Sounding the Alarm from the Parapet: Network Intrusion Detection System Concepts

Shawn M. Jones

CS 555, Spring 2013

1 Introduction

Ever since the Morris Worm brought down 10% of the Internet in 1988 (Dekker, 1997), security has been a topic of serious consideration on the Internet. Even last year, twenty-five years later, 63% of corporations had their computer systems attacked and were unaware of the attacks until external entities informed them (Strohm, 2013). It is not enough to only protect computing resources, but also to identify when such protections have failed. This is the bailiwick of the Intrusion Detection System.

There are many types of intrusion detection systems: host-based, network-based, distributed, etc. This paper will focus on Network Intrusion Detection Systems (NIDS). It will not cover methods of *prevention*, which is an extra feature many NIDS offer, instead focusing on *alerts*, which can lead to many desired actions. The paper will cover the concept of operations and deployment of these systems, while also discussing concerns with their use and possible vulnerabilities to watch out for. Then it explores a hint at the future of these devices.

2 Concept of Operations

Fortunately for those seeking to protect their networks, most intruders have less technical skill than previous generations and now rely heavily on automated tools shared on the Internet (McHugh, Christie, and Allen, 2000). This makes the output and behavior of such tools predictable. This predictable output allows for pattern recognition to be used to identify a possible intrusion (Innella, 2001).

NIDS capture packets from the communications medium (wireless, copper, fiber, etc.), then they perform *deep packet inspection* looking at headers and payload for patterns known to be used in attacks. Such patterns are loaded at start-up from a database of *signatures* that can be rapidly matched to the packets going by. If a match occurs, action can be taken based on the match, often alerting someone to the possible intrusion. Because of this capability, NIDS can alert a network administrator to many different types of intrusions, from port scans to viruses (Kurose and Ross, 2012).

NIDS perform the capture by setting the listening interface to *promiscuous mode*. Under normal conditions, packets that are not destined for the listening host, or are otherwise unsuitable to be processed by upper layer software, are discarded and not available to an application using that interface. Under *promiscuous mode* all packets are passed to a higher layer application, regardless of quality or content (Hunt, 1998).

The switch that the packets pass through can be configured to have a *spanning port*, which duplicates all packets sent across the switch so that the NIDS can inspect them on its listening interface. Alternatively, a *network tap* can serve the same purpose. This allows the NIDS to be invisible to the network it is monitoring (Scarfone and Mell, 2007).

Alternatively, NIDS can capture packets and perform anomaly-based detection. These NIDS use machine learning to determine the normal behavior of packets on the network. Once the NIDS has a good baseline, it can then compare that model against a running network. Any traffic that is statistically different can set off an alarm for action (Kurose and Ross, 2012) (Scarfone and Mell, 2007).

Even though there is a lot of promising research in using anomaly-based detection, signature-based NIDS are the current standard in real-world usage (Gascon, Orfila, and Blasco, 2011).

3 Deployment

Depending on the complexity of the network, an organization will typically deploy NIDS at network boundaries. Each of these boundaries is also usually protected by a firewall, or at the minimum, a filtering router.

In order to provide a separation of concerns between the duties of the NIDS, the NIDS functionality can be broken up into individual nodes. The roles of these the devices can be broken up into *sensor*, *analyzer*, and *management console*. The *sensor* extracts suspicious packets from the network it monitors. The *analyzer* takes into account all of the detected behavior and can determine if an actual attack is taking place. The *management console* can then take the suggestion from the analyzer and notify network administrators (McHugh, Christie, and Allen, 2000).

Consider the typical small corporate network that contains three separate zones, as shown in Figure **??**. The first zone exists on the Internet-facing side of the network's Internet router. The second zone consists of a demilitarized zone (DMZ) that allows the world access to its Internet-facing servers (for services like web and email), but contains a firewall blocking Internet access to any internal computing resources. The third zone is the internal computing resources. This "hard, crunchy shell" protects the "soft, gooey center" of the internal network (Cheswick, 1990).

The first NIDS sensor is deployed in front of the Internet-facing router to analyze any traffic entering from the Internet into the DMZ. This allows network administrators to gauge the threats that exist for the network as a whole, seeing the possible intrusions before any of their security is in place (McHugh, Christie, and Allen, 2000).

The second NIDS sensor is deployed in front of the firewall to analyze any traffic that has made

Figure 1: Typical NIDS Deployment

it into the DMZ, but has not made it into the internal network. This allows network administrators to detect any intrusions that may be targeted at internal hosts. If a machine inside the DMZ is compromised, the NIDS can alarm network administrators if it starts behaving like a launch point for further attacks (McHugh, Christie, and Allen, 2000).

A third NIDS sensor is then deployed behind the firewall to analyze traffic that makes it to the internal network. This NIDS exists to validate that the firewall rules are working effectively (McHugh, Christie, and Allen, 2000). Through the use of these three NIDS, one can also watch an attack pass through the DMZ into the internal network, indicating that the firewall needs additional configuration.

Because of the possibility of compromise, a NIDS sensor should not actually accept connections on the network it is listening to. This means that each system must have at least two physical network connections: one for promiscuous listening in order to detect intrusions, but also a second to use to report the alerts back to a central logging server. For best security, this reporting connection should be connected to a “back network” that is not directly accessible from the network being monitored (McHugh, Christie, and Allen, 2000).

4 Concerns and Criticisms

Sometimes it is difficult to identify real attacks from incorrectly configured systems, resulting in *false positives* (Scarfone and Mell, 2007). This is why it is important to couple the knowledge of detected intrusions with knowledge of the possible intent of the source (McHugh, Christie, and Allen, 2000). For example, if one is caring for an Air Force network and they see a series of possible attacks coming from a known enemy nation, the chance is high that the attacks are legitimate. The same detected intrusion coming from an in-house 20-year veteran network administrator’s test bench is more likely to be a poor configuration.

Taking intent into account, one can configure the system to ignore the “friendly fire” from systems that are incorrectly configured, or contain some vendor-specific packet contents that resemble a known attack. As a network gets larger and larger, identifying such “friendly fire” can become difficult to manage as each new “incident” requires additional NIDS configuration (Scarfone and Mell, 2007). Such configuration must be evaluated by someone with a good understanding of how the network should operate (Rash, 2002).

In approximately 65% of new deployments, administrators tend to stick with vendor-provided signatures, not using custom signatures or anomaly detection because of the high number of false alarms from these other features (Gascon, Orfila, and Blasco, 2011).

It is important to note that NIDS that rely upon signatures cannot detect *insider attacks* because insiders do not exploit the network with known vulnerabilities; they already have the necessary access (Nazer and Selvakumar, 2011). Anomaly-based NIDS are superior in this case.

Aside from false positives and insider attacks, signature-based NIDS rely upon the patterns being known *a priori*. This means that zero-day attacks cannot be readily detected. Signatures take a lot of effort and testing to develop in order to best avoid false alarms and ensure accuracy, which can allow zero-day attacks to take hold in the interim (Gascon, Orfila, and Blasco, 2011).

If its hardware is not well planned, a NIDS sensor can become overwhelmed with traffic, dropping packets and failing to detect issues (Kurose and Ross, 2012). Some NIDS can be configured to detect high traffic and skip some traffic to compensate, but of course, can then miss an actual attack (Scarfone and Mell, 2007).

Packets with encrypted payloads cannot further be analyzed, leading to attacks that escape detection (Nazer and Selvakumar, 2011). This shortcoming can apply equally to both signature-based and anomaly-based NIDS (Scarfone and Mell, 2007).

NIDS that rely on anomaly detection can be difficult to configure and tweak, especially in well-established networks. The principal issue lay in acquiring the baseline for the network. It is especially difficult to determine at what point the traffic becomes statistically significant enough to sound an alarm (Kurose and Ross, 2012). Even more alarming is the possibility that the network is already compromised during the recording of the initial baseline (Nazer and Selvakumar, 2011).

NIDS using anomaly detection are good at detecting attacks that have unusual patterns, or occur in a small period of time, but are not as capable of detecting slow or isolated attacks, especially if they use the same ports and protocols as normal traffic (Scarfone and Mell, 2007).

5 Vulnerabilities

Especially because NIDS are a network security tool, it is necessary to discuss how they can also be vulnerable to compromise.

As noted before, NIDS can be overwhelmed, especially through a distributed denial of service (DDoS) attack (Scarfone and Mell, 2007). Of course, an organization experiencing a DDoS has likely already detected the attack through the loss of one or more services.

NIDS are susceptible to *blinding*, whereby an attacker sends spurious traffic that will cause the sensor to trigger many alerts in a short period of time. This traffic is not designed to attack any real targets, but will only keep the sensor busy while the attacker runs a real attack in parallel, distracting any analyzer (human or otherwise) from detecting the attacker's true intentions (Scarfone and Mell, 2007).

Also, just like all software they are susceptible to the bugs and other coding errors. Using specially crafted packets sent to the listening port, an attacker can crash the NIDS sensor (Skyttä, 2007), can hide data from the sensor (Konovalenko, 2008), modify its logs (Rohlf, 2007), and remotely execute code on the sensor (Mehta, 2007). This runs the full gambit of knocking the sensor offline, which can be mitigated by monitoring its status, to using the NIDS sensor as a launch point for other attacks. All of these vulnerabilities have since been fixed in Sourcefire's Snort product, but they represent the possibilities of the kinds of issues a NIDS can face if it contains a software bug. They do illustrate that even a device that just listens can be a victim and that network administrators should patch their NIDS sensor just like other tools.

Of course, using the back network prevents the sensor from being vulnerable to attacks on its other services, such as SSH and file sharing. Without such separation, the sensor can be victim to the vulnerabilities of these other services (McHugh, Christie, and Allen, 2000).

6 Research Areas and The Future of NIDS

Multiple research areas exist for NIDS, such as improving the speed of matching signatures, improving anomaly based detection, reducing false positives, and resolving issues with NIDS in different environments, such as wireless sensor networks and cloud computing. This section is by no means exhaustive.

Signature-based systems can be slow under load, delaying or failing to detect intrusions as they occur. Speeding this up is the focus of some research, looking to combine new hashing techniques and an updated finite state model to more quickly resolve whether or not a signature matches (Dixit, Gupta, and Pal, 2012).

Post-processing is being tested as a way of reducing the number of false positives. This post processing is broken into several components that weigh each alert based on statistics gathered on previous alerts and statistics known about real attacks (Spathoulas and Katsikas, 2010). Others have researched using data-mining and machine-learning as a post-processing mechanism to reduce false positives (Pietraszek and Tanner, 2005).

Anomaly-based detection is the target of a lot of research, evaluating the effectiveness of various machine learning algorithms, such as Support Vector Machines (Renjit and Shunmauganathan, 2011), Principal Component Analysis (Zargar and Baghaie, 2011), Tanimoto k-nearest neighbor classifiers (Sharma and Lal, 2011), and countless others. Even though these algorithms are not new, the research is focused on efficiency and reduction of false positives.

As noted before, anomaly-based detection can be susceptible to “poisoning attacks”. These attacks are the result of malicious behavior being detected during the learning phase as baseline behavior. The result of these attacks being that intruders can mask their behavior because it is already considered a “normal” part of the baseline. Some are developing algorithms that can limit or eliminate the effectiveness of such attacks (Long et al., 2011).

An often forgotten environment is the wireless sensor network (WSN). The conditions under which WSNs (e.g. low power, processing capability) exist provide special challenges for intrusion detection. Due to the specialized nature of these networks, their NIDS are often anomaly based and focused on defense in addition to detection (Rassam, Maarof, and Zainal, 2012).

Some research exists in solving intrusion problems related to cloud-based computing and the use of NIDS for cloud-based applications. NIDS can be used to not only identify attacks and their sources, but can change the behavior of firewalls and routers in other regions of the cloud, allowing one region's detection to inform the rest of the cloud network (Modi, et al., 2012).

7 Conclusion

NIDS can be effective in detecting attackers trying to get into a network, and are needed as part of a complete network security solution. By placing sensors at key points in the network infrastructure, attacks can be viewed as they occur and actions can be taken. These sensors can also show when existing defenses are working or failing.

However, they are not without their issues. NIDS must be configured to eliminate false alarms. Signature-based NIDS require upkeep as new attacks are being devised all of the time. Just like other software, NIDS can be susceptible to many of the same vulnerabilities as other software services. Both of these factors requires constant vigilance on the part of the administrator. In spite of these issues, they are still key to the security of a network because they indicate when an attack is taking place.

Research is helping fix some of these issues. Performance and false-positive reduction, as well as improving anomaly-based detection, is on the horizon for NIDS, providing hope for network administrators using them. Non-LAN network environments, like cloud computing and wireless sensor networks, also have NIDS challenges that are being addressed by research.

Networks are not unlike other fortresses. Centuries ago, nobility employed laborers to build great castles consisting of high walls, moats, and strong gates. In spite of these defenses, the moats could be crossed and the gates and walls could be scaled. The siege was taxing on the attacker only if the defender could see it coming. All it took was a lone astute guard, standing on the parapet, seeing the attack coming and sounding the alarm.

8 References

Cheswick, B. The Design of a Secure Internet Gateway. *Proceedings of the Summer UNIX Conference*, (1990), pp. 233-237.

Dixit, U., S. Gupta, and O. Pal. Speedy Signature Based Intrusion Detection System Using Finite State Machine and Hashing Techniques. *International Journal of Computer Science Issues (IJCSI)*, vol. 9 issue 5 (September 2012), pp. 387-391.

Dekker, M. Security of the Internet. *The Froehlich/Kent Encyclopedia of Telecommunications*, vol. 15 (1997), pp. 231-255, http://www.cert.org/encyc_article/tocencyc.html, accessed Mar 12, 2013.

Gascon, H., A. Orfila, and J. Blasco. Analysis of Update Delays in Signature-Based Network Intrusion Detection Systems. *Computers & Security*, vol. 30 (2011) pp. 613 - 624.

Hunt, C. TCP/IP Network Administration, 2nd ed. O'Reilly, 1998.

Innella, P. The Evolution of Intrusion Detection Systems. *Security Focus/Symantec Connect*, (2001), http://www.symantec.com/connect/articles/evolution-intrusion-detection-systems, accessed Mar 12, 2013.

Konovalenko, A. Bug #235901 [CVE-2008-1804] Snort IP fragment TTL evasion vulnerability. *Launchpad Bugs: Ubuntu*, (2008), https://bugs.launchpad.net/ubuntu/+source/snort/+bug/235901, accessed Mar 12, 2013.

Kurose, J. and K. Ross. Computer Networking: A Top-Down Approach, 6th ed. Pearson, 2012.

Long, J., W. Zhao, F. Zhu, and Z. Cai. Active Learning to Defend Poisoning Attack Against Semi-

Supervised Intrusion Detection Classifier. *International Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, vol. 19, supple. 1 (December 2011), pp. 93- 106.

McHugh, J., Christie, A. and J. Allen. Defending Yourself: The Role of Intrusion Detection Systems. *IEEE Software*, (September/October 2000), pp. 42-51.

Mehta, N. Snort, Sourcefire, and Nortel Threat Protection IDS/IPS DCE/RPC buffer overflow. *IBM Internet Security Systems*, (Feb 19, 2007), http://xforce.iss.net/xforce/xfdb/31275, accessed Mar 16, 2013.

Modi, N., D. Patel, A. Patel, and M. Rajarajan. Integrating Signature Apriori based Network Intrusion Detection System (NIDS) in Cloud Computing. *Procedia Technology*, vol. 6 (2012) pp. 905 - 912.

Nazer, G. and A. Selvakumar. Current Intrusion Detection Techniques in Information Technology - A Detailed Analysis. *European Journal of Scientific Research*, vol. 65, no. 4, pp. 611-624.

Pietraszek, T. and A. Tanner. Data Mining and Machine Learning - Towards Reducing False Positives in Intrusion Detection. *Information Security Technical Report*, vol. 10, no. 3 (2005), pp. 169-183.

Rash, W. Intrusion detection: Too Much Information. *ZDNet* (2002),
http://www.zdnet.com/news/intrusion-detection-too-much-information/297126,
accessed Mar 12, 2013.

Rassam, M., M. Maarof, and A. Zainal. A Survey of Intrusion Detection Schemes in Wireless Sensor Networks. *American Journal of Applied Sciences*, vol. 9 issue 10 (October 2012), pp. 1636

- 1652.

Renjit, J. and K. Shunmuganathan. Network based anomaly intrusion detection system using SVM. *Indian Journal of Science and Technology*, vol. 4 number 9 (Sep 2011), pp. 1105-1108.

Rohlf, C. Calpytix Security Advisory CX-2007-001. *Calyptix Security*, (Jan. 11, 2007), http://labs.calyptix.com/advisories/CX-2007-01.txt, accessed Mar 16, 2013.

Strohm, C. Cyber Attackers' Tactics Outpace Companies' Responses. *Bloomberg*, (2013), http://www.bloomberg.com/news/2013-03-12/cyber-attackers-tactics-outpace-companies-responses. html, accessed Mar 12, 2013.

Scarfone, K. and P. Mell. National Institute of Standards and Technology. *Guide to Intrusion Detection and Prevention Systems (IDPS)*. Gaithersburg: GPO, 2007. http://csrc.nist.gov/publications/ nistpubs/800-94/SP800-94.pdf, accessed Mar 12, 2013.

Sharma, A. and S. Lal. Tanimoto Based Similarity Measure for Intrusion Detection System. *Journal of Information Security*, vol. 2 (2011), pp. 195-201.

Spathoulas, G. and S. Katsikas. Reducing False Positives in Intrusion Detection Systems. *Computers & Security*, vol. 29 (2010) pp. 35-44.

Skyttä, V. Bug 232109 - (CVE-2007-1398) CVE-2007-1398: snort DoS. *Red Hat Bugzilla* (2007), https://bugzilla.redhat.com/show_bug.cgi?id=232109, accessed Mar 16, 2013.

Zargar, G. and T. Baghaie. Category-Based Intrusion Detection Using PCA. *Journal of Information Security*, vol. 3 (2012), pp. 259-271.