Barbarians at the Gate

Intrusion detection systems are becoming increasingly important in network security. Here's a primer on what they do and how they work—and an evaluation of four popular sentries.

by Greg Saoutine et al.

September 2002 — When discussing network security, many IT professionals associate security technologies with network access control mechanisms such as firewalls, proxy servers and router access control lists. They're designed to restrict the flow of traffic into and out of corporate networks to reduce the risk of compromise by malicious individuals on the Internet. However, even the best filtering devices do very little to protect a network if Internet-accessible devices are vulnerable to attacks. Malicious individuals can use open communications channels (such as HTTP, e-mail and DNS) to compromise a system on a network remotely, and use that compromised system as a launching pad to attack and compromise other "trusted" systems on the network.

Many of today's IT infrastructures have become so large, complex and geographically dispersed that IT managers may not learn about the defacement of their Web page in a foreign data center for several days. In the case of one organization, an IT manager learned of a system compromise after a user notified the corporate public relations office. Public relations notified a top corporate officer, who relayed the information to the manager. This isn't how most of us would like to learn about such occurrences.

The good news is that intrusion detection systems (IDS) can fill such security gaps by providing a way to monitor networks for the presence of active exploits and misuse. This article provides a brief overview of network-based intrusion detection that can help you evaluate your current security infrastructure.

What is IDS?

Intrusion detection is a security process designed to monitor and analyze system or network events to detect possible misuse or unauthorized access. Broadly defined, IDS are technologies intended to support the intrusion detection process. They can be categorized into network-based (NIDS) and host-based systems (HIDS). Within these two categories, a number of sub-categories have emerged: embedded IDS, honeypots, file integrity checkers and even anti-virus scanners. New concepts and terms are evolving in this field as the technology progresses and vendors attempt to introduce new functionality and differentiate their products in this growing space. Companies are beginning to build upon concepts such as protocol anomaly detection and distributed IDS event correlation; but the majority of the systems currently in use are classified as pattern-matching systems.

IDS is similar to a motion detector installed in your house that's connected to a central alarm system. When locks fail to keep intruders outside, the sensor will detect motion and trigger an alert. Of course, it's important to locate the sensor properly in your house and activate it. After all, a sensor in your garage won't help against intruders breaking in through your kitchen window.

Network Intrusion Detection Systems

NIDS monitor network traffic and report on specific events on the network. NIDS are usually based on two principles: pattern-matching and statistical or algorithmic anomaly detection. Pattern-matching deals with identifying a specific sequence or attribute within a packet, such as matching source or destination IP address or certain characters within packet headers and payloads. Statistical or algorithmic anomaly detection identifies whether the traffic meets certain benchmarks, such as detecting the number of SYN packets sent per minute or a number of connections attempted in a given period.

In NIDS terms, the patterns are usually called "signatures," and statistical or algorithmic benchmarks are called "rules." For example, the Code Red worm can be identified within the network traffic as a packet addressed to a destination HTTP port 80 and containing the following pattern (or signature) within its payload (simplified for publication):

```
GET /default.ida? XXXXXXXXXXXXXXXX
XXXXXXXXXXXXX%u9090%u6858%
```

Similarly, unauthorized login attempts or brute force attacks can be detected via reporting on a statistical or algorithmic anomaly (NIDS rule) of 10 to 20 unsuccessful FTP login requests per minute. Modern NIDS can be programmed to report almost any kind of hypothetical anomaly.

Product Information

Dragon
Starts at $3,000 for 10 MB of network bandwidth, or $750 per host.
Enterasys Networks
Portsmouth, New Hampshire
603-501-5500
www.enterasys.com/ids/

ISS RealSecure
$8,995 per server for version 7.0.
Internet Security Systems
Atlanta, Georgia
404-236-2600
www.iss.net

NetProwler
1-50 nodes, $2,995; 1-150 nodes, $5,995
Symantec Corp.
Cupertino, California
408-253-9600
http://enterprisesecurity.symantec.com/

Snort with ACID Console
Free (open source)
www.snort.org

It's important to mention that many NIDS products (including those reviewed in this article) allow network administrators to create signatures and rules. Therefore, NIDS can be used not only to detect active attacks, but also for network

troubleshooting purposes or to fish for any packets that could be interesting to networking personnel. For example, we've found it useful to watch for the use of "public" and "private" community strings in SNMP packets, as our corporate policy dictates that all SNMP community strings must be changed from the default. We're able to use NIDS to identify systems that don't comply with this rule.

Of course, the ability of an IDS system to do its job implies that the IDS sensor is capable of "sniffing" all traffic to and from monitored systems and matching each packet with the database of malicious signatures. This fact necessitates the placement of the NIDS on network segments where all traffic to be monitored can be captured. The use of mirrored or span ports on switches or placement hubs on the uplink ports of switches or router interfaces can usually get the job done. In redundant network architecture, each redundant link must have its own NIDS. Avoid creating single points of failure on the network when designing an intrusion detection solution.

The majority of studies show that most malicious attacks occur on the internal network, and organizations should seriously consider implementing IDS not only in the DMZ or other externally accessible segments, but also on the internal network. Threats of unauthorized access from within the internal network are also high given the growing number of third parties (consultants, vendors and employees) having access to corporate networks. The biggest challenge in monitoring the internal network for malicious activity is that such activity may come from authorized access—users may not have to use hacking techniques reportable by the IDS. For that reason, more and more organizations choose to use host-based intrusion detection systems (HIDS) in conjunction with the NIDS.

Host-Based Intrusion

Host-based intrusion detection systems are designed to detect host-based events, such as using the Windows 2000 "Run As" command or gaining legitimate access to a confidential file, which may signify unauthorized access if issued during off hours. Host-based IDS mechanisms range from basic system auditing to advanced detection methods and have proved to be most efficient in tracking potential system misuse from users on the internal network.

In many cases, IDS vendors offer both network and host-based systems, consolidating the alerts into a common reporting console. While host-based intrusion detection is an important component of an organization's security strategy, this article focuses on NIDS, the network-based approach.

IDS is Not a Cure-All

Intrusion detection systems aren't the magic security bullet some people perceive them to be. Encryption, designed to prevent unauthorized disclosure of information, is particularly troublesome for intrusion detection systems. Widely used network security protocols such as SSL and SSH, as well as IPSec used in virtual private networks, preclude NIDS from inspecting network traffic for comparison against attack signatures. Encrypted payloads make these protocols excellent vehicles for an attacker to evade detection by an IDS.

Another important limitation for NIDS is network speed. Hardware limitations and inefficiencies in pattern-matching algorithms impose limitations on the number of packets that can be processed for a given period. Once this limit is reached, IDS usually starts "dropping" packets (ignoring the traffic). Few NIDS available on the market today can work on gigabit networks.

To make it easier for the black hat community to attack infrastructures protected with NIDS, a number of tools that circumvent the operation of NIDS sensors have emerged. Some of these tools (such as Stick, which is described later) are able to send an entire database of attack signatures past a NIDS sensor and make the console light up like a Christmas tree generating false alerts. Organizations must have a clear understanding of the risks such tools represent to the NIDS' operations and how to react to these alerts. Usually the attackers try to evade detection by your NIDS in order to execute real attacks and make it hard (if not impossible) to detect them.

Knowledge, Process and Technology

While IDS is an important part of the organizational defense posture, it's only effective when coupled with knowledge and established processes.

By knowledge, we mean your in-depth understanding of network protocols, awareness of new attacks, and understanding of your networks (in other words, how layer 2 and 3 traffic is flowing through the network to and from specific servers). By process, we mean diligent application of signature upgrade procedures, continuous monitoring, ability to validate certain events, and your preparedness to react properly to detected events.

IDS itself won't change the security of your organization overnight, but, when combined with knowledge and processes, it will slowly but surely pinpoint current network deficiencies and enable you to react in response to detected attacks.

Despite what the vendors say, effective NIDS are a relatively high-maintenance technology. They can't merely be configured once on the network and left to run by themselves. New signatures, network changes and new applications all require IDS tweaking. To achieve maximum effectiveness, signatures must be frequently updated, thresholds must be tuned to reduce false positives, and administrators must monitor output to obtain an understanding of normal traffic patterns for their environment. Additions and deletions of network components and servers must be accompanied by necessary changes to the NIDS. Failure to do so may result in being unable to pick up important events related to the newly deployed production subnets. These activities can be resource intensive if proper processes and administrative resources aren't in place. To alleviate the strain these requirements may impose, a number of third-party managed security service offerings have sprung up to handle outsourcing of these activities.

Today, NIDS can also be configured to manage the traffic proactively on their own or by communicating with firewalls and other network components in response to events. For example, ISS RealSecure, Snort and Dragon can reset TCP sessions based on a specific trigger (by sending an RST packet to the source). ISS RealSecure can also integrate with Microsoft ISA Server and Checkpoint FireWall-1 to cause an automated configuration change based on certain events. (We should note that if you're familiar with Dragon, then you know it's a purely Unix tool. Why is it showing up in these pages? Because these days most IDSs are Unix-based—and Dragon's an excellent example of the genre.)

5 Deployment Caveats

1. Deployment of IDSs need to account for VPN, IPsec and other encryption usage. For example, if you allow encrypted traffic to and from Web and application servers, IDS may not be able to help.
2. For ultimate defense, IDSs must cover redundant and fail-over links. This may be an engineering challenge (how do you properly match a signature that happens to be split among several wires?) and budgetary pressure due to extra hardware and software requirements.
3. Make sure sensitive administrative information is properly secured. In one case, we noticed a NIDS capturing sensitive information, which could have been used to gain access to corporate firewalls and perform reconfiguration. The IDS was outsourced and firewalls were managed in-house—not a separation of duties we'd want to rely on! (Does your IDS capture your telnet password?)
4. How IDS is "tapped" into networks may significantly affect the usability of this technology. Make sure you consult your router and switch vendor prior to any implementations. Some vendors have limitations, which don't allow for capturing of all the traffic on a specific device.
5. Last but not least, users and customers may need to be notified about the presence of such technology on your network. We believe, however, that in most (but not all) cases, the presence of NIDS should remain a secret and NIDS shouldn't respond to any network events. The fact that user traffic is monitored may have legal implications and may have to be disclosed to the user community via a high-level communication.

Can They Be Trusted?

Most industry experts agree that intrusion detection systems haven't reached the point where they can be trusted to respond to attacks by automatically reconfiguring network equipment. The problem with automatic response to network events (such as system reconfiguration) is that your IDS installation becomes a potential threat. If attackers realize that you block port 80 to the source IP of a malicious packet, they can easily spoof legitimate source IPs of your partners and clients in the malicious packets addressed to your Web server, thus denying service to the legitimate users. However, if you perceive a denial-of-service as having less impact on your environment than unauthorized access, you may still choose to deploy automatic system reconfiguration.

Another complication comes from the architecture of IDSs. In many cases, enterprise deployments require communication between agents (traffic-sniffing devices) and managers (devices that govern the distribution of policies and how alerts are processed between the management stations and consoles). For this, firewalls need to be configured for the "trusted" IP addresses in the DMZ (or even worse, on the Internet) to be allowed access to the network management station. If proper consideration isn't given to such configurations, NIDS deployment may lead to a lot of damage. In some cases, you might need to create a separate management network to use additional network interfaces on the agent machines.

To summarize, the following are the tasks generally performed by a network IDS:

- Near real-time analysis of network traffic for known attack signatures.
- Near real-time detection and reporting of network traffic anomalies.
- Alerting administrative personnel when a potential attack is detected.
- Taking a pre-defined corrective action in response to an event.
- Sophisticated non-real time reporting and data analysis.

The Test Network

To perform our product evaluations, we set up a Win2K (SP2)-based IIS 5.0 Web server in a test lab. For the sake of simplicity, the network interface of the Web server was plugged into the same hub as the NIDS agents, allowing for the NIDS to capture all traffic directed toward the Web server. We implemented no firewalls to filter traffic to or from the Web server. We performed testing on a LAN, as well as across the Internet.

The Attackers

We used the following tools to scan the Web server for services and vulnerabilities, as well as execute attacks against the server and the NIDS:

- *Nmap port scanner.* Nmap is a fast, efficient port scanner. Nmap TCP stealth port scan, UDP scan and ping sweep were executed against the Web server. www.insecure.org
- *ISS Internet Scanner.* This is one of the most popular port and vulnerability scanners. Vulnerability scans, as well as actual denial-of-service (DoS) and buffer overflow attacks were executed against the Web server. www.iss.net
- *Nessus vulnerabity scanner.* Nessus is a popular open source vulnerability scanner. Vulnerability scans, as well as actual DoS and buffer overflow attacks were executed against the Web server. www.nessus.org.
- *Arirang CGI scanner.* Arirang is a fast, efficient scanner designed to identify common gateway interface (CGI) vulnerabilities. www.freebsd.org/ports/security.html.
- *Stick.* Stick is an IDS obfuscation tool. It's designed to send attack signatures from a Snort signature file against the target, creating a large number of false alarms on the NIDS. Note: Because Stick spoofs legitimate source IP addresses, it should never be used on networks connected to the Internet. http://packetstorm.dnsi.info/distributed.
- *Fragroute.* Fragroute's objective is the opposite of Stick: It fragments outgoing malicious packets within the source of attack in a way that's supposed to make the target NIDS blind to these attacks. www.monkey.org/~dugsong/fragroute.

The Contenders

We tested the functionality of the following intrusion detection systems.

NetProwler 3.5.1

Symantec NetProwler (Figure 1) supports Windows 98, NT and Win2K. It's an old-time IDS system; the current version dates back to August 2000, which is probably attributable to the acquisition of Axent by Symantec. NetProwler Commercial support is included in the base contract.

Figure 1. NetProwler gives a detailed breakdown of a network attack.

NetProwler consists of console, manager and agent modules. Agent and manager work only on NT 4.0 with SP3 and later. Console works on Windows 9x, Win2K and NT. (It may work on XP but we haven't tested this). NetProwler has a signature synchronization feature that allows for an automatic download of newer rules. The product allows users to configure various alerts as well as execute various commands based on triggered signatures. Users can push policies (containing signatures and rules) to remote NIDS sensors, thus allowing for centralized remote administration.

ISS RealSecure 6.5

ISS RealSecure 6.5 supports Windows NT 4.0/2000, SunOS 2.5.1, Solaris 2.6, 7, 8, HP-UX 11, IBM AIX 4.3, Nokia Appliance IPSO, and Linux RH 7.1. Both NIDS and HIDS agents can be supported from the same console. You can customize policies and select specific signatures. Most of the signatures for the known attacks are proprietary and must be downloaded from the vendor via a remote update. ISS Real Secure also allows users to define signatures via a simple API interface. Commercial support is included in the base contract, and managed service offerings are available.

Installation of RealSecure 6.5 on Win2K requires MSDE2000, the RealSecure Console (Figure 2), Event Collector, and sensor (network or host/server). All communications between the agents and the event collectors are encrypted. Users must provide a license key during the installation.

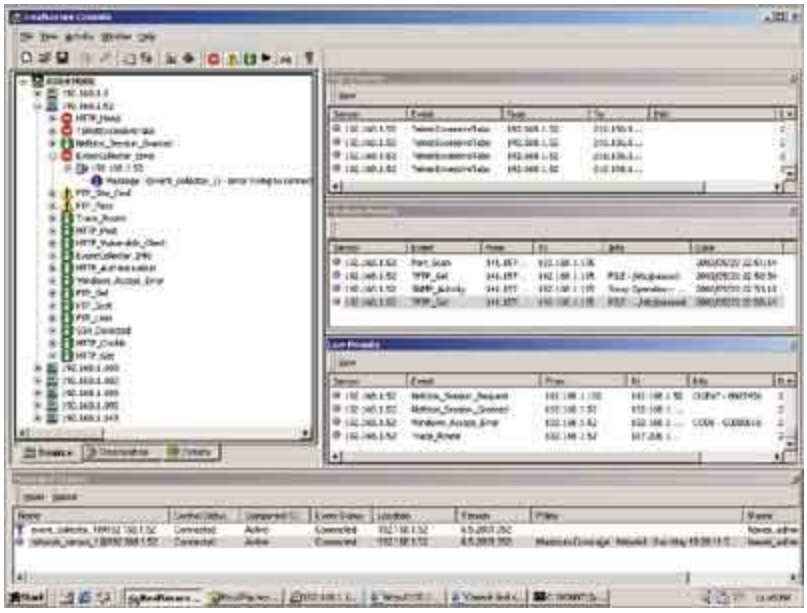

Figure 2. ISS's RealSecure allows users to select which attacks should be reported and how the reporting should occur. *(Click image to view larger version.)*

Among the features provided by RealSecure are automated signature updates, signature selection, signature creation (simple API), manipulation of network traffic (connection reset), plug-ins for other products (ISA server, Checkpoint FW-1), session recording and playback, SNMP traps, e-mail alerts, remote policy management from the central console and reporting and graphs (Crystal Reports).

RealSecure 7.0 was released after this article was written.

Dragon 5.0.2

Dragon supports Linux, Solaris Sparc, Solaris x86, HP UX, FreeBSD, and OpenBSD. This Unix-based product and the console requires (similar to Snort) a basic Web server. All alerts and reporting are displayed within a Web browser (Figure 3). Dragon provides a low-level interface for editing and configuring signatures; a feature for automatic signature updates is available. Commercial support is included in the base contract.

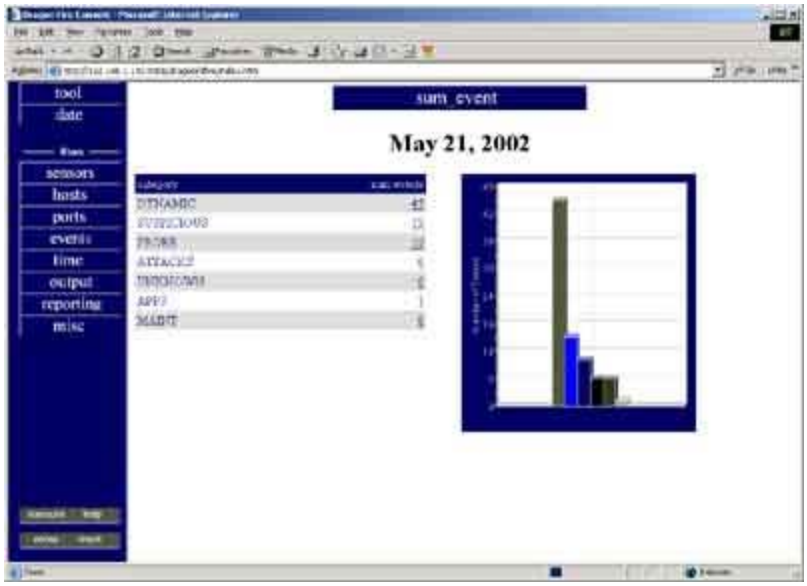

Figure 3. Enterasys Networks' Dragon has a Web interface for creating verbose and graph reports on the captured data. *(Click image to view larger version.)*

Most of the components (including the database) shipped with Dragon are proprietary. The exception is a Web server, which is required for Dragon installation. All communications between the agents and console are encrypted. Users must provide a license key during the installation.

Most of the signatures for the known attacks are proprietary and must be downloaded from the vendor via a remote update. Users can push policies (containing signatures and rules) to remote NIDS sensors, thus allowing for centralized remote administration. Among its features are automated manipulation of network traffic (connection resets).

Snort 1.8.7b121 with Analysis Console for Intrusion Detection 0.96b21

Snort is an open source tool that supports Windows 95, 98, Me and XP; NT; Win2K; Solaris; FreeBSD; OpenBSD; and Linux. Source code is also provided for compilation on other platforms.

Snort is a free IDS partially based on the TCPDump. On its own it can sniff traffic, log traffic to a text file or log alerts. It can also log to a database, create SNMP traps, log to an XML file, post to an HTTP server, send SMB alerts, log to the EventLog and log to a syslog server. It can be combined with the Analysis Console for Intrusion Detection (ACID) console (Figure 4), MySQL, PHP, and a Web server to add more robust alert reporting and analysis. The open source community has developed a number of tools to complement and extend Snort's native capabilities. Signatures for publicly disclosed vulnerabilities are available from the Snort Web site and third-party sites such as the www.whitehats.com "arachNIDs" database. The Snort rules scripting language facilitates the creation of new signatures and the customization of existing signatures.

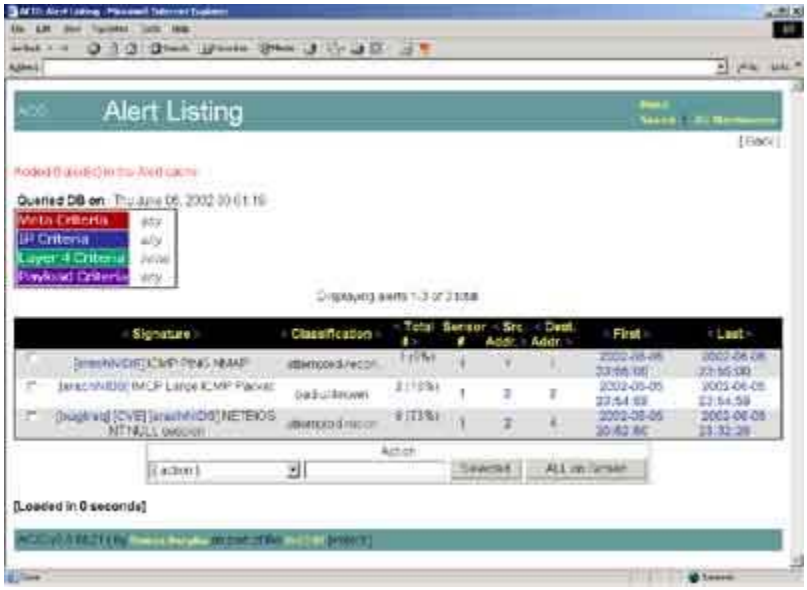

Figure 4. Working in concert with Snort, ACID can conveniently group attacks by classification, reporting sensor and source and destination IP address. *(Click image to view larger version.)*

If your company prefers commercial support, it's available from Silicon Defense, www.silicondefense.com. Also, Source Fire, www.sourcefire.com, provides support for Snort-based hardware appliances. Free support is available via Internet mailing lists.

Installation of Snort requires a Web server supporting Perl and PHP, the ACID console (a number of PHP scripts), database for alert logging, and WinPcap packet capture driver. Comprehensive Windows installation instructions are available from multiple sources listed at www.silicondefense.com/techsupport/downloads.htm.

The Tale of the Tape

Our product tests used evaluation versions of all software, except Snort. Dragon 5.0.2 provides a limited number of signatures and capabilities in its evaluation version. All tests were performed using additional sniffers and session loggers to ensure that expected traffic was, indeed, passed between the attacker and Web server.

All NIDS detected TCP stealth scans performed by nmap. Snort identified nmap as the tool used for the port scan, while other products merely established the fact that port scans were performed. Dragon didn't report any alarms associated with the UDP scan, while RealSecure, NetProwler and Snort triggered several alerts related to UDP scanning, unsolicited DNS replies (NetProwler) and UDP-based attacks (Snort). UDP scan alert on the three out of four products was a correct alarm. The rest of the reported events can be considered false positives in this case (or extra "noise"), as no actual UDP attacks were executed as part of this activity. Only Snort accurately detected a ping sweep and established that it was

performed by nmap. While port scans are common on the Internet, an activity such as this may contradict corporate policy and require an investigation if detected on the internal network.

All four products easily identified a vulnerability scan executed by ISS Network Scanner 6.2.1. However, only RealSecure and Dragon identified ISS Network Scanner as the agent probing for vulnerabilities and attacks against the target system. Snort reported more than 100 alerts in the first two minutes of the scan, and NetProwler identified a port scan as well as multiple TFTP and FTP attacks. The fact that some of the products are able to accurately recognize the signatures of the tools used to probe for vulnerabilities or execute the attacks may prove useful in the validation of events and further escalation and investigation.

All four products detected a massive 30-minute Nessus scan via the Internet. RealSecure and Dragon logged 21 and 13 unique alert signatures, respectively, precisely identifying some of the attacks (such as SNMP vulnerabilities specific to HP OpenView and Cisco's IOS). NetProwler caught just a handful of attacks related to Back Orifice (a popular Trojan back door), TFTP, FTP, as well as port scans. Snort generated more than 2,000 alerts related to about 30 unique signatures.

RealSecure, NetProwler and Snort identified probes for common gateway interface (CGI) vulnerabilities generated by Arirang. The evaluation version of Dragon remained silent. Since port 80 (HTTP) remains open on the majority of corporate firewalls, it's critical to ensure that CGI scripts on the Web servers have strong security. You'll also want to know if probing for specific CGI components applicable to your environment takes place.

The test using Stick proved the point that IDS is subject to obfuscation through flooding. All four tools generated thousands of alerts within seconds. However, because Stick is designed to send the IDS rule base toward the target, it comes as no surprise and proves only one point: Depending on the flow of traffic visible to the NIDS, as well as traffic filtering patterns around the NIDS, alert reporting and processing may become unmanageable. Note, however, that IDSs are being improved to become much more resistant to Stick.

One other useful application of Stick is the ability to stress-test your IDS system. Because Stick is designed to produce as much traffic as the hardware of the source system will allow, it's possible to establish the CPU utilization, packet drop rate and other important limits of the NIDS.

The "fragroute" test showed that RealSecure can reassemble fragmented attack packets correctly and still report on the same attack signatures generated by Stick. Dragon didn't pick up any attacks, but did generate hundreds of informational messages reporting on the "odd TCP flag combinations." Therefore, instead of correctly reassembling the packets, Dragon identified "protocol anomaly" in the network traffic and alluded to suspicious activity. In a production environment, protocol anomaly validation and investigation can be incredibly difficult and time consuming.

Our tests determined that all the products are effective in detecting and reporting on suspicious traffic and malicious attacks. In choosing your IDS solution, you need a clear understanding of product features and functionality, along with your organization's objectives of the implementation (simple detection of known attacks vs. network traffic monitoring for operational and troubleshooting purposes). An organization should assess whether it's able to manage and respond to certain events internally or whether it should rely on a managed service provider. Finally, the IT department may choose to test and evaluate the product on its own or rely on technical evaluations provided by various research labs (such as Neohapsis at www.neohapsis.com/neolabs/default.php or Miercom at www.mier.com/ testserv.html).

And the Winner is...

In our opinion, Snort stands out in the evaluation. It's a system embraced by many security professionals, and it comes with powerful interface additions and a large number of signatures that can be tailored to your requirements. Support for Snort is also available through commercial providers like Source Fire.

ISS, publisher of RealSecure, appears to have improved in its effectiveness and product support over the last year. In our experience, ISS products are easy to use and equipped with useful features, though by means of proprietary technology and in some cases at the expense of performance.

NetProwler appears to be the most outdated NIDS product. It's especially alarming that its critical components are limited to NT 4.0, as Microsoft is phasing out sales and support of the OS. If Symantec is serious about NetProwler, it should be at work on a next-generation release. Dragon proved to be a solid system featuring Java-based Web reporting interfaces. Enterasys just recently released version 6 of the software. Dragon goes beyond the NIDS arena and today provides a full suite of IDS features, including third-party IDS managed services.

12 Steps to Successful IDS Deployment

1. Define your objectives and establish a preliminary budget. Identify critical systems and/or network segments that you need to monitor. Establish what you're looking for and where such traffic can be captured in its entirety. Based on this analysis, you'll be able to define the scope of your implementation as well as start the product evaluation process. In many cases, budget will drive further product selection.
2. Plan your IDS deployment. Remember that a successful NIDS deployment should be part of the overall

security strategy. Your IT department has to budget for both the cost of implementation as well as tune-up, monitoring, alerting and response procedures. Decide whether to build this expertise and process in-house, outsource it or do both.

3. Select the products based on what's important to your organization. You should at least consider: a) placement preferences and performance requirements (line speeds); b) business applications to monitor (Web, e-mail or general); c) threats and technologies to detect (protocols, internal/external access); d) degree of signature customization available; e) reporting mechanisms; and f) available real-time alert notification methods.
4. Architect your NIDS deployment. Decide whether to deploy a management network or integrate with the main network. Design encryption channels and access controls for reporting of alerts, build redundancy in the reporting infrastructure, and establish channels for console reporting as well as automated pager/e-mail alert notification.
5. Prioritize the alerts you want your NIDS system to generate.
6. Establish incident response procedures. Without incident response, NIDS deployment may turn out to be useless or just be used to collect forensics data. All parties involved in the NIDS management and monitoring have to have a very clear understanding of what action to take based on a specific type of alert. The incident response procedure should specify actions to take in case of NIDS obfuscation events, as well as in the case of critical security events. The procedures should clearly state when or if to call law enforcement and the possibility of discontinuing IT operations in case of a serious compromise. Finally, the procedures should state requirements for data collection and retention for forensic analysis, if needed. You can obtain more information on incident response procedures from CERT at www.cert.org/nav/index_red.html.
7. Conduct a pilot implementation. At this stage, you should still be able to evaluate some of the products and test your hypothesis regarding security events you're trying to detect. You can also test your incident response procedures at this point.
8. Train personnel responsible for NIDS deployment, management and administration. If the NIDS is deployed in-house, provide extensive training at all levels. If the NIDS is outsourced, associates serving as liaisons with the vendor should be trained in NIDS technologies and incident response procedures.
9. Deploy NIDS.
10. Test your installation. Industry averages show it takes two to 10 months to tune a NIDS implementation. Continuous testing, data analysis and configuration adjustments ensure your NIDS deployment works as expected.
11. Proactively monitor news and vendor announcements for new threats and attacks. Perform timely updates to your NIDS installation to detect all attacks and anomalies of interest.
12. Assess your threats and evaluate the adequacy of your NIDS deployment as your network topology continues to change.

Keeping the Barbarians at Bay

Intrusion detection systems are starting to achieve the level of ubiquity in the corporate environment that firewalls attained in the late 1990s. More and more, NIDS are seen as an important component of a secure e-business infrastructure—as proven by the number of regulatory agencies and industry groups requiring their presence. In the U.S., this is especially true of the financial services and healthcare industries, as well as other industries that are part of the "critical infrastructure." While we've given a broad overview of IDS and sampled some of the better known products, this is really just an introduction to the topic. If you have security responsibilities on your network, you owe it to your employer or clients to delve deeper into intrusion detection and learn all you can about this growing and increasingly crucial field.

Sergey Perfiliev, MCSE, CCDP, Sun Certified System Administrator, is a Certified Information Systems Auditor with 10-plus years of developing, administering and auditing client-server and distributed systems. His interests focus on emerging issues in information security. He can be reached at sergperfi@yahoo.com.

Robert Corti, MCSE, CCNA, CCDA, CISSP, has spent several years as a New York City-based security consultant for Fortune 1000 companies. Currently, Robert is a risk management professional for a New York-based technology company and can be reached at rmcorti@yahoo.com.

Greg Saoutine, MCSE, CCNA, Sun Certified System Administrator, has spent several years as a security consultant in the New York City area. Currently, Greg is an IT security manager for a Fortune 500 company. You can contact Greg Saoutine about "Barbarians at the Gate" at gsaoutine@hotmail.com.