Using importance sampling to compute p-values in computer intrusion detection

Daniel Q. Naiman
Department of Mathematical Sciences
Johns Hopkins University
Baltimore, MD 21218 USA
daniel.naiman@jhu.edu

1. Introduction

As dramatic advances in information technology continue to play an ever-increasing role in world communications and commerce, the problem of protecting computer and network infrastructures from unwanted use has become recognized as a critical one. Since protection, or equivalently, *detection,* is achieved through extensive data mining and pattern recognition, the problem is highly statistical in nature. Statisticians have been contributing and will continue to contribute in a meaningful way to the development of successful approaches to solving this problem.

There are two related but distinct approaches to computer intrusion detection: *misuse detection* and *anomaly detection.* In misuse detection, known methods for hacking into computers are studied and categorized and characteristic patterns of these methods are determined and stored. A detector is devised that monitors computer activity and attempts to recognize characteristics of intrusion immersed in data representing normal activity.

On the other hand, *anomaly detection* involves the collection of data representing normal activity (without misuse) and the development of statistical models describing typical patterns of variation in this activity. A detector then is devised based on monitoring the system for general classes of deviations from normal behavior.

Thus, a critical difference between the two approaches is in the degree to which known misuse information is relied upon. One would expect that implementation of a misuse detectors would tend to lead to better methods for known types of attacks, while anomaly detectors would be more likely to prove useful for newly devised attacks. It is likely that the most successful intrusion detection systems will involve the conjunction of both methods.

The subject of this paper is anomaly detection, multiple comparisons and importance sampling. As is true for so many areas of current statistical investigation, in computer intrusion detection

- huge data sets are available,
- vastly efficient techniques are used to mine extensively for patterns, and

- evaluation and setting of error rates is important.

Anomaly detection provides us with a modern application framework in which to develop methodology for answering the multiple comparisons question; "How does setting familywise error rates for statistical detection procedures in the context of search?" Beyond the application, we are motivated to determine whether existing multiple comparisons methods are useful in this context. If they are not, we would like to find how can they be enhanced so as to be made useful?

2. Experiments with anomaly detection

We describe some experiments carried out by ourselves and others to form the basis for new anomaly detection tools. These experiments fall into 3 categories, based on the character or effect of misuse that one is attempting to detect:

- user masquerading at a computer terminal,
- exploitation of vulnerabilities in a network server, and
- corruption of program code.

The first experiments are aimed at the developing tools for detecting the presence of an illegitimate user masquerading as a legitimate one. To this end, experiments are conducted to in order to characterize the variation in a user's use of the mouse and keyboard.

When a user types at the keyboard, what useful kinds of variations can be described? Keyboard event data define a *marked point process* that can be mined to determine the most informative ways to characterize individual users. Even the point process defined for an individual key (e.g. the backspace key) is of interest for characterizing the manner in which individual uses the keyboard. Variations in typing speed, lengths of time between keyboard events, degree of use of certain key combinations, speed of typing for certain key combinations, and frequencies of certain words typed are all candidates for statistical investigation.

Mouse usage can be recorded and timed as well, and this leads to another interesting data structure to investigate. The position of the cursor on the computer screen can be collected over time for an individual, and the result is a 2-dimensional time series. Simple operations, such as moving the cursor from one window position on the computer screen to a particular window's corner to iconify, can be recorded and characteristics of the motion (speed, acceleration, curvature) can be studied. We wonder if information about the manner in which users carry out the simple act of iconification is sufficient to distinguish them.

Ultimately, the information gathered in these experiments would enable us to develop a detector that could be trained to gather information on an individual's usage, and respond to sufficient departures from this.

A second type of misuse arises when vulnerabilities in a network service (a web server for example based on the HTTP protocol) are exploited, resulting in the undesirable execution of code by the server. In our experiments, an HTTP server is monitored for a period of time as it responds to incoming network requests. All of the commands that are executed by the server are recorded using standard tools for tracing a system process. The executed commands are then grouped according to the process threads that produced them. Each process thread can be viewed as series of system calls so a thread can be represented as a word in an appropriate alphabet. Since HTTP is a connectionless protocol requests from the same client will end up going to different server threads over time, and intrusions are likely to be distributed over many threads.

Normal variations in usage for the server can be described by developing a thread taxonomy based on an investigation of words produced. A detector can be designed based on a probabilistic models for new word formation as in Forrest et. al. (1994) and Warrander et. al (1999).

Finally, a similar analysis can be used to address misuse leading to software corruption. Once a computer is successfully infiltrated and sufficient privelages are obtained by the attacker, one common form of malicious behavior is to modify a commonly used program and arrange for each subsequent legitimate to produce some undesirable result. The problem of detecting such code modifications can be addressed by modeling variations in the sequence of system calls obtained under normal usage conditions. Experiments yielding system call data for the gcc compiler have been carried out by Marchette (1999), and attempts to analyze these data will be described.

Statistical models that allow for normal usage to evolve will not be discussed here, but their introduction adds a fascinating new dimension to the problem.

3. False alarm rates via importance sampling

It is typically the case that specifying false alarm rates for anomaly-based intrusion detection systems is a difficult process. The detection event is usually comprised of a huge family of simple events, so one is confronted with a considerable multiple comparisons problem. Even computing a Bonferroni bound for the false detection probability can be an extremely challenging problem. When the Bonferroni bound is available there is good reason to be concerned about the possibility that it is far too conservative.

The problems fit naturally into a scan statistic framework described in Naiman and Priebe (2001) where a rather general importance sampling algorithm is presented for computing p-values. This algorithm gives, in a single iteration, an unbiased estimator for the correction factor to the Bonferroni bound, and the results of multiple independent iterations can be averaged in the usual way. A single iteration of this algorithm can be described as follows.

First, a detection event is generated randomly, where each event is given probability weight proportional to its probability. (Consequently, a minimal requirement for implementation of this method is the ability to calculate all of the contributions to the Bonferroni inequality.) Next, a realization of the data is generated conditionally on the particular detection event occurring. The desired unbiased estimator is then the reciprocal of the number of detection events that occur for the sampled realization.

In addition to guiding the construction of an algorithm to determine the true error rate, this result clarifies in a precise manner the source of conservatism in the Bonferroni inequality: the tendency for a single detection event to be associated with other detections as well. While this algorithm is simple to describe, the practicality of its implementation relies heavily upon the assumption that the second step, in which data are sampled conditionally, can be carried out in an efficient manner. In simple situations, it has been found to perform especially well in comparison with naive Monte Carlo sampling when the probability of detection is small.

In the remainder of this talk, efforts to implement this importance sampling algorithm in more complex situations coming from the computer intrusion detection problems above will be described. Some additional work that has been done related to the importance sampling algorithm will be described, and some directions for future methodological improvements will be outlined.

REFERENCES

S. Forrest, A.S. Perelson, L. Allen, R. and Cherukuri. (1994). "Self-Nonself Discrimination in a Computer." *Proceedings of the 1994 IEEE Symposium on Research in Security and Privacy.*

Marchette, D. (1999). Private communication.

Naiman, D.Q. and Priebe, C.E. (2001). "Computing Scan Statistic p-Values using Importance Sampling with Applications to Genetics and Medical Image Analysis". *Journal of Computational and Graphical Statistics, to appear.*

Warrander, C., Forrest, S., Pearlmutter, B. (1999) "Detecting intrusions using system calls: Alternative data models." *1999 IEEE Symposium on security and Privacy.*