Toward a Framework for Internet Forensic Analysis

Vyas Sekar Yinglian Xie David A. Maltz Michael K. Reiter Hui Zhang
Carnegie Mellon University

ABSTRACT

The world of network security is an arms race where attackers constantly change the signatures of their attacks to avoid detection. Aiding the white-hats in this race is one fundamental invariant across all network attacks (present and future): for the attack to progress there must be communication among attacker, the associated set of compromised hosts and the victim(s), *and this communication is visible to the network.* We argue that the Internet architecture should be extended to include auditing mechanisms that enable the forensic analysis of network data, with a goal of identifying the true originator of each attack — even if the attacker recruits innocent hosts as zombies or stepping stones to propagate the attack. In this paper we outline an approach to the problem of *Attacker Identification* and *Attack Reconstruction,* describe the challenges involved, and explain our efforts that show the promise of this approach.

1. INTRODUCTION

Many types of Internet attacks utilize indirection as a means to hide their source. For example, the act of utilizing a chain of compromised machines, or "stepping stones," in an attack is a common means of foiling a defender's attempts to locate the source of an attack. Similarly, distributed denial-of-service (DDoS) attacks are often launched from compromised computers, sometimes called "zombies", both to harness the power of many machines and to obfuscate where the true source of the attack lies. Today, such indirection is a highly successful means to provide anonymity to attackers.

In this paper we define an approach with the promise to dramatically change investigations of Internet-based attacks. Our high-level vision is an investigative capability for the Internet or intranets that permits identification and fine-grained analysis of the communication patterns leading to an attack, particularly including those attacks that utilize indirection as a means to hide their true source. Our goal is to bridge indirection techniques, such as stepping stones and zombies, to locate the original source of the attack or entry point to an intranet and to reconstruct how an attack unfolded.

We believe that this capability can significantly strengthen the hand of administrators in deterring attacks or permitting the correction of weak points in a network perimeter. For example, if an attack that propagated within an intranet can be traced back to its initial entry into the intranet—e.g., revealing an errant service or modem permitting connections that circumvent the firewall, or a user who is careless in the email attachments he opens—then that entry point can be corrected. And, of course, the ability to trace large-scale attacks to their ultimate source is a first step toward holding that attacker accountable.

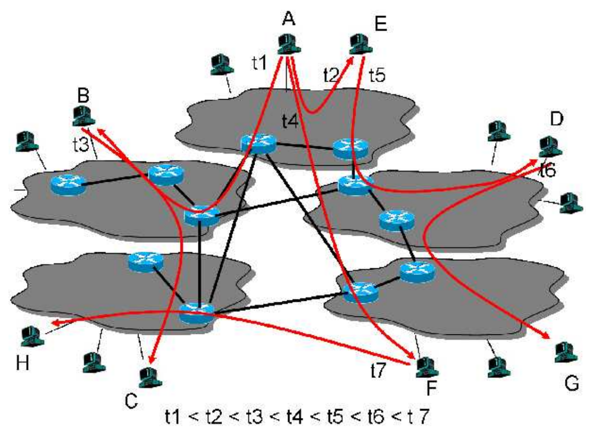
The key observation is that attacks utilizing indirection often exhibit network behavior that may not seem suspicious at the level of individual flows, but that nevertheless exhibit a discernible pattern when traffic is observed collectively. We outline a number of ways this observation can be exploited to detect and trace attacks, and we define the required infrastructure and its properties. Finally, we show promising early results from one of the several methods we are investigating.

Our approach differs from existing capabilities in significant ways. While "traceback" [20, 5, 21] and related techniques attempt to identify the source(s) of packets received by a victim, in modern attacks this source is almost always a zombie, not the real attacker. Instead, our goal is to bridge this indirection and continue the trace back to the true source of the attack. Other approaches to attack and intrusion detection tend to focus on improving mechanisms for local observations or developing better methods for worm signature detection [11, 10]. Our approach can build from whatever local observations are available to extract the global outline of the attack, and then ultimately determine the series of hosts through which the attack propagated.

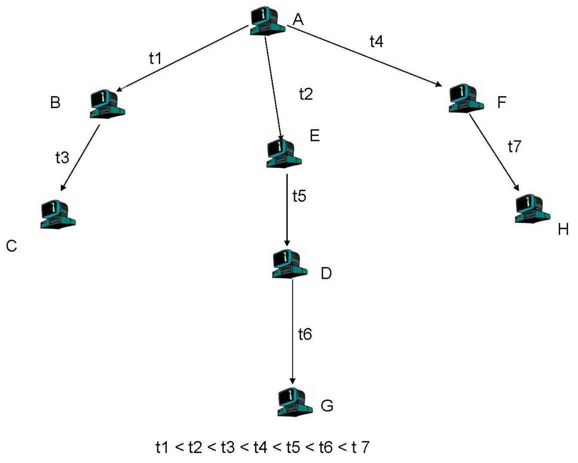
2. PROBLEM DEFINITION

We define the problem of tracing and reconstruction of arbitrary network attacks in terms of two fundamental components, *Attacker Identification* and *Attack Reconstruction*. These act as building blocks on which attack investigation can be based and attackers held accountable. Attacker Identification is the ability to accurately pinpoint the source(s) of the attack or infection. Attack Reconstruction is the process of inferring which communications carry the attack forward. This not only identifies the compromised hosts for subsequent correction, but also provides crucial information about the attack propagation that can help in precluding future attacks of a similar kind. The focus of our work on identifying the true source of attacks through Attacker Identification and Attack Reconstruction differentiates it from other projects that seek to identify when an attack is occurring or to reactively blunt the effect of an attack already in progress.

Figure 1 (a) shows a general model of how an arbitrary



(a)



(b)

Figure 1: (a) A simplified network topology showing routers, hosts, and ISP boundaries. The arrows between hosts illustrate communications that spread an attack through the network, (b) the host attack tree showing the propagation of the attack reaching the end systems B-H

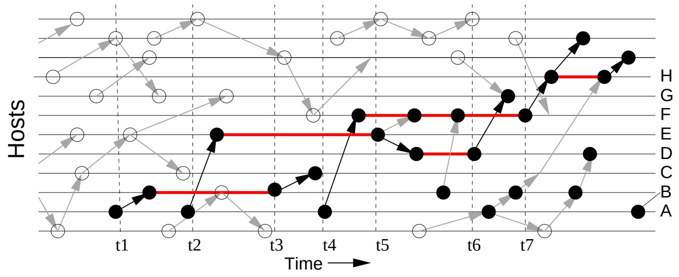


Figure 2: An example host contact graph, with flows that advance the attack marked with darker arrows, and the causal relationship marked with thick lines

network attack propagates from one host to another across administrative domains. We mark each of the attack flows with a timestamp to indicate when the attack was received by the victim. The attack originates at host A at time t_1, and then propagates to hosts B to H through a *host attack tree* shown in Figure 1 (b). Figure 2 shows the *host contact graph* of a hypothetical network undergoing an attack. Time advances left to right and each horizontal line represents a host in the network, with each arrow representing communication between two hosts. The highlighted nodes represent compromised hosts and the highlighted arrows represent the edges that propagate the attack, which correspond to edges in the host attack tree in Figure 1 (b). For example, at time t_5, host E contacts both host F and D. However, only the arrow from E to D is highlighted because D gets infected by this contact whereas F was infected at time t_2, before t_5.

Given the host contact graph, Attack Reconstruction identifies which edges advanced the attack, and marks those edges in black along with each infected host from the time it was infected on. Even infected hosts may continue their normal activities and not all infection attempts succeed, so Attack Reconstruction must carefully infer which communication initiated by an infected host should be marked as carrying the attack, and reconstruct the host attack tree using the marked edges. Attacker Identification operates by working its way back up the host attack tree to find the root sources of the attack. Even when the ultimate root and source of the attack tree cannot be found (e.g., due to missing data), the higher level nodes of the attack tree(s) that are reconstructed point the way toward the true attack source and provide a starting point for out-of-band human investigation.

3. A FRAMEWORK FOR INVESTIGATING ATTACKS

Identifying the propagation of an attack is particularly difficult as the adversary is intelligent: attackers are bound to come up with smarter mechanisms trying to evade detection. However, there is one fundamental invariant across all attacks (present and future): for the attack to progress there must be communication among attacker, the associated set of compromised hosts and the victim(s), and *this communication is visible to the network*.

The communication between attackers and victims may be subtle and invisible when observed from any single host without foreknowledge of the attack signature, but potentially it will stand out when viewed globally as the attack propagates. As a simple example, efforts to detect stepping stones have been premised on the identification of flows that exhibit closely correlated packet contents or inter-packet timings [26, 7]; while each flow in isolation may seem innocuous, together they reveal suspicious behavior. In general, our approach is to develop algorithms that correlate the communication events among individual hosts and identify the patterns that indicate a propagating attack. Since no accurate signature is required, our approach has the potential to be robust against changes in the behavior of attacks.

Applying our approach to a large scale network requires a means to: (1) gather and query host communication records from distributed network locations; and (2) design analysis algorithms for identifying global communication patterns given the host contact graph.

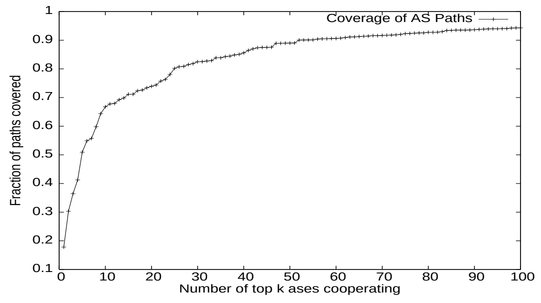


Figure 3: Path coverage vs number of highest degree ASes selected.

3.1 Network Auditing

We need widely deployed infrastructure support where distributed collection points log traffic records and store them in repositories for querying. We focus on end-host connectivity patterns in terms of directional network "flows", where each flow identifies a directional communication event between a source and a destination, and carries timestamps indicating the start and stop time of the flow. Compared with packet traces, flow records are more compact representations and need not capture packet payloads, which might be construed as a violation of end-user privacy.

At the scale of an intranet, traffic logging can be deployed as pervasively as necessary. At the Internet scale, the problem of obtaining an approximation of the complete host contact graph appears intractable. However, there are features of the Internet topology and routing that suggest auditing devices can be deployed at a relatively small number of locations to approximately construct the host contact graph. First, Internet routing is highly constrained by inter-domain policies and the AS hierarchy, which implies that top level ASes will observe a significant fraction of the traffic. Second, the Internet AS-level connectivity graph has been shown to be a power-law graph [8], suggesting that high degree ASes are good candidates for logging deployment.

To estimate how many observation points should be deployed in the Internet for a given fraction of the flows to be logged at least once, we investigate the optimal path coverage problem, related to the problem of deploying reverse path filters [16] and containment filters [14], assuming flows uniformly distribute among all available Internet paths. We define an AS graph $G = \langle V, E \rangle$, where V is the set of ASes, and E is the set of inter-AS connections. Let P be the set of paths used for routing traffic in G. We say an AS a ($a \in V$) *covers* a path p ($p \in P$) if a is on the path p. Given these notions, we want to select the smallest AS subset $V' \subseteq V$ that covers α percent of the paths in P.

We consider a greedy heuristic where the ASes are selected based on their degrees, with higher degree ASes given higher preference. Figure 3 shows the fraction of AS-paths that are "covered" by the first k high-degree ASes, using AS paths and AS degrees obtained from RouteViews [19] in Feb 2004. The figure shows that if the 50 ASes with highest node degrees deployed flow auditing, these ASes would "cover" approximately 90% of all the paths, and hence be able to log any flow traversing one of these paths.

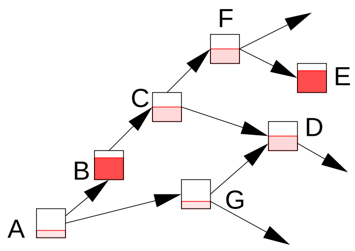


Figure 4: Concept behind attack reconstruction: the probability of being involved in an attack should be reinforced by network events.

To help reveal the causal relationship between flows for attack reconstruction, ideally all logging devices in the auditing infrastructure should have perfect time synchronization (perhaps achieved using GPS time sources or NTP). However, it appears both practical and sufficient to ensure two flows are timestamped consistent with the causal relationship between them (if any) [12]. "Causal consistent timestamps" mean that if a flow A is, in fact, caused by flow B, then A will have a timestamp later than B. This property is achieved if there is any single router that records both flows of interest or if there are any two time-synchronized routers (e.g., they are in the same administrative domain) where flow A is recorded by one router and flow B by the other.

3.2 Techniques for Attack Reconstruction

The key component of the analysis lies in the algorithmic components that analyze the data log to extract "interesting" communication patterns that are part of an ongoing network attack. The attack reconstruction algorithms may operate over a centralized data repository (in the case of an intranet), or will need to interact with distributed query routing and data retrieval mechanisms (in the case of a larger internet).

The essence of our reconstruction method is to reveal the causal relationship between communications by correlating local observations. Figure 4 illustrates the concept. The fraction of each host that is filled in represents the probability that that host is involved in the attack. These initial probability estimates are based on locally observed behavior, such as those a local host-based Intrusion Detection System (IDS) might observe. Based on the local observations alone, it is likely B and E are involved in an attack. Given the communication pattern among the hosts, however, it is extremely likely that hosts C, D, and F are also involved, although their locally observable behavior was not suspicious enough to trigger an IDS.

3.2.1 *Initial Local Observations*

The input to Attack Reconstruction is one or more estimates of the probability that a host is involved in an attack. We can leverage prior work in this area, as these estimates can come from any existing IDS or network monitoring system that detects the occurrence of attacks in the

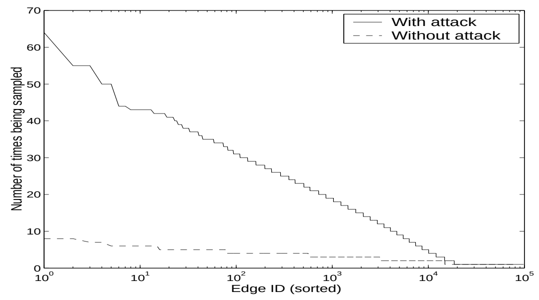


Figure 5: Distribution of link samples from a simulated host contact graph with 10,000 hosts.

network. In our current prototype, we calculate the probability of involvement from the observable behavior of the node as found in the audit logs.

We have been evaluating measures including the fanout of the host in any given time interval, the rate at which the host generates traffic, and the set of destinations that the host contacted within a time interval. For each of these measures, we use anomaly detection mechanisms to observe deviations from normal behavior and compute a confidence estimate for the probability an host is involved in an attack.

3.2.2 Attack Reconstruction Using Host Contact Graphs

Given the initial probabilities of hosts involved in an attack, the host contact graph obtained from the audit logs is used to refine the involvement probability estimates and reconstruct the attack. The goals are to: (1) identify the edges that successfully attack a new host (i.e., are causal) and (2) the top-level nodes of the host attack tree. We are currently investigating several classes of algorithms for Attack Reconstruction and Attacker Identification:

Maximum-likelihood methods: These methods use the involvement probabilities and the host contact graph to compute the most probable path that the attacker used to reach each victim. We are exploring the use of Bayes Nets, Reinforcement Models, and Statistical Likelihood computation to create a feedback loop. Here, the global information from the packet logs is used to improve the confidence in the involvement probabilities, and also identify nodes which are involved in the attack but do not exhibit directly observable anomalous behavior. Such a feedback loop may help refine the host attack tree being reconstructed and reduce the false-negative and the false-positive rate.

Random walk sampling method: Our random walk sampling technique stems from the observation that a *host attack graph* — a subgraph of the host contact graph formed by the edges making up a wide-spread attack — is significantly larger than most of the other subgraphs in the host contact graph that are formed by normal communication. The goal of the technique is to determine if a host attack graph exists in the contact graph, and separate it out from the other innocent subgraphs.

The technique works by randomly selecting edges (communication flows) in the host contact graph and from each edge conducting a random walk backwards in time along the host contact graph. Since a host attack graph is large compared with normal host contact subgraphs, random walks are more likely to stumble across edges/nodes from the host attack graph and follow it backwards than normal contact subgraphs. Correlating the walks to identify the most common subpaths identifies the hosts closest to the source of the attack. Since the attack is tree structured, paths will tend to converge at the higher levels of the host attack tree — meaning that subpaths that occur more frequently are closer to the root. A variety of sampling techniques are being investigated to improve the performance of the algorithm, from biasing the start of the random walks toward hosts with higher initial estimates of infection to alternate ways of selecting which edges the walk should traverse.

Figures 5 and 6 show preliminary results of the random walk sampling technique applied to the host contact graph generated by a simulated worm attack. Figure 5 shows the distribution of the frequency with which each edge appeared during a set of random walks (covering 5% of the edges) performed through the host contact graph (distribution shown on a semi-log scale, and truncated at 10^5 edges). Before an attack takes place, the host contact subgraphs are much smaller. The frequency of an edge being traversed during the random walks tends to distribute more evenly than the frequency of an edge being traversed during times of a worm propagation. Edges that occur most frequently in random walks are more likely to involve infected hosts. By picking only those nodes involved in high frequency edges, we extract a list of infected nodes that has a very small false negative rate (2%). The list of infected nodes and timing information from the host contact graph is then used by a backtracking algorithm to reconstruct the host attack tree — Figure 6 shows the higher levels of the host attack tree from the source to the first few infected nodes.

4. DISCUSSION

Deploying an Internet-scale forensic analysis system for Attacker Identification and Attack Reconstruction is a non-trivial task. There will be a number of challenges in practice as a result of the sheer scale of the Internet traffic and the need for cooperation between different service providers.

a) A tremendous amount of data would be required to represent the complete host contact graph of the Internet. Our proposed method for identifying attack propagation paths relies on a network auditing system to log end host communication records. To bound the amount of audit data a network might observe, let us assume that a major ISP has O(100) POPs, and each POP has 10 Gbps capacity towards the middle of the ISP's network. The total amount data flowing through the network could then be 10^2 POPs/ISP $\times$ 10^{10} bits/sec/POP $= 10^{12}$ bits/sec. Now, supposing an average packet size of 1000 bits and an average flow of 10 packets, that would be 10^{12} bits/sec $/(10^1$ packets/flow $\times$

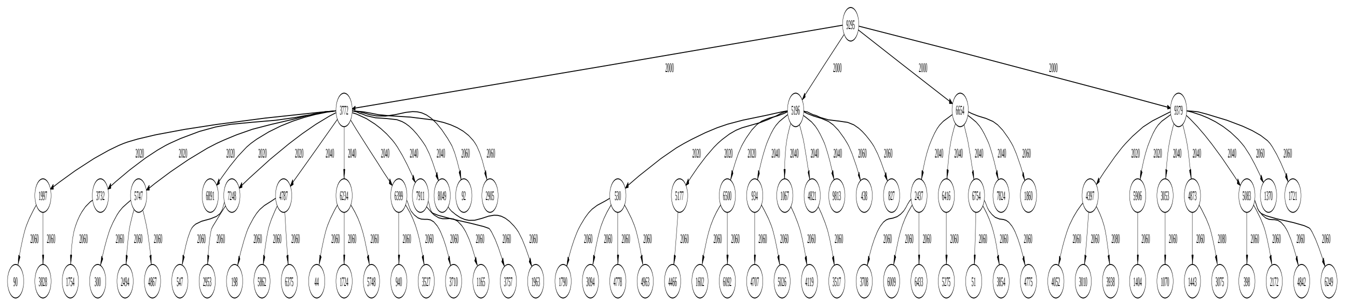


Figure 6: The output result of a simulation study using random walk based sampling, showing the the top levels of attack tree. The top-level sources and spreaders are identified with very high accuracy.

10^3 bits/packet) $= 10^8$ flows/sec.

So, if we wished to store a record for each flow sent over the network, it would require 10^8 flow records/sec $\times 10^2$ bits/flow record $= 10^{10}$ bits/sec , i.e., 10 Gbps for the ISP or 1% of the overall network capacity (100×10 Gbps). In order to keep one hour of data, the storage obligation for the ISP would be roughly 4.5 TB, distributed among the POPs.

Thus the sheer volume of the data to be collected, stored and analyzed represents a problem of scalable and efficient data collection and analysis, but the quantities involved are by no means inconceivable.[1]

b) The complete host contact graph for the entire Internet will not be available. Given the large number of ISPs and administrative domains (ADs) that make up the Internet, it is likely that auditing will be deployed in a piecemeal fashion across the network. Some ADs will have very complete auditing, others will have none, and many will audit packets only at their borders and peering points with other ADs. Attack Reconstruction algorithms will need to operate in an environment where some communications between hosts are logged multiple times, and others are not logged at all. For example, our proposed random walk method relies on statistical sampling of the traffic traces, making it intrinsically robust to many kinds of missing data. Even when critical data is systematically missing, the method still produces a partial host attack tree that can be used as a basis for further out-of-band investigation.

In the same way that distributing auditing functionality is difficult, the amount of data involved means that the audit information cannot ever be centrally located. Reconstruction algorithms will need to be designed to either query distributed repositories of audit data, or to ship partial reconstruction state between the audit repositories.

c) Privacy concerns must be addressed. Issues of trust and cooperation between domains raise challenges with respect to protecting both domain proprietary information and end user privacy. It is particularly important to prevent the execution of queries that can either retrieve an arbitrary part of the entire host contact graph recorded by an AD (hence leaking business data about the AD) or read out all flows to or from an arbitrary host (hence violating the privacy of a normal host). For example, (1) an AD can require that a flow record to or from a host be provided as a query input before more flow records involving the host are returned; (2) an AD can rate limit queries, or accept queries from only trusted networks; and (3) an AD can choose to elide records from the set of flows that match a query, instead of returning the entire set. In particular, during random walk sampling of the data, we need to randomly select a preceding flow to continue the walk. On reception of a query requesting all incoming flows to a specified host, an AD can randomly select a small number of candidate flows from its logs, which will be aggregated with those from other ADs for another round of random selection.

The creation of a network auditing service also serves as a practical application of work by ourselves and others [22] to develop techniques that limit the disclosure of private information while still allowing useful inferences to be drawn across sets of data.

d) The security of the auditing system itself must be maintained. If data is to be useful to law enforcement authorities, the auditing system must be constructed to maintain a "chain of custody" that can convince a jury that the data cannot have been tampered with after being collected. It must also be impossible for an outside attacker to "frame" an uninvolved host by creating traffic that implicates the host. Further, the auditing system itself will likely become a favorite point of attack. It must be defended against attackers who might first DoS attack the auditing system, then launch attacks without running the risk of detection.

5. RELATED WORK

The problem of identifying anomalous events that signal the onset of an attack has been addressed in [3, 9]. There has also been work that exploits other specific characteristics of attacks [15, 25] for attack detection. These studies are complementary to the effort of Attack Reconstruction in that they may provide guidance on the start time, the type, or the degree of virulence of an attack, which can be used to tune the performance of Attack Reconstruction.

One of the important problems in this domain is that of IP traceback, which is the identification of the path from

[1]Today, 5 TB of disk space costs $5,000 to $10,000.

an attacker to the victim. The knowledge of the routers on the path can be used to stop the attack flows and also deal with attacks which spoof their source addresses [21, 20, 5, 4]. These techniques expose activity at the bottom-most levels of attacks that may use indirection, while our work is focused on bridging indirections to seek the original attack source. While the use of packet-digests to enable single-packet IP traceback [21] appears especially similar to the use of flow logging to enable attack reconstruction, we highlight two distinctions. First, working with flow-level data significantly reduces the amount of data that needs to be stored. Second, our approach is more robust to missing data and non-cooperative administrative domains, as the path between the source and the destination usually passes through multiple domains. Our techniques only require one of the domains to log the flow, whereas single-packet IP traceback requires packet digests to be logged at every router on the path.

There are studies that use epidemiological models to capture the spread and rapid propagation of Internet worms [23, 6, 14]. These works, although mostly theoretical, shed light on the speed of propagation and the scale of attacks.

Intrusion detection systems [17, 18, 24] have been widely deployed both on hosts and network gateways to monitor the activity of hosts within a network, and also provide means of enabling filters to cull out "bad" traffic (both incoming and outgoing). There have also been recent suggestions for the deployment of active policing of sources [13] and the introduction of stricter access control in allowing connections from untrusted hosts [1, 2]. These techniques can provide mechanisms for actively preventing or defending attacks once the weak entry points in the network have been identified via reconstruction of attack propagation paths.

6. SUMMARY

In this paper, we have argued the importance of a system for identifying attack propagation paths and attack sources. While end-system based approaches to defense and response show promise in the short-term, we believe that an architecture for network audit analysis is a fundamental property that should be built into the network. The capabilities of Attacker Identification and Attack Reconstruction will provide accountability for attacks in both wide area networks and intranets with the promise of both deterring future attackers and speeding the recovery after an attack occurs.

However, the distributed and heterogeneous nature of the Internet poses significant challenges toward realizing Internet-scale forensic analysis systems, requiring research along many aspects of network security. Smart attackers are bound to come up with mechanisms that outwit existing signature-based detection and analysis techniques. However, network attacks will always involve communication between the attackers and the attacked. By addressing the problem from the perspective of understanding the communication patterns this framework, once realized, can be used in coordination with other well-understood attack response mechanisms (e.g. address blacklisting, rate limiting) to contain the damage caused by attacks, aid in recovery following attacks, and ultimately identify and deter the attackers themselves.

7. REFERENCES

[1] D. Adkins, K. Lakshminarayanan, A. Perrig, and I. Stoica. Taming IP Packet Flooding Attacks. In *ACM SIGCOMM HotNets II*, 2003.

[2] T. Anderson, T. Roscoe, and D. Wetherall. Preventing Internet Denial-of-Service with Capabilities. In *ACM HotNets II*, 2003.

[3] P. Barford, J. Kline, D. Plonka, and A. Ron. A Signal Analysis of Network Traffi c Anomalies. In *ACM SIGCOMM IMW*, Nov. 2002.

[4] S. Bellovin, M. Leech, and T. Taylor. ICMP traceback messages. Internet draft, work in progress, 2001.

[5] H. Burch and B. Cheswick. Tracing Anonymous Packets to Their Approximate Source. In *USENIX LISA*, 2000.

[6] Z. Chen, L. Gao, and K. Kwiat. Modeling the Spread of Active Worms. In *IEEE INFOCOM*, 2003.

[7] D. L. Donoho, A. G. Flesia, U. Shankar, V. Paxson, J. Coit, and S. Staniford. Multiscale Stepping-Stone Detection: Detecting Pairs of Jittered Interactive Streams by Exploiting Maximum Tolerable Delay . In *RAID*, 2002.

[8] M. Faloutsos, P. Faloutsos, and C. Faloutsos. On Power-Law Relationships of the Internet Topology. In *ACM SIGCOMM*, 1999.

[9] A. Hussain, J. Heidemann, and C. Papadopoulos. A Framework for Classifying Denial of Service Attacks. In *ACM SIGCOMM*, 2003.

[10] H. Kim and B. Karp. Autograph: Toward Automated, Distributed Worm Signature Detection. In *USENIX Security Symposium*, 2004.

[11] C. Kreibich and J. Crowcroft. Honeycomb – Creating Intrusion Detection Signatures Using Honeypots. In *ACM HotNets II*, 2003.

[12] L. Lamport. Time, Clocks, and the Ordering of Events in a Distributed System . *Communications of the ACM*, 21:558–565, July 1978.

[13] J. Mirkovic, G. Prier, and P. Reiher. Attacking DDoS at the Source. In *ICNP*, Nov. 2002.

[14] D. Moore, C. Shannon, G. Voelker, and S. Savage. Internet Quarantine: Requirements for Containing Self-Propagating Code. In *IEEE INFOCOM*, Apr. 2003.

[15] D. Moore, G. M. Voelker, and S. Savage. Inferring Internet Denial-of-Service activity. In *USENIX Security Symposium*, pages 9–22, Aug. 2001.

[16] K. Park and H. Lee. On the Effectiveness of Route-Based Packet Filtering for Distributed DoS Attack Prevention in Power-Law Internets . In *ACM SIGCOMM*, 2001.

[17] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time . In *7th USENIX Security Symposium*, Jan. 1998.

[18] M. Roesch. Snort - Lightweight Intrusion Detection for Networks. In *USENIX LISA*, 99.

[19] University of Oregon Route Views Project. `http://www.routeviews.org`.

[20] S. Savage, D. Wetherall, A. Karlin, and T. Anderson. Practical Network Support for IP Traceback. In *ACM SIGCOMM*, Aug. 2000.

[21] A. C. Snoeren, C. Partridge, L. A. Sanchez, C. E. Jones, F. Tchakountio, S. T. Kent, and W. T. Strayer. Hash-Based IP Traceback . In *ACM SIGCOMM*, Aug. 2001.

[22] D. Song, D.Wagner, and A.Perrig. Practical Solutions for Search on Encrypted Data. In *IEEE Symposium on Security and Privacy*, May 2000.

[23] S. Staniford, V. Paxon, and N. Weaver. How to Own the Internet in Your Spare Time. In *11th USENIX Security Symposium*, 2002.

[24] S. Staniford-Chen, S. Cheung, R. Crawford, M. Dilger, J. Frank, J. Hoagland, K. Levitt, C. Wee, R. Yip, and D. Zerkle. Grids: A graph-based intrusion detection system for large networks. In *National Information Systems Security Conference*, 1996.

[25] J. Wu, S. Vangala, L. Gao, and K. Kwiat. An Effective Architecture and Algorithm for Detecting Worms with Various Scan Techniques. In *NDSS*, 2004.

[26] Y. Zhang and V. Paxson. Detecting Stepping Stones. In *9th Usenix Security Symposium*, 2001.