Tracking the Role of Adversaries in Measuring Unwanted Traffic

Mark Allman
ICSI

Paul Barford
University of Wisconsin

Balachander Krishnamurthy & Jia Wang
AT&T Labs–Research

Abstract

Measurements related to security are being carried out on many sites on the Internet at network ingress points, between specific points on the Internet, and across the wide area Internet. The goals range from identifying sources of and possibly filtering unwanted traffic, to characterizing and coming up with new mechanisms for deterring attacks. Most of the measurements do not *systematically* consider adversarial traffic aimed at their measurement system. We explore the role adversaries can play and present a taxonomy on the potential impact of unwanted traffic on measurement systems. Our goal is to both enhance the robustness of such systems and spur development of tools that can alter the playing field by increasing the cost to adversaries.

1 Introduction

Attacks on the Internet are steadily rising and range from identity spoofing, spam, phishing, flooding links, distributed denial of service attacks, worms, virus, to organized targeted attacks by botnets. Various interesting and pertinent solutions have been proposed to reduce the impact of this unwanted traffic—authenticating senders, identifying malware, filtering, throttling, rate limiting, sharing information, etc. Research on unwanted traffic has examined attacks traditionally from the viewpoint of its impact on the user, the end systems, and the network. In addition, numerous measurement projects have characterized malware, examined the breadth of their impact, and helped in the search for solutions.

What has been missing is a *systematic examination* of the specific impact of *adversarial* traffic on the measurement systems that have been created to deal with the malware. Systems to mitigate security threats have been proposed that often also take into account an active adversary trying to evade or otherwise attack them. However, the adversary is considered to be narrowly focused on defeating the mechanism under study. We sketch a framework for a more systematic evaluation that considers the impact an adversary can have on the measurements, and lays the groundwork for considering how these effects can propagate across different types of measurement.

Interference has been anecdotally discussed in certain contexts, such as Web performance measurement systems like Keynote [4] and eValid Test Suite [2]. These systems use a globally distributed set of platforms to monitor customer Web sites by sending periodic requests to selected Web pages to ensure availability and simultaneously measuring latency. Since the location of the monitoring clients is often known, it is not surprising that some Web sites game the system by providing a different (higher) quality of service to requests from such monitoring clients to be perceived as a better provisioned site. This would differ from actual user experience of the site.

Such interference is relatively benign and has impact on competitive businesses. If the systems instead dealt with security related measurements, an active adversary has a strong incentive to use techniques to bypass the system or compromise it. Even if the security infrastructure is not entirely compromised, inferences made without taking into account active adversaries could be flawed. The flaws could be from the perspective of false negatives, false positives, and overall effectiveness.

In this paper we examine the ways in which measurements and thus inferences can be skewed as a result of adversarial actions. Our goal is to offer a framework for which researchers and practitioners can evaluate the impact of adversaries on their measurements from a variety of angles. We first illustrate the adversarial nature of measuring unwanted traffic in § 2. We then present a taxonomy of the impact this traffic has on the measurement systems in § 3. Finally, we conclude with a look towards future work.

2 Measuring Adversaries

In this paper we focus our attention on measurement for security-related tasks. Unlike standard measurement tasks, this type of measurement explicitly involves an *adversary*. That is, network operators are using the measurements to either enforce some given policy on the traffic (*e.g.,* with a firewall or intrusion prevention system) or to better understand the behavior of some remote entity with the idea of informing future detection and response

mechanisms (*e.g.,* with Honeypots). On the other hand, attackers sometimes try quite hard to not be noticed and to circumvent efforts at being measured and characterized. This tension is fundamental to all security-related measurement activities. In this section, we first survey the types of measurement systems used in the Internet today for security-related measurement. We then categorize the types of unwanted traffic that these systems are likely to encounter and the possible impact of this unwanted traffic.

2.1 Measurement Systems

Many schemes have been developed and used to measure and characterize unwanted traffic. We focus on four broad categories of popular measurement systems.

Firewalls: One of the most widely deployed mechanisms for detecting and controlling unwanted traffic is a firewall at the edge of an enterprise network. We consider "firewalls" to be a range of devices and activities. One end of the spectrum are simple Access Control Lists on routers that approve or deny traffic based on per-packet features (*e.g.,* IP addresses, transport layer port numbers, etc.). Firewalls can also be dedicated machines on the network path that keep per-flow state and support complex security policies such as capping the sending rate of a connection or the rate at which a machine can initiate connections. Firewalls also have a range of logging capabilities, from none to quite detailed.

NIDS: A second type of security-related measurement system is a Network Intrusion Detection Systems (NIDS), such as Snort [8] or Bro [6]. NIDS are deployed at strategic points to monitor incoming and outgoing traffic for all the hosts on a network. NIDS can produce both alarms for suspicious activities, as well as interface with firewalls to automatically stop ongoing attacks. NIDS use either signatures or algorithms to separate benign and malicious traffic. For instance, a well-known string (signature) within an HTTP GET request may indicate that an incoming packet is attempting to infect a local Web server with a particular worm. In another case, a NIDS may employ an algorithm to watch connection attempts over time to identify a remote host as a scanner ([6] shows one such algorithm). Another common technique for NIDS to use is to somehow characterize "normal" traffic and then flag deviations from that baseline (so-called "anomaly detection").

Honeypots: The third class of measurement systems we consider are network Honeypots [7, 13]. These systems either are real hosts or mimic real hosts attached to routed but otherwise unused address space. All traffic that arrives at these hosts is presumed to be either the result of a mis-configuration or malicious. Therefore, by actively responding to the queries, the Honeypots can be used to characterize unwanted traffic for the purposes of (i) warning operators (and other devices such as firewalls and NIDS) of previously unseen attacks and (ii) providing trends that help to improve operators situational awareness [12].

Application-Level Filters: These systems reside at the application layer and attempt to determine whether arriving traffic or requested traffic is "wanted" or not. The two most prevalent kinds of filters are for incoming email and for Web requests routed through a proxy server. Arriving email is often scanned for viruses and spam using a myriad of techniques from matching signatures to statistical techniques based on the prevalence of various terms in the email. Another example is enterprises using Web proxies that filter requests based on the content presumed to be associated with a given URL (*e.g.,* companies barring employees from accessing unsavory Web sites).

2.2 Adversarial Traffic

We next look at the range of traffic types that present challenges for security-related measurement systems. We break this down into three groups: (i) traffic that attempts to attack the measurement system or its resources directly, (ii) traffic that attempts to evade or circumvent the measurement systems, and (iii) traffic that intentionally avoids the measurement systems to prevent characterization.

2.2.1 Direct Attacks

Direct attacks on security-related measurement infrastructure can come in two forms.

First, attacks can simply try to DDoS the systems resources, be it bandwidth, memory, table entries, etc. As an example, some IDS systems such as Bro [6] maintain a large amount of state about various streams of traffic at various layers in the protocol stack simultaneously. One method of attacking such a system would be to attempt to make the system track more traffic streams than it could handle and therefore be able to sneak a malicious flow by the system without the system noticing (and, therefore, generating an alarm) [1]. Alternatively, many IDS systems attempt to buffer out-of-order packet arrivals to construct the exact byte stream an application is given. An attacker could attempt to use such a buffer against the measurement system and fill enough memory that new traffic would not be appropriately monitored. Another form of attack is to slowly ramp up "background noise" (legitimate Web fetches or similar benign traffic) in the victim's network to bring the measurement systems closer to the brink of failing to be able to keep up, with an attack then simply providing the proverbial straw that breaks the entire system.

A second type of direct attack on security-related measurement infrastructure is to compromise the measurement platform itself. For instance, the Witty worm [9] compromised hosts running various Internet Security Systems (ISS) products. Once a security-related measurement system has been compromised the alerts generated or actions taken are clearly questionable at best.

2.2.2 Evasion

We next focus on attacks that attempt to evade the measurement systems. We provide several examples of this sort of attack, but do not claim that this is a comprehensive list.

Among the simplest evasion techniques are splitting up payloads amongst multiple small packets (using IP fragmentation, multiple TCP packets, etc.). A simple measurement system that makes judgments on each arriving packet independently of all other packet arrivals will be easily fooled by splitting some string across packets. For instance, instead of sending "`root`" in one packet it gets broken into two packets with "`ro`" and "`ot`".

Another tactic is to attempt to circumvent a firewall that does not pass traffic from some application by using a non-standard port for the application in question, but one that the firewall does allow. For instance, using TCP port 80 for *ssh* traffic rather than for Web traffic. This can be exploited by benign users that are simply trying to get work done or by adversaries who are trying to hide their traffic from security monitors.

Evasion tactics get trickier. For instance, an attacker may leverage the IP TTL to show a NIDS a superset of the packets that will actually arrive at the end host, thus giving the NIDS a potentially bogus view of what might be happening on the end system [3].

Another trick (from [3]) is to reorder and retransmit segments to try to confuse a NIDS system. For instance, if "`root`" is to be transmitted the attacker might send "`m`" with sequence number 4, "`oo`" with sequence numbers 2 and 3, a "`t`" with sequence number 4 and finally an "`r`" with sequence number 1. This leaves the NIDS with an inconsistent view. Did the attacker send "`root`" or "`room`"? The TCP specification is ambiguous and actual behavior varies. Therefore, the NIDS may get fooled into thinking something benign was transmitted instead of something malicious, or visa-versa.

NIDS systems that use anomaly detection are susceptible to attacks whereby the adversary attempts to re-define the notion of "normal" such that it is easier for an attack to fly "under the radar". A similar situation exists for Honeypots which commonly use statistical methods to interpret data.

Spam filters may represent the single system that attackers most often attempt to circumvent. Basic systems that simply match strings are dealt with by changing the case of text, mis-spelling words, breaking lines at different points, etc. More advanced statistical techniques for finding spam are gamed by adding benign sounding words, mis-spelling words, using HTML formatting tricks to split words up in the actual email, but have those words put back together when rendered for the user, including the spam message in a graphic instead of as text, etc. This is a year's old arms race without any signs of stopping, illustrating the difficulty of the task for measurement systems that attempt to characterize an adversary.

2.2.3 Avoidance Attacks

A final type of attack is an avoidance attack. The idea is that if an attacker can determine that it is interacting with a Honeypot rather than a "real" end system then the attacker may avoid the block of addresses the Honeypot is using. This kind of blacklisting can leave a void in the characterization of some specific malicious activities. This void then may propagate to other measurement systems. For example, IDS systems then will not have the signatures that a Honeypot may have been able to generate. This problem can be addressed by making the Honeypots as realistic as possible and by making the Honeypots monitor a constantly changing set of addresses to defeat blacklisting [5].

3 Modalities for Measurement Pollution by Unwanted Traffic

Our adversarial models imply that Internet measurements can be affected by unwanted traffic in a variety of ways. However, *adversarial intent* alone is insufficient to fully explain the scope of this issue. We posit that a complete description of how unwanted traffic effects measurements must include consideration of the *target measurement system*. For our purpose this means both the "sensor" component of the measurement system that is used to collect the raw data, and the "analysis engine" that is used to transform the raw data into the form that is presented to and considered by users.

For this paper, we consider four systems (explained in the previous section) used for the specific purpose of measuring unwanted traffic—firewalls, NIDS, Honeypots and application-level filters. Unwanted traffic can pollute the results reported by each of these systems in different ways.

We describe a taxonomy of the ways in which unwanted traffic from adversaries can change the measurements generated by firewalls, NIDS, and Honeypots based on two concepts: *consistency* and *isolation*. Both

of these concepts are defined in terms of a set of packets $P = p_1 \ldots p_n$ that arrive at the measurement system, and the resulting log entries available to users $A = a_1 \ldots a_m$. We define a measurement system to be consistent when a given set of packets P_i always results in the same set of log entries A_j. We define isolation in a measurement system if a given set of packets P_i results only in the set of log entries A_j. It is important to emphasize that the log entries A_j for each of the three target measurement systems are different. In the case of NIDS, log entries are alarms that include summary information from the rule that matched a set of packets and some detail on the packets themselves. Entries in firewall logs are similar. In contrast, log entries for Honeypots are often only packet traces that can include either headers alone or headers plus payloads. With these definitions, we enumerate the taxonomy as follows:

Consistent/Isolated This is the case where the measurement system is behaving correctly and log entries are not altered in unexpected ways by unwanted traffic. Specifically, given instances of unwanted traffic $p_i, \ldots, p_t$ generate log entries $a_i, \ldots, a_t$ and no other set of packets P_w will have an impact on the log entries. This ideal, predictable behavior is the baseline against which other cases must be compared.

Consistent/Non-isolated This is the case where the measurement system behaves in a consistent, predictable way, but where a given instance of unwanted traffic changes the log entries caused by other unwanted traffic. Specifically, consider that a given instance of unwanted traffic P_i results in log entry A_i. If another set of unwanted traffic P_j arrives along with P_i then the resulting alarm will change to A_k The following example illustrates a real world instance of this case.

Two sequences of packets were created using the MACE malicious workload generator [10] and submitted to a system running the Snort NIDS version 2.4.4 with rules public release 2.4. Sequence S_1 triggers a pair of alarms (shown in Table 1) because of the string "/hsx.cgi" in the URI (first alarm) and the string "../../" (second alarm) in the payload (either after "?" for a GET request or in the body of a POST). Sequence S_1 consists of three packets: the first for the HTTP request up to the end of the "/hsx.cgi" string and the second and third packets consisting of the payload "../". The second sequence, S_2, differs from the first in that an additional packet was inserted between the second and third packets consisting of "`\x08/`", which is equivalent to a backspace followed by a forward slash. When Snort observes sequence S_2 only the first alarm in Figure 1 is fired, even though it is semantically equivalent to S_1. This example is similar in most respects to common insertion exploits; the difference being that in this case the resulting log entries are changed, not omitted. This problem is similar to the issue of normalization [3], but at a payload semantic level.

Inconsistent/Isolated This is the case where the measurement system generates two (or more) different sets of alarms A_{i1} and A_{i2} when the same set of packets P_i arrives at the system in two separate measurement epochs. The difference between this case and the last is that the sets of alarms A_{iX} are not predictable, and that the alarms generated by other sets of packets P_j are not affected by this behavior. This case can arise when certain kinds of randomness are introduced into the environment that lead to unpredictable behavior.

It is difficult to construct a concise examples for this case since inconsistency in the pure sense should not be bounded over time. However, we offer two examples of inconsistent/isolated behavior that illustrate the idea. The first example is an extension of the example in Table 1. It is likely that there are other signatures for attacks on a particular service that could be affected by the presence of a backspace character in a packet – but unlikely that all signatures in the database would be vulnerable. If a series of these backspace packets were directed toward a NIDS like Snort that considers traffic on a packet-by-packet basis, their impact would be unpredictable and would depend directly on whether the packets arrived in an order sufficient to alter the resulting log entries. This is inconsistent behavior. The fact that only signatures susceptible to the backspace could be affected means that the behavior is isolated.

A second example of this case relates to the idea of packet interleaving described in [11]. For NIDS that are connection-oriented such as Bro, signatures can be expressed in the form of state machines in which multiple packets from a source may be required to raise an alarm. In this case, two different signatures might have common initial sequences. If the same set of packets from a given service (such as NetBIOS) are observed by the NIDS in a different order between two measurement epochs, there is both the possibility that a different order of alarms will be logged and that a different set of alarms will be raised. These kinds of experiments were conducted in [11], and both of these outcomes were observed.

Inconsistent/Non-isolated This final case is where the measurement system's behavior in terms of the resulting logs is truly unpredictable from one epoch to the next. In this case, if a set of packets $p_i, \ldots, p_j$ result in log entries $a_i, \ldots, a_j$ in one measurement epoch, the same set of packets could result in log entries $a_k, \ldots, a_l$ in the next measurement epoch. This case is important because it illustrates that randomness in unwanted packet streams may not only be an issue of arrival order as in the case above, but can also be caused by the entirely different issue of denial of service. As discussed earlier, it is well known that many different measurement

Packet sequence S_1:	payload 1: `POST /hsx.cgi HTTP/1.0\r\nContent-length: 10\r\n\r\n` payload 2: `../` payload 3: `../` payload 4: `\x00\x00\x00\x00`
Packet sequence S_2:	payload 1: `POST /hsx.cgi HTTP/1.0\\r\\nContent-length: 10\r\n\r\n` payload 2: `../` payload 3: `\x08/` payload 4: `../` payload 5: `\x00\x00\x00\x00`
Snort rule #1:	uricontent:"/hsx.cgi"; (*i.e.,* if /hsx.cgi appears in the URI, raise an alarm)
Snort rule #2:	uricontent:"/hsx.cgi"; content:"../../"; content:"%00"; distance:1; (*i.e.,* if /hsx.cgi appears in the URI, plus content of ../.. followed by null byte in payload, raise an alarm)
Alarm #1:	[**] [1:1113:5] WEB-MISC http directory traversal [**] [Classification: Attempted Information Leak] [Priority: 2] 04/18-18:46:45.587011 10.2.23.103:25868 => 10.2.0.2:80 TCP TTL:64 TOS:0x0 ID:41059 IpLen:20 DgmLen:58 DF ***AP*** Seq: 0xECAE373E Ack: 0x43344649 Win: 0x8218 TcpLen: 32 TCP Options (3) => NOP NOP TS: 633549769 553538167 [Xref => http://www.whitehats.com/info/IDS297]
Alarm #2:	[**] [119:18:1] (http_inspect) WEBROOT DIRECTORY TRAVERSAL [**] [Classification: Attempted Information Leak] [Priority: 2] 04/18-18:46:45.587011 10.2.23.103:25868 => 10.2.0.2:80 TCP TTL:64 TOS:0x0 ID:41059 IpLen:20 DgmLen:58 DF ***AP*** Seq: 0xECAE373E Ack: 0x43344649 Win: 0x8218 TcpLen: 32 TCP Options (3) => NOP NOP TS: 633549769 553538167 [Xref => http://www.whitehats.com/info/IDS297]

Table 1: Snort generates both alarm#1 and #2 when it receives packet sequence S_1. However, Snort only generates alarm #1 when it receives packet sequence S_2 which includes a back space character in the fourth packet. This is an example of consistent, non-isolated behavior.

systems are susceptible to DoS attacks including NIDS and firewalls [10]. DoS attacks can cause unpredictable resource consumption in these systems leading to unpredictable packet loss and other effects that have the potential to alter what is ultimately logged by the measurement system.

4 Conclusions and Future Work

We have made an initial examination of how adversaries could impact security related measurements that are ongoing at various protocol layers in numerous sites across the Internet. We have sought to enumerate adversarial traffic types and to provide a taxonomy of the impact of this traffic on security related measurements. Our next steps in this study include conducting experiments with a spectrum of unwanted traffic and measurement systems to assess the actual impact in a controlled environment. Our ultimate goal is to develop generic tools and practices that can augment the ability of measurement systems to be robust against unwanted traffic.

Acknowledgments

Mark Allman's work was supported in part by the National Science Foundation under grants ITR/ANI-0205519, NSF-0433702 and STI-0334088. Paul Barford's work was supported in part by the National Science Foundation grants CNS-0347252, CCR-0325653. The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of the above government agencies or the U.S. Government.

References

[1] S. Crosby and D. Wallach. Denial of Service via Algorithmic Complexity Attacks. In *USENIX Security Symposium*, 2003.

[2] eValid WebSite Analysis and Testing Suite. http://www.soft.com/eValid/.

[3] M. Handley, C. Kreibich, and V. Paxson. Network Intrusion Detection: Evasion, Traffic Normalization, and End-to-End Protocol Semantics. In *Proc. of the 10th USENIX Security Symposium*, August 2001.

[4] Keynote. http://keynote.com.

[5] B. Krishnamurthy. Mohonk: Mobile Honeypots to Trace Unwanted Traffic Early. In *Proc. of the ACM SIGCOMM Network Troubleshooting Workshop*, 2004.

[6] V. Paxson. Bro: A System for Detecting Network Intruders in Real-Time. In *Proc. of the 7th USENIX Security Symposium*, January 1998.

[7] N. Provos. A Virtual Honeypot Framework. In *Proc. of the 13th USENIX Security Symposium*, August 2004.

[8] M. Roesch. Snort - Lighweight Intrusion Detection for Networks. In *Proc. of the 13th USENIX Systems Administration Conference*, November 1999.

[9] C. Shannon and D. Moore. The Spread of the Witty Worm. *IEEE Security and Privacy*, 2(4):46–50, Aug. 2004.

[10] J. Sommers, V. Yegneswaran, and P. Barford. A Framework for Malicious Workload Generation. In *Proc. of the ACM Internet Measurement Conference*, October 2004.

[11] J. Sommers, V. Yegneswaran, and P. Barford. Recent Advances in Network Intrusion Detection Systems Tuning. In *Proc. of the 40th IEEE Conference on Information Sciences and Systesm*, March 2006.

[12] V. Yegneswaran, P. Barford, and V. Paxson. Using Honeynets for Internet Situational Awareness. In *Proc. of the ACM/USENIX Fourth Workshop on Hot Topics in Networks*, November 2005.

[13] V. Yegneswaran, P. Barford, and D. Plonka. On the Design and Use of Internet Sinks for Network Abuse Monitoring. In *Proc. of the Symposium on Recent Advances in Intrusion Detection*, September 2004.