NISTIR 7007

An Overview of Issues in Testing Intrusion Detection Systems

Mell, P. M.

U. S. DEPARTMENT OF COMMERCE
Technology Administration
National Institute of Standards and Technology
Gaithersburg, MD 20899

July 11, 2003

U.S. DEPARTMENT OF COMMERCE
Donald L. Evans, Secretary

TECHNOLOGY ADMINISTRATION
Phillip J. Bond, Under Secretary for Technology

NATIONAL INSTITUTE OF STANDARDS AND TECHNOLOGY
Arden L. Bement, Jr., Director

An Overview of Issues in Testing Intrusion Detection Systems[1]

Authors: National Institute of Standards and Technology ITL: Peter Mell, Vincent Hu, Massachusetts Institute of Technology Lincoln Laboratory: Richard Lippmann, Josh Haines, Marc Zissman

Abstract

While intrusion detection systems are becoming ubiquitous defenses in today's networks, currently we have no comprehensive and scientifically rigorous methodology to test the effectiveness of these systems. This paper explores the types of performance measurements that are desired and that have been used in the past. We review many past evaluations that have been designed to assess these metrics. We also discuss the hurdles that have blocked successful measurements in this area and present suggestions for research directed toward improving our measurement capabilities.

Keywords: intrusion, detection, evaluation, detection, false alarm, false positive, vulnerability, attack, ROC, stealthy, traffic

1.0 Introduction

Within the last four years, the use of commercial intrusion detection system (IDS) technology has grown considerably, and IDSs are now standard equipment for large networks. According to International Data Corp., the projected revenue from IDSs and related services is expected to rise to $443.5 million in the year 2002 from an estimated $350 million in IDS revenue realized in 2001 [1]. Despite this enormous investment in IDS technology, no comprehensive and scientifically rigorous methodology is available today to test the effectiveness of these systems. Some might blame consumers for not demanding such effectiveness measures before purchasing IDSs. Some might also blame research-funding entities for not investing more money in this area. DARPA, which does fund research in measurement methodologies, is the exception. However, quantitative IDS performance measurements are not available because there are research hurdles that must be overcome before we can create such tests. This paper outlines the quantitative measurements that we need, the obstacles that are impeding progress to developing these measurements, and our ideas for research in IDS performance measurement methodology to overcome those obstacles.

This work was sponsored by the Defense Advanced Research Projects Agency under Air Force Contract F19628-00-C0002. The identification of any commercial product or trade name does not imply endorsement or recommendation by the National Institute of Standards and Technology.

2.0 Motivations for Quantitative Evaluations

There are many potential customers for the results of quantitative evaluations of IDS accuracy. Acquisition managers need such information to improve the process of system selection, which is too often based only on the claims of the vendors and limited-scope reviews in trade magazines. Security analysts who review the output of IDSs would like to know the likelihood that alerts will result when particular kinds of attacks are initiated. Finally, R&D program managers need to understand the strengths and weaknesses of currently available systems so that they can effectively focus research efforts on improving systems, and measure their progress.

3.0 Quantitatively Measurable IDS Characteristics

In this section we list a partial set of measurements that can be made on IDSs. We focus specifically upon those measurements that are quantitative and that relate to detection accuracy.

3.1 Coverage

This measurement determines which attacks an IDS can detect under ideal conditions. For signature-based systems, this would simply consist of counting the number of signatures and mapping them to a standard naming scheme. For non-signature based systems, one would need to determine which attacks out of the set of all known attacks could be detected by a particular methodology. The number of dimensions that make up each attack makes this measurement difficult. Each attack has a particular goal (e.g. denial of service, penetration, or scanning), works against particular software running on particular versions of an operating systems or against a particular protocol, and leaves evidence or a trace in different locations. Attacks may also depend upon the hardware of the system that is attacked, on the version of a protocol used, or on the mode of operation used (e.g. stealthiness techniques). Researchers emphasize a variety of different features in their measurement studies including: the attack goal; the victim type' the data that must be collected to obtain evidence of the attack; whether the attack uses stealthy IDS evasion techniques; and combinations of these features. The result is that some researchers define attacks with coarse granularity (and acknowledge that each attack has multiple targets and modes of operation), while others define attacks at the finest level of granularity (where each attack has a very specific target configuration and mode of operation). This disparity concerning the proper level of granularity for viewing attacks makes it difficult to count the number of attacks that an IDS detects and to compare the coverage of multiple IDSs. This problem is somewhat alleviated by the Common Vulnerabilities and Exposures (CVE), which is a standard list of virtually all known vulnerabilities [2]. However, the CVE approach does not solve this problem when multiple attacks are used to exploit the same vulnerability using different approaches to evade IDS systems. To address this issue, the CVE standards group has started a project to name attacks, but this work is still in the research stages.

Another problem with assessing the coverage of attacks is determining the importance of different attack types. Different sites may assign widely varying costs and importance to detecting different types of attacks. Managers of E-commerce sites may not be interested in detecting and analyzing a scan or surveillance attacks that are used to determine hosts and other resources on a network. Often these attacks have no effect on their business and often can not be prevented. E-commerce managers, however may be very interested in detecting the onset of distributed denial of service (DDoS) attacks and in detecting successful host compromises or web defacements. This emphasis is differs from the approach of many commercial evaluations that include many probe and surveillance attacks. Military sites, however, may pay a great deal of attention to surveillance attacks in an attempt to determine precursors to more serious threats.

In addition, most sites are not able to detect failed attacks seeking vulnerabilities that no longer exist on a site. Attacks may fail because operating systems have been patched and are no longer vulnerable to specific attacks, because systems with the required operating system do not exist, or because a firewall or other protective device blocks important attack components. A particular site may monitor only a few out of the thousands of current known vulnerabilities listed in the CVE, and those few will change as systems are replaced, patched, and reconfigured and as new vulnerabilities are discovered. A comprehensive evaluation with hundreds of old attacks that exploit patched vulnerabilities may not be applicable to a site running well maintained web servers. A more focused evaluation including the five most recent web and DNS attacks may be more suitable.

3.2 Probability of False Alarms

This measurement determines the rate of false positives produced by an IDS in a given environment during a particular time frame. A false positive or false alarm is an alert caused by normal non-malicious background traffic. Some causes for Network IDS (NIDS) include weak signatures that: alert on all traffic to a high-numbered port used by a backdoor; search for the occurrence of a common word such as “help” in the first 100 bytes of SNMP or other TCP connections; or detect common violations of the TCP protocol. They can also be caused by normal network monitoring and maintenance traffic generated by network management tools. It is difficult to measure false alarms because an IDS may have a different false positive rate in each network environment and there is no such thing as a “standard” network. Also, it is difficult to determine aspects of network traffic or host activity that will cause false alarms. As a result, it may be difficult to guarantee that we can produce the same number and type of false alarms in an IDS test as are found in real networks. Lastly, IDSs can be configured and tuned in a variety of ways in order to reduce the false positive rate. This makes it difficult to determine which configuration of an IDS should be used for a particular false positive test.

A Receiver Operating Characteristic Curve is an aggregate of the probability of false alarms and the probability of detection measurements. We mention it because of its importance in the IDS testing community. This curve

summarizes the relationship between two of the most important IDS characteristics: false positive and detection probability. In Figure 1, we show a receiver operating characteristic (ROC) curve produced by two systems in an IDS test. The x axis shows the percentage of false alarms produced during a test and the y axis shows the percent of detected attacks at any given false alarm percentage. Note that an IDS can be operated at any given point on the curve. In this example, system 1 can be tuned to a variety of operating points while system 2 would be difficult to configure because it has only one realistic operating point. By definition, a receiver operating characteristic curve shows probabilities on the x and y axes, but sometimes the unit of measurement for normal traffic is difficult to define. As a result, researchers have used false alarms per unit time on the x axis. A unit of measurement is required to determine the maximum number of false alarms that could occur over a given time period. This unit of measurement depends critically on how features are extracted in an IDS from the input data used and is difficult to determine for commercial IDSs and other black-box systems. For evaluation purposes, it is probably better to plot detection rate versus false alarms per unit time. These curves are not truly ROC curves, but they convey information of importance when analyzing and comparing IDSs. In Figure 1, both systems were network-based IDSs, and the unit of measurement was the total number of packets transmitted over the network. The analysis assumed that, at worst, an IDS could issue one alert per packet, and the maximum number of alerts was the total number of packets transmitted.

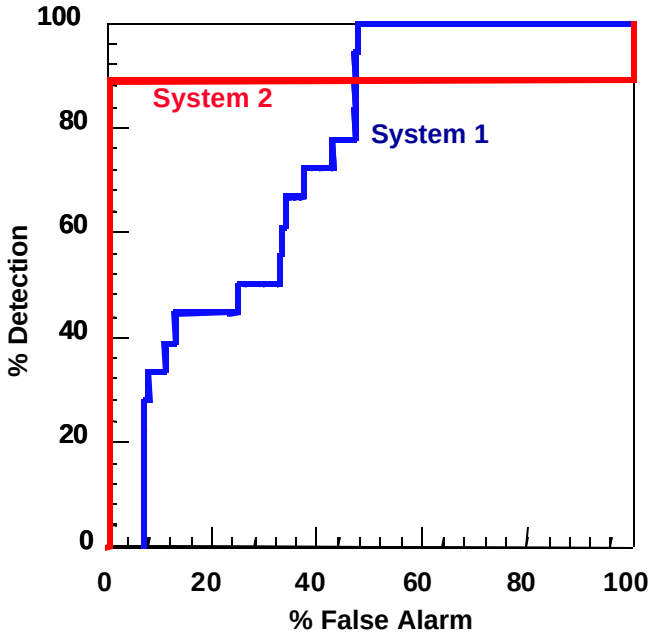

Figure 1. Receiver operating characteristic (ROC) curve plotting the percentage attacks detected versus the percentage of false alarms.

An interesting aspect of ROC curves is that there is an optimal operating point for an IDS, given a particular network being monitored. However, to determine it, one must know the cost of a false alarm, the value of a correct detection, and the prior probabilities of normal and attack traffic.

3.3 Probability of Detection

This measurement determines the rate of attacks detected correctly by an IDS in a given environment during a particular time frame. The difficulty in measuring the detection rate is that the success of an IDS is largely dependent upon the set of attacks used during the test. Also, the probability of detection varies with the false positive rate, and an IDS can be configured or tuned to favor either the ability to detect attacks or to minimize false positives (see section 3.2 for an explanation of this). One must be careful to use the same configuration during testing for false positives and hit rates.

Further, an NIDS can be evaded by stealthy versions of attacks. An NIDS may detect an attack when it is launched in a simple straightforward manner, but not when even simple approaches to stealthiness are used. Techniques used to make attacks stealthy include fragmenting packets, using various types of data encoding, using unusual TCP flags, encrypting attack packets, spreading attacks over multiple network sessions, and launching attacks from multiple sources [3, 4].

3.4 Resistance to Attacks Directed at the IDS

This measurement demonstrates how resistant an IDS is to an attacker's attempt to disrupt the correct operation of the IDS. Attacks against an IDS may take the form of:

1. Sending a large amount of non-attack traffic with volume exceeding the IDS's processing capability. With too much traffic to process, an IDS may drop packets and be unable to detect attacks.
2. Sending to the IDS non-attack packets that are specially crafted to trigger many signatures within the IDS, thereby overwhelming the IDS's human operator with false positives or crashing alert processing or display tools.
3. Sending to the IDS a large number of attack packets intended to distract the IDS's human operator while the attacker instigates a real attack hidden under the "smokescreen" created by the multitude of other attacks.
4. Sending to the IDS packets containing data that exploit a vulnerability within the IDS processing algorithms . Such attacks will only be successful if the IDS contains a known coding error that can be exploited by a clever attacker. Fortunately, very few IDSs have had known exploitable buffer overflows or other vulnerabilities.

3.5 Ability to Handle High Bandwidth Traffic

This measurement demonstrates how well an IDS will function when presented with a large volume of traffic. Most network-based IDSs will begin to drop packets as the traffic volume increases, thereby causing the IDS to miss a percentage of the attacks. At a certain threshold, most IDSs will stop detecting

any attacks. This measurement is almost identical to the "resistance to denial of service measurement" when the attacker sends a large amount of non-attack traffic to the IDS. The only difference is that this measurement calculates the ability of the IDS to handle particular volumes of normal background traffic.

3.6 Ability to Correlate Events

This measurement demonstrates how well an IDS correlates attack events. These events may be gathered from IDSs, routers, firewalls, application logs, or a wide variety of other devices. One of the primary goals of this correlation is to identify staged penetration attacks. Currently, IDSs have only limited capabilities in this area.

3.7 Ability to Detect Never Before Seen Attacks

This measurement demonstrates how well an IDS can detect attacks that have not occurred before. For commercial systems, it is generally not useful to take this measurement since their signature-based technology can only detect attacks that had occurred previously (with a few exceptions). However, research systems based on anomaly detection or specification-based approaches may be suitable for this type of measurement. Usually systems detecting attacks that had never been detected before produce more false positives than those that do not have this feature.

3.8 Ability to Identify an Attack

This measurement demonstrates how well an IDS can identify the attack that it has detected by labeling each attack with a common name or vulnerability name or by assigning the attack to a category.

3.9 Ability to Determine Attack Success

This measurement demonstrates if the IDS can determine the success of attacks from remote sites that give the attacker higher-level privileges on the attacked system. In current network environments, many remote privilege-gaining attacks (or probes) fail and do not damage the system attacked. Many IDSs, however, do not distinguish the failed from the successful attacks. For the same attack, some IDSs can detect the evidence of damages (whether the attack has succeeded) and some IDSs detect only the signature of attack actions (with no indication whether the attack succeeded or not). The ability to determine attack success is essential for the analysis of the attack correlation and the attack scenario; it also greatly simplifies an analyst's work by distinguishing between more important successful attacks and the usually less damaging failed attacks. Measuring this capability requires the information about failed attacks as well as successful attacks.

3.10 Capacity Verification for NIDS

The NIDS demands higher-level protocol awareness than other network devices such as switches and routers; it has the ability of inspection into the deeper level of network packets. Therefore, it is important to measure the ability of a NIDS to capture, process and perform at the same level of accuracy under a given network

load as it does on a quiescent network. For example, Hall [34] has proposed a test methodology and traffic metrics for standardized capacity benchmarking of NIDS. The NIDS customers can then use the standardized capacity test results for each metric and a profile of their networks to determine if the NIDS is even capable of sustaining inspection of the traffic.

3.11 Other Measurements

There are other measurements, such as ease of use, ease of maintenance, deployments issues, resource requirements, availability and quality of support etc. These measurements are not directly related to the IDS performance but may be more significant in many commercial situations.

4.0 Existing IDS Testing Efforts

IDS testing efforts vary significantly in their depth, scope, methodology, and focus. Table 1 summarizes the characteristics of some of the more developed efforts listed in chronological order. This table demonstrates that evaluations have increased in complexity over time to include more IDSs and more attack types, such as stealthy and denial of service (DoS) attacks. Only research evaluations have included novel attacks designed specifically for the evaluation, evaluated the performance of anomaly detection systems, and generated ROC curves. Evaluations of commercial systems have included measurements of performance under high traffic loads. Traffic loads were generated using real high-volume background traffic mirrored from a live network and also with commercial load-testing tools. The remainder of this section provides details concerning all the evaluations shown in Table 1, as well as for a few other more limited evaluations. Evaluations of research systems are described first, followed by descriptions of commercial system evaluations.

4.1 *University of California at Davis (UCD)*

Early research efforts at the University of California at Davis led to the first IDS testing platform that automatically launched attacks using interactive telnet, FTP, and rlogin sessions [5]. Scripts of normal and attack sessions were launched during tests that evaluated that ability of an IDS to distinguish intrusions from normal behavior and to operate when the computer running the IDS was under high load. An early network intrusion detection system called the Network Security Monitor (NSM) [6], which uses keyword matching to detect attacks in network traffic, was evaluated. The evaluation used a few attacks including password guessing, transmitting a password file to a remote host, and exploiting a vulnerability in the load module program to obtain super user status on a Unix machine. NSM missed some of these attacks until additional keywords were added. Under high CPU loads, it dropped packets and did not successfully detect attacks.

4.2 *IBM Zurich*

A similar IDS testing platform, not included in Table 1 (detail data not available), was developed at the IBM Zurich Research Laboratory to support IDS research [7]. Automated attacks and background traffic were generated as in [5] and used to improve IDS systems that were designed to detect attacks against FTP servers. Background traffic was generated only for FTP servers, and a small number of FTP attacks were scripted. It was noted that initial low detection and high false alarm rates were improved by cyclical repeatable testing and that generating realistic normal background traffic was complex and time-consuming. A focus of this effort was an attempt to completely automate the tasks required to run an evaluation.

4.3 *MIT Lincoln Laboratory (MIT/LL)*

MIT/LL has performed the most extensive quantitative IDS testing to date. Sponsored by the Department of Defense Advanced Research Projects Agency (DARPA), MIT/LL conducted large-scale testing of research IDSs in 1998 and 1999 [8, 9, 10]. The 1999 evaluation effort used a test bed, which generated live background traffic similar to that on an Air Force base containing hundreds of users on thousands of hosts. More than 200 instances of 58 attack types (including stealthy and novel attacks) were embedded in seven weeks of training data and two weeks of test data. Different from their 1998 evaluation, the 1999 evaluation was designed primarily to measure the ability of systems to detect new attacks without first training on instances of these attacks. Automated attacks were launched against three UNIX victim machines (SunOS, Solaris, Linux), Windows NT hosts, and a router in the presence of background traffic. Detection rates for attacks, false alarm rates for background traffic, and ROC curves were generated for more than 18 research IDS systems and attack categories including DoS, probe, remote-to-local, and user-to-super-user attacks. Attacks were counted as detected if an IDS produced an alert for the attacked machine that indicated traffic or actions on a victim host generated by the attack. An analysis of attack identification was also performed to determine if IDSs could provide the correct name for old attacks and attack details including the IP source address, ports used, and beginning and end of the attack. In addition, detailed analyses of detections and high-scoring false alarms were performed for a smaller number of high-performance systems to determine why specific attacks were missed. Results demonstrated that a combination of network and host-based approaches would have provided the best attack coverage. Also, novel attacks are difficult to detect because signatures do not generalize to new attacks and because network protocols or host audit logs were not analyzed sufficiently to extract attack evidence. Systems that detected old attacks could provide the correct names for these attacks and provide accurate information on the attack source, protocols, and ports used.

The MIT/LL evaluations resulted in the development of an intrusion detection corpus that includes weeks of background traffic and host audit logs, and hundreds of labeled and documented attacks. This corpus, used extensively by researchers, has been used as part of a data mining competition [11], and was

recently posted to a public web site [12]. This data has been used by Anzen Computing [13] to evaluate five commercial IDSs. Network tcpdump packet traces from the MIT/LL corpus were played back on a testbed in real time past commercial products, and truth markings were used to score output alerts. In addition, traffic for a few attacks was modified to determine whether the commercial systems were susceptible to the IDS evasion techniques described in [3]. The best commercial system detected only about half of the 43 attacks used, and three of the five commercial systems were not susceptible to IDS evasion techniques. Recent extensions of the MIT/LL evaluations led to a new corpus containing traffic produced by two complex distributed denial of service attacks and an evaluation testbed entitled LARIAT that automates most of the tasks required for real-time evaluations [14]. A review of the MIT/LL evaluations [15] suggests further extensions including more detailed analyses of reasons for misses and false alarms, development of calibrated sets of background traffic, inclusion of commercial products, and use of many more attacks.

4.4 *Air Force Research Laboratory (AFRL)*

AFRL, also under DARPA funding, participated with MIT/LL in the 1998 and 1999 evaluations of research IDS's. Their approach was similar to MIT/LL (they used some of MIT/LL's traffic and attack generation software) but focused upon testing IDSs in real-time in a more complex hierarchical network environment. IDS systems were installed in a testbed, four hours of background traffic were run, and attacks were launched against hosts in the midst of this background traffic. AFRL simulated a large network by developing software (also used by MIT/LL) to dynamically assign arbitrary source IP addresses to individual network sessions running on testbed computers. The 1998 test [16] evaluated three mature research signature-based IDSs and a government off-the-shelf (GOTS) baseline system similar to NSM [6]. Research systems had substantially lower false alarm rates than the GOTS system, but overall attack detection accuracy was still roughly only 25% at acceptable low false alarm rates. Further real-time evaluations were performed in 1999 on the same testbed, but no publications are currently available that describe this work. A summary briefing to DARPA noted that 1999 evaluations included commercial IDS systems, more complex background traffic, and a wider range of attacks and research systems.

4.5 *MITRE*

The MITRE Corporation [17] hosted the Intrusion Detection Fly-Off, one of the earliest evaluations of commercial and government-developed IDSs. This activity was an investigation into the characteristics and capabilities of network-based IDSs. Seven IDSs were tested using a two-phase approach. Phase I included relatively simple attacks using tools such as SATAN. This phase gave IDS operators an opportunity to familiarize themselves with the IDSs, and gave attackers an opportunity to practice attacking the systems. Phase II was designed to simulate a determined attack, involving both simple and more complex, stealthy attacks. Qualitative results were generated according to the following categories: real-time alerting capabilities; reporting capabilities; off-line analysis

capabilities; response capabilities; and remote management system. These analyses revealed weaknesses in the implementation and stability of some of the systems tested. Two of the commercial systems tested detected many more low-level attacks than the three government-developed systems, presumably because they had many more signatures for these attacks. The government-developed systems, however, were better suited to detecting and analyzing high-level attacks performed during interactive sessions. Attackers, given knowledge of the signatures used in the IDSs, were able to create stealthy attacks that were not detected.

4.6 *Neohapsis/Network-Computing*

Since 1999, Network Computing magazine has sponsored evaluation of commercial intrusion detection products by Neohapsis Laboratories[18-20]. The most recent review [20] included 13 commercial IDSs and the open-source Snort IDS [21]. These reviews assess performance under highly realistic traffic loads and have grown in complexity over the years. Qualitative results focus on practical characteristics including ease-of-use of the management framework, stability, cost effectiveness, signature quality/depth, and ease of customization. Quantitative results include the number of attacks detected [18-20] and sometimes the maximum traffic rates that can be handled before the IDS starts dropping packets, fails to perform signature checking, or fails to reassemble fragmented packets [19]. Table 1 shows characteristics of the most recent evaluation [20]. Realistic background traffic was created by mirroring traffic from DePaul University in Chicago onto an isolated testbed network. This traffic was from a backbone network with roughly 10,000 hosts and traffic rates of 30- to 88-M bits/sec and 5,000 to 7,000 packets per second. The evaluation showed that only seven of the IDSs tested could operate at these high traffic loads and that one crashed after only a few minutes. No careful analysis was made of false alarm rates. Nine recent attacks were launched against eight network IDSs. One detected all nine attacks, two detected only two attacks, and the other six detected from five to eight attacks. Each IDS was scored in the areas of management framework, signature quality/depth, stability of engine, cost-effectiveness, and customization. Detailed tables were also provided for each IDS, including features such as availability of a back-end database API, ability to reassemble fragmented packets and TCP streams, automatic signature update capabilities, availability of CVE cross references for signatures, and ability to log and display offending packets. A somewhat surprising result is that the open-source Snort IDS was rated third, an outcome that was better than 7 of the 9 commercial products tested.

4.7 *The NSS Group*

The NSS Group evaluated IDSs and vulnerability scanners in 2000 and 2001. The report of the 2001 IDS evaluation [22] includes 15 commercial IDS products and the open-source Snort IDS [21]. The bulk of this report presents detailed information for each IDS on its architecture, ease of installation and configuration, and the types of reporting and analysis provided. Twelve IDSs were compared using either 18 or 66 commonly available exploits including port

scans, DoS, Distributed DoS, Trojans, Web, FTP, SMTP, POP3, ICMP, and Finger attacks. Attacks were counted as detected only when they were reported "in as straightforward and clear a manner as possible." Attack detection rates were measured on a 100 M bits/s network with no traffic and with small (64 byte), real-world, and large (1514 byte) packets that consumed 0%, 25%, 50%, 75%, and 100% of the network bandwidth. In addition, IDS attack detection for fewer attacks (7 to 8) was measured using packet manipulation IDS evasion techniques described in [3] and HTTP syntax manipulation provided by a tool designed to hide web attacks called "whisker." Vulnerability to two tools named "stick" and "snot" designed to generate packets that elicit false alarms on IDSs was also determined. Attack detection rates are difficult to compare to other studies because correct detection required correct labeling and not simply detection. Detection rates with no traffic were high and above 80% for all but two IDSs. Detection rates fell off dramatically for some systems under high traffic loads, but remained high for others. Most systems were not vulnerable to the IDS evasion techniques tested and did not respond to tools designed to produce false alarms.

4.8 *Network World Fusion*

A more limited review of five commercial IDSs was reported by Network World Fusion magazine [23]. This review discussed ease of setup and use as well as features, but it also evaluated detection accuracy using 27 common attacks launched against three victim machines. These included stealthy web attacks generated using the "whisker" attack tool. Systems were tested using no background traffic and with artificially-generated loads of 40 M bits/sec (9,700 packets/sec), 60 M bits/sec (14,200 packets/sec), and 90 M bits/sec (67,000 packets/sec) on a 100 M bit/sec testbed network. Under no load, from roughly 78% to 93% of the 27 attacks were detected. Under the highest load of 90 M bits/sec, performance degraded substantially. Detection accuracy fell to 15% for one system and ranged from 63% to 78% for the other four.

	MITRE 1997	UC Davis 1997	MIT /LL 1998	MIT /LL 1999	AFRL 1998	Neo-hapsis 2001	NSS 2001	Network World 2001
Attacks								
Number of Attacks	>10	4	38	56	19	9	66	27
Number of victims	6	1	4	5	4	3	3	3
New/Novel Attacks			Ö	Ö				
Stealthy Attacks	Ö	Ö	Ö	Ö		Ö	Ö	Ö
DoS Attacks	Ö		Ö	Ö	Ö		Ö	Ö
Metrics								
Probability of False Alarms			Ö	Ö	Ö			
Probability of Detection			Ö	Ö	Ö	Ö	Ö	Ö

Receiver Operating Curve			Ö	Ö	Ö			
Perf. Under High Traffic						Ö	Ö	Ö
Background traffic								
Real						Ö		
Generate on Testbed		Ö	Ö	Ö	Ö	Ö	Ö	Ö
ID Systems								
Number	7	1	10	19	4	14	16	5
Commercial	Ö					Ö	Ö	Ö
Signature	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö
Anomaly			Ö	Ö				
Network	Ö	Ö	Ö	Ö	Ö	Ö	Ö	Ö
Host			Ö	Ö	Ö	Ö	Ö	

Table 1: Characteristics of IDS tests

5.0 Challenges of IDS Testing

There are several aspects of IDSs that make IDS testing challenging.

5.1 Difficulties in Collecting Attack Scripts and Victim Software

One problem that has inhibited progress in this field is the difficulty of collecting attack scripts and victim software. It is difficult and expensive to collect a large number of attack scripts. While such scripts are widely available on the Internet, it takes time to find relevant scripts to a particular testing environment. Once a script is identified, our experience is that it takes roughly one person-week to review the code, test the exploit, determine where the attack leaves evidence, automate the attack, and integrate it into a testing environment. As shown in Table 1, the number of attack types used in evaluations performed in 2001 ranges from 9 [20] to 66 [22], and many evaluations are heavily weighted towards scan and reconnaissance attacks due to the availability of these tools. A related problem is that it is difficult (but seemingly necessary) to collect appropriate victim software associated with each attack script. Often the various scripts only work against very specific version numbers of software that is difficult to obtain. The software itself may be difficult to obtain due to the expense or possibly the obscurity of the software. In addition, older vulnerable versions of the software may not be easily obtainable from the vendor. Vulnerable software is necessary for these evaluations and attack packets (or host audit logs) can't simply be fed into an IDS. Without information about the features used to detect attacks, the only way to determine whether an attack is detected is to successfully run the attack.

5.2 Differing Requirements for Testing Signature Based vs. Anomaly Based IDSs

Although most commercial IDSs are signature based, many research systems are anomaly-based, and it would be ideal if an IDS testing methodology would work for both of them. This is especially important since we would like to compare the performance of upcoming research systems to existing commercial ones. However, creating a single test to cover both types of systems presents some problems. First anomaly-based systems require normal traffic for training that does not include attacks. The MIT/LL 1999 evaluation [9,10], provided this type of training data specifically to train anomaly detection systems, but it was not available for other evaluations. Anomaly based-systems also may learn artifacts of the testing methodology and thereby perform well without actually detecting any real attacks. This may happen when all the attacks in a test are launched from a particular user, IP address, subnet, or MAC address. However, anomaly systems may learn subtle characteristics that are hard to predetermine such as packet window size, ports, typing speed, command set used, TCP flags, or connection durations which enable them to perform artificially well in the test environment. Conversely, testing signature based IDSs also presents some problems. Each signature based IDS detects a particular set of attacks. This means that the performance of a signature-based IDS in a test will, to a large degree, reflect the set of attacks used in the test. This presents a problem since researchers must decide which attacks to include, and this decision may arbitrarily favor one of the tested systems. To a lesser extent, anomaly systems can also have this problem since they will only process certain data streams when looking for attacks and they can't detect attacks in the unexamined data streams.

5.3 Differing Requirements for Testing Network Based vs. Host Based IDSs

Testing host based IDSs presents some difficulties not present when testing network based IDSs. In particular, network based IDSs can be tested in an off-line manner by creating a log file containing TCP traffic and then replaying that traffic to IDSs, as reported in [8-13]. This is convenient as all of the IDSs do not have to be tested at the same time, and the repeatability of the test is easy to achieve. Alternately, host based IDSs use a variety of system inputs in order to determine whether or not a system is under attack. This set of inputs changes between IDSs. Also, host based IDSs are designed to monitor a host as opposed to a single data feed (like network based IDSs). This makes it difficult to replay activity from log files in order to test a host based IDS. Since it is difficult to test a host based IDS in an off-line manner, researchers must explore more difficult real-time testing. Real-time testing presents problems of repeatability and consistency between runs.

5.4 Four Approaches to Using Background Traffic in IDS Tests

Most IDS testing approaches can be classified in one of four categories with regard to their use of background traffic: testing using no background traffic/logs, testing using real traffic/logs, testing using sanitized traffic/logs, and testing using simulated traffic/logs. While there may be other valid approaches, most

researchers find it necessary to choose among these categories when designing their experiments. Furthermore, it is not yet clear which approach is the most effective for testing IDSs since each has unique advantages and disadvantages.

Testing using no background traffic/logs

Many evaluations [19,22,23] test IDSs using no background traffic as a reference condition. In such experiments, an IDS is set up on a host or network on which there is no activity. Then, computer attacks are launched on this host or network to determine whether or not the IDS can detect the attacks. This technique can determine an IDS hit rate but can say nothing about false positives. This approach is useful for verifying that an IDS has signatures for a set of attacks and that the IDS can properly label each attack. Furthermore, testing schemes using this approach are often much less costly to implement than the ones that include, sanitize, or create background traffic or logs.

However, one drawback is that tests using this scheme are based on the implicit assumption that an IDSs ability to detect an attack is same regardless of background activity. At low levels of background activity it is not known whether or not this assumption is true (although we believe that it is). At high levels of background activity we know that IDSs performance often degrades (e.g. see [19, 22, 23] and thus the assumption may no longer be true.

Testing using real traffic/logs

Some researchers have tested IDSs by injecting attacks into a stream of real background activity [20]. This is a very effective approach for determining the hit rate of an IDS given a particular level of background activity. Hit rate tests using this technique may be well received because the background activity is real and it contains all of the anomalies and subtleties of background activity. Furthermore, this technique enables the comparison of IDS hit rates at differing levels of activity.

However, there are some drawbacks to using this technique:

1. It is usually not possible to have a repeatable test using real traffic since it is difficult politically and technically to store and replay large amounts of real traffic (especially in a backbone environment). Current hardware has trouble replaying network packets at speeds of over 100 Mb/sec and attempts to parallelize this process results in packet sequencing problems.
2. These experiments usually use a small set of victim machines that are set up for the sole purpose of being attacked during the test. Some IDSs may be able to detect that only these machines are attacked and thus artificially elevate their performance in the test. This is not a problem with most commercial systems but it does exist with many anomaly based research systems.
3. The real background activity used may contain anomalies unique to the network, which somehow favor one IDS over another. This could happen if a test network heavily used a particular protocol that was processed more deeply by a particular IDS. The IDS, which processes one protocol primarily, may

then miss other attacks that it should detect.

4. It is difficult to use this technique to determine false positive rates (and ROC curves). It is virtually impossible to guarantee the identification of all of the attacks that naturally occurred in the background activity, and this limitation hinders false positive rate testing. However, we hope that in practice it will become possible to use real traffic to determine false positive rates by manual analysis of the traffic logs, in addition to using statistical sampling techniques (to reduce the manual workload). Even with manual analysis, it will be impossible to guarantee that all attacks have been found, and some assumptions/approximations will have to be made.
5. It is difficult to publicly distribute the test since there are privacy concerns related to the use of real background activity.
6. Replay may damage the timings unless the replay honors the timestamps and the *tecdump* records (for TCP/IP NIDS) the timestamps to a granularity that is sufficiently fine.

Testing using sanitized traffic/logs

To overcome the political and privacy problems of using, analyzing, and/or distributing real background activity; some researchers have proposed using sanitized background activity. Examples of sanitized traffic are cleansed TCP packet headers used in studies [24]. These headers were recorded during normal traffic for use in modeling network statistics and not to evaluate intrusion detection systems. In this approach, real background activity is prerecorded and then sanitized to remove any sensitive data. Then, attack data is injected within the sanitized data stream. This can be accomplished either by replaying the sanitized data and running attacks concurrently or by separately creating attack data and then inserting this data into the sanitized data. The advantage of this approach is that the test data can be freely distributed and the test is repeatable.

However, some difficulties exist when using this approach:

1. Sanitization attempts may end up removing much of the content of the background activity thus creating a very unrealistic environment.
2. Sanitization attempts may fail causing an accidental release of sensitive data. This scenario is very possible since it is infeasible for a human to verify the sanitization of a large volume of data. We feel that this risk is one that most organizations will not tolerate for the sake of a research project.
3. Since attacks have to be injected somewhat artificially into the sanitized data stream (regardless of the method used), the attacks will not realistically interact with the background activity. For example, buffer overflow attacks may be launched against a web server and cause the server to crash, but normal background requests to the web server may continue. This lack of interaction between the attacks and the background activity could be a problem when testing IDSs.
4. When sanitizing real traffic, it may be difficult to remove attacks that existed in the data stream. If one is merely testing hit rates, then having unidentified attacks in the data is an inconvenience but not a major problem. However,

having unidentified attacks in the background activity would pose a problem for any false positive testing. Further, sanitizing data may remove information needed to detect attacks.

Testing by generating traffic on a testbed network

The most common approach to testing IDSs is to create a testbed network with hosts and network infrastructure that can be successfully attacked and to generate background traffic on this network [5, 7, 8-10, 12, 16, 19, 22, 23]. The testbed network includes victims of interest with background traffic generated by complex traffic generators that model the actual network traffic statistics [8-10]. Simpler commercial traffic generators can also be used to create a small number of packet types at a high rate [19, 22, 23]. Network traffic and host audit logs can be recorded in such a testbed for later playback [8-10,12] or evaluations can be performed in real time [16, 22, 23]. An advantage of this approach is that the data can be distributed freely since it does not contain any private or sensitive information. Another advantage is that we can guarantee that the background activity does not contain any unknown attacks since we created the background activity using the simulator. Lastly, IDS tests using simulated traffic are usually repeatable since one can either replay previously generated background activity or have the simulator regenerate the same background activity that was used in a previous test. This kind of test would appear to be one of the best since you can test both hit rates and false positive rates and use them to create ROC curves.

However, there are several difficulties to this approach:

1. It is very costly and difficult to create a simulation.
2. It may be difficult to simulate a high bandwidth environment due to resource constraints.
3. The need for different types of traffic to model various networks; for example, USAF traffic differs from traffic on academic networks.

6.0 IDS Testing Research Recommendations

This section presents a variety of research recommendations in the area of IDS testing. We first present recommendations for improving data sets and then present recommendations for enhancing metrics.

6.1 Shared Datasets

There is a great need for IDS testing datasets that can be shared openly between multiple organizations. There are very few such datasets that have even semi-realistic data or have the attacks within the background traffic labeled. Existing datasets [12, 29] have been used frequently by researchers. Without such shareable datasets IDS researchers must either expend enormous resources creating proprietary datasets or use fairly simplistic data for their testing.

6.2 Attack Traces

In section 5 we pointed out that it was difficult and expensive to collect a large set of attacks scripts for the purposes of IDS testing. A possible alternative is to use attack "traces" instead of real attacks. Attack traces are the log files that are produced when an attack is launched and that specify exactly what happened during the attack. Such traces usually consist of files containing network packets or systems logs that correspond to an instance of an attack. We need a better understanding of the advantages and disadvantages of replaying such traces as a part of an IDS test. In addition, there is a great need to provide the security community with a large set of attack traces. Such information could be easily added to and would greatly augment existing vulnerability databases such as [25] or [26]. The resulting vulnerability/attack trace databases would aid IDS testing researchers and would also provide valuable data for IDS developers.

6.3 Cleansing Real Data

Real data generally cannot be distributed due to privacy and sensitivity issues. Research into methods to remove the confidential data within background traffic while preserving the essential features of the traffic could enable the use of such data within IDS tests. Such an advance would alleviate the need for researchers to spend additional effort creating expensive simulated environments.

Another problem with real background data is that it may contain attacks about which we know nothing. It is possible, however, that such attacks could be automatically removed. One idea is to collect a trace of events in the real world and use a simulation system to produce data similar to those in the collected trace. The simulator would use its traffic generation routines to approximate the real world trace. After the creation and storage of the simulated activity, the original collected traces could be discarded. Thus, we have a system that can model real world environments with the model only restricted by the complexity of the simulator. Since the test data is created by the simulation system with only a model of the system being simulated, traffic in the trace that does not conform to the specified model (and that the simulator can thus not reproduce), such as attacks, would be filtered out by default. In addition, there should no longer be any concern about privacy issues since the traces that drive the simulator would contain only traffic summaries and not proprietary traffic content.

6.4 Sensor and Detector Alert Datasets

Some intrusion correlation systems do not use a raw data stream (like network or audit data) as input, but instead rely upon alerts and aggregated information reports from IDSs and other sensors [27, 28]. We need to develop systems that can generate realistic alert log files for testing correlation systems. A solution is to deploy real sensors and to “sanitize” the resulting alert stream by replacing IP addresses. Sanitization in general is difficult for network activity traces but it is relatively easy in this special case since alert streams use well defined formats and generally contain little sensitive data (the exception being IP addresses and

possibly passwords). Alternately, some statistical techniques for generating synthetic alert datasets from scratch are presented by [14].

6.5 Real-Life Performance Metrics

ROC curves are created by stepping through alerts emitted by the detector in order of confidence or severity. The goal is to show how many alerts must be analyzed to achieve a certain level of performance and, by applying costs, to determine an optimal point of operation. The confidence or severity-based ROC curve however is not a good indicator of how the intrusion detection system will perform with an intelligent human administrator sitting at the console. The human administrator does not consider the IDS alerts alone, but can make use of additional information such as network maps, user trouble reports, and learned knowledge of common false alarms when considering which alerts to analyze first. Thus the "alert ordering" used as a basis of the ROC is often not realistic. A further problem is that few current detection systems output a continuous range of scores but instead output only a few priorities (low/medium/high). Thus the ROC consists of only a few very coarse points.

It might be useful to use alert type, source and/or destination IP address along with severity or confidence to order a set of IDS alerts for the purpose of estimating cost and performance of a detector. Alerts could be ordered in "bins" either in the order that an administrator will examine them or the reverse (the order in which they will be "tuned" out and ignored). Then the ROC-like curve could be plotted by considering alerts in that order and plotting a point for each bin. One example would be to place alerts in bins by alert type and Internet-side IP address and to order them by the number of alerts in each bin. Often an administrator will dismiss very frequently occurring alerts as false alarms by analyzing only a few alerts of a particular type or source and then ignoring (or writing a rule to drop) all subsequent similar alerts. The ROC-like plot could be created to show the number or percentage of attacks that would be tuned out or missed as a result of ignoring each subsequent alert type, source, or combination. The curve could provide a much more realistic basis for comparing attack detection and false alarm performance and for estimating the cost of using the intrusion detection product at various levels of performance.

6.6 New Technologies

Newly evolving technologies in IDS include: "meta-IDS" technologies [30] that attempt to ease the burden of cross-vendor data management; "IDS appliances " [31] that promise increased processing power and more robust remote management capabilities; and "Application-layer" technologies [32] that filter potential attack traffic to downstream scanner on dedicated network segments. These "new directions" focus on new technologies for enterprises or service providers [33] and represent examples of research efforts to solve the difficulties of false positives, traffic bottlenecks and distinguishing serious attacks from nuisance alarms.

7.0 Conclusion

In this paper, we have described some of the previous efforts to measure IDS accuracy, and we have outlined some of the difficulties that have been encountered. We believe that a periodic, comprehensive evaluation of IDSs could be valuable for acquisition managers, security analysts and R&D program managers. However, because both normal and attack traffic are so variable from site to site, and because normal and attack traffic evolve over time, these evaluations will likely be complex and expensive to conduct. To enable evaluations to be made more efficiently, we recommend that the community find ways to create, label, share and update relevant data sets containing normal and attack activity.

8.0 References

[1] Howell D., *"hackers often choose their corporate targets"*, Investors Business Daily, January 30, 2002

[2]. Mann D. E. and Christey S. M., *Towards a Common Enumeration of Vulnerabilities*, 2nd Workshop on Research with Security Vulnerability Databases, Purdue University, West Lafayette, Indiana, January 21-22, 1999, http://www.cve.mitre.org/docs/cerias.html.

[3] Ptacek T.H. and Newsham T.N., *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*, 1998, Secure Networks Inc., http://secinf.net/info/ids/idspaper/idspaper.html.

[4] http://www.networkmagazine.com/article/NMG20000517S0075

[5] Puketza N., Chung M., Olsson R. O., and Mukherjee B., *A Software Platform for Testing Intrusion Detection Systems.* IEEE Software, 1997. 14,(5), 43-51, http://seclab.cs.ucdavis.edu/papers/pdfs/np-mc-97.pdf.

[6] Heberlein T.L., Dias G. V., Levitt K. N., Mukherjee B., Wood J., and Wolber D., *A Network Security Monitor*, in *Proceedings 1990 IEEE Symposium on Research in Security and Privacy*. 1990, IEEE Computer Society Press: Los Alamitos, CA. p. 296-304, http://seclab.cs.ucdavis.edu/papers/pdfs/th-gd-90.pdf.

[7] Debar H., Dacier M., Wespi A., and Lampart S., *A workbench for intrusion detection systems*, March 1998, IBM Zurich Research Laboratory, Ruschlikon, Switzerland.

[8]Lippmann R.P., Fried D.J., Graf I., Haines J.W., Kendall K.R., McClung D., Weber D., Webster S.E., Wyschogrod D., Cunningham R.K., and Zissman M.A., Evaluating Intrusion Detection Systems: The 1998 DARPA Off-Line Intrusion Detection Evaluation, in Proceedings of the 2000 DARPA Information Survivability Conference and Exposition (DISCEX), Vol. 2, 12-26, 2000, IEEE Computer Society Press: Los Alamitos, CA, http://www.ll.mit.edu/IST/pubs/discex2000-rpl-paper.pdf.

[9]Lippmann, R.P., Haines J.W., Fried D.J., Korba J., and Das K., The 1999 DARPA OffLine Intrusion Detection Evaluation. Computer Networks, 2000. 34(2), 579-595, http://www.ll.mit.edu/IST/ideval/pubs/2000/1999Eval-ComputerNetworks2000.pdf.

[10]Lippmann R.P. and Haines J., Analysis and Results of the 1999 DARPA Off-Line Intrusion Detection Evaluation, in Recent Advances in Intrusion Detection, Third International Workshop,

RAID 2000 Toulouse, France, October 204, 2000 Proceedings, H. Debar, L. Me, and S.F. Wu, Editors. 2000, Springer Verlag, 162-182, http://link.springer.de/link/service/series/0558/bibs/1907/19070162.htm.

[11] Elkan C., *Results of the KDD'99 Classifier Learning Contest*, September 1999, Sponsored by the International Conference on Knowledge Discovery in Databases, http://www-cse.ucsd.edu/users/elkan/clresults.html.

[12] MIT Lincoln Laboratory, DARPA Intrusion Detection Evaluation Data Sets, Jan. 2002, http://www.ll.mit.edu/IST/ideval/data/data_index.html

[13] Song D., Shaffer G., and Undy M., *Nidsbench - A network intrusion detection test suite*. 1999Recent Advances in Intrusion Detection, Second International Workshop, RAID 1999, West Lafayette, Indianahttp://citeseer.nj.nec.com/cache/papers/cs/19418/http:zSzzSzwww.monkey.orgzSz~dugsongzSztalkszSznidsbench-slides.pdf/song99nidsbench.pdf.

[14] Haines J.A., Rossey L.M., Lippmann R.P., and Cunningham R. K.. *Extending the DARPA Off-Line Intrusion Detection Evaluations* in *Darpa Information Survivability Conference and Exposition (DISCEX) II*. 2001. Anaheim, CA: IEEE Computer Society.

[15] McHugh J., *Testing intrusion detection systems: A critique of the 1998 and 1999 DARPA intrusion detection system evaluations as performed by Lincoln Laboratory*. ACM Transactions on Informationand System Security, 2000. **3,**(4), 262-294, http://www1.acm.org/pubs/citations/journals/tissec/2000-3-4/p262-mchugh/.

[16] Durst R., Champion T., Witten B., Miller E., and Spagnuolo L., *Testing and Evaluating Computer Intrusion Detection Systems.* Communications of the ACM, 1999. **42,**(7), 53-61, http://www1.acm.org/pubs/articles/journals/cacm/1999-42-7/p53-durst/p53-durst.pdf.

[17] Aguirre S.J. and Hill W.H., *Intrusion Detection Fly-Off: Implications for the United States Navy*, September 1997, MITRE Technical Report MTR 97W096, McLean, Virginia.

[18] Shipley G., *ISS RealSecure Pushes Past Newer IDS Players*. Network Computing, 17 May 1999, http://www.networkcomputing.com/1010/1010r1.html.

[19] Shipley G., *Intrusion Detection, Take Two*. Network Computing, 15 November 1999, http://www.nwc.com/1023/1023f1.html.

[20] Mueller P. and Shipley G., *Dragon claws its way to the top*. Network Computing, 20 August 2001, 45-67, http://www.networkcomputing.com/1217/1217f2.html.

[21] Roesch M., *Snort - Lightweight Intrusion Detection for Networks*, in *USENIX 13th Systems Administration Conference - LISA '99*. 1999: Seattle, Washington, http://www.usenix.org/publications/library/proceedings/lisa99/roesch.html.

[22] The NSS Group, *Intrusion Detection Systems Group Test (Edition 2)*. December 2001, Oakwood House, Wennington, Cambridgeshire, Englandhttp://www.nss.co.uk/ids/.

[23] Yocom B. and Brown K., *Intrusion battleground evolves*, Network World Fusion, October 8 2001, 53-62, http://www.nwfusion.com/reviews/2001/1008bg.html.

[24] National Laboratory for Applied Network Research, NLAR network traffic packet header traces, 2002, http://pma.nlanr.net/Traces/.

[25] Mell, P. and Grance T., ICAT Metabase CVE Vulnerability Search Engine, 2002, National Institute of Standards and Technology, http://icat.nist.gov/.http://icat.nist.gov.

[26] Security Focus Bugtraq Vulnerability Database, http://securityfocus.com.

[27] Vigna G., Kemmerer R.A., Blix P. "Designing a Web of Highly-Configurable Intrusion Detection Sensors", Recent Advances in Intrusion Detection (RAID 2001), Davis, CA, October 2001.

[28] Valdes A., and Skinner K. "Probabilistic Alert Correlation", Recent Advances in Intrusion Detection (RAID 2001), Davis, CA, October 2001.

[29] Forrest S., Ingham K., Inoue H,, Domayaji A., and Warrender C. "Sequence-based Intrusion Detection", Computer Immune Systems Data Sets and Software. Computer Science, The University of New Mexico, http://www.cs.unm.edu/%7Eimmsec/data-sets.htm.

[30]Loshin P., "Meta Detection", Information Security, page 52 TruSecure corporation, August 2001.

[31]Briney A., "The Age Of Appliances", Information Security, page 56 TruSecure corporation, August 2001.

[32]Kessler G. C., "IDS-In-Depth", Information Security, page 58 TruSecure corporation, August 2001.

[33]Briney A., "New Directions in Intrusion Detection", Information Security, page 49 TruSecure corporation, August 2001. www.infosecuritymag.com

[34] Hall M., and Wiley K., "Capacity Verification for High Speed Network Intrusion Detection Systems", Fifth International Symposium on Recent Advances in Intrusion Detection (RAID 2002), Zurich, Switzerland, October 16-18, 2002.