Statistical Problems with Statistical-based Intrusion Detection

L. Todd Heberlein

Version 1.0

In 1987 Dorothy Denning wrote in her seminal paper "An Intrusion-Detection Model" the following words: "exploitation of a system's vulnerabilities involves abnormal use of the system; therefore, security violations could be detected from abnormal patterns of usage." With these words, Denning kicked off 20 years of research, development, and publications in anomaly-based intrusion detection, where systems build statistical profiles of normal usage patterns and detect variations from those profiles. Unfortunately, the statistics behind statistical-based detection can lead to some unintuitive results, from surprisingly high numbers of false alarms to the potential of making a site less secure. This paper highlights some of the problems, causes, and implications anomaly-based detection.

1 Introduction

Intrusion detection techniques generally fall into two categories: signature-based detection and anomaly-based detection. Signature-based detection typically looks for a specific types of activity considered suspicious, often by applying a combination of pattern matching (e.g., string matching) and rules. For example, a signature might look for more than 512 bytes sent to the specific server (call it "foobar") indicating an attempt to exploit a known buffer overflow vulnerability. Anomaly-based detection, on the other hand, uses statistical techniques to learn the behavior of a subject and then looks for deviations, or anomalies, in behavior. For example, an anomaly-based detector might build a profile that says a client sends to the "foobar" server an average 50 bytes with a standard deviation of 10 bytes; it might then generate an alert for any data transmissions to the server that exceeds 4 standard deviations from the average.

The concept of using anomaly detection as a means to detect security violations generally traces its roots to Dorothy Denning's paper "An Intrusion-Detection Model", and the paper succinctly explains its rationale:

> *"The model is based on the hypothesis that exploitation of a system's vulnerabilities involves abnormal use of the system; therefore, security violations could be detected from abnormal patterns of system usage."* [1]

Because anomaly-based detection does not use a priori knowledge of specific vulnerabilities or attacks, one of its primary strengths over signature-based detection, and what is often cited as *the* primary strength over signature-based detection, is its ability to detect novel, previously unknown attacks [2-12]. Unfortunately, the number of novel, previously unknown attacks a typical site might experience in a year is extremely small, while the amount of low-level data used to discover these attacks (e.g., system calls or network packets) is huge. This disparity in data distribution, very few attacks in huge numbers of non-attack events, makes it easy to accidentally misrepresent or misinterpret statistics and their implications for a given detection technique.

The goals of this paper are to illustrate the problems, explain why they exist, and highlight some of the implications of these problems. We hope not to dissuade continuing work in anomaly-based detection but to highlight the challenges. We also hope that by highlighting, and hopefully clarifying, the challenges, organizational managers, users, program managers, and researchers of anomaly-based detection can more effectively communicate how a given technique is expected to operate in operational environments.

2 The Missing Information: Data Distribution

Researchers usually describe the performance of an intrusion detection systems or technique in terms of false positive and false negative rates, also known as Type I and Type II errors. For the field of intrusion detection, a false positive error, also called a Type I error, occurs when the system misidentifies a non-attack event as an attack; that is, the system generates a false alarm. The false positive *rate* is the *percent* of non-attack events that are identified as attacks. A false negative error, also called a Type II error, occurs when the system misidentifies an attack event as a non-attack event; in other words, it misses the attack. A false negative *rate* is the *percent* of attacks that are missed. Sometimes researchers use the term "detection rate" to describe the percentage of attacks that were detected[1]. This is simply one minus the false negative rate.

Unfortunately, these two data points are insufficient to understanding how a system will actually perform. What a user wants to know (or should want to know) is: Given an attack report, what is the probability that it actually represents an attack? In other words, of the reports the user actually receives, what is the true positive rate? This question cannot be answered with the numbers provided in many publications, presentations, or promotional material.

The missing piece of data is the underlying distribution of the data. That is, of all the events analyzed, what percent represent the attacks the system is looking for. Adding in this information can lead to surprising results. Section 2.1 introduces the surprising results once this data distribution information is factored in. Section 2.2 shows how to use the data distribution information to generate false positive rate requirements. Section 2.3 shows how easy it is to mischaracterize the actual performance of two systems when only using detection rates and false positive rates.

2.1 The False Positive Paradox

Anomaly detection faces two important and related problems. The first is the issue of the underlying distribution of the data, and this leads to huge numbers of false positives. Academic publications will tout their approach by promoting their high detection rates and low false positive rates, but in real world operations these numbers quickly become meaningless. Consider the following hypothetical example. A given system and environment has the following qualities:

- $\Pr\left(R \mid A\right) = 95\%$ This is the probability of the system generating a report given that there is an attack. In other words, this is the detection rate.
- $\Pr\left(R \mid \overline{A}\right) = 0.5\%$ This is the probability that the system generates a report given the at there was not an attack. In other words, this is the false positive rate.

On face value, such a system appears to be really good. It detects new attacks 95% of the time and only generates a false positive 0.5% of the time. Its Receiver Operating Characteristic (ROC) curve (a common presentation format in academic publications) would look fantastic. But the real question that needs to be asked is: *If the system reports an attack, what is the probability*

[1] In the intrusion detection community, "detection rate" seems to be a preferred over "false negative rate."

that there is really an attack. In other words, we want to know $\Pr(A \mid R)$. To answer this, we still need one more piece of information – the underlying distribution of the data. That is, how many attack and non-attack events are there. Suppose the answer is:

- $\Pr(A) = 0.001\%$ This is the probability that an event is a new attack that the network administrator needs to pay attention to. In other words, the vast majority of activity on the network (99.999%) is benign (users visiting web sites, chatting, downloading/uploading content, etc.) or is already handled by simple prevention mechanisms (firewalls, patches, etc.), and only a few events need to rise to the level of an administrator's attention. The probability of an event not being an attack, $\Pr(\overline{A})$, is simply $1 - \Pr(A)$.

  Now we can apply Bayes' theorem to find the answer to $\Pr(A \mid R)$:

$$\Pr(A \mid R) = \frac{\Pr(R \mid A)\Pr(A)}{\Pr(R \mid A)\Pr(A) + \Pr(R \mid \overline{A})\Pr(\overline{A})} = 0.002$$

Figure 1: False Positive Problem

As the equation shows, given a report that there is an attack, the probability that it is actually an attack is only 0.2%. In other words, 99.8% of the time the detection system is wrong! The reason behind this apparent paradox is the underlying distribution of the data – there is just a huge amount of activity that doesn't need the attention of an administrator, and this volume simply swamps what on face value appears to be a really good intrusion detection system.

This paradox, which has been identified in other security fields [13], is incredibly important to a user of such a system. If the user is told that the system has a detect rate of 95% and a false alarm rate of only 0.5% she may have too much confidence in the system and overreact to a report, perhaps falsely accusing someone of malicious deeds or cutting off legitimate network activity. If such a system is integrated into an automated response system, the results could be disastrous. If, on the other hand, she is told that 499 of every 500 reports is wrong, then she may respond much more cautiously to a report.

Reporting just detection rates and false positive rates without including estimates of the underlying data distribution can lead to gross misinterpretation of the expected performance of the system.

2.2 Generating False Alarm Requirements

Lets take the same problem and look at it from another perspective. Suppose the organization specifies the number of false positive reports it is willing to tolerate for each real attack report, and they then want to search for a system that has a false positive rate that meets their requirements. If the organization specifies $\Pr(A \mid R)$, what is the value for $\Pr(R \mid \overline{A})$, the false positive rate, that is needed to meet this requirement? A little algebraic manipulation on Baye's theorem gives the answer:

$$\Pr(R \mid \overline{A}) = \Pr(R \mid A) \times \frac{\Pr(A)}{\Pr(\overline{A})} \times \left[\frac{1}{\Pr(A \mid R)} - 1\right]$$

Figure 2: Defining False Positive Requirement

This equation can be simplified with the following assumptions. First, given that most events are not attacks, the argument $\Pr(\overline{A})$ can be approximated by 1, so that term can be removed from the denominator. Second, if the probability that a report is actually an attack,

$\Pr(A \mid R)$, can be expressed in terms of 1 out of a N (e.g., 1 out of 100 reports is actually an attack), then the third term can be rewritten as $[N-1]$. This gives us the simplified equation:

$$\Pr(R \mid \overline{A}) = \Pr(R \mid A) \times \Pr(A) \times [N-1]$$

Figure 3: Simplified Requirement Equation

This equation has three elements:

- $\Pr(R \mid A)$ – this is the detection rate; that is, given that there is an attack, what is the probability that the system will generate a report.
- $\Pr(A)$ – this is the underlying distribution of the data; that is, what is the probability that any particular event is an attack.
- $N-1$ – this represents the number of false positive reports you are willing to tolerate for each true positive report. For example, if you are willing to tolerate only 1 true alert for every 1000 alerts, N=1000, so $N-1$ represents the 999 false alarms are willing to tolerate.

Populating this equation with reasonably true numbers illustrate the magnitude of the problem. My Mac OS 10.4.8 workstation generates approximately 5 million BSM audit records in a typical day. For many analyses, each system call represents a single event [10, 14], so for a single host, there would be approximately 5 million events to evaluate each day. Consider a moderate sized department with 200 workstations, and suppose they typically experience a previously unknown attack (i.e., a novel attack not detected by signature-based IDS) about four times a year (or roughly 1 novel attack every 3 months). Finally, assume the organizations wants to be able to detect these previously unknown attacks 85% of the time, and they are willing to tolerate 999 false alarms for every real attack reported. What would be maximum allowed false alarm rate, $\Pr(R \mid \overline{A})$, for a system that would meet these requirements?

$$\Pr(R \mid \overline{A}) = 0.85 \times \left(1.1 \times 10^{-11}\right) \times 999 = 9.3x10^{-9}$$
$$= 0.0000000093$$

Figure 4: Maximum False Positive Rate Allowed to Satisfy Organization

This false alarm rate, 0.0000000093, is probably unrealistic for an operational system. Even if the organization is willing to drop their detection rate all the way down to 8.5% or increase the number of false alarms to 9,999 for every real attack, this would only change the false alarm rate to 0.000000093 – still a completely unrealistic number.

The problem is that the equation is so strongly dominated by the term for the probability that any particular event would be part of an attack, $\Pr(A) = 1.1 \times 10^{-11}$. This factor is driven by two facts: (1) there are such a huge number of low level events not associated with an attack (5 million system calls per host per day), and (2) novel attacks for any given site are extremely rare.

Besides identifying the challenge that an intrusion detection system targeted at novel attacks must face, it highlights the amount of testing of the intrusion detection system required to provide reasonable estimates of such low false alarm rates. To validate that an intrusion detection system has a false positive rate on the order of 10^{-8} requires more than testing on a few million or few tens of millions of audit records.

2.3 Greater False Alarm Rates Can Mean Better Performance

Suppose company **A** has an intrusion detection system that they claim, for an 85% detection rate, has a false alarm rate of 0.1%. Then company **B** releases an intrusion detection systems that they claim, for the same 85% detection rate, has a false alarm rate of only 0.001%, one hundred time better than company **A**'s system. Obviously company **B** has the superior product, or do they?

The problem is that these numbers are incomplete. They do not reflect the events that are measured and their underlying distribution. For example, suppose company **A**'s system measures the sequence of program executions (i.e., this is the event by which they determined the false alarm rate), whereas company **B**'s system measures the sequence of system calls (which is the event by which they measured their false alarm rate). If the average program execution generates 200 system calls, and the chance that a site will experience an attack is independent of the intrusion detection system deployed, we can compare the operational performance of the two system. Again, assuming novel attacks are still rare (i.e., 1 in 50,000 program executions), we can use Baye's theorem to calculate the probability that upon receiving a report from each system that it actually represents a real attack:

	System A	**System B**
$\Pr(R \mid A)$	0.85	0.85
$\Pr(R \mid \overline{A})$	0.001	0.00001
$\Pr(A)$	0.00002	0.0000001
$\Pr(A \mid R)$	0.0167	0.00843

Table 1: Performance of Two Systems

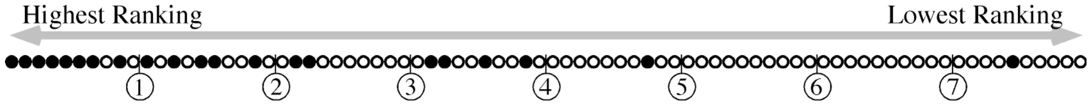
As the table shows, despite company **A**'s system having a false positive rate 100 times smaller than that of company **B**'s, the probability that a received report from company **B**'s system is actually an attack is only half that company **A**'s. In other words, while company **B**'s system's false positive rate appears to be 100 times better than company **A**'s system, its actual performance is only half as good as company **A**'s. The problem is that company **B**'s system analyzes a much larger number of small events, so that even though its false positive rate appears really good, the underlying data distribution conspires against it.

This simple example illustrates two important points. First, when comparing the purported performance of two different techniques or systems, careful attention must be given to what data sources each approach is measuring and what each approach considers an event. Otherwise, the statistics can easily be misleading. Second, by choosing to measure a data source with much fewer events, a system can experience much higher false positive rates yet still perform much better.

3 Receiver Operator Characteristics Curves

Intrusion detection presentations and publications frequently use receiver operating characteristic (ROC) curves to summarize the efficacy of a particular approach. ROC curves are used in many fields to characterize the relationship between detection rates and false alarm rates [15], some examples in intrusion detection literature can be found in [2, 4, 10, 12, 14, 16, 17]. Unfortunately, ROC curves can easily be misleading.

Figure 5 illustrates the basics of ROC curves. Across the top are a list of simulated events (represented by circles) that were ranked by some sensor from highest to lowest – in this case "highest" means the most likely to be an attack. Through some verification (e.g., human verification), all the actual attacks are labeled as black circles. The ROC curve on the right summarizes the sensor's performance in correctly identifying attacks at different levels of sensitivity. The horizontal axis represents, for a given sensitivity level, the percent of non-attacks that were identified by the sensor (the percent is represented as the fraction from 0 to 1), and the vertical axis represents, for the same sensitivity level, the percent of attacks that were identified (again, as a fraction from 0 to 1). The ideal position for a sensor would be in the upper-left corner: 100% attack detection and 0% false alarms. The red curve represents the full curve for 80 sensitivity levels possible for the simulated events. For illustration purposes, we have also highlighted seven sensitivity levels, showing where they lie on the list of the attack events, showing a table of their false-positive and true-positive rates, and where these values lie on the ROC curve (the black triangles). For example, at the sensitivity level labeled (1), 8 attacks are selected (38%) and 2 false positives are selected (3%).



● Attack Event

○ Non-attack Event

Label	False Positive	True Positive
1	0.03	0.38
2	0.12	0.62
3	0.25	0.71
4	0.36	0.90
5	0.51	0.95
6	0.68	0.95
7	0.85	0.95

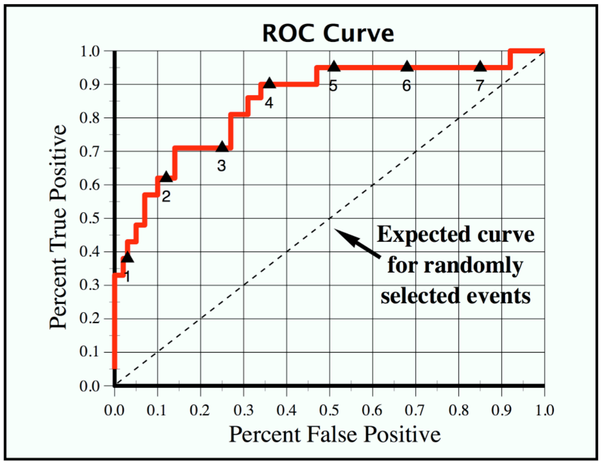


Figure 5: ROC Curve Overview

The first problem with using ROC curves for evaluating intrusion detection sensor performance is that many systems do not rank their events; the sensor simply indicates that it believes that an event is an attack or not an attack [17]. This means that there is only a single data point represented by a pair of numbers: the percent of detects and the percent of false alarms identified by the intrusion detection sensor as attacks. The problem is that a graphical summary of data (i.e., the ROC curve) should, among other things, "present many numbers in a small space" and "make large data sets coherent" [18]. Using a graphic to summarize a single data point is a missing the point of using a graphic. Using a ROC curve is also misleading in that one of the primary purposes of the ROC curve is to show the tradeoffs between detection rate and false alarm rate, and drawing a line from the lower left corner of the ROC graph (0% detect, 0% false alarm) to the single data point (the single reported detect and false alarm rate) seems to imply that the user can tradeoff detect and false alarm rates by moving up and down the curve (which is just a straight line). However, if the a report only identifies a single data point, this may indicate that there isn't a tunable sensitivity level, so a tradeoff cannot actually be made. Or. at best there may be a tunable sensitivity level but the report does not identify the behavior of the ROC curve as it is adjusted. In either case, the ROC curve with only a single data point is misleading.

The second problem is more serious and is related to the underlying distribution of the data discussed earlier. As discussed previously, for very rare events (e.g., a novel attack observed in system call data) acceptable false positive rates must be extremely small (e.g., perhaps on the order of 10^{-3} to 10^{-8}), so the operationally *relevant* portion of a ROC chart may span only a very tiny fraction of the left side of the chart. For such data distributions, to show a chart that spans the entire spectrum of false positives (from 0.0. to 1.0) as shown in Figure 5 is useless at best and misleading at worst.

Some publications do focus their ROC charts on this small portion of false positives (e.g., [10]), but others do not. When ROC charts are used, estimations of the underlying data distributions should be discussed, the ROC charts should focus on relevant portions of the data, and the operational implications of the values should be explained (e.g., "Based on our estimates of data distribution, a false positive rate of 0.001 implies that the system will generate 200 false alarms for every true attack report.").

4 Cost of a False Alarm

Many recent publications on anomaly detection evaluate the performance of their systems using labeled data (e.g., Lincoln Labs intrusion detection evaluation data). While having ground truth on a set of data (even if the data is synthesized) in order to evaluate a particulate detection technique is important, it masks a very important problem: in operational settings the cost of determining if a report is a true or false positive/alarm can be very expensive and can vary greatly between techniques.

For signature-based techniques, the cost to determine if a report is of concern to the administrator can be relatively small because by its nature the signature carries considerable amount of semantic information with it that can be integrated with information from other sources. For example, the signature can include information describing the attack it is searching for and the vulnerability that the attack is exploiting (e.g., a CVE identifier), so a report can easily be integrated with vulnerability scanner database and databases containing links to patches [19-21]. Thus, an alert from a signature component can quickly tell you (1) what the attack is, (2) what the vulnerability the attack is attempting to exploit, (3) whether the target is vulnerable, and (4) where to go to patch the system if it is vulnerable.

For anomaly-based detection systems, however, the identifying the underlying cause of an unusual event can often require extensive analysis by network and system administrators, and often the underlying cause for the anomalous event is left unresolved [22-25]. Much of the problem stems from the lack of semantic information from the alert. To illustrate this problem, imagine a perfect anomaly detection sensor that only reports real attacks (i.e., no false alarms), and it is monitoring a relatively new application or operating system that over the life of the software may have hundreds of vulnerabilities and exploits (e.g., Microsoft Internet Explorer 7 or Vista). When the anomaly detection sensor generates a report, which of the hundreds of potential vulnerabilities is being attacked? Is the anomalous response by the software indicative of a successful attack or a failed attack (e.g., the system has been patched, but the error handling still takes an anomalous path through the code)? Now consider that a moderately sized computer network may have dozens or hundreds of applications (clients, servers, stand-alone applications, local daemons, etc.), all of which may be the victim of an attack. Does the local network or system administrator have the necessary depth of knowledge in each of these applications in order to diagnose the underlying cause of anomalous behavior in any one of these applications? How much time will he need to invest to acquire that depth of knowledge? Now add in the fact that the vast majority of anomalous reports may be benign anomalies – anomalous events that are not attacks. How much time should a network or system administrator invest in investigating an anomaly report in an application for which he knows very little about, especially considering he knows it will in all likelihood be a false alarm?

Thus, the virtue provided by anomaly-based sensors of detecting all future attacks is also a burden in that it requires the network or system administrator to determine which, if any, of the future attacks might be occurring. The vagaries inherent in anomaly reports can dramatically increase the cost of processing each report.

Thus, comparing the true and false positive statistics between different techniques using labeled data, while important, is insufficient to understanding the relative costs between the techniques when deployed in operational environments where reports do not come with the aid of labels.

5 Opportunity Costs

With rare exceptions, most people run intrusion detection systems not for the purpose of catching attacks but as part of an effort to provide a secure and useful environment for their users. With that in mind, it is important to remember that a human processing an alert from an intrusion detection sensor incurs an opportunity cost. The Shorter Oxford English Dictionary defines the term as follows [26]:

Opportunity Cost – The loss of other alternatives when one alternative is chosen.

For our purposes, the "alternatives" given up in order to process an alert from an intrusion detection system include all the other actions the person could do to help provide a more secure and functional environment for her users. Some examples include:

- checking that patches are properly installed and systems are properly configured;
- taking additional training courses on operating systems, routers, firewalls, server applications, etc. so that she is more likely to configure the components in a secure manner;
- removing accounts for users who have left the organization;
- performing source code reviews on locally developed code (e.g., web server applications);
- making sure the locations of all information equipment (machines, removed disks, USB drives) are known, and making sure information on portable equipment (e.g., laptops) is encrypted in case it is stolen;
- developing security training materials for her users;
- training her users in good security practices (e.g., techniques for choosing and managing passwords).

As discussed in previous sections, statistical-based detection techniques that are looking for rare, novel attacks in large quantities of data will probably produce large numbers of false alarms, and investigating the underlying cause of each of these false alarms can be a very time-consuming process. The cumulative time spent processing these alerts must be considered in terms of what was not done during this time. Ironically, if too much time is spent chasing down the underlying cause of false alarms at the cost of not applying good preventative security measures, an intrusion detection system could inadvertently make a site less secure.

6 Conclusions

Intrusion detection techniques are generally divided into two major classes: signature-based detection and anomaly-based detection. Whereas signature-based detection looks for specific types of activity that is presumed to be malicious, anomaly-based detection uses statistical techniques to learn the normal behavior of some subject and reports when the subject is behaving abnormally. The presumption, originally articulated in [1] is that when a subject is behaving maliciously, it is also behaving anomalously; therefore, detecting anomalous behavior should identify malicious behavior.

A primary benefit of anomalous-based detection over signature-based detection is that anomalous-based detection does not need to know about a particular attack or malicious behavior ahead of time in order to detect it, so it can detect novel attacks. However, for any given site the probability that they are the victim of a previously unknown attack is very small, and the data sources monitored to detect these attacks can produce prodigious amounts of data. This paper covers the implications of these facts.

First we discussed the false positive paradox. Despite what appears to be great detection and false alarm rate, an anomaly-based detector may still generate mostly false positive reports. The reason for this is a highly skewed data distribution – novel attacks are very rare, and the data to detect the attacks generate large amounts of data. Next, we identified how to determine the maximum false positive rate that would satisfy a required false alarm report rate, and we populated the equation with some potentially realistic numbers. The result was a maximum false positive rate on the order of 10^{-8}, a staggeringly small number. We also looked at an intrusion detection system with a much better false positive rate than another system, but it actually performs much worse. We then looked at the implications of these numbers when using ROC curves for summarizing the performance of an intrusion detection system; we are concerned this graphing technique is often misused. Next, we examined the cost of processing an alert generated by an anomalous-based detection sensor, and finally, we looked at the opportunity cost of processing these alerts. When viewing the cost of processing false alarms in terms of opportunity costs, one conclusion is that running intrusion detection systems may decrease the security of a site.

Clearly there are other concerns for evaluating the potential efficacy of a particular technique in a particular environment; for example, [14] discusses how the entropy of the underlying data source can affect the detection and false positive rates. However, the importance of the topics covered in this paper are often overlooked, and researchers, program managers, potential customers should cognizant of these issues when presenting or consuming information about intrusion detection performance.

7 References

[1] D. E. Denning, "An Intrusion-Detection Model," *Software Engineering, IEEE Transactions on,* vol. SE-13, pp. 222-232, 1987.

[2] M. Al-Subaie and M. Zulkernine, "Efficacy of Hidden Markov Models Over Neural Networks in Anomaly Intrusion Detection," in *30th Annual International Computer Software and Applications Conference, 2006.*, 2006, pp. 325-332.

[3] J. M. Estevez-Tapiador, P. Garcia-Teodoro, and J. E. Diaz-Verdejo, "Stochastic protocol modeling for anomaly based network intrusion detection," in *First IEEE International Workshop on Information Assurance, 2003.*, 2003, pp. 3-12.

[4] A. K. Ghosh, A. Schwartzbard, and M. Schatz, "Learning Program Behavior Profiles for Intrusion Detection," in *Proceedings of the 1st Workshop on Intrusion Detection and Network Monitoring* Santa Clara, CA, USA: USENIX, 1999.

[5] X. A. Hoang and J. Hu, "An efficient hidden Markov model training scheme for anomaly intrusion detection of server applications based on system calls," in *12th IEEE International Conference on Networks, 2004.*, 2004, pp. 470-474 vol.2.

[6] S. Jha, K. Tan, and R. A. Maxion, "Markov chains, classifiers, and intrusion detection," in *Proceedings of the 14th IEEE Computer Security Foundations Workshop*, 2001, pp. 206-219.

[7] R. Lippmann, J. W. Haines, D. J. Fried, J. Korba, and K. Das, "The 1999 DARPA Off-Line Intrusion Detection Evaluation," *Computer Networks,* vol. 34, pp. 579-595, 2000.

[8] Y. Qiao, X. W. Xin, Y. Bin, and S. Ge, "Anomaly intrusion detection method based on HMM," *IEEE Electronics Letters,* vol. 38, pp. 663-664, 2002.

[9] M. Stillerman, C. Marceau, and M. Stillman, "Intrusion Detection for Distributed Applications," *Communcations of the ACM,* vol. 47, pp. 62-69, July 1999.

[10] C. Warrender, S. Forrest, and B. Pearlmutter, "Detecting intrusions using system calls: alternative data models," in *Proceedings of the 1999 IEEE Symposium on Security and Privacy, 1999.*, 1999, pp. 133-145.

[11] N. Ye, "A Markov Chain Model of Temporal Behavior for Anomaly Detection," in *Proceedings of the 2000 IEEE Workshop on Information Assurance and Security* United States Military Academy, West Point, NY, 2000.

[12] N. Ye, Y. Zhang, and C. M. Borror, "Robustness of the Markov-chain model for cyber-attack detection," *IEEE Transactions on Reliability,* vol. 53, pp. 116-123, 2004.

[13] B. Schneier, "Crypto-Gram: September 30, 2001," 2001.

[14] R. A. Maxion and K. M. C. Tan, "Benchmarking anomaly-based detection systems," in *Proceedings International Conference on Dependable Systems and Networks, 2000.*, 2000, pp. 623-630.

[15] I. H. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*: Morgan Kaufmann, 2000.

[16] R. Durst, T. Champion, B. Witten, E. Miller, and L. Spagnuolo, "Testing and Evaluating Computer Intrusion Detection Systems," *Communications of the ACM,* vol. 42, pp. 53-61, July 1999.

[17] R. P. Lippmann, D. J. Fried, I. Graf, J. W. Haines, K. R. Kendall, D. McClung, D. Weber, S. E. Webster, D. Wyschogrod, R. K. Cunningham, and M. A. Zissman, "Evaluating intrusion detection systems: the 1998 DARPA off-line intrusion detection evaluation," in *Proceedings of the DARPA Information Survivability Conference and Exposition*, 2000, pp. 12-26 vol.2.

[18] E. R. Tufte, *The Visual Display of Quantitative Information*. Cheshire, CT: Graphics Press, 1999.

[19] R. E. Gleichauf, W. A. Randall, D. M. Teal, S. V. Waddell, and K. J. Ziese, "Method and system for adaptive network security using network vulnerability assessment," Patent 6301668, USA, Cisco Technology, Inc., 1998.

[20] L. T. Heberlein, "Before Applying New Technologies," Net Squared, Inc., Davis, CA TR-2001-05, April 2001.

[21] P. Mell, "National Vulnerability Databaser: a comprehensive cyber vulnerability resource," National Institute of Standards and Technology, 2007.

[22] F. E. Feather, "Fault Detection in Ethernet Network via Anomaly Detection," in *Electrical and Computer Engineering*. vol. Ph.D. Pittsburgh, PA: Carnegie Mellon University, 1992.

[23] L. T. Heberlein, "Why Anomaly Detection Sucks," Net Squared, Inc., Davis, CA TR-2005-02-01, Feb 8 2005.

[24] L. T. Heberlein, G. V. Dias, K. N. Levitt, B. Mukherjee, J. Wood, and D. Wolber, "A network security monitor," in *Proceedings of the 1990 IEEE Computer Society Symposium on Research in Security and Privacy* Oakland, CA, 1990, pp. 296-304.

[25] R. A. Maxion and F. E. Feather, "A case study of Ethernet anomalies in a distributed computing environment," *IEEE Transactions on Reliability,* vol. 39, pp. 433-443, 1990.

[26] L. Brown, Ed., "Opportunity Cost," in *Shorter Oxford English Dictionary*, fifth ed. vol. 2: Oxford University Press, 2002.