Coverage in Intrusion Detection Systems

Marcus J. Ranum
<mjr@nfr.com>

Chief Technology Officer,
NFR Security, Inc.
http://www.nfr.com

Abstract

Intrusion Detection Systems *(hereafter abbreviated as "IDS") are a topic that has recently garnered much interest in the computer security community. In the last few years, this interest level has spurred the development of a variety of approaches to providing IDS capabilities that are both reliable and low-impact in terms of management or cost. When presented with different types of IDS one might be tempted to assume that one approach or another was inherently superior. In fact, the mixture of approaches used for IDS offers the security analyst a unique opportunity in terms of the synergies inherent in combined techniques. This paper discusses these synergies, termed "****IDS Coverage****" for short, the way in which the mixed strengths of existing approaches cancel out many of their weaknesses to produce an extremely reliable IDS capability.*

Introduction: The Importance of Coverage

In implementing defensive systems for computer security, the accepted reality is that a chain is only as strong as its weakest link. In intrusion detection, however, the situation is slightly different. Detecting hostile actions becomes a matter of probability, and the likelihood of detecting a given hostile action depends on the number and type of sensors arrayed to detect it. The simplest metaphor is real-world intrusion detection systems – known as "burglar alarms." In the author's house there are: motion detectors, magnetic door and window switches, and glass breakage detectors. These three sensor types each have varying strengths and weaknesses, but taken together they provide much better coverage than any single sensor can afford. For example, if a burglar were to kick out a pane of glass to open a door, it might set off all three sensors simultaneously. The glass break detector hears the pane shatter, the motion sensor detects the motion of the door's opening, and the door's magnetic reed switch detects that it is no longer in contact with the door frame. This represents a ***best case detection coverage***: all sensors agree as to what happened. A worst case of detection coverage occurs when none of the sensors detect anything whatsoever. However, it's obvious (at least in the real-world example) that it's harder for the attacker to effectively mask or evade three different sensors than it would be to avoid any individual one. Does the same apply for computer IDS? As we shall see, it does. The most popular types of IDS available today have highly complimentary strengths and weaknesses – combinations that suggest that, while either solution alone is better than nothing, both solutions together may be much better than the end user would expect.

Another important aspect of good IDS coverage is correlation and agreement. ***IDS Correlation*** is the process of associating IDS events when there may not necessarily be a relationship between the events. ***IDS Agreement*** is the process of weeding out duplicate IDS events in the case where multiple sensors agree that the same thing has taken place. The author's home burglar alarm performs a simple form of IDS agreement: any and all events sufficient to trigger an alarm will be lumped together into a single alert being generated. A more complex form of agreement in a home burglar alarm would occur if the alarm called the police and said "glass broke, a door was opened, and there was movement" instead of just "something is wrong." IDS correlation is a similar process but it's typically time-based: "glass broke and within 500 seconds there was also movement." The reason it is valuable to have multiple sensors agree is because each different sensor strengthens the analyst's understanding of what is taking place – because each different sensor sees the world in a different way.

Some sites perform this kind of search for agreement today using a variety of IDS to analyze the same data. If Product A agrees with Product B the analyst's confidence in the diagnosis is stronger than if Product A reports a significant attack in progress and Product B reports a minor software problem. Counter-intuitively, many security administrators are *more* confident in the case where Product A reports a significant attack and Product B *fails to report anything at all*. In other words, *agreement* is valued more than contradictory information. It is easier for the analyst to be comfortable with the assumption that one IDS "missed" an attack than with the knowledge that two IDS disagree as to the significance or nature of an attack. This observation is going to be critical in the future, if IDS data is used as a primary factor in decision-making for reactive security systems. A security analyst will be much more comfortable inserting a screening rule in a router based on a non-contradictory diagnosis than a contradictory one. The only way to decide who is right is to research the attack traces in more detail – by which time it is possibly too late.

In the future, the author expects to see IDS become primary sensors in a security decision support infrastructure, in which a key factor will be the coverage afforded by different types of sensors. Since each type of sensors is very effective at some kinds of analysis, the trick will be to ensure that sensors agree as much as possible, yet are still sufficiently sensitive to independently raise an alarm when they are certain that a problem has occurred.

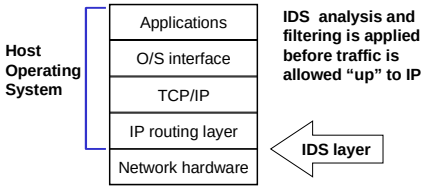
IDS Evolutionary Trends

Let's briefly examine the two primary types of IDS, so that we can understand their various strengths and weaknesses as we search for an optimal way to combine them. Historically, the first IDS were ***host-based IDS*** (hereafter abbreviated as "HIDS"). The first IDS products to achieve significant market success, however, were ***network IDS*** (hereafter abbreviated as "NIDS"). Today, there is a variety of NIDS and HIDS available, and we are beginning to see some hybrid systems that exist between the two categories and are best considered on a case-by-case basis.

The first IDS designs were HIDS based on academic research in the late 1980's.[1] First generation HIDS relied almost entirely on system log data and C2 security logs.[2] These systems were primarily based on auditing existing event information: correlating data that had already been collected by other subsystems within the computer. Unfortunately, the early HIDS concept had a fatal flaw:

Figure 1:

Shim-style Hybrid IDS



The whole premise of HIDS is based on the assumption that the system will be compromised – making it impossible to place trust in the data upon which the HIDS relies.

Early HIDS were designed to operate on large multi-user systems running timeshared operating systems. As such, they tended to focus on detecting attempts by one user to access another user's files or to elevate their

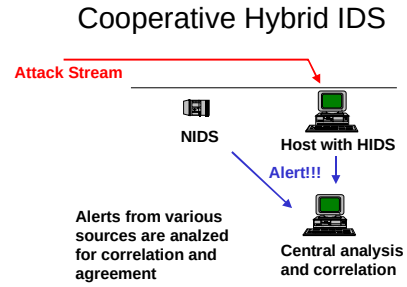
[1] Dorothy Denning (1986) "An Intrusion Detection Model"
[2] IDS and DIDS, research systems developed at SRI and Stalker, a commercial product from Haystack Labs

permissions and compromise the system's security. Prior to the advent of today's point-and-click hacking tools, such exploratory activities on the part of a user were easy to detect in a timely manner. In today's hacking environment an attack can be launched and completed in under a millisecond – including the deletion of incriminating system log files or audit information needed by a HIDS. Cheap, popular desktop computing environments also cause problems for HIDS because many popular desktop operating systems such as Windows 95-98 and Windows ME lack system logging facilities, operating system permissions protections, or a strong notion of user identity. Until systems were networked it was basically pointless to try to detect a user attempting to gain elevated permissions in an operating system where everyone has full access simply by walking up to the keyboard.

Since HIDS were having trouble adapting to the new environment of distributed desktops and networks, IDS designers turned to a new approach: the NIDS. A NIDS operates by accessing traffic off a network's broadcast medium. With an ethernet-type network, this entails placing the interface card into "promiscuous" mode – a mode in which the card collects all the traffic crossing the network segment whether or not it is destined to the system which is listening. The listening system, the NIDS, then tries to detect attack patterns within the collected traffic, treating the network traffic as data to be examined passively. NIDS are today's predominant IDS architecture.

Today, IDS designers are mixing the properties of HIDS and NIDS in various creative ways to produce hybrid systems. The simplest form of hybrid is the "shim" IDS as illustrated in Figure 1. In the Shim-IDS, an interface layer is added into the operating system's network stack so that all traffic reaching the system is first passed through the IDS interface for analysis. This approach basically makes each host run a mini-NIDS within its operating system environment. It has some of the advantages of both a NIDS and a HIDS and some of their disadvantages, as well. For now, we're looking at IDS from a perspective of coverage, so we'll only compare and contrast hybrid systems in so far as it affects their ability to broaden their detection ability. Another popular form of hybrid is NIDS/HIDS that have been designed to work together (see Figure 2).[3] Such cooperative IDS allow a great deal of flexibility in selecting where and how to cover analysis within a network and across hosts.

Figure 2:



Coverage: Things NIDS do Best

The NIDS approach has a number of attractive properties, which have made them the predominant IDS technique in the past five years:

- NIDS are installed per network segment rather than per host; coverage of 100 systems might require only one NIDS compared to installing a HIDS on each of the 100 systems.
- NIDS deal with traffic as abstract data; a denial of service or "death packet" which might crash a target host will not affect the NIDS.

[3] This is the approach the author's company, NFR Security, has chosen to take.

- NIDS reassemble and analyze traffic with an awareness of possible network-layer errors (e.g.: bad IP checksums) that might be shielded from the host/application level software that a HIDS relies on.
- NIDS are operating environment independent; a Windows machine looks pretty much the same to a NIDS as a UNIX machine, since they are both running TCP/IP and standard Internet applications.
- NIDS may be invisible to the attacker; it is possible for a NIDS to monitor a network without revealing its presence, while a HIDS will almost certainly leave some software "footprint" on systems where it is installed.

In other words, NIDS excel at detecting network-level abnormalities and abuses. NIDS have a few areas in which they are weak when compared to a HIDS:

- NIDS may miss packets due to congestion on the network link that they are monitoring.
- NIDS do not have a good notion of user identity; since TCP/IP traffic does not convey an association between the logged-in user and the connection/traffic, it is only possible to infer who did what by circumstantial evidence. For example, a NIDS can possibly tell the user-id of a web-surfer from information offered by their browser, but a sophisticated attacker can mask that information easily.
- NIDS may not have a good notion of what traffic the target system actually *received*, since its view is only of traffic that it saw being *sent*. In order for a NIDS to be confident that it is correctly analyzing the traffic sent to the target, it must track the acknowledgement packets and TCP windows in each data connection. This is very difficult analysis to perform accurately, and few commercial NIDS even try.[4]
- NIDS may have difficulty knowing if an attack is *relevant* to a particular target. While this isn't a major shortcoming per se, it is a potential irritant. If an attacker launches a buffer overrun attack designed to work against a Windows web server, it will not affect a SPARC system running Solaris – the NIDS will probably still generate an alert because it sees the attack. The NIDS would have difficulty telling the administrator accurately whether or not the attack had any effect.

In other words, NIDS' weaknesses primarily have to do with their ability to understand what is going on within the host: who the user is, how the host is interpreting the attacks as seen, and whether the attack worked on the host. By itself a NIDS is still a valuable tool, but a sophisticated attacker might be able to exploit its shortcomings to mask their actions.

Coverage: Things HIDS do Best

HIDS reside within the host operating environment and, as such, have access to whatever information they can glean from the system itself. The first generation HIDS only dealt with data that had already been collected for them by existing audit facilities.

[4] The NFR Security NIDS solution tracks packet acknowledgements returned by the target host. This makes it impossible to fool the NFR NIDS using out-of-sequence spoofed packets or packets with short Time-to-live (TTL) counts to obscure an attack.

The current generation HIDS are more aggressive about collecting information – they install agents that monitor system processes, check and watch registry entries, observe who is logged in and where they logged in from, and collect data about what programs are running, including their CPU usage and file accesses. Another important function that appears in many HIDS is file integrity checking. The HIDS is given a list of important system files to check, which are cryptographically check summed and compared against a checksum database. If the files are changed, an alert is generated. File integrity checking is a very effective tool for detecting the installation of Trojan horses, the addition of new users, or the alteration of system configurations. Finally, when all the data from a HIDS is taken together, the administrator has an impressive degree of coverage regarding events within the host. For example, the file integrity check for a critical system file might fail and the administrator might check the HIDS logs pertaining to who was logged in at the time of the change, and what programs they were running. Since the HIDS is effectively a part of the target, it is able to provide very good data about what was happening to the host, and when.

Strengths of HIDS are:

- HIDS are able to associate users and programs with their effects on a system; they can tell you what user issued what command and when.
- HIDS are part of the target and are therefore able to give very good information about the state of the system during an attack.
- HIDS only have to deal with attacks directed at the target itself; they do not have to worry about capturing all the packets that cross a network – therefore they are much less computationally expensive and have relatively low performance impact on the host platform.
- HIDS are able to access system information directly since they run on the system itself. This means that HIDS can examine files, operating system configurations, software release levels, etc.

As with the NIDS, the strengths of the HIDS relate directly to its weaknesses:

- Since the HIDS is part of the target, any information it provides becomes suspect the second an attack succeeds against the target. Logs the HIDS relies on may be altered or deleted, or the HIDS software itself may be deleted or tampered with.
- Since the HIDS operates at a higher level up the network stack than the NIDS, it may not have information pertaining to events lower down the network stack. For example, if a typical TCP/IP stack receives a packet that is not part of a valid connection, it simply rejects it and never notifies any other part of the system. A NIDS would notice and record the event, but a HIDS never even sees the event since the TCP/IP stack correctly discarded the packet.
- HIDS will have difficulty detecting attacks that completely wipe out the target system. Imagine if your system is brought under a “ping of death” attack and the operating system itself crashes. When the operating system is crashed, the HIDS has crashed along with it and no alert is generated. A NIDS would have immediately detected the “ping of death” packet and been unaffected since it treats packet data as abstract information for analysis, not traffic to act upon.

- HIDS analyze only the state of each individual system the HIDS is running upon. In order to provide adequate coverage of a group of systems, the data from each HIDS must be moved to a central location for correlation and data reduction.
- Operationally, HIDS are more expensive to deploy, since they involve installing software on every system that is to be monitored.
- HIDS platform coverage may be limited, since they must be ported to every desired platform. Most HIDS vendors support one or two platforms (e.g.: Windows, and one or two flavors of UNIX) few support more than three or four.

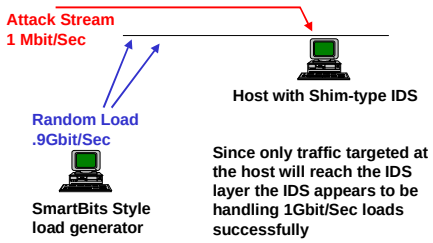
HIDS weaknesses primarily revolve around packet-oriented attacks or the weakness of the host itself when it comes to resisting attack.

Coverage Properties of Hybrid ("shim") IDS

The shim-type hybrid IDS are an interesting blend of the strengths and weaknesses of HIDS and NIDS. In practice, they operate much like a NIDS – they collect traffic at a packet level, process it, and detect or deflect attacks. But, like a HIDS, they do it on a per-host basis. Each system's IDS layer only processes the network traffic directed to that system, greatly reducing the bandwidth capacity the IDS needs to be able to cope with. Since the individual system only deals with traffic directed at itself, the CPU requirements for doing IDS processing are also lower. However, the IDS layer of one machine will be unable to detect attacks directed against an adjacent machine – the shim-type IDS are very much an "every man for himself" approach to IDS design.

Figure 3:

Benchmarking Shim-type IDS



The shim-type IDS also have an interesting property when benchmarked by naïve users – they may appear to be able to handle much higher data loads than they actually are able to. Consider a benchmark rig such as illustrated in Figure 3. We show a target host with a shim-type IDS installed, an attack generator that is generating a flow of attacks aimed at the target, and a traffic generator such as a SmartBits that generates a high peak load in order to saturate the network. Since the SmartBits is generating random traffic that is not necessarily aimed at the target host, the shim-type IDS only has to process the 1Mbit/Sec stream generated by the attack generator. An unsophisticated benchmarker might see these results and conclude that the shim-type IDS was capable of handling extreme loads. A more accurate test would be to configure the SmartBits to generate all of its traffic with the target host as the destination IP address.

Shim-type IDS have most of the advantages of a NIDS except for in the area of deployment, since they must be deployed on every host in order to function. Additionally, since they act as a shim, they may interfere with other applications that shim the TCP/IP stack in the operating system – firewalls and VPNs may not function correctly with shim-type IDS. They typically lack most of the advantages of HIDS, since they are usually network-oriented rather than application and operating system oriented. From a standpoint of pure coverage, however, they are very similar to NIDS except for the fact

that they only analyze traffic destined to a single host instead of an entire network segment.

Detection Coverage of Denial of Service and Spoofing Attacks

To illustrate the importance of good IDS coverage, let us examine a set of attacks launched against a hypothetical web site. The web site is running a version of UNIX with a version of popular open-source web server software. Monitoring the web site is a NIDS connected to its 100Mbit hub. Loaded into the web server's operating environment is a HIDS that is configured to monitor system logs, process table entries, user actions, and perform file integrity checks on the site's top-level pages.

Suppose that a conventional web-based exploitation is attempted against the server. The HIDS will detect the attack easily and quickly. On the host, however, we will see different results depending on whether or not the software on the host is actually vulnerable. If the software is vulnerable and the host is compromised, then the HIDS may detect the second-order effects of the attack as the hacker begins to exploit his foothold in the system. If the host is not vulnerable, the HIDS may not detect the attack since the host software may have successfully and silently resisted it. Most of the time, if an attack is launched it will be only the NIDS that generates an alarm, assuming that the host is configured securely. It is possible that the NIDS and HIDS will produce alerts that *agree* but it is more likely that they will produce alerts that *correlate*.

In another case, the attack launched against this web site might be a web-based attack over an SSL-encrypted link. In that event, the NIDS will be unable to provide coverage beyond recording that the encrypted connection took place. The HIDS, however, will be able to provide full coverage if the web server software logs an error, or if the server is compromised and the attacker begins to exploit his success.

Denial of service attacks can be particularly difficult for IDS. If a saturation attack such as a SYN flood attack were launched against our web server, the NIDS will pretty quickly be able to identify what is going on. The HIDS, however, will either identify that nothing is happening (because its access to the network is jammed) or that it is being denied service. Similarly with "ping of death" attacks – if the server is vulnerable, the NIDS will detect the attack and the HIDS will report nothing because the system it was running on has completely crashed. Suppose the site is brought under a distributed denial of service attack using spoofed source and a random target address within the web server's subnet. In this case, the web server doesn't even "see" the traffic since none of the traffic is directed at it. The HIDS will record nothing of interest, but the NIDS will record a tremendous amount of traffic to the spoofed addresses. If a shim-type HIDS were installed on the server, it would detect no attack – in spite of the fact that the web site was inaccessible during the duration of the incident.

Summary and Observations

This paper has illustrated some of the cases in which various IDS approaches will yield different results of varying effectiveness. It is the belief of the author that, in order to effectively cover a network's security, a combination of tools must be used. Both HIDS and NIDS have complimentary strengths and weaknesses which, when combined, yield a very robust detection capability. For the end user wishing to install IDS, it is critical to understand the properties of the IDS technologies they plan to deploy, so that they can gain the maximum benefit from the exercise.

What does the future hold? The author believes that as IDS technologies continue to evolve, they will more closely resemble their real-world counterparts. Instead of isolated sensor units, the IDS of the future will consist of sensor units that report to master visualization consoles which are responsible for checking whether alerts from the sensors agree or correlate to likely event-chains. In the future, IDS, firewalls, VPNs, and related security technologies will all come to interoperate to a much higher degree. As IDS data becomes more trustworthy because of better coverage, firewalls and VPN administrators will be more comfortable with reacting based on the input from the IDS. The current generation of IDS (HIDS and NIDS) are quite effective already; as they continue to improve they will become the backbone of the more flexible security systems we expect to see in the not-too-distant future.

For more Information

- The IDS mailing list is hosted at: ***ids@uow.edu.au.*** To subscribe, E-mail a message reading:
  subscribe ids to majordomo@uow.edu.au
- SecurityFocus.com runs a forum on IDS technologies. See
  http://www.securityfocus.com/ids
- Various Frequently Asked Questions (FAQs) regarding IDS:
  http://www.ticm.com/kb/faq/idsfaq.html
  http://www-rnks.informatik.tu-cottbus.de/~sobirey/ids.html
- IDS mailing list archives:
  http://msgs.securepoint.com/ids
- NFR Security, Inc:
  http://www.nfr.com

About the Author

Marcus J. Ranum is the Chief Technology Officer of NFR Security, Inc. Marcus has been designing computer security solutions since he developed the first commercial Internet firewall product (the DEC SEAL) in 1989. In 1992 he developed the TIS Internet Firewall Toolkit (FWTK) an open source system for deploying free firewalls. He also designed, implemented, secured, and managed the *president@whitehouse.gov* Internet connection for The Executive Office of The President during its first year of operation. In 1995 he became a consultant and technology strategist, as well as Chief Scientist for V-One Corporation, a company developing VPN software. In 1996 he founded Network Flight Recorder, Inc, which became NFR Security in 2001. As Chief Technology Officer for NFR Security, he is responsible for designing NFR's industry-leading product suite and conceiving and directing development of new product offerings. Marcus is widely recognized in the industry as a technology visionary and a force behind the computer security industry. In 1999, The Internet Security Conference (TISC) awarded Marcus the "Clue" award for lifetime achievement in computer security and in 2001 the Information Systems Security Association (ISSA) inducted him into their Hall of Fame. Marcus is a frequent lecturer at conferences such as Interop, USENIX, SANS, and the FORTUNE E-security conference, and has written numerous articles and guest columns for magazines worldwide.