to unintentionally identify themselves later. Many organizations have expressed concern over web cookies [Junkbusters 2002] [EPIC 2002].

5. LoJack™-like tools. It may be possible to include, or insert, a program that can respond to later queries. This can be effective, for example, when trying to identify attackers using a stolen laptop; it may be possible to embed an identifying message that the laptop sends (possibly triggered by some other action) that would enable a victim to track down the location of the attacker. A widely-published example of this approach involves R.D. Bridges, who tracked down his sister's stolen iMac using the remote access software Timbuktu [Cohen 2002].

6. Watermarking. In this approach, an attacker receives data that, while not active, enables later identification that the attacker truly is the attacker because that data is unique in some way. For example, [Johnson 1999] provides an introduction to recovering watermarks from images.

An advantage of this technique is that when it works, it can directly reveal the attacker regardless of the number of layers and indirect systems the attacker uses to foil attribution. However, many of these techniques depend on highly technical and specialized approaches that are often easy to foil once an attacker knows about them.

2.10 Observe Honeypot/honeynet

Honeypots and honeynets are systems that appear to be normal systems, but in fact are never used by the defender for normal purposes. Thus, any use of the honeypot or

6 Transclusion is the act of quoting another document on the network, without having to actually copy and paste the content. With transclusion, an author can quote original source material on the web, from the server on which it resides. The term originates from Ted Nelson's Xanadu project, one of the precursors to the World Wide Web. A brief discussion about transclusion is at http://www.theobvious.com/archive/1996/07/08.html.

honeynet is by definition use by an attacker. Although definitions vary, for purposes of this paper a honeypot is a single system, while a honeynet is network of honeypots. Note that the Honeynet project defines the term “honeynet” more narrowly: in their definition, a honeynet must use “real” (not simulated) systems and must only be used for observing (not reacting to) attackers. Early experience with honeynets can be found in [Stoll 1990, Cheswick 1992]. More recent experience from the honeynet project can be found in [Honeynet 2003].

For purposes of this paper, a honeysystem is either a honeypot or a honeynet. Honeysystems often perform simulated activities (to make them look like a normal system) and are specially monitored to observe attacker actions. Honeysystems are typically used for intrusion detection.

In the context of attribution, honeysystems can reveal attack paths in ways unexpected by an attacker. In particular, a honeysystem can be used to immediately reveal the presence of a zombie if the zombie is placed in the honeysystem. An attacker would expect a zombie to delay its attack, but the mere presence of the zombie can reveal the attacker. Honeysystems are typically specially instrumented and monitored, and these additional monitors can be used to directly identify attack paths for attribution.

A significant advantage of honeysystems is that they can aid in countering zombies, as well as speed tracing backwards in any traceback process (due to their instrumentation). However, to work well, honeysystems must be monitored and their results analyzed, which often requires significant expertise. Also, honeysystems only work for attribution if the attacker chooses an attack path through a honeysystem.

2.11 Employ Forward-deployed Intrusion Detection Systems (IDSs)

In this approach, intrusion detection systems (IDSs) are placed as close as possible to potential entry points of attackers. Typically, IDSs are deployed in a defender’s location, to maximize detection of an attack. In contrast, a “forward-deployed” IDS is placed further away from the defended system and closer to the attacker, to maximize attribution information. This paper uses the term “forward-deployed” as an analogy to a military deployment; forward-deployed units are the units intentionally placed closer to an enemy.

Due to the location of their sensors, when forward-deployed IDSs trigger they provide much better information on the attacker’s location than if an attack had to be traced back starting from the victim. In some cases, such systems can also implement automated responses (such as reducing bandwidth or disconnecting the network). Obviously, this technique only works if the IDS system can actually detect when an attacker performs a malicious action. Note that these IDS systems, as with any other IDS deployment, may be based on detecting specific attack patterns, detecting anomalous behavior, or both. For a survey and taxonomy of IDSs, see [Axelsson 2000].

Some attacks or attack patterns (e.g., SYN without ACK, a large number of multiple requests without responses in a request-response protocol) can be pre-positioned, and triggered when they occur. Some tools are specifically designed to detect patterns of known DDoS attacks, which helps in the case of a known DDoS tool; such tools include David Brumley's Remote.Intrusion.Detector (RID), the National Infrastructure Protection Center's TRINOO/Tribal Flood Net/tfn2k detection tool, BindView's Zombie Zapper, and Ramenfind [Dittrich].

Mazu Networks of Cambridge, MA makes hardware devices that it claims are able to detect and thwart DDoS attacks[7]. Mazu claims that their equipment performs a fine-grained traffic analysis (at gigabit network speeds on OC-12 links) and creates a statistical model of "normal" traffic. It then uses this model to detect anomalous traffic that suggests the presence of a DDoS attack. Reactive Networks "FloodGuard" and Arbor network's Peakflow DoS product appear to work on similar principles, by attempting to detect anomalous behavior upstream.

[Arkin 2002] describes multiple approaches for attribution (the first approach, logging all public network traffic, is discussed in section 2.1.1). [Arkin 2002]'s second approach is in essence a set of forward-deployed IDSs; Arkin has the notion of a globally-deployed set of IDSs, which look for known exploit attempts and/or anomalies to quickly identify attackers before their distributed attacks make it more difficult to identify the attacker.

[Templeton 2003] describes various techniques for detecting spoofed packets. For example, a time-to-live (TTL) value different from past values for a source to a destination may suggest a spoofed packet. Detecting spoofed messages can be particularly valuable for attribution.

Forward-deployed IDSs have far greater capability if the attack patterns being detected can be updated frequently and rapidly. However, this imposes complications. As with many other attribution techniques, the systems supporting the attribution must be protected. A forward-deployed IDS system tends to be more vulnerable to an attacker than many other tools due to its location. As a result, a forward-deployed IDS might be disabled, remotely controlled (to respond with forged information), or have its patterns revealed. Since an attacker may be able to control the IDS, there may be good reasons for not revealing all known intrusion patterns in a forward-deployed IDS, because this would enable an attacker to know exactly what attacks will not be detected. Reducing the number of attacks detected by an IDS reduces the IDS' effectiveness.

If the IDS tool can update its patterns rapidly, it rapidly also becomes an "input debugging" tool. When used this way, the IDS can detect pre-identified attack patterns, and if a new pattern becomes known, it can be used to detect the next occurrence. However, unlike input debugging, forward-deployed IDS systems can detect the initial occurrence of an attack and do not require multiple messages to begin attribution.

An advantage of this approach is that the IDS tool, if deployed sufficiently close to the attacker, can immediately detect attacks it is configured to detect, without the log

[7] Technical information about Mazu Networks was obtained from their web site: http://www.mazunetworks.com/solutions.html and http://www.mazunetworks.com/howitworks.html.

overhead required by log & query systems. However, there are disadvantages as well. This approach suffers from the problems of all IDSs: they are prone to a large number of false positives and/or false negatives, requiring constant surveillance. As a practical matter, the many false positives and false negatives mean that the alerts must be forwarded for logging and later analysis, greatly weakening the potential for rapid response in many circumstances and requiring specialized labor. Also this technique can be difficult to widely deploy without other information about an attacker (e.g., where they are likely to attack from) and is most effective when placed close to an attacker (which is difficult for external attackers).

2.12 Perform Filtering (e.g., Network Ingress Filtering)

Another technique to aid attribution is to filter messages so that certain links only permit messages to pass if they meet certain criteria that ease attribution. A receiver of a message that does not meet the criteria can then be assured that either the filtering was not successfully implemented (e.g., the attacker broke the filtering mechanism) or that the filtered link was not used.

A simple example would be a set of MTAs that reject email messages not signed by certain trusted senders. Another example would be modifying a network's architecture to remove links that stymie attribution and limiting what can pass over the remaining links.

The "perform filtering" technique can be specifically focused on supporting attribution by devising a network so that any message entering the network must have data that correctly identifies where it entered the network. This particular application of the technique, when applied at the IP packet level, is called the "network ingress filtering" approach. Due to its many advantages, the network ingress filtering approach is particularly emphasized in this section.

2.12.1 Network Ingress Filtering Definition

Network ingress filtering is an approach that restricts network traffic by requiring that all messages entering a network have a valid "source" value for that network entry point. Ideally, the valid values for different entry points should be non-overlapping so that any entering message uniquely identifies its entry point. Overlaps create ambiguity) but even in the presence of overlap this approach can be useful since the approach would reduce the number of possible entry points. Network ingress filtering for IP is defined in detail in IETF RFC 2827 [IETF 2827]. The ability to implement this kind of capability is recommended in the earlier IETF RFC 1812 [IETF 1812, 96]. Documents such as [SANS 2000b] describe how to implement network ingress filtering in more detail.

The network ingress filtering approach could be applied to protocols other than IP. For example, mail transfer agents (MTAs) could refuse to transfer email claiming to come from an invalid location (e.g., a "From:" or "Received:" entry inconsistent with the sender's IP address). However, since most experience with network ingress filtering has been with IP, the following text will concentrate on its use in IP.

2.12.2 Network Ingress Filtering Implementation

Implementing network ingress filtering of IP packets only requires simple packet filtering, a capability built into most of today's routers and a fundamental capability included in any firewall. Thus, implementation primarily involves reconfiguring the existing routers (or in rare cases, inserting firewalls) that connect other networks to the network being protected. In short, all connections to other networks must ensure that the source information is valid for the given connection.

Figure 4 shows a sample network ingress filtering configuration. In this example, the filtered network has connections to an external network through multiple gateways, as well as connections to many internal networks through major routers. All of those routers (E1, E2, E3, GW1, and GW2) are configured so that a packet's source address must be in a valid range to pass through those routers. Thus, in this figure, the attacker connected through router E11 who is attacking victim 1 must reveal that they are located in network E1, given certain caveats discussed below.

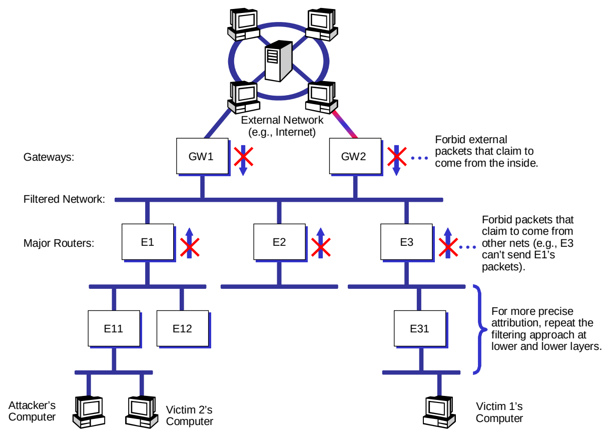

External Network
(e.g., Internet)
Gateways:
GW1
GW2
Forbid external packets that claim to come from the inside.
Filtered Network:
Major Routers:
E1
E2
E3
Forbid packets that claim to come from other nets (e.g., E3 can't send E1's packets).
E11
E12
E31
For more precise attribution, repeat the filtering approach at lower and lower layers.
Attacker's Computer
Victim 2's Computer
Victim 1's Computer

Figure 4. Network Ingress Filtering

For a typical wide area network (WAN) that connects to both the "outside" and to "internal" networks, two kinds of configurations are necessary to change an unfiltered WAN into a filtered WAN:

1. Routers connecting to the outside must forbid messages that try to enter the filtered network from the outside, yet claim to come from the inside. This is a generally accepted practice, as doing otherwise can permit many security problems, and is the fundamental rule implemented by even trivial firewalls.

2. Routers connecting to "internal" networks must forbid messages that try to enter the filtered network from the internal network, yet claim to come from somewhere other than that specific internal network. For example, if an internal network is allocated the IP address range 204.69.207.x (more

formally, 204.69.207.0/24), the router to the WAN must filter (remove) any packet coming from that network that did not use the allocated 204.69.207 prefix.

A specific example of such a WAN is the DoD NIPRNet, which currently has 21 gateways (equivalent to GW*x* in Figure 4) and approximately 1,500 "first-tier connections" (equivalent to the major routers E*x* in Figure 4). Consequently, turning the NIPRNet into a filtered network would require a minor reconfiguration of approximately 1,521 routers. Such reconfiguration is non-trivial, but these routers' configurations must be maintained anyway, and this reconfiguration can occur over time.

Network ingress filtering can be implemented on multiple connected networks (such as a larger WAN, all of the WANs below it, and some highly sensitive LANs below them). As more networks are filtered, network ingress filtering provides more and more precise attribution information.

The filtering rules only need to be sufficiently precise to identify the point of ingress into that particular filtered network. If a router connecting to an internal network supports a large number of valid prefixes, it does not need separate rules for each, merely a larger range that includes the valid prefixes and does not also include a range allowed by a different router. Requiring routers to have simple and non-overlapping IP address ranges is already a highly desirable property in TCP/IP network design, because allowing arbitrary prefixes leads to huge routing tables that can complicate administration and impose significant networking overhead. Thus, for most routers, the rules to support network ingress filtering can be fairly simple, with a few exceptions for special cases (such as networks that have moved from one region to another yet kept their old IP addresses).

If a given address can legitimately enter a network through multiple entry points, all of those entry points must permit it. This weakens the value of network ingress filtering for those addresses. This is the case, for example, when supporting some deployments of Mobile IP if the source IP address can legitimately move between entry points of the filtered network. However, even when weakened, network ingress filtering can still be useful for attribution. In such cases, the source address information is ambiguous, but it still reduces the number of possible entries to a small set (this information can be combined with other information to perform more detailed information).

Network ingress filtering requires that all (or nearly all) entry points of the filtered network implement filtering. There is the danger that filters will not be correctly configured or will be incorrectly configured later. One solution is to use automated test programs that attempt to send spoofed information, and then report a problem if they succeed. Such automated test programs are easy to create. The ORNL spoof testing service (at http://www.csm.ornl.gov/~dunigan/oci/spoof.html) is intended to become such a service, as is ICSA's NetLitmus (http://www.icsa.net).

There are other names for network ingress filtering, namely "egress filtering" and "reverse firewalling." These alternate names are used because of the way network ingress filtering is often implemented. In many cases network ingress filtering is implemented on a larger network that smaller local networks connect to. From the point of view of these local networks, this approach filters the messages leaving (egressing) the local network. Since network ingress filtering is often implemented using firewalls, but with rules preventing some messages from *leaving* the network instead of *entering* the network, the term reverse firewalling is also used. However, when the network ingress filtering approach is used on other network architectures these terms can quickly become misleading. Thus, the IETF refers to these filters as "ingress" filters, because regardless of the network architecture, network ingress filtering always filters messages entering the filtered network. This paper follows the IETF's naming convention as defined in [IETF 2827].

2.12.3 Network Ingress Filtering for Attribution

Network ingress filtering aids attribution, because it forces an attacker to reveal (in most cases) information about their network location.

More specifically, when attacked, a victim knows one of the following is true:

1. Network ingress filtering has not been implemented properly. There are simple tests that can be used to automatically determine if filtering has been implemented properly.

2. A filter (e.g., major router) and/or other networking component inside the network has had a security breach and consequently no longer filters correctly. In some cases testing can find this, but simple testing may not detect subtle breaches. There are ways to reduce this risk, such as using multiple filters in sequence, router monitors, and hardened routers. In addition, passive router monitors (implemented so that they cannot transmit back onto the network) can examine router inputs and outputs to detect possible leaks.

3. The attacker's location is so "close" to the victim that the attacker never passed through a filter. For example, see the case where the attacker attacks Victim 2 in Figure 4. In this case, the set of possible locations is obviously fairly small, significantly aiding attribution.

4. The message header's source information gives information on the location of the attacker. If the attack came from the outside, it will have an outside address; if it came from the inside, it will have an inside address identifying which inside location it came from, in the range forced by the filter(s). Usually, this will be a single entry point into a network, though if the filters have been weakened (e.g., to support some uses of Mobile IP), this may be a set of possible entry points.

Again, as multiple levels of network ingress filtering are added, the message header's source value will give increasingly precise information on the attack's network location.

2.12.4 Network Ingress Filtering Advantages

There are a number of advantages to network ingress filtering for attribution:

1. It is relatively easily implemented today at low cost with existing infrastructure. Implementation requires policies, configuration of existing filters (routers and firewalls) by network administrators, upgrades of components in rare cases, and simple occasional tests to ensure that the filtering is in operation. There is little to buy, existing personnel and products can be used to implement it, and since it is a relatively simple configuration, training costs are not expected to be high. This does not mean that it is trivial; any policy change requires coordination, and reconfiguring all routers on any major network is not trivial. However, it is less expensive than many other approaches, since many other approaches require deploying a large number of new hardware and/or software components as well as training personnel to learn new skills.

2. It can be deployed incrementally, with incremental improvements in attribution. This can be viewed in two ways:

    a. The routers on a given filtered network can have their rules modified in stages, instead of trying to modify all routers at once. Although the benefits of ingress filtering are primarily obvious when all (or nearly all) routers of a network do the filtering, even implementing the approach on a subset of routers can aid attribution by reducing the number of possible paths that must be traced back. Adding the rules incrementally can also aid in identifying the cause of any unintentional problem.

    b. Network ingress filtering can be beneficially implemented on a single major backbone and provide a benefit, and later deployed on additional networks to provide increasingly more precise attribution. This property – of not requiring all networks to be changed simultaneously – also simplifies deployment.

3. Implementation of network ingress filtering can be made a requirement for connection to certain networks. Thus, once a network ingress filtering regime is set up, it should be easy to maintain.

4. Network ingress filtering supports attribution without requiring message logs of unrelated messages or additional network bandwidth. Logging systems have complications due to the difficulties of acquiring data at speed, storing logs, and retrieving the correct data later. Techniques that send new messages or extend messages take additional network bandwidth. The filtering technique (including specifically the network ingress filtering approach) requires neither.

5. Network ingress filtering is generally transparent to users. Since the filtering rules simply enforce what users "should" do, users are generally unaware of the filtering; tools simply work as usual.

6. There are no known legal impediments. Since the filtering rules simply prevent forging of source addresses, there are no known laws that forbid deployment. In certain cases some state laws even forbid forging "from" information (e.g., some state anti-spam laws).

2.12.5 Network Ingress Filtering Disadvantages

Of course, network ingress filtering is not perfect. There are a number of disadvantages to network ingress filtering when used for attribution:

1. Network ingress filtering must be implemented by *nearly every* entry point into a given filtered network to be effective. If some entry points do not implement the rules, then any attack message *may* have also been sent through those entry points. If there are more than a few entry points that do not implement the filtering rules, the value of network ingress filtering for attribution goes down rapidly (unless supported by other techniques).

2. Networks that must support multiple entry paths, such as some uses of Mobile IP and permanently enabled backup routes, supply weaker attribution information for those messages. In the worse case, it can somewhat interfere with fault tolerance, since the rules could forbid alternative routes that might be desirable. This is actually a variation of the first point; if the filtering rules must allow multiple entry paths, then a given message may have taken any of those entry paths, making the message harder to attribute.

3. Network ingress filtering is primarily useful for internal network attribution and to determine if an attack came from the "outside." Since generally only "internal" networks can be required to implement network ingress filtering rules, the approach is only useful for those networks. Thus, in the shorter term, this approach is probably more useful for organizations that are concerned about threats from within networks they control.

   In the longer term, this approach could be applied to countries or even internationally to give more attribution information. For example, U.S. law could be modified to require network ingress filtering on U.S. Internet Service Providers (ISPs). In that case, network ingress filtering could aid attribution of a (probably intermediate) system inside the U.S. and identify messages that originated from outside the U.S. as well. Note, however, that it would not reliably identify the network for messages originating from outside the U.S., because attackers could send messages through interconnected non-U.S. networks. Thus, attackers would quickly move to use at least some intermediaries outside U.S. jurisdiction.

Worldwide (international) implementation is technically possible, but this is probably impractical – the amount of worldwide cooperation required is simply too great. Also, worldwide implementation would aid against independent attackers, but not against nation-states. For both the U.S. and worldwide scenarios, the number of systems that would need to be configured would be extremely large, and if any were compromised, bogus information could be sent. Indeed, if the approach were introduced countrywide or worldwide, there would be many places where messages could be surreptitiously inserted. This would not eliminate the value of the filtering, but it would reduce its value.

4. Network ingress filtering only identifies the attacker network or range of networks; it does not (necessarily) identify an individual host. Thus, the approach does not eliminate all spoofing; it simply reduces the range of spoofed values. As a result, it must be combined with other techniques once the network or range of networks has been identified.

5. As with many other techniques listed here, network ingress filtering by itself it can only attribute a stepping stone. This approach cannot, by itself, identify the source behind the stepping stone. Tracing backwards through stepping stones requires other techniques.

6. Network ingress filtering is problematic to naively implement in some non-IP protocols. Network ingress filtering is a general approach, but most of the literature examines it only from the viewpoint of filtering IP packets. It can be more problematic to re-implement the same ideas at the level of store-and-forward network protocols such as email, since email's whole purpose is to pass on messages originally sent by others. Thus, if the same technique were directly applied without modification to email protocols instead of the IP layer, it could interfere with mail forwarding and other useful capabilities. This does not mean the approach could not be used in such protocols; various wrapping techniques or usage limitations could still enable use of the approach in other protocols. Note that this has nothing to do with network ingress filtering when applied to IP; email passes through IP-level network ingress filtering without problems on a correctly configured IP network.

7. Network ingress filtering imposes some administrative overhead, especially to initially deploy as well as to maintain.

8. Network ingress filtering imposes a performance cost, because every router implementing the filters must check new rules for every message. The performance cost depends on the complexity of the rules. The rules must be sufficient to uniquely identify a router. Since this is also a highly desirable property in TCP/IP network design (to simplify routing tables), the performance impact will be small in many TCP/IP networks. In many cases, network ingress filtering can be implemented using one or two rules that can be checked without noticeable degradation of router or network performance.

However, this will not be true for all networks. The approach may have a particularly large performance and administrative overhead on routers that support a very large number of different noncontiguous address ranges.

In particular, routers that are very close to their maximum load may need to be upgraded due to the additional overhead.

2.12.6 Filtering Advantages and Disadvantages

Many of advantages and disadvantages of the network ingress filtering approach also apply to any other approach implementing the filtering technique.

Advantages of the "perform filtering" technique include: it is often easily implemented with existing infrastructure, it does not require maintaining logs, it is usually transparent to users, and there are rarely legal impediments. Disadvantages include the fact that it must be implemented at nearly every relevant entry point, it can often only identify that a message came from "outside" the suite of filters (instead of its exact source), it often only identifies a range of sources (not the specific source), it can be difficult to employ on some protocols, it imposes administrative overhead to install and maintain the filters, and it imposes performance costs to execute the filters.

2.13 Implement Spoof Prevention

Protocols and/or their implementations can be modified, configured, or replaced to limit spoofing, simplifying attribution. Note that this technique is different from filtering techniques. Filtering techniques, such as the network ingress filtering approach, limit the source address value used in the data sent through the network, and are imposed near the sender's location or in the intermediate network. In contrast, protocol spoof prevention verifies that there is a valid connection back to the sender, and is imposed near or by the receiver's location.

In some cases, systems can be reconfigured to make spoofing more difficult:

1. Insecure protocols that are easily spoofed can be modified, reconfigured, or replaced to use different, more secure protocols that perform the same function. For example:

    a. The old rcp protocol can be replaced with the ssh protocol extensions to support file copying.

    b. UDP packets are easy to spoof on an internet, but TCP packets are more difficult to spoof. This is because TCP requires an initial two-way exchange of sequence numbers. Thus, protocols which can use either UDP or TCP can be configured to use only TCP, making spoofing more difficult.

2. Easily-spoofed protocols can be tunneled inside other protocols that resist spoofing. For example, an organization could implement a Virtual Private Network (VPN) using IPSEC, and then require all communication to go through the VPN.

3. Implementations of protocols with anti-spoof capabilities can be hardened so they are difficult to exploit. For example, on many systems, TCP sequence numbers are easily guessed, making spoofing of them much simpler. In contrast, some TCP implementations are designed to make TCP sequence numbers much harder to guess. Thus, replacing or upgrading systems to eliminate easily-guessed TCP sequence numbers can aid attribution by making certain kinds of spoofing difficult. The problem of easily-guessed TCP sequence numbers has been known for years, and recommendations to improve this situation are publicly documented [Bellovin 1996]. Unfortunately, many widely-deployed systems still have these problems. [Zalewski 2001] found that, in 2001, only Linux and a not-yet-deployed version of OpenBSD had difficult-to-guess TCP sequence numbers of the many systems tested. In contrast, other common systems had "more or less serious flaws that make short-time TCP sequence number prediction attacks possible." Windows 2000 and Windows NT4 SP6a were considered mildly vulnerable to attacks; older versions of Windows were extremely vulnerable, as were several widely-used Unix implementations.

4. Stronger authentication approaches and practices could prevent attackers from spoofing that they are (other) legitimate users. Some protocols have optional authentication approaches that, if enabled, can make spoofing far more difficult. Eliminating cleartext, default, or easily guessed authentication passwords would make it more difficult for an attacker to forge or hide an identity.

Carefully designed protocols can simultaneously make spoofing and successful DDoS attacks more difficult. If a protocol requires a time-consuming authentication operation when the client makes an initial request, the system is vulnerable to DDoS attacks. This is because an attacker can simply send large numbers of invalid requests, overwhelming the defender's resources. This problem occurs whenever the attacker can create invalid requests significantly faster than the defender can validate them. There are well-known techniques for dealing with these issues, such as:

1. Protocols can at least determine that there is a bi-directional path by first requiring that a nonce be exchanged (the TCP protocol does this).

2. If the defender must track all partially-opened connections, attackers may still be able to quickly overwhelm storage resources; this is the basis of the SYN attack against TCP implementations [CERT 1996]. Techniques such as "SYN cookies" can prevent this by requiring that the defender respond with a value that requires little overhead to validate [Bernstein].

3. The protocol can require that a client first solve a “puzzle.” A “puzzle” is any value which costs servers little time to verify but costs the client a significant amount of time to solve. Thus, attackers acting as clients can send invalid puzzle solutions – but the server can quickly reject them – or they can solve the puzzles – slowing the DDoS attack. No protocol design can truly prevent DDoS attacks, but puzzles can make such attacks more difficult. Puzzles also make spoofing somewhat more difficult, as they force the attacker to expose a channel that can (at least temporarily) reach them.

Other approaches can also make spoofing (at various protocol levels) more difficult. [Jung 1993] presents an approach, named “Caller Identification System in the Internet Environment” (CISIE), where during the process of logging into a remote host, the originating host must present a trace for the user (which the destination then verifies and logs). This approach requires that every host support such queries and that the approach be implemented for each protocol. [Buchholz] tried to re-implement CISIE and found significant difficulties in doing so. In particular, a way to match outgoing connections to incoming connections is needed; this is possible, but the lack of detail on this problem suggests that CISIE has not yet been fully implemented.

Templeton [2003] describes various techniques for detecting spoofed packets. For example, a time-to-live (TTL) value different from past values for a source to a destination may suggest a spoofed packet. This could be combined with protocols that attempt to determine if the other participant is truly the intended participant, or an imposter.

The Deciduous approach [Deciduous, Chang 1999, 2000, undated] requires that IPSEC’s Authentication Headers (AH) be used by at least some of the routers in the communication path, and uses these headers to help identify the source. Their implementation requires significant modification for use: application programs must be modified and a new operating system kernel call must be added to permit applications to identify the security associations attached to the received data.

An example of this technique at a higher protocol level than IP would be a policy that rejects unsigned and unvalidated email at the recipient’s final mail transfer agent (MTA) or email reader (this could also be considered an extreme form of a filter). At this time, such a policy is probably impractical in many situations, but such a policy could be required in some situations and might become practical for more users in the future.

A far more pervasive and controversial approach is “eDNA,” an approach briefly examined by DARPA in 2002. In this approach, portions of the Internet would be designated as “public network highways” which would be designed to forbid anonymity. To access these portions, all network and client resources would be required to maintain traces of user information (called eDNA) so the user could be uniquely identified as having visited a web site, having started a process, or having sent a packet. A user would need to enter a digital version of unique personal identifiers (like a fingerprint or voice), which would then be turned into an electronic signature appended to any message. SRI was asked to briefly investigate the concept, and in August 2002, SRI brought together respected computer security researchers as part of the investigation. Almost all participants strongly criticized the concept, on both technical and privacy grounds, and several believed the approach would not solve the problems it was trying to address. In the end, DARPA decided to not pursue eDNA further [Markoff 2002] [McCullagh 2002] [DARPA 2002].

An advantage of the spoof reduction technique is that it greatly reduces the number of intermediate systems that must be examined by other attribution techniques. Where protocols and/or implementations can be easily modified, configured, or replaced to limit spoofing, this approach can be very inexpensive as well.

There are disadvantages as well. In cases where protocols and/or their implementations cannot be easily modified, configured, or replaced, this technique can be very expensive. In some cases, it may be possible to “wrap” the protocol inside some other more secure protocol (such as IPSEC), but this is not always true. The technique is generally not useful against stepping stones, since stepping stones can correctly implement a protocol while hiding the attacker’s location and identity. While this technique can simplify attribution, it will generally need the aid of other attribution techniques.

2.14 Secure Hosts/Routers

Attackers often use multiple intermediate systems to foil attribution. Therefore, attribution can be aided by reducing the number of intermediate systems an attacker can employ.

This can be accomplished through increased security of hosts and routers, including removing unnecessary services from each. A robust security patch process should be employed to ensure that all vendor security alerts and patch releases are rapidly prioritized, tested, and deployed on all relevant systems by system administrators. Vulnerability scanning (both host-based and network-based) should be used to help identify any unpatched vulnerabilities. Vulnerabilities found should be rapidly fixed. General techniques for hardening systems are widely discussed elsewhere and are not further discussed here.

The approach of securing hosts and routers is particularly helpful in reducing broadcast amplification. [SANS 2000a] recommends the following (among other steps):

- Network hardware vendors should ensure that routers can turn off the forwarding of IP directed broadcast packets as described in RFC 2644 and that this is the default configuration of every router (network system administrators need to ensure this is true when the routers are installed)

- Unless an organization is aware of a legitimate need to support broadcast or multicast traffic within its environment, the forwarding of directed broadcasts should be turned off. Even when broadcast applications are legitimate, an organization should block certain types of traffic sent to "broadcast" addresses (e.g., ICMP Echo Reply) messages so that its systems cannot be used to implement Smurf attacks.

In particular, system and network administrators should turn off the “echo” and “chargen” services unless they have a specific need for those services. This is in general good advice for all network services – network services should be disabled unless there is a specific need for them.

An advantage of this approach is that it is needed for securing systems in any case. A disadvantage is that, by themselves, these approaches are not enough to support attribution; they simply make other attribution processes easier to perform.

2.15 Surveil Attacker

If there is sufficient evidence to suggest that a particular person or set of persons might be an attacker, various surveillance approaches can be used that specifically target those suspects. These include examining email messages, keyboard sniffers, electromagnetic radiation surveillance, and other such techniques. Even logs of phone numbers or email addresses contacted can be valuable. Computer forensics approaches can be used to examine the storage devices of suspects’ computers.

Naturally, there are a number of strict laws controlling the application of these privacy-invading techniques, so these are not techniques that can be requested or applied lightly. In a few cases, these techniques can be used immediately once a particular attacker or set of attackers is suspected. For example, employers are permitted to perform certain kinds of monitoring on employees. Service providers and equipment owners are also allowed to perform certain kinds of monitoring of their own equipment. Employees and customers could be required to sign documents specifically permitting monitoring in certain cases. In many other cases, these techniques can only be applied after legal actions (such as the granting of a warrant).

An advantage of attacker surveillance is that it can often confirm if a given attacker truly did or did not perform an attack, even if the attacker uses sophisticated techniques to avoid attribution. A serious disadvantage of the technique is that there needs to be some reason to suspect the attacker in the first place, as well as an opportunity to perform the surveillance. Also, even under surveillance an attacker may manage to perform an undetected attack, since surveillance is never perfect.

2.16 Exploit Reverse Flow

Many protocols are bi-directional, including those used by attackers. If data flows back to the attacker or an attacker’s intermediary, this flow may be modified or followed to support attribution.

An example of this technique is the approach called “sleepy watermark tracing” (SWT). In this approach, when the defender wishes to attribute an attack the defender injects a watermark into the reverse (return) data flow. A watermark is simply data that would not normally be detected by an attacker. For example, in the telnet and rlogin protocols, a defender returning the string:

> "**See me**abc\b\b\b \b"

would look the same to an attacker as a defender who returned the string:

> "**See me**"

In SWT, network server applications (such as telnetd and rlogind) on the defender's host system are modified to be "watermark-enabled," so that on command they can insert these watermarks. SWT guardian gateways are then used to detect and report the presence of these watermarks. SWT is described in [Wang 2001a]. SWT response options are further described in [Wang 2001b].

One advantage of the technique is that it can attribute immediately through a large number of stepping stones, if the data is not transformed through processes such as encryption. However, there are many disadvantages. Most such implementations (such as SWT) require significant changes to pre-existing implementations. Detectors of the data in the reverse flow must be placed in locations that can actually observe the data, and there is always the danger of false positives from the detectors. Also, hosts that transform the data (such as encrypting the data) may foil the technique.

2.17 Combine Techniques

Since every technique has its strengths and weaknesses, it is probably more effective to combine the various techniques to perform attribution. For example, network sensors could be forward deployed closer to where an attacker might attack from. If the network sensors include initial rules for known attacks, they would be considered forward-deployed IDSs in this grouping. If they also supported rapid run-time requests for new patterns, they could also support input debugging. Network ingress filtering might be useful to reduce the number of possible networks an attacker came from, and then other attribution techniques could be used to identify the attacker's location more precisely.

Also note that different protocol layers can provide different information that together provide better attribution information. Many implementers concentrate on the IP layer; since IP is a common layer, attribution approaches based on the IP layer are more general. However, combining information from various protocol layers (such as IP, authentication logs, MTA logs, higher-level protocols, and lower level protocols) can add information that examining only one layer will miss.

In theory, an advantage of combining techniques is that it can overcome many of the disadvantages of individual techniques. However, currently there is relatively little experience (or available automation) for combining techniques. Also, combining techniques cannot overcome the old phrase "garbage in, garbage out" – if the results of the individual techniques are worthless, combining them will not help.

3. Issues in Attribution Techniques

This section discusses some of the issues common to many attribution techniques.

3.1 Prepositioning of Tools and Trust is Critical

Many attribution techniques cannot be applied to an attack unless the attribution implementations and trust relationships have been prepositioned. This is particularly obvious with logging systems; it is impossible to query a log unless the logging system has already been deployed. However, this is true for many other techniques, such as network ingress filtering.

Even if the technology does not need to be prepositioned, trust relationships need to be prepositioned for attacks to be attributed in a timely manner. For example, input debugging for a single attack using a simple pattern does not take much time to implement technically as long as the routers are inside a single administrative domain. However, rapidly attributing attackers through paths going through external administrative domains often requires some sort of pre-existing trust relationship between the person performing the attribution and the administrators of those external domains. The external domain administrators need to know if the request is coming from a legitimate source (with a legitimate reason to know the answer), and the requestor needs to know if they are truly communicating with the correct external domain administrators. Thus, trust relationships (manual or automated) must be developed so that when requests are made they will be honored in a timely way.

3.2 Prepositioning Tools and Trust in External Networks is Difficult

Attacks often originate “outside” of the network being attacked. However, as noted above, to be effective many attribution techniques require some sort of cooperation by networks along the path from the attacker to the victim. Gaining such trust, unfortunately, can be very difficult. Even when trust is gained, convincing others to implement attribution tools can be a significant challenge.

3.3 Networks and Systems Can be Configured to Ease Attribution: Changing the Terrain

Networks and systems can be configured to simplify attribution in a variety of ways. Network routers and systems can be hardened against attack, spoofable protocols can be eliminated or limited, cleartext passwords can be eliminated, and broadcast amplification/reflection can be disabled. Attribution can be aided even more directly through techniques such as network ingress filtering, honeypots, and forward-deployed IDS systems.

We refer to intentionally reconfiguring a network to ease attribution as ***changing the terrain***. In physical warfare, defending militaries spend a great deal of money to modify the terrain to impede their enemy and aid themselves, and have done so throughout history (e.g., castles, roads, mines, trenches). In the same way, a defender can modify their computer network and related networks to impede their attacker and aid themselves.

3.4 Attribution is Often Easier Against Insiders

It somewhat easier to perform attribution of inside attackers or inside intermediaries compared to systems outside a defender's administrative control. This is because of the factors noted in sections 3.1 through 3.3. Since many attribution techniques require prepositioning, but prepositioning is difficult to perform on outside networks, many techniques can only be fully employed against inside people or systems. Also, since an organization can generally only control the network configuration and architecture, a defender can generally only change their own network to support attribution. In addition, organizations can generally monitor their own networks more effectively and have more legal options for performing this monitoring.

This is not universally true, and there are some countervailing forces. Insider personnel would tend to know more about an organization's defenses, and thus might be able to circumvent them. For example, inside personnel are more likely to know what systems are actually honeypots, and avoid them. Inside systems are more trusted than outside systems, and exploitation of those trust relationships is less likely to be detected. Some insiders (such as some of the network administrators) may be specially trusted with control over the systems used for attribution, or with secret information vital for its effective use, and be able to thwart attribution. Nevertheless, many attribution techniques do not fundamentally depend on secrecy of the technique, a single inside attacker might not know some key pieces of information, and inter-system trust can be limited. For example, if multiple attribution techniques are used, most inside personnel are less likely to know of all of them.

Thus, attribution is in some ways easier to accomplish against inside personnel or systems than against outsiders. This does not invalidate attribution techniques, because insiders perpetrate a significant proportion of all attacks.

3.5 Build Attribution Techniques into Common Components

Deploying separate components for attribution, and pre-positioning them where prepositioning is necessary, is expensive in both time and money. Thus, it will be strongly resisted by many. In many cases, it would be better to ensure that many of these techniques are built into common commercially-available components such as routers, firewalls, operating systems, and common network services (including authentication services, email, and so on). This would ease the burden of widespread deployment, both inside and outside a network. In particular, support for attribution would be added without effort during routine upgrade or replacement. However, convincing developers to include these capabilities into their components is not necessarily easy.

Developers may not see sufficient value in incorporating attribution techniques into their products, so it may often require up-front negotiation and payment to have some attribution capabilities added to existing commercially-available products:

1. For proprietary components, adding such capabilities will often require negotiation and payment of the developer of the component. In some cases another option is available: developing a separate "plug-in." An advantage of "plug-ins" is that developing the plug-in can be competed, and a plug-in can often be implemented more rapidly (because negotiation with the product vendor is reduced or unnecessary). However, developing plug-ins can be difficult (depending on the flexibility of the plug-in architecture), the likelihood of deployment is reduced (because adding plug-ins takes additional administrative time), and plug-ins are likely to be more difficult to maintain over time as the product evolves.

2. For open source software components, these options (paying the developer and/or developing a plug-in) are available, plus one more: the DoD could perform the modifications directly to the software. An advantage of directly modifying the software is that the change can be competed and implemented immediately, with far less implementation difficulty (e.g., because there is no need to only stay inside a "plug-in" architecture). However, changing the software directly has maintenance impacts. If these changes are not merged back into the trusted repository of the open source software, long-term maintenance costs can become large. This is because the open source software would change over time, diverging from the modified software. Thus, in many cases this suggests a better approach would be to try to convince the trusted developers of that open source software project to accept the changes. Early negotiation with those who maintain the open source software could be essential to increase the likelihood of the work being incorporated into the "main branch" of the software.

Another way to encourage including attribution capabilities in common components would be to ensure that these capabilities are added to relevant DoD Protection Profiles (PPs). In some cases acquiring a product is contingent on a Common Criteria (CC) evaluation of the product against a PP. Adding the requirement to the PP might encourage vendors to meet the requirement.

A related issue would be how the attribution capability is operationally enabled: is it always enabled, enabled by default (but it can be configured to disable it), or disabled by default (but it can be configured to enable it)? Clearly, from the point of view of attribution, being always enabled is the best alternative. However, since some techniques have the potential to invade privacy or lower performance, that may not be acceptable to other customers. For many techniques, trying to make it impossible to disable the capability is a waste of time. Administrators who truly want to disable the capability can often combine the component with other components to thwart attribution, or use a different component. However, it may be fruitful to try to have some capabilities enabled by default. Any discussion with a developer of a commercially-available product (proprietary or open source) should include discussions on how the capability will be enabled.

3.6 Attribution Requires Funding

Clearly it will cost money to build or buy attribution capabilities (either as separate components or as additions to other components), and there are administrative costs for installing, maintaining, and using components supporting attribution. How will these capabilities be paid for?

There is little evidence that the commercial sector is willing to primarily shoulder the costs of these capabilities. Commercial companies *are* concerned about DDoS attacks, for example, but they are often only interested in reducing their effect, not in actually identifying the attacker. Even if the cost of attribution were reduced to zero (an unlikely scenario), there seems little benefit to identifying attackers in many cases. Bringing a lawsuit against an attacker is quite likely to be very expensive, and it is unlikely that these costs would be recovered. A company bringing a lawsuit risks failing to convict, and it is unclear if a conviction would actually reduce attacks. Some companies are very concerned about unwanted publicity any such lawsuit would entail. Indeed, many companies are unlikely to see attributing attackers as their job. Most commercial companies appear to view identifying attackers as a law enforcement or military task, not a commercial one.

Laws could be enacted requiring certain attribution capabilities be embedded in products for sale, or requiring providers of services to implement certain capabilities. This is unlikely to be effective for most techniques. Most techniques' costs are sufficiently large (especially if research is also required) that any such effort would be strongly resisted. One exception to this may be network ingress filtering; it might be possible to impose a requirement on Internet Service Providers to require that any data entering their backbones go through such a filter, at least for non-peers.

Another approach would be for the U.S. government or DoD to require that certain products must have certain attribution capabilities before they will be acquired. In theory, vendors would add those capabilities and then pass those costs back to users through higher prices. This alternative approach only works with proprietary vendors, who receive funding through usage licenses. This would require convincing the vendor that the cost can be recovered with a profit - an argument that they may not accept. It should be noted that in many markets the U.S. government and/or DoD has a very small portion of the market. A vendor can be disadvantaged by spending money to create a specialized feature only wanted by a small portion of the market, because competing vendors will spend money on capabilities desired by more customers instead. As a result, the DoD may not have any viable application with attribution capabilities, or may only have inferior applications from vendors who decide to add the attribution capability in the hope that the U.S. government or DoD will have to buy it instead of a competing superior product.

If the government (including the DoD) wants the ability to attribute attacks, in many cases the government may need to pay for it directly. One approach is to fund development and deployment of these abilities for widely-used applications. More than one product of each category should be funded, so that the government is not locked into a single product. If the government cannot switch to another product, the vendor will probably raise prices substantially and is less likely to provide good service.

3.7 Standards are Needed

Standards are critically necessary for attribution for the following reasons:

- Interoperability. Many techniques require automated interaction (for speed) between many different organizations and echelons, and it is improbable that exactly the same vendor would be used for all of them. Different organizations will have preferences for different vendors for a variety of reasons (pre-existing relationships, low cost, enhanced functionality, working well with existing infrastructure, and so on). Thus, for attribution to be effective on a wide scale, attribution standards to support inter-vendor interoperability will be necessary.

- Lower cost. By avoiding a proprietary solution from a single vendor, users may be able to select between a variety of offerers. Competition between vendors usually results in a lower price for consumers.

- Lower risk. If a vendor goes out of business or stops supporting a product, another can be used.

- Increased flexibility. If a product doesn't provide what is desired, another product can be used or the extensions can be developed (the latter is particularly easy if it's an open source product).

In theory, a standard could be held secret inside the DoD, but a component that is so widely deployed would be difficult to keep secret. In addition, to attribute external attackers the information would have to be released anyway, so it is almost certainly better to start by developing publicly available standards from the beginning. Note that some will have significant privacy concerns, so standards development should include efforts to address those concerns.

Standards should be open, in particular:

- Standards should be publicly defined and held. This way, no single vendor controls others, permitting competition. Organizations that support development of publicly defined and held standards include the Internet Engineering Task Force (IETF), the World Wide Web Consortium (W3C), the

Institute of Electrical and Electronics Engineers, Inc. (IEEE), the American National Standards Institute (ANSI), and the International Organization for Standardization (ISO). The IETF and W3C are more commonly used for internet-related standards; they are also faster to respond and redistribute standards freely (increasing the number of potential competitors). Thus, for many attribution standards, these organizations might be preferred.

- Standards should not be patent-encumbered. A standard that cannot be implemented without a patent license gives a special advantage to the patent holder(s). Such patents constrain or prevent competition, and thus undermine the advantages of standards listed above. Both the W3C and IETF strongly discourage patent-encumbered standards for these reasons.

Some specifications that could form the basis of standards for attribution include the following:

- IDIP (of CITRA). This has two layers, the application layer (that uses CISL) and the message layer.
- Common Intrusion Specification Language (CISL)
- Ident extensions (for SPIE)
- ICMP Traceback Messages (iTrace) – this is an IETF draft
- Results of the Intrusion Detection Exchange Format Working Group (IETF idwg), including the Intrusion Detection Message Exchange Format (IDMEF) and intrusion detection exchange protocol (IDXP). Information on this working group is available at http://www.ietf.org/html.charters/idwg-charter.html
- RID-DoS, a simple draft protocol for inter-network provider communication. This protocol defines messages for trace request, trace authorization, and source found. More information is in [Moriarty 2003].
- Network Ingress Filtering, IETF RFC 2827 (note that this is already a standard)

3.8 Attribution Techniques Must Be Secured

Clearly, attribution techniques themselves need to be secured. They should be resistant to subversion by an attacker, in particular, attackers should not be able to corrupt the data used for attribution or prevent attribution by directly attacking the attribution components. An attribution technique should not create a new avenue of exploitation (e.g., by creating a new technique for performing a denial of service attack against the system). In many cases, this will require authentication and checking for authorization, and intrusion detection systems should note unauthorized requests.

Many of these techniques require trust between multiple different organizations, making securing these components more difficult.

3.9 Attribution Should Usually Be Hidden from the Attacker

In many cases, an attribution technique should not reveal to an attacker that an attribution process exists or that one is being executed. This is especially difficult if the administrators of domains along the path are colluding with the attacker. “Random” queries can help (where occasionally messages are randomly selected for attribution). However, employing a large number of random queries is only practical if the technique is highly automated and does not interfere with normal operation.

However, in some cases hiding attribution capabilities may not be desirable. Some attackers may decide to not attack at all if they knew that they risked attribution, or may break off an attack if they believe an attribution attack is ongoing. Thus, a known attribution capability may sometimes serve as a useful deterrence. The attribution capability or technique may not even be real, or it may appear to use one technique when in fact another is being used.

Organizations will need to decide if they wish to hide or reveal the existence of an attribution process, as well as the details of that process. Organizations will also need to determine how they intend to implement the hiding or revealing.

3.10 Sensor Placement Is Important

Many attribution techniques are based on the principle of establishing sensors of some kind, then analyzing and using the information from that sensor. Clearly, the information gained depends on the placement of those sensors. Two sensor placement issues are particularly relevant: sensor location, and whether or not the sensor is “in-line”:

1. *Location.* Clearly sensors can only be useful if they are placed where they can acquire useful data. This suggests that for attribution purposes, sensors should be placed not only near a defender, but also as near to the attacker as possible so the attacker can be more accurately attributed. To support traceback, sensors must be located at relevant intermediate points as well, to enable a defender to quickly locate the attack path.

2. *In-line or Non-in-line.* Sensors can be placed as in-line sensors or as non-in-line (monitoring) sensors. In-line sensors require that all sensing operations be complete (e.g., initial logging) before additional normal processing occurs. Non-in-line (monitoring) sensors passively observe operations instead, but if they cannot keep up they lose data. The disadvantage of in-line sensors is that they may slow down overall processing. The disadvantage of monitoring sensors is that, if a network or system becomes overwhelmed, such sensors may lose critical data, and this is exactly the time where such data may be needed. Fundamentally, this is a trade-off between the quality of attribution information (in-line) and performance (non-in-line).

3.11 Many Attribution Techniques Require Funding for Technology Transition

Clearly, there are a number of attribution techniques. However, many of these techniques have only been implemented as non-robust prototypes, if they have been implemented at all. Some of these techniques have been developed with DARPA funding (such as [Snoeren], [Sanchez], [Schnackenberg], and [Sterne]), but DARPA does not have an obligation to ensure that its research work is eventually turned into working, useful products, even when that work is extremely promising. Some work (such as [Burch 2000]) can only be used under special legal circumstances and is unlikely to be a commercially viable product. For some techniques, government development is the only alternative if it is to be developed at all.

Thus, there is a significant need for a technology transition plan with significant funding if some of the research concepts are to be turned into working products.

3.12 Legal/Policy Issues Intertwine

Although this paper concentrates on the technical issues, any deployment must carefully consider the legal and policy issues with attribution. Many attribution techniques can only be employed by people in certain roles. Laws and policies are often unclear, and may need to be clarified (or possibly revised) to employ some attribution techniques. See [Aldrich 2002] for more on legal issues in attribution.

3.13 Need to Protect Privacy and Freedom of Speech

Some attribution technologies can be misused to subvert privacy, eliminate anonymity, and eliminate pseudonymity. This is especially a concern if attribution technologies developed in democracies are acquired and redeployed by governments with abusive human rights records to suppress freedom of speech and democracy movements.

Members of Congress have already expressed similar concerns. For example, the "Global Internet Freedom Act" (S 3093 IS)[8] was proposed in the U.S. Senate on October 10, 2002, to "develop and deploy technologies to defeat Internet jamming and censorship." Their concern is that various countries keep their citizens from freely accessing the Internet and obtaining international political, religious, and economic news and information; the proposed bill lists examples of such countries as Burma, Cuba, Laos, North Korea, the People's Republic of China, Saudi Arabia, Syria, and Vietnam. This is similar to anti-jamming techniques already used by the Voice of America. If these countries could easily use attribution techniques against their own citizenry when those citizens accessed or shared some kinds of information (e.g., on democracy or religion), and jail or kill its citizenry for doing so, then attribution techniques could be used to

[8] The text of the Global Internet Freedom Act is available at http://thomas.loc.gov/cgi-bin/query/z?c107:S:3093:

suppress independent thought in other countries. This result is not in the best interests of the U.S.; indeed, it's not in the best interest of humanity.

Even without the concern of abuse by foreign governments, U.S. citizens will certainly want their privacy against what they may perceive as unwarranted government intrusion. Indeed, the fourth amendment to the U.S. Constitution guarantees that people must be secure "against unreasonable searches and seizures."

Clearly, attribution techniques that pose less danger to privacy should be the ones most encouraged.

3.14 Required Attribution Times Will Continue to Shrink

Some attacks will be slow, over a period of possibly months. But other attacks will be rapid, on the order of milliseconds. [Staniford 2002] discusses techniques for attacking large numbers of systems in very short times. For rapid attacks, attribution techniques will need to rapidly attribute the attacker before the attacker can "get away" or any useful data is hidden by a mass of distracting data. This suggests that automated attribution will be increasingly necessary, and that manual techniques will become increasingly worthless against certain kinds of attacks.

3.15 Attribution is Inherently Limited

All technical means for attribution are inherently limited. These limitations include attribution delay, failed attribution, and misattribution.

3.15.1 Attribution Delays

If an attacker uses a zombie to perform a significantly delayed automated attack, it becomes extremely difficult to attribute the attack path preceding that zombie. Even when the attacker does not intentionally include a delay, there is usually a delay in the defender's response that an attacker can exploit. This delay in the defender's response has many sources: the defender must determine that the message is an attack (or at least that it is worth attributing), perform the attribution, decide on a response, and implement the response. An attacker may be gone before attribution has identified or located the attacker, and/or the attribution may have been made but too late to perform an effective response.

These weaknesses can be partially countered by considering certain kinds of pre-attack activity to be a form of attack and performing attribution. Examples of such pre-attack activity include footprinting, scanning, and enumeration of systems on a network. However, some of these activities are legitimate and/or not really attacks, and they occur constantly on the open Internet. Thus, attribution activities that have a high cost should not normally be used simply to attribute actions that may not be precursors to an attack or are reoccurring.

3.15.2 Failed Attribution

An attribute technique may fail to attribute an attacker. Widespread prepositioning and the use of multiple techniques can help, but are not guaranteed.

3.15.3 Misattribution

An attribution process may identify the wrong location or identity of an attacker, a problem that this paper will refer to as *misattribution.* There are many possible causes for misattribution, including defective software, incorrect data, incorrectly interpreted data, and ambiguous data. Since attackers can perform various counter-measures, attackers may intentionally send (or try to send) incorrect data to an attribution process.

Attackers may even wish to cause misattribution as their primary purpose, rather than actually be successful at the attack. For example, if there is already tension and conflict between two adversaries (e.g., two countries A and B), a third party (C) could try to attack one (A) and cause the attack to be misattributed to the other party (B). Thus, the third party could escalate a conflict between others simply by forging attacks.

Ideally an attribution process would also report the confidence level in the attribution, but this information is often not available. Thus, any use of attribution information must account for the fact that attribution always carries with it some uncertainty.

4. Conclusions

We conclude the following:

1. There are a large number of different attribution techniques. Each technique has its strengths and weaknesses; no single technique replaces all others.

2. Attribution is difficult and inherently limited. In particular, attackers can cause attacks to be delayed and perform their attacks through many intermediaries in many jurisdictions, making attribution difficult. In some cases this can be partly countered, for example, by treating some information-gathering techniques as attacks (and attributing them), using multiple techniques, and using techniques that resist this problem (such as exploiting/forcing attacker self-identification and attacker surveillance). Nevertheless, because of the difficulty and uncertainty in performing attribution, computer network defense should not *depend* on attribution. Instead, attribution should be part of a larger defense-in-depth strategy.

3. Attribution tends to be easier against insiders or insider intermediaries.

4. Prepositioning is necessary for many attribution techniques.

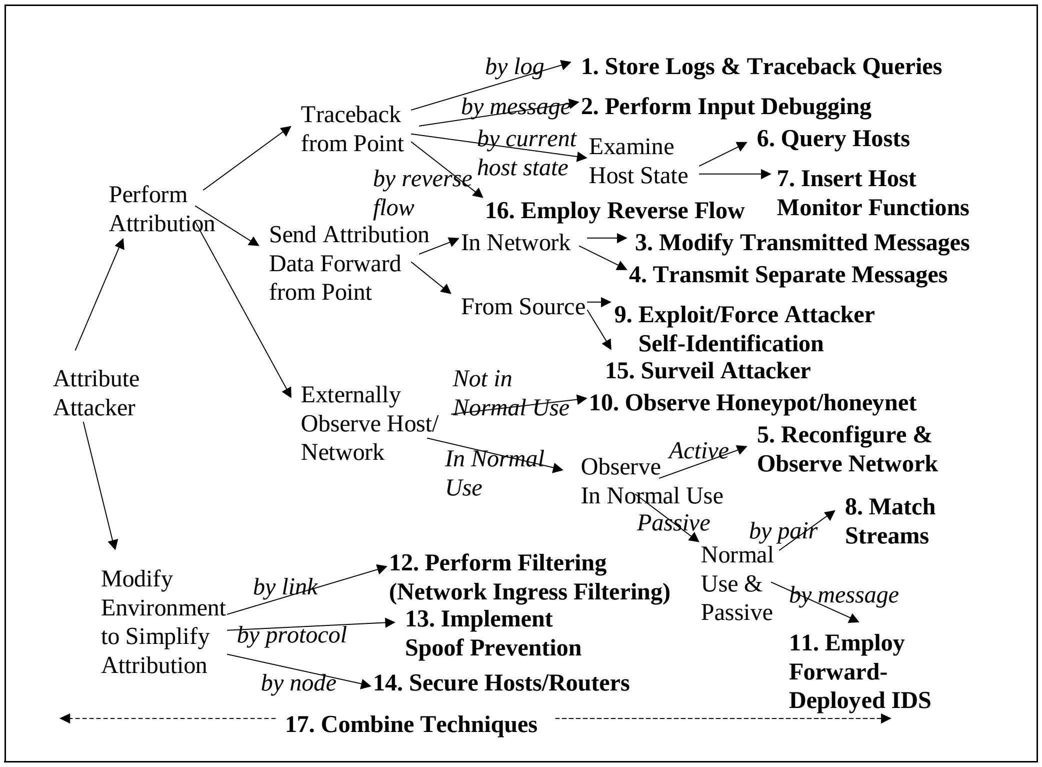
5. Many techniques are immature and will require funding before they are ready for deployment. If the DoD wishes to have a robust attribution capability, it must be willing to fund its development and deployment.

6. A useful first step for the DoD would be to ***change the terrain*** of its own network. By this, we mean modify DoD computers and networks to aid attribution techniques. This includes hardening routers and hosts so exploiting them as intermediaries is more difficult, limiting spoofable protocols, disabling broadcast amplification/reflection, and implementing network ingress filtering. Changing the terrain should also be applied to key networks the DoD relies on, to the extent the DoD can convince those network owners to do so.

Appendix. Attribution Technique Taxonomy

There are existing taxonomies of attribution techniques. For example, [Wang 2001a] divides "tracing approaches" into two categories: host-based and network-based, each of which can be classified as being active or passive. However, these taxonomies do not appear to suggest the many possible techniques described in this paper.

Figure A.1 presents a possible taxonomy of the attribution techniques as defined in this paper. The figure shows that the task of attributing attackers can be divided into techniques that actually perform attribution, as well as techniques that modify the environment to simplify attribution. Performing attribution can be further subdivided into techniques that trace backwards from some given point, techniques that send data forward from a given point, and techniques that view network/host from an external view and extract attribution information using that viewpoint.



Attribute Attacker
Perform Attribution
Traceback from Point
by log
1. Store Logs & Traceback Queries
by message
2. Perform Input Debugging
by current host state
Examine Host State
6. Query Hosts
7. Insert Host Monitor Functions
by reverse flow
16. Employ Reverse Flow
Send Attribution Data Forward from Point
In Network
3. Modify Transmitted Messages
4. Transmit Separate Messages
From Source
9. Exploit/Force Attacker Self-Identification
15. Surveil Attacker
Externally Observe Host/ Network
Not in Normal Use
10. Observe Honeypot/honeynet
In Normal Use
Observe In Normal Use
Active
5. Reconfigure & Observe Network
Passive
Normal Use & Passive
by pair
8. Match Streams
by message
11. Employ Forward-Deployed IDS
Modify Environment to Simplify Attribution
by link
12. Perform Filtering (Network Ingress Filtering)
by protocol
13. Implement Spoof Prevention
by node
14. Secure Hosts/Routers
17. Combine Techniques

Figure A.1. Attribution Technique Taxonomy

Tracing backwards from some point (typically the defender) back toward the attacker can use different kinds of information. One approach is to require intermediate systems to store logs (historical information of some kind) that can be later queried. Another approach is to request intermediate systems to report the "next time" a message of a certain pattern is detected. A third approach is to examine the current state of a host (or a router, but usually routers have no interesting "current state"). Hosts can be queried if they support such queries, and if not, querying capabilities can be inserted into them. Many protocols have bi-directional data flows; the reverse flow leads back to the attacker or attacker intermediary and may be exploitable as well.

Sending attribution data forward can occur from some point inside the intermediate network, or from the source (attacker). Inside a network, the data can be sent by modifying messages as they are sent, or via separately-transmitted messages. It may be possible to attribute the attacker directly at the source, by exploiting data the attacker sends and/or by surveillance of the attacker.

Externally observing the host/network may also provide attribution information. Systems (or virtual systems) that are not being used for normal work may be set up specifically to support detection and attribution, i.e., honeypots or honeynets. Systems that are being used for normal work can be actively modified to support attribution (i.e., reconfiguring the network and using those results to support attribution). Alternatively, the systems can be passively monitored for attribution information: messages can be used individually, or pairs of messages can be used to identify matching flows (also called streams).

The environment could be modified to support attribution. The environment's links, protocols, and/or nodes can all be modified to make it more difficult for an attacker to hide their location or identity. In some cases, the environmental modifications can reveal so much about an attacker that they have the effect of performing attribution by themselves. This is particularly true for network ingress filtering, a specific approach using the "perform filtering" technique.

Note that techniques can be combined. In some cases, one technique can compensate for the weaknesses of another.

This taxonomy is probably not complete. It is quite possible that there will be future attribution techniques that will require this taxonomy to be extended. An area particularly likely to be expanded is externally observing hosts/networks in normal use.

Better taxonomies will probably be developed in the future. However, the taxonomy of Figure A.1 should aid understanding of the many techniques already documented in the public literature.

References

Note: To aid rapid acquisition of particular references, URLs are provided for many references. Some of these URLs may no longer be valid.

[Aldrich 2002] Aldrich, Rick. July 9, 2002. *Computer Network Defense Attribution: A Legal Perspective.* Prepared for the Defense-wide Information Assurance Program (DIAP).

[Asaka 1999a] Asaka, Midori, Atsushi Taguchi, and Shigeki Goto, "The Implementation of IDA: An Intrusion Detection Agent System", in *Proceedings of the 11th FIRST Conference 1999*, Brisbane, Australia, June 1999. http://www.silicondefense.com/research/itrex/archive/tracing-papers/asaka99_tracing_using_mobile_agents.pdf

[Asaka 1999b] Asaka, Midori, Shunji Okazawa, Atsushi Taguchi, and Shigeki Goto. June 1999. "A Method of Tracing Intruders by Use of Mobile Agents", INET'99. http://www.silicondefense.com/research/itrex/archive/tracing-papers/asaka99_tracing_using_mobile_agents.pdf

[Asaka 1999c] Asaka, Midori, Masahiko Tsuchiya, Takefumi Onabuta, Shunji Okazawa, and Shigeki Goto. November 1999. "Local Attack Detection and Intrusion Route Tracing", IEICE Transaction on Communications, Vol.E82-B No.11, pp.1826-1833. http://www.silicondefense.com/research/itrex/archive/tracing-papers/asaka99local_attack_detection_and_tracing.pdf

[Arkin 2002] Arkin, Ofir. 2002. "Trace-Back: A Concept for Tracing and Profiling Malicious Computer Attackers." London, England: Atstake Limited.

[Axelsson 2000] Axelsson, Stefan. March 14, 2000. "Intrusion Detection Systems: A Survey and Taxonomy." Technical Report No 99-15, Dept. of Computer Engineering, Chalmers University of Technology, Sweden. http://citeseer.nj.nec.com/axelsson00intrusion.html

[Bellovin 1989] Bellovin, Steve. 1989. "Security Problems in the TCP/IP Protocol suite." *ACM Computer Communications Review* 19/2. pp. 32 - 48.

[Bellovin 1996] Bellovin, Steve. May 1996. "Defending Against Sequence Number Attacks." IETF RFC 1948. http://www.ietf.org/rfc/rfc1948.txt

[Bellovin 2000] Bellovin, Steve. "ICMP Traceback Messages" draft-bellovin-itrace-00.txt http://www.research.att.com/~smb/papers/draft-bellovin-itrace-00.txt

[Bernstein] Bernstein, D.J. "Syn Cookies." http://cr.yp.to/syncookies.html.

[Buchholz] Buchholz, Florian, Thomas E. Daniels, Benjamin Kuperman, Clay Shields. "Packet Tracker: Final Report." CERIAS. http://www.cerias.purdue.edu/events/traceback/packettrackerreport.pdf

[Burch 2000] Burch, H., and B. Cheswick. "Tracing anonymous packets to their approximate source." Proc. Usenix LISA '00, December 2000.

[Carrier 2002] Carrier, Brian, and Clay Shields. 2002. A Recursive Session Token Protocol For Use in Computer Forensics and TCP Traceback. http://citeseer.nj.nec.com/510558.html

[CERT 1996] CERT. September 19, 1996. "TCP SYN Flooding and IP Spoofing Attacks." CERT Advisory CA-1996-21. http://www.cert.org/advisories/CA-1996-21.html

[Chang 1999] Chang, H.Y., R. Narayan, S.F. Wu, B.M. Vetter, M. Brown, J.J. Yuill, X. Wang, C. Sargor, F. Jou, F. Gong. May 1999. "Deciduous: Decentralized Source Identification for Network-based Intrusions," 6th IFIP/IEEE International Symposium on Integrated Network Management, IEEE Communications Society Press. http://www.silicondefense.com/research/itrex/archive/tracing-papers/chang99deciduos.pdf

[Chang 2000] Chang, H.Y., P. Chen, A. Hayatnagarkar, R. Narayan, P. Sheth, N. Vo, C. L. Wu, S.F. Wu, L. Zhang, X. Zhang, F. Gong, F. Jou, C. Sargor, and X. Wu. January 2000. "Design and Implementation of A Real-Time Decentralized Source Identification System for Untrusted IP Packets." *Proceedings of the DARPA Information Survivability Conference & Exposition (DISCEX 2000). IEEE Computer Society Press*. http://www.silicondefense.com/research/itrex/archive/tracing-papers/chang00design_and_implementation_of_realtime.pdf or http://shang.csc.ncsu.edu/papers/desimpdeciduous.pdf

[Chang undated] Chang, Ho-Yen, S.Felix Wu, C. Sargor, X. Wu. Undated. "Towards Tracing Hidden Attackers on Untrusted IP Networks." http://www.silicondefense.com/research/itrex/archive/tracing-papers/chang00towards_tracing_hidden_attackers.pdf

[Cheswick 1992] Cheswick, Bill. January 1992. "An Evening with Berferd: In Which a Cracker is Lured, Endured, and Studied." *Proceedings of the Usenix Winter 92 Conference.*

[Cisco 1] Cisco Systems. Characterizing and Tracing Packet Floods Using Cisco Routers. http://www.cisco.com/warp/public/707/22.html

[Cisco 2] Cisco Systems. IP Source Tracking on Cisco 12000 Series Internet Routers. http://www.cisco.com/univercd/cc/td/doc/product/software/ios120/120newft/120limit/120s/120s21/ipst.htm

[Cohen 2002] Cohen, Peter. January 25, 2002. "Timbuktu used to recover stolen iMac." *MacCentral.* Mac Publishing LLC. http://maccentral.macworld.com/news/0201/25.timbuktu.php

[DARPA 2002] DARPA. 2002. "DARPA Statement on eDNA and *The New York Times* story of November 22." http://www.darpa.mil/body/pdf/statement.pdf

[Dean 2001] Dean, D., M. Franklin, and A. Stubblefield. February 2001. "An algebraic approach to IP Traceback." *Proceedings of the 2001 Network and Distributed System Security (NDSS) Symposium*. http://www.silicondefense.com/research/itrex/archive/tracing-papers/dean01algebraic_approach.pdf

[Deciduous] Decentralized Source Identification for Network-Based Intrusions. Contact: Chandru Sargor. http://www.anr.mcnc.org/projects/Deciduous/Deciduous.html

[Dietrich 2002] Dietrich, Sven, John McHugh, George Weaver, and Tom Longstaff. March 12, 2002. "Current Active Network Defense Techniques." Active Network Defense (AND) Meeting, June 21, 2002.

[Dittrich] Dittrich, Dave. Distributed Denial of Service (DDoS) Attacks/tools. Website, http://staff.washington.edu/dittrich/misc/ddos

[Doeppner 2000] Doeppner, Thomas W., Philip N. Klein, and Andrew Koyfman. "Using Router Stamping to Identify the Source of IP Packets." 7th ACM Conference on Computer and Communications Security (CCS), Athens, Greece, 2000. pp. 184-189. http://portal.acm.org/citation.cfm?doid=352600.352627

[Dunigan 2001] Dunigan, Tom (thd@ornl.gov). June 2001. Backtracking Spoofed Packets. ORNL/TM-2001/114. http://www.csm.ornl.gov/~dunigan/oci/bktrk.html (listed as "preliminary tech report").

[EPIC 2002] Electronic Privacy Information Center (EPIC). November 5, 2002. "Cookies." http://www.epic.org/privacy/internet/cookies/

[Floyd 2001] Floyd, Sally, Steve Bellovin, John Ioannidis, Kireeti Kompella, Ratul Mahajan, Vern Paxson. July 2001. "Pushback Messages for Controlling Aggregates in the Network." Internet Draft draft-floyd-pushback-messages-00.txt. Submission date Jul. 2001, expiration date Jan. 2002. http://www.silicondefense.com/research/itrex/archive/tracing-papers/draft-floyd-pushback-messages-00.txt

[IETF 1812] Baker, F. June 1995. "Requirements for IP Version 4 Routers." IETF RFC 1812. http://www.ietf.org/rfc/rfc1812.txt

[IETF 2827] Ferguson, P., and D. Senie "Network Ingress Filtering: Defeating Denial of Service Attacks which employ IP Source Address Spoofing." Request for Comments (RFC) 2827. IETF. http://www.ietf.org/rfc/rfc2827.txt

[Honeynet 2003] Honeynet Project. January 7, 2003. *Know Your Enemy: Honeynets*. http://project.honeynet.org/papers/honeynet

[Ioannidis 2002] Ioannidis, John, and Steven M. Bellovin. Feb 6-8, 2002. "Implementing Pushback: Router-Based Defense Against DDoS Attacks." *Proc. of* *the Network and Distributed Systems Security Symposium*. San Diego, CA. http://www.isoc.org/isoc/conferences/ndss/02/proceedings/papers/ioanni.pdf

[Jang 2000] Jang, Heejin and Sangwook Kim. December 2000. "A Self Extension Monitoring for Security Management." *16th Annual Computer Security Applications Conference*, New Orleans, Louisiana. http://www.silicondefense.com/research/itrex/archive/tracing-papers/jang00self-extension_monitoring.pdf

[Jayawal 2002] Jayawal, Vikas, William Yurcik, and David Doss. June 2002. "Internet Hack Back: Counter Attacks as Self-Defense or Vigilantism?" *Proceedings of the IEEE International Symposium on Technology and Society (ISTAS)*, Raleigh, NC.

[Johnson 1999] Johnson, Neil F. 1999. "An Introduction to Watermark Recovery from Images." *Proceedings of the SANS Intrusion Detection and Response (ID'99)*. San Diego, CA, February 9-13, 1999. http://www.jjtc.com/pub/idr99a.htm

[Jung 1993] Jung, H. T., H. L. Kim, Y.M. Seo, G. Choe, S. L. Min, C.S. Kim, and K. Koh. "Caller id system in the internet environment." *UNIX Security Symposium IV Proceedings* (1993), pp. 69-78.

[Junkbusters 2002] Junkbusters. 2002. "How Web Servers' Cookies Threaten Your Privacy" http://www.junkbusters.com/cookies.html

[Kawar 2002] Kawar, Mark. July 26, 2002. "Nebraskans build anti-hacker software." Omaha World Herald.

[Ko] Ko, Calvin, Deborah A. Frincke, Terrence Goan, Jr., L. Todd Heberlein, Karl Levitt, Biswanath Mukherjee, and Christopher Wee. "Analysis of an Algorithm for Distributed Recognition and Accountability." 1st ACM Conference on Computer and Communications Security.

[Lee 2001] Lee, W., and K. Park, "On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack." *Proceedings of IEEE InfoCon 2001*.

[Lee 2002] Lee, Susan C., and Clay Shields. "Technical, Legal, and Societal Challenges to Automated Attack Traceback." *IEEE IT Professiona*l. May/June 2002 (Vol. 4, No. 3) pp. 12-18. http://www.computer.org/itpro/it2002/f3toc.htm

[Markoff 2002] Markoff, John. November 22, 2002. "Agency Weighed, but Discarded, Plan Reconfiguring the Internet." *The New York Times*.

[Mankin 2001] Mankin, Allison, Dan Massey, Chien-Lung Wu, S. Felix Wu, Lixia Zhang. 2001. "On Design and Evolution of 'Intention-Driven' ICMP Traceback." Proceedings of IEEE International Conference on Computer Communications and Networks, 2001. 0-7803-7128-3/01. http://seclab.cs.ucdavis.edu/papers/327-IITrace.pdf

[Mansfield 2000] Mansfield, Glenn, Kohei Ohta, Yohsuke Takei, Nei Kato, and Yoshiaki Nemoto. 2000. "Towards trapping wily intruders in the large", *Computer Networks 34*, pp. 659-670 (2000).

http://www.silicondefense.com/research/itrex/archive/tracing-papers/mansfield00wily_hacker.pdf

[McCullagh 2002] McCullagh, Declan. “Pentagon backs off on Net ID tags.” http://zdnet.com.com/2100-1105-966894.html

[Mirkovic] Mirkovic, Jelena, Janice Martin and Peter Reiher. “A Taxonomy of DDoS Attacks and DDoS Defense Mechanisms.” Computer Science Department, University of California, Los Angeles. Technical report #020018. http://www.lasr.cs.ucla.edu/ddos/ucla_tech_report_020018.pdf

[Merriam-Webster 1983] Merriam-Webster. 1983. Webster’s Ninth New Collegiate Dictionary.Springfield, MA: Merriam-Webster Inc. ISBN 0-87779-508-8.

[Miles 1999] Miles, Stephanie. February 24, 1999. “Intel downplays chip hack report.” C/Net. http://news.com.com/2100-1001-222182.html?legacy=cnet

[Moriarty 2003] Moriarty, Kathleen M. February 10, 2003. “Distributed Denial of Service Incident Handling: Real-Time Inter-Network Defense.” Work in Progress, Internet-Draft. draft-moriarty-ddos-rid-03.txt. Expires August 10, 2003. ftp://ftp.isi.edu/internet-drafts/draft-moriarty-ddos-rid-03.txt

[NAI] Advanced Intrusion Tracing and Response. http://download.nai.com/products/media/nai/pdf/NAI-Labs-AITR-1-5-01.pdf. See also http://www.pgp.com/research/nailabs/adaptive-networks.asp

[Ohta 2000] Ohta, Kohei, Glenn Mansfield, Yohsuke Takei, Nei Kato. July 2000. “Detection, Defense, and Tracking of Internet-Wide Illegal Access in a Distributed Manner.” Proceedings of the 10th Annual Internet Society Conference (INET 2000). http://www.isoc.org/isoc/conferences/inet/00/cdproceedings/1f/1f_2.htm

[ORNL] Oak Ridge National Lab. “Backtracking Spoofed Packets” (web site) http://www.csm.ornl.gov/~dunigan/oci/bktrk.html

[Park 2000] Park, Kihong, and Heejo Lee. June 2000. “On the Effectiveness of Probabilistic Packet Marking for IP Traceback under Denial of Service Attack” Technical Report CSD-00-013, Department of Computer Sciences, Purdue University. http://www.silicondefense.com/research/itrex/archive/tracing-papers/park00effectiveness_technical_paper.pdf. An extended abstract of the report was published with the same title in *Proceedings of the IEEE INFOCOM ’01*, pp. 338-347, 2001.

[Partridge 2001] Partridge, C., C. Jones, D. Waitzman, A. Snoeren. November 2001. “New Protocols to Support Internet Traceback.” http://www.ir.bbn.com/documents/internet_drafts/draft-partridge-ippt-discuss-00.txt

[Provos 2002] Provos, Niels, and Peter Honeyman. “Detecting Steganographic Content on the Internet.” http://isoc.org/isoc/conferences/ndss/02/proceedings

[Purdue 2000] Purdue. Results of the “Attack Traceback Summit Proceedings” of September 6-8, 2000. http://www.cerias.purdue.edu/events/traceback/

[Ricciuti 1999] Ricciuti, Mike. March 7, 1999. "Microsoft admits privacy problem, plans fix." C|Net. http://news.com.com/2100-1040-222673.html?legacy=cnet

[Sager 1998] Sager, Glenn. November 20, 1998. "Security Fun with OCxmon and cflowd." Internet-2 Measurement Working Group. http://www.caida.org/projects/ngi/content/security/1198

[Sanchez 2001] Sanchez, Luis A., Walter C. Milliken, Alex C. Snoeren, Fabrice Tchakountio, Christine E. Jones, Stephen T. Kent, Craig Partrige, and W. Timothy Strayer. "Hardware Support for a Hash-Based IP Traceback." Proceedings of the 2nd DARPA Information Survivability Conference and Exposition (DISCEX II), pp. 146-152, Anaheim, CA, June 2001. http://www.ir.bbn.com/projects/SPIE/pubs/discex01.html

[SANS 2000a] SANS. February 23, 2000. Consensus Roadmap for Defeating Distributed Denial of Service Attacks: A Project of the Partnership for Critical Infrastructure Security. Version 1.10. http://www.sans.org/ddos_roadmap.htm

[SANS 2000b] SANS. Egress Filtering v 0.2. February 29, 2000. http://www.sans.org/y2k/egress.htm

[Savage 2000] Savage, Stefan, David Wetherall, Anna Karlin and Tom Anderson, "Practical Network Support for IP Traceback", *Proceedings of the 2000 ACM SIGCOMM Conference*, pp. 295-306, Stockholm, Sweden, August 2000. See [Savage 2001]. http://www.cs.washington.edu/homes/savage/traceback.html

[Savage 2001] Stefan Savage , David Wetherall , Anna Karlin , and Tom Anderson. June 2001. Network Support for IP Traceback. *IEEE/ACM Transactions on Networking (TON)*. Volume 9, Issue 3.

[Schaeffer 2000] Schaeffer, Richard C., Jr. "TESTIMONY of Richard C. Schaeffer, Jr., Director, Infrastructure and Information Assurance, Office of the Assistant Secretary of Defense (Command, Control, Communication, and Intelligence) before a hearing of the Subcommittee on Government Management, Information, and Technology, July 26, 2000, Computer Security: Cyber Attacks - War without Borders."

[Schnackenberg 2000] Schnackenberg, D., K. Djahandari, and D. Sterne. "Infrastructure for intrusion detection and response." *Proc. First DARPA Information Survivability Conference and Exposition*, Jan. 2000. http://www.silicondefense.com/research/itrex/archive/tracing-papers/schnackenberg00DISCEX_intrusion_detection_and_response.pdf

[Sharp] Sharp, Walter Gary. (then at MITRE) "Key Legal Implications of Computer Network Defense." http://www.blackhat.com/html/bh-multi-media-archives.html

[Silicon Defense] Silicon Defense. *Traceback and Related Papers Archive.* http://www.silicondefense.com/research/itrex/archive/tracing-papers

[Smith] Smith, Richard M. "FAQ: Web Bugs." Verified October 24, 2002. http://www.privacyfoundation.org/resources/webbug.asp

[Snapp 1991a] Snapp, Steven R., James Brentano, Gihan V. Dias, Terrance L. Goan, L. Todd Heberlein, Che-Lin Ho, Karl N. Levitt, Biswanath Mukherjee, Stephen E. Smaha, Tim Grance, Daniel M. Teal, and Doug Mansur. "DIDS (Distributed Intrusion Detection System) - Motivation, Architecture, and an Early Prototype." *Proceedings of the 14th National Computer Security Conference*. Washington, DC, Oct. 1991, pp. 167-176. http://seclab.cs.ucdavis.edu/papers/DIDS.ncsc91.pdf

[Snapp 1991b] Snapp, Steven R., James Brentano, Gihan V. Dias, Terrance L. Goan, Tim Grance, L. Todd Heberlein, Che-Lin Ho, Karl N. Levitt, Biswanath Mukherjee, Douglass L. Mansur, Kenneth L. Pon, and Stephen E. Smaha. "A system for Distributed Intrusion Detection." In COMPCOM Spring '91 Digest of Papers, pages 170-176, February/March 1991. http://seclab.cs.ucdavis.edu/papers/pdfs/ss-jb-91.pdf

[Snapp 1992] Snapp, Steven R., Stephen E. Smaha, Daniel M Teal, and Tim Grance. 1992. "The DIDS (Distributed Intrusion Detection System) Prototype." In *Proceedings of the Summer USENIX Conference*, pages 227-233, San Antonio, Texas, 8-12 June 1992. USENIX Association.

[Snoeren 2001] Snoeren, Alex C., Craig Partridge, Luis A. Sanchez, Christine E. Jones, Fabrice Tchakountio, Stephen T. Kent, W. Timothy Strayer, "Hash-Based IP Traceback" http://www.acm.org/sigcomm/sigcomm2001/p1.html

[Song 2001] Song, D.X., and A. Perrig. "Advanced and authenticated marking schemes for IP traceback." *Proc. IEEE Infocom '01*, April 2001. http://www.cs.berkeley.edu/~dawnsong/papers/tr-iptrace.ps

[Staniford-Chen 1995a] Staniford-Chen, S.G., "Distributed Tracing of Intruders." Master's Thesis, University of California, Davis. 1995. http://www.silicondefense.com/research/itrex/archive/tracing-papers/staniford95distributed_tracing_of_intruders.pdf

[Staniford-Chen 1995b] Staniford-Chen, S.G., and L. Heberlein. "Holding Intruders Accountable on the Internet." *Proceedings of the 1995 IEEE Symposium on Security and Privacy* (Oakland, CA, May 1995), pp. 39-49. http://seclab.cs.ucdavis.edu/papers/thumb.ieee95.pdf

[Staniford 2000] Staniford, Stuart. *Internt Trap and Trace*. http://www.silicondefense.com/pptntext/TrapTrace_9_07_00.ppt

[Staniford 2002] Staniford, Stuart, Vern Paxon, and Nicholas Weaver. "How to Own the Internet in Your Spare Time." *Proceedings of the 11th USENIX Security Symposium* (Security 02). http://www.icir.org/vern/papers/cdc-usenix-sec02

[Sterne 2001] Sterne, Dan, Kelly Djahandari, Brett Wilson, Bill Babson, Dan Schnackenberg, Harley Holliday, and Travis Reid. "Autonomic Response to Distributed Denial of Service Attacks." RAID 2001. LNCS 2212, pp 134-149. http://www.cse.ogi.edu/~wuchang/cse581_winter2002/papers/22120134.pdf

[Sterne 2002] Sterne, Dan, Kelly Djahandari, Ravindra Balupari, William La Cholter, Bill Babson, Brett Wilson, Priya Narasimhan, Andrew Purtell, Dan Schnackenberg, Scott Linden. "Active Network Based DDoS Defense." Proceddings of the DARPA

Active Networks Conference and Exposition (DANCE 02). ISSN 0-7695-1564-9/02. IEEE Computer Society.

[Stoll 1990] Stoll, Clifford. 1989, 1990. *The Cuckoo's Egg: Tracking a Spy Through the Maze of Computer Espionage*. New York: Doubleday.

[Stone 1999] Stone, Robert. October 1, 1999. "CenterTrack: An IP Overlay Network for Tracking DOS Floods." http://www.us.uu.net/gfx/projects/security/centertrack.pdf. See also its republication in the Proceedings of the 9th Usenix Security Symposium, August 2000. http://www.usenix.org/publications/library/proceedings/sec2000/stone.html

[Templeton 2003] Templeton, Steven J., and Karl E. Levitt (U.C. Davis) "Detecting Spoofed Packets." *Proceedings of The Third DARPA Information Survivability Conference and Exposition* (DISCEX III), Washington, D.C., April 22-24, 2003. http://seclab.cs.ucdavis.edu/papers/DetectingSpoofed-DISCEX.pdf

[Thomas 2001] Thomas, Rob. February 8, 2001. Tracking Spoofed IP Addresses Version 2.0. http://www.cymru.com/~robt/Docs/Articles/tracking-spoofed.html

[Van 1997] Van, Van C. May 29, 1997. A Defense Against Address Spoofing Using Active Networks. MIT Master's Thesis. http://www.sds.lcs.mit.edu/publications/van97.html

[Wang 2001a] Wang, Xinyuan, Douglas S. Reeves, S. Felix Wu, Jim Yuill. 2001. "Sleepy Watermark Tracing: An Active Network-based Intrusion Response Framework." http://www.cs.ucdavis.edu/~wu/publications/2001-03-watermark-ifipsec.pdf or http://arqos.csc.ncsu.edu/papers.htm

[Wang 2001b] Wang, X., D. Reeves, S.F. Wu, "Tracing Based Active Intrusion Response," in *Journal of Information Warfare*, Volume 1, Issue 1, September 2001, pp. 50-61. http://arqos.csc.ncsu.edu/papers.htm

[Wang 2002] Wang, Xinyuan, Douglas S. Reeves, Shyhtsun Felix Wu. 2002. "Inter-Packet Delay Based Correlation for Tracing Encrypted Connections through Stepping Stones." *7th European Symposium on Research in Computer Security (ESORICS 2002)*, Zurich, Switzerland, October 14-16, 2002, Proceedings. Lecture Notes in Computer Science. Springer 2002. ISBN 3-540-44345-2. pp. 244-263. http://arqos.csc.ncsu.edu/papers.htm

[Wheeler 2002] Wheeler, David A. July 2002. *Secure Programming for Linux and Unix HOWTO*. http://www.dwheeler.com/secure-programs

[Wright 2002] Wright, Matthew, Micah Adler, Brian N. Levine, Clay Shields. "An Analysis of the Degradation of Anonymous Protocols." http://isoc.org/isoc/conferences/ndss/02/proceedings

[Wu 2001] Wu, S. F., L. Zhang, D. Massey, and A. Mankin. "Intention-driven ICMP traceback." Internet Draft, IETF, Feb. 2001. draft-wu-itrace-intention-00.txt. Work in progress. http://www.silicondefense.com/research/itrex/archive/tracing-papers/draft-wu-itrace-intention-00.txt

[Yoda 2000] Yoda, Kunikazu, and Hiroaki Etoh. (IBM Japan). October 2000. "Finding a Connection Chain for Tracing Intruders." In *6th European Symposium on Research in Computer Security – ESORICS 2000*, Toulouse, France. Edited by F. Guppens, Y. Deswarte, D. Gollman, and M. Waidner. http://www.trl.ibm.com/projects/security/chaintrace/

[Zalewski 2001] Zalewski, Michal. 2001. Strange Attractors and TCP/IP Sequence Number Analysis. http://razor.bindview.com/publish/papers/tcpseq.html

[Zhang 2000] Zhang, Yin, and Vern Paxon. 2000. "Detecting Stepping Stones." Proceedings of the 9th Usenix Security Symposium, 2000. http://www.icir.org/vern/papers/stepping

Acronyms and Abbreviations

ACC	Aggregate-based Congestion Control
ACK	Acknowledge(d)
AH	Authentication Header
ANSI	American National Standards Institute
BGP	Border Gateway Protocol
CC	Common Criteria
CEF	Cisco Express Forwarding
CERT	formerly Computer Emergency Response Team; now just CERT
CISIE	Caller Identification System in the Internet Environment
CISL	Common Intrusion Specification Language
CITRA	Cooperative Intrusion Traceback and Response Architecture
CNA	Computer Network Attack
CND	Computer Network Defense
CPU	Central Processing Unit
DARPA	Defense Advanced Research Projects Agency
dCEF	Distributed CEF
DDoS	Distributed Denial of Service (a type of attack)
DHCP	Dynamic Host Configuration Protocol
DIAP	Defense-wide Information Assurance Program
DoD	Department of Defense
DOS	Denial of Service
ftp	File Transfer Protocol
HTML	HyperText Markup Language
HTTP	HyperText Transport Protocol
ICMP	Internet Control Message Protocol
ID	Identification
IDA	Institute for Defense Analyses
IDIP	Intruder Detection and Isolation Protocol
IDMEF	Intrusion Detection Message Exchange Format
IDS	Intrusion Detection System
IEEE	Institute of Electrical and Electronics Engineers, Inc.
IETF	Internet Engineering Task Force
IP	Internet Protocol
IPSEC	IP Security
IPv4	Internet Protocol, version 4
IPv6	Internet Protocol, version 6
ISO	International Organization for Standardization
ISP	Internet Service Provider

ISP	Internet Service Provider
iTrace	ICMP Traceback
LAN	Local Area Network
MLSI	Mark Left by Suspected Intruder
MTA	Mail Transfer Agent
NAI Labs	Network Associates Laboratories
ORNL	Oak Ridge National Laboratory
OSI	Open Systems Interconnection
P2P	Peer to Peer
PP	Protection Profile
PPM	Probabilistic Packet Marking
PR/TT	Pen Register/Trap and Trace
RFC	Request for Comments
RID	Remote. Intrusion. Detector
SPIE	Source Path Isolation Engine
STOP	Session Token Protocol
SWT	Sleepy Watermark Tracing
Syn	Synchronize
TCP	Transmission Control Protocol
UDP	User Datagram Protocol
U.S.	United States
WAN	Wide Area Network
W3C	World Wide Web Consortium

REPORT DOCUMENTATION PAGE	*Form Approved OMB No. 0704-0188*

1. REPORT DATE (DD-MM-YY) October 2003	2. REPORT TYPE Study	3. DATES COVERED (From – To)
4. TITLE AND SUBTITLE Techniques for Cyber Attack Attribution		5a. CONTRACT NUMBER DASW01-98-C-0067/ DASW01-02-C-0012/ DASW01-04-C-0003
		5b. GRANT NUMBER
		5c. PROGRAM ELEMENT NUMBERS
6. AUTHOR(S) David A. Wheeler Gregory N. Larsen, Task Leader		5d. PROJECT NUMBER
		5e. TASK NUMBER BC-5-1767
		5f. WORK UNIT NUMBER
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESSES Institute for Defense Analyses 4850 Mark Center Drive Alexandria, VA 22311-1882		8. PERFORMING ORGANIZATION REPORT NUMBER IDA Paper P-3792
9. SPONSORING / MONITORING AGENCY NAME(S) AND ADDRESS(ES) Defense-Wide Information Assurance Program 1215 Jefferson Davis Highway Crystal Gateway 3, Suite 1101 Arlington, VA 22202		10. SPONSOR'S / MONITOR'S ACRONYM ASD(C3INII)
		11. SPONSOR'S / MONITOR'S REPORT NUMBER(S)

12. DISTRIBUTION / AVAILABILITY STATEMENT

Approved for public release, unlimited distribution: 22 March 07.

13. SUPPLEMENTARY NOTES

14. ABSTRACT

This paper summarizes various techniques to perform attribution of computer attackers who are exploiting data networks. Attribution can be defined as "determining the identity or location of an attacker or attacker's intermediary". It concludes that there are many attribution techniques, attribution is difficult and inherently limited, attribution tends to be easier against insiders, and prepositioning is necessary for many attribution techniques. Many techniques are immature and will require funding before deployment. A useful first step for the DoD would be to "change the terrain" of its own network to ease attribution.

15. SUBJECT TERMS

Attribution, traceback, trace back, source track, source tracking, location, identity, attacker, insider, cyber attack, denial of service, distributed denial of service, DDoS, computer network defense, computer network attack, information assurance, computer security, stepping stone, zombie, laundering host, Department of Defense, DoD, change the terrain, prepositioning, internet, TCP/IP, iTrace, network ingress filtering, intrusion detection, stream matching, input debugging, honeypot, cookie.

16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT	18. NUMBER OF PAGES	19a. NAME OF RESPONSIBLE PERSON John Hunter
a. REPORT Unclassified	b. ABSTRACT Unclassified	c. THIS PAGE Unclassified	Unlimited	88	19b. TELEPHONE NUMBER (Include Area Code) (703) 602-9927