

# Defending Yourself: The Role of Intrusion Detection Systems

*Intrusion detection systems are an important component of defensive measures protecting computer systems and networks from abuse. This article considers the role of IDSs in an organization's overall defensive posture and provides guidelines for IDS deployment, operation, and maintenance.*

**John McHugh, Alan Christie, and Julia Allen**

*Software Engineering Institute, CERT Coordination Center*

**A**lthough intrusion detection technology is immature and should not be considered as a complete defense, we believe it can play a significant role in an overall security architecture. If an organization chooses to deploy an IDS, a range of commercial and public domain products are available that offer varying deployment costs and potential to be effective. Because any deployment will incur ongoing operation and maintenance costs, the organization should consider the full IDS life

cycle before making its choice. When an IDS is properly deployed, it can provide warnings indicating that a system is under attack, even if the system is not vulnerable to the specific attack. These warnings can help users alter their installation's defensive posture to increase resistance to attack. In addition, an IDS can serve to confirm secure configuration and operation of other security mechanisms such as firewalls.

After describing the role an IDS might play in an organization, we survey the most commonly used intrusion detection techniques and discuss representative systems from the commercial, public, and research arenas.

## **Intrusions and Intrusion Detection**

Intrusion detection has been an active field of research for about two decades, starting in 1980 with the publication of John Anderson's *Computer Security Threat Monitoring and Surveillance*,<sup>1</sup> which was one of the earliest

papers in the field. Dorothy Denning's seminal paper, "An Intrusion Detection Model,"<sup>2</sup> published in 1987, provided a methodological framework that inspired many researchers and laid the groundwork for commercial products such as those we discuss in this article. Still, despite substantial research and commercial investments, ID technology is immature and its effectiveness is limited.<sup>3</sup> Within its limitations, it is useful as one portion of a defensive posture, but should not be relied upon as a sole means of protection. Many recent media reports point to the need for comprehensive protection of which ID is a crucial part. For example,

Hackers attacked some of America's most popular Web sites yesterday for the third day in a row, walling off frustrated consumers from companies that provide news and stock trading as law enforcement officials launched a nationwide criminal in-

vestigation. ... the computer attacks earlier this week temporarily blocked access to Web sites that read like a Who's Who of the new economy, including Yahoo, eBay, Amazon, CNN.com, and Buy.com.—*Washington Post*, 10 Feb. 2000.

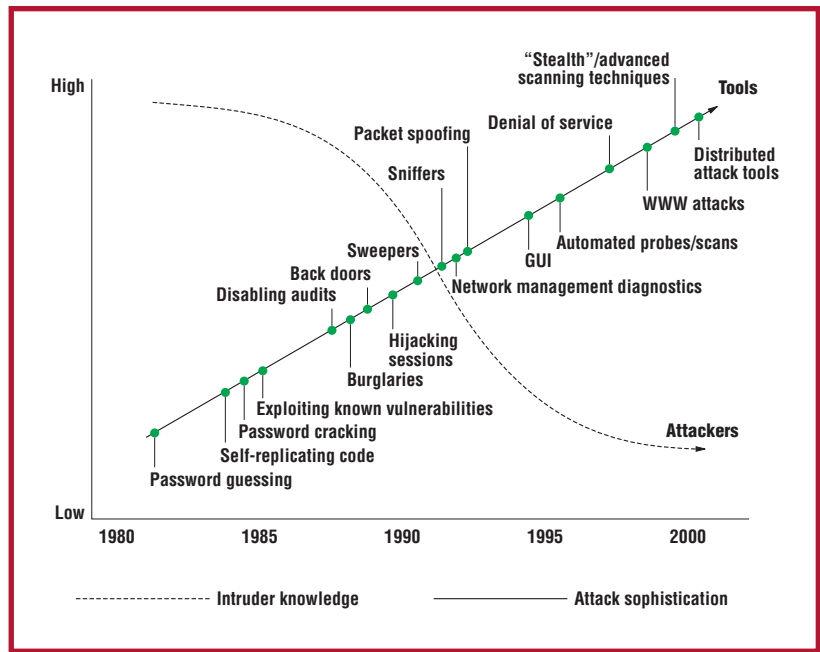
Attacks on the nation's computer infrastructures are a growing problem. The *Washington Post* story reflects the serious and sophisticated nature of recent cyber attacks. Over the past 12 years, the growth of incidents reported to our organization, the CERT Coordination Center, has roughly paralleled the Internet's growth. As e-commerce sites become attractive targets and the emphasis turns from break-ins to denials of service, the situation will likely worsen. Many early attackers simply wanted to prove that they could break into systems; increasingly nowadays, the trend is toward intrusions motivated by financial, political, and military objectives.

In the 1980s, most intruders were experts, with high levels of expertise and individually developed methods for breaking into systems. They rarely used automated tools and exploit scripts. Today, anyone can attack Internet sites using readily available intrusion tools and exploit scripts that capitalize on widely known vulnerabilities. Figure 1, taken from our earlier work ([www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html](http://www.sei.cmu.edu/publications/documents/99.reports/99tr028/99tr028abstract.html)),<sup>3</sup> which describes the attacks, illustrates the relationship between the relative sophistications of attacks and attackers from the 1980s to the present.

Today, damaging intrusions can occur in a matter of seconds. Intruders hide their presence by installing modified versions of system monitoring and administration commands and by erasing their tracks in audit and log files. In the 1980s and early 1990s, denial-of-service attacks were infrequent and not considered serious. Today, successful denial-of-service attacks can put e-commerce-based organizations such as online stockbrokers and retail sites out of business. Successful IDSs can recognize both intrusions and denial-of-service activities and invoke countermeasures against them in real time. To realize this potential, we'll need more accurate detection and reduced false-alarm rates.

### Perspectives on Intrusion: Victims and Attackers

Attacks can involve numerous attackers tar-



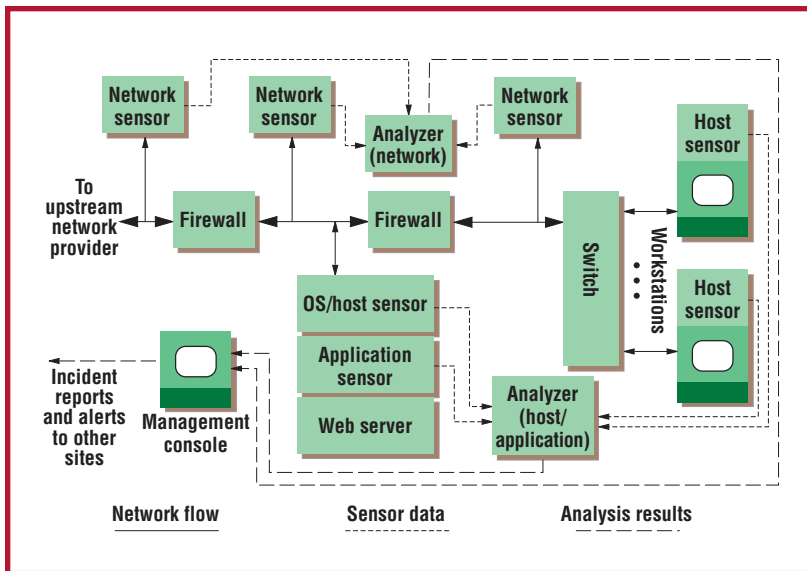
**Figure 1. Attack sophistication versus intruder technical knowledge.**

getting many victims. Defining what constitutes an attack is difficult because multiple perspectives are involved. The attacker viewpoint is typically characterized by intent and risk of exposure. From a victim's perspective, intrusions are characterized by their manifestations, which might or might not include damage. Some attacks produce no manifestations and some apparent manifestations are caused by system or network malfunctions. Some attacks involve the (involuntary) participation of additional machines, usually victims of earlier attacks. For an intrusion to occur, there must be both an overt act by an attacker and a manifestation, observable by the intended victim, that results from that act. We can create taxonomies that organize attacks from either viewpoint. Although a taxonomy based on manifestations would seem to provide a natural framework for discussing IDSs, the most comprehensive evaluation of these systems performed to date uses an attacker-centric taxonomy.<sup>4</sup> An attack victim's view of an attack usually focuses on the manifestations:

- What happened?
- Who is affected and how?
- Who is the intruder?
- Where and when did the intrusion originate?
- How and why did the intrusion happen?

The attacker might have a quite different view:

- What is my objective?
- What vulnerabilities exist in the target system?



**Figure 2. An IDS protected enterprise.**

- What damage or other consequences are likely?
- What exploit scripts or other attack tools are available?
- What is my risk of exposure?

ID's goal is to characterize attack manifestations to positively identify all true attacks without falsely identifying nonattacks. The motivation for using ID technology can vary. Some users might want to collect forensic information to locate and prosecute intruders. Others might use ID to trigger actions to protect computing resources. Still others might use ID as a diagnostic for identifying and correcting vulnerabilities.

### Dimensions of Intrusion Detection

We can characterize IDSs in a variety of ways. Here, we choose the system structure, sensed phenomenology, and detection approach.

Figure 2 illustrates system structure and sensed phenomenology. The figure shows a small enterprise configured with firewalls to isolate its Web server. Computers configured as network sensors extract suspicious packets from the three main network segments and forward them to a network-specific analysis station. The Web server and workstations run software to monitor suspicious interactions with the operating system and report them to a host-specific analysis station. In addition, the Web server looks for abuses such as CGI-bin exploits that are specific to HTTP servers. The analyzers report to a management console that serves as the IDS's user interface. The management console alerts the enterprise administration who might, in turn, report intrusions to in-

cident-response organizations such as the CERT Coordination Center.

More elaborate configurations are possible. An analyzer might use inputs from any or all sensed phenomenologies in deciding whether an attack has taken place. Analyzer outputs can also serve as sensed data for other analyzers.

### Intrusion Detection Approaches

We can view ID as an instance of the general signal-detection problem.<sup>5</sup> In this case, we view intrusion manifestations as the signal to be detected and consider manifestations of "normal" operations to be noise. In classical signal-detection approaches, both the signal and the noise distributions are known, and a decision process must determine whether a given observation belongs to the signal-plus-noise distribution or to the noise distribution. Classical signal detectors use knowledge of both distributions in making a decision, but intrusion detectors typically base their decisions either on signal (signature-based detectors) or noise (anomaly-based detectors) characterizations. Each approach has strengths and weaknesses. Both suffer from the difficulty of characterizing the distributions.

For a signature-based IDS to detect attacks, it must possess an attack description that can be matched to sensed attack manifestations. This can be as simple as a specific pattern that matches a portion of a network packet or as complex as a state machine or neural network description that maps multiple sensor outputs to abstract attack representations. If an appropriate abstraction can be found, signature-based systems can identify previously unseen attacks that are abstractly equivalent to known patterns. They are inherently unable to detect truly novel attacks and suffer from false alarms when signatures match both intrusive and nonintrusive sensor outputs. Signatures can be developed in a variety of ways, from hand translation of attack manifestations to automatic training or learning using labeled sensor data. Because a given signature is associated with a known attack abstraction, it is relatively easy for a signature-based detector to assign names (such as Smurf or Ping-of-Death) to attacks.

Anomaly-based detectors equate "unusual" or "abnormal" with intrusions. Given a complete characterization of the noise distribution, an anomaly-based detector recog-

nizes as an intrusion any observation that does not appear to be noise alone. Characterizing the noise distribution so as to support detection is nontrivial. Characterization approaches have ranged from statistical models of component or system behavior, to neural networks and other AI techniques, to approaches inspired by the human immune system. The primary strength of anomaly detection is its ability to recognize novel attacks. Its drawbacks include the necessity of training the system on noise with the attendant difficulties of tracking natural changes in the noise distribution. Changes can cause false alarms, while intrusive activities that appear to be normal can cause missed detections. Anomaly-based systems have difficulty classifying or naming attacks.

We can also classify IDSs based on the phenomenology that they sense. Network-based systems look at packets on a network segment, typically one serving an enterprise or a major portion of one. While network-based systems can simultaneously monitor numerous hosts, they can suffer from performance problems, especially with increasing network speeds. Many network-based systems make simplifying assumptions about such network pathologies as packet fragmentation<sup>6</sup> and can suffer from resource exhaustion problems when they must maintain attack-state information for many attacked hosts over a long period of time. In spite of these deficiencies, they are popular because they are easy to deploy and manage as stand-alone components and they have little or no impact on the protected system's performance.

Host-based systems operate on the protected host, inspecting audit or log data to detect intrusive activity. A variety of log and audit functions can serve to drive ID algorithms; these can be supplemented by sensors that monitor the interaction of applications with the host operating system. Host-based systems can monitor specific applications in ways that would be difficult or impossible in a network-based system. They can also detect intrusive activities that do not create externally observable behavior. Because they consume resources on the protected host, they can affect performance substantially. Successful intrusions that gain high levels of privilege might be able to disable host-based IDSs and remove traces of their operation. Intrusions that install Unix root kits are examples.

## Organizational Issues

Installing and effectively using IDSs on networks and hosts requires a broad understanding of computer security. Information technology infrastructures are becoming so complex that no one person can understand them, let alone administer them in a way that is operationally secure.

An organization must fully appreciate the commitment required before deploying an IDS. Otherwise, the project could well waste time, money, and staff resources in the IDS life cycle's initial phases. Although these issues are discussed in detail elsewhere,<sup>3</sup> we cover them briefly here to illustrate the problem's scope.

### Preparation

Before an organization invests in security technologies, it must understand which of its assets require protection and determine the real and perceived threats against those assets. We can characterize threats by the likely type of attack and attacker capabilities (that is, resources and goals) and the organization's tolerance for loss of, damage to, or disclosure of protected assets.

Attacker motives can be arbitrary (curiosity or vandalism) or targeted to meet a specific objective such as revenge or gaining a competitive advantage. Motives can make some forms of attack more likely than others. Gaining a competitive advantage might require compromising specific information such as a marketing plan. Each form of attack requires diverse detection strategies. For example, information retrieval is likely to occur during a stealthy attack, while information corruption might require speed. Determining whether the potential attacker is inside or outside the organization's infrastructure affects the type and placement of an IDS.

Often, the most significant obstacle to an information security improvement initiative is lack of management support. (One individual told us that he obtained management sponsorship by demonstrating how easy it was to break into his manager's confidential computer files. This approach is not necessarily recommended, but appears to have been effective in this case!) Surveys conducted by security trade magazines in 1999 cited lack of management support as one of the principle barriers to effective information security.<sup>7,8</sup> This is consistent with our

**Information technology infrastructures are becoming so complex that no one person can understand them, let alone administer them in a way that is operationally secure.**

**Hiring and retaining personnel to competently administer security in general and intrusion detection in particular are increasingly challenging.**

experience at the Software Engineering Institute in implementing security improvement initiatives. Managers have many goals to meet and must often make tradeoffs. Security only becomes important when it impinges on the organization's high-priority interests and reputation.

Deploying and operating an IDS requires significant management support at the level of the corporate chief information officer and information security manager. Without this support, this technology's successful operation and use will be short-lived, sustained only by the interest of those internal champions who believe in its benefit. That enthusiasm is likely to last only until another high-priority item requires their attention.

#### **Defense in Depth**

ID is only one aspect of a layered defensive posture or "defense in depth." Defense in depth begins with the establishment of appropriate and effective security policies. Effective policies help ensure that threats to critical assets are understood, managers and users are adequately trained, and actions to be taken when an intrusion is identified are defined. A good security policy puts ID in its proper perspective and context. Whenever possible, the policy should reflect the mission of the organization that promulgates it. Therefore, it should codify the rules governing enterprise operations as they are reflected in its information infrastructure and should explicitly exclude activities or operations not needed to support the enterprise's mission. A mission-oriented security policy can aid in configuring both firewalls and IDSs.<sup>9</sup>

Establishing a layered security architecture is advantageous whether an IDS is deployed or not. In addition to formulating a security policy, the essential steps consist of implementing user authentication and access controls, eliminating unnecessary services, applying patches to eliminate known vulnerabilities, deploying firewalls, using file integrity checking tools such as Tripwire, and so forth. Because most real-time commercial IDSs base their detection approach on known attempts to exploit known vulnerabilities, an administrator's time is often better spent minimizing vulnerability through the application of patches or other security measures. Detecting and responding to penetration attempts that cannot succeed (such as Unix-specific at-

tempts against a network of Windows machines) is not an effective use of resources, except as an indication of threat level.

Figure 2 illustrates some aspects of defense in depth. Using a network sensor outside the protected network lets the administrator sense the general threat level as indicated by probes and attempts that will be blocked by the outer firewall. Comparing the observations of sensors on both sides of the firewall lets the analyzer be configured to validate the firewall rules. The internal firewall provides an additional layer of defense for the inside workstations by excluding traffic that must reach the Web server from the outside but that should not reach the inside. In addition to helping to validate the inner firewall's rules, it also protects the inside should the Web server be compromised and used as a base to attack the inside.

If we assume that the protected enterprise is mission-oriented and only runs a limited set of applications and protocols, we can configure the inner sensor to recognize as intrusive any unexpected protocols. Host-based sensors on each workstation or server can look for both unexpected applications and abnormal behavior on the part of supported applications and the host operating system. When we combine the use of multiple firewalls and sensors configured to support a mission-specific security policy with a proactive vulnerability remediation policy, the removal of unneeded services, and the regular and careful use of integrity checking tools, the intruder's task becomes much more difficult.

#### **The IDS Life Cycle**

Vendors frequently release new IDS products and aggressively compete for market share. Evaluating these new systems is not a trivial task, and credible, comprehensive product evaluation information is lacking. Hiring and retaining personnel to competently administer security in general and intrusion detection in particular are increasingly challenging. Rapid changes in information technology make it difficult for an organization to implement an effective, long-term security strategy.

**Evaluation and selection.** If an organization plans to acquire an IDS, it should consider the resources available for the system's op-



eration and maintenance and choose one that meets its needs within these constraints. This is difficult because there are no industry standards against which to compare IDSs. The new product cycle for commercial IDSs is rapid, and information and systems quickly become obsolete. Steven Northcutt recommends the use of product guides that are updated at least monthly.<sup>10</sup> Relatively little objective third-party evaluation of IDSs is available, while trade press reports are generally spotty and superficial. Setting up a facility to objectively compare IDSs will be prohibitively expensive for all but the largest potential users, and some third-party or industry-sponsored effort is needed. A bit later, we discuss some of the technical issues involved in IDS evaluation.

Marketing literature rarely describes how well a given IDS finds intruders and how much work is required to use and maintain that system in a fully functioning network with significant daily traffic. IDS vendors usually specify which prototypical attacks their systems can find, but without access to deployment environments, they cannot describe how well their systems detect real attacks while avoiding false alarms.

Topics to consider include detection and response characteristics, use of signature- and anomaly-based approaches, diagnosis accuracy (false-alarm rate), ease of use, effectiveness of user interface, and quality of vendor support. Edward Amoroso and Richard Kwapniewski recently provided guidance in selecting an IDS,<sup>11</sup> and the Computer Security Institute ([www.gocsi.com](http://www.gocsi.com)) has a number of relevant Web pages, including a list of questions for IDS vendors.

**Deployment.** Deployment issues to address include placement of sensors to maximize protection for the most critical assets, configuring the IDS to reflect security policy, installing appropriate signatures and other initial conditions, establishing forensic procedures to preserve evidence for possible prosecutions, and determining when (if ever) and what automatic responses are allowed. Users must develop procedures for handling IDS alerts and consider how to correlate alerts with other information such as system or application logs. Integrating the IDS into a comprehensive system man-

agement framework would simplify this latter task. The Intrusion Detection Working Group of the Internet Engineering Task Force is developing a common alert format that will let IDS alerts from different systems be reported to a common display console. Several commercial IDS products will work within the Tivoli Enterprise management framework; we can expect additional developments of this sort in the future.

**Operation and use.** Once an organization deploys an IDS, it must monitor the system and respond to the alerts that it reports. This means establishing roles and responsibilities for analyzing and acting on alerts, monitoring the outcomes of both manual and automatic responses, and so forth.


IDSs themselves are logical targets for attack.<sup>6</sup> Smart intruders who realize that an IDS has been deployed on a network they are attacking will likely attack the IDS first, disabling it or forcing it to provide false information (distracting security personnel from the actual attack in progress). In addition, many commercial and research ID tools have security weaknesses resulting from flawed design assumptions. These can include failing to encrypt log files, omitting access control, and failing to perform integrity checks on IDS files.

**Maintenance.** Maintenance activities include installing new signatures as they become available, as well as installing periodic IDS upgrades. Sensor placement should be revisited periodically to ensure that system or network changes have not reduced the effectiveness of the IDS.

Use of technology alone is not sufficient to maintain network security. An organization must attract, train, and retain qualified technical staff to operate and maintain ID technologies. In today's market, qualified intrusion analysts and system/network administrators who are knowledgeable about and experienced in computer security are hard to find.

## Intrusion Detection Technology

Commercial ID technology is immature and dynamic to the point of instability. New vendors appear, only to be absorbed by others. Both commercial and research products evolve rapidly. One consequence of this rapid change is that product lists, surveys, and re-



**Smart intruders who realize that an IDS has been deployed on a network they are attacking will likely attack the IDS first, disabling it or forcing it to provide false information.**

A variety of commercial, research, and public domain ID tools are available. Here is a sampling.

### Commercial products

The commercial products described here represent two approaches. We have chosen one system that provides real-time performance based on a combination of network and host sensors using signatures. The other system provides post hoc detection based on anomalous file system changes.

Given today's volatile marketplace, it's best to use a Web search to locate current products, reviews, and so forth. Commercial product literature is generally weighted towards marketing, which often makes it difficult to determine the product's functionality and detection approach. Virtually no commercial literature addresses issues such as the frequencies of false alarms, missed detections, or the system's sensitivity to traffic loads.

**RealSecure.** RealSecure from Internet Security Systems ([www.iss.net](http://www.iss.net)) is a real-time IDS that uses a three-part architecture consisting of a network-based recognition engine, a host-based recognition engine, and an administrator's module. The network-recognition engine runs on dedicated workstations to provide network intrusion detection and response. Each network-recognition engine monitors a network segment looking for packets that match attack signatures. When a network-recognition engine detects intrusive activity, it can respond by terminating the connection, sending alerts, recording the session, reconfiguring firewalls, and so forth. It also passes an alarm to the administrator's module or a third-party management console.

The host-based engines analyze log data to recognize attacks. Each host engine examines its system's logs for evidence of intrusions and security breaches. Log data can contain information that is difficult or impossible to infer from network packet data. The host engine can prevent further incursions by terminating user processes or suspending user accounts. It can also take actions similar to those performed by a network engine.

An administrative module manages multiple-recognition engines. The result is comprehensive protection, easily configured and administered from a single location. The administrative module is supplied with both recognition engines and is also available as a plug-in module for a variety of network and systems management environments.

**Tripwire.** Tripwire is a file integrity assessment tool ([www.tripwire.com](http://www.tripwire.com)—both commercial and public domain versions are available) that is useful for detecting the effects of an intrusion. Trip-

wire creates a database of critical system file information that includes file lengths and cryptographic checksums based on each file's contents. Tripwire compares current information with a previously generated baseline and identifies changed files. Tripwire will report modified files, but the user must decide whether the modifications resulted from an intrusion. Because most monitored files are not expected to change except when new software versions are installed, changes usually indicate an unexpected or unauthorized activity.

For reliable Tripwire results, users must protect the database and program from tampering, either by maintaining them offline or online using read-only storage media, for example. Configuring Tripwire can be problematic, especially for large, multi-use systems because it is not easy to determine which files associated with some services and applications are expected to change and which are not.

### Public-domain tools

Shadow and Snort, two public-domain ID tools, are unlikely to have the same level of support as commercial systems, so users will need a higher level of technical expertise to install and manage them. The effort involved is likely to pay off with a better understanding of ID and its strengths and limitations.

**Shadow.** The Shadow ([www.nswc.navy.mil/ISSEC/CID](http://www.nswc.navy.mil/ISSEC/CID)) project is a joint venture of Naval Surface Weapons Center Dahlgren, Network Flight Recorder, the National Security Agency, and the SANS Institute.

Shadow uses both sensor and analysis stations. Sensors usually reside at key monitoring points in the network, such as outside a firewall, while the analysis station resides inside the firewall. The sensor is based on public domain packet-capture software and does not preprocess the data, thus preventing an intruder from determining the detection objectives by capturing an unprotected sensor. Sensors extract packet headers and save them to a file that the analysis station reads periodically. Major support comes from **tcpdump** (a Unix utility that logs network packets) packet filters supplemented by a Perl-based tool to detect slow intrusions that can span multiple log files. The analysis station uses a Web-based interface to display filtering results as well as raw data. Shadow runs on many Unix systems, including FreeBSD and Linux.

**Snort.** Snort is a recent open-source public-domain effort to build a lightweight, efficient, ID tool that can be deployed on a wide variety of Unix platforms. According to the Snort Web site ([www.snort.org](http://www.snort.org)),

views are quickly outdated. The "Technology" sidebar describes a sample of commercial, research, and public domain tools. Relatively little has been done in the area of evaluating IDS systems, but we present the results of several attempts in this area.

Both anecdotal evidence and the results from the few completed evaluations indicate

that current IDSs are not as effective as could be desired. Because no comprehensive evaluation of commercial products has been performed, it is difficult to obtain comparative figures. In 1999, IBM Zurich tested two commercial systems, RealSecure 3.0.x and NetRanger 2.1.2, using exploit scripts and tools available in their vulnerability database that

Snort is a lightweight network intrusion detection system, capable of performing real-time traffic analysis and packet logging on IP networks. It can perform protocol analysis, content searching/matching and can be used to detect a variety of attacks and probes, such as buffer overflows, stealth port scans, CGI attacks, SMB probes, OS fingerprinting attempts, and much more. Snort uses a flexible rules language to describe traffic that it should collect or pass, as well as a detection engine that utilizes a modular plugin architecture. Snort has a real-time alerting capability as well, incorporating alerting mechanisms for syslog, a user specified file, a UNIX socket, or WinPopup messages to Windows clients using Samba's smbclient.

Snort is currently undergoing rapid development. The user community is contributing auxiliary tools for analyzing and summarizing snort logs, providing additional capabilities. More importantly, there is a large group of users who contribute new signatures. As a result, new attacks are quickly represented in the signature database.

### Research prototypes

ID research performed in the early 1990s produced a number of new tools.<sup>1</sup> Although many were developed for academic purposes and were not released or maintained, these tools influenced research efforts and commercial systems. Early efforts focused on host-based solutions; later efforts concentrated on network-based systems. The two systems considered here have both host- and network-based aspects.

**Emerald.** SRI's pioneering ID work began in 1983 when a multivariate statistical algorithm was developed to characterize user behaviors.<sup>2</sup> Emerald (Event Monitoring Enabling Responses to Anomalous Live Disturbances: [www.sdl.sri.com/emerald/](http://www.sdl.sri.com/emerald/)) builds on this earlier work at SRI and uses both deviations from normal user behavior (anomalies), and known intrusion patterns (signatures).

A major goal of Emerald is to provide ID for large, loosely coupled enterprise networks. Such environments are more difficult to monitor and analyze due to the distributed nature of the incoming information. Emerald structures users into a federation of independently administered domains, each with its own network services and trust relationships with other domains. In this context, a centralized repository or analysis facility is likely to result in significant performance degradation. Emerald uses a hierarchical divide-and-conquer approach to address these issues.

The hierarchical approach provides three levels of analysis performed by a three-tiered system of monitors: service, domain, and enterprise monitors. These monitors have the same basic ar-

chitecture: a set of profiler engines (for anomaly detection), signature engines (for signature analysis), and a resolver component for integrating the results generated from the engines.

Emerald is an example of the direction that future ID systems might take. As intruders become more sophisticated in their attacks, they will be increasingly likely to disperse the evidence of their work across networks, making it difficult to sense when a distributed or coordinated attack is occurring. In such situations, the ability to collect, assimilate, correlate, and analyze information emanating from diverse sources in real time becomes essential.

**STAT.** The State Transition Analysis Technique ([www.cs.ucsb.edu/~kemm/netstat.html/](http://www.cs.ucsb.edu/~kemm/netstat.html/)) developed at the University of California Santa Barbara is a method for representing the sequence of actions that an attacker performs to achieve a security violation. The technique provides a general framework in which host-based (Ustat and Winstat), network-based (Netstat), and distributed multihost (Nstat) tools have been built.

State transition analysis uses a graphical notation to represent a penetration, precisely identifying its requirements and the nature of the compromise. Analysis tools use the information contained in a system's audit trail or network traffic to compare the state changes represented in the data to the state transition diagrams of known penetrations. State transition analysis assumes that

- penetrations require the attacker to possess some minimum prerequisite access to the target system (the initial state), and
- all penetrations lead to the acquisition of some previously unheld ability (the compromised state).

The signatures used in the STAT family are abstractions of intrusion scenarios. By using abstract representations of the actions that trigger state transitions, a single signature can represent an entire family of related penetrations including previously unseen variants. Recent work resulted in the development of Statl, a language for specifying intrusive actions. Tools are under development to generate detectors for members of the STAT family from Statl descriptions.

### References

1. B. Mukherjee, L.T. Heberlein, and K.N. Levitt, "Network Intrusion Detection," *IEEE Network*, Vol. 8, No. 3, May-June 1994, pp. 26-41.
2. D. Anderson et al., *Detecting Unusual Program Behavior Using the Statistical Component of the Next-Generation Intrusion Detection Expert System (NIDES)*, Tech. Report SRI-CSL-95-06, Computer Science Laboratory, SRI Int'l, Menlo Park, Calif., 1995.

were compatible with their test environment and for which the IDS claimed coverage in its documentation.<sup>12</sup> RealSecure detected 30 of 42 attacks, while NetRanger detected 18 of 32. Deployed in an operational setting, RealSecure issued some 8,000 alarms in a month, over half of which were due to a weekly scan of the network performed for maintenance

purposes. Comparable figures are not given for NetRanger, apparently because of performance problems. Both systems had fairly high false-alarm rates, but issued false alarms for different classes of activity.

The most comprehensive evaluations of IDS systems reported to date were the 1998<sup>4</sup> and 1999<sup>13</sup> offline evaluations performed



## About the Authors



**John McHugh** is a senior member of the technical staff in the Networked Systems Survivability Program at the Software Engineering Institute, where he performs research on information assurance and computer security. He has a PhD in computer science from the University of Texas, an MS in computer science from the University of Maryland, and a BS in physics from Duke University. He sits on the advisory board of the *International Journal of Information Security* and is the past chair of the IEEE CS Technical Committee on Security and Privacy. Contact him at Carnegie Mellon Univ., Software Eng. Inst., CERT Coordination Center, 4500 5th Ave., Pittsburgh, PA 15213-3890; jmchugh@cert.org.

**Alan Christie** is a senior member of the technical staff at the Software Engineering Institute where he is currently heading a project that aims to codify how incident response experts interpret, integrate, and abstract incident data. He is also active in supporting the development and promoting the use of the Easel simulator. He has an MS in computer science and a PhD in nuclear engineering from Carnegie Mellon University. In 1995, he published *Software Process Automation: The Technology and its Adoption* (Springer-Verlag, New York). Contact him at Carnegie Mellon Univ., Software Engineering Inst., CERT Coordination Center, 4500 5th Ave., Pittsburgh, PA 15213-3890; amc@sei.cmu.edu; <http://www.sei.cmu.edu/staff/amc/>.



**Julia Allen** is a senior member of the technical staff at the Software Engineering Institute, where she leads the identification and documentation of best current security practices for networked systems. She holds a BS in computer science from the University of Michigan, an MS in electrical engineering from the University of Southern California, and an executive business certificate from the University of California at Los Angeles. Her publications include a series of five security improvement modules, most notably *Preparing to Detect Signs of Intrusion and Responding to Intrusions*. Contact her at Carnegie Mellon Univ., Software Eng. Inst., CERT Coordination Center, 4500 5th Ave., Pittsburgh, PA 15213-3890; jha@sei.cmu.edu.

by MIT's Lincoln Laboratory. The systems evaluated are the results of research funded by DARPA. In these evaluations, investigators took sensor data in the form of sniffed network traffic, Solaris BSM audit data, Windows NT audit data (added in 1999), and file system snapshots and tried to identify the intrusions that had been carried out against a test network during the data-collection period. The test network consisted of a mix of real and simulated machines; background traffic (noise) was artificially generated by the real and simulated machines while the attacks were carried out against the real machines. Training data contained a variety of attacks that were identified in the corresponding documentation. The data used for evaluation contained

a mix of attacks that had been present in the training data and previously unseen attacks.

An analysis of the 1998 evaluation<sup>4</sup> shows a number of serious flaws, including failures to appropriately validate the background data used (especially with respect to its ability to cause false alarms), the lack of an appropriate unit of analysis for reporting false alarms, and the use of questionable or inappro-

appropriate data-analysis and presentation techniques.<sup>3</sup> The data rate represented by the test data is not given, but calculations based on the data set sizes indicate an average rate of a few tens of kilobits per second.

A total of 32 attack types were present in 1998. These were organized according to an attacker-centric taxonomy into four categories: denial of service, remote to local, user to root, and probing/surveillance, without regard to manifestation. The best system could only detect about 75% of the 120 attacks present in the evaluation data. Lincoln's 1998 report does not give the percentage of the attack types detected.<sup>4</sup> False alarms are reported "per day" rather than as a percentage of possible alarm cases. The best system generated two false alarms per day, while most systems produced some tens of false alarms per day.

These figures are problematic for several reasons. The data's noise component is responsible for the false alarms, but the report makes no comparison between real and artificial data with respect to false-alarm characteristics. Any given "natural" environment might produce more (or fewer) false alarms at the data rate used. If the data's false-alarm characteristics are proportional to the data rate, deploying the evaluated systems on networks carrying a few megabits per second of traffic could result in a hundredfold increase in the false-alarm rate. The 1999 evaluation produced similar, but slightly better, results for detection and false-alarm performance over a substantially broader base of attacks. The real improvement is one in breadth of coverage rather than in effectiveness.

Despite its shortcomings, the Lincoln evaluation indicates that even the best of the research IDS systems falls far short of the DARPA goals for detection and false-alarm performance. The Air Force Rome Lab has built a real-time test-bed<sup>15</sup> based on the system used by Lincoln to generate its offline evaluation data. They have used this system to evaluate a few of the research systems, with results similar to those obtained in the offline evaluation.

All of the evaluations performed to date indicate that IDSs are only moderately successful at identifying known intrusions and quite a bit worse at identifying those that have not been seen before. This renders automatic response to intrusions, a goal of both the research and commercial commu-

### Acknowledgments

We thank the members of the CERT staff who contributed to the report on which this article is based:<sup>3</sup> William Fithen, Jed Pickel, Ed Stoner, James Ellis, Eric Hayes, Jerome Marella, and Bradford Willke. Stefan Axelsson of Chalmers University provided insight into the characterization of ID as a signal-detection problem. Many members of the ID community have influenced our thinking on this topic. Among them are Ed Amoroso, Dick Kemmerer, Gene Kim, Jay Larrew, Teresa Lunt, Roy Maxion, Phil Porras, and Mark Woods.

The Software Engineering Institute is sponsored by the U.S. Department of Defense. CERT and CERT Coordination Center are registered servicemarks of Carnegie Mellon University.

nities, a dubious prospect. Blocking an attack by dynamically reconfiguring a firewall to block the intruding source runs the risk of a self-imposed denial-of-service attack if it is done in response to an event wrongly identified as intrusive. Anecdotal evidence suggests that legitimate network diagnostic activities have resulted in the temporary blockage of network traffic on at least one US military application system.

**I**n the fall of 1999, we briefly installed two commercial (ISS RealSecure and Cisco NetRanger) and two public-domain (Shadow and Network Flight Recorder) network-based IDSs in the CERT DMZ. We wanted to follow the vendor's instructions and observe each tool's performance. We did not intend to evaluate a specific tool or to compare one tool with another, nor did we want to measure detection and false-alarm performance.

We found that commercial ID tools were easier to install than public-domain tools. Although none had an understandable, easy-to-use configuration interface, the commercial tools did employ graphical interfaces, whereas the public domain tools did not. All of the tools required labor-intensive signature tuning. We found no indication of any integration between vulnerability scanners and configuration interfaces (ISS is integrating their IDS and vulnerability scanner) even though most IDS vendors sell vulnerability analyzers. The configuration process would be simpler if signatures associated with detected vulnerabilities could be loaded automatically.

The commercial products that we installed did not provide sufficient supporting data (such as raw packets) to verify events they claimed to detect. The use of proprietary algorithms and signatures made it difficult to determine why an alert occurs. Distinguishing between intrusions and false alarms required manual investigation. In most cases, the analyst had to examine log files for supporting evidence.

IDS products based on current signature-based analysis do not provide a complete ID solution but do produce useful results in specific situations and configurations. The majority of IDSs we examined appeared to provide good capabilities for enhanced network monitoring and might be more useful in this capacity than for intrusion detection.

ID research is proceeding along a number of fronts. Both the commercial and research communities are pursuing the development of new IDS algorithms. In the absence of fundamental breakthroughs, we anticipate only modest improvements in this area. Somewhat more promising are efforts to improve both detection and false alarm performance by combining or correlating the outputs of diverse sensors and obtaining information from multiple locations. This should prove effective under the assumption that a major cyber attack will target multiple installations simultaneously with similar manifestations. Nonetheless, much of the current effort seems to be aimed at detecting attacks that are made by relatively unskilled and unfocused attackers. We believe that a greater threat lies in narrowly focused attacks launched by adversaries who will make serious attempts to avoid detection. These attacks are likely to escape detection by current and proposed IDS. We are working to develop an analysis center that will combine IDS outputs with traditional open- and closed-source intelligence gathering, network traffic analysis, and other relevant information to provide a more complete picture of adversarial activity. ☉

## References

1. J.P. Anderson, *Computer Security Threat Monitoring and Surveillance*, tech. report, James P. Anderson Co., Fort Washington, Pa., 1980.
2. D.E. Denning, "An Intrusion Detection Model," *IEEE Trans. Software Eng.*, Vol. SE-13, No. 2, Feb. 1987, pp. 222-232.
3. J. Allen et al., *State of the Practice of Intrusion Detection Technologies*, Tech Report CMU/SEI-99-TR-028, Carnegie Mellon Univ., Software Engineering Inst., Pittsburgh, 2000.
4. R. Lippmann et al., "Evaluating Intrusion Detection Systems: The 1998 DAPA Offline Intrusion Detection Evaluation," *Discex 2000*, Vol. 2, IEEE Computer Soc. Press, Los Alamitos, Calif., 2000, pp. 12-26.
5. J.P. Egan, *Signal Detection Theory and ROC Analysis*, Academic Press, San Diego, 1975.
6. T.H. Ptacek and T.N. Newsham, *Insertion, Evasion, and Denial of Service: Eluding Network Intrusion Detection*, 1998; [www.aciri.org/vern/Ptacek-Newsham-Evasion-98.ps](http://www.aciri.org/vern/Ptacek-Newsham-Evasion-98.ps).
7. A. Briney, "Got Security?" *Information Security Magazine*, Vol 2, No. 7, July 1999, pp. 20-23.
8. A.K. Larson, "Global Security Survey: Virus Attack," *Information Week*, No. 743, July 12 1999, pp. 42-4, 48, 50, 52-3, 56.
9. C. Boeckman, "Getting Closer to Policy-Based Intrusion Detection," *Information Security Bulletin*, Vol. 5, No. 4, May 2000, pp. 13-22.
10. S. Northcutt, *Network Intrusion Detection*, New Riders, Indianapolis, 1999.
11. E. Amoroso and R. Kwapniewski, "A Selection Criteria for Intrusion Detection Systems," *Proc. 14th Ann. Computer Security Applications Conf.*, IEEE Computer Soc. Press, Los Alamitos, Calif., 1998, pp. 280-288.
12. H. Debar, "Testing Intrusion Detection Systems, Presentation to Groupe OSSIR," July 1999; [www.ossir.org/ftp/supports/99/debar/index1.html](http://www.ossir.org/ftp/supports/99/debar/index1.html).
13. R. Lippmann et al., "The 1999 DARPA Offline Intrusion Detection Evaluation," to be published in *RAID 2000*, LNCS, Springer-Verlag, New York, No. 1907.
14. J. McHugh, "The 1998 Lincoln Lab IDS Evaluation—A Critique," to be published in *RAID 2000*, LNCS, Springer-Verlag, New York, No. 1907, 2000.
15. R. Durst et al., "Testing and Evaluating Computer Intrusion Detection Systems," *Comm. ACM*, Vol. 42, No. 7, 1999, pp. 53-61.