Editors: Richard Ford, rford@se.fit.edu
Michael Howard, mikehow@microsoft.com

Mixing Wheat with the Chaff

Creating Useful Test Data for IDS Evaluation

William H. Allen
Florida Institute of Technology

As the use of intrusion detection systems (IDSs) continues to climb and as researchers find more ways to detect attacks amid a vast ocean of data, the problem of testing IDS solutions has reared its ugly head. Showing that one technique is better than another or training an IDS about normal usage requires test data. As it turns out, collecting or creating such a data set is something of a catch-22. If the data already contains attacks, researchers will train the IDS to see the attacks as normal; the IDS could then fail to register them as malicious events in the future. The most efficient way, however, to determine whether a large data set contains malicious events is to scan it with an existing IDS. Thus, any attacks that the existing IDS fails to find are presented to the new IDS as normal data—leading to potential false negatives. Clearly, breaking this cycle requires an independent source of verifiable, attack-free training data with which to train IDSs.

Over the past two decades, intrusion detection approaches have evolved from simple statistical analysis to sophisticated techniques inspired by such diverse areas as immunology, expert systems, data mining, and neural networks. Researchers have applied those detection techniques to user behaviors (keystroke timing, for example, or file and program usage patterns), network traffic (including packet header fields or connection patterns), and interactions between programs and their host operating system (system calls, registry changes, or file access, for example). In all of these cases, testing and validation of new anomaly-based intrusion detection techniques requires a record of typical (that is, normal) behaviors for use in constructing statistical models, training learning-based systems, or creating rules that describe acceptable actions.

After an IDS is tuned or programmed to recognize normal behaviors, researchers must expose it to malicious behaviors to determine whether the system can detect the difference. This also requires a verifiable source of data. As noted earlier, an IDS might detect events or fail to detect anomalies (produce false negatives). If the IDS is exposed to a different source of normal data, a third possibility exists: it might detect an abnormal event where none exists (a false positive). As I explain later, this third outcome can have a significant impact on an IDS's effectiveness.

Challenges in creating training data

In addition to being attack-free, training data must accurately represent real network, user, and system activity, and it must be continually updated to reflect new protocols, applications, or changes in user behavior. For example, once an IDS has been trained to recognize the normal activities of a team of users, it will log radically different system usage patterns if that team is later assigned to a new project. Unless the IDS is retrained to these new, normal behavior patterns, security officers will waste time dealing with false alarms. Similarly, if a new network-aware application, such as a peer-to-peer workgroup collaboration tool, is installed, security officers must retrain the network IDS to accept new behaviors, even though the new behaviors might still evolve as users explore the tool's capabilities.

Another important requirement that's often difficult to satisfy is the need for the training data to realistically recreate the bandwidth or activity level of normal behaviors. If a network IDS is trained with low-bandwidth data sets, for example, and then exposed to real traffic, it might consider the increased traffic rate to be an indicator of an attack and generate thousands of false alarms. Without a realistic training data set, researchers can't thoroughly test a particular IDS approach's performance and robustness.

Privacy concerns can also inhibit the collection of training data. Some IDS techniques examine specific details within the system calls or network packets that they monitor, so sanitized training data might no longer contain cues that would

For our readers

How do you collect attack-free test data? We'd like to hear some of your experiences with intrusion-detection systems. Send your stories to Richard Ford at rford@se.fit.edu.

trigger an alert or might actually produce false alarms. A sanitization technique that replaces users' email addresses with xxxxx@somewhere.net, for example, would convert a network trace file that contains hundreds of different email addresses into one that appears to send all email messages to one destination address—an event that an IDS would register as a large-scale flooding attack on an email server. Blindly replacing email addresses with random strings can produce the opposite effect—hiding an attack's presence on a specific user or host. The best, although most time-consuming, approach is to replace each email address with a randomly generated string, reusing the same string for each email occurrence. Security officers could use similar techniques to replace login IDs, URLs, registry keys, and filenames (John_Smith_resume.doc, for instance) that contain personal information.

A major drawback to anomaly-based detection schemes is the virtual certainty that the IDS will report at least some benign events as attacks due to inadequate training or because of unanticipated, but harmless, anomalies in real-world data. However, as long as this false-alarm rate is low enough, it won't present a significant problem. For instance, a host-based IDS that generates one or two false alarms a day won't overly concern most security-conscious users; they might even accept the alerts as proof that the system is working. A constant stream of false alarms, however, would likely lead most users to turn off the IDS's alert-reporting features, preventing it from protecting the network or host system effectively. On a system-wide level, security officers must not be presented with a high number of false alarms because it's their duty to track down each alert to determine whether it's the result of malicious or benign activity. Clearly, the accuracy with which training data matches real-world activities will have a significant impact on the false-alarm rate an IDS generates when deployed. This fact wasn't widely appreciated until Stefan Axelsson demonstrated that even a small rate (1 in 10,000) of false positives could generate an unacceptable ratio of false alarms to real detections.[1]

Although dangerous if not carefully contained, one surefire way to create realistic attack behaviors is to execute live attack code.

What use are IDSs without attacks?

Besides attack-free training data, IDS researchers must be able to reliably recreate attacks under controlled circumstances so that they can determine their systems' detection accuracy. Researchers have used two main approaches to produce attack behaviors. Although dangerous if not carefully contained, one surefire way to create realistic attack behaviors is to execute live attack code. Assuming that researchers can prevent an unintentional release onto the real network, this method requires access to the original attack code or a detailed analysis of the attack's behavior so that researchers can write a program that produces the same results. Rather than execute live attack code, some researchers choose to generate synthetic network traffic that attempts to recreate attack behaviors. This approach can produce results that are useful for IDS testing, but researchers must take care to ensure that all of the original attack's meaningful characteristics are faithfully reproduced. Of course, it could be difficult to determine in advance which characteristics will prove most useful for attack detection.

Regardless of which approach is used to create attack data, researchers must merge the attacks with a realistic background of attack-free data and carefully record the location of the attacks within that data. Although researchers could mix the attacks with the same attack-free data that they used to train an IDS, a more realistic approach would be to combine the attacks with attack-free data that hasn't yet been used for training. Thus, the IDS could scan the attack-free data for false positives before the attack data is added; any alerts generated in this clean data would clearly be false alarms.

The Lincoln Lab data sets

Currently, the de facto standard for comparing IDS approaches is training data produced by the MIT Lincoln Lab for Darpa-sponsored IDS evaluations in 1998 and 1999.[2] The Lincoln Lab data sets include host audit logs and network traces from a testbed network constructed specifically for this purpose. Lincoln Lab researchers based network traffic on models created from real networks and generated by scripts executed on machines running Windows, Unix, and Linux. The 1999 evaluation, for example, provides researchers with 10 days of attack-free data, which they can use for training and determining false-alarm rates, and 15 days of attack data. The evaluation documentation details the attack occurrences so that researchers can verify IDS performance.

Although still in frequent use, the relevance of the Lincoln Lab data sets remains a controversial issue. Not long after the Lincoln Lab completed the evaluations, John McHugh published a detailed critique of the data set creation and IDS evaluation methodologies.[3] McHugh

praised the Lincoln Lab evaluators for their pioneering efforts, but his critique highlighted several issues, including limitations in the method used for generating background traffic, problems with the classification and range of attack types, and questions about the validity of the techniques used for the result comparison. A more recent analysis[4] points out that large portions of Lincoln Lab background traffic fail to match a real network's traffic characteristics and noted that the set of attacks used in the Lincoln Lab data is less representative of the attacks found in today's networks than it was almost a decade ago. (See the Education department on p. 48 for more on Lincoln Lab data sets.)

Interesting new approaches to data generation

The need for more modern data sets has lead many researchers to consider ways to generate test data for evaluating their own IDS designs. The Lincoln Lab evaluators extended their research in IDS evaluation and created Lariat, the Lincoln Adaptable Real-time Information Assurance Testbed.[5] This system can generate realistic background traffic and real-time attacks and provides a graphical interface for configuration and monitoring. Giovanni Vigna and his colleagues use a novel approach to test an IDS by creating variations (which they call mutations) of an existing, detectable attack to determine which minor differences will escape IDS detection.[6] Recognizing that different IDS techniques examine network traffic at different levels of abstraction, another approach by Spyros Antonatos and his colleagues can generate background traffic with different detail levels.[7] Researchers can then simulate application protocols session by session or synthetically produce overall traffic patterns from statistical models.

A recent study[4] discusses possible problems with the prevailing anomaly-detection paradigm and suggests that reevaluating assumptions, such as the idea that all anomalous behaviors are malicious or that attack-free data will always be available to test an IDS, might lead to significant improvements in detecting malicious activity. The study's authors also discuss several factors that could improve IDS training data accuracy.

Successful development and testing of intrusion detection techniques requires a widely available source of robust and accurate training data. Fair comparisons between IDS approaches will depend on the research community's acceptance of that training data. New approaches for the generation of attack-free and labeled attack data are emerging and can potentially provide IDS researchers with a more accurate means of improving their designs and comparing them with the work of others. However, the developers of new methods for creating training data must share their methodology and their data sets to foster better cooperation and allow meaningful comparisons. □

References

1. S Axelsson, "The Base-Rate Fallacy and Its Implications for the Difficulty of Intrusion Detection," *Proc. 6th ACM Conf. Computer and Comm. Security* (CCS 99), ACM Press, 1999, pp. 1–7.
2. R. Lippmann et al., "The 1999 Darpa Off-Line Intrusion Detection Evaluation," *Computer Networks: The Int'l J. Computer and Telecommunications Networking*, vol. 34, no. 4, 2000, pp. 579–595.
3. J. McHugh, "Testing Intrusion Detection Systems: A Critique of the 1998 and 1999 DARPA Intrusion Detection System Evaluations as Performed by Lincoln Laboratory," *ACM Trans. Information and Systems Security*, vol. 3, no. 4, 2000, pp. 262–294.
4. C. Gates and C. Taylor, "Challenging the Anomaly Detection Paradigm: A Provocative Discussion," *Proc. Workshop on New Security Paradigms* (NSPW 06), ACM Press, 2006.
5. L.M. Rossey et al., "Lariat: Lincoln Adaptable Real-Time Information Assurance Testbed," *Proc. IEEE Aerospace Conf.*, IEEE CS Press, 2002, pp. 6-2671–6-2682.
6. G. Vigna, W. Robertson, and D. Balzarotti, "Testing Network-Based Intrusion Detection Signatures Using Mutant Exploits," *Proc. 11th ACM Conf. Computer and Comm. Security* (CCS 04), ACM Press, 2004, pp. 21–30.
7. S. Antonatos, K.G. Anagnostakis, and E.P. Markatos, "Generating Realistic Workloads for Network Intrusion Detection Systems," *Proc. 4th Int'l Workshop on Software and Performance* (WOSP 04), vol. 29, no. 1, ACM Press, 2004, pp. 207–215.

William H. Allen *is an assistant professor of computer science at the Florida Institute of Technology. His research interests include computer and network security, modeling and simulation of network-based attacks, and computer forensics. Allen has a PhD in computer science from the University of Central Florida. He is a member of the IEEE, the ACM, and Usenix. Contact him at wallen@fit.edu.*

The need for more modern data sets has lead many researchers to consider ways to generate test data for evaluating their own IDS designs.